

Northeastern University



US Traffic Accidents Data Analysis

Apache Hadoop

INFO 7245 Engineering Big Data

Professor: Yusuf Ozbek

Tanvi Rajendra Gurav
NUID: 001306848

Table of Contents

Sr. No.	Topic	Pg. No.
1	About Dataset	3
2	Simple MR Job: Number of Traffic Accidents occurred by State	6
3	Simple MR Job: Number of Accidents near Junction VS not a Junction	7
4	Numerical Summarization: Average, Min, Max Visibility (miles), Max StartDate Per Accident Severity	8
5	Numerical Summarization: Most Recent (Max) Accident Date Per City	9
6	Numerical Summarization: Percentage of Accidents in each Month	10
7	Top ‘N’ Filtering Pattern: Top 10 States with Highest Number of Accidents	11
8	Top ‘N’ Filtering Pattern: Top 5 Accident Conducive Weather Conditions	12
9	Secondary Sorting: Number of Accidents Per County Per Year	14
10	Partitioning: Partitioning based on Wind Directions	16
11	Inverted Index Pattern: Cities present in each County	18
12	Binning: Binning based on Accidents on each side of the road (Left/Right)	19
13	Pig: Accidents in Cities belonging to US/Eastern Time zone	21
14	Pig: Counties in CA where accidents happened with poor visibility	22
15	Hive: Cities in the State of Ohio which caused delays on a rainy day	23
16	Hive: Inner join based on Wind Direction	23
17	Hive: Maximum Temperature for each Weather Condition during accidents	23
18	Consolidated Scripts	24
19	Appendix	26
20	References	52

DATASET (<https://www.kaggle.com/sobhanmoosavi/us-accidents>)

Description:

This is a countrywide car accident dataset, which covers 49 states of the USA. The accident data are collected from February 2016 to Dec 2020, using two APIs that provide streaming traffic incident (or event) data. These APIs broadcast traffic data captured by a variety of entities, such as the US and state departments of transportation, law enforcement agencies, traffic cameras, and traffic sensors within the road-networks. Currently, there are about 4.2 million accident records in this dataset.

Acknowledgements:

Please cite the following papers if you use this dataset:

Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, and Rajiv Ramnath. "[A Countrywide Traffic Accident Dataset.](#)", 2019.

Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, Radu Teodorescu, and Rajiv Ramnath. "[Accident Risk Prediction based on Heterogeneous Sparse Data: New Dataset and Insights.](#)" In proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2019.

Content:

This dataset has been collected in real-time, using multiple Traffic APIs. Currently, it contains accident data that are collected from February 2016 to Dec 2020 for the Contiguous United States. Check [here](#) to learn more about this dataset.

Usage Policy and Legal Disclaimer:

This dataset is being distributed only for Research purposes, under Creative Commons Attribution-Noncommercial-ShareAlike license (CC BY-NC-SA 4.0). By clicking on download button(s) below, you are agreeing to use this data only for non-commercial, research, or academic applications. You may need to cite the above papers if you use this dataset.

#	Attribute	Description	Null
1	ID	This is a unique identifier of the accident record.	No
2	Source	Indicates source of the accident report (i.e. the API which reported the accident.).	No
3	TMC	A traffic accident may have a Traffic Message Channel (TMC) code which provides more detailed description of the event.	Yes
4	Severity	Shows the severity of the accident, a number between 1 and 4, where 1 indicates the least impact on traffic (i.e., short delay as a result of the accident) and 4 indicates a significant impact on traffic (i.e., long delay).	No
5	Start_Time	Shows start time of the accident in local time zone.	No
6	End_Time	Shows end time of the accident in local time zone. End time here refers to when the impact of accident on traffic flow was dismissed.	No
7	Start_Lat	Shows latitude in GPS coordinate of the start point.	No
8	Start_Lng	Shows longitude in GPS coordinate of the start point.	No
9	End_Lat	Shows latitude in GPS coordinate of the end point.	Yes
10	End_Lng	Shows longitude in GPS coordinate of the end point.	Yes
11	Distance(mi)	The length of the road extent affected by the accident.	No
12	Description	Shows natural language description of the accident.	No
13	Number	Shows the street number in address field.	Yes
14	Street	Shows the street name in address field.	Yes
15	Side	Shows the relative side of the street (Right/Left) in address field.	Yes
16	City	Shows the city in address field.	Yes
17	County	Shows the county in address field.	Yes
18	State	Shows the state in address field.	Yes
19	Zipcode	Shows the zipcode in address field.	Yes
20	Country	Shows the country in address field.	Yes
21	Timezone	Shows timezone based on the location of the accident (eastern, central, etc.).	Yes
22	Airport_Code	Denotes an airport-based weather station which is the closest one to location of the accident.	Yes
23	Weather_Timestamp	Shows the time-stamp of weather observation record (in local time).	Yes
24	Temperature(F)	Shows the temperature (in Fahrenheit).	Yes

#	Attribute	Description	Null
25	Wind_Chill(F)	Shows the wind chill (in Fahrenheit).	Yes
26	Humidity(%)	Shows the humidity (in percentage).	Yes
27	Pressure(in)	Shows the air pressure (in inches).	Yes
28	Visibility(mi)	Shows visibility (in miles).	Yes
29	Wind_Direction	Shows wind direction.	Yes
30	Wind_Speed(mph)	Shows wind speed (in miles per hour).	Yes
31	Precipitation(in)	Shows precipitation amount in inches, if there is any.	Yes
32	Weather_Condition	Shows the weather condition (rain, snow, thunderstorm, fog, etc.)	Yes
33	Amenity	A POI annotation which indicates presence of amenity in a nearby location.	No
34	Bump	A POI annotation which indicates presence of speed bump or hump in a nearby location.	No
35	Crossing	A POI annotation which indicates presence of crossing in a nearby location.	No
36	Give_Way	A POI annotation which indicates presence of give_way in a nearby location.	No
37	Junction	A POI annotation which indicates presence of junction in a nearby location.	No
38	No_Exit	A POI annotation which indicates presence of no_exit in a nearby location.	No
39	Railway	A POI annotation which indicates presence of railway in a nearby location.	No
40	Roundabout	A POI annotation which indicates presence of roundabout in a nearby location.	No
41	Station	A POI annotation which indicates presence of station in a nearby location.	No
42	Stop	A POI annotation which indicates presence of stop in a nearby location.	No
43	Traffic_Calming	A POI annotation which indicates presence of traffic_calming in a nearby location.	No
44	Traffic_Signal	A POI annotation which indicates presence of traffic_signal in a nearby location.	No
45	Turning_Loop	A POI annotation which indicates presence of turning_loop in a nearby location.	No
46	Sunrise_Sunset	Shows the period of day (i.e. day or night) based on sunrise/sunset.	Yes
47	Civil_Twilight	Shows the period of day (i.e. day or night) based on civil_twilight .	Yes
48	Nautical_Twilight	Shows the period of day (i.e. day or night) based on nautical_twilight .	Yes
49	Astronomical_Twilight	Shows the period of day (i.e. day or night) based on astronomical_twilight .	Yes

1. Number of Traffic Accidents occurred by State

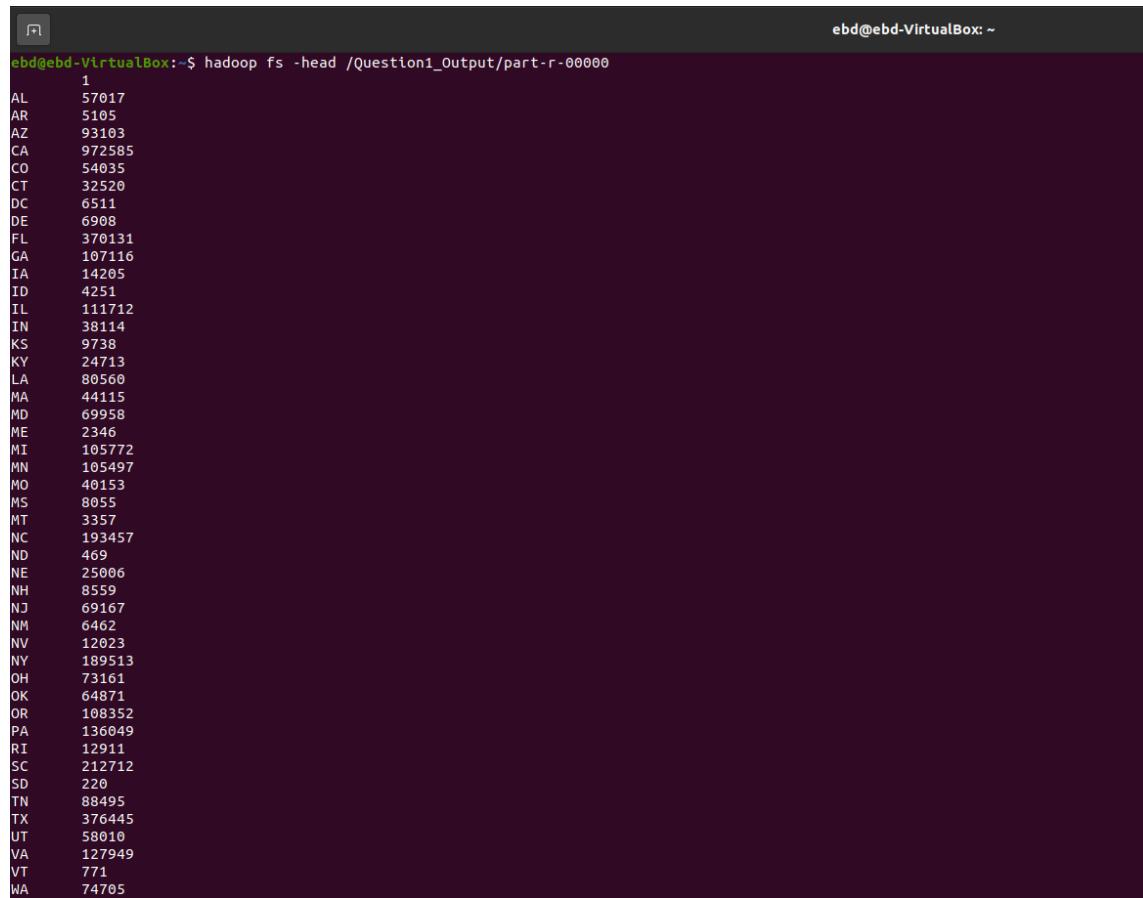
Description: This is a simple MapReduce Job that gives us an aggregate count of accidents occurred in each state in United States.

Analysis: We could analyze that the state of California (CA) has highest number of accidents i.e., 972,585.

Command to execute the jar:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficA  
ccidentsProject-1.0-SNAPSHOT.jar Question1.DriverClass /US_Accidents_Dec20.csv  
/Question1_Output
```

Output of the MapReduce Job:



```
ebd@ebd-VirtualBox:~$ hadoop fs -head /Question1_Output/part-r-00000  
1  
AL 57017  
AR 5105  
AZ 93103  
CA 972585  
CO 54035  
CT 32520  
DC 6511  
DE 6908  
FL 370131  
GA 107116  
IA 14205  
ID 4251  
IL 111712  
IN 38114  
KS 9738  
KY 24713  
LA 80560  
MA 44115  
MD 69958  
ME 2346  
MI 105772  
MN 105497  
MO 40153  
MS 8055  
MT 3357  
NC 193457  
ND 469  
NE 25006  
NH 8559  
NJ 69167  
NM 6462  
NV 12923  
NY 189513  
OH 73161  
OK 64871  
OR 108352  
PA 136049  
RI 12911  
SC 212712  
SD 220  
TN 88495  
TX 376445  
UT 58010  
VA 127949  
VT 771  
WA 74705
```

2. Number of Accidents near Junction VS not a Junction

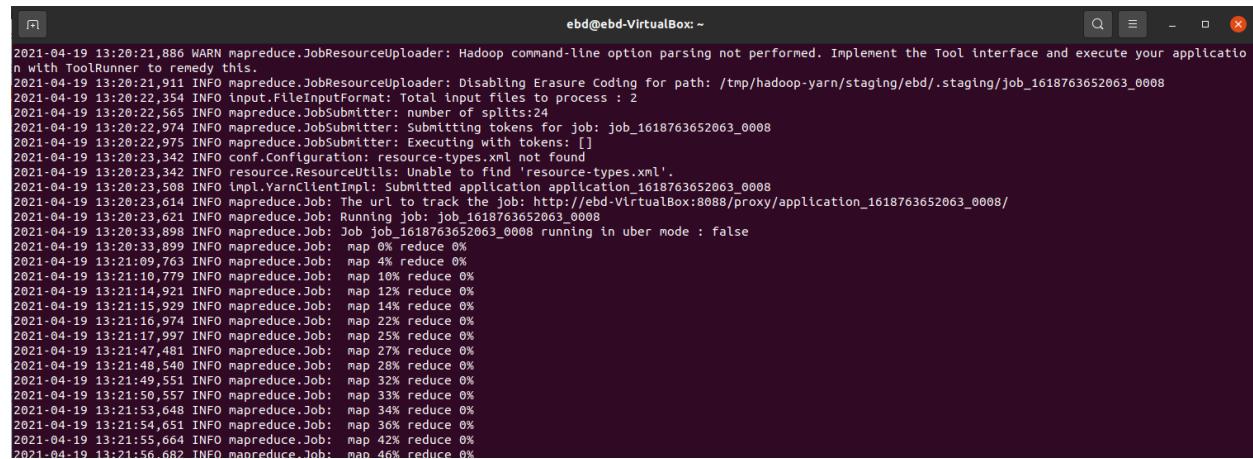
Description: This is a simple MapReduce Job that gives us an aggregate count of accidents occurred near a Junction versus number of accidents not near a Junction. **Junction is a Boolean column** having ‘True’ or ‘False’ values. Hence, we map a meaningful string to Boolean value using HashMap with key as Junction Boolean column and substitute string as value.

Analysis: We could analyze that a greater number of accidents took place where junction was not present i.e., 7,785,798

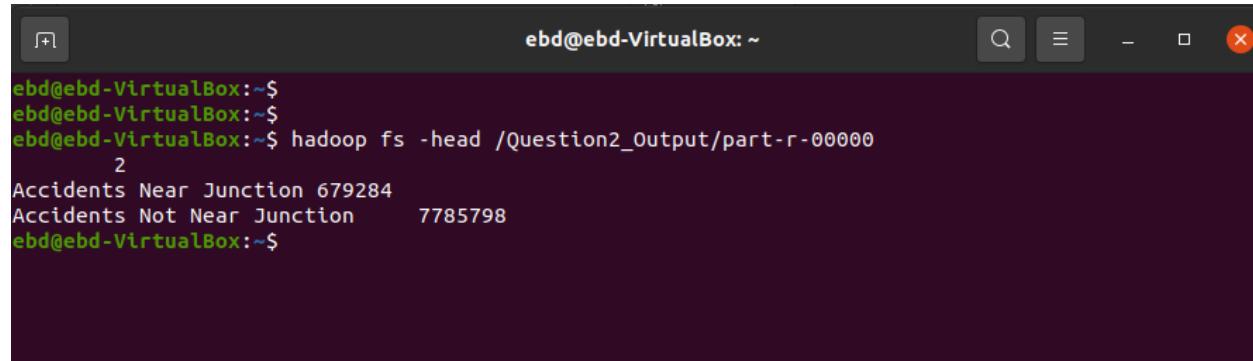
Command to execute the jar:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficA  
ccidentsProject-1.0-SNAPSHOT.jar Question2.DriverClass /US_Accidents_Dec20.csv  
/Question2_Output
```

Output of the MapReduce Job:



```
2021-04-19 13:20:21,886 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.  
2021-04-19 13:20:21,911 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/ebd/.staging/job_1618763652063_0008  
2021-04-19 13:20:22,354 INFO FileInputFormat: total input files to process : 2  
2021-04-19 13:20:22,565 INFO mapreduce.JobSubmitter: number of splits:24  
2021-04-19 13:20:22,974 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1618763652063_0008  
2021-04-19 13:20:22,975 INFO mapreduce.JobSubmitter: Executing with tokens: []  
2021-04-19 13:20:23,342 INFO conf.Configuration: resource-types.xml not found  
2021-04-19 13:20:23,342 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.  
2021-04-19 13:20:23,508 INFO impl.YarnClientImpl: Submitted application application_1618763652063_0008  
2021-04-19 13:20:23,614 INFO mapreduce.Job: The url to track the job: http://ebd-VirtualBox:8088/proxy/application_1618763652063_0008/  
2021-04-19 13:20:23,621 INFO mapreduce.Job: Running job: job_1618763652063_0008 running in uber mode : false  
2021-04-19 13:20:33,899 INFO mapreduce.Job: map 0% reduce 0%  
2021-04-19 13:21:09,763 INFO mapreduce.Job: map 4% reduce 0%  
2021-04-19 13:21:10,779 INFO mapreduce.Job: map 10% reduce 0%  
2021-04-19 13:21:14,921 INFO mapreduce.Job: map 12% reduce 0%  
2021-04-19 13:21:15,929 INFO mapreduce.Job: map 14% reduce 0%  
2021-04-19 13:21:16,974 INFO mapreduce.Job: map 22% reduce 0%  
2021-04-19 13:21:17,997 INFO mapreduce.Job: map 25% reduce 0%  
2021-04-19 13:21:47,481 INFO mapreduce.Job: map 27% reduce 0%  
2021-04-19 13:21:48,540 INFO mapreduce.Job: map 28% reduce 0%  
2021-04-19 13:21:49,551 INFO mapreduce.Job: map 32% reduce 0%  
2021-04-19 13:21:50,557 INFO mapreduce.Job: map 33% reduce 0%  
2021-04-19 13:21:53,648 INFO mapreduce.Job: map 34% reduce 0%  
2021-04-19 13:21:54,651 INFO mapreduce.Job: map 36% reduce 0%  
2021-04-19 13:21:55,664 INFO mapreduce.Job: map 42% reduce 0%  
2021-04-19 13:21:56,682 INFO mapreduce.Job: map 46% reduce 0%
```



```
ebd@ebd-VirtualBox:~$  
ebd@ebd-VirtualBox:~$  
ebd@ebd-VirtualBox:~$ hadoop fs -head /Question2_Output/part-r-00000  
2  
Accidents Near Junction 679284  
Accidents Not Near Junction 7785798  
ebd@ebd-VirtualBox:~$
```

3. Average, Min, Max Visibility (miles), Max StartDate Per Accident Severity

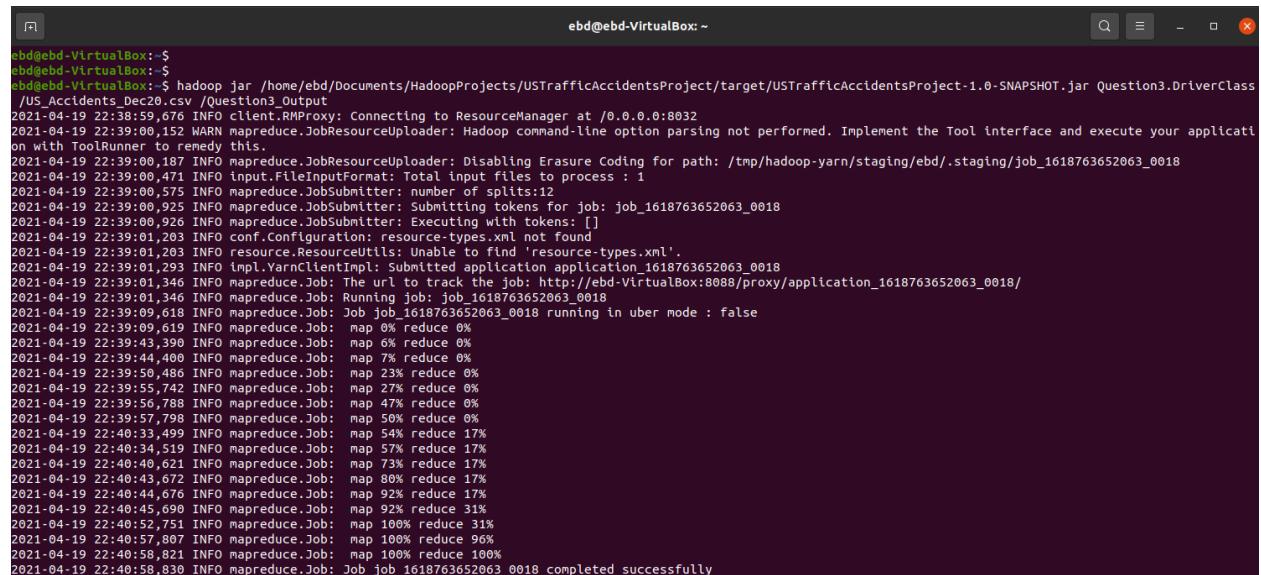
Description: This MapReduce Job performs Numerical Summarization that gives **Maximum, Minimum & Average Visibility along with Maximum Accident Start Date according to accident Severity** which ranges from 1 to 4. This is achieved with a custom class which implements Writable class.

Analysis: We could analyze that lowest average visibility of 9.091942 has caused most severe accident (Severity 4) with most recent accident case on 31st October 2020.

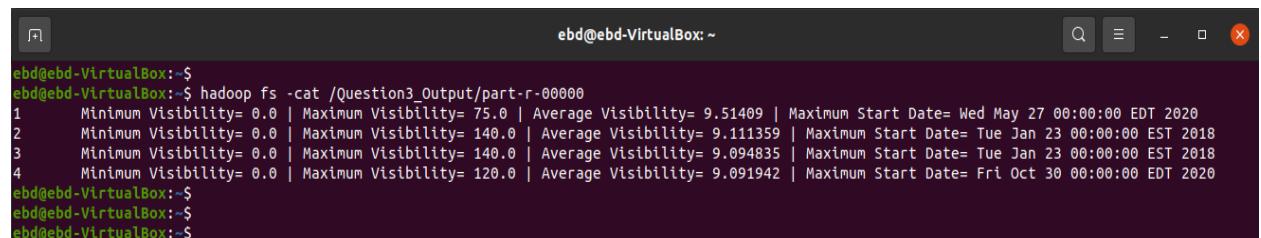
Command to execute the jar:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsProject-1.0-SNAPSHOT.jar Question3.DriverClass /US_Accidents_Dec20.csv  
/Question3_Output
```

Output of the MapReduce Job:



```
ebd@ebd-VirtualBox:~$ hadoop jar /home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsProject-1.0-SNAPSHOT.jar Question3.DriverClass /US_Accidents_Dec20.csv /Question3_Output  
2021-04-19 22:38:59,676 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
2021-04-19 22:39:00,152 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.  
2021-04-19 22:39:00,187 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/ebd/.staging/job_1618763652063_0018  
2021-04-19 22:39:00,471 INFO input.FileInputFormat: Total input files to process : 1  
2021-04-19 22:39:00,575 INFO mapreduce.JobSubmitter: number of splits:12  
2021-04-19 22:39:00,925 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1618763652063_0018  
2021-04-19 22:39:06,926 INFO mapreduce.JobSubmitter: Executing with tokens: []  
2021-04-19 22:39:01,203 INFO conf.Configuration: resource-types.xml not found  
2021-04-19 22:39:01,203 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.  
2021-04-19 22:39:01,293 INFO impl.YarnClientImpl: Submitted application application_1618763652063_0018  
2021-04-19 22:39:01,346 INFO mapreduce.Job: The url to track the job: http://ebd-VirtualBox:8088/proxy/application_1618763652063_0018/  
2021-04-19 22:39:01,346 INFO mapreduce.Job: Running job: job_1618763652063_0018  
2021-04-19 22:39:09,618 INFO mapreduce.Job: Job job_1618763652063_0018 running in uber mode : false  
2021-04-19 22:39:09,619 INFO mapreduce.Job: map 0% reduce 0%  
2021-04-19 22:39:43,390 INFO mapreduce.Job: map 6% reduce 0%  
2021-04-19 22:39:44,400 INFO mapreduce.Job: map 7% reduce 0%  
2021-04-19 22:39:50,486 INFO mapreduce.Job: map 23% reduce 0%  
2021-04-19 22:39:55,742 INFO mapreduce.Job: map 27% reduce 0%  
2021-04-19 22:39:56,788 INFO mapreduce.Job: map 47% reduce 0%  
2021-04-19 22:39:57,798 INFO mapreduce.Job: map 50% reduce 0%  
2021-04-19 22:40:33,499 INFO mapreduce.Job: map 54% reduce 17%  
2021-04-19 22:40:34,519 INFO mapreduce.Job: map 57% reduce 17%  
2021-04-19 22:40:40,621 INFO mapreduce.Job: map 73% reduce 17%  
2021-04-19 22:40:43,672 INFO mapreduce.Job: map 80% reduce 17%  
2021-04-19 22:40:44,676 INFO mapreduce.Job: map 92% reduce 17%  
2021-04-19 22:40:45,690 INFO mapreduce.Job: map 92% reduce 31%  
2021-04-19 22:40:52,751 INFO mapreduce.Job: map 100% reduce 31%  
2021-04-19 22:40:57,.897 INFO mapreduce.Job: map 100% reduce 96%  
2021-04-19 22:40:58,.821 INFO mapreduce.Job: map 100% reduce 100%  
2021-04-19 22:40:58,.830 INFO mapreduce.Job: Job job_1618763652063_0018 completed successfully
```



```
ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question3_Output/part-r-00000  
1 Minimum Visibility= 0.0 | Maximum Visibility= 75.0 | Average Visibility= 9.51409 | Maximum Start Date= Wed May 27 00:00:00 EDT 2020  
2 Minimum Visibility= 0.0 | Maximum Visibility= 140.0 | Average Visibility= 9.111359 | Maximum Start Date= Tue Jan 23 00:00:00 EST 2018  
3 Minimum Visibility= 0.0 | Maximum Visibility= 140.0 | Average Visibility= 9.094835 | Maximum Start Date= Tue Jan 23 00:00:00 EST 2018  
4 Minimum Visibility= 0.0 | Maximum Visibility= 120.0 | Average Visibility= 9.091942 | Maximum Start Date= Fri Oct 30 00:00:00 EDT 2020  
ebd@ebd-VirtualBox:~$  
ebd@ebd-VirtualBox:~$  
ebd@ebd-VirtualBox:~$
```

4. Most Recent Accident Date per City

Description: This MapReduce Job performs Numerical Summarization that gives **Maximum Accident Date for each City** in United States.

Analysis: We could analyze that in most of the cities, latest accident case was in the year 2020 with month ranging from Jan to Dec.

Command to execute the jar:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsProject-1.0-SNAPSHOT.jar Question4.DriverClass /US_Accidents_Dec20.csv  
/Question4_Output
```

Output of the MapReduce Job:

```
ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question4_Output/part-r-00000 | head -n 50  
Aaronsburg      Latest Accident Occurred On: Wed Aug 05 00:00:00 EDT 2020  
Abbeville       Latest Accident Occurred On: Thu Dec 24 00:00:00 EST 2020  
Abbotsford      Latest Accident Occurred On: Fri Mar 20 00:00:00 EDT 2020  
Abbott          Latest Accident Occurred On: Wed Feb 19 00:00:00 EST 2020  
Abbottstown     Latest Accident Occurred On: Wed Dec 09 00:00:00 EST 2020  
Aberdeen         Latest Accident Occurred On: Thu Dec 31 00:00:00 EST 2020  
Aberdeen Proving Ground Latest Accident Occurred On: Wed Jul 17 00:00:00 EDT 2019  
Abernathy        Latest Accident Occurred On: Thu Sep 03 00:00:00 EDT 2020  
Abilene          Latest Accident Occurred On: Fri Sep 25 00:00:00 EDT 2020  
Abingdon         Latest Accident Occurred On: Thu Dec 31 00:00:00 EST 2020  
Abington         Latest Accident Occurred On: Thu Dec 31 00:00:00 EST 2020  
Abiquiu          Latest Accident Occurred On: Fri Sep 22 00:00:00 EDT 2017  
Abita Springs    Latest Accident Occurred On: Fri Apr 13 00:00:00 EDT 2018  
Abrams           Latest Accident Occurred On: Sat May 05 00:00:00 EDT 2018  
Absarokee        Latest Accident Occurred On: Tue Dec 08 00:00:00 EST 2020  
Abscon           Latest Accident Occurred On: Wed Dec 23 00:00:00 EST 2020  
Acampo            Latest Accident Occurred On: Wed Dec 30 00:00:00 EST 2020  
Accident          Latest Accident Occurred On: Fri Dec 11 00:00:00 EST 2020  
Ackokeek          Latest Accident Occurred On: Thu Dec 31 00:00:00 EST 2020  
Accomac           Latest Accident Occurred On: Fri Nov 13 00:00:00 EST 2020  
Accord            Latest Accident Occurred On: Wed Jan 15 00:00:00 EST 2020  
Ackerman          Latest Accident Occurred On: Tue Jan 22 00:00:00 EST 2019  
Ackworth          Latest Accident Occurred On: Sat Aug 12 00:00:00 EDT 2017  
Acme              Latest Accident Occurred On: Mon Dec 14 00:00:00 EST 2020  
Acra              Latest Accident Occurred On: Wed Jan 11 00:00:00 EST 2017  
Acton              Latest Accident Occurred On: Thu Dec 31 00:00:00 EST 2020  
Acushnet          Latest Accident Occurred On: Mon Jun 08 00:00:00 EDT 2020  
Acworth           Latest Accident Occurred On: Sat Dec 26 00:00:00 EST 2020  
Ada               Latest Accident Occurred On: Thu Nov 12 00:00:00 EST 2020  
Adah              Latest Accident Occurred On: Sat Dec 21 00:00:00 EST 2019  
Adair              Latest Accident Occurred On: Sun Dec 27 00:00:00 EST 2020  
Adair Village     Latest Accident Occurred On: Mon Jul 29 00:00:00 EDT 2019  
Adairsville       Latest Accident Occurred On: Sat Dec 19 00:00:00 EST 2020  
Adams              Latest Accident Occurred On: Sun Dec 20 00:00:00 EST 2020  
Adams Center      Latest Accident Occurred On: Fri Feb 01 00:00:00 EST 2019  
Adams Run          Latest Accident Occurred On: Fri Dec 11 00:00:00 EST 2020  
Adamsburg         Latest Accident Occurred On: Wed Oct 14 00:00:00 EDT 2020  
Adamstown         Latest Accident Occurred On: Tue Dec 08 00:00:00 EST 2020  
Adamsville        Latest Accident Occurred On: Thu Dec 10 00:00:00 EST 2020  
Addis              Latest Accident Occurred On: Fri Nov 20 00:00:00 EST 2020  
Addison           Latest Accident Occurred On: Thu Dec 31 00:00:00 EST 2020  
Addy               Latest Accident Occurred On: Sun Aug 04 00:00:00 EDT 2019  
Addyston          Latest Accident Occurred On: Fri Jun 26 00:00:00 EDT 2020  
Adel               Latest Accident Occurred On: Thu Dec 31 00:00:00 EST 2020  
Adelanto          Latest Accident Occurred On: Sun Dec 27 00:00:00 EST 2020  
Adell              Latest Accident Occurred On: Fri Dec 08 00:00:00 EST 2017
```

5. Percentage of accidents in each month

Description: This MapReduce Job gives **the Percentage of Accidents took place in each month** from 2016 to 2020.

Analysis: We could analyze that the percentage of accidents were more in later 4 months (Fall Season) of each year from October to December ranging from 10.85% to 14.86%.

Command to execute the jar:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsProject-1.0-SNAPSHOT.jar Question5.DriverClass /US_Accidents_Dec20.csv  
/Question5_Output
```

Output of the MapReduce Job:

```
ebd@ebd-VirtualBox:~$ hadoop jar /home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsProject-1.0-SNAPSHOT.jar  
Question5.DriverClass /US_Accidents_Dec20.csv /Question5_Output  
2021-04-20 10:32:28,721 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
2021-04-20 10:32:29,328 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and  
execute your application with ToolRunner to remedy this.  
2021-04-20 10:32:29,355 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/ebd/.staging/job_1618763652063_0024  
2021-04-20 10:32:29,687 INFO input.FileInputFormat: Total input files to process : 1  
2021-04-20 10:32:29,799 INFO mapreduce.JobsSubmitter: number of splits:12  
2021-04-20 10:32:30,138 INFO mapreduce.JobsSubmitter: Submitting tokens for job: job_1618763652063_0024  
2021-04-20 10:32:30,163 INFO mapreduce.JobsSubmitter: Executing with tokens: []  
2021-04-20 10:32:30,629 INFO conf.Configuration: resource-types.xml not found  
2021-04-20 10:32:30,629 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.  
2021-04-20 10:32:30,811 INFO impl.YarnClientImpl: Submitted application application_1618763652063_0024  
2021-04-20 10:32:30,853 INFO mapreduce.Job: The url to track the job: http://ebd-VirtualBox:8088/proxy/application_1618763652063_0024/  
2021-04-20 10:32:30,853 INFO mapreduce.Job: Running job: job_1618763652063_0024  
2021-04-20 10:32:42,218 INFO mapreduce.Job: Job job_1618763652063_0024 running in uber mode : false  
2021-04-20 10:32:42,219 INFO mapreduce.Job: map 0% reduce 0%  
2021-04-20 10:33:18,922 INFO mapreduce.Job: map 1% reduce 0%  
2021-04-20 10:33:24,402 INFO mapreduce.Job: map 2% reduce 0%  
2021-04-20 10:33:25,584 INFO mapreduce.Job: map 5% reduce 0%  
2021-04-20 10:33:31,749 INFO mapreduce.Job: map 7% reduce 0%  
2021-04-20 10:33:34,616 INFO mapreduce.Job: map 11% reduce 0%  
2021-04-20 10:33:38,032 INFO mapreduce.Job: map 12% reduce 0%
```

```
ebd@ebd-VirtualBox:~$  
ebd@ebd-VirtualBox:~$  
ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question5_Output/part-r-00000  
April      8.52%  
August     9.28%  
December   14.86%  
February   8.09%  
January    8.59%  
July       7.35%  
June       8.83%  
March      8.35%  
May        8.44%  
November   14.06%  
October    13.23%  
September   10.85%  
ebd@ebd-VirtualBox:~$
```

6. Top 10 States with Highest Number of Accidents

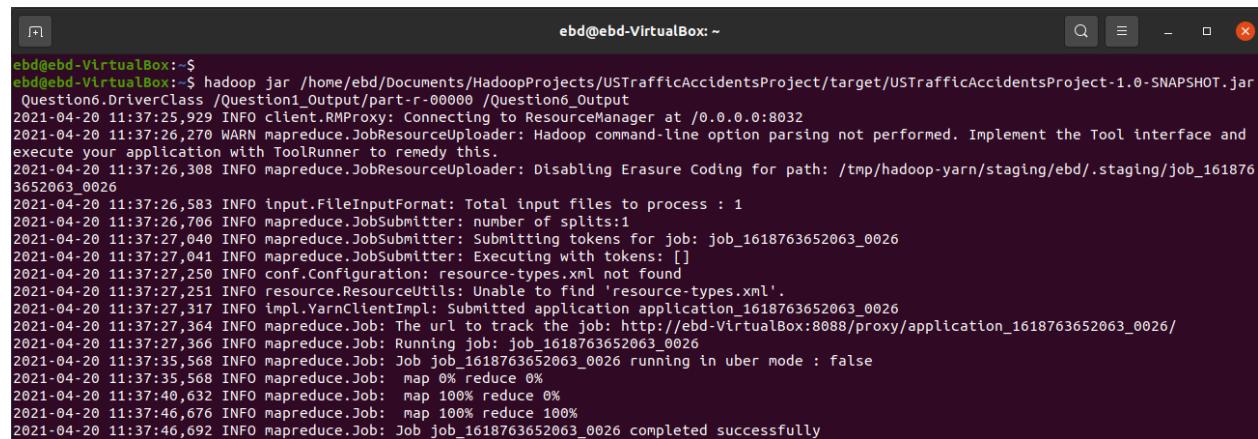
Description: This MapReduce Job performs Top ‘N’ Filtering Pattern that gives Top 10 States with highest number of Accidents from year 2016 to 2020.

Analysis: We could analyze that the state of California tops the number of Traffic Accidents in United States following Texas and Florida.

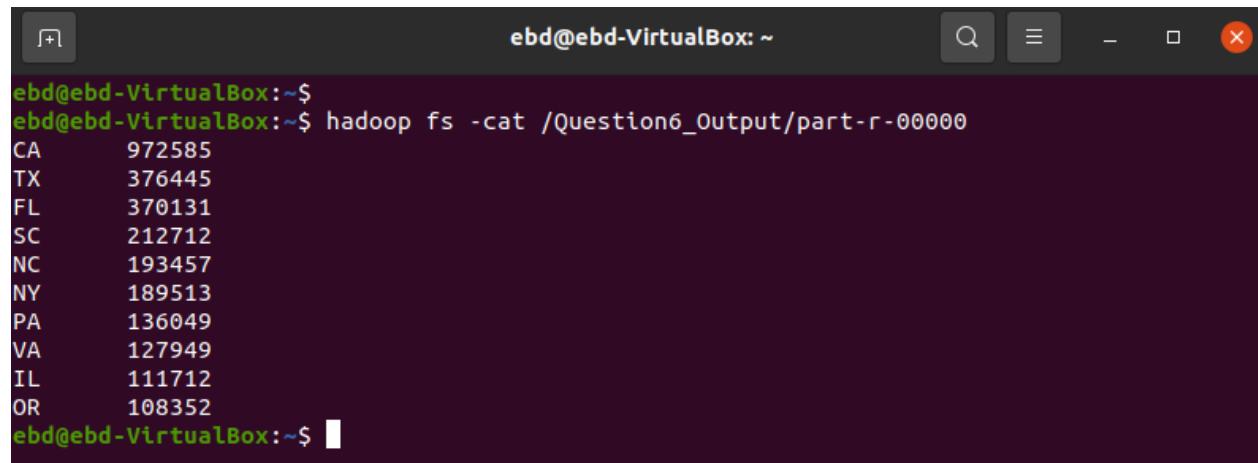
Command to execute the jar:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficA  
ccidentsProject-1.0-SNAPSHOT.jar Question6.DriverClass /Question1_Output/part-r-  
00000 /Question6_Output
```

Output of the MapReduce Job:



```
ebd@ebd-VirtualBox:~$ hadoop jar /home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsProject-1.0-SNAPSHOT.jar Question6.DriverClass /Question1_Output/part-r-00000 /Question6_Output  
2021-04-20 11:37:25,929 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
2021-04-20 11:37:26,270 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.  
2021-04-20 11:37:26,308 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/ebd/.staging/job_1618763652063_0026  
2021-04-20 11:37:26,583 INFO input.FileInputFormat: Total input files to process : 1  
2021-04-20 11:37:26,706 INFO mapreduce.JobSubmitter: number of splits:1  
2021-04-20 11:37:27,040 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1618763652063_0026  
2021-04-20 11:37:27,041 INFO mapreduce.JobSubmitter: Executing with tokens: []  
2021-04-20 11:37:27,250 INFO conf.Configuration: resource-types.xml not found  
2021-04-20 11:37:27,251 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.  
2021-04-20 11:37:27,317 INFO impl.YarnClientImpl: Submitted application application_1618763652063_0026  
2021-04-20 11:37:27,364 INFO mapreduce.Job: The url to track the job: http://ebd-VirtualBox:8088/proxy/application_1618763652063_0026  
2021-04-20 11:37:27,366 INFO mapreduce.Job: Running job: job_1618763652063_0026  
2021-04-20 11:37:35,568 INFO mapreduce.Job: Job job_1618763652063_0026 running in uber mode : false  
2021-04-20 11:37:35,568 INFO mapreduce.Job: map 0% reduce 0%  
2021-04-20 11:37:40,632 INFO mapreduce.Job: map 100% reduce 0%  
2021-04-20 11:37:46,676 INFO mapreduce.Job: map 100% reduce 100%  
2021-04-20 11:37:46,692 INFO mapreduce.Job: Job job_1618763652063_0026 completed successfully
```



```
ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question6_Output/part-r-00000  
CA 972585  
TX 376445  
FL 370131  
SC 212712  
NC 193457  
NY 189513  
PA 136049  
VA 127949  
IL 111712  
OR 108352  
ebd@ebd-VirtualBox:~$
```

7. Top 5 Weather Conditions most conducive to Traffic Accidents

Description: This MapReduce Job performs Top ‘N’ Filtering Pattern that gives **Top 5 Weather Conditions most conducive to Traffic Accidents**.

Analysis: We could analyze that most of the traffic accidents occur when the weather is Fair/Clear or Mostly/Partly Cloudy or Outcast

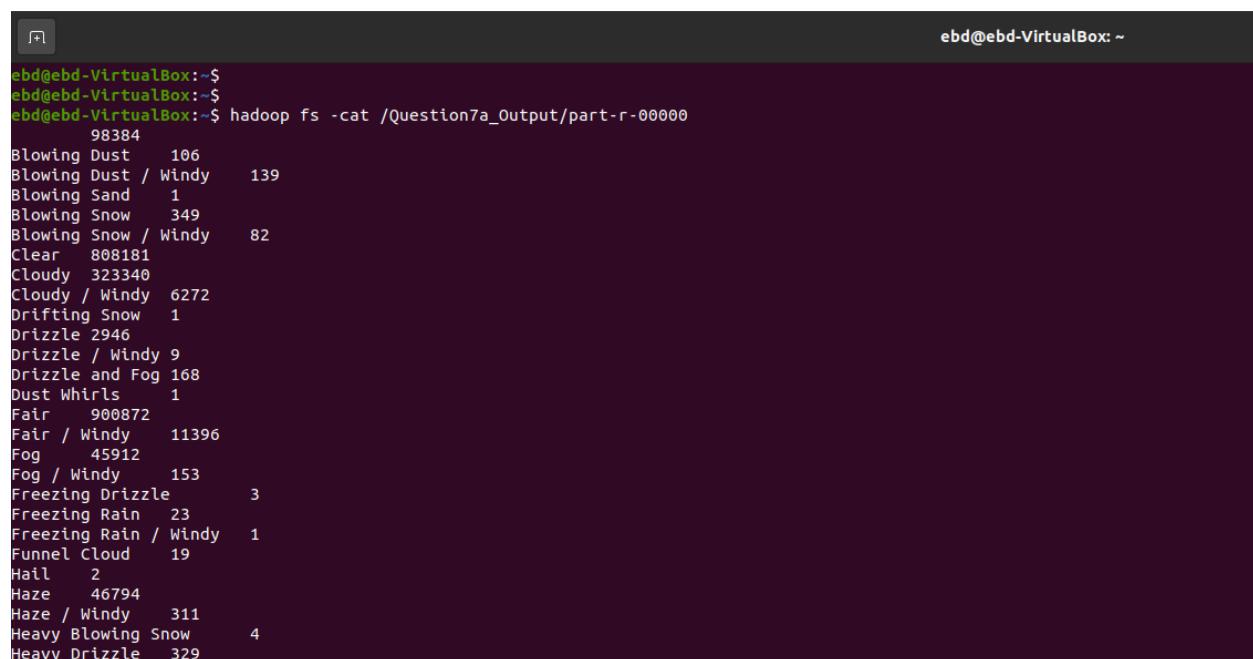
a. Command to get the count of accidents per weather condition:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficA  
ccidentsProject-1.0-SNAPSHOT.jar Question7.a.DriverClass /US_Accidents_Dec20.csv  
/Question7a_Output
```

b. Command to get the Top 5 weather conditions from above output:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficA  
ccidentsProject-1.0-SNAPSHOT.jar Question7.b.DriverClass /Question7a_Output/part-  
r-00000 /Question7b_Output
```

Output of the first MapReduce Job:



```
ebd@ebd-VirtualBox:~$  
ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question7a_Output/part-r-00000  
98384  
Blowing Dust      106  
Blowing Dust / Windy    139  
Blowing Sand      1  
Blowing Snow      349  
Blowing Snow / Windy   82  
Clear             808181  
Cloudy            323340  
Cloudy / Windy     6272  
Drifting Snow     1  
Drizzle           2946  
Drizzle / Windy    9  
Drizzle and Fog   168  
Dust Whirls       1  
Fair              900872  
Fair / Windy       11396  
Fog               45912  
Fog / Windy        153  
Freezing Drizzle    3  
Freezing Rain      23  
Freezing Rain / Windy  1  
Funnel Cloud       19  
Hail               2  
Haze              46794  
Haze / Windy       311  
Heavy Blowing Snow   4  
Heavy Drizzle      329
```

Using the first output as input to the second job for filtering

```
ebd@ebd-VirtualBox:~$ hadoop jar /home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsProject-1.0-SNAPSHOT.jar Question7.b.DriverClass /Question7a_Output/part-r-00000 /Question7b_Output
2021-04-21 03:23:19,815 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2021-04-21 03:23:20,357 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2021-04-21 03:23:20,423 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/ebd/.staging/job_1618763652063_0029
2021-04-21 03:23:21,095 INFO input.FileInputFormat: Total input files to process : 1
2021-04-21 03:23:21,853 INFO mapreduce.JobSubmitter: number of splits:1
2021-04-21 03:23:22,397 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1618763652063_0029
2021-04-21 03:23:22,398 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-04-21 03:23:22,646 INFO conf.Configuration: resource-types.xml not found
2021-04-21 03:23:22,646 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-04-21 03:23:22,770 INFO Impl.YarnClientImpl: Submitted application application_1618763652063_0029
2021-04-21 03:23:22,841 INFO mapreduce.Job: The url to track the job: http://ebd-VirtualBox:8088/proxy/application_1618763652063_0029/
2021-04-21 03:23:22,841 INFO mapreduce.Job: Running job: job_1618763652063_0029 running in uber mode : false
2021-04-21 03:23:31,081 INFO mapreduce.Job: Job job_1618763652063_0029 running in uber mode : false
2021-04-21 03:23:31,082 INFO mapreduce.Job: map 0% reduce 0%
2021-04-21 03:23:36,171 INFO mapreduce.Job: map 100% reduce 0%
2021-04-21 03:23:42,252 INFO mapreduce.Job: map 100% reduce 100%
2021-04-21 03:23:43,268 INFO mapreduce.Job: Job job_1618763652063_0029 completed successfully
2021-04-21 03:23:43,268 INFO mapreduce.Job: Job job_1618763652063_0029 completed successfully
```

```
ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question7b_Output/part-r-00000
Fair      900872
Clear     808181
Mostly Cloudy   571743
Partly Cloudy    397415
Overcast      382485
ebd@ebd-VirtualBox:~$
```

8. Number of Accidents Per County Per Year

Description: This MapReduce Job performs ‘Secondary Sorting’ where we are sorting, and partitioning based on year from 2016 to 2020. We get 5 partitions as shown in the output.

Command to execute the jar:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficA  
ccidentsProject-1.0-SNAPSHOT.jar Question8.DriverClass /US_Accidents_Dec20.csv  
/Question8_Output
```

Output of the MapReduce Job:

```
ebd@ebd-VirtualBox:~$ hadoop jar /home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsProject-1.0-SNAPSHOT.jar Question8.DriverClass  
/US_Accidents_Dec20.csv /Question8_Output  
2021-04-21 05:14:39,831 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
2021-04-21 05:14:40,399 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.  
2021-04-21 05:14:40,430 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/ebd_staging/job_1618763652063_0031  
2021-04-21 05:14:40,737 INFO input.FileInputFormat: Total input files to process : 1  
2021-04-21 05:14:40,853 INFO mapreduce.JobSubmitter: number of splits:12  
2021-04-21 05:14:41,198 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1618763652063_0031  
2021-04-21 05:14:41,199 INFO mapreduce.JobSubmitter: Executing with tokens: []  
2021-04-21 05:14:41,462 INFO conf.Configuration: resource-types.xml not found  
2021-04-21 05:14:41,462 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.  
2021-04-21 05:14:41,702 INFO impl.YarnClientImpl: Submitted application application_1618763652063_0031  
2021-04-21 05:14:41,785 INFO mapreduce.Job: The url to track the job: http://ebd-VirtualBox:8088/proxy/application_1618763652063_0031/  
2021-04-21 05:14:41,803 INFO mapreduce.Job: Running job: job_1618763652063_0031  
2021-04-21 05:14:50,058 INFO mapreduce.Job: Job job_1618763652063_0031 running in uber mode : false  
2021-04-21 05:14:50,059 INFO mapreduce.Job: map 0% reduce 0%  
2021-04-21 05:15:31,113 INFO mapreduce.Job: map 1% reduce 0%  
2021-04-21 05:15:33,137 INFO mapreduce.Job: map 2% reduce 0%  
2021-04-21 05:15:36,262 INFO mapreduce.Job: map 3% reduce 0%  
2021-04-21 05:15:37,401 INFO mapreduce.Job: map 7% reduce 0%  
2021-04-21 05:15:38,409 INFO mapreduce.Job: map 8% reduce 0%
```

```
ebd@ebd-VirtualBox:~$  
ebd@ebd-VirtualBox:~$ hadoop fs -ls /Question8_Output/  
Found 6 items  
-rw-r--r-- 1 ebd supergroup 0 2021-04-21 05:17 /Question8_Output/_SUCCESS  
-rw-r--r-- 1 ebd supergroup 1726 2021-04-21 05:17 /Question8_Output/part-r-00000  
-rw-r--r-- 1 ebd supergroup 1687 2021-04-21 05:17 /Question8_Output/part-r-00001  
-rw-r--r-- 1 ebd supergroup 1710 2021-04-21 05:17 /Question8_Output/part-r-00002  
-rw-r--r-- 1 ebd supergroup 1718 2021-04-21 05:17 /Question8_Output/part-r-00003  
-rw-r--r-- 1 ebd supergroup 1718 2021-04-21 05:17 /Question8_Output/part-r-00004  
ebd@ebd-VirtualBox:~$
```

```

ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question8_Output/part-r-00001 | head -n 10
County: Abbeville| Year: 2016 | Accidents:      58
County: Acadia| Year: 2016 | Accidents:       8
County: Accomack| Year: 2016 | Accidents:     12
County: Ada| Year: 2016 | Accidents:     74
County: Adair| Year: 2016 | Accidents:    14
County: Adams| Year: 2016 | Accidents:   361
County: Addison| Year: 2016 | Accidents:     1
County: Aiken| Year: 2016 | Accidents:  662
County: Aitkin| Year: 2016 | Accidents:      6
County: Alachua| Year: 2016 | Accidents:  129
cat: Unable to write to output stream.
ebd@ebd-VirtualBox:~$
```

```

ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question8_Output/part-r-00002 | head -n 10
County: Abbeville| Year: 2017 | Accidents:      97
County: Acadia| Year: 2017 | Accidents:     23
County: Accomack| Year: 2017 | Accidents:    21
County: Ada| Year: 2017 | Accidents:     53
County: Adair| Year: 2017 | Accidents:    37
County: Adams| Year: 2017 | Accidents:  1307
County: Addison| Year: 2017 | Accidents:     13
County: Aiken| Year: 2017 | Accidents:  1377
County: Aitkin| Year: 2017 | Accidents:     22
County: Alachua| Year: 2017 | Accidents:  454
cat: Unable to write to output stream.
ebd@ebd-VirtualBox:~$
```

```

ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question8_Output/part-r-00003 | head -n 10
County: Abbeville| Year: 2018 | Accidents:     138
County: Acadia| Year: 2018 | Accidents:     43
County: Accomack| Year: 2018 | Accidents:    20
County: Ada| Year: 2018 | Accidents:     90
County: Adair| Year: 2018 | Accidents:     10
County: Adams| Year: 2018 | Accidents:  2285
County: Addison| Year: 2018 | Accidents:      3
County: Aiken| Year: 2018 | Accidents:  1296
County: Aitkin| Year: 2018 | Accidents:     15
County: Alachua| Year: 2018 | Accidents:  608
cat: Unable to write to output stream.
ebd@ebd-VirtualBox:~$
```

```

ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question8_Output/part-r-00004 | head -n 10
County: Abbeville| Year: 2019 | Accidents:     112
County: Acadia| Year: 2019 | Accidents:     19
County: Accomack| Year: 2019 | Accidents:    24
County: Ada| Year: 2019 | Accidents:     70
County: Adair| Year: 2019 | Accidents:     44
County: Adams| Year: 2019 | Accidents:  2663
County: Addison| Year: 2019 | Accidents:      5
County: Aiken| Year: 2019 | Accidents:  1019
County: Aitkin| Year: 2019 | Accidents:     78
County: Alachua| Year: 2019 | Accidents:  294
cat: Unable to write to output stream.
ebd@ebd-VirtualBox:~$
```

```

ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question8_Output/part-r-00000 | head -n 10
County: Abbeville| Year: 2020 | Accidents:     213
County: Acadia| Year: 2020 | Accidents:     34
County: Accomack| Year: 2020 | Accidents:    34
County: Ada| Year: 2020 | Accidents:   1107
County: Adair| Year: 2020 | Accidents:    103
County: Adams| Year: 2020 | Accidents:  2603
County: Addison| Year: 2020 | Accidents:     11
County: Aiken| Year: 2020 | Accidents:  1061
County: Aitkin| Year: 2020 | Accidents:     153
County: Alachua| Year: 2020 | Accidents:  795
cat: Unable to write to output stream.
ebd@ebd-VirtualBox:~$
```

9. Partitioning based on Wind Direction

Description: This MapReduce Job performs ‘Partitioning’ where we partition our input with 4.2million records based on 23 Wind Directions. Each file contains the records corresponding to specific Wind Direction and maps the Wind Direction Abbreviation to its description using ‘Inner Join’.

Command to execute the jar:

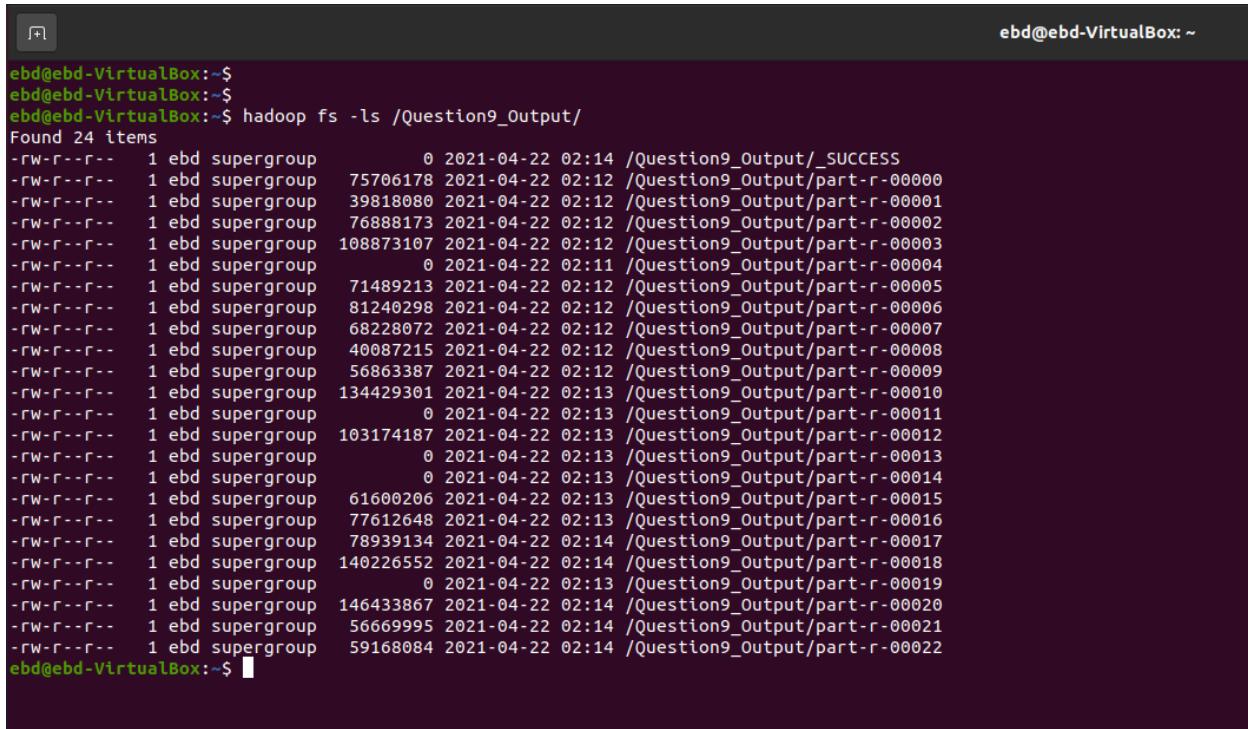
```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficA  
ccidentsProject-1.0-SNAPSHOT.jar Question9.DriverClass /US_Accidents_Dec20.csv  
/WindDirection.csv /Question9_Output
```

Wind Direction Input:

	A	B
1	Wind_Direction	Description
2	Calm	Calm
3	E	East
4	East	East
5	ENE	East-Northeast
6	ESE	East-Southeast
7	N	North
8	NE	Northeast
9	NNE	North-Northeast
10	NNW	North-Northwest
11	North	North
12	NW	Northwest
13	S	South
14	SE	Southeast
15	South	South
16	SSE	South-Southeast
17	SSW	South-Southwest
18	SW	Southwest
19	VAR	Variable
20	Variable	Variable
21	W	West
22	West	West
23	WNW	West-Northwest
24	WSW	West-Southwest

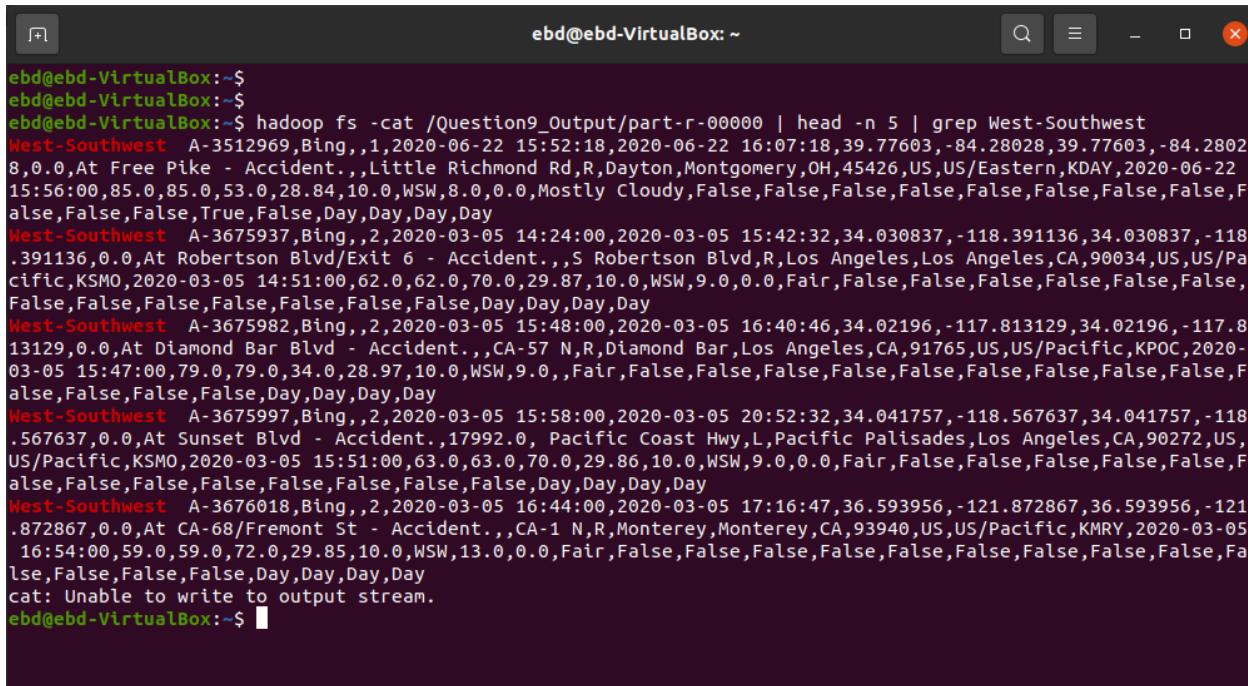
Output of the MapReduce Job:

23 Partitions, each file containing data related to one Wind Direction.



```
ebd@ebd-VirtualBox:~$ 
ebd@ebd-VirtualBox:~$ 
ebd@ebd-VirtualBox:~$ hadoop fs -ls /Question9_Output/
Found 24 items
-rw-r--r-- 1 ebd supergroup          0 2021-04-22 02:14 /Question9_Output/_SUCCESS
-rw-r--r-- 1 ebd supergroup 75706178 2021-04-22 02:12 /Question9_Output/part-r-00000
-rw-r--r-- 1 ebd supergroup 39818080 2021-04-22 02:12 /Question9_Output/part-r-00001
-rw-r--r-- 1 ebd supergroup 76888173 2021-04-22 02:12 /Question9_Output/part-r-00002
-rw-r--r-- 1 ebd supergroup 108873107 2021-04-22 02:12 /Question9_Output/part-r-00003
-rw-r--r-- 1 ebd supergroup          0 2021-04-22 02:11 /Question9_Output/part-r-00004
-rw-r--r-- 1 ebd supergroup 71489213 2021-04-22 02:12 /Question9_Output/part-r-00005
-rw-r--r-- 1 ebd supergroup 81240298 2021-04-22 02:12 /Question9_Output/part-r-00006
-rw-r--r-- 1 ebd supergroup 68228072 2021-04-22 02:12 /Question9_Output/part-r-00007
-rw-r--r-- 1 ebd supergroup 40687215 2021-04-22 02:12 /Question9_Output/part-r-00008
-rw-r--r-- 1 ebd supergroup 56863387 2021-04-22 02:12 /Question9_Output/part-r-00009
-rw-r--r-- 1 ebd supergroup 134429301 2021-04-22 02:13 /Question9_Output/part-r-00010
-rw-r--r-- 1 ebd supergroup          0 2021-04-22 02:13 /Question9_Output/part-r-00011
-rw-r--r-- 1 ebd supergroup 103174187 2021-04-22 02:13 /Question9_Output/part-r-00012
-rw-r--r-- 1 ebd supergroup          0 2021-04-22 02:13 /Question9_Output/part-r-00013
-rw-r--r-- 1 ebd supergroup          0 2021-04-22 02:13 /Question9_Output/part-r-00014
-rw-r--r-- 1 ebd supergroup 61600206 2021-04-22 02:13 /Question9_Output/part-r-00015
-rw-r--r-- 1 ebd supergroup 77612648 2021-04-22 02:13 /Question9_Output/part-r-00016
-rw-r--r-- 1 ebd supergroup 78939134 2021-04-22 02:14 /Question9_Output/part-r-00017
-rw-r--r-- 1 ebd supergroup 140226552 2021-04-22 02:14 /Question9_Output/part-r-00018
-rw-r--r-- 1 ebd supergroup          0 2021-04-22 02:13 /Question9_Output/part-r-00019
-rw-r--r-- 1 ebd supergroup 146433867 2021-04-22 02:14 /Question9_Output/part-r-00020
-rw-r--r-- 1 ebd supergroup 56669995 2021-04-22 02:14 /Question9_Output/part-r-00021
-rw-r--r-- 1 ebd supergroup 59168084 2021-04-22 02:14 /Question9_Output/part-r-00022
ebd@ebd-VirtualBox:~$
```

Output of individual file:



```
ebd@ebd-VirtualBox:~$ 
ebd@ebd-VirtualBox:~$ 
ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question9_Output/part-r-00000 | head -n 5 | grep West-Southwest
West-Southwest A-3512969,Bing,,1,2020-06-22 15:52:18,2020-06-22 16:07:18,39.77603,-84.28028,39.77603,-84.2802
8,0.0,At Free Pike - Accident.,,Little Richmond Rd,R,Dayton,Montgomery,OH,45426,US,US/Eastern,KDAY,2020-06-22
15:56:00,85.0,85.0,53.0,28.84,10.0,WSW,8.0,0.0,Mostly Cloudy,False,False,False,False,False,False,False,F
alse,False,True,False,Day,Day,Day
West-Southwest A-3675937,Bing,,2,2020-03-05 14:24:00,2020-03-05 15:42:32,34.030837,-118.391136,34.030837,-118
.391136,0.0,At Robertson Blvd/Exit 6 - Accident.,,S Robertson Blvd,R,Los Angeles,Los Angeles,CA,90034,US,US/Pa
cific,KSMO,2020-03-05 14:51:00,62.0,62.0,70.0,29.87,10.0,WSW,9.0,0.0,Fair,False,False,False,False,False,
False,False,False,False,False,Day,Day,Day
West-Southwest A-3675982,Bing,,2,2020-03-05 15:48:06,2020-03-05 16:40:46,34.02196,-117.813129,34.02196,-117.8
13129,0.0,At Diamond Bar Blvd - Accident.,,CA-57 N,R,Diamond Bar,Los Angeles,CA,91765,US,US/Pacific,KPOC,2020-
03-05 15:47:00,79.0,79.0,34.0,28.97,10.0,WSW,9.0,,Fair,False,False,False,False,False,False,F
alse,False,False,Day,Day,Day
West-Southwest A-3675997,Bing,,2,2020-03-05 15:58:00,2020-03-05 20:52:32,34.041757,-118.567637,34.041757,-118
.567637,0.0,At Sunset Blvd - Accident.,,17992.0, Pacific Coast Hwy,L,Pacific Palisades,Los Angeles,CA,90272,US,
US/Pacific,KSMO,2020-03-05 15:51:00,63.0,63.0,70.0,29.86,10.0,WSW,9.0,0.0,Fair,False,False,False,F
alse,False,False,False,False,Day,Day,Day
West-Southwest A-3676018,Bing,,2,2020-03-05 16:44:00,2020-03-05 17:16:47,36.593956,-121.872867,36.593956,-121
.872867,0.0,At CA-68/Fremont St - Accident.,,CA-1 N,R,Monterey,Monterey,CA,93940,US,US/Pacific,KMRY,2020-03-05
16:54:00,59.0,59.0,72.0,29.85,10.0,WSW,13.0,0.0,Fair,False,False,False,False,False,F
alse,False,False,Day,Day,Day
cat: Unable to write to output stream.
ebd@ebd-VirtualBox:~$
```

10. Cities present in each County

Description: This MapReduce Job performs '**Invert Index Algorithm**' where we find out all the cities corresponding to each county from our data.

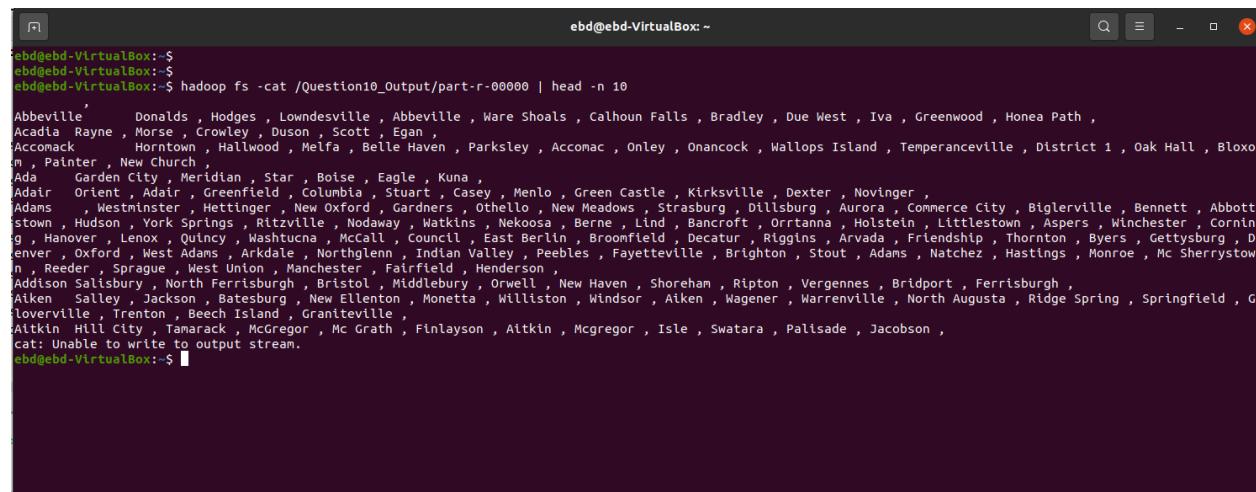
Analysis: We could analyze the number of cities where traffic accidents have taken place, that is captured in our data which is grouped under corresponding County.

Command to execute the jar:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficA  
ccidentsProject-1.0-SNAPSHOT.jar Question10.DriverClass /US_Accidents_Dec20.csv  
/Question10_Output
```

Output Format:

County City1, City2, City3, City4.... CityN



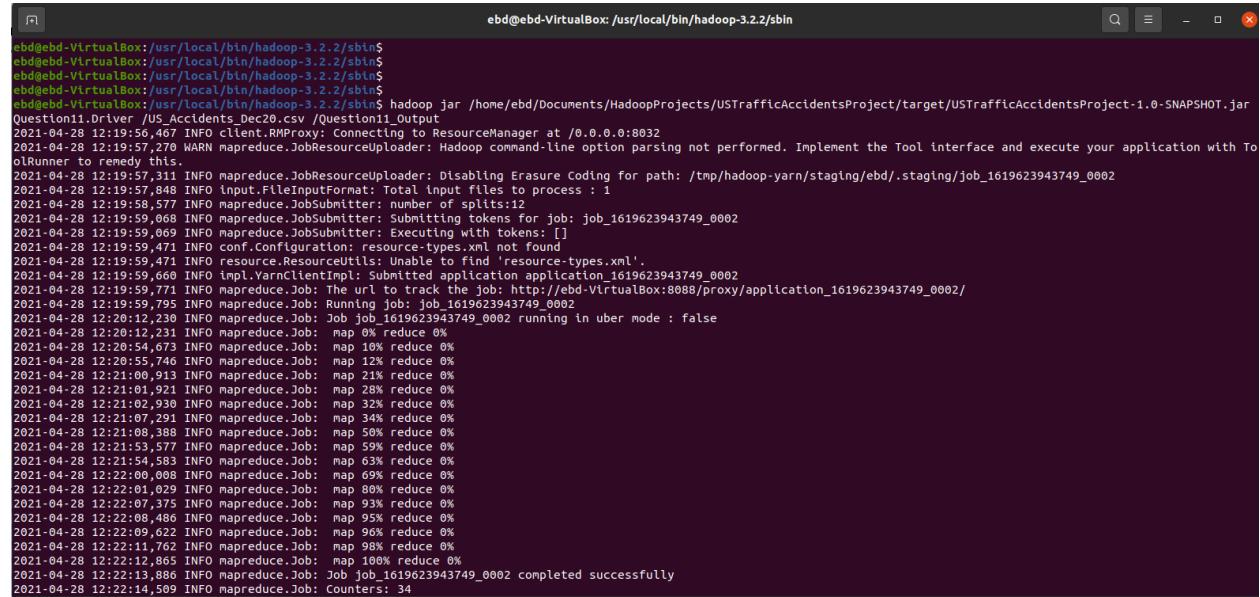
```
ebd@ebd-VirtualBox:~  
ebd@ebd-VirtualBox:~$ hadoop fs -cat /Question10_Output/part-r-00000 | head -n 10  
Abbeville , Donalds , Hodges , Lowndesville , Abbeville , Ware Shoals , Calhoun Falls , Bradley , Due West , Iva , Greenwood , Honea Path ,  
Acadia Rayne , Morse , Crowley , Duson , Scott , Egan ,  
Accomack Hornetown , Hallwood , Melfa , Belle Haven , Parksley , Accomac , Onley , Onancock , Wallops Island , Temperanceville , District 1 , Oak Hall , Blox  
om , Painter , New Church ,  
Ada Gardet City , Meridian , Star , Boise , Eagle , Kuna ,  
Adair Orient , Adair , Greenfield , Columbia , Stuart , Casey , Menlo , Green Castle , Kirksville , Dexter , Novinger ,  
Adams , Westminster , Hettinger , New Oxford , Gardners , Othello , New Meadows , Strasburg , Dillsburg , Aurora , Commerce City , Biglerville , Bennett , Abbott  
stown , Hudson , York Springs , Ritzville , Nodaway , Watkins , Nekoosa , Berne , Lind , Bancroft , Decatur , Riggins , Arvada , Friendship , Thornton , Byers , Gettysburg , D  
enver , Oxford , Lenox , Quincy , Washucna , McCall , Council , East Berlin , Broomfield , Decatur , Riggins , Arvada , Friendship , Thornton , Byers , Gettysburg , D  
enver , Oxford , West Adams , Arkdale , Northglenn , Indian Valley , Peebles , Fayetteville , Brighton , Stout , Adams , Natchez , Hastings , Monroe , Mc Sherrystow  
n , Reeder , Sprague , West Union , Manchester , Fairfield , Henderson ,  
Addison Salisbury , North Ferrisburgh , Bristol , Middlebury , Orwell , New Haven , Shoreham , Ripton , Vergennes , Bridport , Ferrisburgh ,  
Aiken Salley , Jackson , Batesburg , New Ellenton , Monetta , Williston , Windsor , Aiken , Wagener , Warrenton , North Augusta , Ridge Spring , Springfield , G  
loverville , Trenton , Beech Island , Graniteville ,  
Aitkin Hill City , Tamarack , McGregor , Mc Grath , Finlayson , Aitkin , McGregor , Isle , Swatara , Palisade , Jacobson ,  
cat: Unable to write to output stream.  
ebd@ebd-VirtualBox:~$
```

11. Binning based on Accidents on each side of the road (Left/Right)

Description: This MapReduce Job performs ‘Binning’ that partitions the data based on accidents occurred on each side of the road

Command to execute the jar:

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficA  
ccidentsProject-1.0-SNAPSHOT.jar Question11.Driver /US_Accidents_Dec20.csv  
/Question11_Output
```



```
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$  
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$  
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$  
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$  
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$ hadoop jar /home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsProject-1.0-SNAPSHOT.jar Question11.Driver /US_Accidents_Dec20.csv /Question11_Output  
2021-04-28 12:19:56.467 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
2021-04-28 12:19:57.270 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.  
2021-04-28 12:19:57.311 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/ebd/.staging/job_1619623943749_0002  
2021-04-28 12:19:57.467 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
2021-04-28 12:19:57.844 INFO input.FileInputFormat: Total input files to process : 1  
2021-04-28 12:19:58.577 INFO mapreduce.JobSubmitter: number of splits:12  
2021-04-28 12:19:59.068 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1619623943749_0002  
2021-04-28 12:19:59.069 INFO mapreduce.JobSubmitter: Executing with tokens: []  
2021-04-28 12:19:59.471 INFO conf.Configuration: resource-types.xml not found  
2021-04-28 12:19:59.660 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.  
2021-04-28 12:19:59.771 INFO impl.YarnClientImpl: Submitted application application_1619623943749_0002  
2021-04-28 12:19:59.795 INFO mapreduce.Job: The url to track the job: http://ebd-VirtualBox:8088/proxy/application_1619623943749_0002/  
2021-04-28 12:20:12.230 INFO mapreduce.Job: Running job: job_1619623943749_0002 running in uber mode : false  
2021-04-28 12:20:12.231 INFO mapreduce.Job: map 0% reduce 0%  
2021-04-28 12:20:54.673 INFO mapreduce.Job: map 10% reduce 0%  
2021-04-28 12:20:55.746 INFO mapreduce.Job: map 12% reduce 0%  
2021-04-28 12:21:00.913 INFO mapreduce.Job: map 21% reduce 0%  
2021-04-28 12:21:01.921 INFO mapreduce.Job: map 28% reduce 0%  
2021-04-28 12:21:02.930 INFO mapreduce.Job: map 32% reduce 0%  
2021-04-28 12:21:07.291 INFO mapreduce.Job: map 34% reduce 0%  
2021-04-28 12:21:08.388 INFO mapreduce.Job: map 50% reduce 0%  
2021-04-28 12:21:53.577 INFO Mapreduce.Job: map 59% reduce 0%  
2021-04-28 12:21:54.583 INFO Mapreduce.Job: map 63% reduce 0%  
2021-04-28 12:22:00.068 INFO Mapreduce.Job: map 69% reduce 0%  
2021-04-28 12:22:01.029 INFO Mapreduce.Job: map 80% reduce 0%  
2021-04-28 12:22:07.375 INFO Mapreduce.Job: map 93% reduce 0%  
2021-04-28 12:22:08.480 INFO Mapreduce.Job: map 95% reduce 0%  
2021-04-28 12:22:09.622 INFO Mapreduce.Job: map 96% reduce 0%  
2021-04-28 12:22:11.762 INFO Mapreduce.Job: map 98% reduce 0%  
2021-04-28 12:22:12.865 INFO Mapreduce.Job: Map 100% reduce 0%  
2021-04-28 12:22:13.886 INFO Mapreduce.Job: Job job_1619623943749_0002 completed successfully  
2021-04-28 12:22:14.589 INFO mapreduce.Job: Counters: 34
```

```
[+] ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$ hadoop fs -ls /Question11_Output
Found 37 items
-rw-r--r-- 1 ebd supergroup 16847492 2021-04-28 12:41 /Question11_Output/LeftSide-m-00000
-rw-r--r-- 1 ebd supergroup 23360480 2021-04-28 12:41 /Question11_Output/LeftSide-m-00001
-rw-r--r-- 1 ebd supergroup 25034131 2021-04-28 12:41 /Question11_Output/LeftSide-m-00002
-rw-r--r-- 1 ebd supergroup 26735377 2021-04-28 12:41 /Question11_Output/LeftSide-m-00003
-rw-r--r-- 1 ebd supergroup 28178590 2021-04-28 12:41 /Question11_Output/LeftSide-m-00004
-rw-r--r-- 1 ebd supergroup 27948758 2021-04-28 12:41 /Question11_Output/LeftSide-m-00005
-rw-r--r-- 1 ebd supergroup 266070647 2021-04-28 12:42 /Question11_Output/LeftSide-m-00006
-rw-r--r-- 1 ebd supergroup 26263995 2021-04-28 12:42 /Question11_Output/LeftSide-m-00007
-rw-r--r-- 1 ebd supergroup 24962239 2021-04-28 12:42 /Question11_Output/LeftSide-m-00008
-rw-r--r-- 1 ebd supergroup 21584699 2021-04-28 12:42 /Question11_Output/LeftSide-m-00009
-rw-r--r-- 1 ebd supergroup 23197270 2021-04-28 12:42 /Question11_Output/LeftSide-m-00010
-rw-r--r-- 1 ebd supergroup 20015002 2021-04-28 12:42 /Question11_Output/LeftSide-m-00011
-rw-r--r-- 1 ebd supergroup 122134127 2021-04-28 12:41 /Question11_Output/RightSide-m-00000
-rw-r--r-- 1 ebd supergroup 110856403 2021-04-28 12:41 /Question11_Output/RightSide-m-00001
-rw-r--r-- 1 ebd supergroup 109183950 2021-04-28 12:41 /Question11_Output/RightSide-m-00002
-rw-r--r-- 1 ebd supergroup 107482168 2021-04-28 12:41 /Question11_Output/RightSide-m-00003
-rw-r--r-- 1 ebd supergroup 106039279 2021-04-28 12:41 /Question11_Output/RightSide-m-00004
-rw-r--r-- 1 ebd supergroup 106268747 2021-04-28 12:41 /Question11_Output/RightSide-m-00005
-rw-r--r-- 1 ebd supergroup 108147326 2021-04-28 12:42 /Question11_Output/RightSide-m-00006
-rw-r--r-- 1 ebd supergroup 107953658 2021-04-28 12:42 /Question11_Output/RightSide-m-00007
-rw-r--r-- 1 ebd supergroup 109255213 2021-04-28 12:42 /Question11_Output/RightSide-m-00008
-rw-r--r-- 1 ebd supergroup 112633057 2021-04-28 12:42 /Question11_Output/RightSide-m-00009
-rw-r--r-- 1 ebd supergroup 111020565 2021-04-28 12:42 /Question11_Output/RightSide-m-00010
-rw-r--r-- 1 ebd supergroup 114202771 2021-04-28 12:42 /Question11_Output/RightSide-m-00011
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:42 /Question11_Output/_SUCCESS
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:41 /Question11_Output/part-m-00000
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:41 /Question11_Output/part-m-00001
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:41 /Question11_Output/part-m-00002
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:41 /Question11_Output/part-m-00003
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:41 /Question11_Output/part-m-00004
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:41 /Question11_Output/part-m-00005
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:42 /Question11_Output/part-m-00006
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:42 /Question11_Output/part-m-00007
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:42 /Question11_Output/part-m-00008
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:42 /Question11_Output/part-m-00009
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:42 /Question11_Output/part-m-00010
-rw-r--r-- 1 ebd supergroup 0 2021-04-28 12:42 /Question11_Output/part-m-00011
```

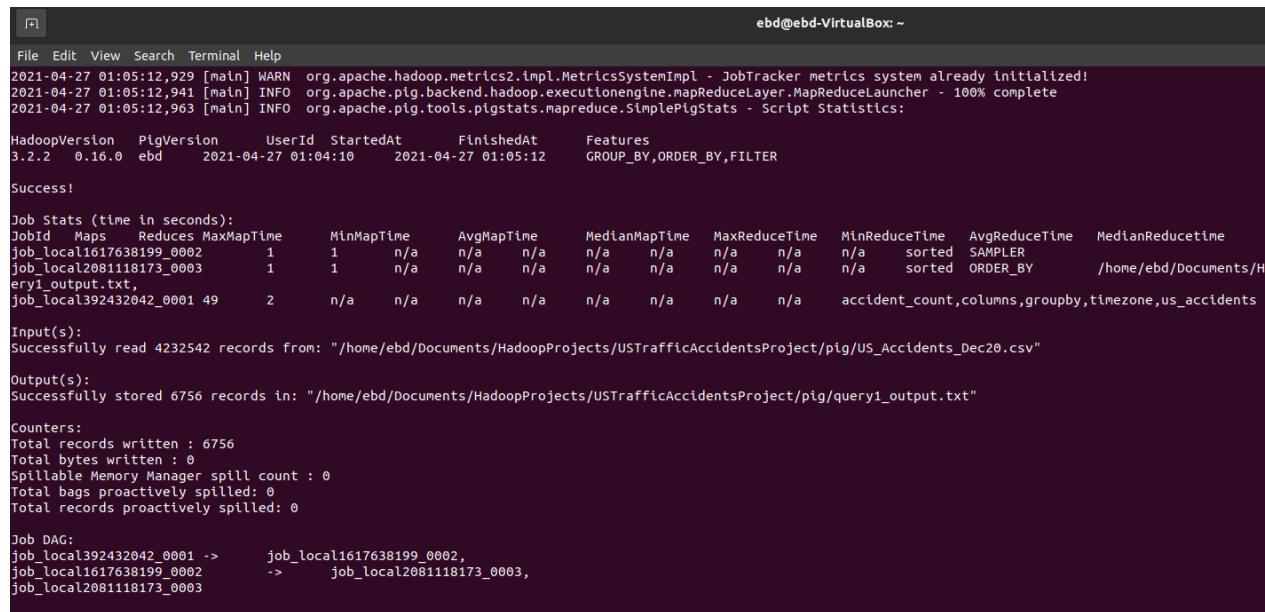
```
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$  
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$  
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$ hadoop fs -head /Question11_Output/LeftSide-m-00000  
A-3866274,Bing,,2,2019-08-31 11:43:00,2019-08-31 12:36:31,38.057425,-121.261317,38.057425,-121.261317,0.0,At CA-99 - Accident.,10901.0,  
N Highway 99,L, Lodi, San Joaquin,CA,95240,US,US/Pacific,KSCK,2019-08-31 11:55:00,89.0,89.0,32.0,29.9,10.0,WNW,7.0,0.0,Fair,False,False,Fa  
lse,False,False,False,False,False,False,False,Day,Day,Day  
A-3866285,Bing,,2,2019-08-31 12:31:00,2019-08-31 13:04:33,38.532699,-121.464689,38.532699,-121.464689,0.0,At 14th Ave/12th Ave Byp - Acc  
ident.,4701.0, Martin Luther King Jr Blvd,L,Sacramento,Sacramento,CA,95820,US,US/Pacific,KSAC,2019-08-31 12:53:00,89.0,89.0,35.0,29.9,10  
.0,SSN,3.0,0.0,Fair,False,False,False,False,False,False,False,Day,Day,Day  
A-3866286,Bing,,2,2019-08-31 12:30:00,2019-08-31 13:22:31,39.420068,-123.807962,39.420068,-123.807962,0.0,At Fort Bragg-Willits Rd - Acc  
ident.,1299.0, S Main St,L,Fort Bragg,Mendocino,CA,95437-5312,US,US/Pacific,KUKI,2019-08-31 12:56:00,91.0,91.0,21.0,29.27,10.0,VAR,3.0,0  
.0,Fair,False,ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$  
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$  
ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$ hadoop fs -head /Question11_Output/RightSide-m-00000  
A-3866272,Bing,,2,2019-08-31 11:38:00,2019-08-31 12:12:33,37.748,-122.15031,37.748,-122.15031,0.0,At 98th Ave/Golf Links Rd - Accident.,  
I-580,W,R,Oakland,Alameda,CA,94605,US,US/Pacific,KOAK,2019-08-31 11:53:00,75.0,75.0,51.0,29.96,10.0,WSW,5.0,0.0,Fair,False,False,Fa  
lse,False,False,False,False,False,False,Day,Day,Day  
A-3866273,Bing,,2,2019-08-31 11:38:00,2019-08-31 13:51:26,38.732177,-121.364635,38.732177,-121.364635,0.0,At Baseline Rd - Accident.,936  
5.0,Walerga Rd,R,Roseville,Placer,CA,95747-9724,US,US/Pacific,KMCC,2019-08-31 11:45:00,84.0,84.0,37.0,29.83,10.0,NW,3.0,,Fair,False,Fa  
lse,False,False,False,False,False,False,Day,Day,Day  
A-3866275,Bing,,2,2019-08-31 11:41:00,2019-08-31 12:54:30,37.359543,-120.64392,37.359543,-120.64392,0.0,At CR-J18/Central Ave/Westside B  
ld - Accident.,CA-99,R,Winton,Medford,CA,95388,US,US/Pacific,KMER,2019-08-31 06:15:00,,,29.71,10.0,CALM,0.0,0.0,Fair,False,Fa  
lse,ebd@ebd-VirtualBox:/usr/local/bin/hadoop-3.2.2/sbin$
```

12. Implementing Data Analysis using Apache Pig

a) Accidents in cities belonging to US/Eastern Time zone



```
query1.pig
Plugins supporting *.pig files found.
1 us_accidents = LOAD '/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/pig/US_Accidents_Dec20.csv' USING PigStorage(',') AS (ID:CHARARRAY, Source:CHARARRA' 23 ^ v i
2
3 columns = FOREACH us_accidents GENERATE $20 as Timezone, $15 as City ;
4
5 timezone = FILTER columns BY NOT City == '' AND Timezone == 'US/Eastern';
6
7 groupby = GROUP timezone BY CONCAT (Timezone, City);
8
9 accident_count = FOREACH groupby GENERATE group, COUNT(timezone) AS count;
10
11 sorted = ORDER accident_count BY count DESC;
12
13 -- DUMP sorted;
14
15 STORE sorted INTO '/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/pig/query1_output.txt';
```



```
File Edit View Search Terminal Help
2021-04-27 01:05:12,929 [main] WARN org.apache.hadoop.metrics2.impl.MetricsSystemImpl - JobTracker metrics system already initialized!
2021-04-27 01:05:12,941 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete
2021-04-27 01:05:12,963 [main] INFO org.apache.pig.tools.pigstats.mapreduce.SimplePigStats - Script Statistics:
HadoopVersion PigVersion UserId StartedAt FinishedAt Features
3.2.2 0.16.0 ebd 2021-04-27 01:04:10 2021-04-27 01:05:12 GROUP_BY,ORDER_BY,FILTER

Success!
Job Stats (time in seconds):
Jobid Maps Reduces MaxMapTime MinMapTime AvgMapTime MedianMapTime MaxReduceTime MinReduceTime AvgReduceTime MedianReducetime
job_local1617638199_0002 1 1 n/a n/a n/a n/a n/a n/a sorted SAMPLER
job_local2081118173_0003 1 1 n/a n/a n/a n/a n/a n/a sorted ORDER_BY /home/ebd/Documents/H
erry1_output.txt,
job_local392432042_0001 49 2 n/a n/a n/a n/a n/a n/a accident_count,columns,groupby,timezone,us_accidents

Input(s):
Successfully read 4232542 records from: "/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/pig/US_Accidents_Dec20.csv"

Output(s):
Successfully stored 6756 records in: "/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/pig/query1_output.txt"

Counters:
Total records written : 6756
Total bytes written : 0
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_local392432042_0001 -> job_local1617638199_0002,
job_local1617638199_0002 -> job_local2081118173_0003,
job_local2081118173_0003
```



Rank	City	Accidents
1	US/EasternCharlotte	88874
2	US/EasternMiami	63086
3	US/EasternRaleigh	52871
4	US/EasternAtlanta	46319
5	US/EasternOrlando	39560
6	US/EasternRichmond	24337
7	US/EasternJacksonville	23702
8	US/EasternColumbia	22610
9	US/EasternIndianapolis	22479
10	US/EasternGreenville	22154
11	US/EasternTampa	17345
12	US/EasternRochester	17325
13	US/EasternDayton	16527

b) Counties in CA where accidents happened with poor visibility (Inner Join)

query1.pig query2.pig

Plugins supporting *.pig files found.

Install plugins Ignore extension

22 ^ v

```
1 -- Counties in CA where accidents happened where visibility < 5
2
3 us_accidents = LOAD '/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/pig/US_Accidents_Dec20.csv' USING PigStorage(',') AS (ID:CHARARRAY, Source:CHARARRAY, TMC:DOUB
4
5 columns = FOREACH us_accidents GENERATE ID, State, Visibility;
6
7 county = FOREACH us_accidents GENERATE ID, County;
8
9 state = FILTER columns BY State == 'CA' AND NOT Visibility is null;
10
11 visibility = FILTER state BY Visibility < 5;
12
13 inner_join = JOIN county BY ID, visibility BY ID;
14
15 records = FOREACH inner_join GENERATE State, County, Visibility;
16
17 county_visibility = DISTINCT records;
18
19 STORE county_visibility INTO '/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/pig/query2_output.txt' using PigStorage('');
```

```
[root@ebs ~]# ./pigstats.sh
ebd@ebd-VirtualBox: ~
2021-04-27 11:18:35,347 [main] INFO  org.apache.pig.tools.pigstats.mapreduce.SimplePigStats - Script Statistics:
HadoopVersion  PigVersion      UserId  StartedAt      FinishedAt      Features
3.2.2  0.16.0  ebd    2021-04-27 11:14:16  2021-04-27 11:18:35  HASH_JOIN,DISTINCT,FILTER

Success!

Job Stats (time in seconds):
JobId  Maps   Reduces  MaxMapTime     MinMapTime     AvgMapTime     MedianMapTime   MaxReduceTime  MinReduceTime  AvgReduceTime  MedianReduceTime
job_local368707957_0001 98       4          n/a           n/a           n/a           n/a           n/a           n/a           columns,county,inner_join,records,state,us_accidents
job_local561651534_0002 1        1          n/a           n/a           n/a           n/a           n/a           n/a           n/a           DISTINCT
pig/query2_output.txt,

Input(s):
Successfully read 4232542 records from: "/home/ebd/Documents/HadoopProjects/USTRafficAccidentsProject/pig/US_Accidents_Dec20.csv"
Successfully read 4232542 records from: "/home/ebd/Documents/HadoopProjects/USTRafficAccidentsProject/pig/US_Accidents_Dec20.csv"

Output(s):
Successfully stored 0 records in: "/home/ebd/Documents/HadoopProjects/USTRafficAccidentsProject/pig/query2_output.txt"

Counters:
Total records written : 0
Total bytes written : 0
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Dag DAG:
job_local368707957_0001 ->      job_local561651534_0002,
job_local561651534_0002
```

	CA;Inyo;0.75
1	CA;Inyo;1.0
2	CA;Inyo;1.8
3	CA;Inyo;2.0
4	CA;Inyo;3.0
5	CA;Kern;0.0
6	CA;Kern;0.06
7	CA;Kern;0.2
8	CA;Kern;0.25
9	CA;Kern;0.5
10	CA;Kern;0.75
11	CA;Kern;0.8
12	CA;Kern;1.0
13	CA;Kern;1.2
14	CA;Kern;1.5

Data Analysis using Hive

Creating and loading data

```
CREATE TABLE us_accidents(ID STRING, Source STRING, TMC DOUBLE, Severity INT, Start_Time STRING, End_Time STRING, Start_Lat DOUBLE, Start_Lng DOUBLE, End_Lat DOUBLE, End_Lng DOUBLE, Distance DOUBLE, Description STRING, Number DOUBLE, Street STRING, Side STRING, City STRING, County STRING, State STRING, Zipcode STRING, Country STRING, Timezone STRING, Airport_Code STRING, Weather_Timestamp STRING, Temperature DOUBLE, Wind_Chill DOUBLE, Humidity DOUBLE, Pressure DOUBLE, Visibility DOUBLE, Wind_Direction STRING, Wind_Speed DOUBLE, Precipitation DOUBLE, Weather_Condition STRING, Amenity BOOLEAN, Bump BOOLEAN, Crossing BOOLEAN, Give_Way BOOLEAN, Junction BOOLEAN, No_Exit BOOLEAN, Railway BOOLEAN, Roundabout BOOLEAN, Station BOOLEAN, Stop BOOLEAN, Traffic_Calming BOOLEAN, Traffic_Signal BOOLEAN, Turning_Loop BOOLEAN, Sunrise_Sunset STRING, Civil_Twilight STRING, Nautical_Twilight STRING, Astronomical_Twilight STRING) row format delimited fields terminated by ',';

CREATE TABLE Wind_Direction(Wind_Direction STRING, Description STRING) row format delimited fields terminated by ',';
```

```
LOAD DATA INPATH '/US_Accidents_Dec20.csv' OVERWRITE INTO TABLE us_accidents;
LOAD DATA INPATH '/WindDirection.csv' OVERWRITE INTO TABLE Wind_Direction;
```

Analysis Queries

```
-- Cities in the State of Ohio which caused delays on a rainy day

SELECT City, COUNT(*) AS NumberOfAccidents FROM us_accidents WHERE
CHARINDEX('delays',Description) > 0 AND City <> '' AND Weather_Condition='Heavy Rain' AND
State='OH' GROUP BY City;

-- Inner join based on Wind Direction

SELECT a.Wind_Direction, b.Description, a.Wind_Speed FROM us_accidents a JOIN Wind_Direction b
ON (a.Wind_Direction=b.Wind_Direction) WHERE a.Wind_Direction <> ''

-- Maximum Temperature for each Weather Condition during accidents

SELECT Weather_Condition, MAX(Temperature) FROM us_accidents WHERE Weather_Condition <> ''
GROUP BY Weather_Condition ORDER BY MAX(Temperature) DESC;
```

Consolidated shell script to run all the map reduce jobs:

```
#!/bin/bash
```

```
# Copying the data to HDFS
```

```
hadoop fs -put ~/Downloads/US_Accidents_Dec20.csv /  
hadoop fs -put ~/Downloads/WindDirection.csv /
```

```
# Simple MR Job: Number of Traffic Accidents occurred by State
```

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question1.DriverClass /US_Accidents_Dec20.csv /Question1_Output
```

```
# Simple MR Job: Number of Accidents near Junction VS not a Junction
```

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question2.DriverClass /US_Accidents_Dec20.csv /Question2_Output
```

```
# Numerical Summarization: Average, Min, Max Visibility (miles), Max StartDate Per Accident  
Severity
```

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question3.DriverClass /US_Accidents_Dec20.csv /Question3_Output
```

```
# Numerical Summarization: Most Recent (Max) Accident Date Per City
```

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question4.DriverClass /US_Accidents_Dec20.csv /Question4_Output
```

```
# Numerical Summarization: Percentage of Accidents in each Month
```

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question5.DriverClass /US_Accidents_Dec20.csv /Question5_Output
```

Top ‘N’ Filtering Pattern: Top 10 States with Highest Number of Accidents

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question6.DriverClass /Question1_Output/part-r-00000  
/Question6_Output
```

Top ‘N’ Filtering Pattern: Top 5 Accident Conducive Weather Conditions

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question7.a.DriverClass /US_Accidents_Dec20.csv  
/Question7a_Output
```

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question7.b.DriverClass /Question7a_Output/part-r-00000  
/Question7b_Output
```

Secondary Sorting: Number of Accidents Per County Per Year

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question8.DriverClass /US_Accidents_Dec20.csv /Question8_Output
```

Partitioning: Partitioning based on Wind Directions

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question9.DriverClass /US_Accidents_Dec20.csv /WindDirection.csv  
/Question9_Output
```

Inverted Index Pattern: Cities present in each County

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question10.DriverClass /US_Accidents_Dec20.csv  
/Question10_Output
```

Binning: Binning based on Accidents on each side of the road (Left/Right)

```
hadoop jar  
/home/ebd/Documents/HadoopProjects/USTrafficAccidentsProject/target/USTrafficAccidentsPr  
oject-1.0-SNAPSHOT.jar Question11.Driver /US_Accidents_Dec20.csv /Question11_Output
```

Appendix

Simple MR Job: Number of Traffic Accidents occurred by State



```
16
17  ► public class DriverClass {
18  ► @ |   public static void main(String args[]) throws IOException, ClassNotFoundException, InterruptedException {
19    // Create a new Job
20    Configuration conf = new Configuration();
21    Job job = Job.getInstance(conf, "AccidentsPerState");
22    job.setJarByClass(DriverClass.class);
23
24    //Telling the program which is mapper/reducer class
25    job.setMapperClass(MapperClass.class);
26    job.setReducerClass(ReducerClass.class);
27
28    //Setting the InputFormat (What the data will be like)
29    job.setInputFormatClass(TextInputFormat.class);
30    job.setOutputFormatClass(TextOutputFormat.class);
31
32    job.setOutputKeyClass(Text.class);
33    job.setOutputValueClass(LongWritable.class);
34
35    FileSystem fs = FileSystem.get(job.getConfiguration());
36
37    //Passing the file on which word count will be performed
38    //Setting the input and output paths
39    Path outDir = new Path(args[1]);
40    FileOutputFormat.setOutputPath(job, outDir);
41    Path inDir = new Path(args[0]);
42    FileInputFormat.addInputPath(job, inDir);
43
44    //Delete the output directory if it exists before execution
45    if(fs.exists(outDir)){
46      fs.delete(outDir, b: true);
47    }
48
49    System.exit(job.waitForCompletion( verbose: true) ? 1 : 0);
50
```

```

⑥ MapperClass.java ×
1 import org.apache.hadoop.io.LongWritable;
2 import org.apache.hadoop.io.Text;
3 import org.apache.hadoop.mapreduce.Mapper;
4
5
6
7 import java.io.IOException;
8
9 public class MapperClass extends Mapper<LongWritable, Text, Text, LongWritable> {
10
11     Text state = new Text();
12     LongWritable count = new LongWritable( value: 1);
13
14     @Override
15     protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
16
17         //Convert the each line which is comma separated into String array
18         String line = value.toString();
19         String[] tokens = line.split( regex: "," );
20
21         //Check if the first column is ID
22         if(!tokens[0].equals("ID")) {
23
24             //set column 17 to Text variable
25             state.set(tokens[17]);
26         }
27
28         //The mapper will emit state key and 1 as count
29         context.write(state, count);
30     }
31 }

```

```

⑦ ReducerClass.java ×
1 import org.apache.hadoop.io.LongWritable;
2 import org.apache.hadoop.io.Text;
3 import org.apache.hadoop.mapreduce.Reducer;
4
5
6
7 import java.io.IOException;
8
9 public class ReducerClass extends Reducer<Text, LongWritable, Text, LongWritable> {
10
11     LongWritable count = new LongWritable();
12
13     @Override
14     protected void reduce(Text key, Iterable<LongWritable> values, Context context) throws IOException, InterruptedException {
15
16         long sum = 0;
17
18         // Iterating through each values and adding
19         for(LongWritable val: values){
20             sum+=val.get();
21         }
22
23         // Converting primitive Long to Hadoop LongWritable
24         count.set(sum);
25
26         // Emitting out state and aggregate count
27         context.write(key, count);
28
29     }
30 }

```

Simple MR Job: Number of Accidents near Junction VS not a Junction

The screenshot shows an IDE interface with two code editors. The top editor is for `DriverClass.java` and the bottom editor is for `MapperClass.java`. Both files are displayed in a monospaced font.

```
DriverClass.java:
```

```
18 public static void main(String[] args) throws IOException, InterruptedException, ClassNotFoundException {
19
20     // Create a new Job
21     Configuration conf = new Configuration();
22     Job job = Job.getInstance(conf, "AccidentsNearJunction");
23     job.setJarByClass(DriverClass.class);
24
25     //Telling the program which is mapper/reducer class
26     job.setMapperClass(MapperClass.class);
27     job.setReducerClass(ReducerClass.class);
28
29     //Setting the InputFormat (What the data will be like)
30     job.setInputFormatClass(TextInputFormat.class);
31     job.setOutputFormatClass(TextOutputFormat.class);
32
33     job.setOutputKeyClass(Text.class);
34     job.setOutputValueClass(LongWritable.class);
35
36     job.setNumReduceTasks(1);
37
38     FileSystem fs = FileSystem.get(job.getConfiguration());
39
40     //Setting the input and output paths
41     Path outDir = new Path(args[1]);
42     FileOutputFormat.setOutputPath(job, outDir);
43     Path inDir = new Path(args[0]);
44     FileInputFormat.addInputPath(job, inDir);
45
46     //Delete the output directory if it exists before execution
47     if(fs.exists(outDir)){
48         fs.delete(outDir, true);
49     }
50
51     System.exit(job.waitForCompletion( verbose: true ) ? 1 : 0);
52 }
```

```
MapperClass.java:
```

```
9 public class MapperClass extends Mapper<LongWritable, Text, Text, LongWritable> {
10
11     Text junction = new Text();
12     LongWritable count = new LongWritable( value: 1 );
13     JunctionClass junctionMap = new JunctionClass();
14
15     @Override
16     protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
17
18         //Convert the each line which is comma separated into String array
19         String line = value.toString();
20         String[] tokens = line.split( regex: "," );
21         String junct = "";
22
23         //Check if the first column is ID
24         if(!tokens[0].equals("ID")) {
25
26             //The object of class Junction Class fetches the method that maps true --> Near Junction and false --> Not Near Junction
27             junct = junctionMap.getJunctionMethod(tokens[36]);
28
29         }
30         //Converting the above mapped object to Text datatype
31         junction.set(junct);
32
33         //The mapper will emit state key and 1 as count
34         context.write(junction, count);
35     }
36 }
37 }
```

```
⑥ ReducerClass.java ×
1 package Question2;
2
3 import org.apache.hadoop.io.LongWritable;
4 import org.apache.hadoop.io.Text;
5 import org.apache.hadoop.mapreduce.Reducer;
6
7 import java.io.IOException;
8
9 public class ReducerClass extends Reducer<Text, LongWritable, Text, LongWritable> {
10
11     LongWritable count = new LongWritable();
12
13     @Override
14     protected void reduce(Text key, Iterable<LongWritable> values, Context context) throws IOException, InterruptedException {
15
16         long sum = 0;
17
18         // Iterating through each values and adding
19         for(LongWritable val: values){
20             sum+=val.get();
21         }
22
23         // Converting primitive Long to Hadoop LongWritable
24         count.set(sum);
25
26         // Emitting out state and count
27         context.write(key, count);
28     }
29 }
```

```
⑥ JunctionClass.java ×
1 package Question2;
2
3 import java.util.Map;
4 import java.util.HashMap;
5
6 public class JunctionClass {
7
8     //Creating a static final Map (key-value) so that one copy of variable exists and can't be reinitialize
9     public static final Map<String, String> JUNCTION_MAP;
10
11     //Creating an immutable map using a static initializer
12     static {
13         JUNCTION_MAP = new HashMap<String, String>();
14         JUNCTION_MAP.put( k: "True", v: "Accidents Near Junction");
15         JUNCTION_MAP.put( k: "False", v: "Accidents Not Near Junction");
16         JUNCTION_MAP.put( k: "", v: "Unknown");
17     }
18
19     //Returns the mapped value for each key (true/false)
20     public String getJunctionMethod(String junction) { return JUNCTION_MAP.get(junction); }
21
22
23 }
24
25 }
```

Numerical Summarization: Avg, Min, Max Visibility, Max StartDate Per Accident Severity

```

DriverClass.java
17 ► @
public static void main(String[] args) throws IOException, InterruptedException, ClassNotFoundException {
18
19     // Create a new Job
20     Configuration conf = new Configuration();
21     Job job = Job.getInstance(conf, "AvgMinMaxVisibility");
22     job.setJarByClass(DriverClass.class);
23
24     //Telling the program which is mapper/reducer class
25     job.setMapperClass(MapperClass.class);
26     job.setReducerClass(ReducerClass.class);
27
28     //Setting the InputFormat (What the data will be like)
29     job.setInputFormatClass(TextInputFormat.class);
30
31     //Mapper emits key which is Severity (IntWritable) and value as Custom Writable class
32     job.setMapOutputKeyClass(IntWritable.class);
33     job.setMapOutputValueClass(MinMaxTuple.class);
34
35     job.setOutputKeyClass(IntWritable.class);
36     job.setOutputValueClass(MinMaxTuple.class);
37
38     job.setNumReduceTasks(1);
39
40     //Setting the input and output paths
41     Path outDir = new Path(args[1]);
42     FileOutputFormat.setOutputPath(job, outDir);|
43     Path inDir = new Path(args[0]);
44     FileInputFormat.addInputPath(job, inDir);
45
46     //Delete the output directory if it exists before execution
47     FileSystem fs = FileSystem.get(job.getConfiguration());
48     if(fs.exists(outDir)){
49         fs.delete(outDir, b: true);
50     }
51
52
MapperClass.java
12 public class MapperClass extends Mapper<LongWritable, Text, IntWritable, MinMaxTuple>{
13
14     private final static SimpleDateFormat sdfformat = new SimpleDateFormat(pattern: "yy-MM-dd");
15     IntWritable s = new IntWritable();
16     MinMaxTuple tuple = new MinMaxTuple();
17
18     @Override
19     protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
20
21         //Convert the each line which is comma separated into String array
22         String line = value.toString();
23         String[] tokens = line.split(regex: ",");
24
25
26         if(!tokens[0].equals("ID")){
27
28             //Checks if the Severity and Visibility are null or empty
29             if(tokens[3].isEmpty() || tokens[3].equals(" ") || tokens[27].equals(" ")|| tokens[27].isEmpty() ) {
30                 return;
31             }
32
33             int severity = Integer.parseInt(tokens[3]);
34
35             //Setting the corresponding visibility in the tuple for each key
36             tuple.setMaxVisibility(Float.parseFloat(tokens[27]));
37             tuple.setMinVisibility(Float.parseFloat(tokens[27]));
38             tuple.setAvgVisibility(Float.parseFloat(tokens[27]));
39             tuple.setCount(1);
40
41             try{
42                 tuple.setMaxStartDate(sdfformat.parse(tokens[4]));
43             }catch (ParseException e){
44                 e.printStackTrace();
45             }
46             s.set(severity);
47             context.write(|,tuple);
48         }
49
50

```

```
ReducerClass.java
8
9     public class ReducerClass extends Reducer<IntWritable, MinMaxTuple, IntWritable, MinMaxTuple> {
10
11     MinMaxTuple tuple = new MinMaxTuple();
12     @Override
13     protected void reduce(IntWritable key, Iterable<MinMaxTuple> values, Context context) throws IOException, InterruptedException {
14
15         float maxVisibility = Float.MIN_VALUE;
16         float minVisibility = Float.MAX_VALUE;
17         long count = 0;
18         float countVisibility = 0;
19         Date maxDate = null;
20         //Iterating through all the values (MinMaxTuple) for each key
21         for(MinMaxTuple t: values){
22
23             if(maxDate == null) {
24                 maxDate = t.getMaxStartDate();
25             }
26             countVisibility += t.getAvgVisibility() * t.getCount();
27             count += t.getCount();
28             if(t.getMinVisibility() < minVisibility) {
29                 minVisibility = t.getMinVisibility();
30             }
31             if(t.getMaxVisibility() > maxVisibility) {
32                 maxVisibility = t.getMaxVisibility();
33                 maxDate = t.getMaxStartDate();
34             }
35         }
36         //Setting the aggregated values in the tuple
37         tuple.setAvgVisibility(countVisibility / count);
38         tuple.setMaxVisibility(maxVisibility);
39         tuple.setMinVisibility(minVisibility);
40         tuple.setMaxStartDate(maxDate);
41         tuple.setCount(count);
42
43         context.write(key, tuple);
44     }
}
```

```
c MinMaxTuple.java x
10     public class MinMaxTuple implements Writable {
11
12         private float minVisibility;
13         private float maxVisibility;
14         private float avgVisibility;
15         private long count;
16         private Date maxStartDate;
17
18     public long getCount() { return count; }
19
20
21     public void setCount(long count) { this.count = count; }
22     public float getMinVisibility() { return minVisibility; }
23     public void setMinVisibility(float minVisibility) { this.minVisibility = minVisibility; }
24     public float getMaxVisibility() { return maxVisibility; }
25     public void setMaxVisibility(float maxVisibility) { this.maxVisibility = maxVisibility; }
26     public float getAvgVisibility() { return avgVisibility; }
27     public void setAvgVisibility(float avgVisibility) { this.avgVisibility = avgVisibility; }
28     public Date getMaxStartDate() { return maxStartDate; }
29     public void setMaxStartDate(Date maxStartDate) { this.maxStartDate = maxStartDate; }
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50     //Serialize the fields of this object to out
51     @Override
52     public void write(DataOutput dataOutput) throws IOException {
53         dataOutput.writeFloat(minVisibility);
54         dataOutput.writeFloat(maxVisibility);
55         dataOutput.writeFloat(avgVisibility);
56         dataOutput.writeLong(count);
57         dataOutput.writeLong(maxStartDate.getTime());
58     }
59
60     //Deserialize the fields of this object from in
61     @Override
62     public void readFields(DataInput dataInput) throws IOException {
63         avgVisibility = dataInput.readFloat();
64         minVisibility = dataInput.readFloat();
65         maxVisibility = dataInput.readFloat();
66         count = dataInput.readLong();
67         maxStartDate = new Date(dataInput.readLong());
68
69     }
70
71     @Override
72     public String toString() {
73         return "Minimum Visibility= " + minVisibility +
74             " | Maximum Visibility= " + maxVisibility +
75             " | Average Visibility= " + avgVisibility +
76             " | Maximum Start Date= " + maxStartDate;
77     }
78 }
```

Numerical Summarization: Most Recent (Max) Accident Date Per City

```
DriverClass.java ×
15
16  ► @   public static void main(String[] args) throws IOException, InterruptedException, ClassNotFoundException {
17
18      // Create a new Job
19      Configuration conf = new Configuration();
20      Job job = Job.getInstance(conf, jobName: "LastestAccidentDatePerCity");
21      job.setJarByClass(DriverClass.class);
22
23      //Telling the program which is mapper/reducer class
24      job.setMapperClass(MapperClass.class);
25      job.setReducerClass(ReducerClass.class);
26
27      //Setting the InputFormat (What the data will be like)
28      job.setInputFormatClass(TextInputFormat.class);
29
30      //Mapper emits key which is City (Text) and value as Custom Writable class
31      job.setMapOutputKeyClass(Text.class);
32      job.setMapOutputValueClass(MinMaxTuple.class);
33
34      job.setOutputKeyClass(Text.class);
35      job.setOutputValueClass(MinMaxTuple.class);
36
37      job.setNumReduceTasks(1);
38
39      //Setting the input and output paths
40      Path outDir = new Path(args[1]);
41      FileOutputFormat.setOutputPath(job, outDir);
42      Path inDir = new Path(args[0]);
43      FileInputFormat.addInputPath(job, inDir);
44
45      //Delete the output directory if it exists before execution
46      FileSystem fs = FileSystem.get(job.getConfiguration());
47      if (fs.exists(outDir)) {
48          fs.delete(outDir, b: true);
49      }
50
51      System.exit(job.waitForCompletion( verbose: true) ? 1 : 0);
```

```
④ MapperClass.java ×
3   import ...
9
10  public class MapperClass extends Mapper<LongWritable, Text, Text, MinMaxTuple> {
11
12      private final static SimpleDateFormat sdfformat = new SimpleDateFormat(pattern: "yy-MM-dd");
13      Text city = new Text();
14      MinMaxTuple tuple = new MinMaxTuple();
15
16      @Override
17  protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
18
19          //Convert the each line which is comma separated into String array
20          String line = value.toString();
21          String[] tokens = line.split(regex: ",");
22
23          //Check if the first column is ID
24          if(!tokens[0].equals("ID")){
25
26              //Checks if the City and Date are null or empty
27              if(tokens[15].isEmpty() || tokens[15].equals(" ") || tokens[4].equals(" ")|| tokens[4].isEmpty() ) {
28                  return;
29              }
30              //Setting city
31              String c = tokens[15];
32
33              try{
34                  tuple.setMaxStartDate(sdfformat.parse(tokens[4]));
35              }catch (ParseException e){
36                  e.printStackTrace();
37              }
38
39              city.set(c);
40              context.write(city ,tuple);
41      }

```

```

c ReducerClass.java x
1 package Question4;
2
3 import ...
4
5
6 public class ReducerClass extends Reducer<Text, MinMaxTuple, Text, MinMaxTuple> {
7
8     MinMaxTuple tuple = new MinMaxTuple();
9
10    @Override
11    protected void reduce(Text key, Iterable<MinMaxTuple> values, Context context) throws IOException, InterruptedException {
12
13        Date maxDate = null;
14
15        for(MinMaxTuple t: values){
16
17            if(maxDate == null) {
18                maxDate = t.getMaxStartDate();
19            }
20
21            //Checking the Max date then the previously set
22            if(maxDate.compareTo(t.getMaxStartDate())<=0) {
23                maxDate = t.getMaxStartDate();
24            }
25
26        }
27
28    }
29
30    tuple.setMaxStartDate(maxDate);
31    context.write(key, tuple);
32
33    }
34

```

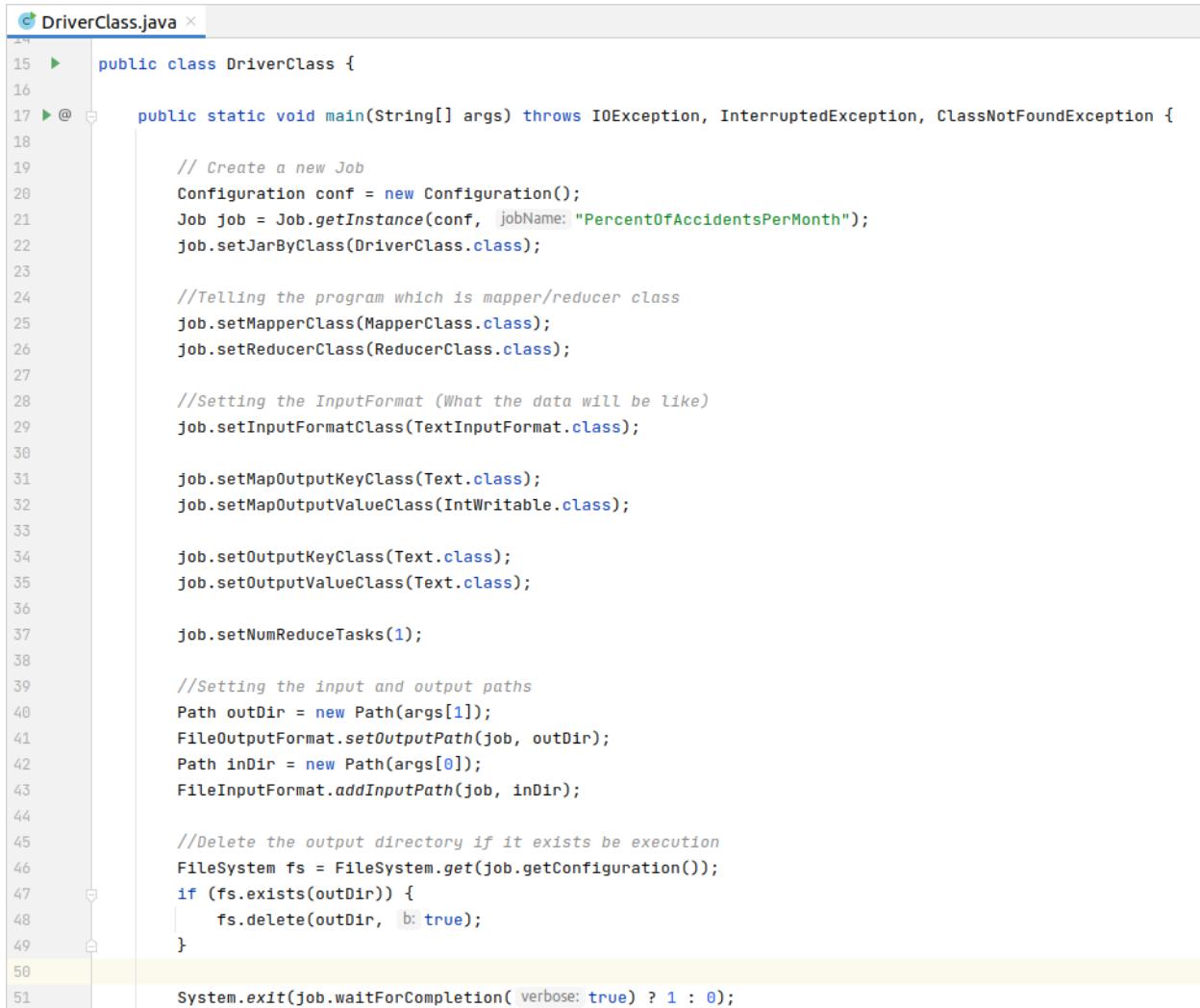


```

c MinMaxTuple.java x
1 package Question4;
2
3 import ...
4
5
6 public class MinMaxTuple implements Writable {
7
8     private Date maxStartDate;
9
10    public Date getMaxStartDate() { return maxStartDate; }
11
12    public void setMaxStartDate(Date maxStartDate) { this.maxStartDate = maxStartDate; }
13
14    @Override
15    public void write(DataOutput dataOutput) throws IOException {
16        dataOutput.writeLong(maxStartDate.getTime());
17    }
18
19    @Override
20    public void readFields(DataInput dataInput) throws IOException {
21        maxStartDate = new Date(dataInput.readLong());
22    }
23
24    @Override
25    public String toString() { return "Latest Accident Occurred On: " + maxStartDate; }
26
27    }
28
29
30
31
32
33
34
35
36
37

```

Numerical Summarization: Percentage of Accidents in each Month



```
DriverClass.java
14
15  public class DriverClass {
16
17  @  public static void main(String[] args) throws IOException, InterruptedException, ClassNotFoundException {
18
19      // Create a new Job
20      Configuration conf = new Configuration();
21      Job job = Job.getInstance(conf, "PercentOfAccidentsPerMonth");
22      job.setJarByClass(DriverClass.class);
23
24      //Telling the program which is mapper/reducer class
25      job.setMapperClass(MapperClass.class);
26      job.setReducerClass(ReducerClass.class);
27
28      //Setting the InputFormat (What the data will be like)
29      job.setInputFormatClass(TextInputFormat.class);
30
31      job.setMapOutputKeyClass(Text.class);
32      job.setMapOutputValueClass(IntWritable.class);
33
34      job.setOutputKeyClass(Text.class);
35      job.setOutputValueClass(Text.class);
36
37      job.setNumReduceTasks(1);
38
39      //Setting the input and output paths
40      Path outDir = new Path(args[1]);
41      FileOutputFormat.setOutputPath(job, outDir);
42      Path inDir = new Path(args[0]);
43      FileInputFormat.addInputPath(job, inDir);
44
45      //Delete the output directory if it exists be execution
46      FileSystem fs = FileSystem.get(job.getConfiguration());
47      if (fs.exists(outDir)) {
48          fs.delete(outDir, true);
49      }
50
51      System.exit(job.waitForCompletion( verbose? 1 : 0));
```

```

⑥ MapperClass.java ×
8 import java.io.IOException;
9 import java.text.ParseException;
10 import java.text.SimpleDateFormat;
11 import java.util.Date;
12
13 public class MapperClass extends Mapper<LongWritable, Text, Text, IntWritable> {
14
15     Date date = new Date();
16     Text month = new Text();
17     IntWritable one = new IntWritable( value: 1);
18
19     SimpleDateFormat sdfformat = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss");
20     SimpleDateFormat monthformat = new SimpleDateFormat( pattern: "MMMMM");
21
22     @Override
23     protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
24
25         //Convert the each line which is comma separated into String array
26         String line = value.toString();
27         String[] tokens = line.split( regex: ",");
28
29         //Check if the first column is ID
30         if(tokens[0].equals("ID")) {
31             String startDate = tokens[4];
32
33             try {
34                 date = sdfformat.parse(startDate);
35             } catch (ParseException e) {
36                 e.printStackTrace();
37             }
38
39             month.set(monthformat.format(date));
40             context.write(month, one);
41         }
42     }
}

⑦ ReducerClass.java ×
1 package Question5;
2
3 import ...
4
5 public class ReducerClass extends Reducer<Text, IntWritable, Text, Text> {
6
7     Text percent = new Text();
8
9     @Override
10    protected void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {
11
12        // The total length of records in the dataset
13        int total = 3513617;
14        int sum = 0;
15
16        for(IntWritable v : values) {
17            sum += v.get();
18        }
19
20        //Calculating the percentage
21        double percentage = ((double) sum / total) * 100;
22        //Formatting into a string
23        percent.set(String.format("%.2f", percentage) + "%");
24
25        context.write(key, percent);
26
27    }
28
29 }
30

```

Top ‘N’ Filtering Pattern: Top 10 States with Highest Number of Accidents

```
DriverClass.java x
17 public static void main(String[] args) throws IOException, InterruptedException, ClassNotFoundException {
18
19     // Create a new Job
20     Configuration conf = new Configuration();
21     Job job = Job.getInstance(conf, "Top10AccidentStates");
22     job.setJarByClass(DriverClass.class);
23
24     //Telling the program which is mapper/reducer class
25     job.setMapperClass(MapperClass.class);
26     job.setReducerClass(ReducerClass.class);
27
28     //Setting the InputFormat (What the data will be like)
29     job.setInputFormatClass(TextInputFormat.class);
30
31     //We use nullWritable class as key
32     job.setMapOutputKeyClass(NullWritable.class);
33     job.setMapOutputValueClass(Text.class);
34
35     job.setOutputKeyClass(NullWritable.class);
36     job.setOutputValueClass(Text.class);
37
38     job.setNumReduceTasks(1);
39
40     //Setting the input and output paths
41     Path outDir = new Path(args[1]);
42     FileOutputFormat.setOutputPath(job, outDir);
43     Path inDir = new Path(args[0]);
44     FileInputFormat.addInputPath(job, inDir);
45
46     //Delete the output directory if it exists before execution
47     FileSystem fs = FileSystem.get(job.getConfiguration());
48     if (fs.exists(outDir)) {
49         fs.delete(outDir, true);
50     }
51
52     System.exit(job.waitForCompletion( verbose: true ) ? 1 : 0);

```

```
MapperClass.java x
1 package Question6;
2
3 import org.apache.hadoop.io.NullWritable;
4 import org.apache.hadoop.io.Text;
5 import org.apache.hadoop.mapreduce.Mapper;
6
7 import java.io.IOException;
8
9 public class MapperClass extends Mapper<Object, Text, NullWritable, Text> {
10
11     @Override
12     protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {
13
14         context.write(NullWritable.get(), value);
15     }
16 }
17
```

```
ReducerClass.java ×
13     public class ReducerClass extends Reducer<NullWritable, Text, NullWritable, Text> {
14
15         static class SortingClass implements Comparator<Integer> {
16
17             @Override
18             public int compare(Integer o1, Integer o2) {
19                 return o2.compareTo(o1);
20             }
21         }
22
23         //TreeMap save the data of top n
24         private TreeMap<Integer, Text> countTreeMap = new TreeMap<>(new SortingClass());
25
26         @Override
27         protected void reduce(NullWritable key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
28
29             for (Text value : values) {
30
31                 //We use the previous formed output from Question1 which is in form "State \t Count"
32                 String line = value.toString();
33                 String[] tokens = line.split( regex: "\t" );
34
35                 //Second column is the count
36                 int count = Integer.parseInt(tokens[1]);
37
38                 countTreeMap.put(count, new Text(value));
39
40                 if (countTreeMap.size() > 10)
41                     countTreeMap.remove(countTreeMap.lastKey());
42             }
43
44             for (Text t : countTreeMap.values())
45                 context.write(NullWritable.get(), t);
46         }
47     }
```

Top ‘N’ Filtering Pattern: Top 5 Accident Conducive Weather Conditions

```
DriverClass.java ×
14
15  ► public class DriverClass {
16
17  ► @ ► public static void main(String[] args) throws IOException, InterruptedException, ClassNotFoundException {
18
19      Configuration conf = new Configuration();
20      Job job = Job.getInstance(conf, "Top5WeatherConditions");
21      job.setJarByClass(DriverClass.class);
22
23      job.setMapperClass(MapperClass.class);
24      job.setReducerClass(ReducerClass.class);
25
26      job.setInputFormatClass(TextInputFormat.class);
27
28      job.setMapOutputKeyClass(NullWritable.class);
29      job.setMapOutputValueClass(Text.class);
30
31      job.setOutputKeyClass(NullWritable.class);
32      job.setOutputValueClass(Text.class);
33
34      job.setNumReduceTasks(1);
35
36      Path outDir = new Path(args[1]);
37      FileOutputFormat.setOutputPath(job, outDir);
38      Path inDir = new Path(args[0]);
39      FileInputFormat.addInputPath(job, inDir);
40
41      FileSystem fs = FileSystem.get(job.getConfiguration());
42      if (fs.exists(outDir)) {
43          fs.delete(outDir, b: true);
44      }
45
46      System.exit(job.waitForCompletion( verbose: true) ? 1 : 0);
47
48  }
49 }
```

```
MapperClass.java ×
1 package Question7.b;
2
3 import ...
4
5 public class MapperClass extends Mapper<Object, Text, NullWritable, Text> {
6
7     @Override
8     protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {
9         context.write(NullWritable.get(),value);
10    }
11 }
```

```
ReducerClass.java ×
1 package Question7.b;
2
3 import ...
10
11 public class ReducerClass extends Reducer<NullWritable, Text, NullWritable, Text> {
12
13     static class SortingClass implements Comparator<Integer> {
14
15         @Override
16         public int compare(Integer o1, Integer o2) { return o2.compareTo(o1); }
17     }
18
19
20
21     private TreeMap<Integer, Text> countTreeMap = new TreeMap<>(new SortingClass());
22
23
24     @Override
25     protected void reduce(NullWritable key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
26
27         for (Text value : values) {
28
29             String[] tokens = value.toString().split( regex: "\t");
30             int count = Integer.parseInt(tokens[1]);
31
32             countTreeMap.put(count, new Text(value));
33
34             if (countTreeMap.size() > 5)
35                 countTreeMap.remove(countTreeMap.lastKey());
36         }
37
38         for (Text t : countTreeMap.values())
39             context.write(NullWritable.get(), t);
40     }
41 }
```

Secondary Sorting: Number of Accidents Per County Per Year

```
DriverClass.java <pre>
19      // Create a new Job
20      Configuration config = new Configuration();
21      Job job = Job.getInstance(config, "SecondarySorting");
22      job.setJarByClass(DriverClass.class);
23
24      job.setGroupingComparatorClass(NaturalKeyGroupComparator.class); // Group comparator - this takes care of the grouping into the iterable t
25      job.setSortComparatorClass(CompositeKeySortComparator.class); // SortComparator -- does secondary sorting
26      job.setPartitionerClass(PartitionerClass.class); // Partitioner - this makes sure that the mapper output goes to relevant reducer
27
28      //Telling the program which is mapper/reducer class
29      job.setMapperClass(MapperClass.class);
30      job.setReducerClass(ReducerClass.class);
31
32      //Setting the InputFormat (What the data will be like)
33      job.setInputFormatClass(TextInputFormat.class);
34      job.setOutputFormatClass(TextOutputFormat.class);
35
36      job.setMapOutputKeyClass(WritableClass.class); // for custom class how the reducer has to group
37      job.setMapOutputValueClass(IntWritable.class);
38
39      job.setOutputKeyClass(Text.class);
40      job.setOutputValueClass(IntWritable.class);
41
42      //Setting the input and output paths
43      Path outDir = new Path(args[1]);
44      FileOutputFormat.setOutputPath(job, outDir);
45      Path inDir = new Path(args[0]);
46      FileInputFormat.addInputPath(job, inDir);
47
48      //Number of reducers set to 5 -- based on 5 partitions
49      job.setNumReduceTasks(5);
50
51      //Delete the output directory if it exists before execution
52      FileSystem fs = FileSystem.get(job.getConfiguration());
53      if (fs.exists(outDir)) {
54          fs.delete(outDir, true);
55      }
</pre>
```

```

④ MapperClass.java ×
7   import java.io.IOException;
8   import java.text.ParseException;
9   import java.text.SimpleDateFormat;
10
11  public class MapperClass extends Mapper<Object, Text, WritableClass, IntWritable> {
12
13      SimpleDateFormat sdfformat = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss");
14      SimpleDateFormat yearformat = new SimpleDateFormat( pattern: "yyyy");
15
16      @Override
17  protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {
18
19          String year = null;
20          String line = value.toString();
21          String[] tokens = line.split( regex: ",");
22
23          if (!tokens[0].equals("ID")) {
24
25              if (tokens[4].isEmpty() || tokens[16].isEmpty() || tokens[4].equals(" ") || tokens[16].equals(" "))
26                  return;
27
28
29          String county = tokens[16];
30
31          try {
32              year = yearformat.format(sdfformat.parse(tokens[4]));
33          } catch (ParseException e) {
34              e.printStackTrace();
35          }
36
37          WritableClass writableObj = new WritableClass(county, year);
38          context.write(writableObj, new IntWritable( value: 1));
39
40      }
41  }
④ ReducerClass.java ×
1  package Question8;
2
3  import org.apache.hadoop.io.IntWritable;
4  import org.apache.hadoop.io.Text;
5  import org.apache.hadoop.mapreduce.Reducer;
6
7  import java.io.IOException;
8
9  public class ReducerClass extends Reducer<WritableClass, IntWritable, Text, IntWritable> {
10
11      Text text = new Text();
12      IntWritable count = new IntWritable();
13
14      @Override
15  protected void reduce(WritableClass key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {
16
17      int sum = 0;
18
19      for(IntWritable v : values) {
20          sum += v.get();
21      }
22      text.set("County: "+key.getCounty()+" | Year: " + key.getYear() + " | Accidents: ");
23
24      count.set(sum);
25      context.write(text, count);
26
27  }
28

```

```

c WritableClass.java ×
5 import java.io.DataInput;
6 import java.io.DataOutput;
7 import java.io.IOException;
8
9 public class WritableClass implements WritableComparable<WritableClass> {
10
11     private String county;
12     private String year;
13
14     public WritableClass(String state, String year) {...}
15     public WritableClass() { super(); }
16     public String getCounty() { return county; }
17     public void setCounty(String county) { this.county = county; }
18     public String getYear() { return year; }
19     public void setYear(String year) { this.year = year; }
20
21     @Override
22     public int compareTo(WritableClass o) {...}
23
24     @Override
25     public void write(DataOutput dataOutput) throws IOException {
26         dataOutput.writeUTF(county);
27         dataOutput.writeUTF(year);
28     }
29
30     @Override
31     public void readFields(DataInput dataInput) throws IOException {
32         county = dataInput.readUTF();
33         year = dataInput.readUTF();
34     }
35
36     @Override
37     public String toString() {
38         return "county='" + county + '\'' +
39                 ", year='" + year + '\'';
40     }
41 }

```



```

c PartitionerClass.java ×
1 package Question8;
2
3 import org.apache.hadoop.io.IntWritable;
4 import org.apache.hadoop.mapreduce.Partitioner;
5
6 public class PartitionerClass extends Partitioner<WritableClass, IntWritable> {
7
8     @Override
9     public int getPartition(WritableClass compositeKeyWritable, IntWritable intWritable, int i) {
10         return Integer.parseInt(compositeKeyWritable.getYear()) % i;
11     }
12 }
13

```

```
c NaturalKeyGroupComparator.java x
1 package Question8;
2
3 import org.apache.hadoop.io.WritableComparable;
4 import org.apache.hadoop.io.WritableComparator;
5
6 public class NaturalKeyGroupComparator extends WritableComparator {
7
8     public NaturalKeyGroupComparator() { super(WritableClass.class, createInstances: true); }
11
12     //Which reducer to send
13     @Override
14     public int compare(WritableComparable a, WritableComparable b) {
15         WritableClass w1 = (WritableClass) a;
16         WritableClass w2 = (WritableClass) b;
17
18         return w1.getCounty().compareTo(w2.getCounty());
19
20     }
21 }
22

c CompositeKeySortComparator.java x
1 package Question8;
2
3 import org.apache.hadoop.io.WritableComparable;
4 import org.apache.hadoop.io.WritableComparator;
5
6 public class CompositeKeySortComparator extends WritableComparator {
7
8     protected CompositeKeySortComparator() { super(WritableClass.class, createInstances: true); }
11
12     @Override
13     public int compare(WritableComparable a, WritableComparable b) {
14
15         WritableClass w1 = (WritableClass) a;
16         WritableClass w2 = (WritableClass) b;
17
18         int result = w1.getYear().compareTo(w2.getYear());
19
20         if(result == 0) {
21             return w1.getCounty().compareTo(w2.getCounty());
22         }
23
24         return result;
25     }
26
27 }
```

Partitioning: Partitioning based on Wind Directions

DriverClass.java

```

14
15  public class DriverClass {
16
17  @ | public static void main(String[] args) throws IOException, ClassNotFoundException, InterruptedException {
18
19      Configuration configuration = new Configuration();
20      Job job = Job.getInstance(configuration, jobName: "PartitionWindDirection");
21      job.setJarByClass(DriverClass.class);
22
23      job.getConfiguration().set("join.type", "inner");
24      job.setReducerClass(ReducerClass.class);
25
26      //Partitions the data that is sent to the reducer
27      job.setPartitionerClass(PartitionerClass.class);
28
29      job.setOutputFormatClass(TextOutputFormat.class);
30
31      //Used MultipleOutputs as we are taking two data
32      MultipleInputs.addInputPath(job, new Path(args[0]), TextInputFormat.class, MapperClass.class);
33      MultipleInputs.addInputPath(job, new Path(args[1]), TextInputFormat.class, WindDirectionClass.class);
34
35      job.setOutputKeyClass(Text.class);
36      job.setOutputValueClass(Text.class);
37
38      Path outDir = new Path(args[2]);
39      FileOutputFormat.setOutputPath(job, outDir);
40
41      job.setNumReduceTasks(23);
42
43      FileSystem fs = FileSystem.get(job.getConfiguration());
44      if(fs.exists(outDir)){
45          fs.delete(outDir, b: true);
46      }
47
48      System.exit(job.waitForCompletion( verbose: true) ? 0 : 2);
49
50  }

```

MapperClass.java

```

3 import org.apache.hadoop.io.LongWritable;
4 import org.apache.hadoop.io.Text;
5 import org.apache.hadoop.mapreduce.Mapper;
6
7 import java.io.IOException;
8
9 public class MapperClass extends Mapper<LongWritable, Text, Text, Text> {
10
11     Text outKey = new Text();
12     Text outValue = new Text();
13     boolean first = true;
14
15     @Override
16     protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
17
18         String line = value.toString();
19         String[] tokens = line.split( regex: ",");
20
21         if(tokens[28].equals(" ") || tokens[28].isEmpty() || tokens[0].equals("ID")) {
22             return;
23         }
24         //Setting the wind direction from main dataset
25         String windDir = tokens[28];
26
27         //Converting to hadoop datatype
28         outKey.set(windDir);
29
30         //Setting the whole tuple as value and @ to identify its the tuple from first dataset
31         outValue.set('@' + value.toString());
32
33         context.write(outKey, outValue);
34
35     }

```

```
④ ReducerClass.java ×
10     private ArrayList<Text> list = new ArrayList<Text>();
11     private ArrayList<Text> windDirList = new ArrayList<Text>();
12
13     @Override
14     protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
15
16         list.clear();
17         windDirList.clear();
18
19         //Segregates the tuples from each dataset into arraylist based on what it was tagged with.
20         for (Text text : values) {
21             if (text.charAt(0) == '@') {
22                 list.add(new Text(text.toString().substring(1)));
23             } else if (text.charAt(0) == '#') {
24                 windDirList.add(new Text(text.toString().substring(1)));
25             }
26         }
27
28         innerJoinFunction(context);
29     }
30
31     @
32     public void innerJoinFunction(Context context) throws IOException, InterruptedException {
33
34         String join = context.getConfiguration().get("join.type");
35
36         //Checks the join type from the one set in Driver Class
37         if (join.equalsIgnoreCase("inner")) {
38             if (!list.isEmpty() && !windDirList.isEmpty()) {
39
40                 for (Text dir : windDirList) {
41
42                     for (Text mainTuple : list) {
43                         context.write(dir, mainTuple);
44                     }
45                 }
46             }
47         }
48     }
49 }
```

```
⑤ PartitionerClass.java ×
1 package Question9;
2
3 import org.apache.hadoop.io.Text;
4 import org.apache.hadoop.mapreduce.Partitioner;
5
6 public class PartitionerClass extends Partitioner<Text, Text> {
7
8     @Override
9     public int getPartition(Text text, Text text2, int i) {
10         return text.hashCode() % i;
11     }
12 }
13 }
```

```
⑥ WindDirectionClass.java ×
  package quickstart;

2
3   import org.apache.hadoop.io.LongWritable;
4   import org.apache.hadoop.io.Text;
5   import org.apache.hadoop.mapreduce.Mapper;
6
7   import java.io.IOException;
8
9   public class WindDirectionClass extends Mapper<LongWritable, Text, Text, Text> {
10
11     Text outKey = new Text();
12     Text outValue = new Text();
13
14     @Override
15     protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
16
17       String line = value.toString();
18       String[] tokens = line.split(",");
19
20       //Sets the columns from wind dataset
21       String description = tokens[1];
22       String windDir = tokens[0];
23
24       outKey.set(windDir);
25
26       //Setting the description as value and # to identify its from second dataset
27       outValue.set("#" + description);
28
29       context.write(outKey, outValue);
30
31     }
}
```

Inverted Index Pattern: Cities present in each County

```
DriverClass.java ×
17  public static void main(String[] args) throws IOException, InterruptedException, ClassNotFoundException {
18
19      //Create a job
20      Configuration conf = new Configuration();
21      Job job = Job.getInstance(conf, jobName: "");
22      job.setJarByClass(DriverClass.class);
23
24      //Telling the program which is mapper/reducer class
25      job.setMapperClass(MapperClass.class);
26      job.setReducerClass(ReducerClass.class);
27
28      job.setMapOutputKeyClass(Text.class);
29      job.setMapOutputValueClass(Text.class);
30
31      //Setting the InputFormat (What the data will be like)
32      job.setInputFormatClass(TextInputFormat.class);
33
34      job.setOutputKeyClass(Text.class);
35      job.setOutputValueClass(IntWritable.class);
36
37      //Set number of reducers to 1
38      job.setNumReduceTasks(1);
39
40      //Setting the input and output paths
41      Path outDir = new Path(args[1]);
42      FileOutputFormat.setOutputPath(job, outDir);
43      Path inDir = new Path(args[0]);
44      FileInputFormat.addInputPath(job, inDir);
45
46      //Delete the output directory if it exists before execution
47      FileSystem fs = FileSystem.get(job.getConfiguration());
48      if(fs.exists(outDir)){
49          fs.delete(outDir, b: true);
50      }
51      //wait job to finish and then exists
52      System.exit(job.waitForCompletion( verbose: true ) ? 0 : 1);
53 }
```

```

① MapperClass.java ×
1 package Question10;
2
3 import org.apache.hadoop.io.LongWritable;
4 import org.apache.hadoop.io.Text;
5 import org.apache.hadoop.mapreduce.Mapper;
6
7 import java.io.IOException;
8
9 public class MapperClass extends Mapper<LongWritable, Text, Text, Text> {
10
11     Text city = new Text();
12     Text county = new Text();
13
14     @Override
15     protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
16
17         String co = "";
18         String ci = "";
19         String line = value.toString();
20         String[] tokens = line.split( regex: "," );
21
22         if( !tokens[0].equals("ID") ) {
23             co = tokens[16];
24             ci = tokens[15];
25         }
26         city.set(ci);
27         county.set(co);
28         context.write(county, city);
29
30     }
31 }
32

② ReducerClass.java ×
1 package Question10;
2
3 import org.apache.hadoop.io.Text;
4 import org.apache.hadoop.mapreduce.Reducer;
5
6 import java.io.IOException;
7 import java.util.HashSet;
8
9 public class ReducerClass extends Reducer<Text, Text, Text, Text> {
10
11     //Random storage of input
12     HashSet<String> hs = new HashSet<>();
13
14     @Override
15     protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
16
17         hs.clear();
18         StringBuffer sb = new StringBuffer("");
19
20         //Put all cities in hashset
21         for(Text v: values){
22             hs.add(v.toString());
23         }
24         //Iterate hashset
25         for(String v: hs) {
26             sb.append(v);
27             sb.append(" , ");
28         }
29
30         context.write(key, new Text(sb.toString()));
31     }
32 }
33

```

Binning: Binning based on Accidents on each side of the road (Left/Right)

```

18 public static void main(String[] args) throws IOException, InterruptedException, ClassNotFoundException {
19
20     //Create a job
21     Configuration conf = new Configuration();
22     Job job = Job.getInstance(conf, "BinningBasedOnSides");
23
24     //Binning splits data up in the map phase instead of in the partitioner
25     //Hence not needing a reducer
26     job.setJarByClass(Driver.class);
27     job.setMapperClass(MapperClass.class);
28
29     job.setOutputKeyClass(Text.class);
30     job.setOutputValueClass(NullWritable.class);
31
32     //MultipleOutputs class sets up the job's output to write multiple distinct files
33     MultipleOutputs.addNamedOutput(job, namedOutput: "RoadSideBins", TextOutputFormat.class, Text.class, IntWritable.class);
34     MultipleOutputs.setCountersEnabled(job, enabled: true);
35
36     job.setNumReduceTasks(0);
37
38     //Setting the input and output paths
39     Path outDir = new Path(args[1]);
40     FileOutputFormat.setOutputPath(job, outDir);
41     Path inDir = new Path(args[0]);
42     FileInputFormat.addInputPath(job, inDir);
43
44     //Delete the output directory if it exists before execution
45     FileSystem fs = FileSystem.get(job.getConfiguration());
46     if(fs.exists(outDir)){
47         fs.delete(outDir, b: true);
48     }
49
50     System.exit(job.waitForCompletion( verbose: true ) ? 0 : 1);
51 }
52 }

C MapperClass.java x
9
10    public class MapperClass extends Mapper<Object, Text, Text, NullWritable> {
11
12        private MultipleOutputs<Text, NullWritable> multipleOutputs = null;
13
14        @Override
15        protected void setup(Context context) throws IOException, InterruptedException {
16            multipleOutputs = new MultipleOutputs(context);
17        }
18
19        @Override
20        public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
21
22            String line = value.toString();
23            String[] tokens = line.split( regex: "," );
24
25            if(!tokens[0].equals("ID")) {
26                return;
27            }
28            if(!tokens[14].equalsIgnoreCase( anotherString: "" )) {
29                String roadside = tokens[14];
30                //bin name, key, value, out put bin name
31                if (roadSide.equalsIgnoreCase( anotherString: "L"))
32                    multipleOutputs.write( namedOutput: "RoadSideBins", value, NullWritable.get(), baseOutputPath: "LeftSide");
33                if (roadSide.equalsIgnoreCase( anotherString: "R"))
34                    multipleOutputs.write( namedOutput: "RoadSideBins", value, NullWritable.get(), baseOutputPath: "RightSide");
35            }
36        }
37
38        //After mapper it runs and closes the program
39        @Override
40        protected void cleanup(Context context) throws IOException, InterruptedException {
41            multipleOutputs.close();
42        }
43    }
44

```

References

<https://timepasstechies.com/category/programming/data-analytics/mapreduce/>

<https://arxiv.org/abs/1906.05409>

<https://arxiv.org/abs/1909.09638>

https://smoosavi.org/datasets/us_accidents

<https://github.com/aamend/hadoop-mapreduce/blob/master/design-pattern/src/main/java/com/aamend/hadoop/mapreduce/designpattern/join/ReduceSideJoin.java>