

Muthaiah Ramanathan

021456145

CS 267 – Lab2

This homework task is accomplished using Cloudera VM. This sandbox is equipped with Hadoop Map Reduce environment.

IDE used – Eclipse

JDK 1.8

Jfreechart library to plot the graph

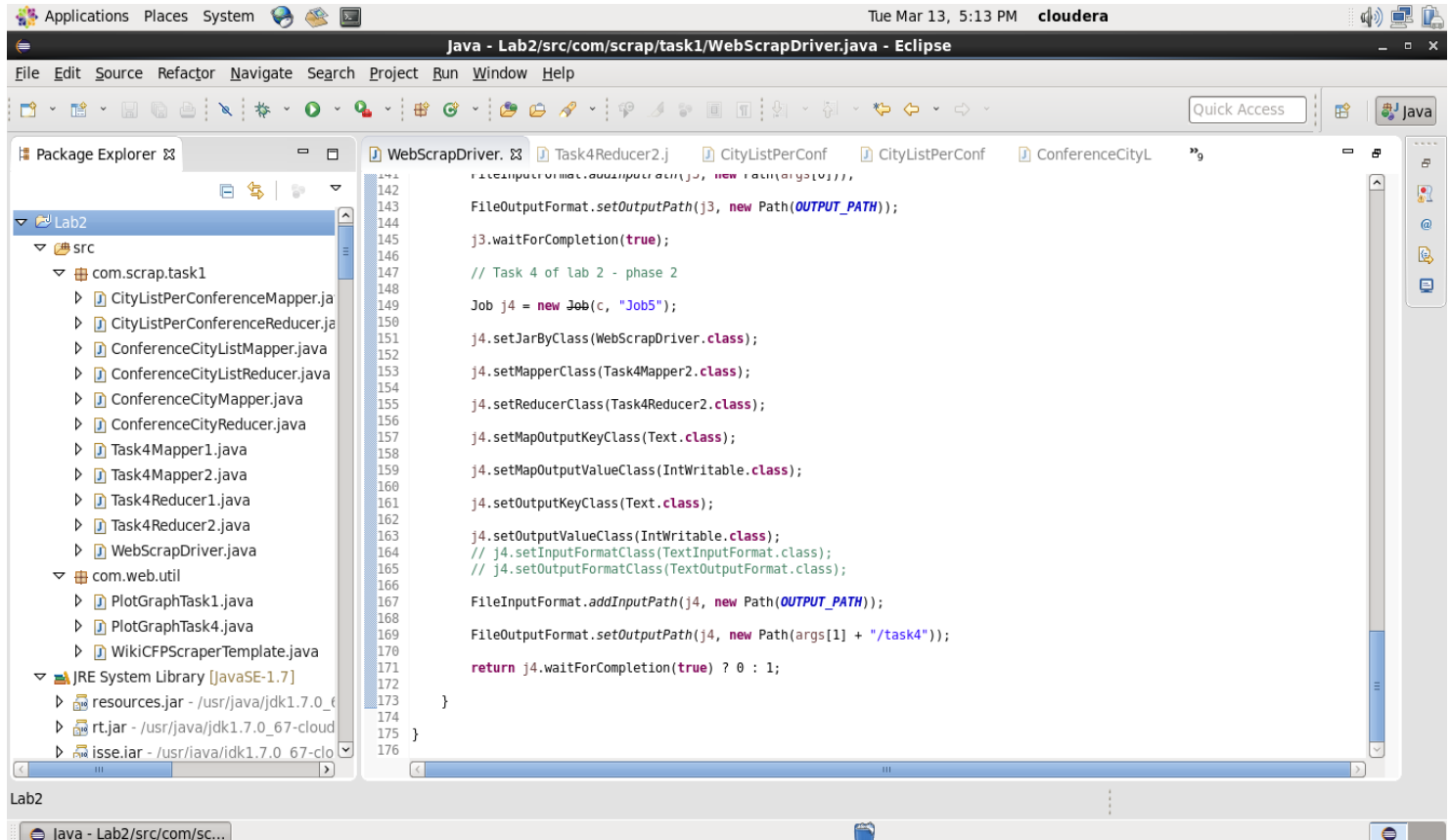


Fig 1: Project structure

### Commands:

#### To get the top 10 records

```
cat part-r-000001| awk '{FS=" ";$0=$0; print $NF"|"$0}'| sort -n -r|cut -d'|' -f2
```

#### To execute the Hadoop Program

```
hadoop jar "/home/cloudera/Desktop/Lab_2/Lab.jar" com.scrap.task1.Task1Driver /user/cloudera/lab2/input /user/cloudera/lab2/output1
```

### Part 1 - Crawler

Web crawler is a program that acts as an automated script which browses through the internet in a systematic way. The web crawler looks at the keywords in the pages, the kind of content each page has and the links, before returning the information to the search engine. This process is known as Web crawling. A web crawler gathers pages from the web and then, indexes them in a methodical and automated manner to support search engine queries.

**Description:**

Initially, as given in the Homework page, I tried to install the Apache Nutch tool on Cloudera's Virtual Machine used for Lab1. But, there was environment issue in setting up the config file. So, I developed a web crawler is developed and customized to crawl the Wificfp website and fetch results for categories,

1. Bigdata
2. Datamining
3. Databases
4. Artificial Intelligence

**Procedure:**

The Web Crawler is implemented to crawl the webpage of the given URI and return the HTML content. From the HTML content, the Java code will parse the tags and retrieve the appropriate contents.

**Part 2- Data Cleaning**

The data from the crawler will be with some noise and unwanted contents. To cleanse the data, we use a tool called Open Refine.

**Open Refine**

OpenRefine (formerly Google Refine) is a powerful tool for working with messy data: cleaning it; transforming it from one format into another; and extending it with web services and external data. OpenRefine can help explore large data sets with ease. OpenRefine can be used to link and extend your dataset with various webservices. Some services also allow OpenRefine to upload your cleaned data to a central database.

**Description**

1. The data has location as NA for conferences. Those rows are removed.
2. There are data that has location detail in one of the below formats.

Format 1: city, state, country

Format 2: city

Format 3: country

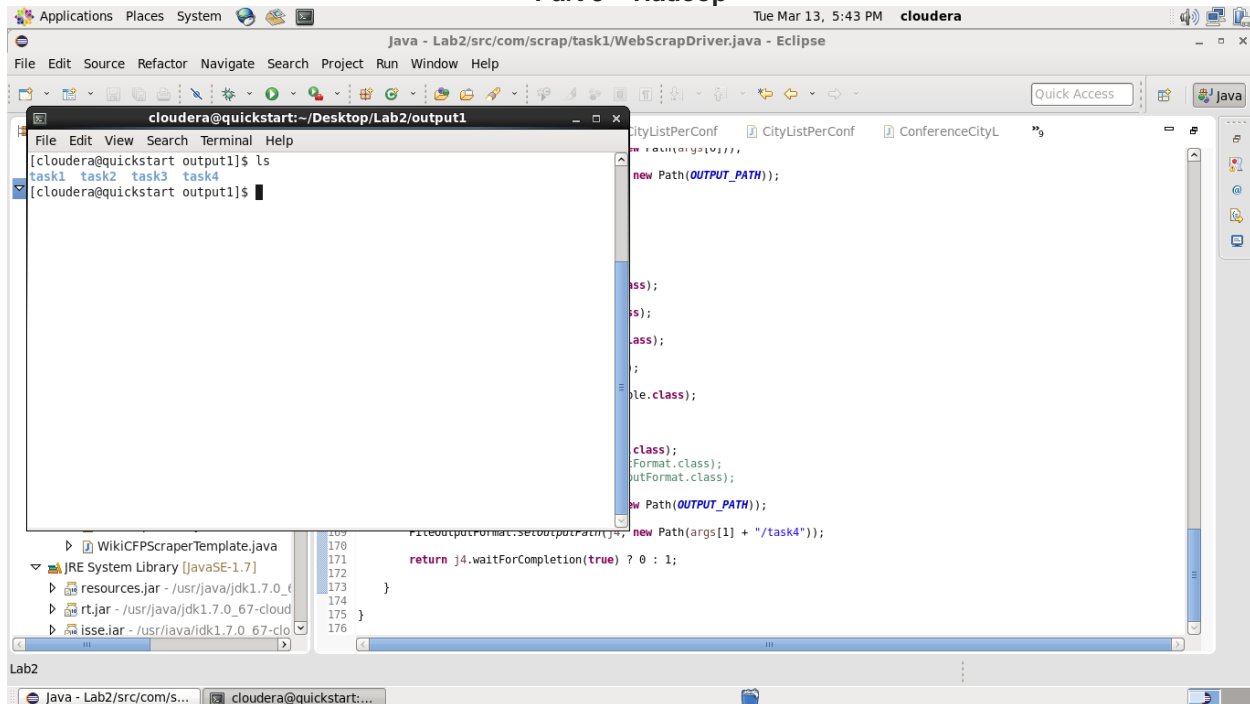
Format 4: city, state

Format 5: city, country

If location has more than one field (Format 1,4,5), the first field is considered as city. Nevertheless, if the location has just one data (say either city or country), we do not have a dictionary to differentiate if the field is a city or a country. So, we ignore those data as well using open refine.

3. The first column, the acronym of the conference contains the year as well. This column is split and made into two columns one containing the abbreviation of the conference and the other containing the year of the conference.

## Part 3 – Hadoop



The screenshot shows the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations and development tools. The main editor area displays a Java file named `CityListPerConf` with the following code:

```
new Path(OUTPUT_PATH);

// ...

new Path(OUTPUT_PATH));

// ...

return j4.waitForCompletion(true) ? 0 : 1;
```

The terminal window at the bottom shows the following commands and output:

```
[cloudera@quickstart output1]$ ls
task1 task2 task3 task4
[cloudera@quickstart output1]$
```

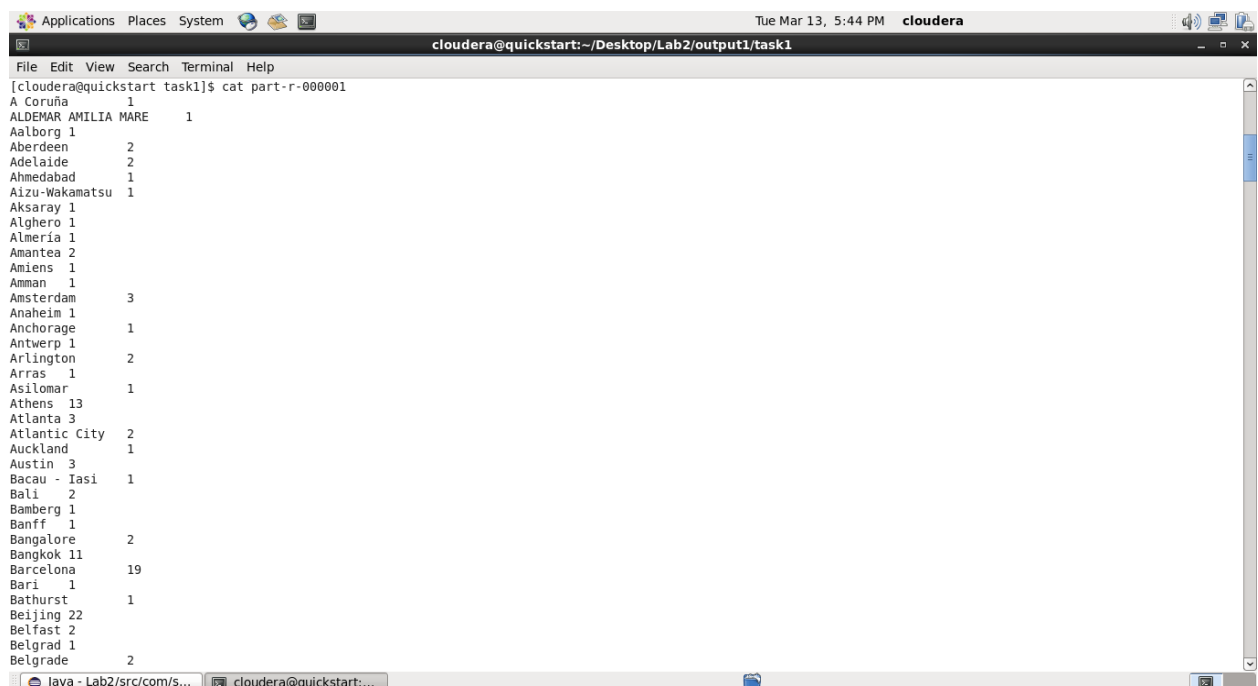
1. Compute and plot the number of conferences per city. Which are the top 10 locations?

### Map phase:

We parse the TSV file containing the data. We split and get the city name and write it as the key. The value is written as 1, meaning the count.

### Reduce Phase:

For every city, count the number of occurrences from the output of map.



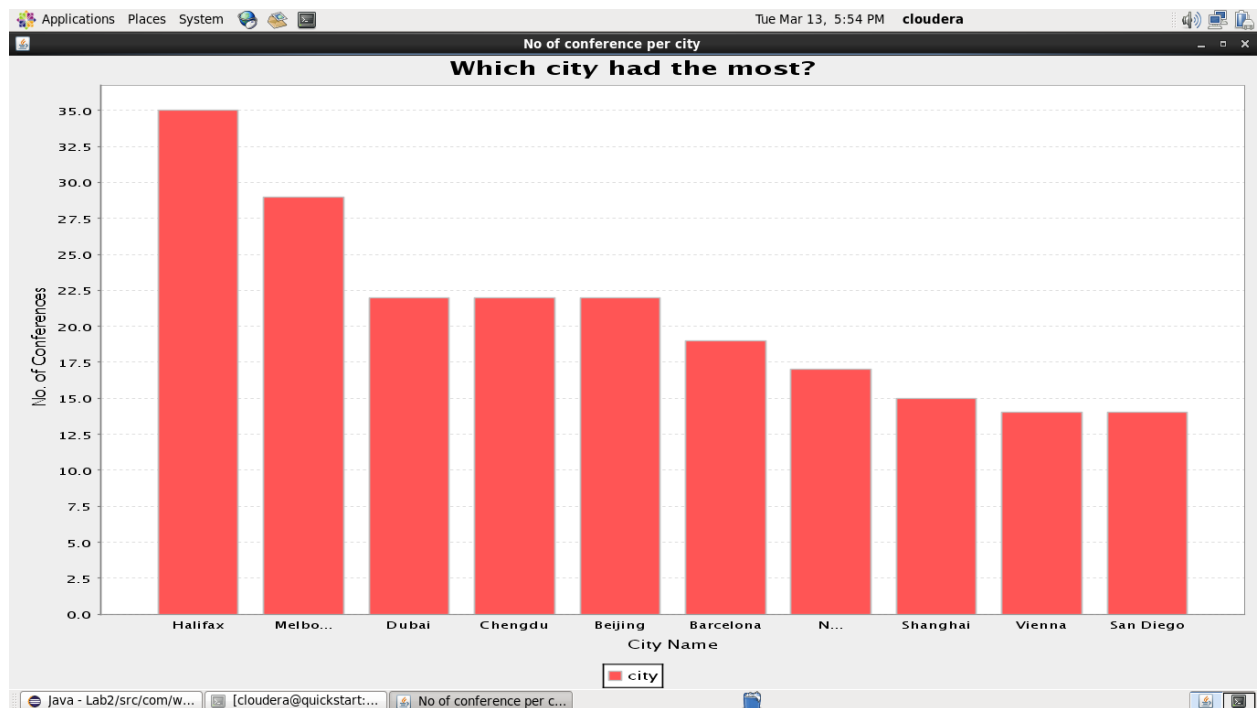
The screenshot shows a terminal window with the following output:

```
cloudera@quickstart:~/Desktop/Lab2/output1/task1
[cloudera@quickstart task1]$ cat part-r-000001
A Coruña 1
ALDEMAR AMILIA MARE 1
Aalborg 1
Aberdeen 2
Adelaide 2
Ahmedabad 1
Aizu-Wakamatsu 1
Aksaray 1
Alghero 1
Almeria 1
Amantea 2
Amiens 1
Amman 1
Amsterdam 3
Anaheim 1
Anchorage 1
Antwerp 1
Arlington 2
Arras 1
Asilomar 1
Athens 13
Atlanta 3
Atlantic City 2
Auckland 1
Austin 3
Bacau - Iasi 1
Bali 2
Bamberg 1
Banff 1
Bangalore 2
Bangkok 11
Barcelona 19
Bari 1
Bathurst 1
Beijing 22
Belfast 2
Belgrad 1
Belgrade 2
```

## Top 10 locations:

```
Applications Places System Tue Mar 13, 5:52 PM cloudera
cloudera@quickstart:~/Desktop/Lab2/output/task1
File Edit View Search Terminal Help
[cloudera@quickstart task1]$ cat part-r-000001 | awk '{FS=" ";$0=$0; print $NF|" "$0}' | sort -n -r | cut -d'|' -f2 | head -n10
Halifax 35
Melbourne 29
Dubai 22
Chengdu 22
Beijing 22
Barcelona 19
New Orleans 17
Shanghai 15
Vienna 14
San Diego 14
[cloudera@quickstart task1]$
```

## Graph plot:



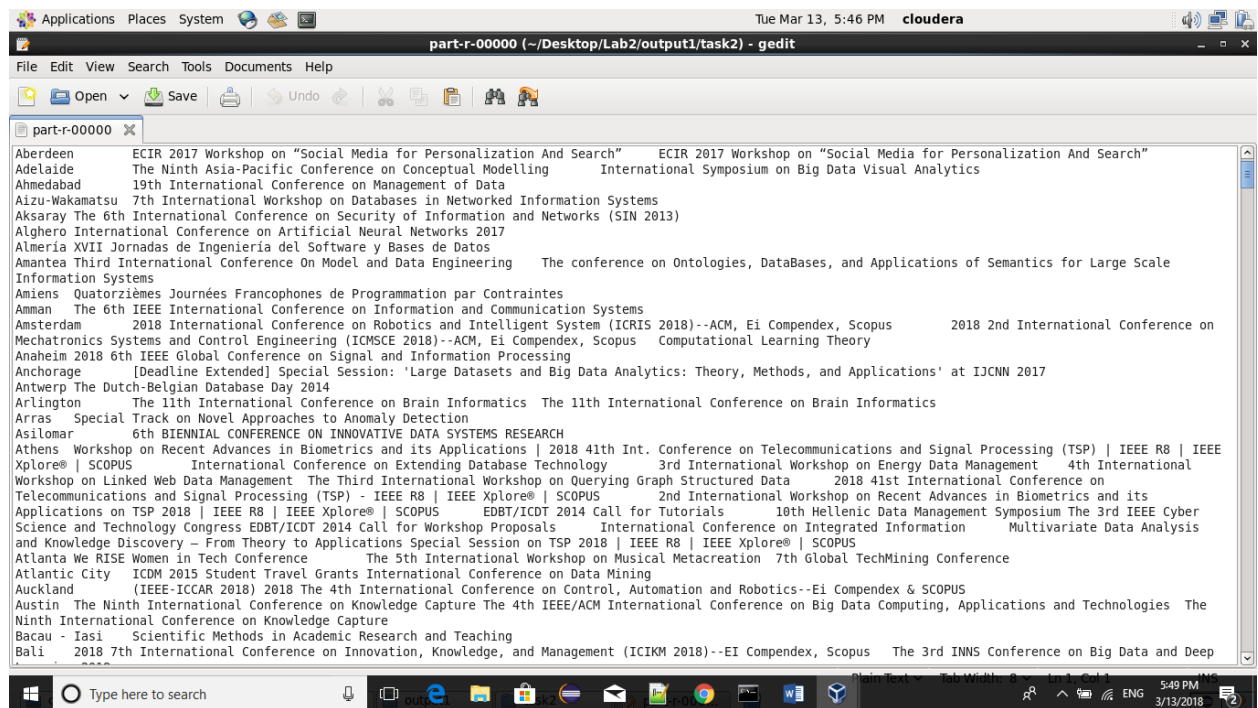
2. Output the list of conferences per city

## Map phase:

We parse the TSV file containing the data. We split and get the city name and write it as the key. The value is written as conference name.

## Reduce Phase:

For every city, we concatenate the conference names as a list from the output of map.



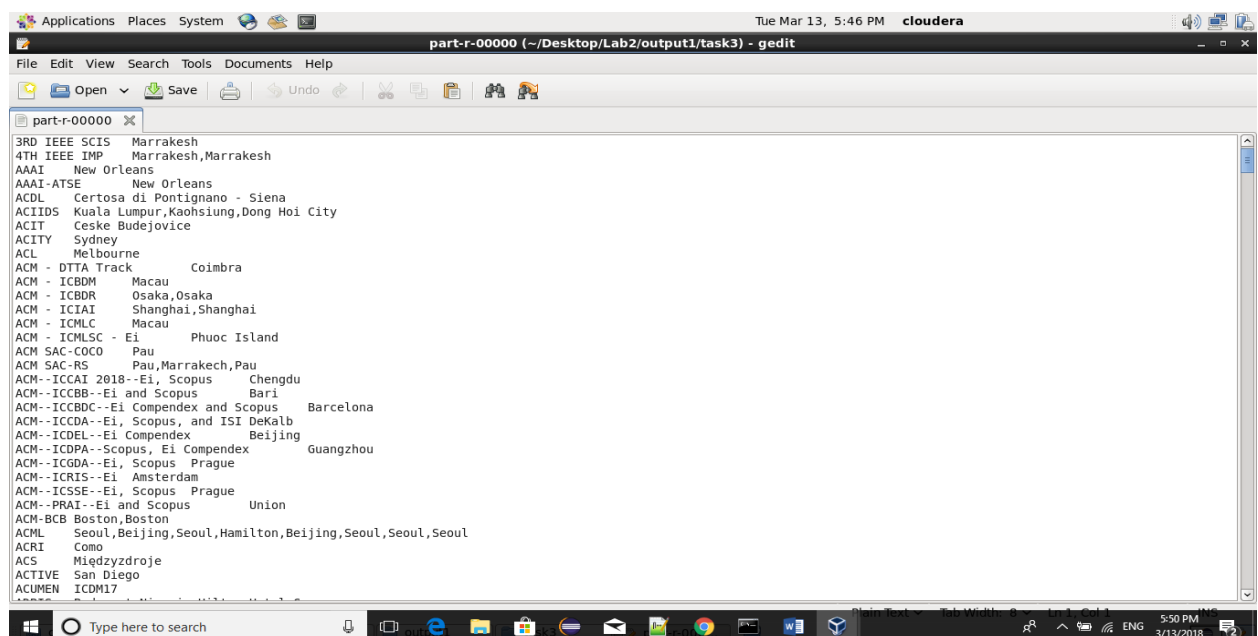
3. For each conference regardless of the year (e.g., KDD), output the list of cities.

## Map phase:

We parse the TSV file containing the data. We split the conference name without considering the year in it and write it as the key. The value is written as city name.

## Reduce Phase:

For every conference, we concatenate the city names separated by a comma as a list from the output of map.



4. For each city compute and plot a time series of #conferences per year.

This task has 2 parts. I learnt “Chaining map reduce jobs”. So, for this task, the flow is

**[Input] Map1 → Reduce1 → Map2 → Reduce2 [Output]**

We have three inputs here. The city, the year and the name of the conference.

### Map Phase 1:

We split and take the city from the TSV file input line and write this as a key. For the value, we split and take the conference name and concatenate with the year from the TSV input file. This is written as an intermediate output in the Map phase.

### Reduce Phase 1:

In the reduce phase, for every city as key in the map output, we concatenate the corresponding values and write as <city, list of conferences+year> as output of the reducer.

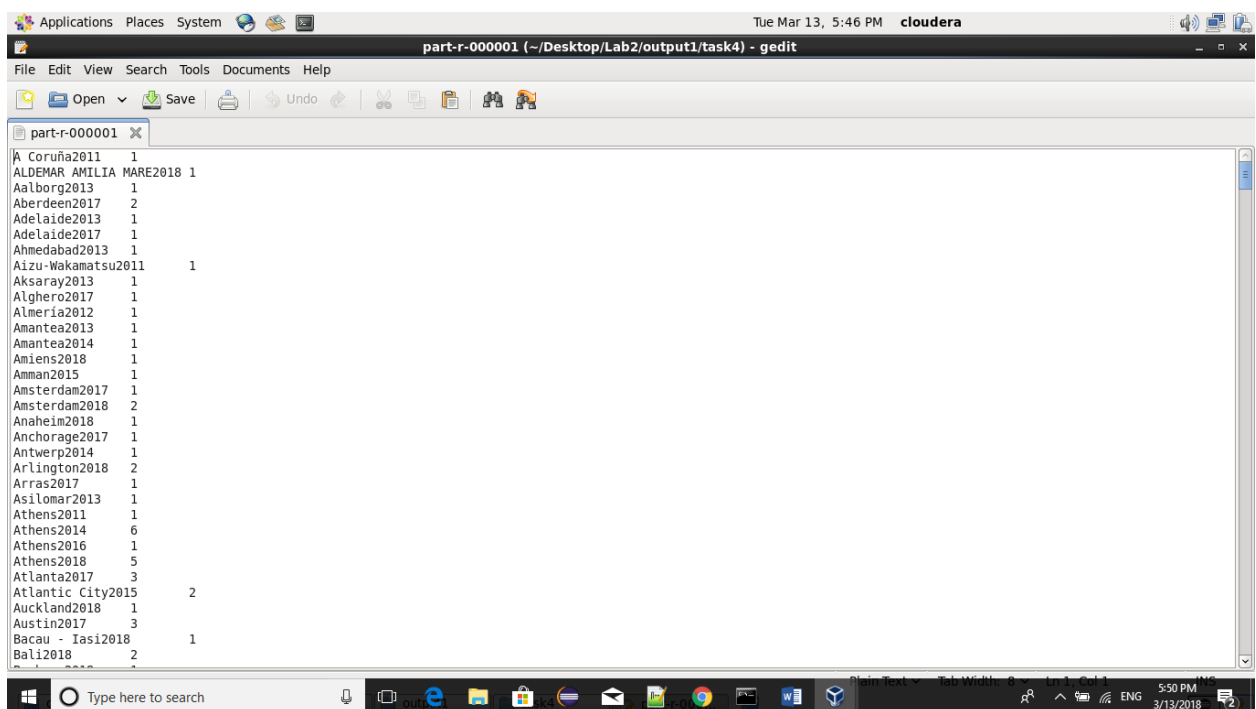
### Map Phase 2:

From the output of the previous reduce phase, we read a line from the file (The file is kept in the HDFS itself). For every line, we fetch the city and concatenate with the year (substring from the conference field (we concatenated in first map phase)) and written as key and value is written as 1, meaning the count.

### Reduce Phase 2:

In this phase, for each conference, every year, we add up all the 1s and write as the value. This output contains the answer for this task.

Now this output file is brought to the local file system from the HDFS and given as input for the graph plotting phase.



The screenshot shows a gedit window titled "part-r-000001 (~/.Desktop/Lab2/output1/task4) - gedit". The window displays a TSV file with the following content:

City	Year	Count
A Coruña	2011	1
ALDEMAR	AMILIA MARE	2018 1
Aalborg	2013	1
Aberdeen	2017	2
Adelaide	2013	1
Adelaide	2017	1
Ahmedabad	2013	1
Aizu-Wakamatsu	2011	1
Aksaray	2013	1
Alghero	2017	1
Almeria	2012	1
Amantea	2013	1
Amantea	2014	1
Amiens	2018	1
Amman	2015	1
Amsterdam	2017	1
Amsterdam	2018	2
Anaheim	2018	1
Anchorage	2017	1
Antwerp	2014	1
Arlington	2018	2
Arras	2017	1
Asilomar	2013	1
Athens	2011	1
Athens	2014	6
Athens	2016	1
Athens	2018	5
Atlanta	2017	3
Atlantic City	2015	2
Auckland	2018	1
Austin	2017	3
Bacau - Iasi	2018	1
Bali	2018	2

Applications Places System Tue Mar 13, 5:55 PM cloudera

### No of conference per city per year

## Which city had the most?

Year	AMIENS	ABERDEEN	ADELIADE	ALGHERO	AMANTEA	AMMAN	AMSTERDAM	ANAHEIM	ANCHORAGE	ANTWERP	ARLINGTON	ARRAS	ATHENS	ATLANTA	ATLANTIC CITY	AUCKLAND	AUSTIN
2014	6	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
2015	0	0	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0
2016	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
2017	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2018	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

java - Lab2/src/com/w... cloudera@quickstart:... No of conference per c...

### Graph Plotting phase

JFreechart is a library open sourced to plot the graphs. Downloaded the latest version of JFreeChart.zip from the link

<http://www.jfree.org/jfreechart/download/>

The dataset for the plot must be prepared for x axis and y axis.

When the `plot()` method is called the graph gets plotted as an applet.

## Lessons learnt

1. Apache Nutch did not work as expected as the config file set up was difficult on cloudera machine.
2. Initially, I had developed multiple driver classes in the same project, one for each task. I learn the concepts of “Chaining Map reduce jobs” and making the map reduce job dependent on another job. Now I just have only one driver class and when trigger the command to start Job J1, all other Jobs J2 till J5 are executed in a sequence.
3. Learnt to use JFreeChart for plotting the graph.



```

1
2
3 package com.web.util;
4
5 import java.io.BufferedReader;
6 import java.io.BufferedWriter;
7 import java.io.File;
8 import java.io.FileOutputStream;
9 import java.io.FileWriter;
10 import java.io.IOException;
11 import java.io.InputStreamReader;
12 import java.io.OutputStreamWriter;
13 import java.io.Writer;
14 import java.net.URL;
15 import java.net.URLConnection;
16 import java.net.URLEncoder;
17 import java.util.ArrayList;
18 import java.util.List;
19
20 /**
21  * @author Muthaiah
22  * WikiCFPScraperTemplate - Class that encapsulates the functionality to crawl
23  * wikicfp web pages
24  */
25 public class WikiCFPScraperTemplate {
26     public static int DELAY = 7; // Delay while crawling the wikicfp web pages
27
28     /**
29      * crawl() - to crawl all the wikicfp web pages for the categories - data
30      * mining, data bases, machine learning and artificial intelligence crawl()
31      * - crawls the web page from page # 1 to page # 20 for each category The
32      * crawled results are written to the file as [Conference Acronym |
33      * Conference Name | Location]
34      */
35     public static void crawl() {
36         try {
37             String[] uriCategory = { "data mining", "databases",
38                                     "machine learning", "artificial intelligence" };
39             int numOfPages = 20;
40             int firstPage = 1;
41             // File to store the output of the crawl
42             String fileName = "wikicfp_crawl_" + ".txt";
43             Writer writer = new BufferedWriter(new OutputStreamWriter(
44                 new FileOutputStream(fileName), "UTF-8"));
45             // File to store if any error occured
46             String checkName = "check.txt";
47             File file = new File(checkName);
48             file.createNewFile();
49             BufferedWriter writer2 = new BufferedWriter(new FileWriter(file));
50             writer2.write("collect missing data...");
51             writer2.newLine();
52             for (int k = 0; k < uriCategory.length; k++) {
53                 // now start crawling the all 'numOfPages' pages from
54                 // 'firstPage'
55                 for (int i = firstPage; i <= numOfPages; i++) {
56                     // Create the request to read the page
57                     // and get the number of total results
58                     String linkToScrape = "
59                     http://www.wikicfp.com/cfp/call?conference="
60                     + URLEncoder.encode(uriCategory[k], "UTF-8")
61                     + "&page=" + i;
62                     String content = getPageFromUrl(linkToScrape);
63                     // parse or store the content of page 'i' here in 'content'
64                     ArrayList<List<List<String>>> out = read(i, content);
65                     for (int il = 0; il < out.get(0).size(); il++) {
66                         for (int i2 = 0; i2 < out.get(0).get(il).size(); i2++) {
67                             String tmp = out.get(0).get(il).get(i2);
68                             writer.write(tmp);
69
70                             if (i2 < out.get(0).get(il).size() - 1) {
71                                 writer.write("\t");
72                             }
73                         }
74                     }
75                 }
76             }
77         }
78     }
79 }

```

```

72         writer.write("\n");
73
74     }
75
76     // If data miss occurs
77     if (out.size() == 2) {
78         for (int i11 = 0; i11 < out.get(1).size(); i11++) {
79             for (int i22 = 0; i22 < out.get(1).get(i11).size();
80                 i22++) {
81                 String tmp2 = out.get(1).get(i11).get(i22);
82                 writer2.write(tmp2);
83
84                 if (i22 < out.get(1).get(i11).size() - 1) {
85                     writer2.write(" ");
86                 }
87                 writer2.newLine();
88
89             }
90         }
91
92         Thread.sleep(DELAY * 1000); // Wikicfp courtesy policy for
93                                     // crawling
94     }
95
96     writer2.newLine();
97     writer2.write("Collecting missing data end.");
98
99 }
100 writer.close();
101 writer2.close();
102 } catch (IOException e) {
103     e.printStackTrace();
104 } catch (InterruptedException e) {
105     e.printStackTrace();
106 }
107 }
108
109 public static ArrayList<List<List<String>>> read(int page, String content) {
110     ArrayList<List<List<String>>> pack = new ArrayList<List<List<String>>>();
111
112     ArrayList<List<String>> interesting = new ArrayList<List<String>>(); //
113     crawl data
114     ArrayList<List<String>> empty = new ArrayList<List<String>>(); // empty data
115
116     // Select the table
117     int ini = content.indexOf("table cellpadding=\"3\"");
118     int end = content.indexOf("/table", ini);
119     int i = 1; // item
120
121     // crawl the table (20 items per page)
122     while (true) {
123         List<String> element = new ArrayList<String>();
124         List<String> emptyOne = new ArrayList<String>();
125
126         // Use URL link to get acronym
127         int pre = content.indexOf("a href=", ini);
128         // After the table or not find
129         if (pre >= end || pre == -1) {
130             break;
131         }
132         pre = content.indexOf(">", pre);
133         int post = content.indexOf("</", pre);
134         String acronym = content.substring(pre + 1, post).trim();
135
136         element.add(acronym);
137         // Check acronym
138         String tmp1 = page + "_" + i + "_acronym";
139         if (acronym.equals("")) {
140             emptyOne.add(tmp1);
141         }
142         // Get name

```

```

143     pre = content.indexOf("<td align=\"left\"", post);
144     pre = content.indexOf(">", pre);
145     post = content.indexOf("</", pre);
146     String name = content.substring(pre + 1, post).trim();
147     element.add(name);
148
149     // Check name
150     String tmp2 = page + "_" + i + "_name";
151     if (name.equals("")) {
152         emptyOne.add(tmp2);
153     }
154
155     // get location
156     pre = content.indexOf("<td align=\"left\"", post);
157     pre = content.indexOf("<td align=\"left\"", pre + 1);
158     pre = content.indexOf(">", pre);
159     post = content.indexOf("</", pre);
160     String location = content.substring(pre + 1, post).trim();
161     element.add(location);
162
163     // Check location
164     String tmp3 = page + "_" + i + "_location";
165     if (location.equals("")) {
166         emptyOne.add(tmp3);
167     }
168
169     // If location is 'N/A', it is a journal
170     if (!location.equals("N/A")) {
171         interesting.add(element);
172         if (!emptyOne.isEmpty()) {
173             empty.add(emptyOne);
174         }
175     }
176
177     ini = post;
178     i++;
179 }
180
181 pack.add(interesting);
182 if (!empty.isEmpty()) {
183     pack.add(empty);
184 }
185
186 return pack;
187 }
188
189 /**
190  * Given a string URL returns a string with the page contents Adapted from
191  * example in
192  * http://docs.oracle.com/javase/tutorial/networking/urls/readingWriting
193  * .html
194  *
195  * @param link
196  * @return
197  * @throws IOException
198  */
199 public static String getPageFromUrl(String link) throws IOException {
200     URL thePage = new URL(link);
201     URLConnection yc = thePage.openConnection();
202     // Change encoding to 'UTF-8'
203     BufferedReader in = new BufferedReader(new InputStreamReader(
204         yc.getInputStream(), "UTF-8"));
205     String inputLine;
206     String output = "";
207     while ((inputLine = in.readLine()) != null) {
208         output += inputLine + "\n";
209     }
210     in.close();
211     return output;
212 }
213
214 }

```

```

package com.scrap.task1;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
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.util.GenericOptionsParser;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

import com.web.util.WikiCFPScraperTemplate;

/**
 * @author Muthaiah WebScrapDriver - A driver class that creates and executes
 *         Hadoop Jobs for task 1, 2, 3, 4 of Lab2
 */
@SuppressWarnings("unused")
public class WebScrapDriver extends Configured implements Tool {

    private static final String OUTPUT_PATH = "intermediate_output";

    /**
     * @param args
     * @throws Exception
     * Driver method - entry point for the project
     */
    public static void main(String[] args) throws Exception

    {
        // ToolRunner.run(new Configuration(), new Task1Driver(), args);
        WikiCFPScraperTemplate.crawl();
    }

    /**
     * (non-Javadoc)
     *
     * @see org.apache.hadoop.util.Tool#run(java.lang.String[])
     */
    @SuppressWarnings("deprecation")
    @Override
    public int run(String[] args) throws Exception {
        // Task 1 of Lab 2

        Configuration c = new Configuration();
        Job j = new Job(c, "Job1");

        j.setJarByClass(WebScrapDriver.class); // driver class loading

        j.setMapperClass(ConferenceCityMapper.class); //mapper class for the Job
        j.setReducerClass(ConferenceCityReducer.class); //reducer class for the job
        j.setMapOutputKeyClass(Text.class); //ouput key format of the mapper
        j.setMapOutputValueClass(IntWritable.class); //output value format of the
        mapper
        j.setOutputKeyClass(Text.class); //ouput key format of the reducer
        j.setOutputValueClass(IntWritable.class); //output value format of the reducer
    }
}

```

```
288     FileInputFormat.addInputPath(j, new Path(args[0])); // Input path of the tsv
289     file from command line arguments
290
291     FileOutputFormat.setOutputPath(j, new Path(args[1] + "/task1")); //Output
292     path in HDFS file system
293
294     j.waitForCompletion(true); // Not to start next job until this job is
295     completed
296     // Task 2 of Lab 2
297
298     Job j1 = new Job(c, "Job2");
299
300     j1.setJarByClass(WebScrapDriver.class);
301
302     j1.setMapperClass(ConferenceCityListMapper.class);
303
304     j1.setReducerClass(ConferenceCityListReducer.class);
305
306     j1.setMapOutputKeyClass(Text.class);
307
308     j1.setMapOutputValueClass(Text.class);
309
310     j1.setOutputKeyClass(Text.class);
311
312     j1.setOutputValueClass(Text.class);
313
314     FileInputFormat.addInputPath(j1, new Path(args[0]));
315
316     FileOutputFormat.setOutputPath(j1, new Path(args[1] + "/task2"));
317
318     j1.waitForCompletion(true);
319     // Task 3 of Lab 2
320
321     Job j2 = new Job(c, "Job3");
322
323     j2.setJarByClass(WebScrapDriver.class);
324
325     j2.setMapperClass(CityListPerConferenceMapper.class);
326
327     j2.setReducerClass(CityListPerConferenceReducer.class);
328
329     j2.setMapOutputKeyClass(Text.class);
330
331     j2.setMapOutputValueClass(Text.class);
332
333     j2.setOutputKeyClass(Text.class);
334
335     j2.setOutputValueClass(Text.class);
336
337     FileInputFormat.addInputPath(j2, new Path(args[0]));
338
339     FileOutputFormat.setOutputPath(j2, new Path(args[1] + "/task3"));
340
341     j2.waitForCompletion(true);
342
343     // Task 4 of Lab 2- Phase 1
344
345     Job j3 = new Job(c, "Job4");
346
347     j3.setJarByClass(WebScrapDriver.class);
348
349     j3.setMapperClass(Task4Mapper1.class);
350
351     j3.setReducerClass(Task4Reducer1.class);
352
353     j3.setMapOutputKeyClass(Text.class);
354
355     j3.setMapOutputValueClass(Text.class);
356
357     j3.setOutputKeyClass(Text.class);
358
359     j3.setOutputValueClass(Text.class);
360
361     // j3.setInputFormatClass(TextInputFormat.class);
```

```

358         // j3.setOutputFormatClass(TextOutputFormat.class);
359
360         FileInputFormat.addInputPath(j3, new Path(args[0]));
361
362         FileOutputFormat.setOutputPath(j3, new Path(OUTPUT_PATH));
363
364         j3.waitForCompletion(true);
365
366         // Task 4 of lab 2 - phase 2
367
368         Job j4 = new Job(c, "Job5");
369
370         j4.setJarByClass(WebScrapDriver.class);
371
372         j4.setMapperClass(Task4Mapper2.class);
373
374         j4.setReducerClass(Task4Reducer2.class);
375
376         j4.setMapOutputKeyClass(Text.class);
377
378         j4.setMapOutputValueClass(IntWritable.class);
379
380         j4.setOutputKeyClass(Text.class);
381
382         j4.setOutputValueClass(IntWritable.class);
383         // j4.setInputFormatClass(TextInputFormat.class);
384         // j4.setOutputFormatClass(TextOutputFormat.class);
385
386         FileInputFormat.addInputPath(j4, new Path(OUTPUT_PATH));
387
388         FileOutputFormat.setOutputPath(j4, new Path(args[1] + "/task4"));
389
390         return j4.waitForCompletion(true) ? 0 : 1;
391     }
392 }
393
394 }

```

Code for Mapper - Task 1

```

400 package com.scrap.task1;
401
402 import java.io.IOException;
403
404 import org.apache.hadoop.io.IntWritable;
405 import org.apache.hadoop.io.LongWritable;
406 import org.apache.hadoop.io.Text;
407 import org.apache.hadoop.mapreduce.Mapper;
408
409 //Task1
410 /**
411  * @author Muthaiah
412  * ConferenceCityMapper - Class that encapsulates the Mapper function for Task 1 of
413  * Lab2
414  */
415 public class ConferenceCityMapper extends
416     Mapper<LongWritable, Text, Text, IntWritable> {
417
418     /* (non-Javadoc)
419     * @see org.apache.hadoop.mapreduce.Mapper#map(KEYIN, VALUEIN,
420     org.apache.hadoop.mapreduce.Mapper.Context)
421     */
422     public void map(LongWritable key, Text value, Context con)
423         throws IOException, InterruptedException {
424
425         String line = value.toString(); // A line in the input file
426         String city = line.split("\t")[3]; // Split the line with tab spaces & the
427         4th field - city
428
429         Text outputKey = new Text(city.split(",")[0].trim()); // The 0th field - the
430         conference acronym

```

```

427         // Write the output of the mapper < Conference , City>
428         con.write(outputKey, new IntWritable(1)); // the count 1
429     }
430 }
431
432 }

```

Code for Reducer - Task 1

```

433
434 package com.scrap.task1;
435
436 import java.io.IOException;
437
438 import org.apache.hadoop.io.IntWritable;
439 import org.apache.hadoop.io.Text;
440 import org.apache.hadoop.mapreduce.Reducer;
441 //task1
442 /**
443  * @author Muthaiah
444  * ConferenceCityReducer - Class that encapsulates the Reducer function for Task 1
445  * of Lab2
446  */
447 public class ConferenceCityReducer extends
448     Reducer<Text, IntWritable, Text, IntWritable> {
449
450     /* (non-Javadoc)
451      * @see org.apache.hadoop.mapreduce.Reducer#reduce(KEYIN, java.lang.Iterable,
452      * org.apache.hadoop.mapreduce.Reducer.Context)
453      */
454     public void reduce(Text city, Iterable<IntWritable> values, Context con)
455         throws IOException, InterruptedException {
456
457         int cityCount = 0;
458
459         for (IntWritable value : values) {
460
461             cityCount += value.get(); // Sum all the 1s for every city
462         }
463         //Write the output of the reducer < City, # of conferences>
464         con.write(city, new IntWritable(cityCount));
465     }
466 }
467
468 }

```

Code for Mapper - Task 2

```

469
470 package com.scrap.task1;
471
472 import java.io.IOException;
473
474 import org.apache.hadoop.io.LongWritable;
475 import org.apache.hadoop.io.Text;
476 import org.apache.hadoop.mapreduce.Mapper;
477 //task 2
478 /**
479  * @author Muthaiah
480  * ConferenceCityListMapper - Class that encapsulates the Mapper fucntion for Task 2
481  * of Lab2
482  */
483 public class ConferenceCityListMapper extends Mapper<LongWritable, Text, Text, Text>{
484
485     /* (non-Javadoc)
486      * @see org.apache.hadoop.mapreduce.Mapper#map(KEYIN, VALUEIN,
487      * org.apache.hadoop.mapreduce.Mapper.Context)
488      */
489     public void map(LongWritable key, Text value, Context con)
490         throws IOException, InterruptedException {
491
492         String line = value.toString(); // A line in the input tsv file
493         String conference[] = line.split("\t"); // Split the line by tab spaces to
494         get 4 fields

```

```

495         Text outputKey = new Text(conference[3].split(",")[0].trim()); // The 0th
496         field from the input file - city
497         //Write the output of the <city: conf title>
498         con.write(outputKey, new Text(conference[2].trim())); // The 2nd field from
499         the input file - Conference Title
500     }
501 }
502
503                                     Code for Reducer - Task 2
504
505 package com.scrap.task1;
506
507 import java.io.IOException;
508
509 import org.apache.hadoop.io.Text;
510 import org.apache.hadoop.mapreduce.Reducer;
511
512 //task 2
513 /**
514  * @author Muthaiah
515  * ConferenceCityListReducer - Class that encapsulates the Reducer function for Task
516  * 2 of Lab2
517  */
518 public class ConferenceCityListReducer extends Reducer<Text, Text, Text, Text> {
519     Text value = new Text();
520
521     /* (non-Javadoc)
522     * @see org.apache.hadoop.mapreduce.Reducer#reduce(KEYIN, java.lang.Iterable,
523     org.apache.hadoop.mapreduce.Reducer.Context)
524     */
525     public void reduce(Text city, Iterable<Text> values, Context con)
526         throws IOException, InterruptedException {
527         StringBuilder builder = new StringBuilder();
528         for (Text text : values) {
529             builder.append(text.toString());
530             builder.append("\t"); // Concatenate all conference names
531         }
532         value.set(builder.toString());
533         //Write the output of the reduce phase - <City, List of conference titles>
534         con.write(city, new Text(value));
535     }
536 }
537
538                                     Code for Mapper - Task 3
539
540 package com.scrap.task1;
541
542 import java.io.IOException;
543
544 import org.apache.hadoop.io.LongWritable;
545 import org.apache.hadoop.io.Text;
546 import org.apache.hadoop.mapreduce.Mapper;
547
548 /**
549  * @author Muthaiah
550  * CityListPerConferenceMapper - Class that encapsulates the Mapper function for
551  * Task 3 of Lab2
552  */
553 public class CityListPerConferenceMapper extends Mapper<LongWritable, Text, Text,
554 Text>{
555     /* (non-Javadoc)
556     * @see org.apache.hadoop.mapreduce.Mapper#map(KEYIN, VALUEIN,
557     org.apache.hadoop.mapreduce.Mapper.Context)
558     */
559     public void map(LongWritable key, Text value, Context con)
560         throws IOException, InterruptedException {

```



```

561 String line = value.toString(); // A line in the input file (tsv format)
562 String conferences[] = line.split("\t");
563 //Form the key for the output of map phase
564 Text outputKey = new Text(conferences[0].trim()); // The 0th column -
conference acronym
565 //Write intermediate output from mapper <key, value>
566 con.write(outputKey, new Text(conferences[3].split(",")[0].trim())); // The
3rd column, the city
567
568 }
569
570 }
571
572

```

#### Code for Reducer - Task 3

```

574 package com.scrap.task1;
575
576 import java.io.IOException;
577
578 import org.apache.hadoop.io.Text;
579 import org.apache.hadoop.mapreduce.Reducer;
580
581 //task 3
582 /**
583  * @author Muthaiah
584  * CityListPerConferenceReducer - Class that encapsulates the Reducer fucntion for
Task 3 of Lab2
585  */
586 public class CityListPerConferenceReducer extends
Reducer<Text, Text, Text, Text> {
587     Text value = new Text(); // A line from intermediate output of mapper
588
589     /*
590     * (non-Javadoc)
591     *
592     * @see org.apache.hadoop.mapreduce.Reducer#reduce(KEYIN,
593     * java.lang.Iterable, org.apache.hadoop.mapreduce.Reducer.Context)
594     */
595     public void reduce(Text conference, Iterable<Text> values, Context con)
throws IOException, InterruptedException {
596
597         StringBuilder builder = new StringBuilder();
598         for (Text text : values) {
599             builder.append(text.toString());
600             builder.append(","); // append a comma to values.
601         }
602         // delete the comma at the end of the string
603         builder.setLength(builder.length() - 1);
604         value.set(builder.toString());
605
606         // Write output of the reduce phase
607         con.write(conference, new Text(value));
608     }
609 }
610
611 }
612
613 }
614

```

#### Code for Mapper - Task 4 - Phase 1

```

617 package com.scrap.task1;
618
619 import java.io.IOException;
620
621 import org.apache.hadoop.io.Text;
622 import org.apache.hadoop.mapreduce.Mapper;
623
624 /**
625  * @author Muthaiah
626  * Task4Mapper1 - Class that encapsulates the Mapper function for Task 4 of Lab2
(Mapper phase 1)
627  */
628 public class Task4Mapper1 extends Mapper<Object, Text, Text, Text>{
629

```

```

630     /* (non-Javadoc)
631      * @see org.apache.hadoop.mapreduce.Mapper#map(KEYIN, VALUEIN,
        org.apache.hadoop.mapreduce.Mapper.Context)
632      */
633     public void map(Object key, Text value, Context con)
634         throws IOException, InterruptedException {
635         // A line from the map file
636         String line = value.toString();
637         // Split the line with a tab spaces
638         String conferences[] = line.split("\t");
639         //Output key for the intermediate map results - the city
640         Text outputKey = new Text(conferences[3].split(",")[0].trim());
641         //Write the output with configuration object
642         con.write(outputKey, new Text(conferences[0].trim().concat(conferences[1])));
643     }
644 }
645
646 }

```

Code for Reducer - Task 4 - Phase 1

```

650 package com.scrap.task1;
651
652 import java.io.IOException;
653
654 import org.apache.hadoop.io.Text;
655 import org.apache.hadoop.mapreduce.Reducer;
656
657 /**
658  * @author Muthaiah
659  * Task4Reducer1 - Class that encapsulates the Reducer function for Task 4 of Lab2
        (Reduce phase 1)
660  */
661 public class Task4Reducer1 extends Reducer<Text, Text, Text, Text> {
662     Text value = new Text();
663
664     /* (non-Javadoc)
665      * @see org.apache.hadoop.mapreduce.Reducer#reduce(KEYIN, java.lang.Iterable,
        org.apache.hadoop.mapreduce.Reducer.Context)
666      */
667     public void reduce(Text city, Iterable<Text> values, Context con)
668         throws IOException, InterruptedException {
669
670         StringBuilder builder = new StringBuilder();
671         for (Text text : values) {
672             builder.append(text.toString());
673             builder.append("#"); // Append with a # for use in the second map phase
674         }
675         value.set(builder.toString());
676         // Write the output with configuration object
677         con.write(city, new Text(value));
678     }
679 }
680
681 }

```

Code for Mapper - Task 4 -  
Phase 2

```

684 package com.scrap.task1;
685
686 import java.io.IOException;
687
688 import org.apache.hadoop.io.IntWritable;
689 import org.apache.hadoop.io.Text;
690 import org.apache.hadoop.mapreduce.Mapper;
691
692 /**
693  * @author Muthaiah
694  * Task4Mapper2 - Class that encapsulates the Mapper function for Task 4 of Lab2
        (Mapper phase 2)
695  */
696
697 public class Task4Mapper2 extends Mapper<Object, Text, Text, IntWritable> {

```

```

698
699     public void map(Object key, Text value, Context con)
700         throws IOException, InterruptedException {
701
702         String line = value.toString();
703         String input[] = line.split("\t");
704         String city = input[0];
705         String conferences[] = input[1].split("#"); //Split with the # added in
            reduce phase1
706         for (int i = 0; i < conferences.length; i++) {
707             String year = conferences[i].substring(conferences[i].length() - 4,
708                 conferences[i].length());
709             Text outputKey = new Text(city.concat(year)); // Concatenate city with
                year for graph plot
710
711             con.write(outputKey, new IntWritable(1)); // For every city,year
                combination, write 1
712         }
713
714     }
715
716 }
717

```

Code for Reducer - Task 4 - Phase 2

```

718
719 package com.scrap.task1;
720
721 import java.io.IOException;
722
723 import org.apache.hadoop.io.IntWritable;
724 import org.apache.hadoop.io.Text;
725 import org.apache.hadoop.mapreduce.Reducer;
726
727 /**
728  * @author Muthaiah
729  * Task4Reducer2 - Class that encapsulates the Reducer function for Task 4 of Lab2
            (Reduce phase 1)
730  */
731 public class Task4Reducer2 extends
            Reducer<Text, IntWritable, Text, IntWritable> {
732
733     /* (non-Javadoc)
734      * @see org.apache.hadoop.mapreduce.Reducer#reduce(KEYIN, java.lang.Iterable,
            org.apache.hadoop.mapreduce.Reducer.Context)
735      */
736     public void reduce(Text conference, Iterable<IntWritable> values,
737         Context con) throws IOException, InterruptedException {
738
739         int sum = 0;
740
741         for (IntWritable value : values) {
742
743             sum += value.get(); // sum all the 1s for city, year combination
744
745         }
746         //Write the final output with configuration object
747         con.write(conference, new IntWritable(sum));
748
749     }
750
751 }
752

```

Code for plotting graph - Task 1

```

753
754 package com.web.util;
755
756 import java.io.BufferedReader;
757 import java.io.File;
758 import java.io.FileReader;
759 import java.io.IOException;
760 import java.util.ArrayList;
761
762
763 import org.jfree.chart.ChartFactory;
764 import org.jfree.chart.ChartPanel;
765 import org.jfree.chart.JFreeChart;

```

```

766 import org.jfree.chart.plot.PlotOrientation;
767 import org.jfree.data.category.CategoryDataset;
768 import org.jfree.data.category.DefaultCategoryDataset;
769 import org.jfree.ui.ApplicationFrame;
770 import org.jfree.ui.RefineryUtilities;
771
772 /**
773  * @author Muthaiah
774  * PlotGraphTask1- This class encapsulates the functionality to plot graph for task1
775  */
776 @SuppressWarnings("serial")
777 public class PlotGraphTask1 extends ApplicationFrame {
778
779     /**
780      * @param applicationTitle
781      * @param chartTitle
782      * @throws NumberFormatException
783      * @throws IOException
784      */
785     public PlotGraphTask1(String applicationTitle, String chartTitle) throws
786         NumberFormatException, IOException {
787         super(applicationTitle);
788         //Bar chart for plot city vs number of conferences
789         JFreeChart barChart = ChartFactory.createBarChart(chartTitle, "City Name",
790             "No. of Conferences", createDataset(),
791             PlotOrientation.VERTICAL, true, true, false);
792
793         ChartPanel chartPanel = new ChartPanel(barChart);
794         chartPanel.setPreferredSize(new java.awt.Dimension(560, 367));
795         setContentPane(chartPanel);
796     }
797
798     /**
799      * @return
800      * @throws NumberFormatException
801      * @throws IOException
802      */
803     private CategoryDataset createDataset() throws NumberFormatException,
804         IOException {
805         ArrayList<String> cityList = new ArrayList<String>(); // to hold the cities
806         ArrayList<Integer> countList = new ArrayList<Integer>(); // to hold the
807             count of conferences
808         File file = new
809             File("/home/cloudera/Desktop/Lab2/output1/task1/part-r-000001"); //file
810             location
811
812         BufferedReader br = new BufferedReader(new FileReader(file));
813
814         String line;
815         int count=0;
816         while ((line = br.readLine()) != null && count <10){
817             cityList.add(line.split("\t")[0]);
818             countList.add(Integer.valueOf(line.split("\t")[1]));
819             count++;
820         }
821         br.close();
822         final String city = "city";
823
824         final DefaultCategoryDataset dataset = new DefaultCategoryDataset();
825         // Add all the cities and count to the dataset used for plotting graph
826         for(int i=0;i<cityList.size()&& i< countList.size();i++){
827             dataset.addValue(countList.get(i), city, cityList.get(i));
828         }
829
830         return dataset;
831     }
832
833     /**
834      * @param args
835      * @throws NumberFormatException
836      * @throws IOException
837      * Driver function to call the plot functionality
838      */

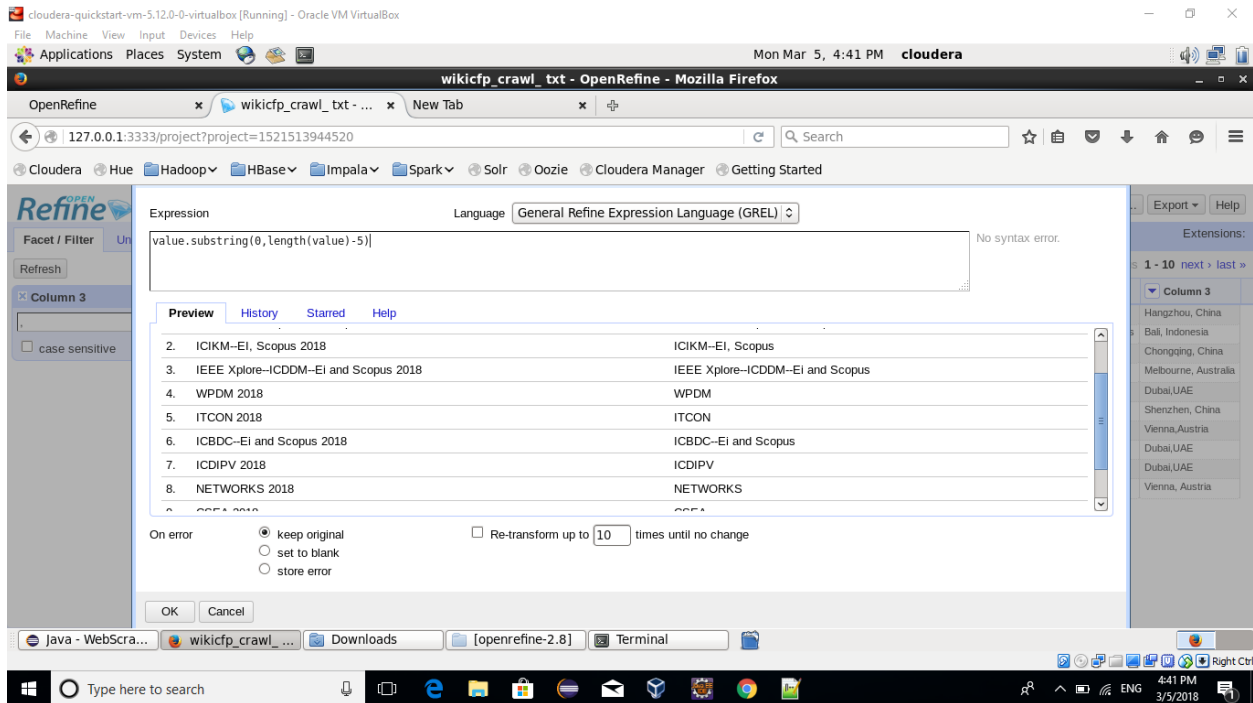
```

```

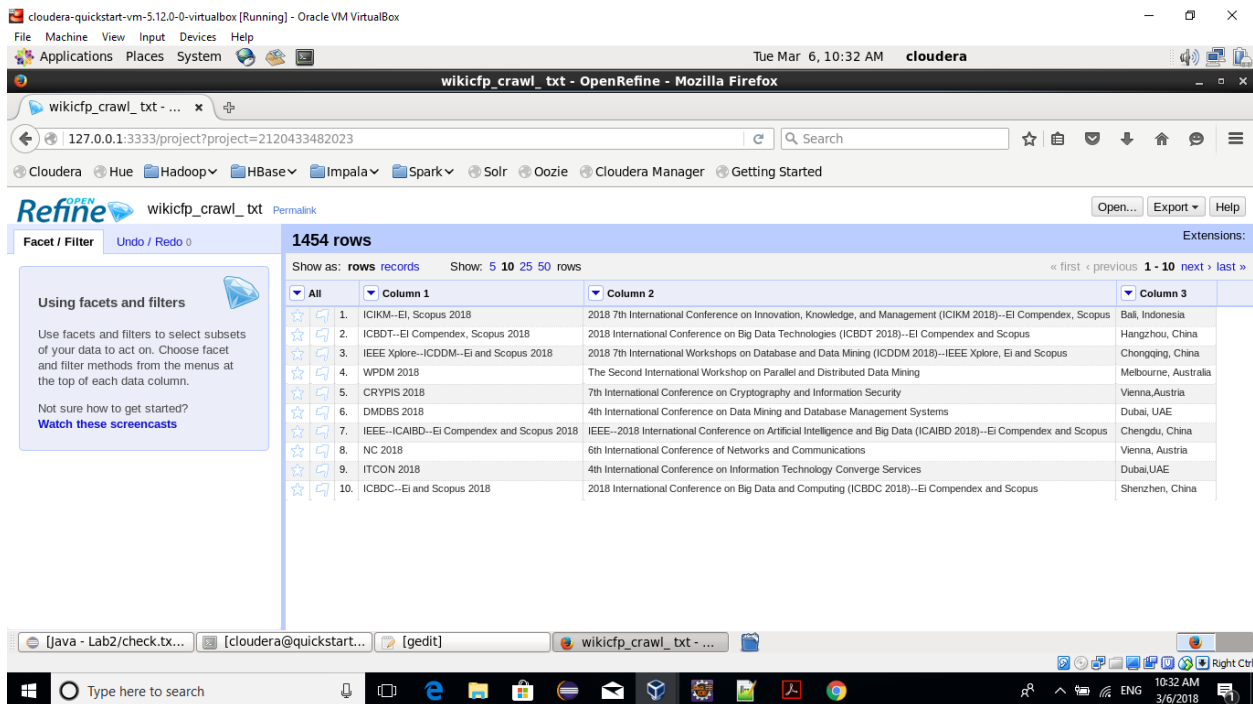
834     public static void main( String[ ] args ) throws NumberFormatException,
      IOException {
835         PlotGraphTask1 chart = new PlotGraphTask1("No of conference per city",
836             "Which city had the most?");
837         chart.pack( );
838         RefineryUtilities.centerFrameOnScreen( chart );
839         chart.setVisible( true );
840     }
841 }
842
843
844
845                                     Code for plotting graph - Task 4
846
847 package com.web.util;
848
849 import java.io.BufferedReader;
850 import java.io.File;
851 import java.io.FileReader;
852 import java.io.IOException;
853 import java.util.ArrayList;
854
855 import org.jfree.chart.ChartFactory;
856 import org.jfree.chart.ChartPanel;
857 import org.jfree.chart.JFreeChart;
858 import org.jfree.chart.plot.PlotOrientation;
859 import org.jfree.data.category.CategoryDataset;
860 import org.jfree.data.category.DefaultCategoryDataset;
861 import org.jfree.ui.ApplicationFrame;
862 import org.jfree.ui.RefineryUtilities;
863
864 @SuppressWarnings("serial")
865 public class PlotGraphTask4 extends ApplicationFrame {
866
867     public PlotGraphTask4(String applicationTitle, String chartTitle)
868         throws NumberFormatException, IOException {
869         super(applicationTitle);
870         //Bar chart to plot year and number of conferences
871         JFreeChart barChart = ChartFactory.createBarChart(chartTitle, "Year",
872             "No. of Conferences", createDataset(),
873             PlotOrientation.VERTICAL, true, true, false);
874
875         ChartPanel chartPanel = new ChartPanel(barChart);
876         chartPanel.setPreferredSize(new java.awt.Dimension(560, 367));
877         setContentPane(chartPanel);
878     }
879
880     /**
881     * @return
882     * @throws NumberFormatException
883     * @throws IOException
884     */
885     private CategoryDataset createDataset() throws NumberFormatException,
886         IOException {
887
888         ArrayList<String> cityList = new ArrayList<String>(); // An arraylist to
889             hold the list of cities
890         ArrayList<Integer> countList = new ArrayList<Integer>(); // An arraylist to
891             hold the list of counts
892         File file = new File(
893             "/home/cloudera/Desktop/Lab2/output1/task4/part-r-000001");
894
895         BufferedReader br = new BufferedReader(new FileReader(file));
896
897         String line;
898         // Plot the graph for years 2018,2017,2016,2015
899         while ((line = br.readLine()) != null) {
900             if ((line.contains("2017") || line.contains("2018")
901                 || line.contains("2016") || line.contains("2015") || line
902                     .contains("2014")) && (line.startsWith("A"))) {
903                 cityList.add(line.split("\t")[0]);
904                 countList.add(Integer.valueOf(line.split("\t")[1]));
905             }
906         }

```

```
904
905     }
906     br.close();
907
908     final DefaultCategoryDataset dataset = new DefaultCategoryDataset();
909
910     for (int i = 0; i < cityList.size() && i < countList.size(); i++) {
911         dataset.addValue(countList.get(i),
912             cityList.get(i).substring(0, cityList.get(i).length() - 4),
913             cityList.get(i).substring(cityList.get(i).length() - 4));
914     }
915     return dataset;
916 }
917
918 /**
919  * @param args
920  * @throws NumberFormatException
921  * @throws IOException
922  */
923 public static void main(String[] args) throws NumberFormatException,
924     IOException {
925     PlotGraphTask4 chart = new PlotGraphTask4(
926         "No of conference per city per year",
927         "Which city had the most?");
928     chart.pack();
929     RefineryUtilities.centerFrameOnScreen(chart);
930     chart.setVisible(true);
931 }
932 }
933
934
```



Using open refine, split year from the acronym of the conference. `value.substring()` is used to remove the year.



The result in open refine after filtering.

cloudera-quickstart-vm-5.12.0-0-virtualbox [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Applications Places System

Tue Mar 6, 10:32 AM cloudera

wikicfp\_crawl\_txt - OpenRefine - Mozilla Firefox

wikicfp\_crawl\_txt - ... x

127.0.0.1:3333/project?project=2120433482023

Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started

Refine wikicfp\_crawl\_txt Permalink

Open... Export Help

Facet / Filter Undo / Redo 0

1454 rows

Show as: rows records Show: 5 10 25 50 rows

Extensions: « first < previous 1 - 10 next > last »

Using facets and filters

Use facets and filters to select subsets of your data to act on. Choose facet and filter methods from the menus at the top of each data column.

Not sure how to get started? Watch these screencasts

All	Column 1	Column 2	Column 3
1.	ICIKM-EI, Scopus 2018	2018 7th International Conference on Innovation, Knowledge, and Management (ICIKM 2018)--EI Compendex, Scopus	Bali, Indonesia
2.	ICBDT-EI Compendex, Scopus 2018	2018 International Conference on Big Data Technologies (ICBDT 2018)--EI Compendex and Scopus	Hangzhou, China
3.	IEEE Xplore-ICDDM-EI and Scopus 2018	2018 7th International Workshops on Database and Data Mining (ICDDM 2018)--IEEE Xplore, EI and Scopus	Chongqing, China
4.	WPDM 2018	The Second International Workshop on Parallel and Distributed Data Mining	Melbourne, Australia
5.	CRYPIS 2018	7th International Conference on Cryptography and Information Security	Vienna, Austria
6.	DMDBS 2018	4th International Conference on Data Mining and Database Management Systems	Dubai, UAE
7.	IEEE-ICAIBD-EI Compendex and Scopus 2018	IEEE--2018 International Conference on Artificial Intelligence and Big Data (ICAIBD 2018)--EI Compendex and Scopus	Chengdu, China
8.	NC 2018	6th International Conference of Networks and Communications	Vienna, Austria
9.	ITCON 2018	4th International Conference on Information Technology Converge Services	Dubai, UAE
10.	ICBDC-EI and Scopus 2018	2018 International Conference on Big Data and Computing (ICBDC 2018)--EI Compendex and Scopus	Shenzhen, China

[java - Lab2/check.tx... [cloudera@quickstart... [gedit] wikicfp\_crawl\_txt - ...

Type here to search

10:33 AM 3/6/2018

cloudera-quickstart-vm-5.12.0-0-virtualbox [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Applications Places System

Tue Mar 6, 10:32 AM cloudera

wikicfp\_crawl\_txt - OpenRefine - Mozilla Firefox

wikicfp\_crawl\_txt - ... x

127.0.0.1:3333/project?project=2120433482023

Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started

Refine wikicfp\_crawl\_txt Permalink

Open... Export Help

Facet / Filter Undo / Redo 0

1454 rows

Show as: rows records Show: 5 10 25 50 rows

Extensions: « first < previous 1 - 10 next > last »

Using facets and filters

Use facets and filters to select subsets of your data to act on. Choose facet and filter methods from the menus at the top of each data column.

Not sure how to get started? Watch these screencasts

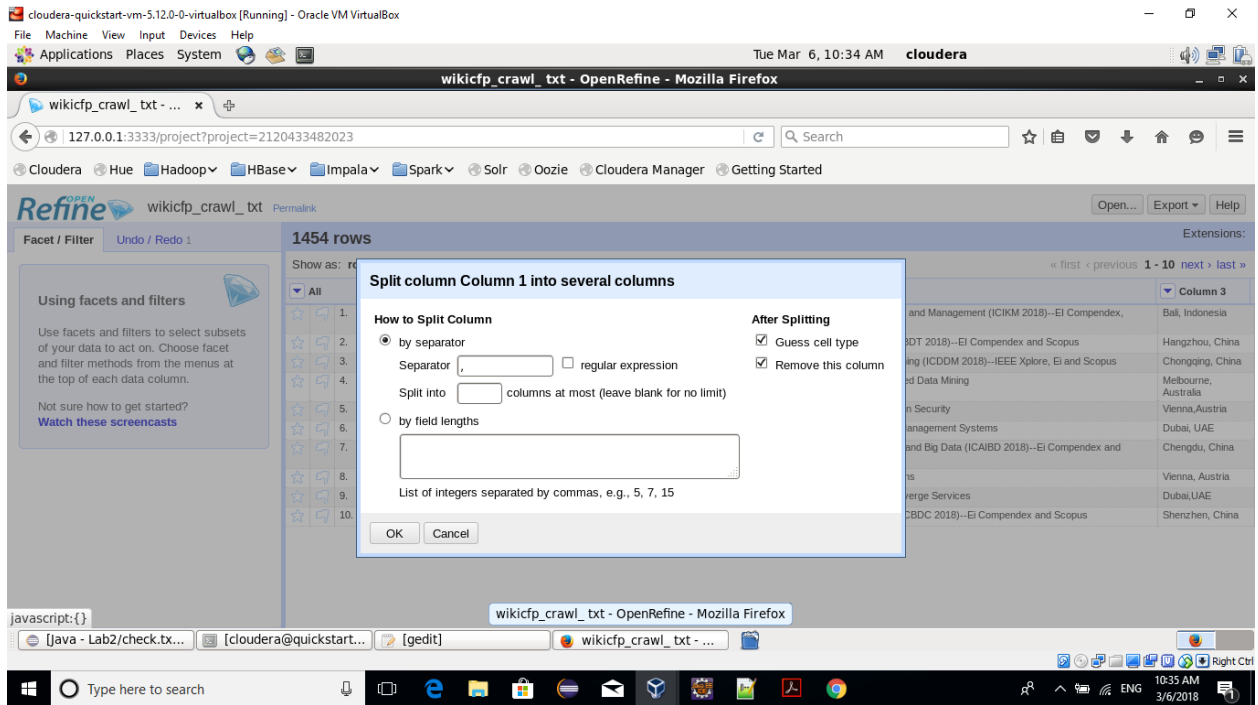
All	Column 1	Column 2	Column 3
1.	Facet	2018 7th International Conference on Innovation, Knowledge, and Management (ICIKM 2018)--EI Compendex, Scopus	Bali, Indonesia
2.	Text filter	scopus 2018	2018 International Conference on Big Data Technologies (ICBDT 2018)--EI Compendex and Scopus
3.	Edit cells	and Scopus 2018	2018 7th International Workshops on Database and Data Mining (ICDDM 2018)--IEEE Xplore, EI and Scopus
4.	Edit column	The Second International Workshop on Parallel and Distributed Data Mining	Chongqing, China
5.	Transpose	ie on Cryptography and Information Security	Melbourne, Australia
6.	Sort..	ie on Data Mining and Database Management Systems	Vienna, Austria
7.	View	ference on Artificial Intelligence and Big Data (ICAIBD 2018)--EI Compendex and Scopus	Dubai, UAE
8.	Reconcile	ie of Networks and Communications	Chengdu, China
9.		ie on Information Technology Converge Services	Vienna, Austria
10.		nce on Big Data and Computing (ICBDC 2018)--EI Compendex and Scopus	Dubai, UAE

javascript:{} [java - Lab2/check.tx... [cloudera@quickstart... [gedit] wikicfp\_crawl\_txt - ...

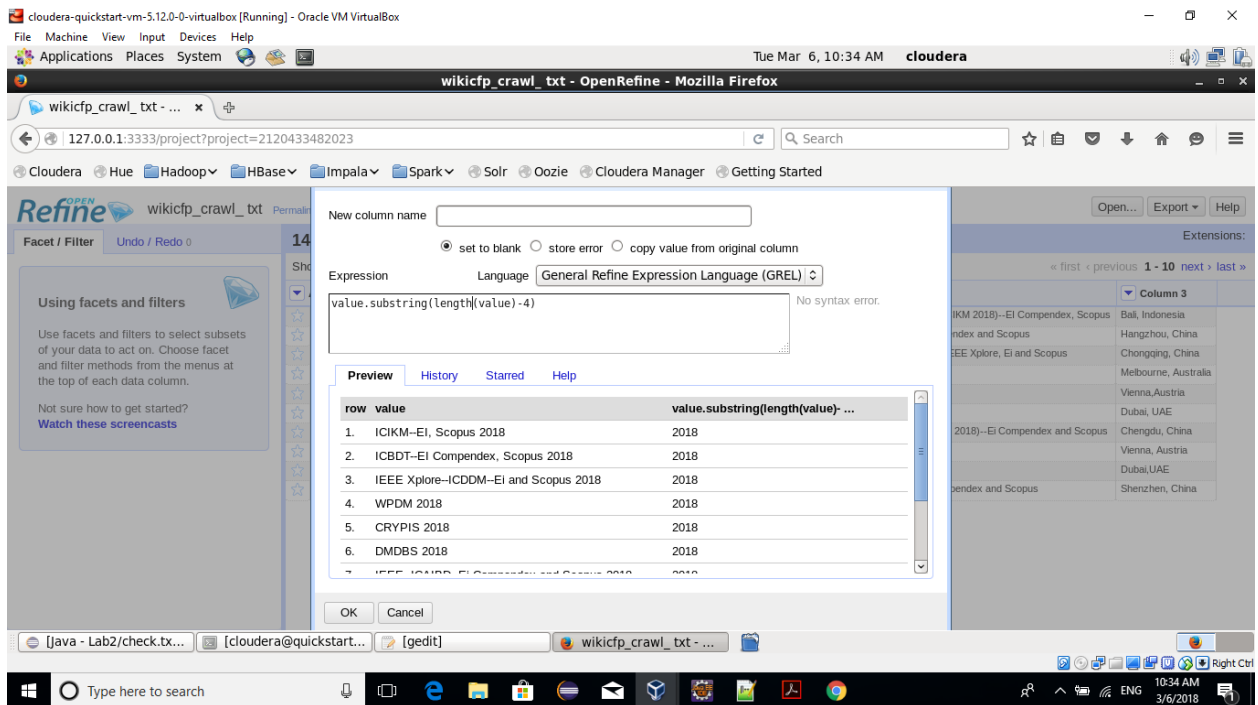
Type here to search

10:33 AM 3/6/2018





Splitting the column into two. Year and acronym separately. Using Edit column → Split into several columns.



Now the year column is made as a new column.

cloudera-quickstart-vm-5.12.0-0-virtualbox [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Applications Places System

Tue Mar 6, 10:34 AM cloudera

wikicfp\_crawl\_txt - OpenRefine - Mozilla Firefox

wikicfp\_crawl\_txt - ... x

127.0.0.1:3333/project?project=2120433482023

Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started

**Refine** wikicfp\_crawl\_txt [Permalink](#) **Create new column year based on column Column 1 by filling 1454 rows with gre:value.substring(length(value)-4)** [Undo](#) [Open...](#) [Export](#) [Help](#)

Facet / Filter [Undo](#) / [Redo](#) 1

**1454 rows** Extensions:

Show as: [rows](#) [records](#) Show: 5 10 25 50 rows « first < previous 1 - 10 next > last »

**Using facets and filters**

Use facets and filters to select subsets of your data to act on. Choose facet and filter methods from the menus at the top of each data column.

Not sure how to get started? [Watch these screencasts](#)

All	Column 1	year	Column 2	Column 3
1.	ICIKM-EI, Scopus 2018	2018	2018 7th International Conference on Innovation, Knowledge, and Management (ICIKM 2018)--EI Compendex, Scopus	Bali, Indonesia
2.	ICBDT-EI Compendex, Scopus 2018	2018	2018 International Conference on Big Data Technologies (ICBDT 2018)--EI Compendex and Scopus	Hangzhou, China
3.	IEEE Xplore-ICDDM-EI and Scopus 2018	2018	2018 7th International Workshops on Database and Data Mining (ICDDM 2018)--IEEE Xplore, EI and Scopus	Chongqing, China
4.	WPDm 2018	2018	The Second International Workshop on Parallel and Distributed Data Mining	Melbourne, Australia
5.	CRYPIS 2018	2018	7th International Conference on Cryptography and Information Security	Vienna, Austria
6.	DMDBS 2018	2018	4th International Conference on Data Mining and Database Management Systems	Dubai, UAE
7.	IEEE-ICAIBD-EI Compendex and Scopus 2018	2018	IEEE-2018 International Conference on Artificial Intelligence and Big Data (ICAIBD 2018)--EI Compendex and Scopus	Chengdu, China
8.	NC 2018	2018	6th International Conference of Networks and Communications	Vienna, Austria
9.	ITCON 2018	2018	4th International Conference on Information Technology Converge Services	Dubai, UAE
10.	ICBDC-EI and Scopus 2018	2018	2018 International Conference on Big Data and Computing (ICBDC 2018)--EI Compendex and Scopus	Shenzhen, China

[java - Lab2/check.tx... [cloudera@quickstart... [gedit] wikicfp\_crawl\_txt - ...

Type here to search

10:35 AM 3/6/2018

cloudera-quickstart-vm-5.12.0-0-virtualbox [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Applications Places System

Tue Mar 6, 10:36 AM cloudera

wikicfp\_crawl\_txt - OpenRefine - Mozilla Firefox

wikicfp\_crawl\_txt - ... x

127.0.0.1:3333/project?project=2506727471688

Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started

**Refine** wikicfp\_crawl\_txt [Permalink](#) [Open...](#) [Export](#) [Help](#)

Facet / Filter [Undo](#) / [Redo](#) 0

[Refresh](#) [Reset All](#) [Remove All](#)

**Column 3** [invert reset](#)

☐ case sensitive ☐ regular expression

**1195 matching rows (1454 total)** Extensions:

Show as: [rows](#) [records](#) Show: 5 10 25 50 rows « first < previous 1 - 10 next > last »

All	Column 1	Column 2	Column 3
1.	ICIKM-EI, Scopus 2018	2018 7th International Conference on Innovation, Knowledge, and Management (ICIKM 2018)--EI Compendex, Scopus	Bali, Indonesia
2.	ICBDT-EI Compendex, Scopus 2018	2018 International Conference on Big Data Technologies (ICBDT 2018)--EI Compendex and Scopus	Hangzhou, China
3.	IEEE Xplore-ICDDM-EI and Scopus 2018	2018 7th International Workshops on Database and Data Mining (ICDDM 2018)--IEEE Xplore, EI and Scopus	Chongqing, China
4.	WPDm 2018	The Second International Workshop on Parallel and Distributed Data Mining	Melbourne, Australia
5.	CRYPIS 2018	7th International Conference on Cryptography and Information Security	Vienna, Austria
6.	DMDBS 2018	4th International Conference on Data Mining and Database Management Systems	Dubai, UAE
7.	IEEE-ICAIBD-EI Compendex and Scopus 2018	IEEE-2018 International Conference on Artificial Intelligence and Big Data (ICAIBD 2018)--EI Compendex and Scopus	Chengdu, China
8.	NC 2018	6th International Conference of Networks and Communications	Vienna, Austria
9.	ITCON 2018	4th International Conference on Information Technology Converge Services	Dubai, UAE
10.	ICBDC-EI and Scopus 2018	2018 International Conference on Big Data and Computing (ICBDC 2018)--EI Compendex and Scopus	Shenzhen, China

[java - Lab2/check.tx... [cloudera@quickstart... [gedit] wikicfp\_crawl\_txt - ...

Type here to search

10:36 AM 3/6/2018

Removing those columns not containing city in location. Remove those columns that do not contain a comma.

cloudera-quickstart-vm-5.12.0-0-virtualbox [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Applications Places System

Tue Mar 6, 10:37 AM cloudera

wikicfp\_crawl\_txt - OpenRefine - Mozilla Firefox

127.0.0.1:3333/project?project=2506727471688

Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started

**Refine**

Facet / Filter

Refresh

Column 3

case sensitive

Expression Language: General Refine Expression Language (GREL)

value.substring(0,length(value)-4) No syntax error.

Preview History Starred Help

row	value	value.substring(0,length(value)-4)
1.	ICIKM-EI, Scopus 2018	ICIKM-EI, Scopus
2.	ICBDT-EI Compendex, Scopus 2018	ICBDT-EI Compendex, Scopus
3.	IEEE Xplore-ICDDM-EI and Scopus 2018	IEEE Xplore-ICDDM-EI and Scopus
4.	WPDM 2018	WPDM
5.	CRYPIS 2018	CRYPIS
6.	DMDBS 2018	DMDBS
7.	IEEE Xplore-EI Compendex and Scopus 2018	IEEE Xplore-EI Compendex and Scopus

On error: ☒ keep original ☐ set to blank ☐ store error ☐ Re-transform up to 10 times until no change

OK Cancel

[java - Lab2/check.tx... [cloudera@quickstart... [gedit] wikicfp\_crawl\_txt - ...

Type here to search

10:38 AM 3/6/2018