HW Assignment 2 - CS267-Fall 2023

Hadoop/MapReduce with Data Cleaning Hands-on Exercises

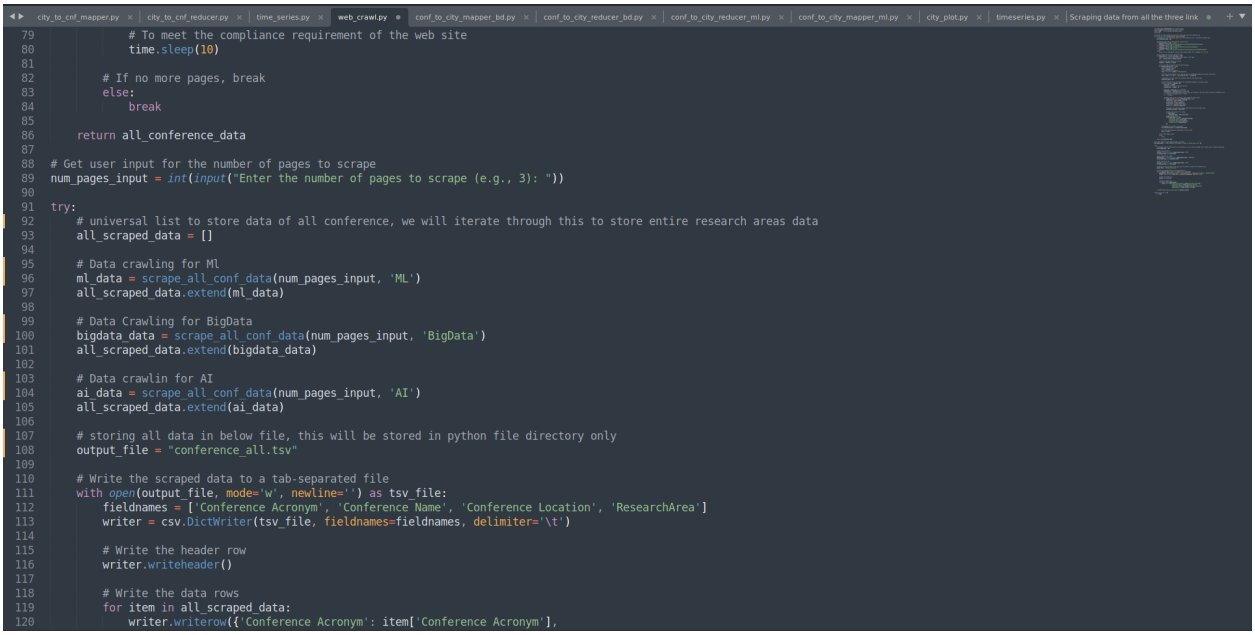
Name : Sanket Kulkarni (017413461)

Assignment Report Content:

Exercise 1: Data Crawling to Collect Data

Web Crawler: Scraping data from all the three links in one go by taking input of the number of pages we want to scrape.
Links:
http://www.wikicfp.com/cfp/call?conference=Big%20Data
http://www.wikicfp.com/cfp/call?conference=machine%20learning
http://www.wikicfp.com/cfp/call?conference=Artificial%20Intelligence

Source Code Snippets: From calling to function wise



```
            # To meet the compliance requirement of the web site
            time.sleep(10)

        # If no more pages, break
        else:
            break

    return all_conference_data

# Get user input for the number of pages to scrape
num_pages_input = int(input("Enter the number of pages to scrape (e.g., 3): "))

try:
    # universal list to store data of all conference, we will iterate through this to store entire research areas data
    all_scraped_data = []

    # Data crawling for Ml
    ml_data = scrape_all_conf_data(num_pages_input, 'ML')
    all_scraped_data.extend(ml_data)

    # Data Crawling for BigData
    bigdata_data = scrape_all_conf_data(num_pages_input, 'BigData')
    all_scraped_data.extend(bigdata_data)

    # Data crawlin for AI
    ai_data = scrape_all_conf_data(num_pages_input, 'AI')
    all_scraped_data.extend(ai_data)

    # storing all data in below file, this will be stored in python file directory only
    output_file = "conference_all.tsv"

    # Write the scraped data to a tab-separated file
    with open(output_file, mode='w', newline='') as tsv_file:
        fieldnames = ['Conference Acronym', 'Conference Name', 'Conference Location', 'ResearchArea']
        writer = csv.DictWriter(tsv_file, fieldnames=fieldnames, delimiter='\t')

        # Write the header row
        writer.writeheader()

        # Write the data rows
        for item in all_scraped_data:
            writer.writerow({'Conference Acronym': item['Conference Acronym'],
```

Sanket Kulkarni : HW2

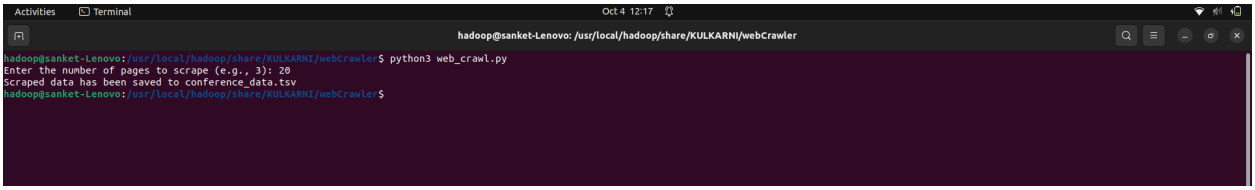
```python
from bs4 import BeautifulSoup # for data scrapnig
import requests # for hitting the pages with HTTP
import time     # for storing info about delay
import csv

# Function to scrape conference data for a given page count and research area
def scrape_all_conf_data(num_pages, research_area):
    # Initialize a list which store data of all the conferences for a particular research area
    all_conference_data = []

    # Define the base URL for the selected research area
    if research_area.lower() == 'ml':
        base_url = "http://www.wikicfp.com/cfp/call?conference=Machine%20Learning"
    elif research_area.lower() == 'bigdata':
        base_url = "http://www.wikicfp.com/cfp/call?conference=Big%20Data"
    elif research_area.lower() == 'ai':
        base_url = "http://www.wikicfp.com/cfp/call?conference=Artificial%20Intelligence"
    else:
        raise ValueError("Invalid research area. Please choose 'ML', 'BigData', or 'AI'.")

    # Loop through the specified number of pages
    for page_number in range(1, num_pages + 2):
        # URL for the current page based on page number in the loop
        url = f"{base_url}&page={page_number}"

        # Send an HTTP GET request to the URL
        response = requests.get(url)

        # if we get 200 OK response from the HTTP requests
        if response.status_code == 200:
            # parse the html content
            html = response.text
            # initialisation of soup lib
            soup = BeautifulSoup(html, 'html.parser')

            # The data on the website is in rows and rows are identified based on the color of the row.
            line = soup.find_all('tr', bgcolor=["#f6f6f6", "#e6e6e6"])

            # Initialize a list to store the extracted data for the current page
            conference_data = []

            # Loop through the rows and extract the information (similar to previous code)
```

```python
                location_row = line[i + 1]

                event_col = event_row.find_all('td')
                location_col = location_row.find_all('td')
                # Previously I had added below check as well, but thought to do entire data cleaning in OpenRefine only
                # if location_columns[1].text == 'N/A':
                #     continue

                # Column check to data scraping, count should be larger tha 3
                if len(event_col) >= 3 and len(location_col) >= 3:
                    event_name = event_col[0].find('a')
                    # Store conf name if it exists
                    event_fname = event_col[1].text
                    # Store conf location if it exists
                    event_loc = location_col[1].text

                    # Extract the conference acronym name directly from the event name
                    conference_acronym = event_name

                    # Store conf acronym if it exists
                    if event_name:
                        conference_name = event_name.text
                    # data append format
                    conference_data.append({
                        'Conference Acronym': conference_acronym,
                        'Conference Name': event_fname,
                        'Conference Location': event_loc,
                        'Research Area': research_area
                    })

            # Appending all the data to one list
            all_conference_data.extend(conference_data)

            # To meet the compliance requirement of the web site
            time.sleep(10)

        # If no more pages, break
        else:
            break

    return all_conference_data
```

Asking for page inputs:

```
Activities  Terminal                    Oct 4 12:09
hadoop@sanket-Lenovo: /usr/local/hadoop/share/KULKARNI/webCrawler
hadoop@sanket-Lenovo:/usr/local/hadoop/share/KULKARNI/webCrawler$ python3 web_crawl.py
Enter the number of pages to scrape (e.g., 3): 20
```

Result after web crawler ran:

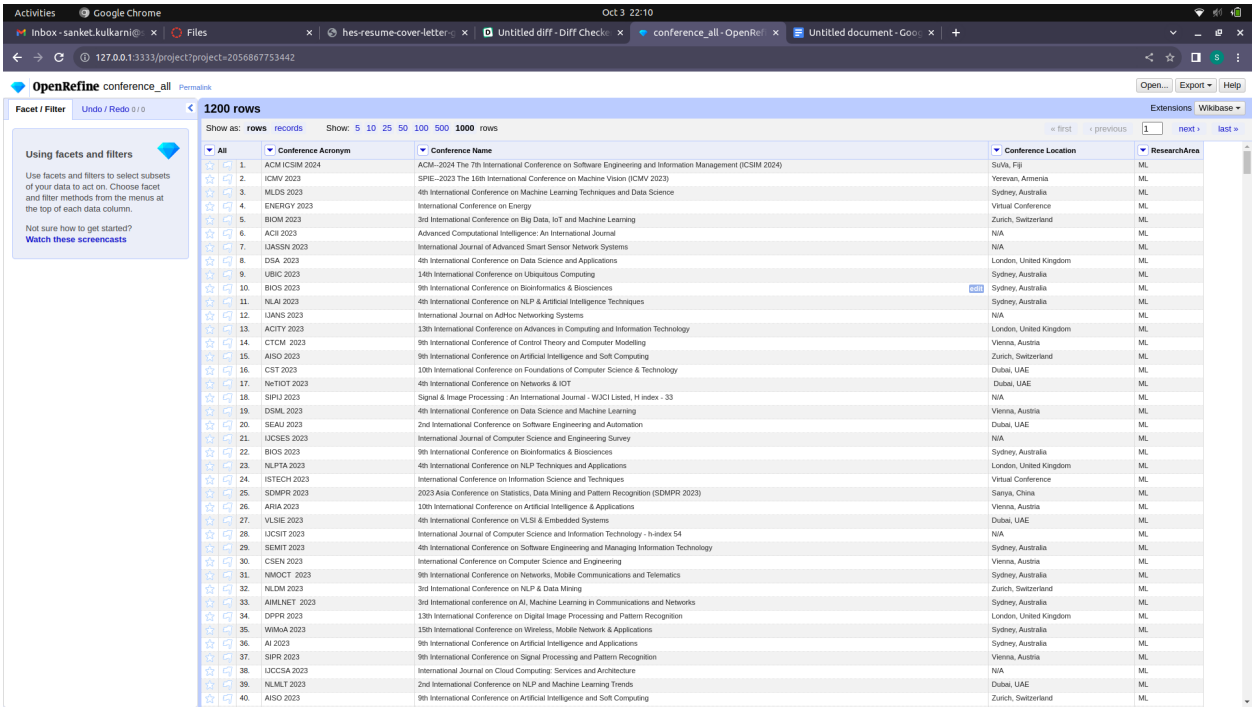


Web Crawler Working:

- Take the input about how many pages you want to scrape
- I have called web crawler function three times in code to store the data of all research areas
- Used the ‘beautiful soap’ library of python to scrape the data.
- To install beautiful soap I used below command:
- Pip install bs4
- Pip install requests
- Loop through all the research areas one by one and scrape data for it.
- Add a delay of 8-10 seconds to comply with policies of wikicfp
- Now, looping through the table for 20 pages and extracting the rows based on colors
- For each page we have a list, append this to total list
- This total list will then further appended to overall research list
- The output is stored as a tsv file

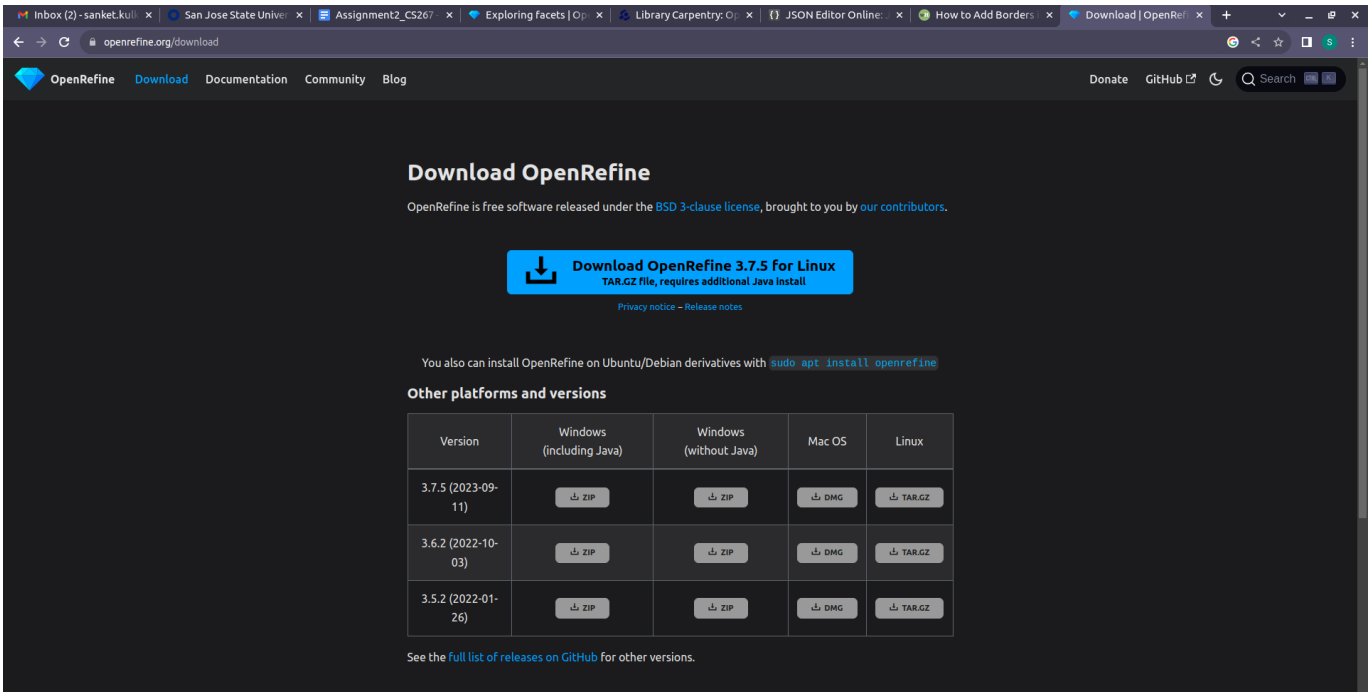
The output file looks like this:

Generated Conference Data for all 1200 rows for 20 pages per research area (AI/ML/BigData)

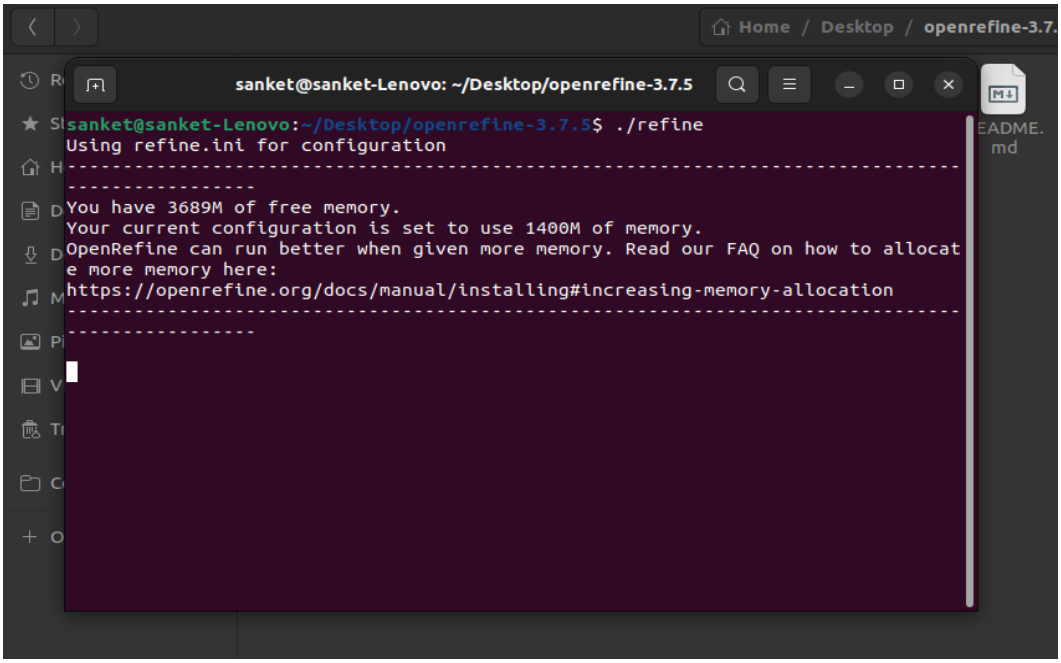


Exercise 2: Data Cleaning using Open Refine

Download Open Refine from google:

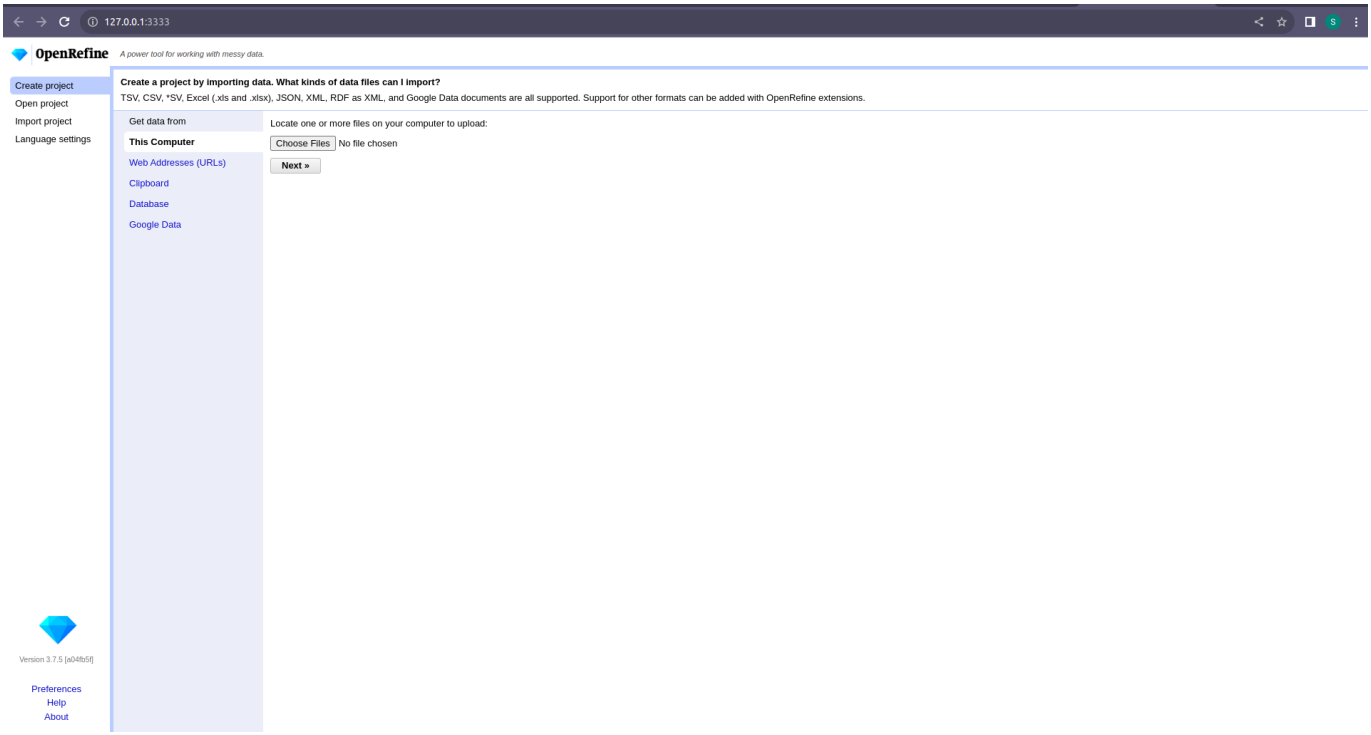


Unzip the folder
Once done, open a terminal from this folder
To run on terminal type **./refine**



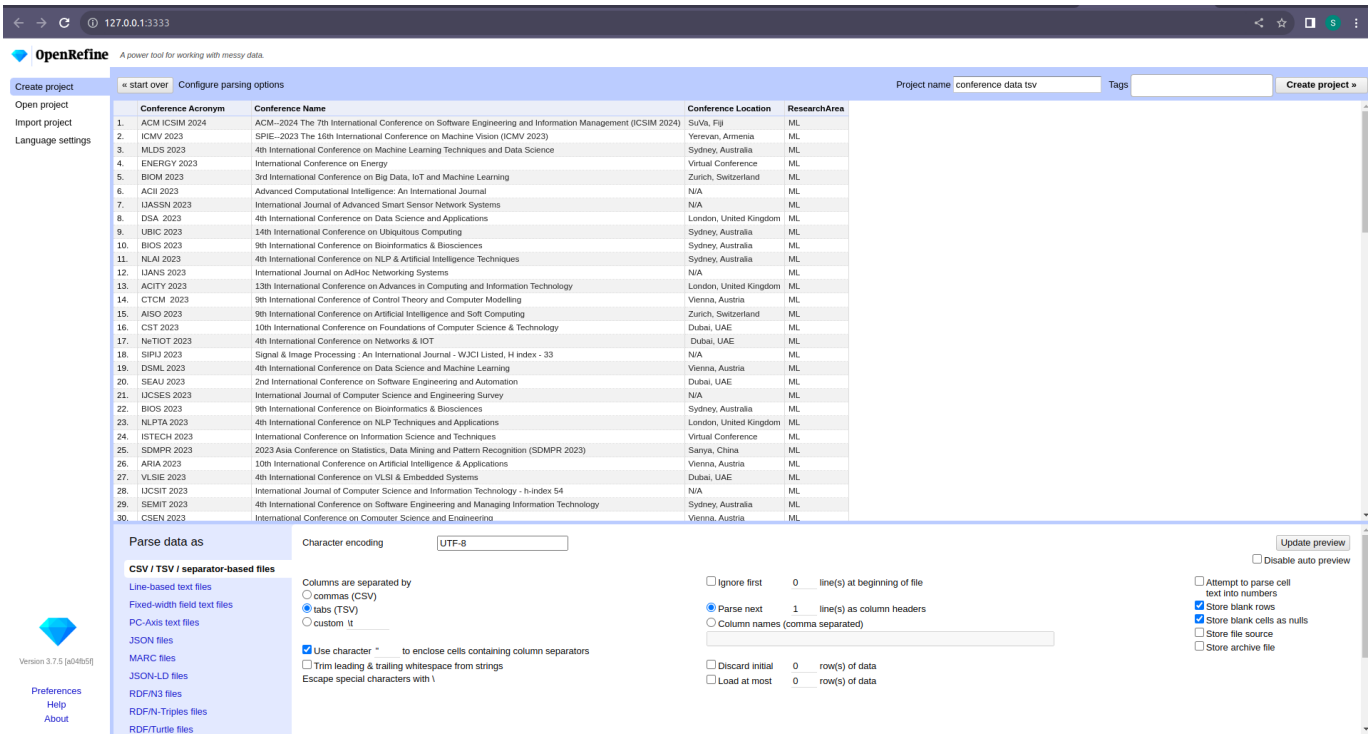
This will open a localhost window on to your browser:

Sanket Kulkarni : HW2



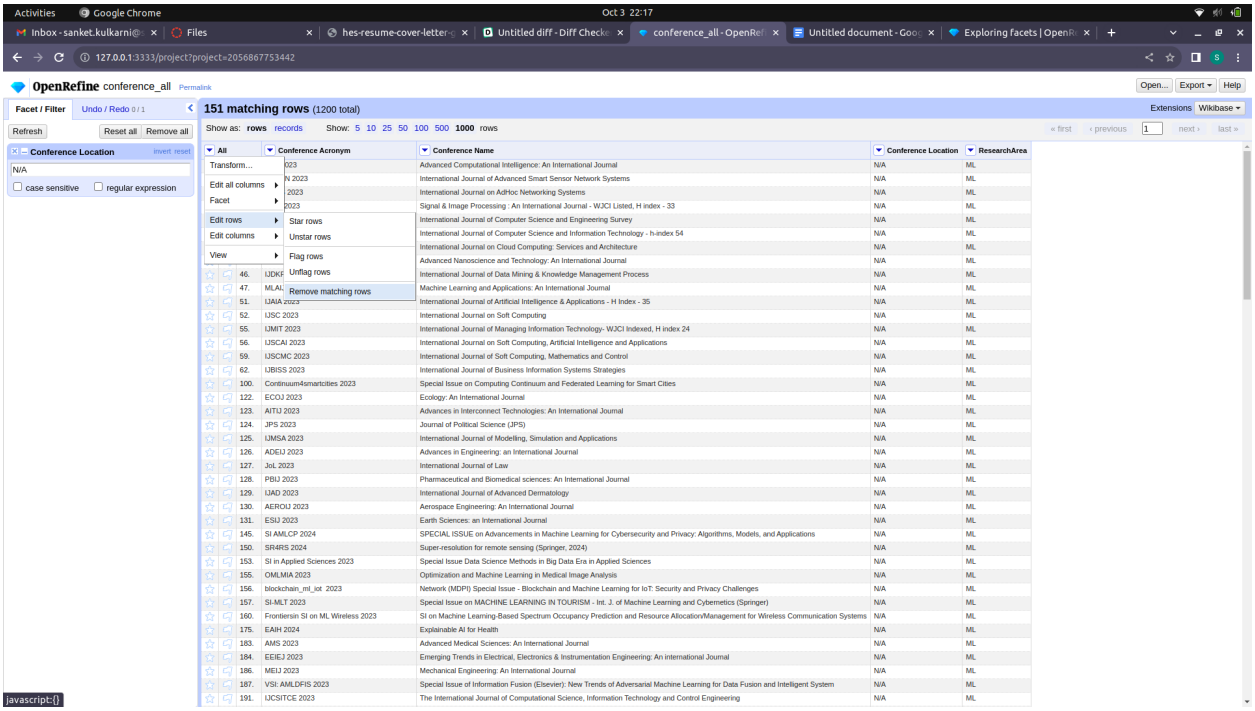
Chose the file that we generated from web crawler in exercise 1

It will generate a preview, which would look like this:

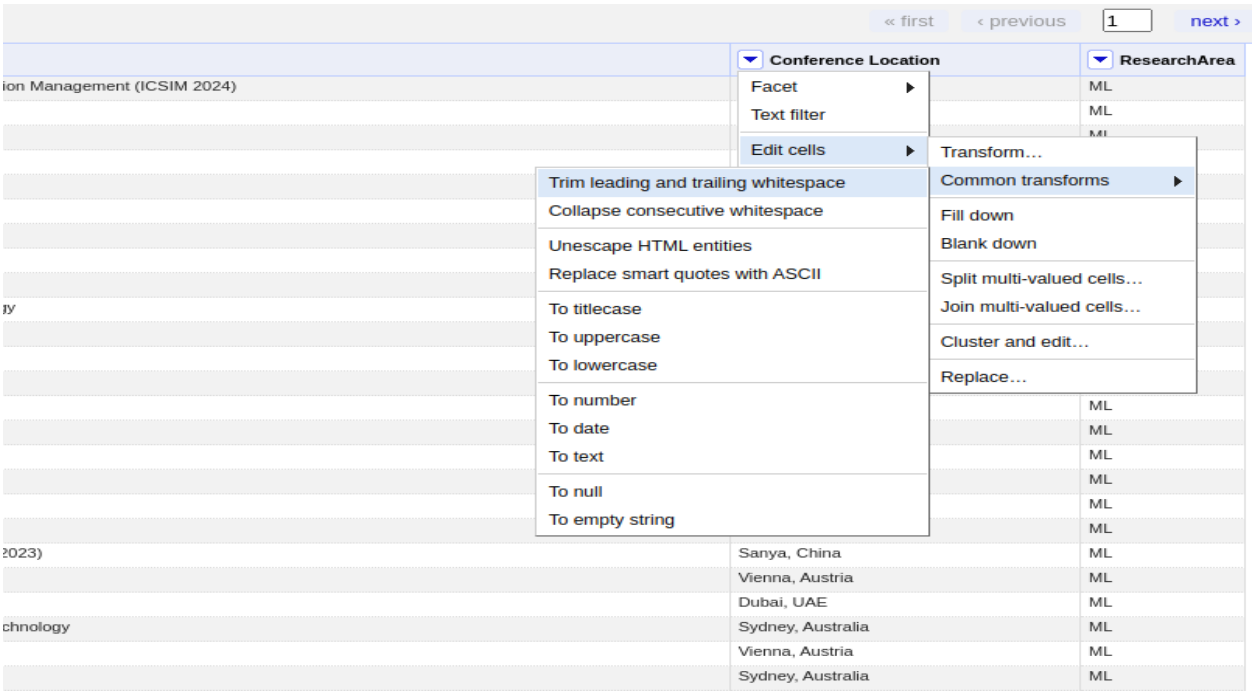


Sanket Kulkarni : HW2

Apply a text filter on Conference Location column to remove all N/A locations:
This generated 151 rows and now select all of them and remove

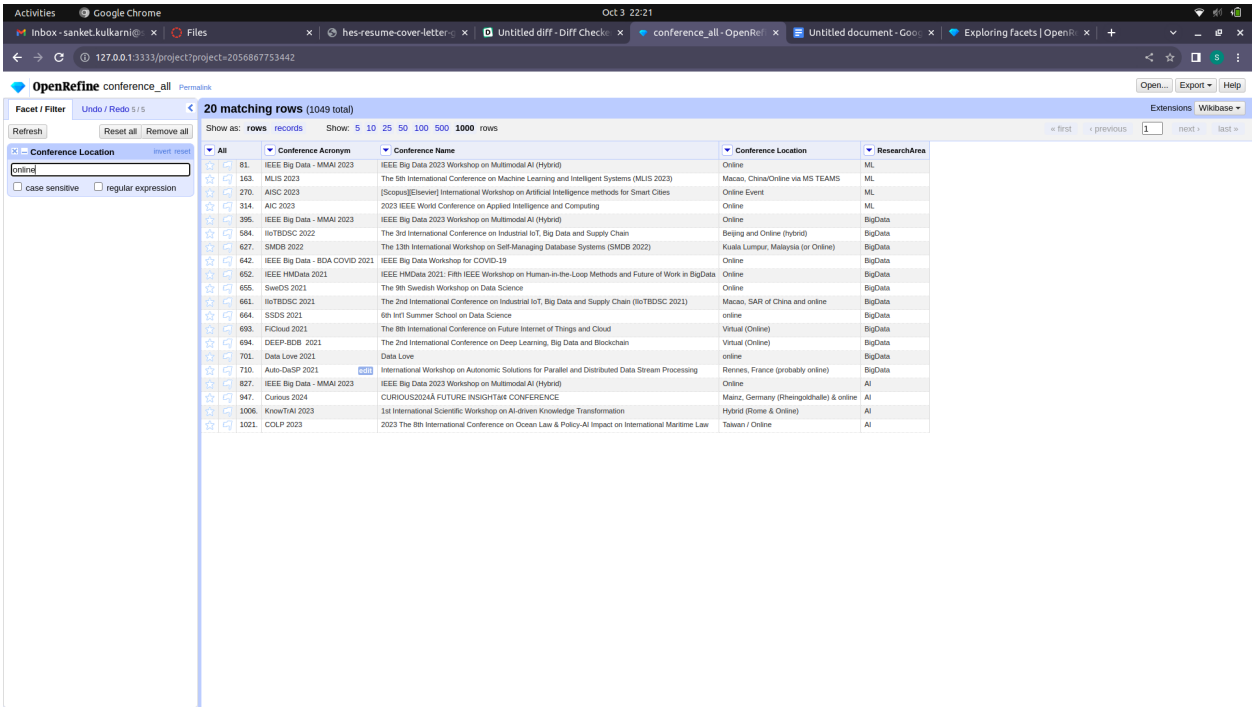


Remove leading and trailing spaces in all columns

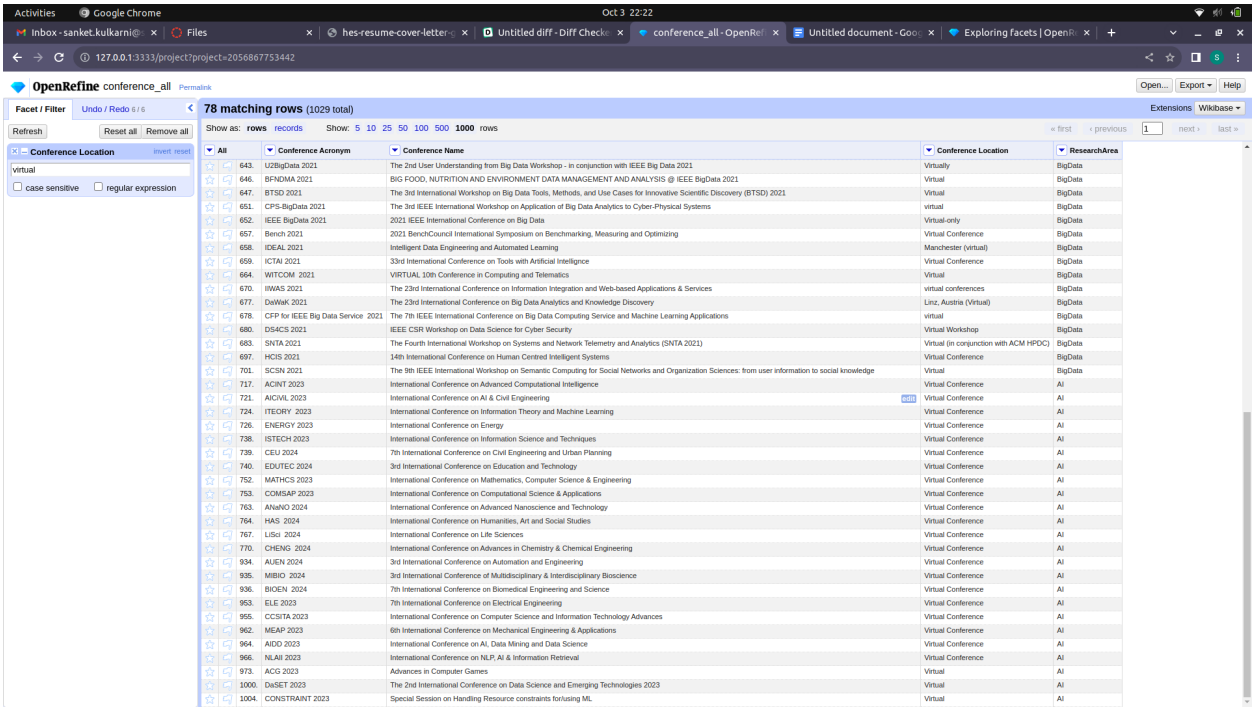


Sanket Kulkarni : HW2

Removing all rows with location as online, you can filter these rows using text filter in column drop down

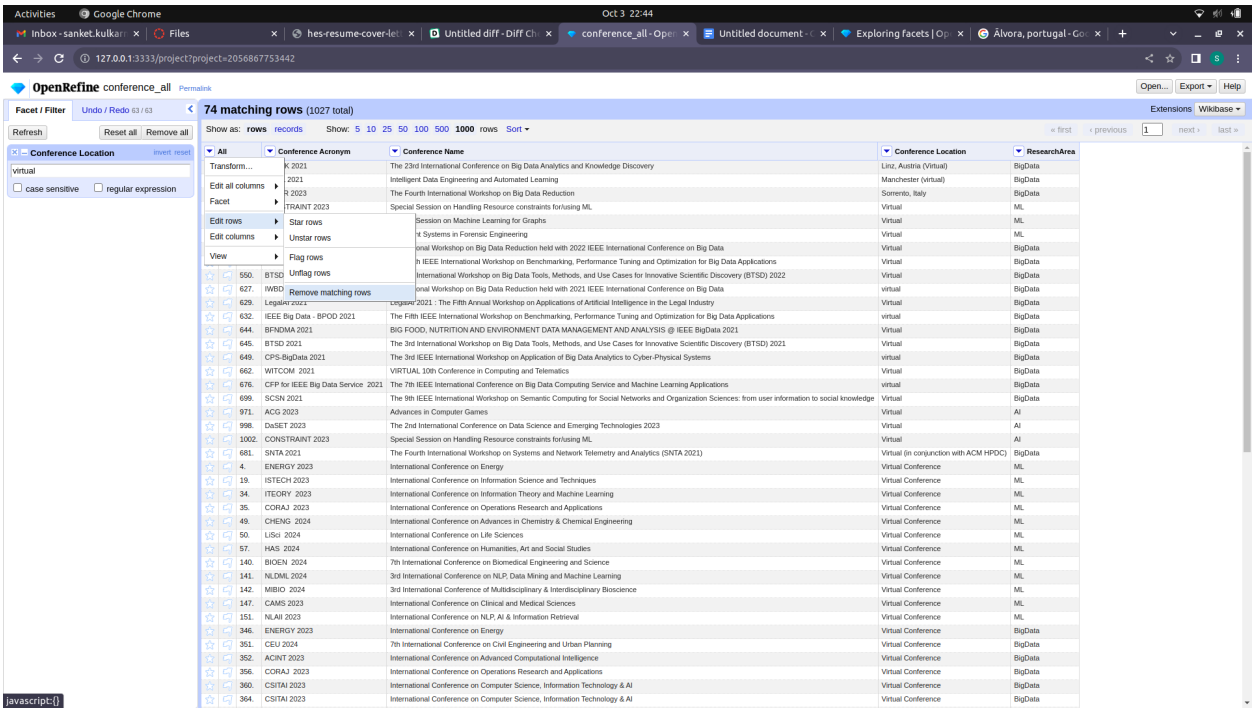


Removing all rows with location as virtual, you can filter these rows using text filter in column drop down

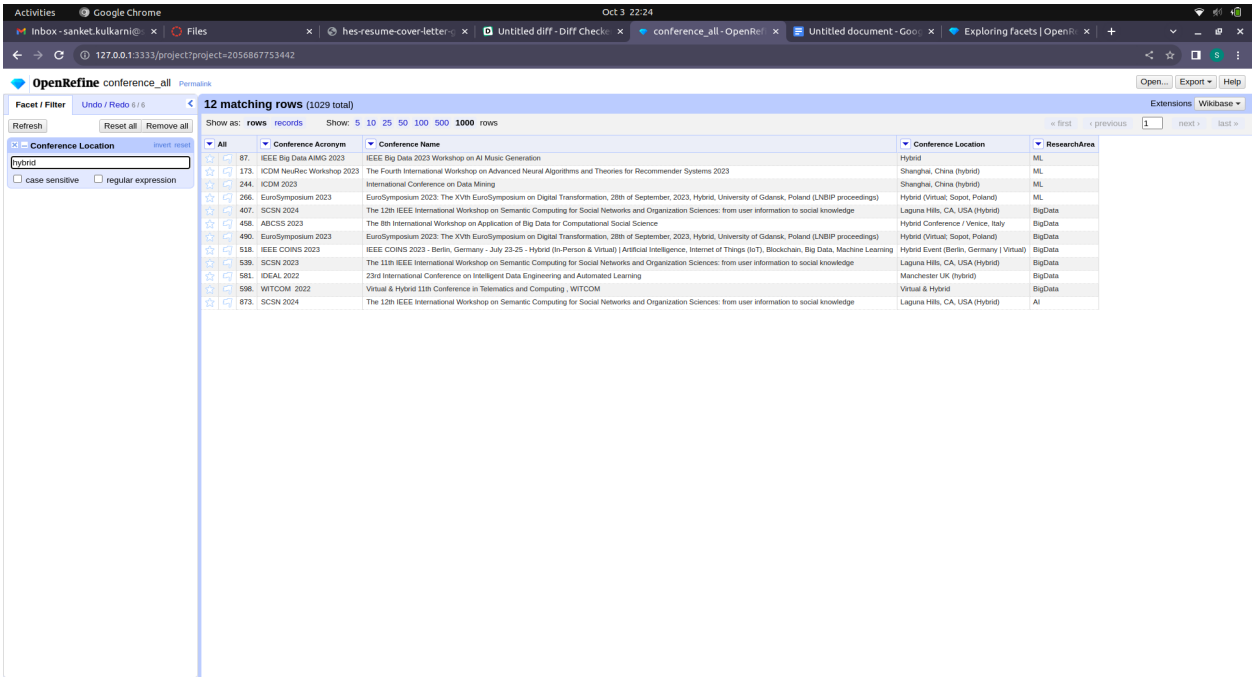


Sanket Kulkarni : HW2

Unselecting the hybrid locations rows from these as guided by the professor

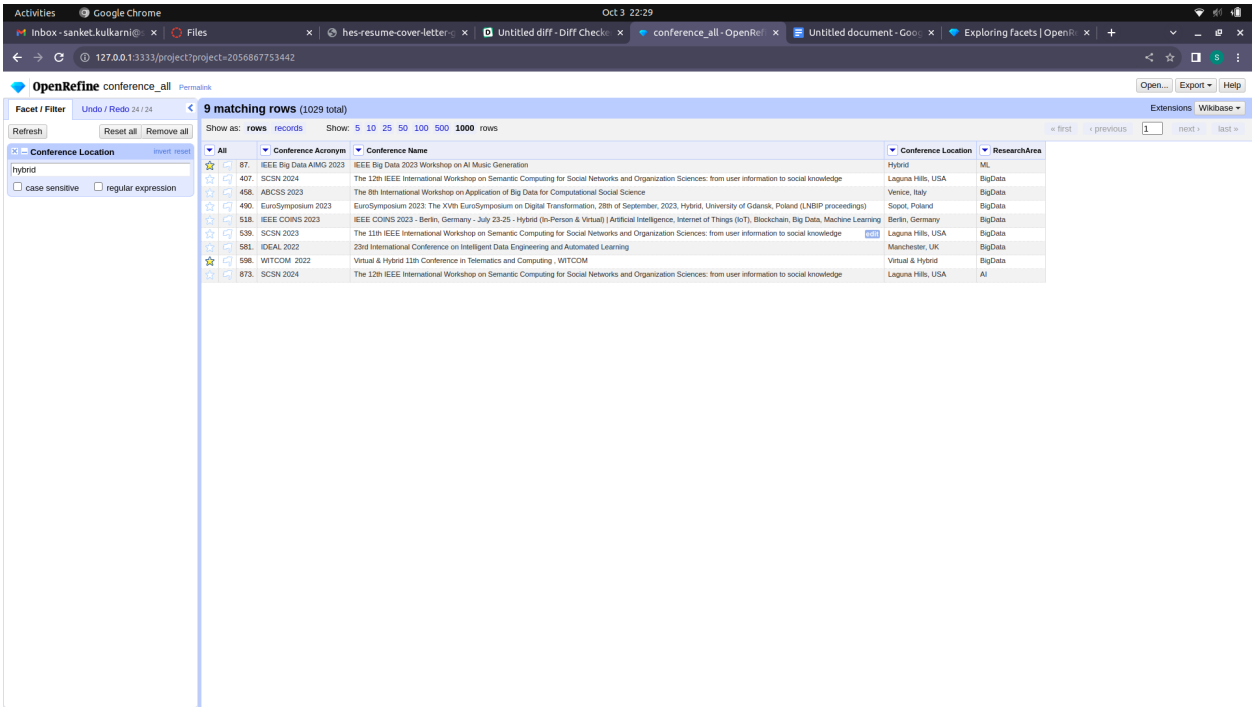


Found below rows for hybrid and cleaning the data manually here, just keeping the city and country and not the hybrid word for the further processing

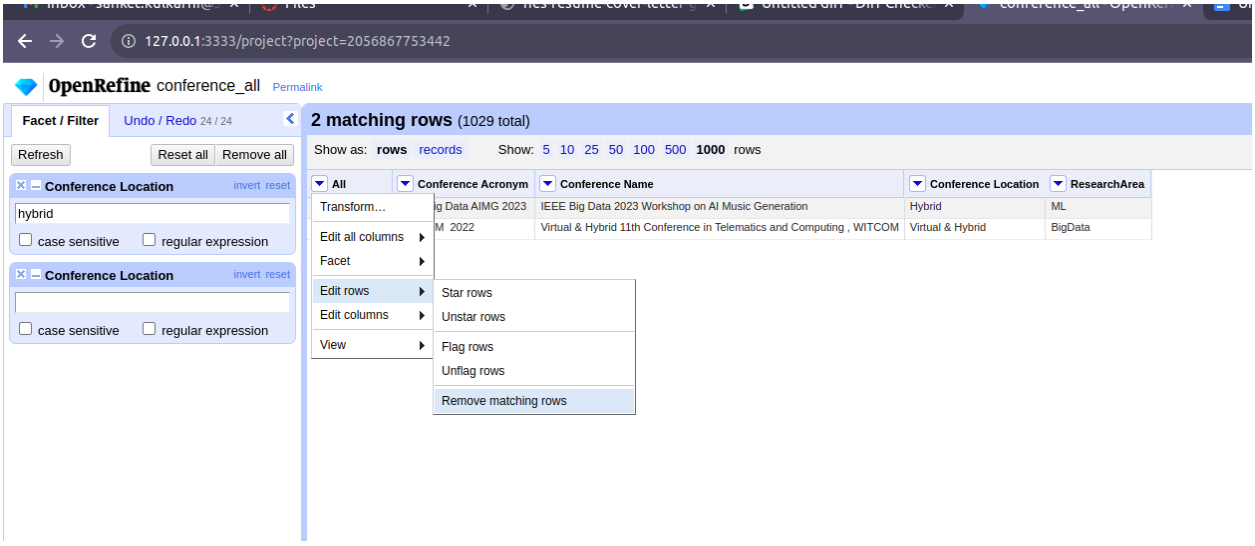


Sanket Kulkarni : HW2

After cleaning , we have two rows whose location is not mentioned so flagging those

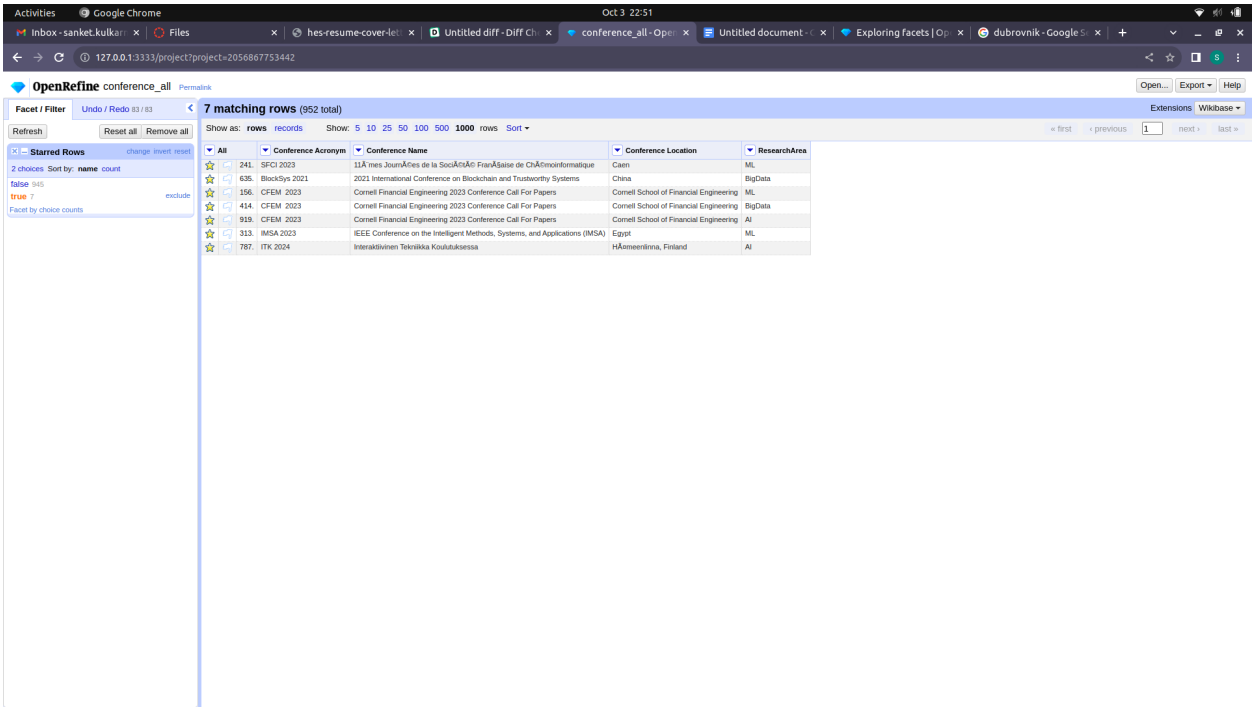


Removing the flagged two rows

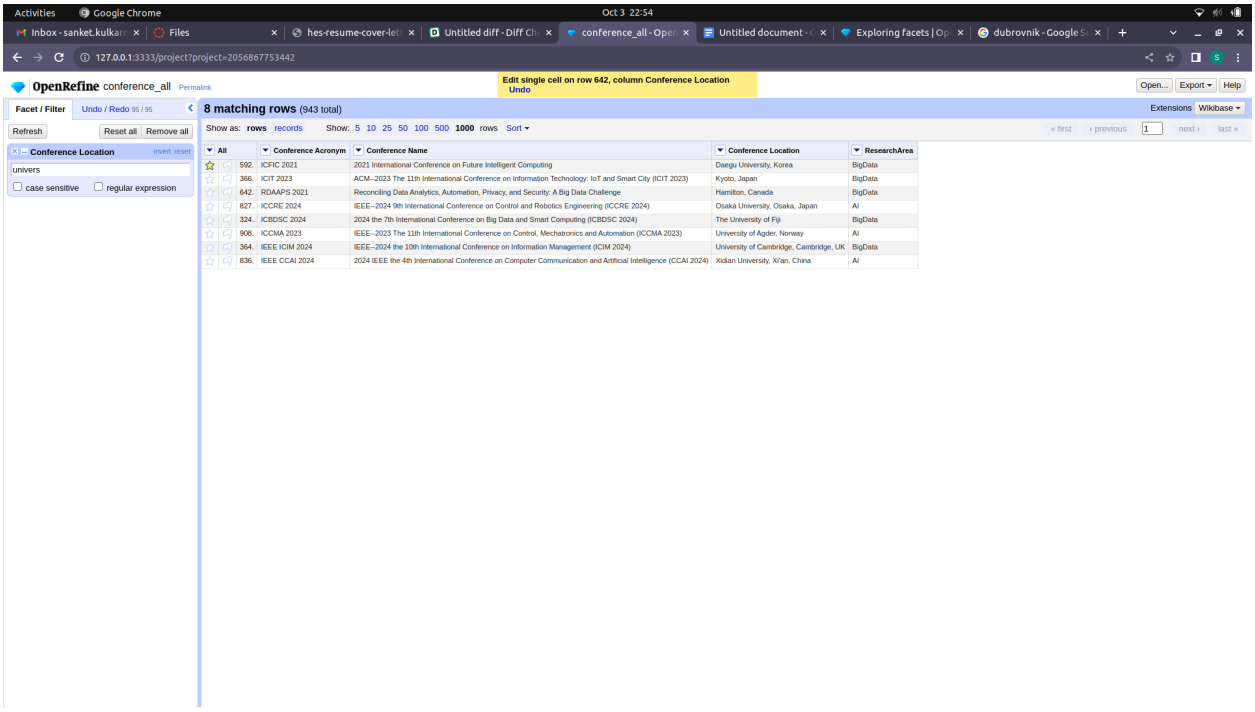


Sanket Kulkarni : HW2

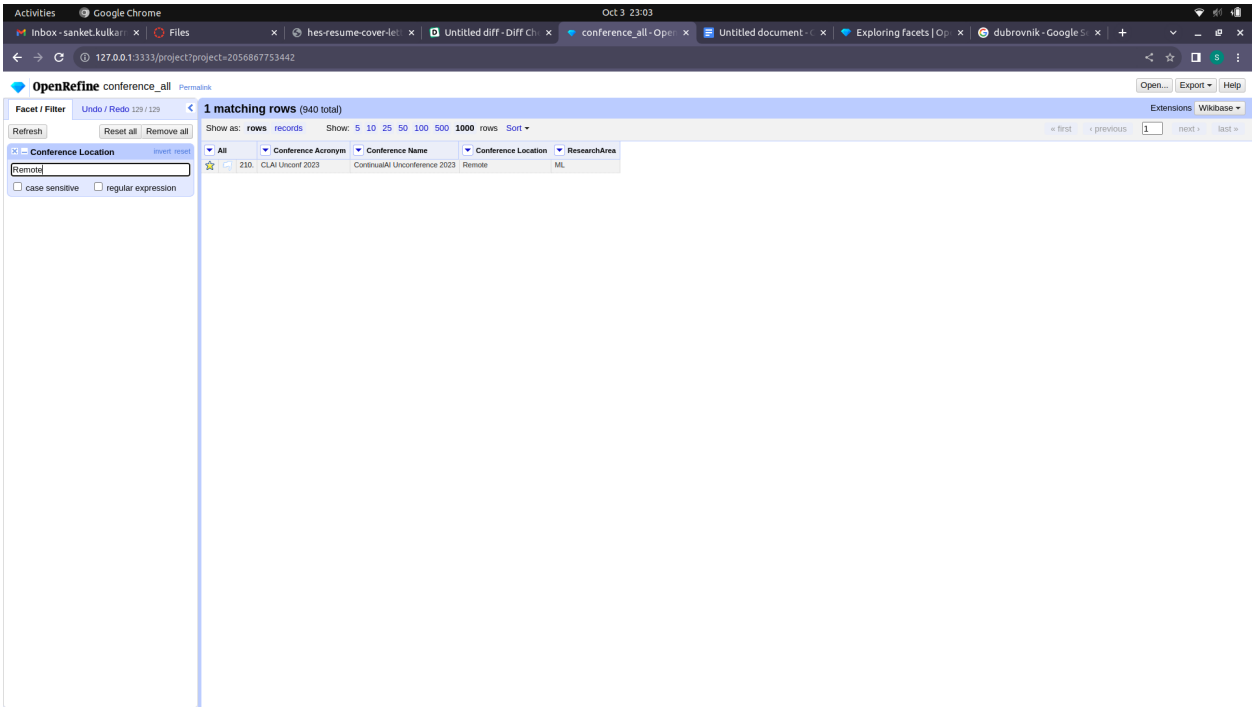
Removing few more rows with no city mentioned and only country is given



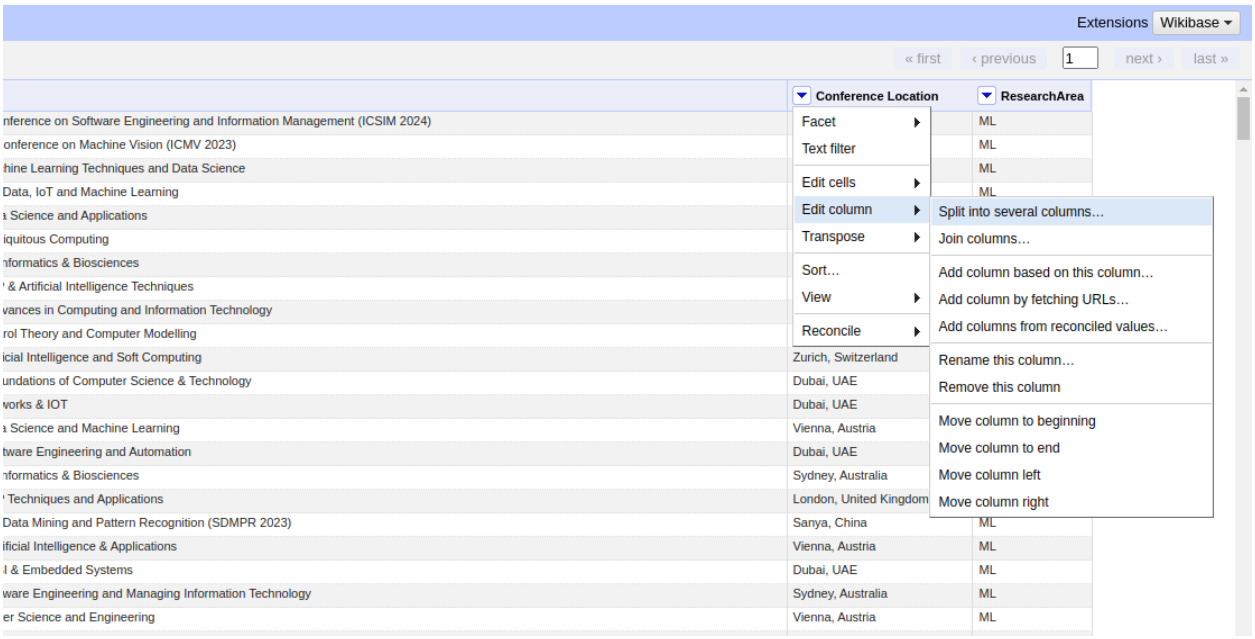
Making changes to university location as city and country



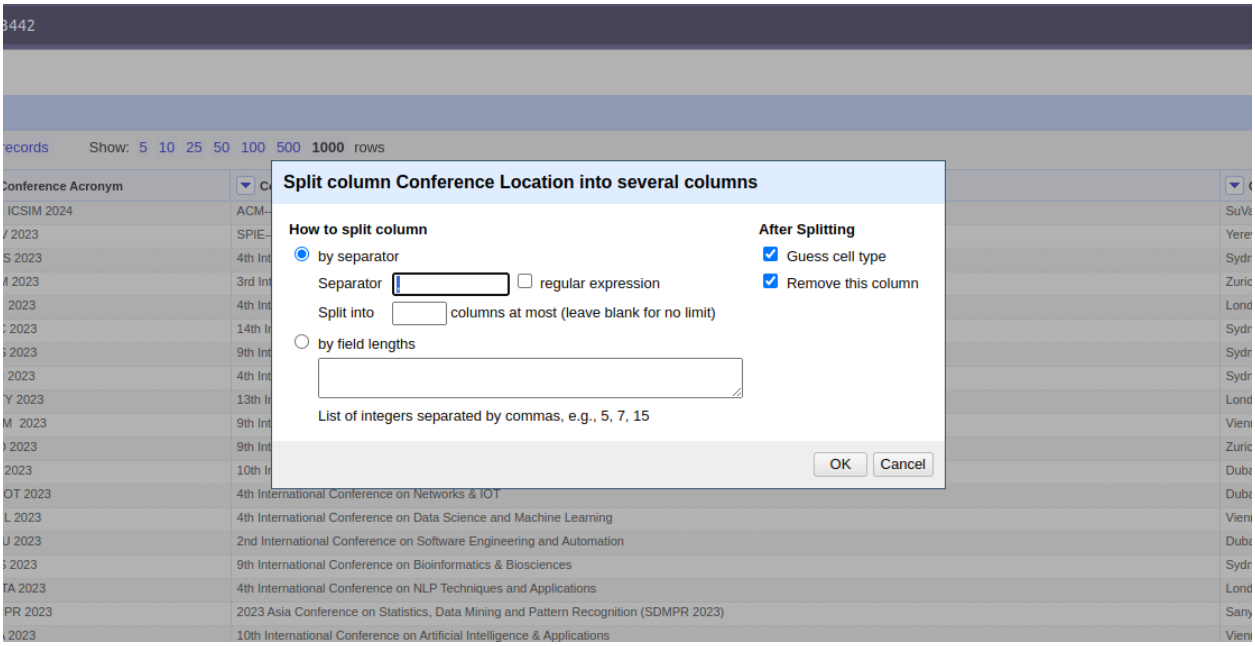
Few locations with Remote space, hence removing those



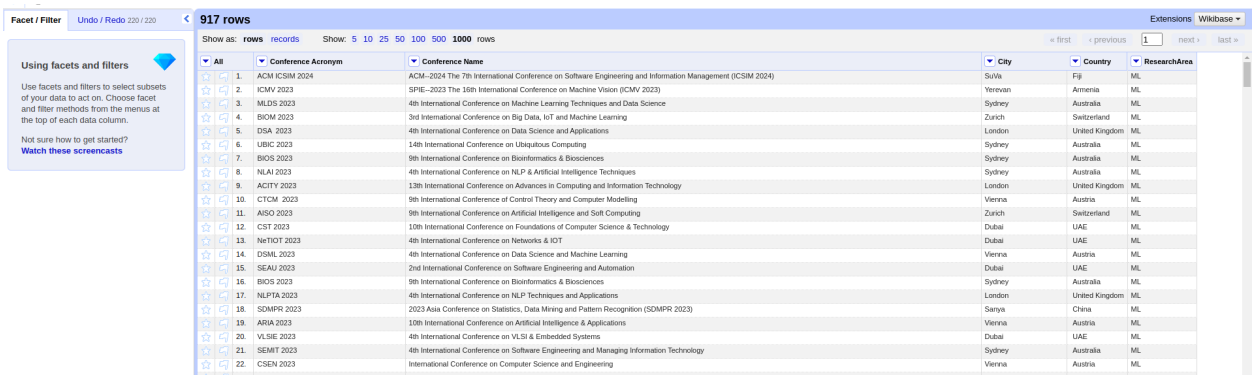
Splitting the column into 2, separating cities and countries in two columns now as we would feed our mapper a cleaned data of city



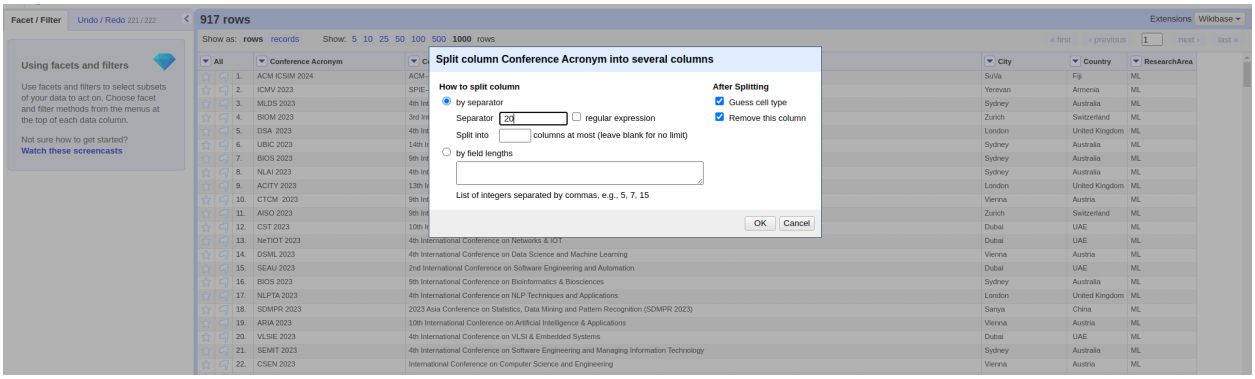
Sanket Kulkarni : HW2



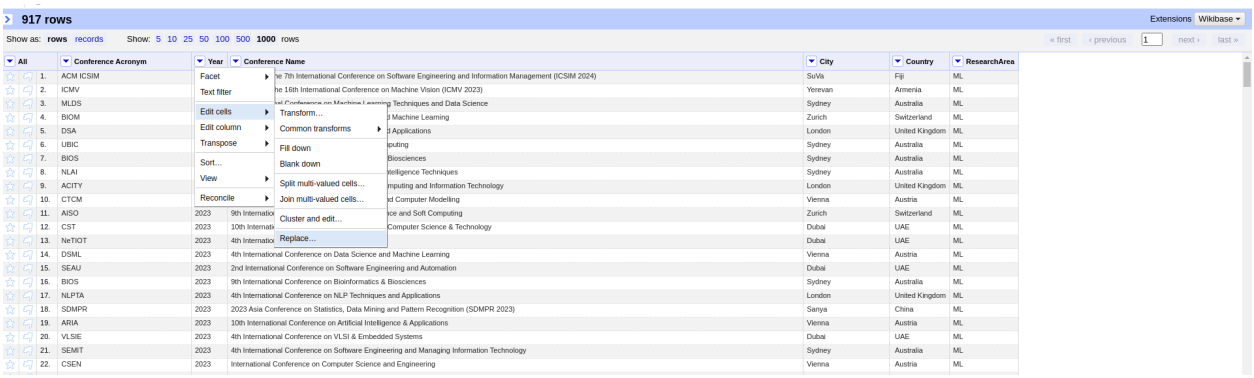
Result looks like, also renamed city and country as column names



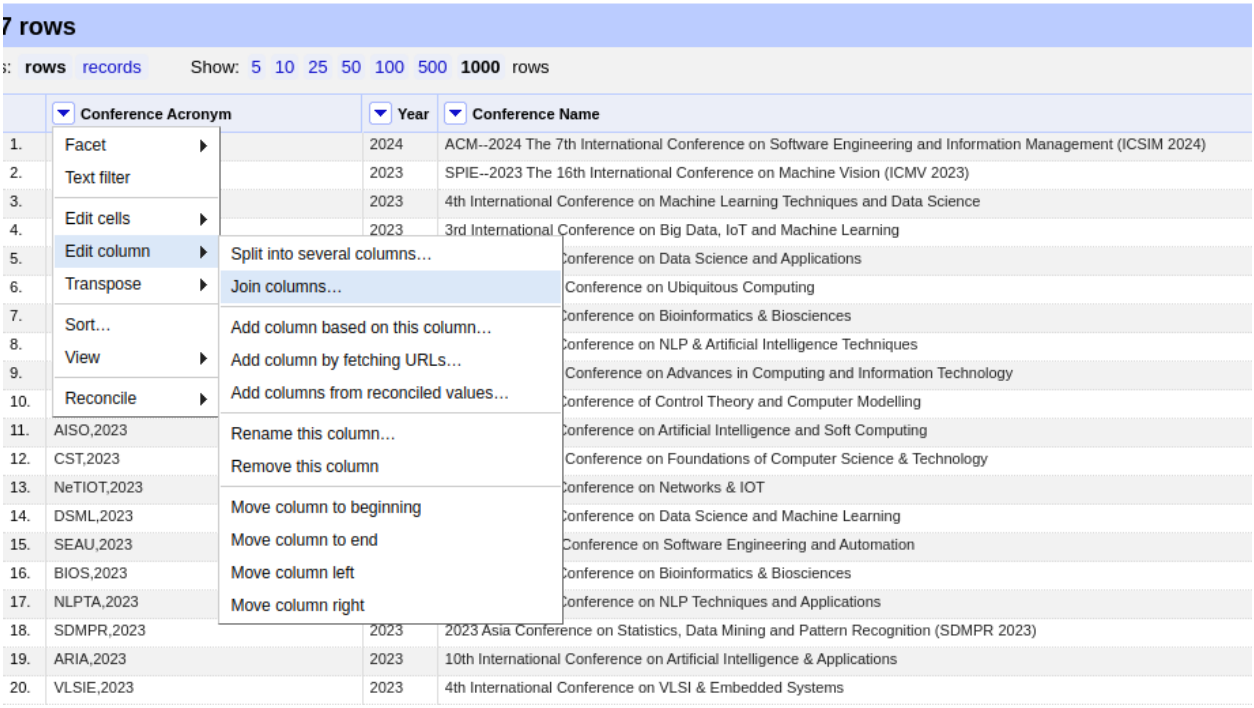
Splitted acronym in two columns, so that we have year separated from the acronym name



Once Splitting Done, I added formatting to the year by using replace and year now has only four digits resembling to its value



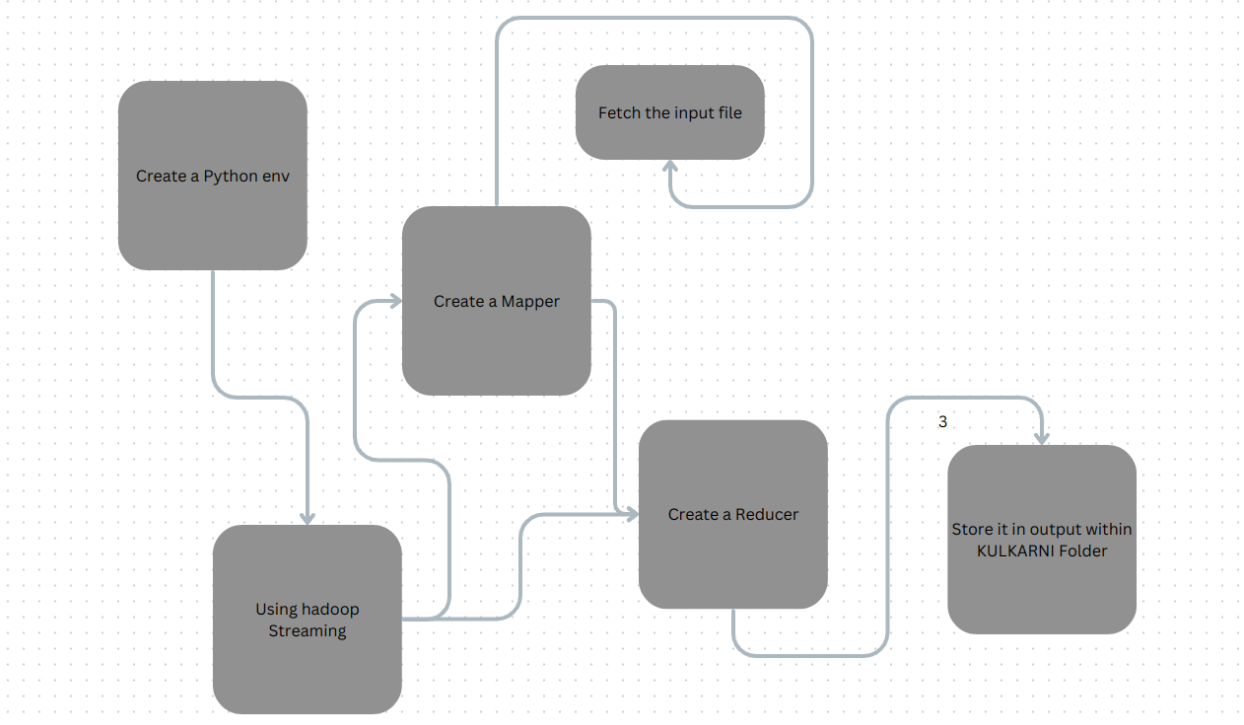
Used the join command to add the conferences with year and downloading this dataset for question 2



The downloaded file is tsv file and is stored in the downloads after we export it from OpenRefine. After a lot of data cleaning and manipulations, I was left with around 915 rows, which had proper data in it. As guided by the professor, I focused on removing N/A in locations, spaces leading/trailing, blank / null values, Special characters in name, Made city names same using clustering and few of them manually, Removed university names in locations, acronym splitted into year for processing, etc.

Exercise 3: Hadoop

A) Compute and plot the number of conferences per city. Which are the top 10 locations? (1 Mapper & 1 Reducer)



Initial command to run the mapper and reducer:

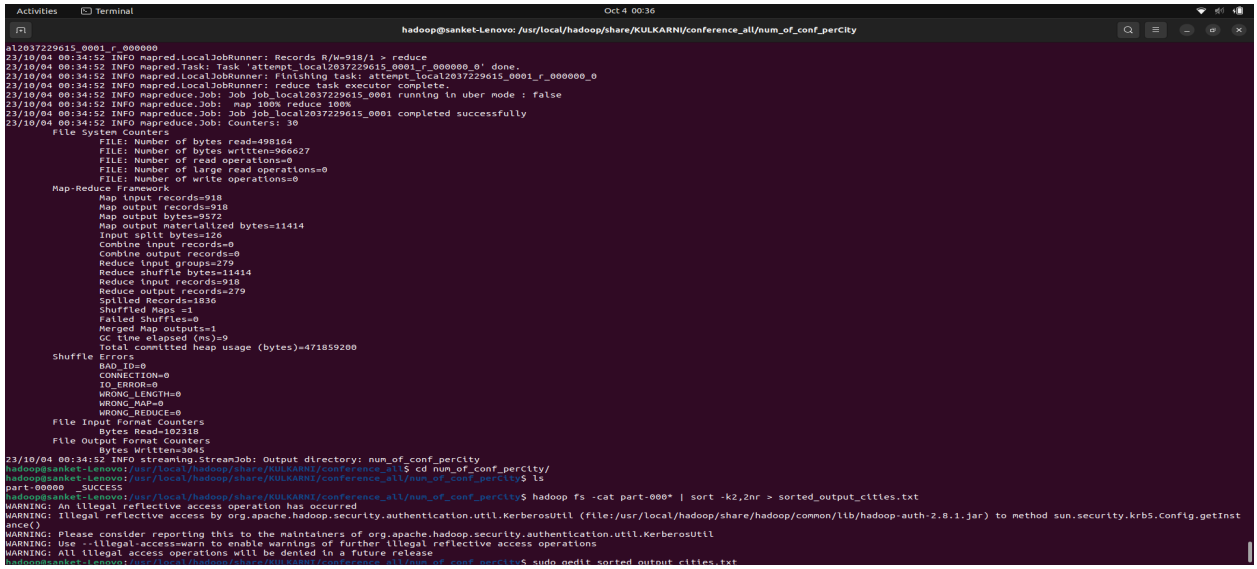
```
hadoop@sanket-Lenovo:/usr/local/hadoop/share/KULKARNI/conference_all$ ls
conference_aimlbd.tsv  num_conf_percity_mapper.py  num_conf_percity_reducer.py
hadoop@sanket-Lenovo:/usr/local/hadoop/share/KULKARNI/conference_all$ hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-*.jar \
-files num_conf_percity_mapper.py,num_conf_percity_reducer.py -mapper "python3 num_conf_percity_mapper.py" -reducer "python3 num_conf_percity_reducer.py" \
-input conference_aimlbd.tsv -output num_of_conf_perCity
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr/local/hadoop/share/hadoop/common/lib/hadoop-auth-2.8.1.jar) to method sun.security.krb5.Config.getInst
ance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.KerberosUtil
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
23/10/04 00:34:50 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.session-id
23/10/04 00:34:50 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
23/10/04 00:34:50 INFO jvm.JvmMetrics: Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
23/10/04 00:34:50 INFO mapred.FileInputFormat: Total input files to process : 1
23/10/04 00:34:50 INFO mapreduce.JobSubmitter: number of splits:1
23/10/04 00:34:51 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local2037229615_0001
23/10/04 00:34:51 INFO mapred.LocalDistributedCacheManager: Localized file:/usr/local/hadoop/share/KULKARNI/conference_all/num_conf_percity_mapper.py as file:/tmp/hadoop-hadoop/mapred/local/1696404891483/num_con
f_percity_mapper.py
23/10/04 00:34:51 INFO mapred.LocalDistributedCacheManager: Localized file:/usr/local/hadoop/share/KULKARNI/conference_all/num_conf_percity_reducer.py as file:/tmp/hadoop-hadoop/mapred/local/1696404891484/num_co
nf_percity_reducer.py
23/10/04 00:34:51 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
23/10/04 00:34:51 INFO mapreduce.Job: Running job: job_local2037229615_0001
23/10/04 00:34:51 INFO mapred.LocalJobRunner: OutputCommitter set in config null
23/10/04 00:34:51 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapred.FileOutputCommitter
23/10/04 00:34:51 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 1
23/10/04 00:34:51 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failures: false
23/10/04 00:34:51 INFO mapred.LocalJobRunner: Waiting for map tasks
23/10/04 00:34:51 INFO mapred.LocalJobRunner: Starting task: attempt_local2037229615_0001_m_000000_0
23/10/04 00:34:51 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 1
23/10/04 00:34:51 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failures: false
23/10/04 00:34:52 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
23/10/04 00:34:52 INFO mapred.MapTask: Processing split: file:/usr/local/hadoop/share/KULKARNI/conference_all/conference_aimlbd.tsv:0+102318
23/10/04 00:34:52 INFO mapred.MapTask: numReduceTasks: 1
23/10/04 00:34:52 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
23/10/04 00:34:52 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
23/10/04 00:34:52 INFO mapred.MapTask: soft limit at 83886080
23/10/04 00:34:52 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600
23/10/04 00:34:52 INFO mapred.MapTask: kvstart = 26214396; length = 6553600
23/10/04 00:34:52 INFO mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$MapOutputBuffer
23/10/04 00:34:52 INFO streaming.PipeMapRed: PipeMapRed exec [/usr/bin/python3, num_conf_percity_mapper.py]
23/10/04 00:34:52 INFO Configuration.deprecation: mapred.work.output.dir is deprecated. Instead, use mapreduce.task.output.dir
23/10/04 00:34:52 INFO Configuration.deprecation: map.input.start is deprecated. Instead, use mapreduce.map.input.start
23/10/04 00:34:52 INFO Configuration.deprecation: mapred.task.is.map is deprecated. Instead, use mapreduce.task.ismap
23/10/04 00:34:52 INFO Configuration.deprecation: mapred.task.id is deprecated. Instead, use mapreduce.task.attempt.id
23/10/04 00:34:52 INFO Configuration.deprecation: mapred.tip.id is deprecated. Instead, use mapreduce.task.id
23/10/04 00:34:52 INFO Configuration.deprecation: mapred.local.dir is deprecated. Instead, use mapreduce.cluster.local.dir
23/10/04 00:34:52 INFO Configuration.deprecation: map.input.file is deprecated. Instead, use mapreduce.map.input.file
23/10/04 00:34:52 INFO Configuration.deprecation: mapred.skip.on is deprecated. Instead, use mapreduce.job.skiprecords
23/10/04 00:34:52 INFO Configuration.deprecation: map.input.length is deprecated. Instead, use mapreduce.map.input.length
23/10/04 00:34:52 INFO Configuration.deprecation: mapred.job.id is deprecated. Instead, use mapreduce.job.id
23/10/04 00:34:52 INFO Configuration.deprecation: user.name is deprecated. Instead, use mapreduce.job.user.name
```

Command :

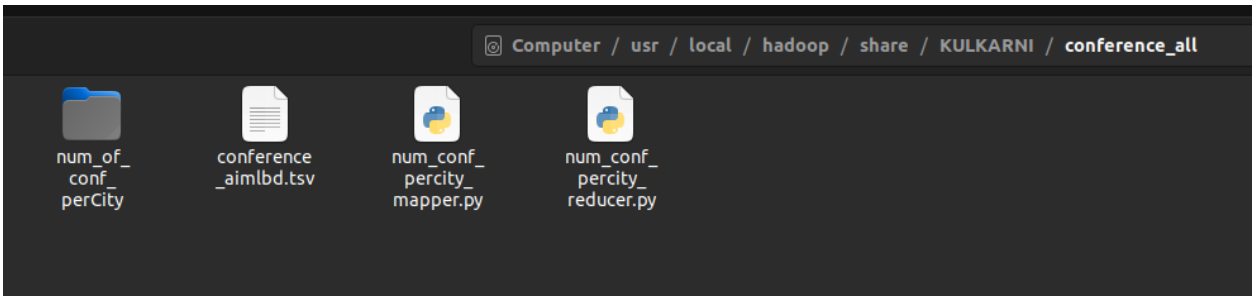
Sanket Kulkarni : HW2

```
hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-*.jar \
-files num_conf_percity_mapper.py,num_conf_percity_reducer.py -mapper "python3
num_conf_percity_mapper.py" -reducer "python3 num_conf_percity_reducer.py" \
-input conference_aimlbd.tsv -output num_of_conf_perCity
```

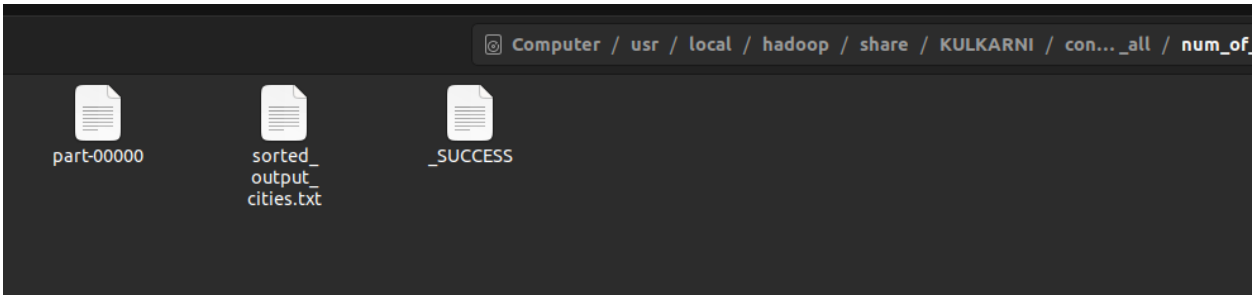
Successfully ran screenshot:



Folder Structure where output is generated and program was ran:



Output Location:



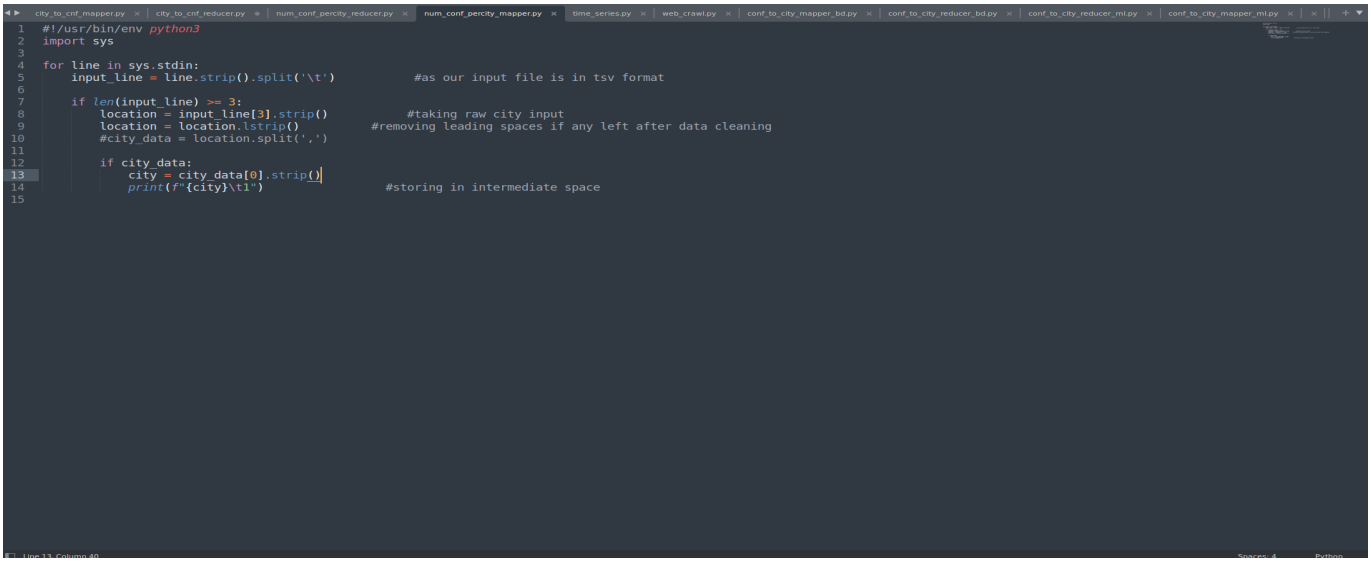
Sorted output:

Top 10 locations are:

Sr. NO	City Name	Count
1	Singapore	31
2	Sydney	30
3	Chengdu	25
4	Beijing	23
5	Bangkok	22
6	Zurich	22
7	Abu Dhabi	21
8	Shanghai	20
9	Laguna Hills	18
10	Dubai	17

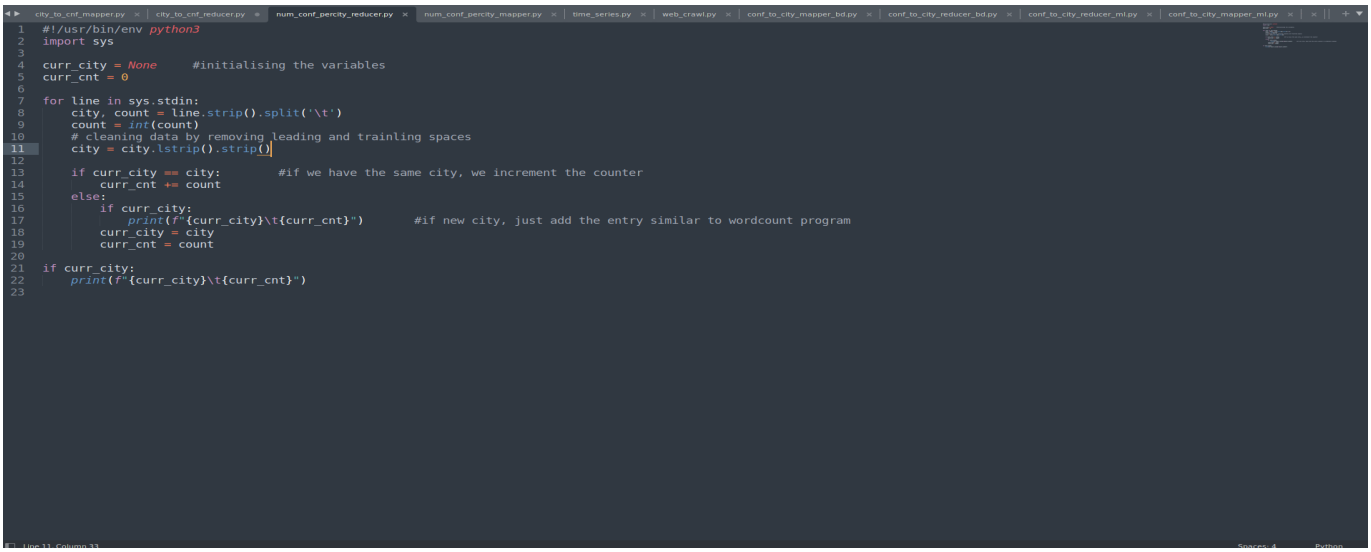
Code Snippet:
Mapper

Sanket Kulkarni : HW2



```python
#!/usr/bin/env python3
import sys

for line in sys.stdin:
    input_line = line.strip().split('\t')           #as our input file is in tsv format

    if len(input_line) >= 3:
        location = input_line[3].strip()            #taking raw city input
        location = location.lstrip()            #removing leading spaces if any left after data cleaning
        #city_data = location.split(',')

        if city_data:
            city = city_data[0].strip()
            print(f"{city}\t1")                 #storing in intermediate space

```

Reducer:

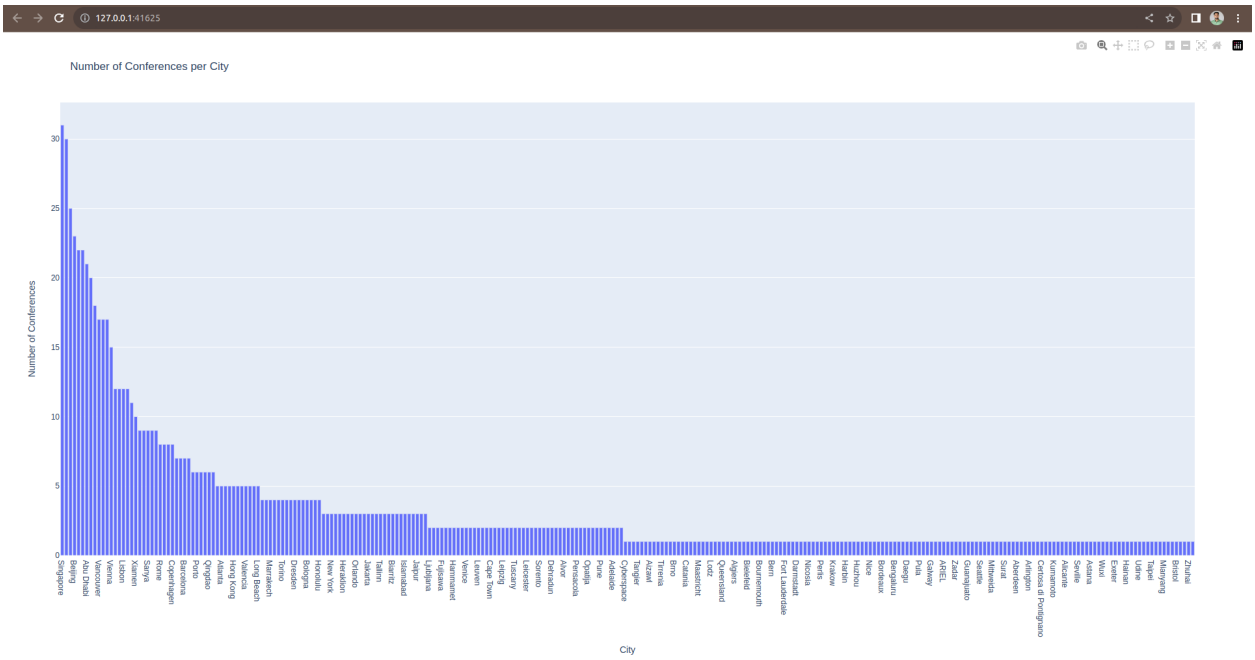


```python
#!/usr/bin/env python3
import sys

curr_city = None     #initialising the variables
curr_cnt = 0

for line in sys.stdin:
    city, count = line.strip().split('\t')
    count = int(count)
    # cleaning data by removing leading and trainling spaces
    city = city.lstrip().strip()

    if curr_city == city:        #if we have the same city, we increment the counter
        curr_cnt += count
    else:
        if curr_city:
            print(f"{curr_city}\t{curr_cnt}")       #if new city, just add the entry similar to wordcount program
        curr_city = city
        curr_cnt = count

if curr_city:
    print(f"{curr_city}\t{curr_cnt}")

```

Plotted the graph for the number of conferences per city using plotly library. Have made the graph in descending order so that it visualizes properly.
Code Snippet:

```python
import pandas as pd
import plotly.express as px

# Loading the TSV file into a DataFrame
df = pd.read_csv('conference_aimlbd.tsv', sep='\t')

# Counting the city occurences
city_conf_counts = df['City'].value_counts().reset_index()
city_conf_counts.columns = ['City', 'Number of Conferences']

# Createing a bar graph using Plotly
fig = px.bar(city_conf_counts, x='City', y='Number of Conferences', title='Number of Conferences per City')
fig.update_xaxes(title='City')
fig.update_yaxes(title='Number of Conferences')

# Show the plot
fig.show()
```

Number of conferences per city



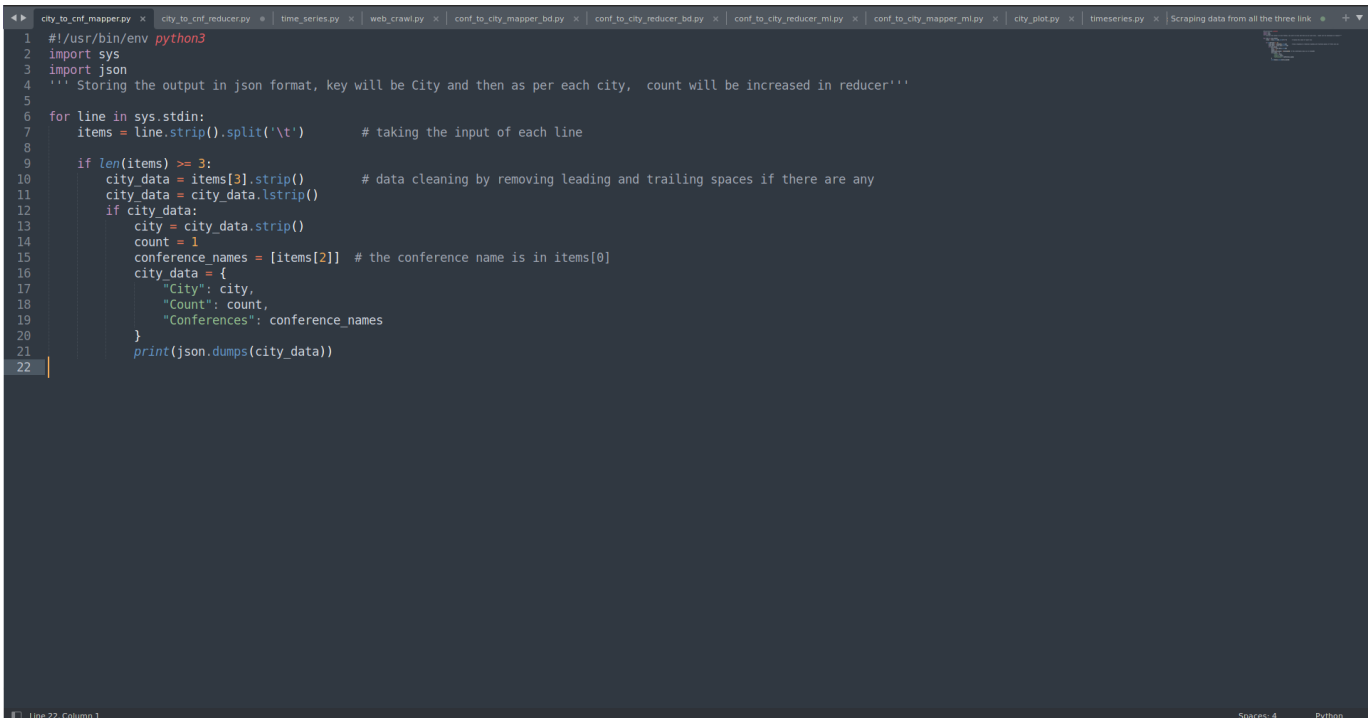
If we hover over the graph we can see all the cities with their corresponding count.

B) Output the list of conferences per city

Steps:

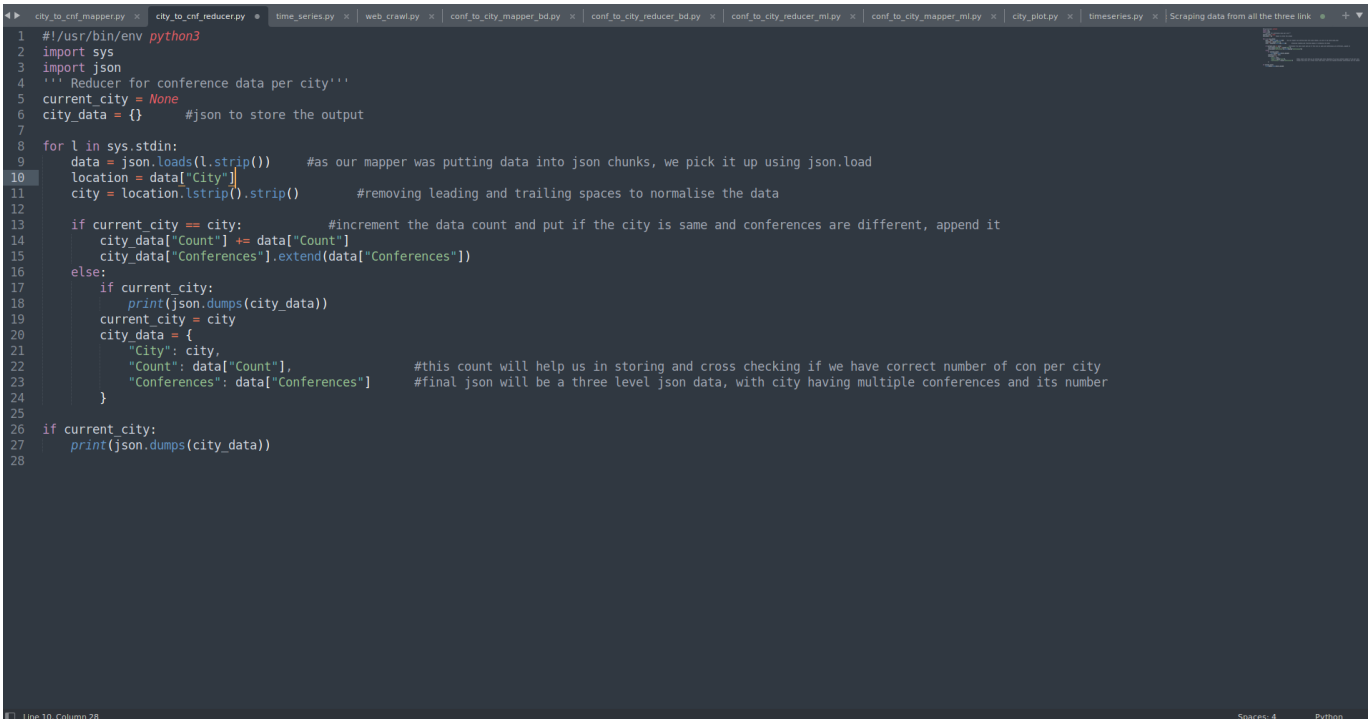
- Take the input line by line
- Remove leading and trailing spaces if any
- Consider City and count and conference name inside the mapper that we will put in the intermediate space
- In the reducer, load the line data and as it was in json format while storing, check for the attributes inside it. Specifically we will look towards location here.
- If our current city is equal to city, we increment count of that city as well as extend the conference name in that list where city will be our key
- Finally, put this entire json inside the output file

Mapper Snippet:



```python
#!/usr/bin/env python3
import sys
import json
''' Storing the output in json format, key will be City and then as per each city,  count will be increased in reducer'''

for line in sys.stdin:
    items = line.strip().split('\t')        # taking the input of each line

    if len(items) >= 3:
        city_data = items[3].strip()        # data cleaning by removing leading and trailing spaces if there are any
        city_data = city_data.lstrip()
        if city_data:
            city = city_data.strip()
            count = 1
            conference_names = [items[2]]  # the conference name is in items[0]
            city_data = {
                "City": city,
                "Count": count,
                "Conferences": conference_names
            }
            print(json.dumps(city_data))
```

Reducer Snippet:

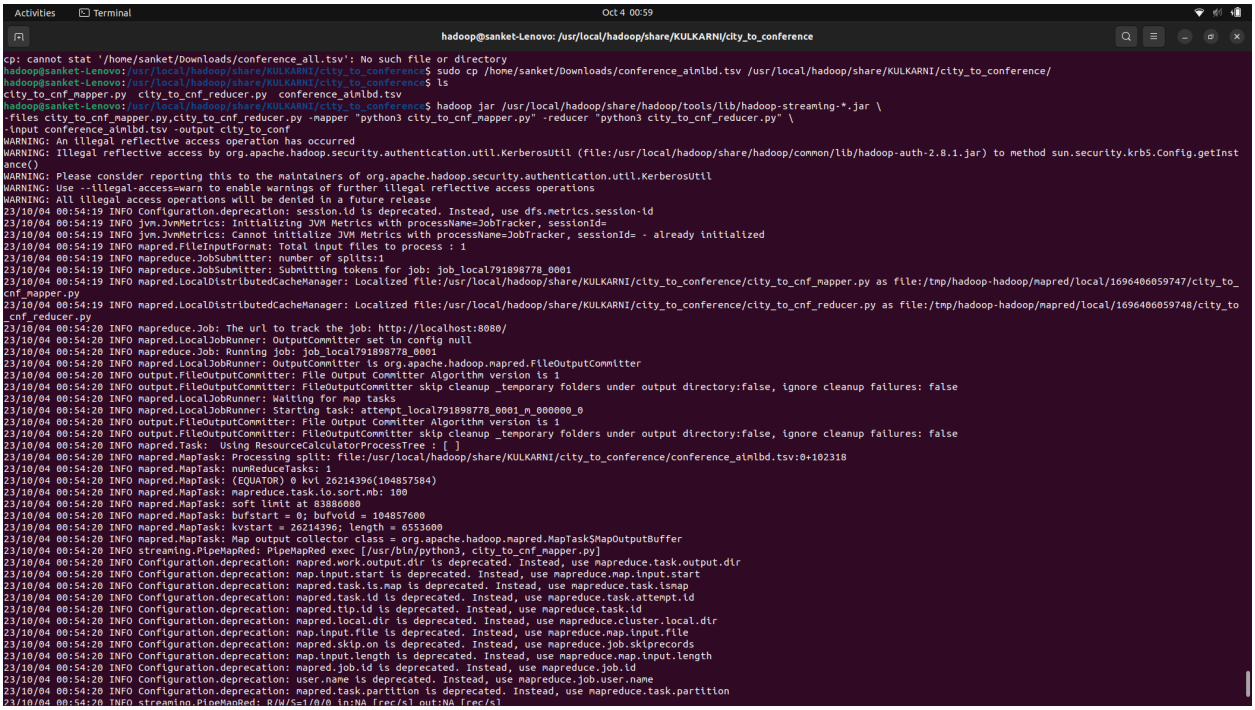


```python
#!/usr/bin/env python3
import sys
import json
''' Reducer for conference data per city'''
current_city = None
city_data = {}      #json to store the output

for l in sys.stdin:
    data = json.loads(l.strip())     #as our mapper was putting data into json chunks, we pick it up using json.load
    location = data["City"]
    city = location.lstrip().strip()        #removing leading and trailing spaces to normalise the data

    if current_city == city:            #increment the data count and put if the city is same and conferences are different, append it
        city_data["Count"] += data["Count"]
        city_data["Conferences"].extend(data["Conferences"])
    else:
        if current_city:
            print(json.dumps(city_data))
        current_city = city
        city_data = {
            "City": city,
            "Count": data["Count"],                 #this count will help us in storing and cross checking if we have correct number of con per city
            "Conferences": data["Conferences"]      #final json will be a three level json data, with city having multiple conferences and its number
        }

if current_city:
    print(json.dumps(city_data))
```

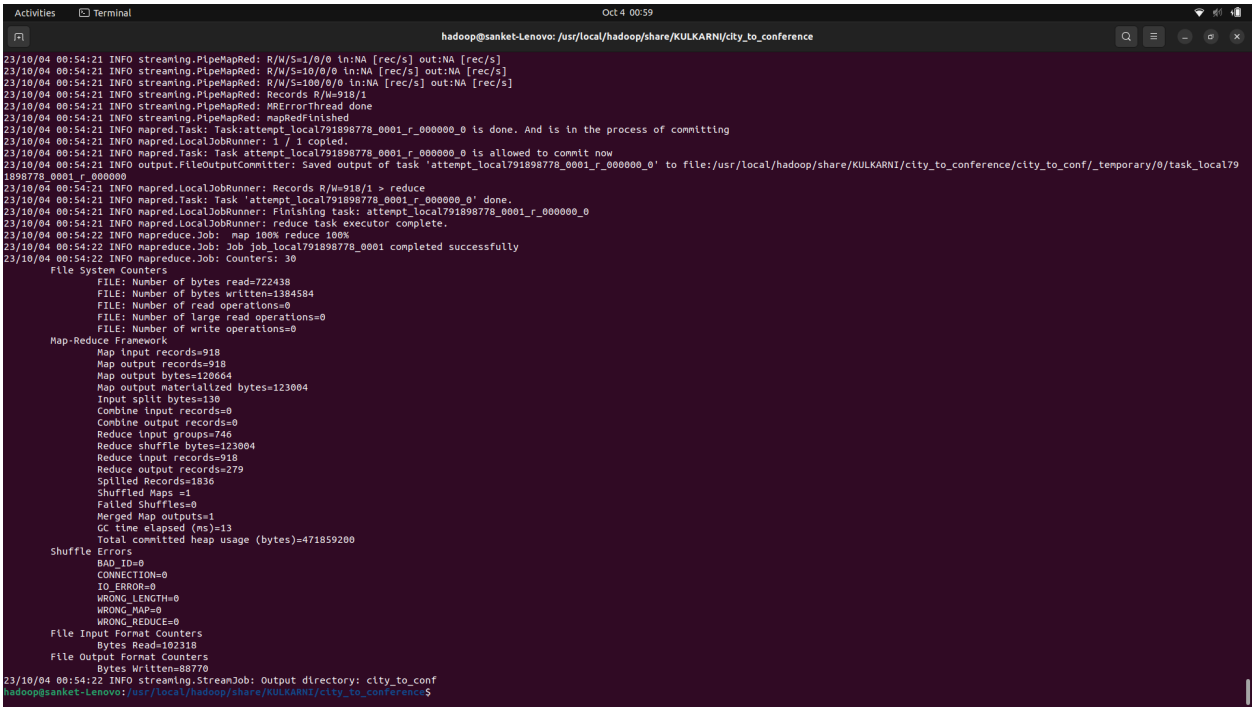
Program Running command:

```
hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-*.jar \
-files city_to_cnf_mapper.py,city_to_cnf_reducer.py -mapper "python3 city_to_cnf_mapper.py" -reducer "python3 city_to_cnf_reducer.py" \
-input conference_aimlbd.tsv -output city_to_cnf
```

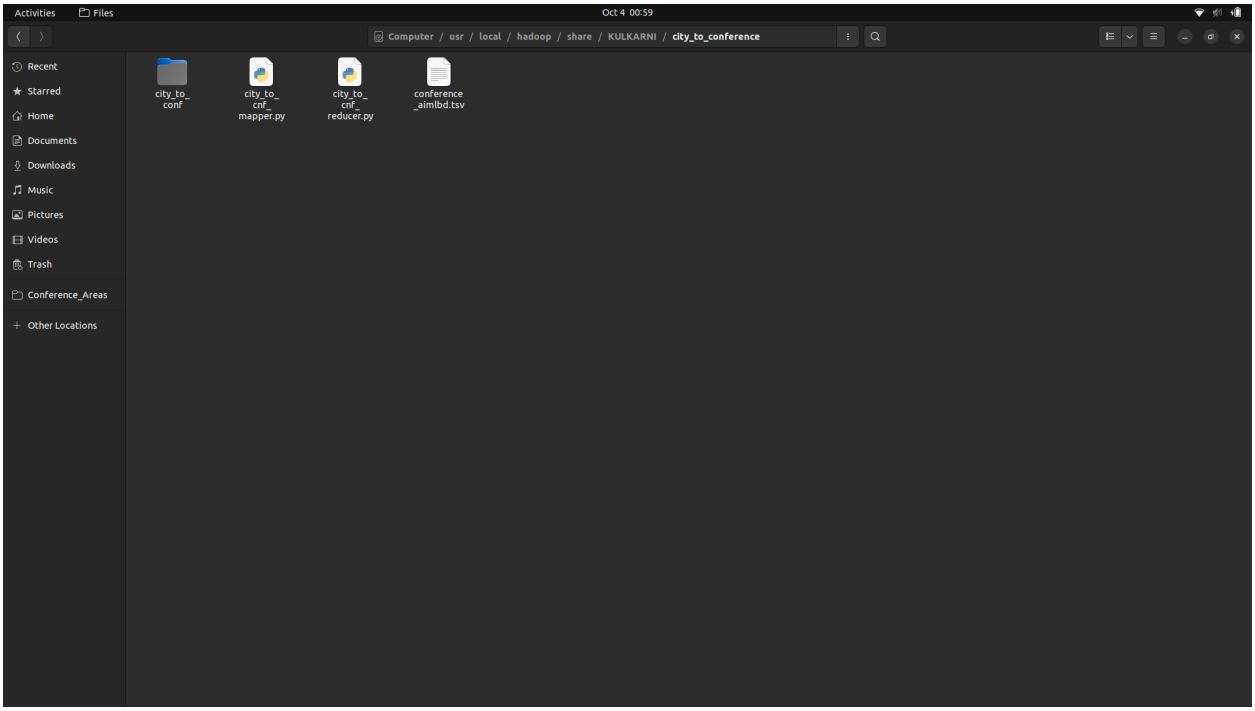
Sanket Kulkarni : HW2



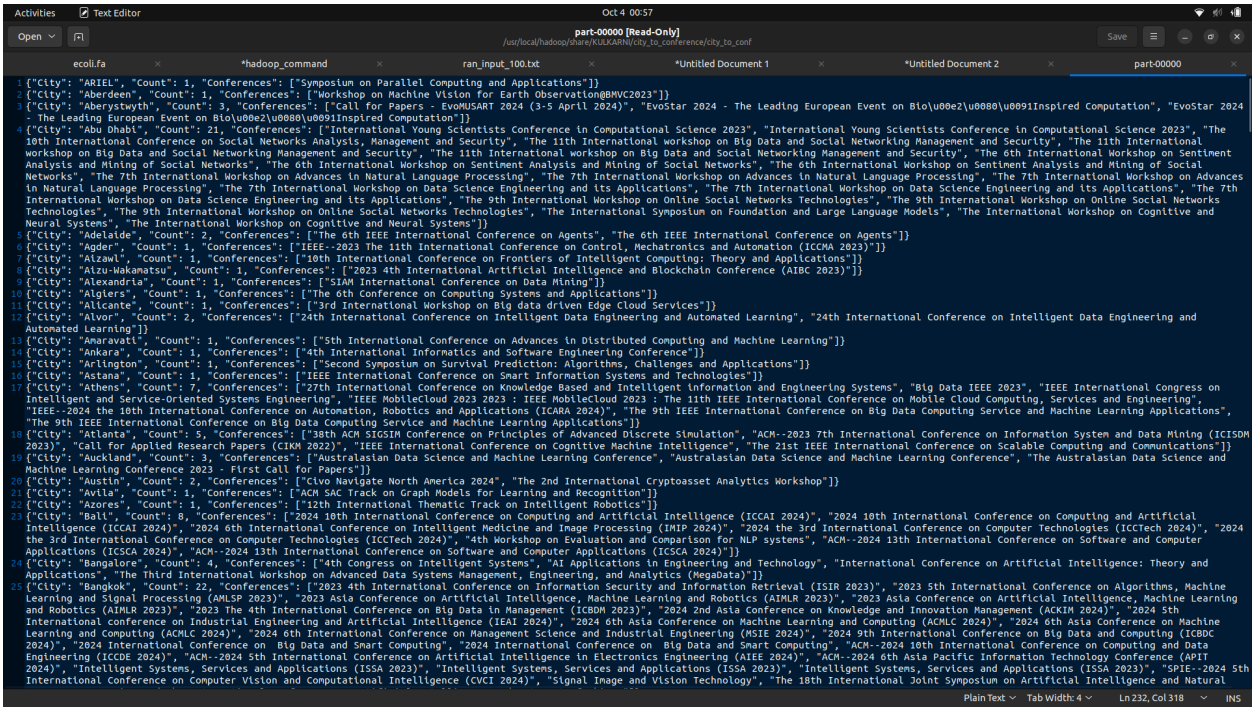
Program Completion event:



Directory Looks like:



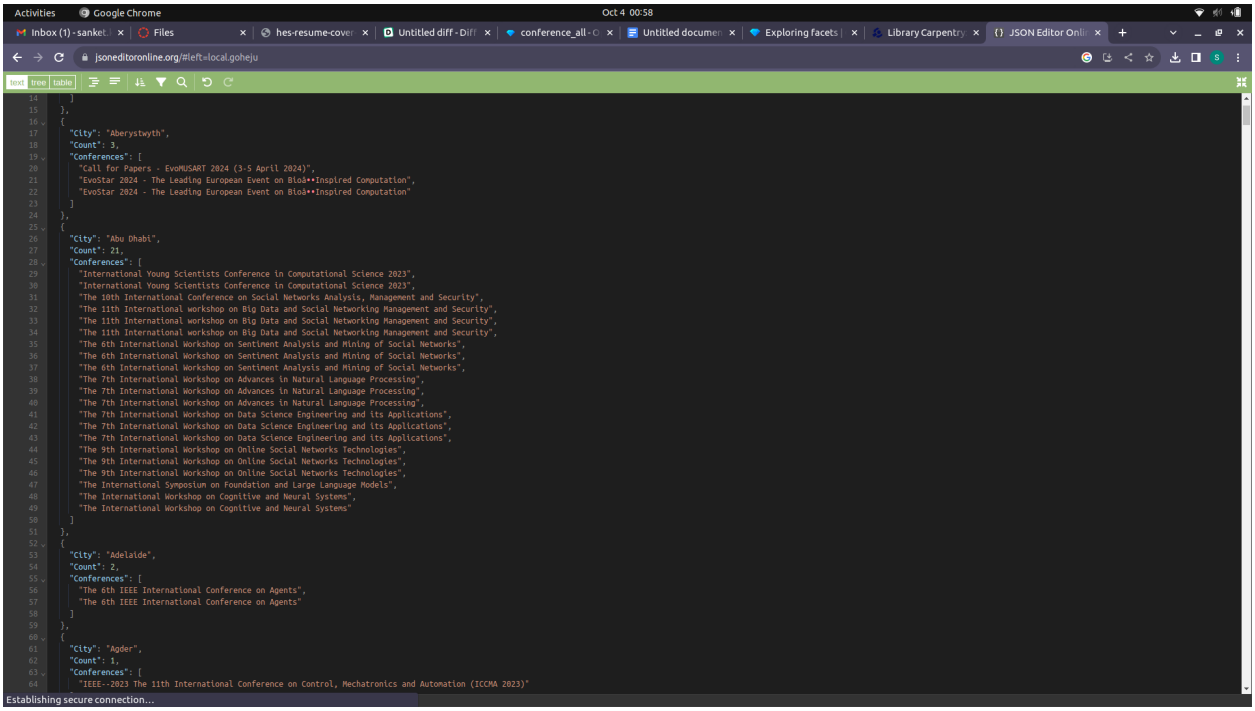
Output file is stored in a JSON format for better understanding:



JSon cross verifying Singapore with 31 conferences which matches our earlier output


```
Machine Learning and Computing (ICMLC 2024)"]}
{"City": "Singapore", "Count": 31, "Conferences": ["2022 IEEE Symposium Series on Computational Intelligence", "2023 5th International Conference on Big Data Engineering and Technology (BDET 2023)", "2023 5th International Conference on Software Engineering and Development (ICSED 2023)", "2023 Asia Conference on Big Data, Information and System Security (BDISS 2023) -EI Compendex", "2023 Asia Conference on Communications, Signal Processing and Information Technology (CSPIT 2023) -EI Compendex", "2023 Asia Conference on Communications, Signal Processing and Information Technology (CSPIT 2023) -EI Compendex", "2023 Asia Conference on Communications, Signal Processing and Information Technology (CSPIT 2023) -EI Compendex", "2023 International Conference on Big Data Analysis and Applicatcion (ICBDAA 2023)", "2023 International Conference on Data, Information and Computing Science (CDICS 2023)", "2023 The 11th International Conference on Information Technology and Science (ICITS 2023)", "2023 The 6th International Conference on Mechanical, System and Control Engineering (ICMSC 2023)", "2023 The 9th International Conference on Data Processing and Applications (ICDPA 2023)", "2023 The 9th International Conference on Digital Manufacturing and Automation (ICDMA 2023)", "2023 the 9th International Conference on Mechatronics System and Robots (ICMSR 2023)", "2024 10th International Conference on Control, Automation and Robotics (ICCAR 2024)", "2024 5th International Conference on Supply Chain Management (ICSCM 2024)", "2024 8th International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence (ISMSI 2024)", "2024 8th International Conference on Machine Learning and Soft Computing (ICMLSC 2024)", "2024 8th International Conference on Machine Learning and Soft Computing (ICMLSC 2024)", "2024 International Workshop on Control Sciences and Automation (WCSA 2024)", "2024 The 2nd International Conference on Cloud Computing and Computer Network (CCCN 2024)", "ACM--2023 5th International Conference on Big-data Service and Intelligent Computation (BDSIC 2023)", "ACM--2023 5th International Conference on Big-data Service and Intelligent Computation (BDSIC 2023)", "ACM--2023 9th International Conference on Robotics and Artificial Intelligence (ICRAI 2023)", "ACM--2024 The 7th International Conference on Machine Vision and Applications (ICMVA 2024)", "ACM--2024 The 7th International Conference on Machine Vision and Applications (ICMVA 2024)", "Automated Technology for Verification and Analysis", "IEEE 6th International Conference on Multimedia Information Processing and Retrieval", "IEEE Conference on Artificial Intelligence", "IEEE--2023 8th International Conference on Robotics and Automation Engineering (ICRAE 2023)", "IEEE--2023 the 3rd International Conference on Robotics, Automation and Artificial Intelligence (RAAI 2023)"]}
```


Json with Json viewer:




```
  ]
 },
 {
  "City": "Aberystwyth",
  "Count": 3,
  "Conferences": [
   "Call for Papers - EvoMUSART 2024 (3-5 April 2024)",
   "EvoStar 2024 - The Leading European Event on Bioà••Inspired Computation",
   "EvoStar 2024 - The Leading European Event on Bioà••Inspired Computation"
  ]
 },
 {
  "City": "Abu Dhabi",
  "Count": 21,
  "Conferences": [
   "International Young Scientists Conference in Computational Science 2023",
   "International Young Scientists Conference in Computational Science 2023",
   "The 10th International Conference on Social Networks Analysis, Management and Security",
   "The 11th International workshop on Big Data and Social Networking Management and Security",
   "The 11th International workshop on Big Data and Social Networking Management and Security",
   "The 11th International workshop on Big Data and Social Networking Management and Security",
   "The 6th International Workshop on Sentiment Analysis and Mining of Social Networks",
   "The 6th International Workshop on Sentiment Analysis and Mining of Social Networks",
   "The 6th International Workshop on Sentiment Analysis and Mining of Social Networks",
   "The 7th International Workshop on Advances in Natural Language Processing",
   "The 7th International Workshop on Advances in Natural Language Processing",
   "The 7th International Workshop on Advances in Natural Language Processing",
   "The 7th International Workshop on Data Science Engineering and its Applications",
   "The 7th International Workshop on Data Science Engineering and its Applications",
   "The 7th International Workshop on Data Science Engineering and its Applications",
   "The 9th International workshop on Online Social Networks Technologies",
   "The 9th International workshop on Online Social Networks Technologies",
   "The 9th International workshop on Online Social Networks Technologies",
   "The International Symposium on Foundation and Large Language Models",
   "The International Workshop on Cognitive and Neural Systems",
   "The International Workshop on Cognitive and Neural Systems"
  ]
 },
 {
  "City": "Adelaide",
  "Count": 2,
  "Conferences": [
   "The 6th IEEE International Conference on Agents",
   "The 6th IEEE International Conference on Agents"
  ]
 },
 {
  "City": "Agder",
  "Count": 1,
  "Conferences": [
   "IEEE--2023 The 11th International Conference on Control, Mechatronics and Automation (ICCMA 2023)"
```

C) For each conference regardless of the year (e.g., Big Data, ML, AI), output the list of cities. (1 Mapper & 1 Reducer for each research field)

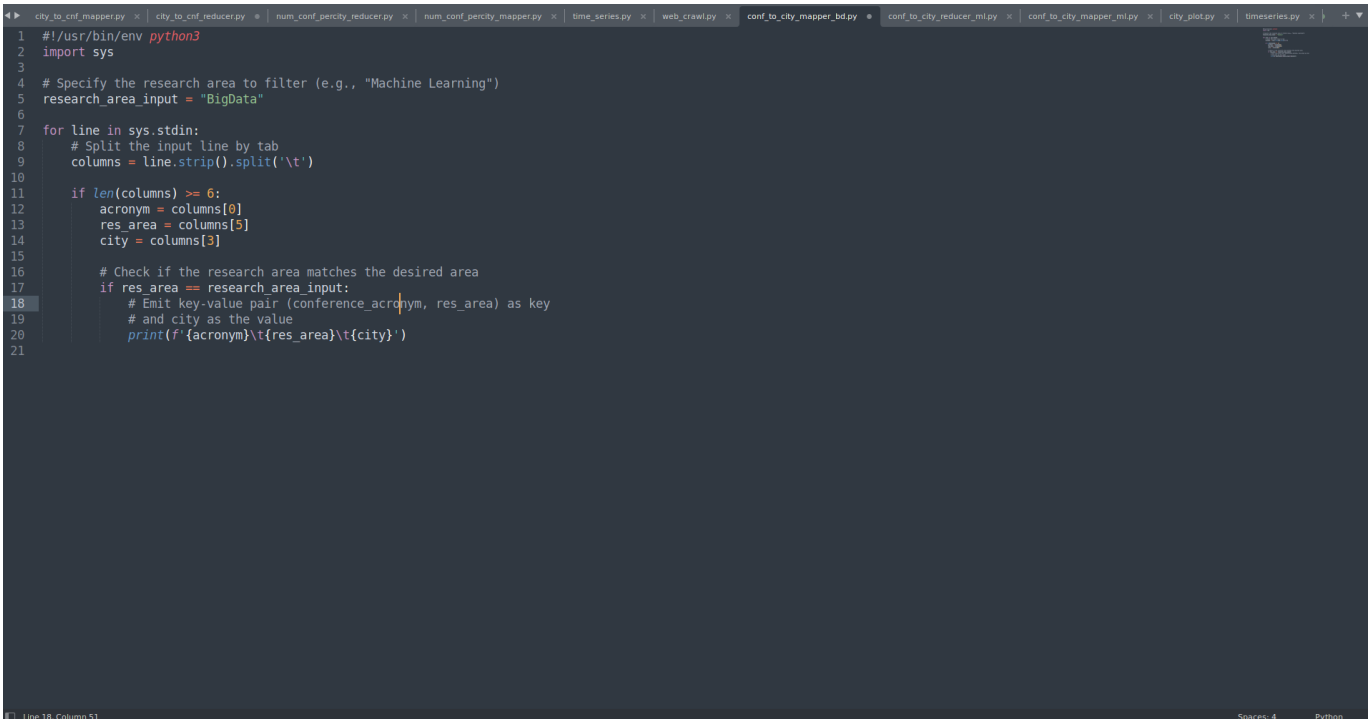
Approach :

- As I had collected data of all three research areas in one tsv only, I had mapped these with respective research areas with an extra column, so that we have data about what research area the conference is from
- Created 3 mappers and 3 reducers all of which belong to respective research areas.
- So we will have 3 output files belonging to each of the research areas as well.

Steps:

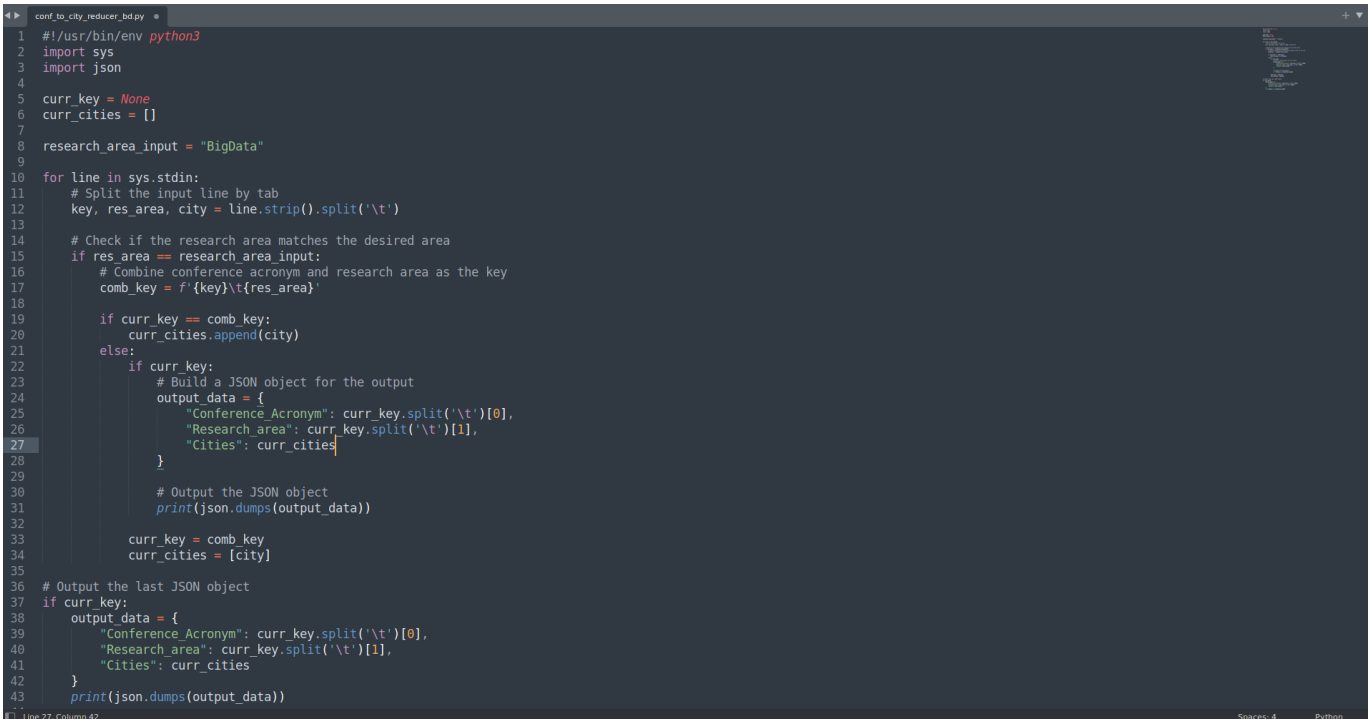
- Create a mapper which will capture the research area we need to run it for.
- I am making the key based on Acronym of the conference, because that will only be the constant value
- Emitting key value pair for acronym, research area as the key and city as the value
- Create a reducer, split the input line we got from the intermediate space.
- Made a combination key of acronym clubbed with research area
- If current key matches the combined key, we append it to the list else we create a new JSON object and next time we append it n that json object if the key matches
- Finally, put this json in output format

Mapper Snippet:



Sanket Kulkarni : HW2

Reducer snippet:

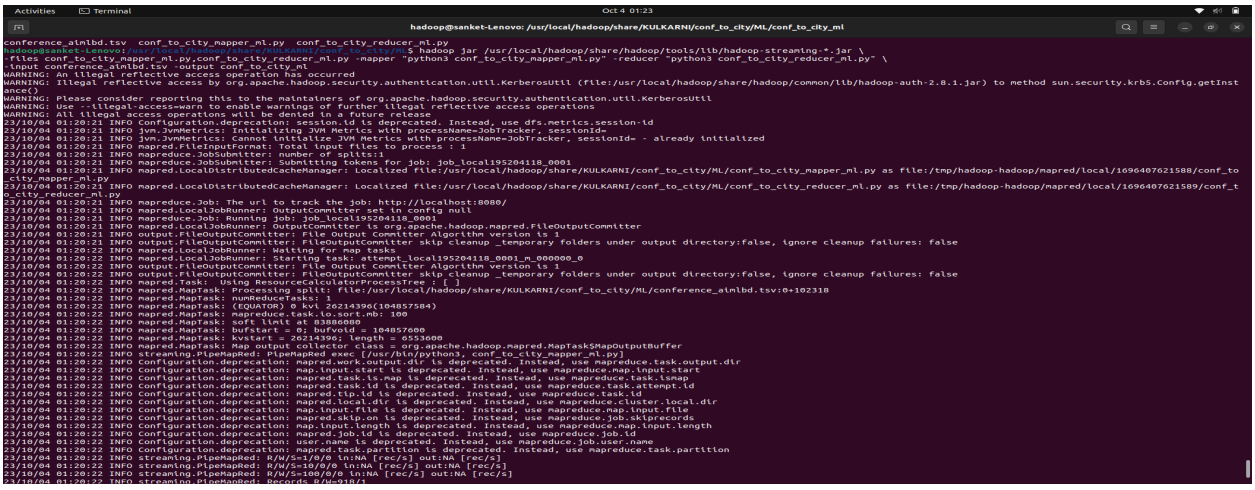


Similar mappers and reducers will be used for ML, AI. Just we need to change the research_area_input

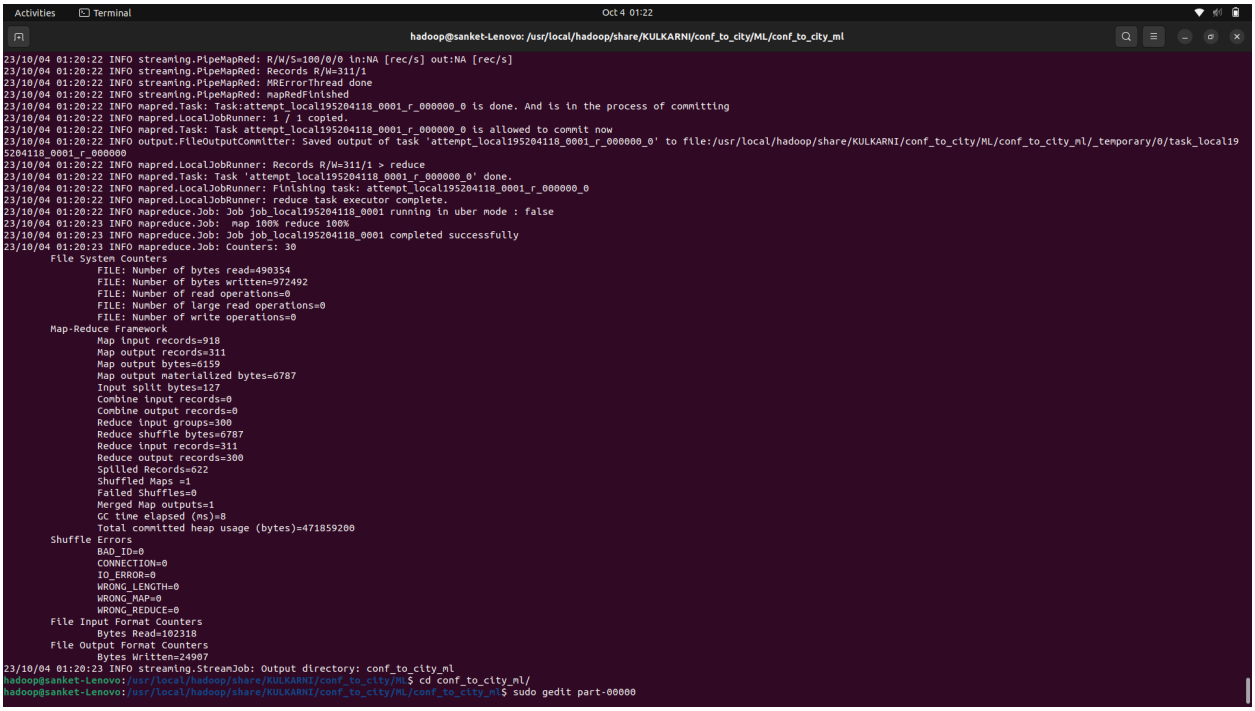
Command: Similar command for AI and Big data, just change the file names

```
hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-*.jar \
-files conf_to_city_mapper_ml.py,conf_to_city_reducer_ml.py -mapper "python3
conf_to_city_mapper_ml.py" -reducer "python3 conf_to_city_reducer_ml.py" \
-input conference_aimlbd.tsv -output conf_to_city_ml
```

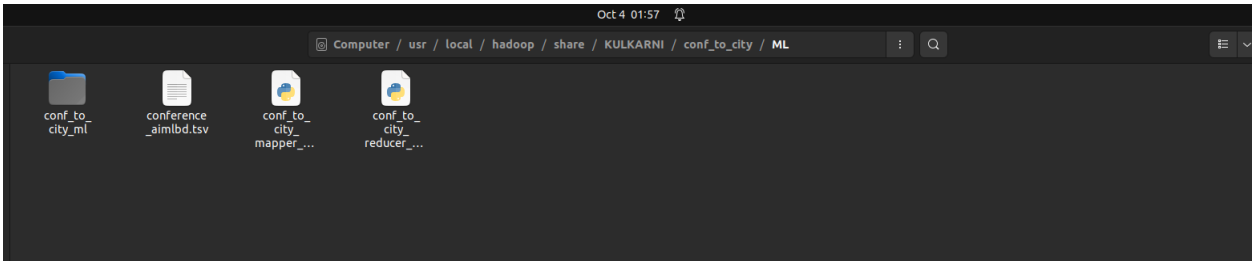
Code execution for ML:



Code Successfully executed ML:



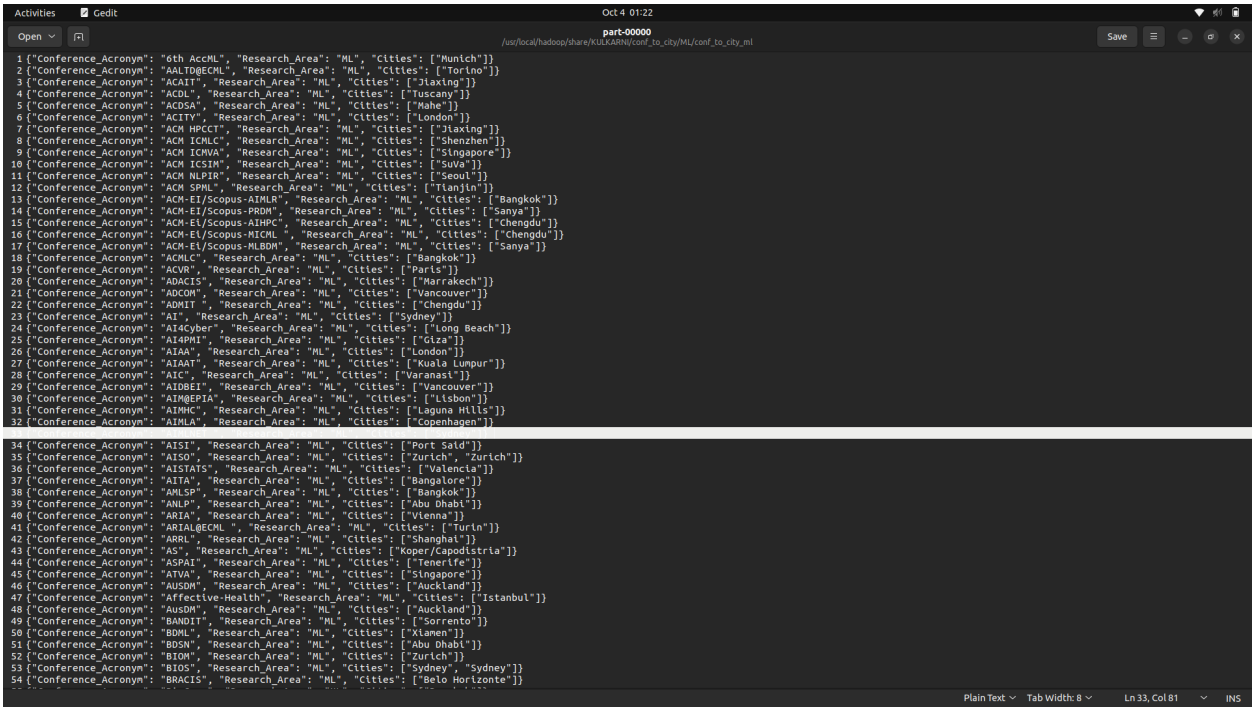
Directory Structure:



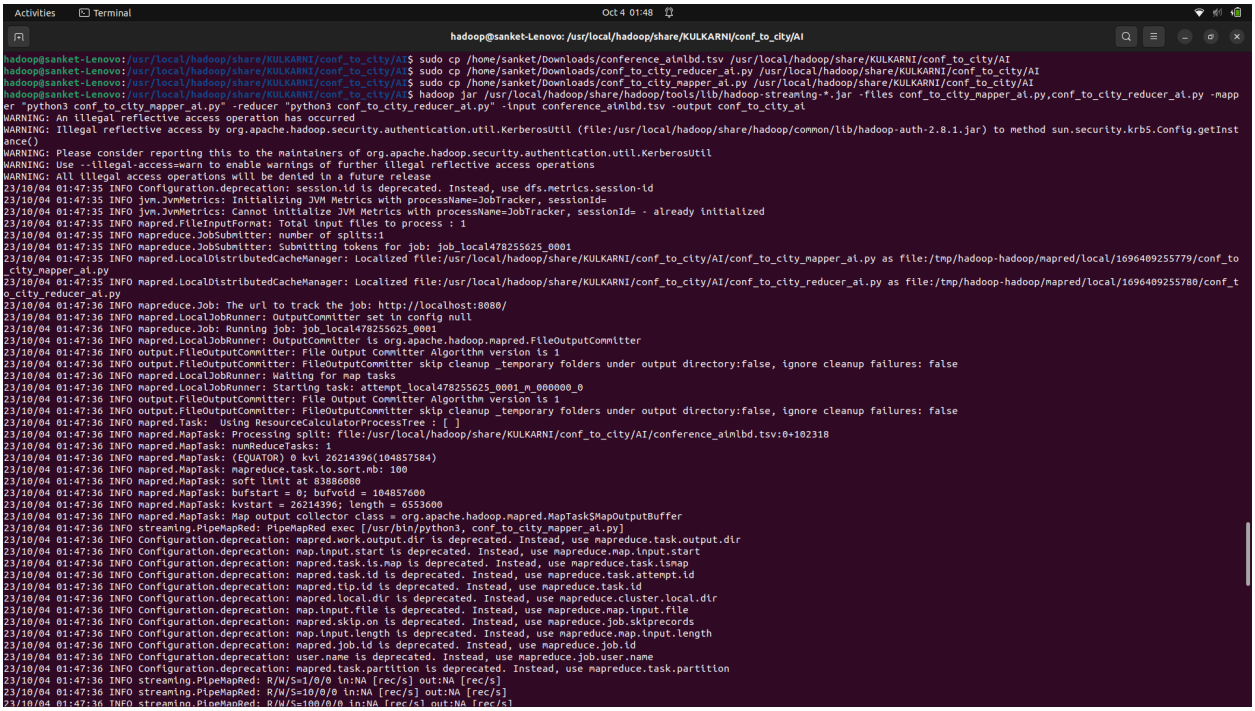
‘

Sanket Kulkarni : HW2

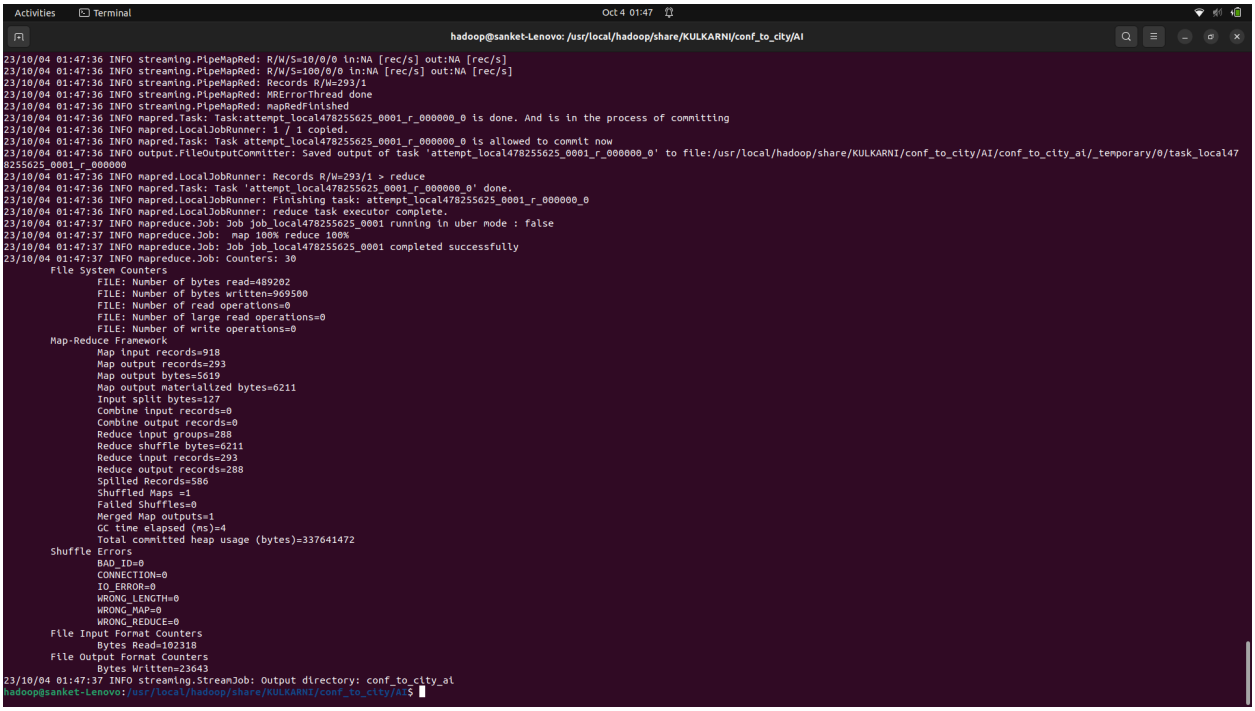
Output for ML as research field:



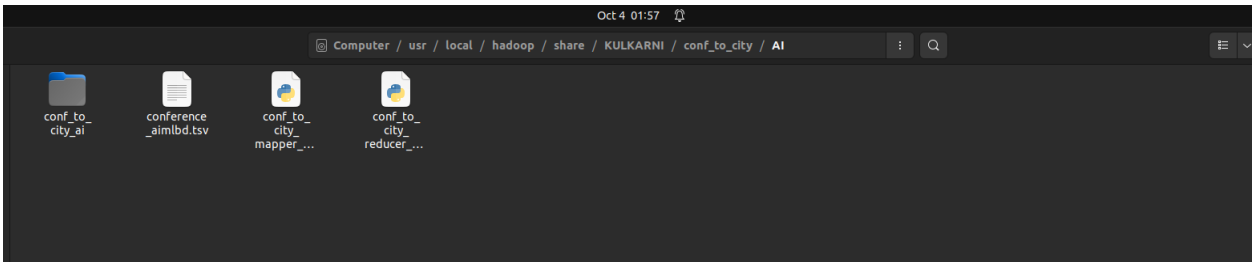
Code Execution AI:



Code Execution Successfully AI:

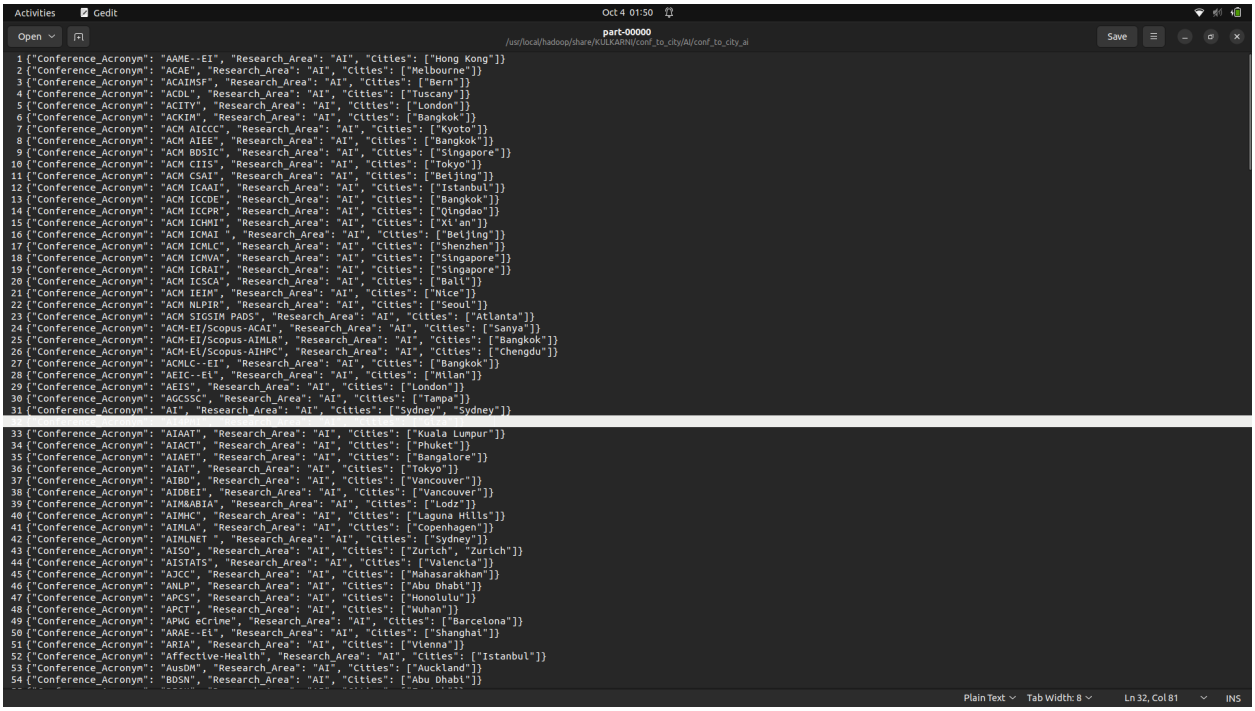


Folder structure for AI:

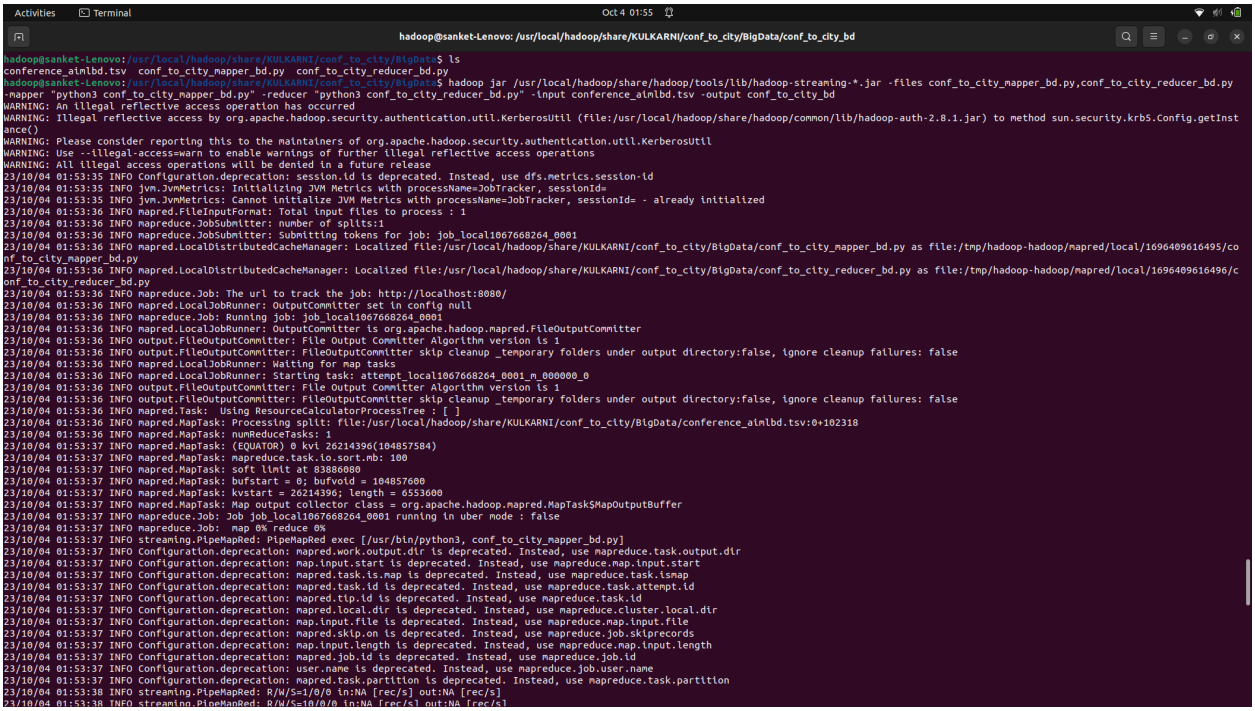


Sanket Kulkarni : HW2

Output in Json format for AI:

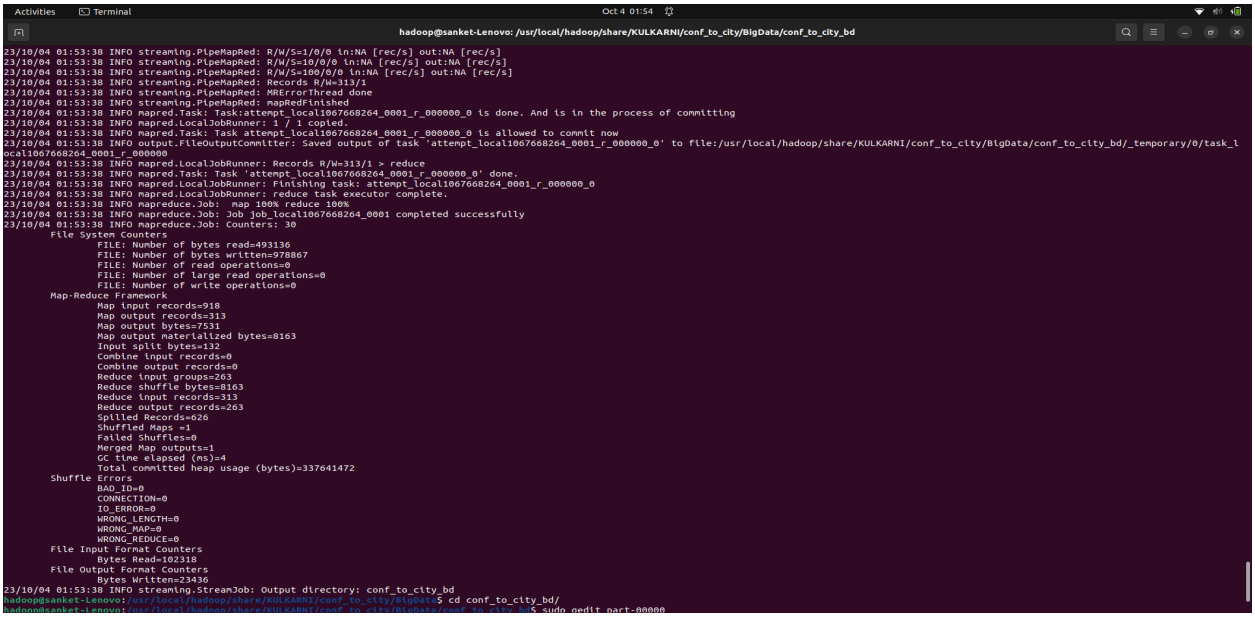


Code Execution of Big Data:

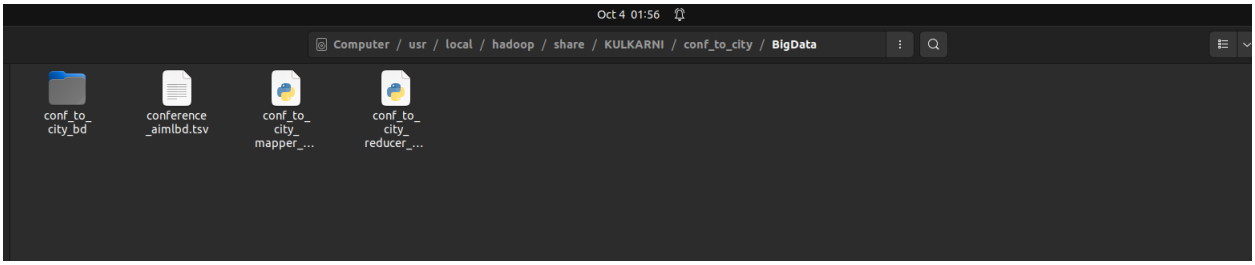


Sanket Kulkarni : HW2

Code execution successful for Big Data:



Folder structure for bigdata:



Output for Big Data:

D) For each city compute and plot a time series of #conferences per year

Approach:

- Create a mapper which will make a key based on the city and year column from the input tsv file.
- In the reducer, keep a count of the number of times city is repeated
- Additionally, after the output is generated we need to plot the graph
- Use plotly to plot the graph, here keep city year on the x axis and count on the y axis
- Once graph is plotted take screenshots with zoom in as well

Code Snippets:

Mapper:

```python
#!/usr/bin/env python3
import sys

# Process each line from standard input
for line in sys.stdin:
    # splitting the input line to make a tuple below
    input_line = line.strip().split('\t')
    if len(input_line) >= 3:
        location = input_line[3].strip()        #taking raw city input
        year =  input_line[1].strip()         #taking raw year input
    # intermediate space printing for reducer
    print(f"{location} {year}\t1")
```

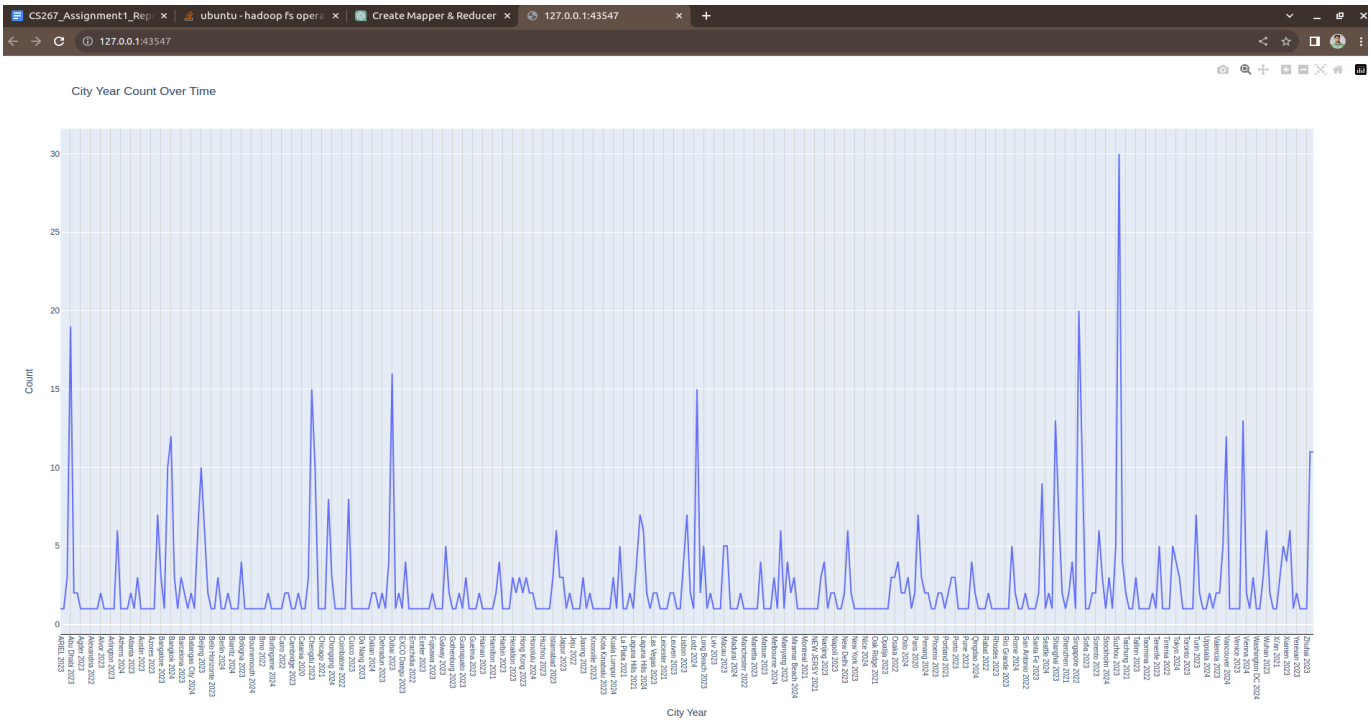
Reducer

```python
#!/usr/bin/env python
import sys

current_key = None
current_count = 0

# Process each line from standard input
for line in sys.stdin:
    # Split the line into key and count
    key, count = line.strip().split('\t')

    # Convert count to an integer
    count = int(count)

    # If the current key is None, or if it's different from the previous key
    if current_key is None or key != current_key:
        # If we have a current key (i.e., not the first key), output its count
        if current_key is not None:
            print(f"{current_key}\t{current_count}")
        # Set the current key and initialize the count
        current_key = key
        current_count = count
    else:
        # If the key is the same as the current key, increment the count
        current_count += count

# Output the last key and its count
if current_key is not None:
    print(f"{current_key}\t{current_count}")
```

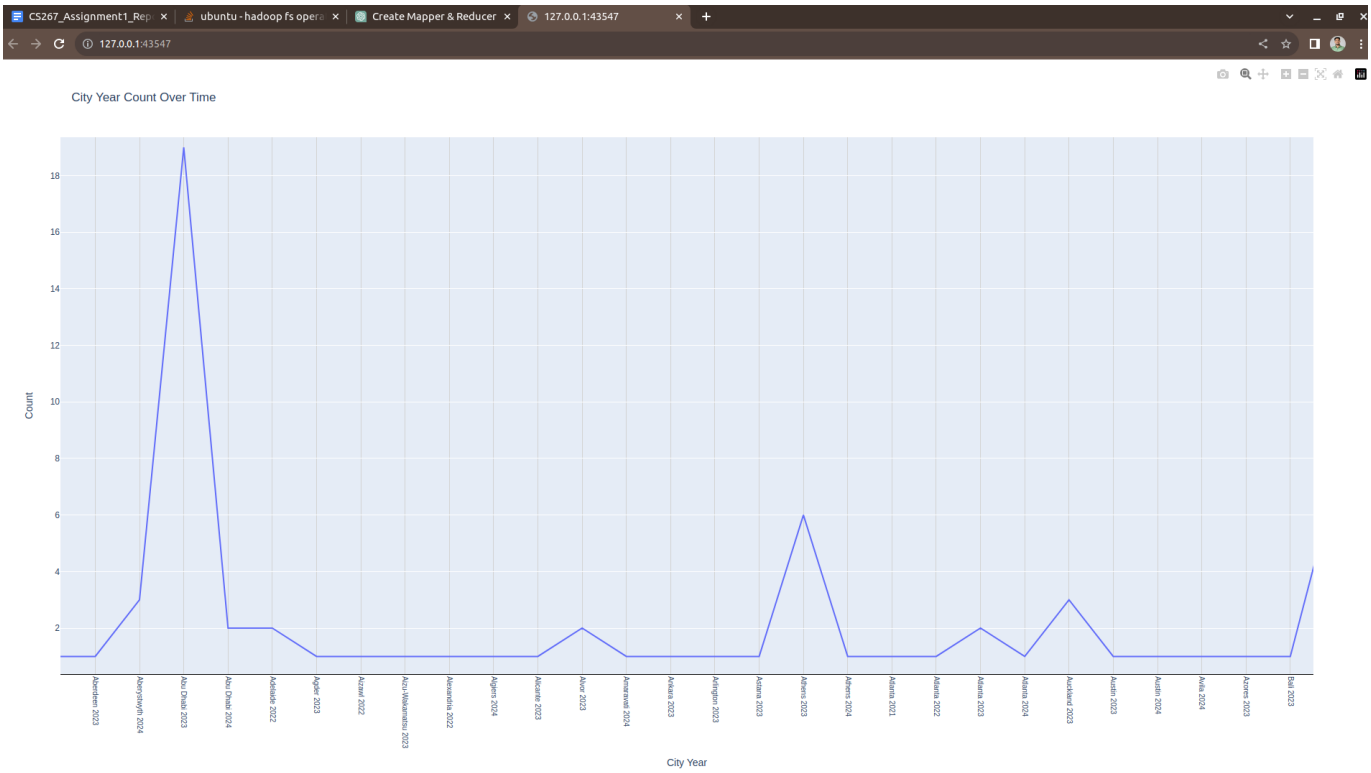
The input file my plotly is using is part-00000 that hadoop reducer generates, I have just renamed it to city_year_count.tsv

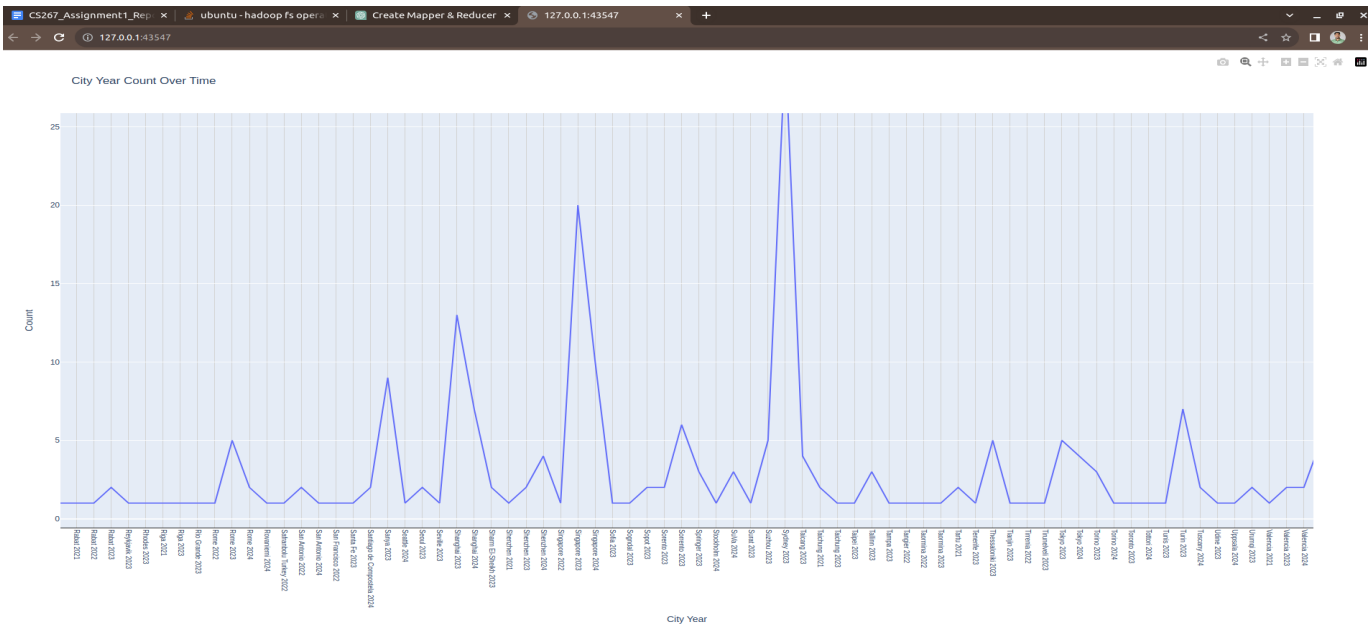
```python
import pandas as pd
import plotly.express as px

# Read data from output file that reducer created
data = []
with open('city_year_count.tsv', 'r') as file:
    for line in file:
        city_year_count = line.strip().split('\t')
        if len(city_year_count) == 2:
            city_year, count = city_year_count
            data.append({'CityYear': city_year, 'Count': int(count)})

# Creating a DataFramee
df = pd.DataFrame(data)

# Create the line graph using Plotly, where 'CityYear' is used as the X-axis
fig = px.line(df, x='CityYear', y='Count', title='City Year Count Over Time')
fig.update_xaxes(
    title_text='City Year',
    tickangle=90,  # Rotate the tick labels for better readability
    tickfont=dict(size=10),  # Adjust the font size of tick labels
    showline=True,  # Show X-axis line
    linewidth=1,  # Line width of X-axis
    linecolor='black',  # Line color of X-axis
    showgrid=True,  # Show gridlines
    gridcolor='lightgray',  # Color of gridlines
    zeroline=False,  # Hide the zero line
)
fig.update_yaxes(title_text='Count')

# Show the plot
fig.show()
```

Graph of conference count per city per year



Zoom in of the screenshots





Outcomes:

Able to run the web crawlers while complying to the policies. Understood Beautiful Soap of Python for crawling
Understood data cleaning using an open refine tool. Got knowledge of facet creations, text filters and clustering
Understood Hadoop ran for combined keys
Understood Graph plotting using plotly library

References:

https://openrefine.org/docs/manual/facets
https://librarycarpentry.org/lc-open-refine/10-data-transformation.html
https://realpython.com/beautiful-soup-web-scraper-python/#:~:text=Beautiful%20Soup%20is%20a%20Python,web%20page%20using%20developer%20tools.
https://plotly.com/python/
https://elixirforum.com/t/create-a-new-map-with-multiple-keys-through-each-iteration-of-map-new/44990
https://groups.google.com/g/openrefine/c/9LB9OT0q2rY
https://openrefine.org/docs/manual/celledit ing#:~:text=One%20way%20of%20doing%20this,the%20cells%20in%20the%20facet.
