UNIVERSITY PARTNER

UNIVERSITY OF WOLVERHAMPTON

HERALD COLLEGE KATHMANDU

6CS030 – BIG DATA

GROUP ASSESSMENT

Student ID : 2038578

Student Name : Sandesh Thapa Magar

Student ID : 2038577

Student Name : David Tamang

Module Leader : Jnaneshwar Bohara

Submitted On : 4/25/2021

Acknowledgement

It might not have been feasible to finish this endeavor report without the participation, help and reasonable cooperation of the Individuals and a few people whose names can't all be incorporated. They value their efforts genuinely and thankfully. Mr. Jnaneshwar Bokhara, Chiran Jvi and Dinesh Saud are very grateful to us for their advice and on-going oversight as well as the knowledge they provide in this study and support for the completion of the Project Report.

We are particularly grateful to our colleague for their intimate readiness and abilities to go ahead and produce this paper. We are therefore obliged to our university and faculty members, who supports us directly or indirectly for our study.

Table of Contents

Work Division

Task Division	Team Members
Introduction to Big Data	Sandesh Thapa Magar
Introduction to Dataset	David Tamang
Import Cleaning of Data	Sandesh Thapa Magar & David Tamang
Analysis and Visualizations	Sandesh Thapa Magar & David Tamang
Comparison	Sandesh Thapa Magar
Conclusion	David Tamang

1. Big Data Introduction

Gigantic data is an emerging state where immense volume of data is conveyed ordinarily as both coordinated, semi-coordinated and unstructured, which will make an issue for a standard methodology and informational index to manage it. An approach to manage trained decisions subject to correct methodology depicts any data arrangement, and is sufficiently wide to use critical level programming ability and methodologies to change the data into an affiliation's asset. The three DBMS open-source systems and the Oracle, MongoDB and Apache Hadoop programming tools have so far been used to solve problems in a wide range of datasets (Juneja & Das, 2019).

Such colossal information comes from a few distinctive sources with intricacies known as 5Vs i.e.

I. Volume: This is normal for the enormous enlightening lists made at high repeat rates.
II. Variety This applies to different types of documents, i.e., ordered, semi-structured, unstructured and everything.
III. Velocity: This investigates how effectively and consistently information can be made from a solicitation.
IV. Veracity: This tends to the exactness, precision, and if credibility of the information.
V. Value: This includes the importance of data from various raw data that are used. It is not feasible merely because of the abundance of information to extract utility from it.

2. Dataset Introduction

We have three different datasets for each database which are somehow similar to death. We have taken “suicide rates overview 1985-2016” dataset from other datasets (Human Development Reports, 2020) (World Bank, 2018) and (WHO, 2018) linked by time and place to identify a corresponding signal to raise suicide rates in the various cohorts globally. Another is “Novel corona virus 2019” and last we have dataset for leading cause of death in USA.

2.1 Explanation of Preference

We have chosen the death mortality rate in those three datasets in different way like from the suicide, corona pandemic and leading cause of death in USA. We will do analysis and visualization of those datasets, and figure out the rate of suicide and its key factor for it, for corona

virus we will try to visualize and do analysis and at last for Cardiovascular diseases we will try to predict heart failure. Suicide datasets can be found [1][1][1], corona dataset on Johns Hopkins GitHub and leading cause from data.gov.

2.2 CSV Dataset

We have suicide csv dataset for the oracle and Hadoop. suicide dataset consists of more than 12 columns, 1000 of records and also leading death dataset has 6 columns and 1000 of record, which will be reduced while cleaning the dataset as required for the project.

2.3 JSON Dataset

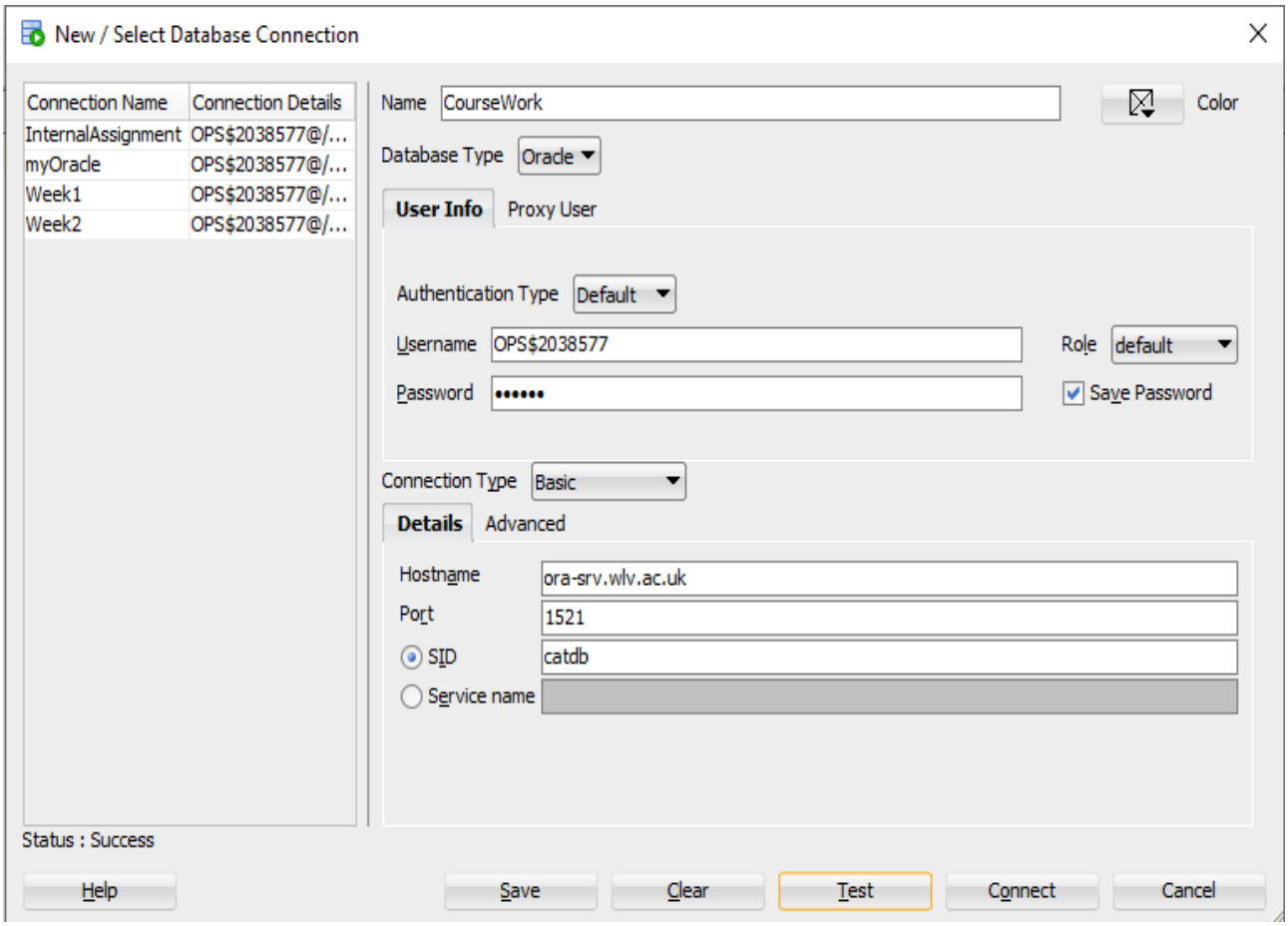
For mongo we have json format dataset of corona virus. It has more then 10000 of record of corona virus of every country that corona virus had affected. We will be analyzing and visualize those datasets according to the requirement. We may reduce or remove some records at time of cleaning the dataset according to its requirements.

Import/Cleaning of Data

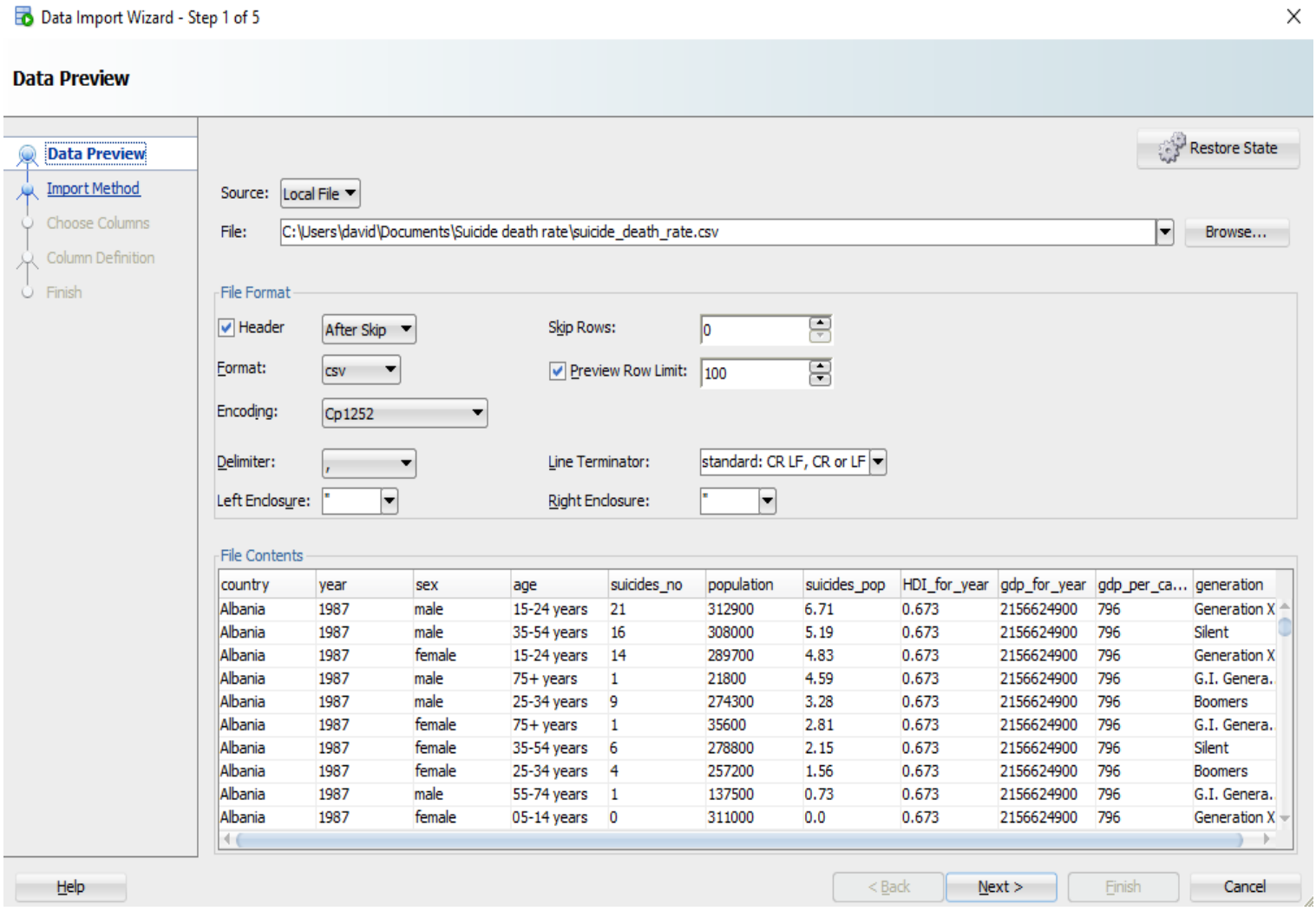
Note: Cleaning of data is in Appendix for all three datasets

Importing CSV data:

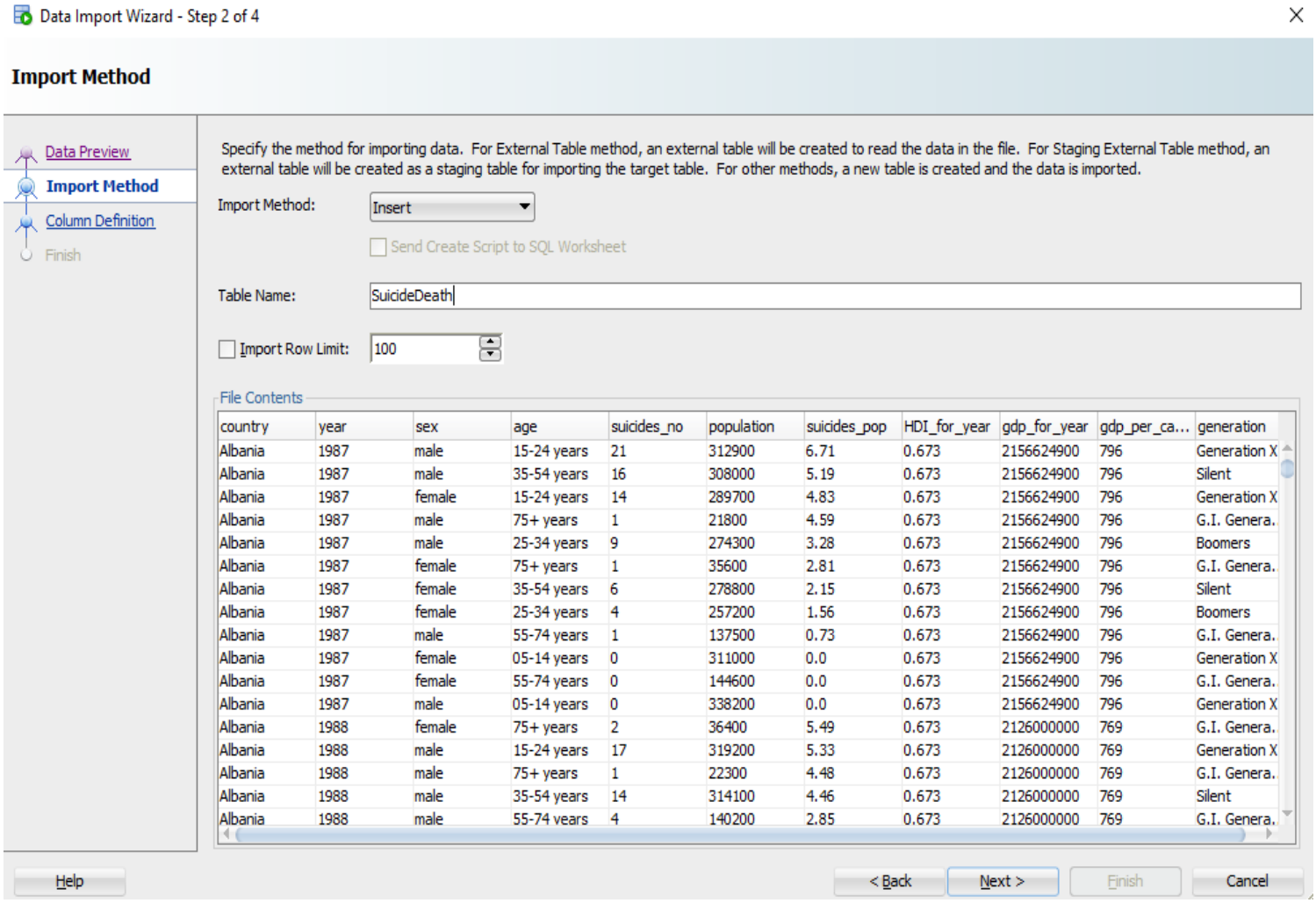
Creating new connection in oracle:



Step 1: Browsing location of CSV file.

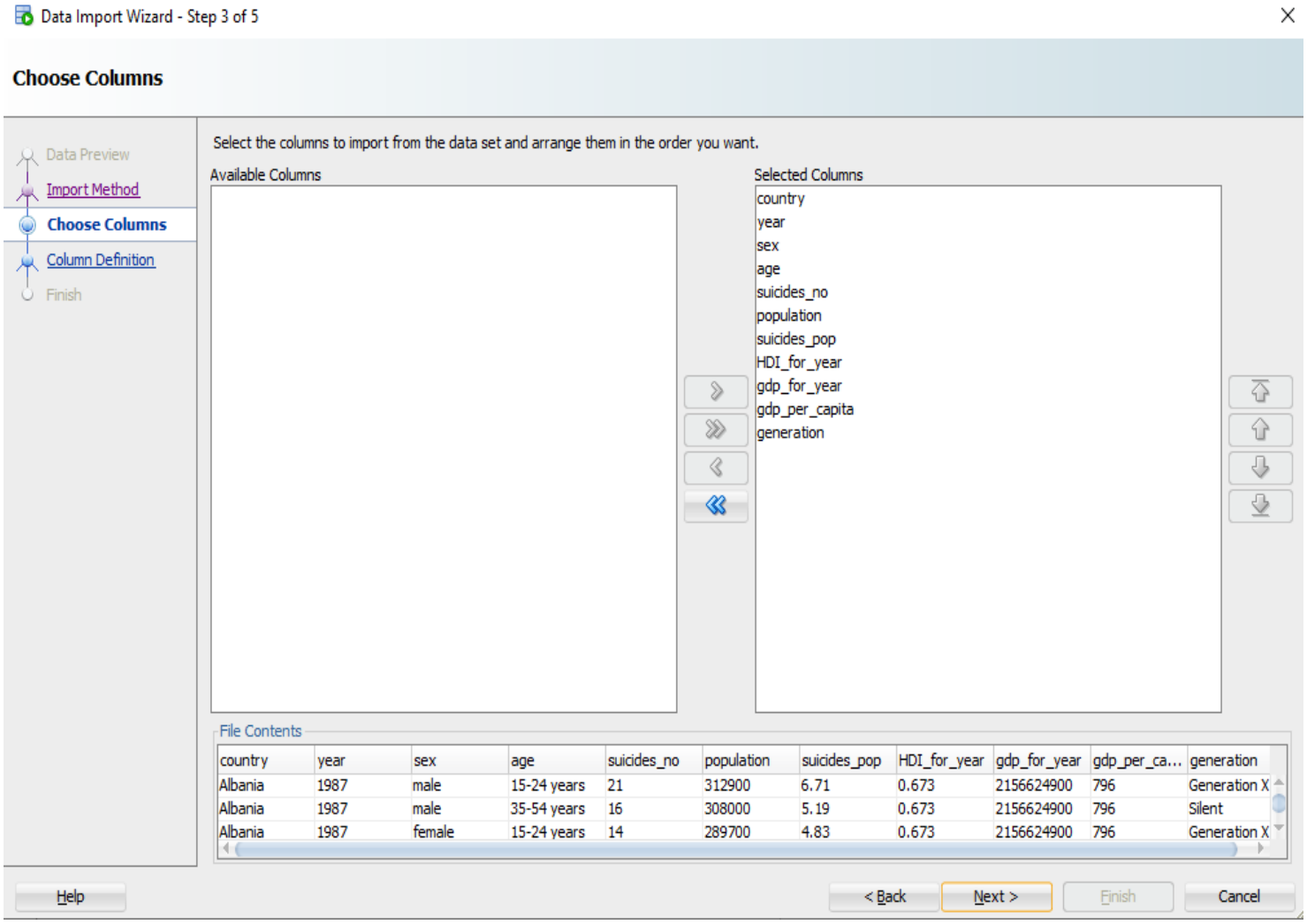


Step 2: Creating new table called SuicideDeath which will store all the csv file data.

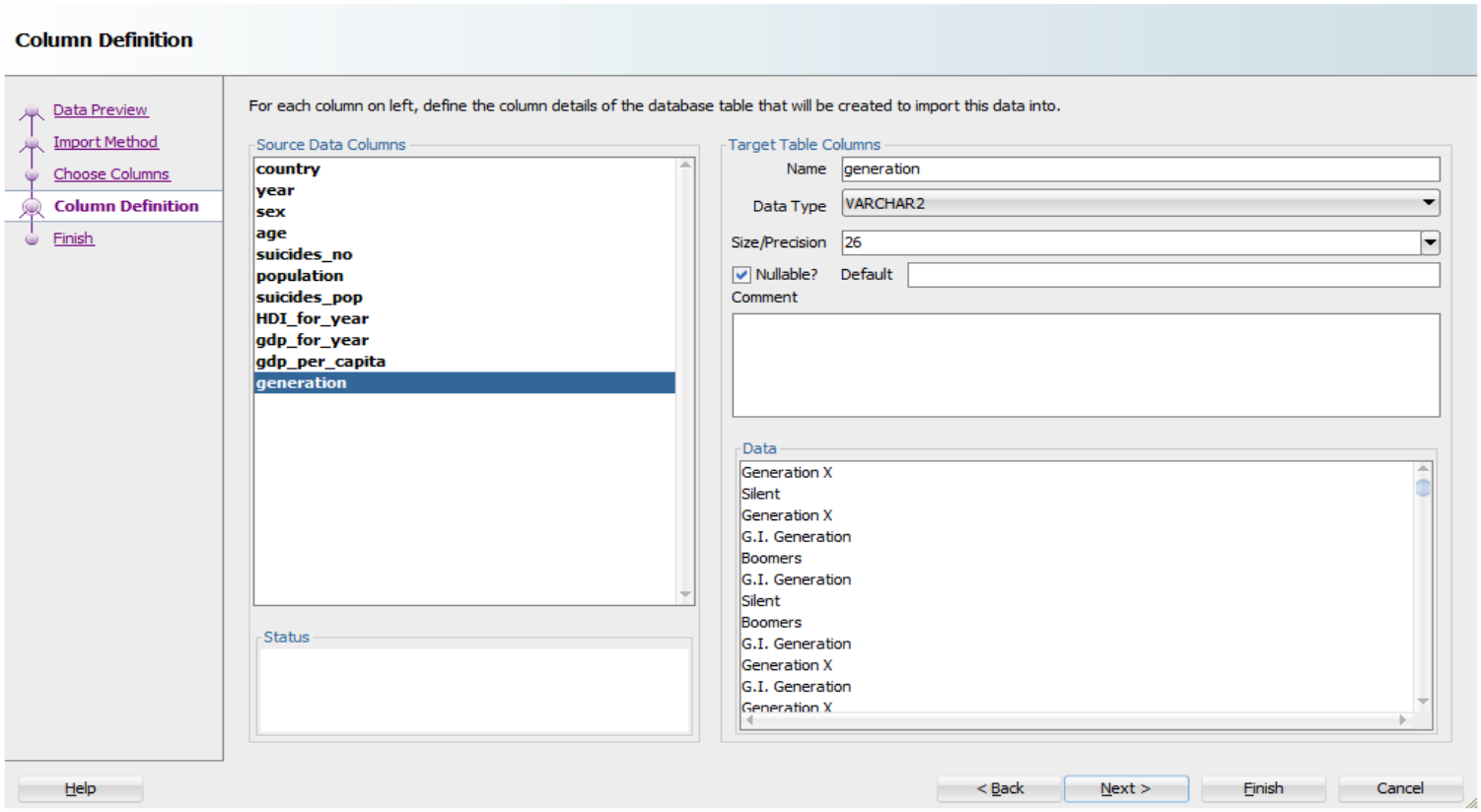


country	year	sex	age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_ca...	generation
Albania	1987	male	15-24 years	21	312900	6.71	0.673	2156624900	796	Generation X
Albania	1987	male	35-54 years	16	308000	5.19	0.673	2156624900	796	Silent
Albania	1987	female	15-24 years	14	289700	4.83	0.673	2156624900	796	Generation X
Albania	1987	male	75+ years	1	21800	4.59	0.673	2156624900	796	G.I. Genera.
Albania	1987	male	25-34 years	9	274300	3.28	0.673	2156624900	796	Boomers
Albania	1987	female	75+ years	1	35600	2.81	0.673	2156624900	796	G.I. Genera.
Albania	1987	female	35-54 years	6	278800	2.15	0.673	2156624900	796	Silent
Albania	1987	female	25-34 years	4	257200	1.56	0.673	2156624900	796	Boomers
Albania	1987	male	55-74 years	1	137500	0.73	0.673	2156624900	796	G.I. Genera.
Albania	1987	female	05-14 years	0	311000	0.0	0.673	2156624900	796	Generation X
Albania	1987	female	55-74 years	0	144600	0.0	0.673	2156624900	796	G.I. Genera.
Albania	1987	male	05-14 years	0	338200	0.0	0.673	2156624900	796	Generation X
Albania	1988	female	75+ years	2	36400	5.49	0.673	2126000000	769	G.I. Genera.
Albania	1988	male	15-24 years	17	319200	5.33	0.673	2126000000	769	Generation X
Albania	1988	male	75+ years	1	22300	4.48	0.673	2126000000	769	G.I. Genera.
Albania	1988	male	35-54 years	14	314100	4.46	0.673	2126000000	769	Silent
Albania	1988	male	55-74 years	4	140200	2.85	0.673	2126000000	769	G.I. Genera.

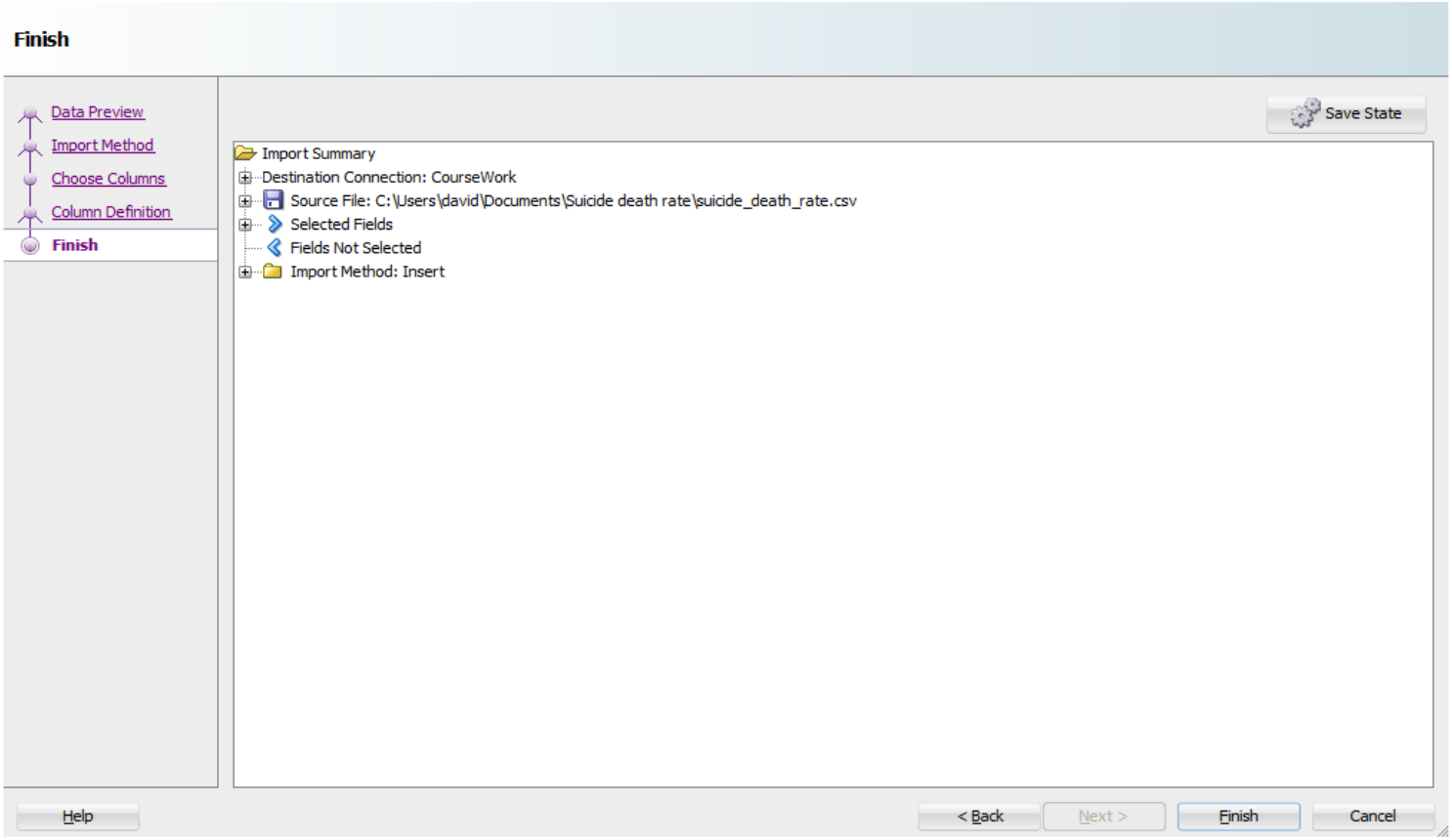
Step 3: Selecting column which is necessary



Step 4: Selecting appropriate datatype for each column.



Step 5: Finishing the import:

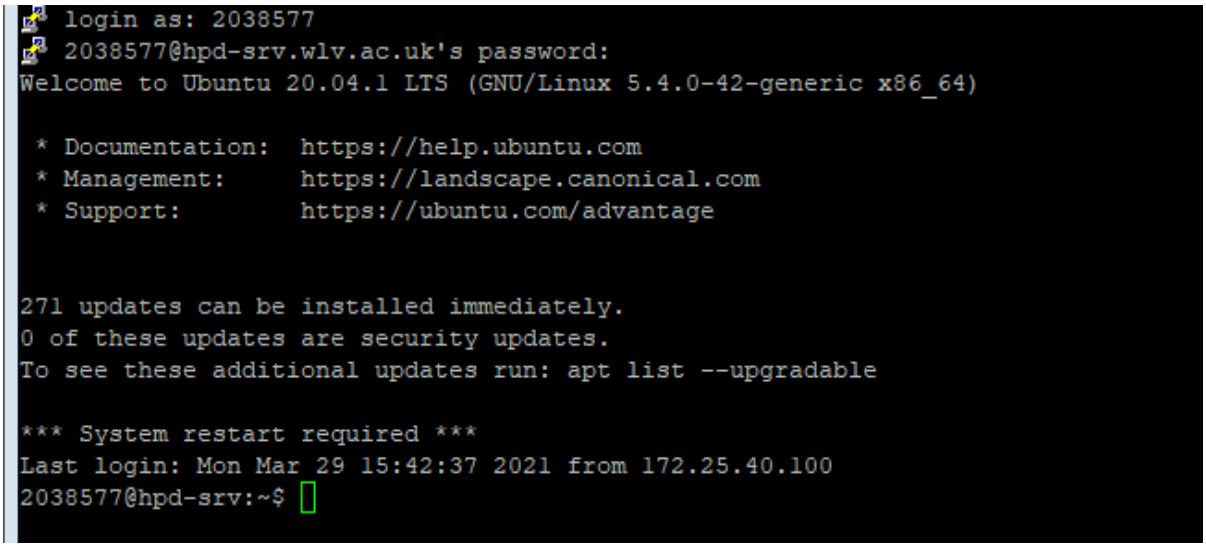


Importing JSON

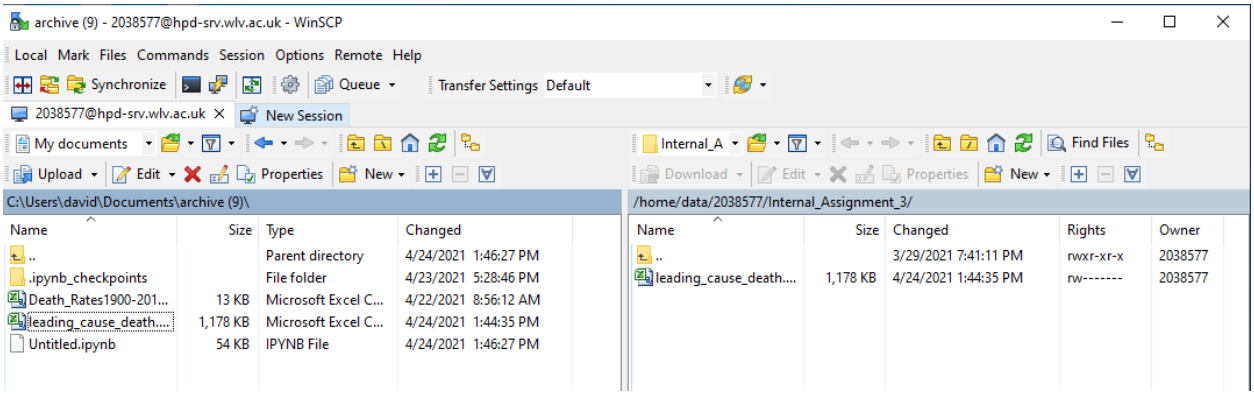
```
C:\Users\Sandesh Thapa>mongoimport --db CourseWork --collection corona --jsonArray D:\Corona.json
2021-04-22T15:09:46.529+0545    connected to: mongodb://localhost/
2021-04-22T15:09:49.530+0545    [###########.............] CourseWork.corona    25.2MB/53.1MB (47.4%)
2021-04-22T15:09:52.530+0545    [######################.] CourseWork.corona    52.6MB/53.1MB (99.1%)
2021-04-22T15:09:52.601+0545    [########################] CourseWork.corona    53.1MB/53.1MB (100.0%)
2021-04-22T15:09:52.601+0545    236017 document(s) imported successfully. 0 document(s) failed to import.
```

Importing CSV for Hadoop

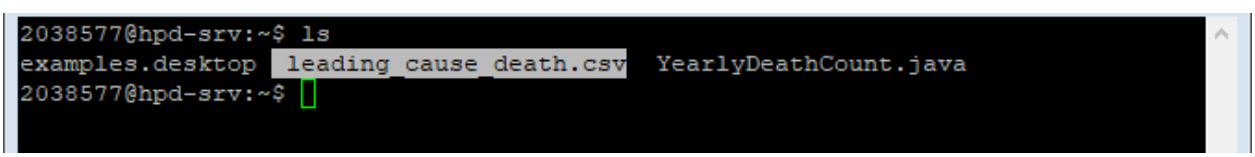
Step 1: Login Using username and Password in putty



Step 2: Upload CSV file in winscp



Step 3: Showing uploading files in terminal



Appropriateness of dataset in each database

The findings showed that oracle performs best in csv format than json when the data is zero for rows. Csv is avoided in standard SQL paging routing to achieve quicker results and produce efficiency. For mongo it was seen that json format of data outperformed the csv data because it is complicated for reading and verbose. In Hadoop we need dataset where we can split and encode any data so csv is used because the text of characters are splitable.

Analysis of the Data and visualizations

Note: Analysis is done in appendix due to word limit

Data Visualization

Oracle

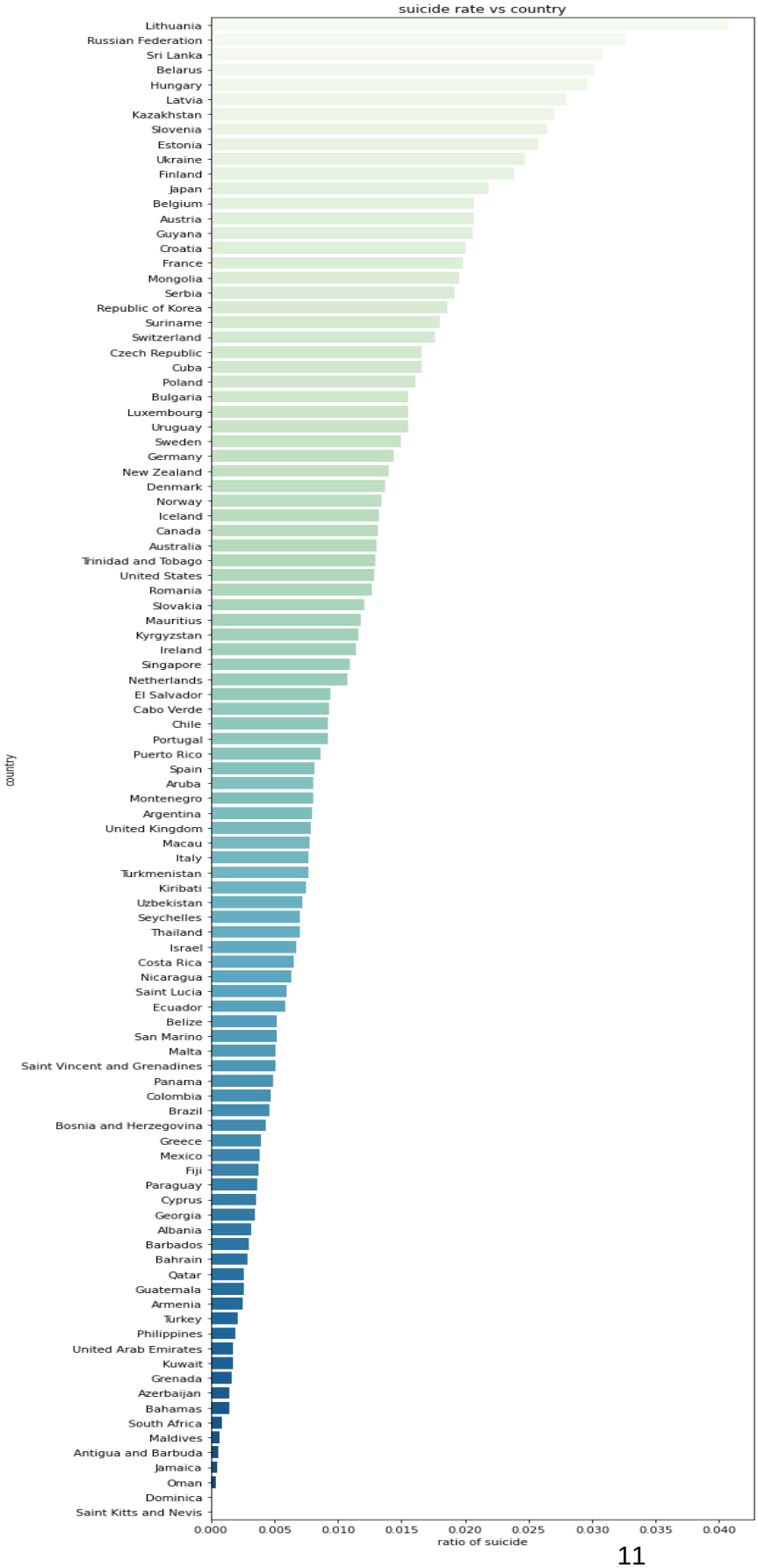


Figure 1 Suicide rate of every country

The above visualization shows the suicide rate on average of each country from lowest rate Saint Kitts and Nevis to highest Lithuania.

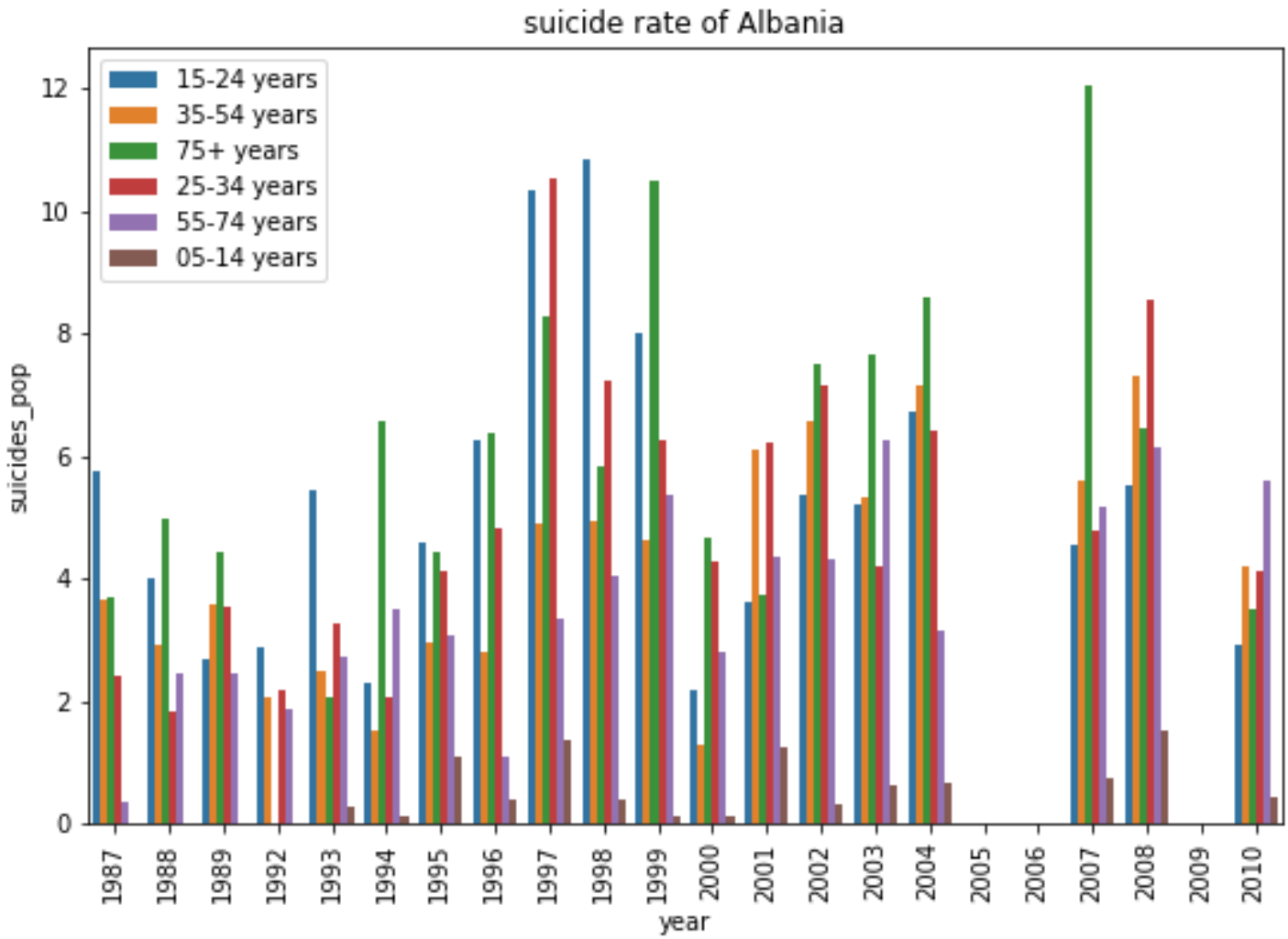
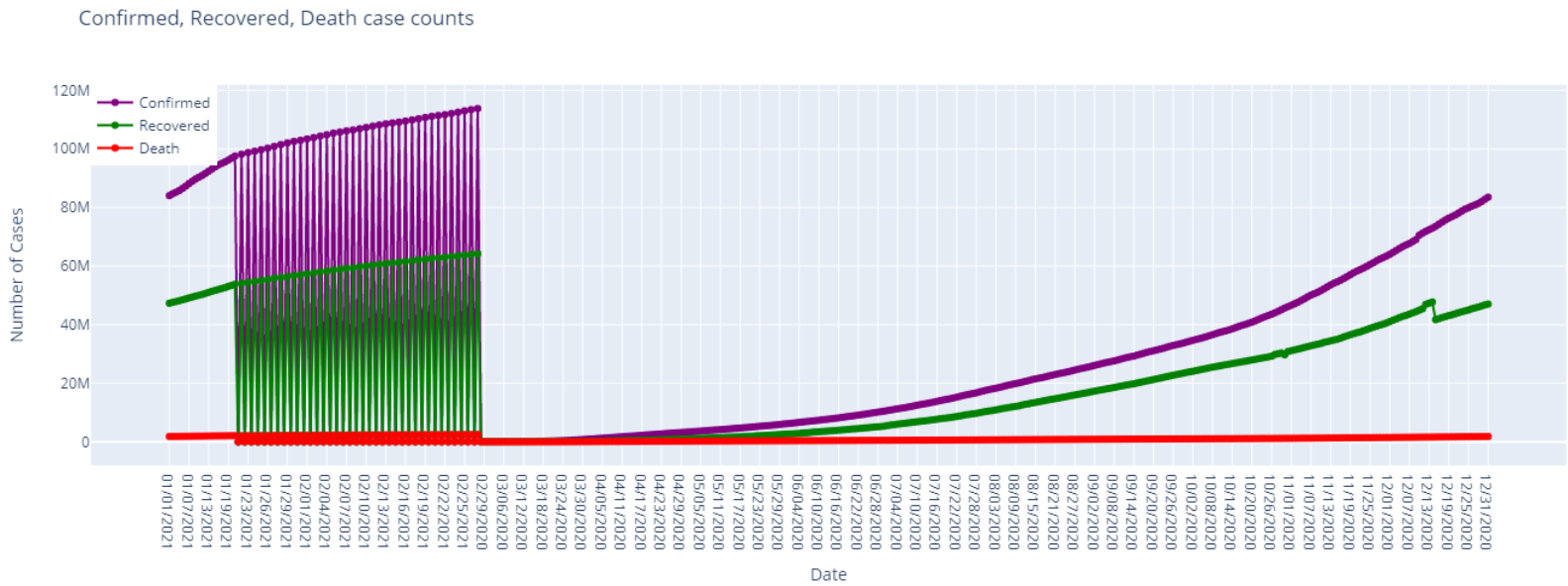


Figure 2 Suicide Rate of Albania

From the above visualization figure, we can see the suicide rate of Albania between different age group. If we see suicide rate between 1987 to 1996 age group of 15-24 have highest death rate and followed by 35-54 and 75 age group in second place. After 1966 we can see the peak in suicide rate from 1997 to 1999. In those time period 15-24, 25-34 have highest suicide rate.

The prevalence of suicide is linked to many factors such as national per capita. We found that the suicide rate is strongly associated with GDP in the world in certain nations, showing a certain degree of decrease throughout the suicide rate in the degree of national income.

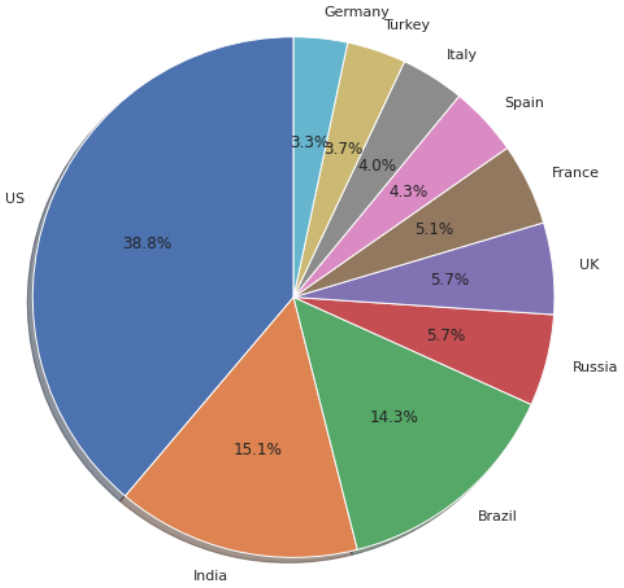
Mongo



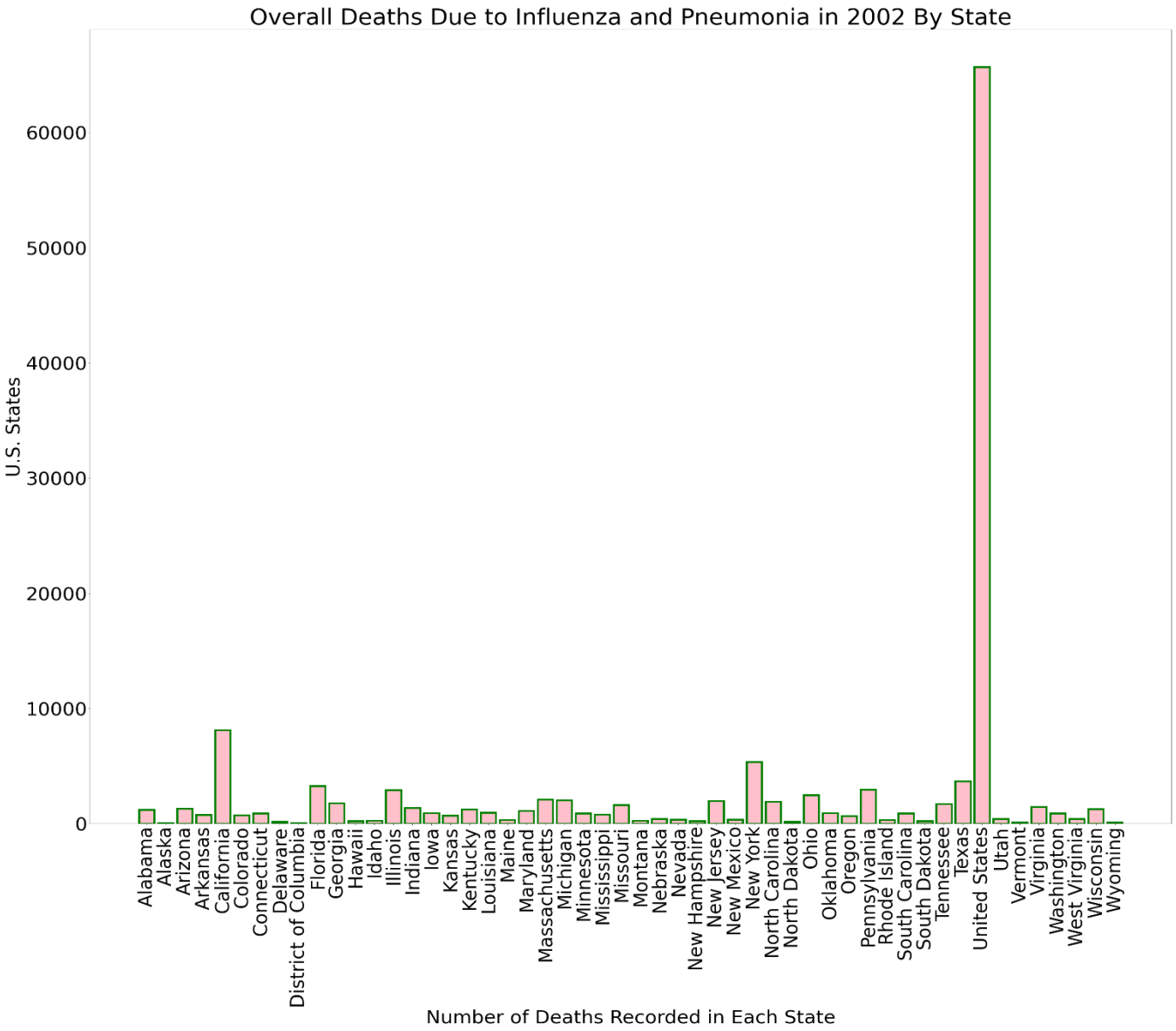
Note: Figure date Starts from 03/06/2020

Above figure shows that at when corona was first confimred there was more 84 thousand confimred case with 36 thousand recovered and 2872 deaths per day worldwide. Those confimred case keep increasing by the end of 2020 and reach upto 83.52186M confirmed case and death of more then 1.8125578M of people and it is still increasing.

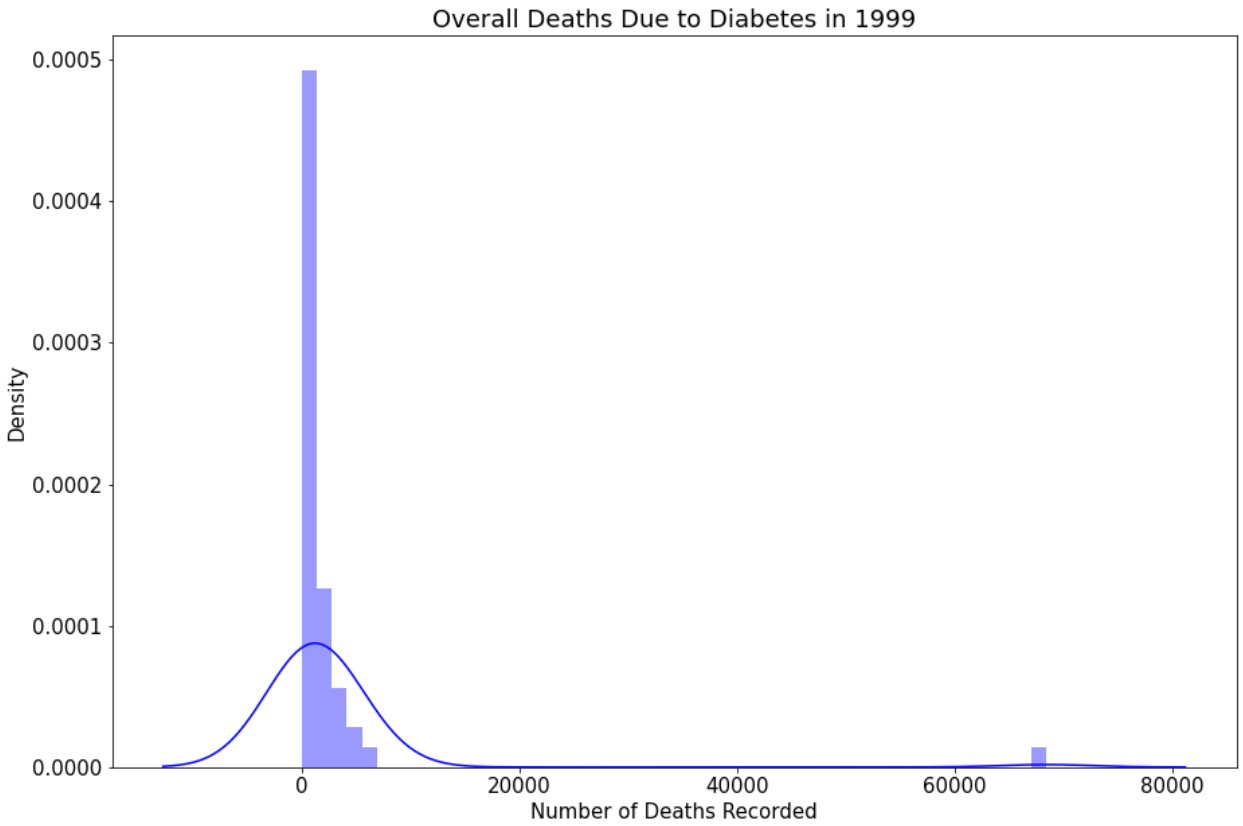
From this pie chart we can see that US was badly affected by the corona virus. Out of 100% corona virus US had 38.8% of virus following closely to each other by India 15.1% and 14.3%. Those three counties almost carried the more then half of the world corona virus.



Hadoop



In this bar chart, the number of deaths reported in 2002 by nation, and in the US as a whole, as a result of flu and pneumonia complications is seen.



From the above histogram, it shows that number of death records in 1999 from diabetes-realted complications

Comparison

Advantage of using three different database management system

Oracle

- Versatility:
  Oracle can operate on any kind of operating system, giving users a lot of versatility.

- Functionality:
  It coordinates undeniable level innovations with cutting edge undertaking arrangements that can be utilized to oversee enormous volumes of information in practically all venture applications.

- Reliability:
  Oracle is a trustworthy data set framework and can likewise be a data framework that performs well when extraction errands are allotted to it.

- Data Recovery:
  Oracle has a feature called Flashback that helps you to retrieve data that has been removed or destroyed, which speeds up data recovery and simplifies management.

- Performance:
  Oracle provides techniques such as using several servers to figure out constant information with actual Application cluster function that can increase computing speed (Malavika, 2019).

Mongo

- Versatility:
  It provides tremendous flexibility in the collection and access to data types, since it is a schema less database.

- Accessibility:
  MongoDB has many community drivers to serve in lesser-known programming languages all main programming languages.

- Speed:
  Data access and query is very speedy since the RAM is used for storing data.
- Cost
  It's highly inexpensive because it eliminates data storage and hardware costs.

- Scalability
  It is really convenient to scale the database. It offers the chance to sharpen horizontally (DataFlair, 2019).

Hadoop

- Flexibility:
  In better places, Hadoop can be utilized for methods like information stockpiling, extortion recognizable proof, and so on.

- Speed:
  The methods used are mostly located on the same computer where the data processes are done very quickly.

- Scalability:
  The computing platforms are very versatile, and they can save and provide massive volumes of data over different parallel servers.

- Safer
  In case of a malfunction, information is copied to different hubs with the end goal that another duplicate is prepared for the clients.

- Cost effective
  It is rendered as an infrastructure for scale-outs to hold all the big data of businesses and can be later used for computation and storage (Stevepaul, 2015).

Disadvantage of using three different database management system.

Oracle

- The machine identity of Oracle is limited to 8.
- Oracle has a special network setup to function correctly.
- It lacks looping functions like and which loops so that remedial processing is not carried out.
- Contrasted with other SQLs, the expense of oracle is extremely high.
- Oracle does not provide integrated functions for data cleaning. For the cleaning of the data resources of third parties are therefore used (Singh, 2020).

MongoDB

- Joins not Supported:
  Like a relationship data set MongoDB doesn't uphold join. In any case, you can utilize the associations highlight by physically coding it. Be that as it may, execution can back off and results can change.

- High Memory Usage:
  For each pair of values, MongoDB stores key names. There is also no data redundancy due to no link capability. This leads to inefficient memory consumption.

- Limited Data Size:
  It does not support document size more than 16mb.

- Limited Nesting:
  There is a limitation that we cannot perform nesting for more than 100 levels (DataFlair, 2019).

Hadoop

- Security Concerns:
  It can be a challenge to only manage complicated systems like Hadoop. In the Hadoop protection model, a basic example can be used, which is disabled due to the complexity. If the manager of the network does not know how to do so, the data might be at enormous risk and it lacks the encryption.

- Vulnerable by Nature:
  With respect to defense, Hadoop's own make-up puts it at risk. The code is almost exclusively written in Java, one of the most commonly used and divisive programs. Java has been extensively abused and has also been embroiled in several abuses of confidentiality.

- Not fit for small Data
  Albeit huge information isn't delivered carefully for enormous associations, not all large information frameworks are proper for little information

necessities. Unfortunately, it happens that Hadoop is one of them. The Hadoop Disseminated Document Framework does not have the capacity to viably oblige the change of little records because of its high-volume nature. Thus, restricted information volumes are not suggested for organizations.

- Potential Stability Issues
  Hadoop has a good extent of strength issues like all open-Source applications. Organizations must ensure that they run the new stable update or run it under a third-Party Provider who is prepared to deal with those issues in order to prevent these problems.

- General limitations
  The article presents as potential alternatives Apache Flume, MillWheel and Google's cloud dataflow. Both of the platforms have in common an opportunity to increase data storage, aggregate and integration quality and reliability. The key argument in the essay is that businesses will skip great advantages when using Hadoop alone [17].

Conclusion and Recommendations

So, in a nutshell we had gone through the practical work on all three databases, it was challenging and interesting. When we operate on these databases, we know that they process vast volumes of data, which makes these databases an irreplaceable part of the big data world. We may use the sql command to view the data in oracles. Because MongoDB is a NoSQL, the document uses json and schema, but supports all csv and json formats. It is also supported for Hadoop.

Different queries were carried out at the time of doing coursework on these databases and we can see clear visualization of data in the form of diagram. Therefore, to maintain accuracy we should feed the data of complete error free by cleaning it. These will lead to much faster and more accurate data needed. Also, while selecting the dataset the user should know the dataset properly, as a result user can take maximum advantage of dataset while analyzing, running quires and visualizing the data.

Appendix

Oracle

Oracle database was made by oracle corporation in 1997, as a connected data set administration framework. It is a multi-model social information base administration framework, especially for the calculation of organization network and information stockpiling. It is one of the principal alternatives for practical arrangements and information stockpiling for organizations. It upholds SQL for communicating with the data set as a query language (Educba, 2020).

MongoDB

MongoDB is a NoSQL information base for the report arranged capacity of high-volume documents. MongoDB utilizes records and records as opposed to utilizing tables and lines as in standard social data sets. Reports are comprised of essential worth combines that are a crucial MongoDB information unit. Assortments incorporate records and capabilities that are equivalent to connection information base tables. MongoDB is a mid-2000s site that has materialized (Guru99, 2021).

Hadoop

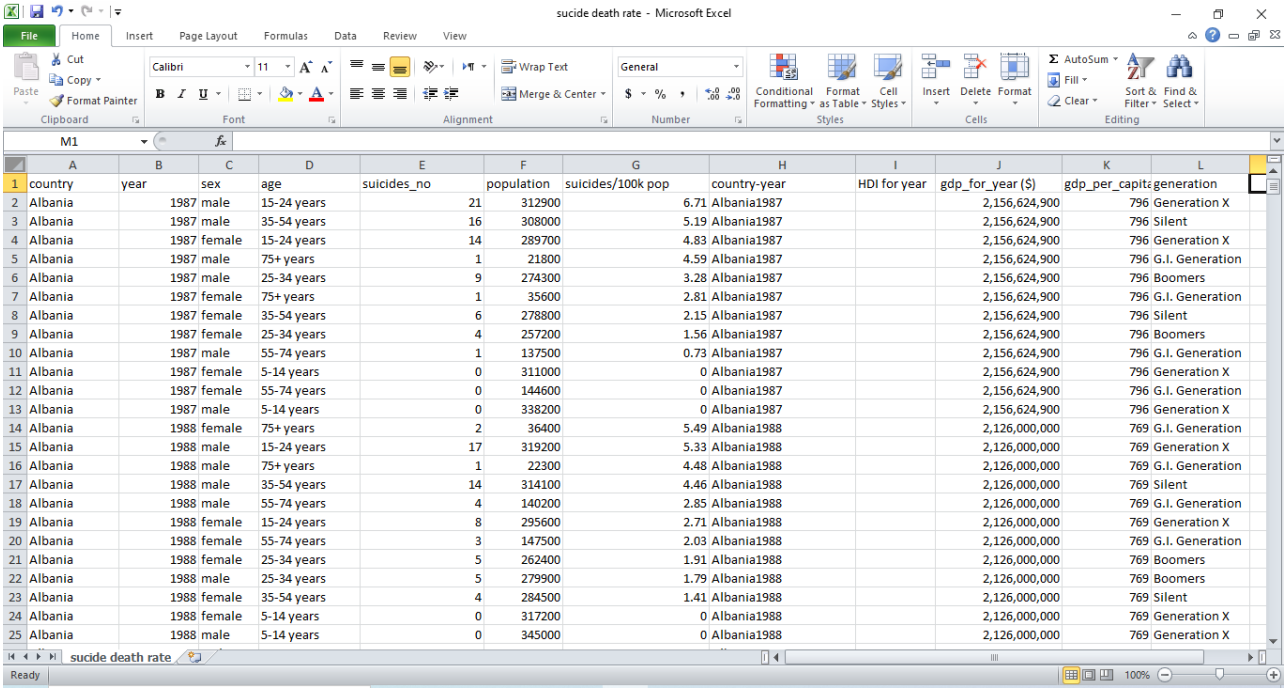
Hadoop is a software open-source architecture for data storage and operation on commodity clusters. It has vast data capacity, immense computing power and the capacity to perform almost unlimited simultaneous tasks or tasks.

Cleaning of data

Oracle

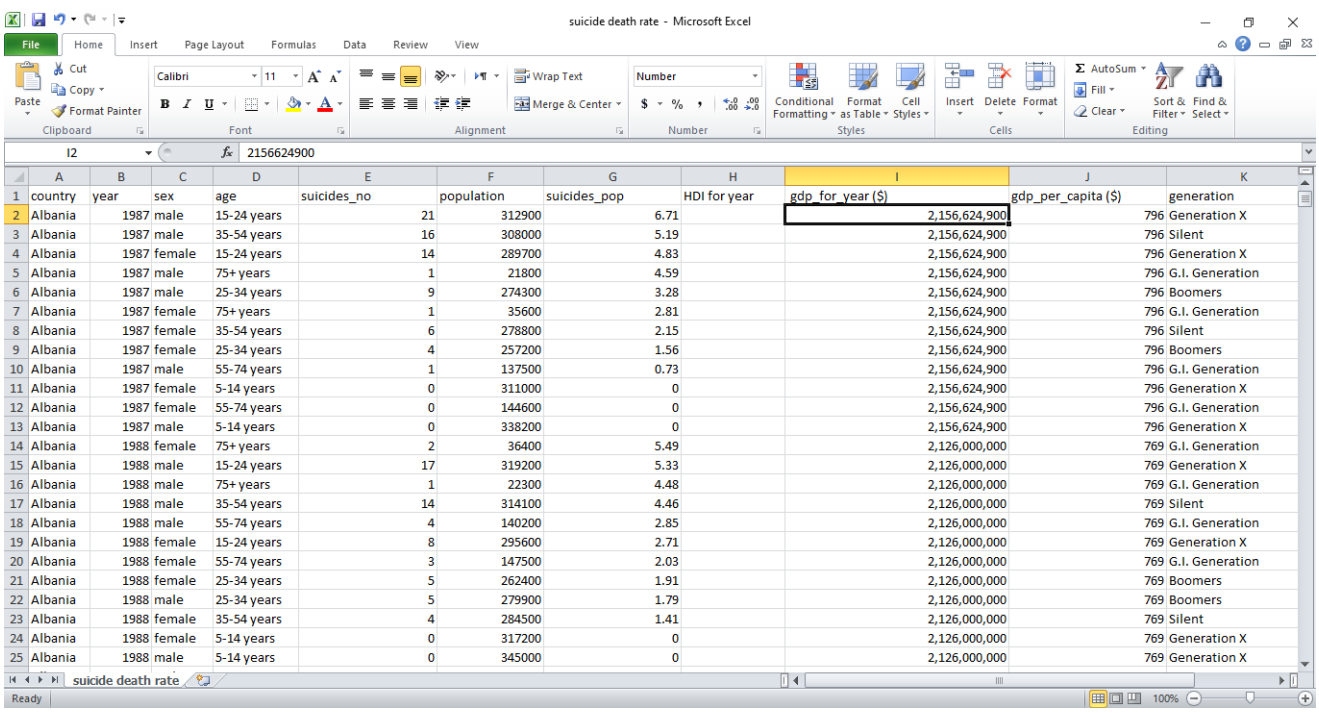
Step 1: Dropping Unnecessary columns

Before



country	year	sex	age	suicides_no	population	suicides/100k pop	country-year	HDI for year	gdp_for_year ($)	gdp_per_capita	generation
Albania	1987	male	15-24 years	21	312900	6.71	Albania1987		2,156,624,900	796	Generation X
Albania	1987	male	35-54 years	16	308000	5.19	Albania1987		2,156,624,900	796	Silent
Albania	1987	female	15-24 years	14	289700	4.83	Albania1987		2,156,624,900	796	Generation X
Albania	1987	male	75+ years	1	21800	4.59	Albania1987		2,156,624,900	796	G.I. Generation
Albania	1987	male	25-34 years	9	274300	3.28	Albania1987		2,156,624,900	796	Boomers
Albania	1987	female	75+ years	1	35600	2.81	Albania1987		2,156,624,900	796	G.I. Generation
Albania	1987	female	35-54 years	6	278800	2.15	Albania1987		2,156,624,900	796	Silent
Albania	1987	female	25-34 years	4	257200	1.56	Albania1987		2,156,624,900	796	Boomers
Albania	1987	male	55-74 years	1	137500	0.73	Albania1987		2,156,624,900	796	G.I. Generation
Albania	1987	female	5-14 years	0	311000	0	Albania1987		2,156,624,900	796	Generation X
Albania	1987	female	55-74 years	0	144600	0	Albania1987		2,156,624,900	796	G.I. Generation
Albania	1987	male	5-14 years	0	338200	0	Albania1987		2,156,624,900	796	Generation X
Albania	1988	female	75+ years	2	36400	5.49	Albania1988		2,126,000,000	769	G.I. Generation
Albania	1988	male	15-24 years	17	319200	5.33	Albania1988		2,126,000,000	769	Generation X
Albania	1988	male	75+ years	1	22300	4.48	Albania1988		2,126,000,000	769	G.I. Generation
Albania	1988	male	35-54 years	14	314100	4.46	Albania1988		2,126,000,000	769	Silent
Albania	1988	male	55-74 years	4	140200	2.85	Albania1988		2,126,000,000	769	G.I. Generation
Albania	1988	female	15-24 years	8	295600	2.71	Albania1988		2,126,000,000	769	Generation X
Albania	1988	female	55-74 years	3	147500	2.03	Albania1988		2,126,000,000	769	G.I. Generation
Albania	1988	female	25-34 years	5	262400	1.91	Albania1988		2,126,000,000	769	Boomers
Albania	1988	male	25-34 years	5	279900	1.79	Albania1988		2,126,000,000	769	Boomers
Albania	1988	female	35-54 years	4	284500	1.41	Albania1988		2,126,000,000	769	Silent
Albania	1988	female	5-14 years	0	317200	0	Albania1988		2,126,000,000	769	Generation X
Albania	1988	male	5-14 years	0	345000	0	Albania1988		2,126,000,000	769	Generation X

After

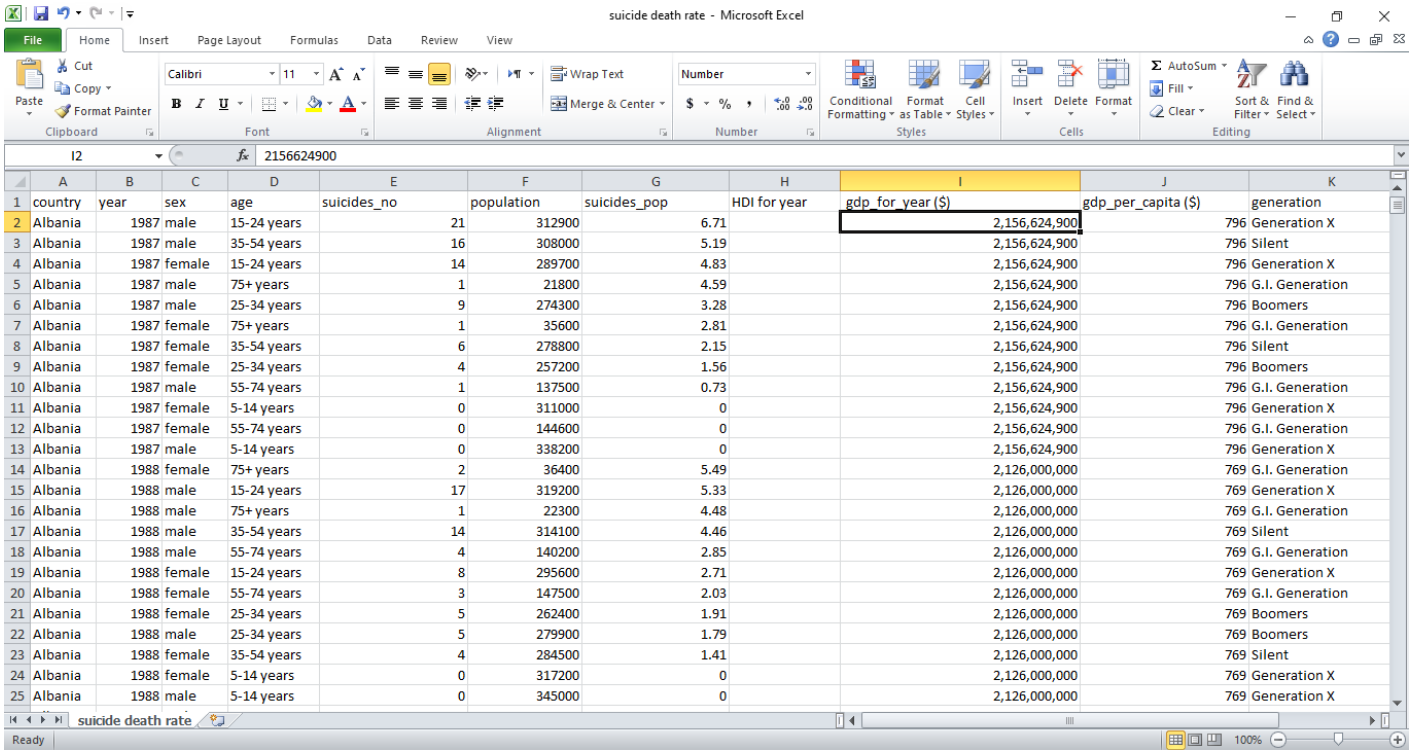


country	year	sex	age	suicides_no	population	suicides_pop	HDI for year	gdp_for_year ($)	gdp_per_capita ($)	generation
Albania	1987	male	15-24 years	21	312900	6.71		2,156,624,900	796	Generation X
Albania	1987	male	35-54 years	16	308000	5.19		2,156,624,900	796	Silent
Albania	1987	female	15-24 years	14	289700	4.83		2,156,624,900	796	Generation X
Albania	1987	male	75+ years	1	21800	4.59		2,156,624,900	796	G.I. Generation
Albania	1987	male	25-34 years	9	274300	3.28		2,156,624,900	796	Boomers
Albania	1987	female	75+ years	1	35600	2.81		2,156,624,900	796	G.I. Generation
Albania	1987	female	35-54 years	6	278800	2.15		2,156,624,900	796	Silent
Albania	1987	female	25-34 years	4	257200	1.56		2,156,624,900	796	Boomers
Albania	1987	male	55-74 years	1	137500	0.73		2,156,624,900	796	G.I. Generation
Albania	1987	female	5-14 years	0	311000	0		2,156,624,900	796	Generation X
Albania	1987	female	55-74 years	0	144600	0		2,156,624,900	796	G.I. Generation
Albania	1987	male	5-14 years	0	338200	0		2,156,624,900	796	Generation X
Albania	1988	female	75+ years	2	36400	5.49		2,126,000,000	769	G.I. Generation
Albania	1988	male	15-24 years	17	319200	5.33		2,126,000,000	769	Generation X
Albania	1988	male	75+ years	1	22300	4.48		2,126,000,000	769	G.I. Generation
Albania	1988	male	35-54 years	14	314100	4.46		2,126,000,000	769	Silent
Albania	1988	male	55-74 years	4	140200	2.85		2,126,000,000	769	G.I. Generation
Albania	1988	female	15-24 years	8	295600	2.71		2,126,000,000	769	Generation X
Albania	1988	female	55-74 years	3	147500	2.03		2,126,000,000	769	G.I. Generation
Albania	1988	female	25-34 years	5	262400	1.91		2,126,000,000	769	Boomers
Albania	1988	male	25-34 years	5	279900	1.79		2,126,000,000	769	Boomers
Albania	1988	female	35-54 years	4	284500	1.41		2,126,000,000	769	Silent
Albania	1988	female	5-14 years	0	317200	0		2,126,000,000	769	Generation X
Albania	1988	male	5-14 years	0	345000	0		2,126,000,000	769	Generation X

In this step column **country-year** is removed because data are not necessary for analysis.

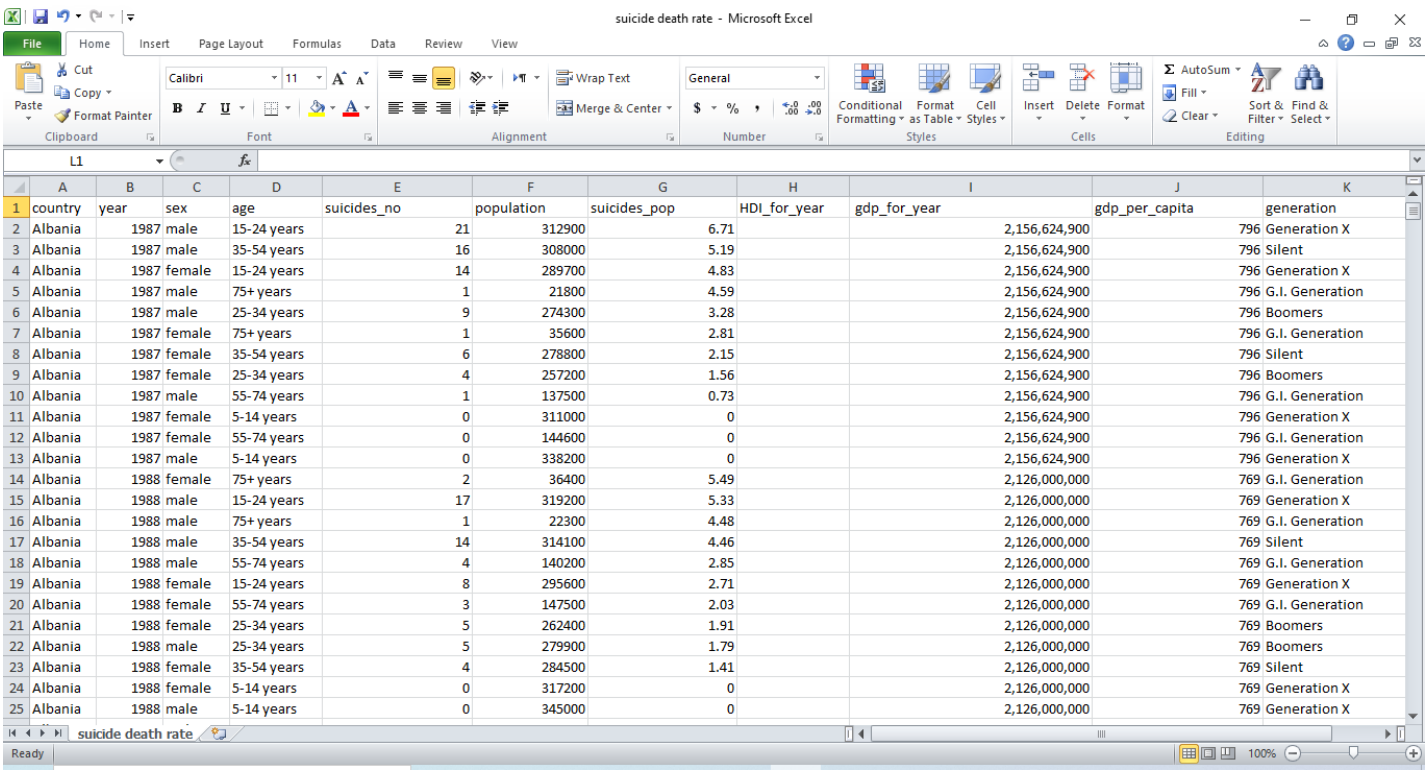
Step 2: Modifying columns name

Before



	country	year	sex	age	suicides_no	population	suicides_pop	HDI for year	gdp_for_year ($)	gdp_per_capita ($)	generation
2	Albania	1987	male	15-24 years	21	312900	6.71		2,156,624,900	796	Generation X
3	Albania	1987	male	35-54 years	16	308000	5.19		2,156,624,900	796	Silent
4	Albania	1987	female	15-24 years	14	289700	4.83		2,156,624,900	796	Generation X
5	Albania	1987	male	75+ years	1	21800	4.59		2,156,624,900	796	G.I. Generation
6	Albania	1987	male	25-34 years	9	274300	3.28		2,156,624,900	796	Boomers
7	Albania	1987	female	75+ years	1	35600	2.81		2,156,624,900	796	G.I. Generation
8	Albania	1987	female	35-54 years	6	278800	2.15		2,156,624,900	796	Silent
9	Albania	1987	female	25-34 years	4	257200	1.56		2,156,624,900	796	Boomers
10	Albania	1987	male	55-74 years	1	137500	0.73		2,156,624,900	796	G.I. Generation
11	Albania	1987	female	5-14 years	0	311000	0		2,156,624,900	796	Generation X
12	Albania	1987	female	55-74 years	0	144600	0		2,156,624,900	796	G.I. Generation
13	Albania	1987	male	5-14 years	0	338200	0		2,156,624,900	796	Generation X
14	Albania	1988	female	75+ years	2	36400	5.49		2,126,000,000	769	G.I. Generation
15	Albania	1988	male	15-24 years	17	319200	5.33		2,126,000,000	769	Generation X
16	Albania	1988	male	75+ years	1	22300	4.48		2,126,000,000	769	G.I. Generation
17	Albania	1988	male	35-54 years	14	314100	4.46		2,126,000,000	769	Silent
18	Albania	1988	male	55-74 years	4	140200	2.85		2,126,000,000	769	G.I. Generation
19	Albania	1988	female	15-24 years	8	295600	2.71		2,126,000,000	769	Generation X
20	Albania	1988	female	55-74 years	3	147500	2.03		2,126,000,000	769	G.I. Generation
21	Albania	1988	female	25-34 years	5	262400	1.91		2,126,000,000	769	Boomers
22	Albania	1988	male	25-34 years	5	279900	1.79		2,126,000,000	769	Boomers
23	Albania	1988	female	35-54 years	4	284500	1.41		2,126,000,000	769	Silent
24	Albania	1988	female	5-14 years	0	317200	0		2,126,000,000	769	Generation X
25	Albania	1988	male	5-14 years	0	345000	0		2,126,000,000	769	Generation X

After

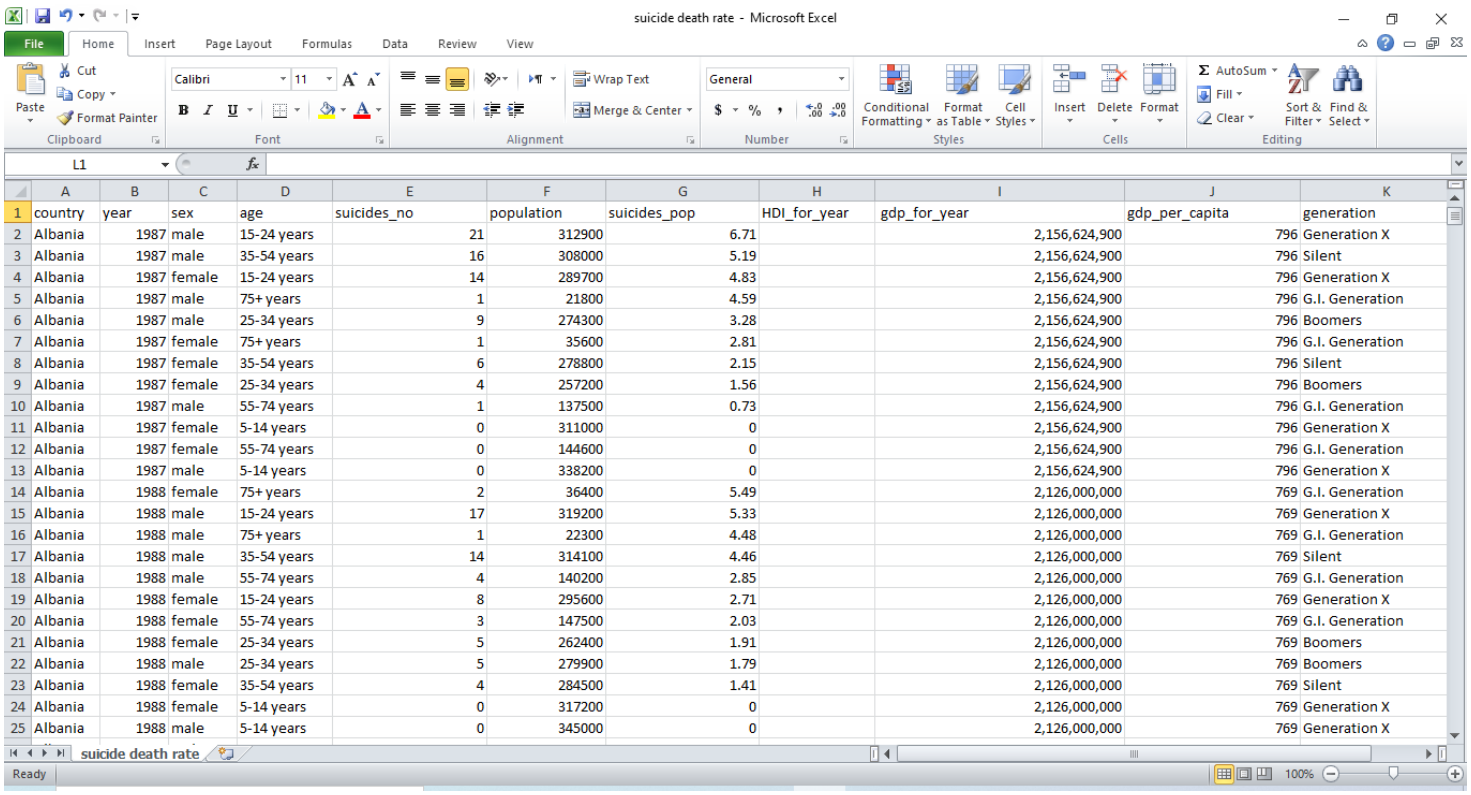


	country	year	sex	age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_capita	generation
2	Albania	1987	male	15-24 years	21	312900	6.71		2,156,624,900	796	Generation X
3	Albania	1987	male	35-54 years	16	308000	5.19		2,156,624,900	796	Silent
4	Albania	1987	female	15-24 years	14	289700	4.83		2,156,624,900	796	Generation X
5	Albania	1987	male	75+ years	1	21800	4.59		2,156,624,900	796	G.I. Generation
6	Albania	1987	male	25-34 years	9	274300	3.28		2,156,624,900	796	Boomers
7	Albania	1987	female	75+ years	1	35600	2.81		2,156,624,900	796	G.I. Generation
8	Albania	1987	female	35-54 years	6	278800	2.15		2,156,624,900	796	Silent
9	Albania	1987	female	25-34 years	4	257200	1.56		2,156,624,900	796	Boomers
10	Albania	1987	male	55-74 years	1	137500	0.73		2,156,624,900	796	G.I. Generation
11	Albania	1987	female	5-14 years	0	311000	0		2,156,624,900	796	Generation X
12	Albania	1987	female	55-74 years	0	144600	0		2,156,624,900	796	G.I. Generation
13	Albania	1987	male	5-14 years	0	338200	0		2,156,624,900	796	Generation X
14	Albania	1988	female	75+ years	2	36400	5.49		2,126,000,000	769	G.I. Generation
15	Albania	1988	male	15-24 years	17	319200	5.33		2,126,000,000	769	Generation X
16	Albania	1988	male	75+ years	1	22300	4.48		2,126,000,000	769	G.I. Generation
17	Albania	1988	male	35-54 years	14	314100	4.46		2,126,000,000	769	Silent
18	Albania	1988	male	55-74 years	4	140200	2.85		2,126,000,000	769	G.I. Generation
19	Albania	1988	female	15-24 years	8	295600	2.71		2,126,000,000	769	Generation X
20	Albania	1988	female	55-74 years	3	147500	2.03		2,126,000,000	769	G.I. Generation
21	Albania	1988	female	25-34 years	5	262400	1.91		2,126,000,000	769	Boomers
22	Albania	1988	male	25-34 years	5	279900	1.79		2,126,000,000	769	Boomers
23	Albania	1988	female	35-54 years	4	284500	1.41		2,126,000,000	769	Silent
24	Albania	1988	female	5-14 years	0	317200	0		2,126,000,000	769	Generation X
25	Albania	1988	male	5-14 years	0	345000	0		2,126,000,000	769	Generation X

In this step symbols like ($) in **gdp_per_capita ($), gdp_for_year ($)** columns are removed**.** In **suicides/100k pop** column name '**/100k**' is replaced with underscore (_). Same as replacement in **suicides/100k pop** column for **HDI for year** column also space between each column name words is replaced with (_).

Step 3: Modifying row values

Before:



country	year	sex	age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_capita	generation
Albania	1987	male	15-24 years	21	312900	6.71		2,156,624,900	796	Generation X
Albania	1987	male	35-54 years	16	308000	5.19		2,156,624,900	796	Silent
Albania	1987	female	15-24 years	14	289700	4.83		2,156,624,900	796	Generation X
Albania	1987	male	75+ years	1	21800	4.59		2,156,624,900	796	G.I. Generation
Albania	1987	male	25-34 years	9	274300	3.28		2,156,624,900	796	Boomers
Albania	1987	female	75+ years	1	35600	2.81		2,156,624,900	796	G.I. Generation
Albania	1987	female	35-54 years	6	278800	2.15		2,156,624,900	796	Silent
Albania	1987	female	25-34 years	4	257200	1.56		2,156,624,900	796	Boomers
Albania	1987	male	55-74 years	1	137500	0.73		2,156,624,900	796	G.I. Generation
Albania	1987	female	5-14 years	0	311000	0		2,156,624,900	796	Generation X
Albania	1987	female	55-74 years	0	144600	0		2,156,624,900	796	G.I. Generation
Albania	1987	male	5-14 years	0	338200	0		2,156,624,900	796	Generation X
Albania	1988	female	75+ years	2	36400	5.49		2,126,000,000	769	G.I. Generation
Albania	1988	male	15-24 years	17	319200	5.33		2,126,000,000	769	Generation X
Albania	1988	male	75+ years	1	22300	4.48		2,126,000,000	769	G.I. Generation
Albania	1988	male	35-54 years	14	314100	4.46		2,126,000,000	769	Silent
Albania	1988	male	55-74 years	4	140200	2.85		2,126,000,000	769	G.I. Generation
Albania	1988	female	15-24 years	8	295600	2.71		2,126,000,000	769	Generation X
Albania	1988	female	55-74 years	3	147500	2.03		2,126,000,000	769	G.I. Generation
Albania	1988	female	25-34 years	5	262400	1.91		2,126,000,000	769	Boomers
Albania	1988	male	25-34 years	5	279900	1.79		2,126,000,000	769	Boomers
Albania	1988	female	35-54 years	4	284500	1.41		2,126,000,000	769	Silent
Albania	1988	female	5-14 years	0	317200	0		2,126,000,000	769	Generation X
Albania	1988	male	5-14 years	0	345000	0		2,126,000,000	769	Generation X

After

Importing numpy and pandas

```
In [22]: import pandas as pd
         import numpy as np
```

In this step pandas python library is used because a csv file contains a lot of data so it is very time consuming to clean data of each cell of the csv file. The pandas is a python library which helps to manipulate data.

Reading CSV file using pandas

```
In [38]: #Creating dataframe
         df = pd.read_csv('suicide_death_rate.csv')
```

In pandas library read_csv method is used to read a csv_file.

```
In [39]: df.head()
Out[39]:
```

	country	year	sex	age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_capita	generation
0	Albania	1987	male	15-24 years	21	312900	6.71	NaN	2,156,624,900	796	Generation X
1	Albania	1987	male	35-54 years	16	308000	5.19	NaN	2,156,624,900	796	Silent
2	Albania	1987	female	15-24 years	14	289700	4.83	NaN	2,156,624,900	796	Generation X
3	Albania	1987	male	75+ years	1	21800	4.59	NaN	2,156,624,900	796	G.I. Generation
4	Albania	1987	male	25-34 years	9	274300	3.28	NaN	2,156,624,900	796	Boomers

The head function displays the first five row data of csv file. It also takes parameter to display how many data's the default value is five. As we can see in gdp_for_year columns data there is comma included between numbers.

```
In [42]: df['gdp_for_year'] = df['gdp_for_year'].str.replace(',','').astype(np.int64)
         df["age"] = df["age"].str.replace("5-14 years","05-14 years")
```

The above figure shows code that converts numbers into string and replaces ',' with empty string. Then it again converts string into numbers.

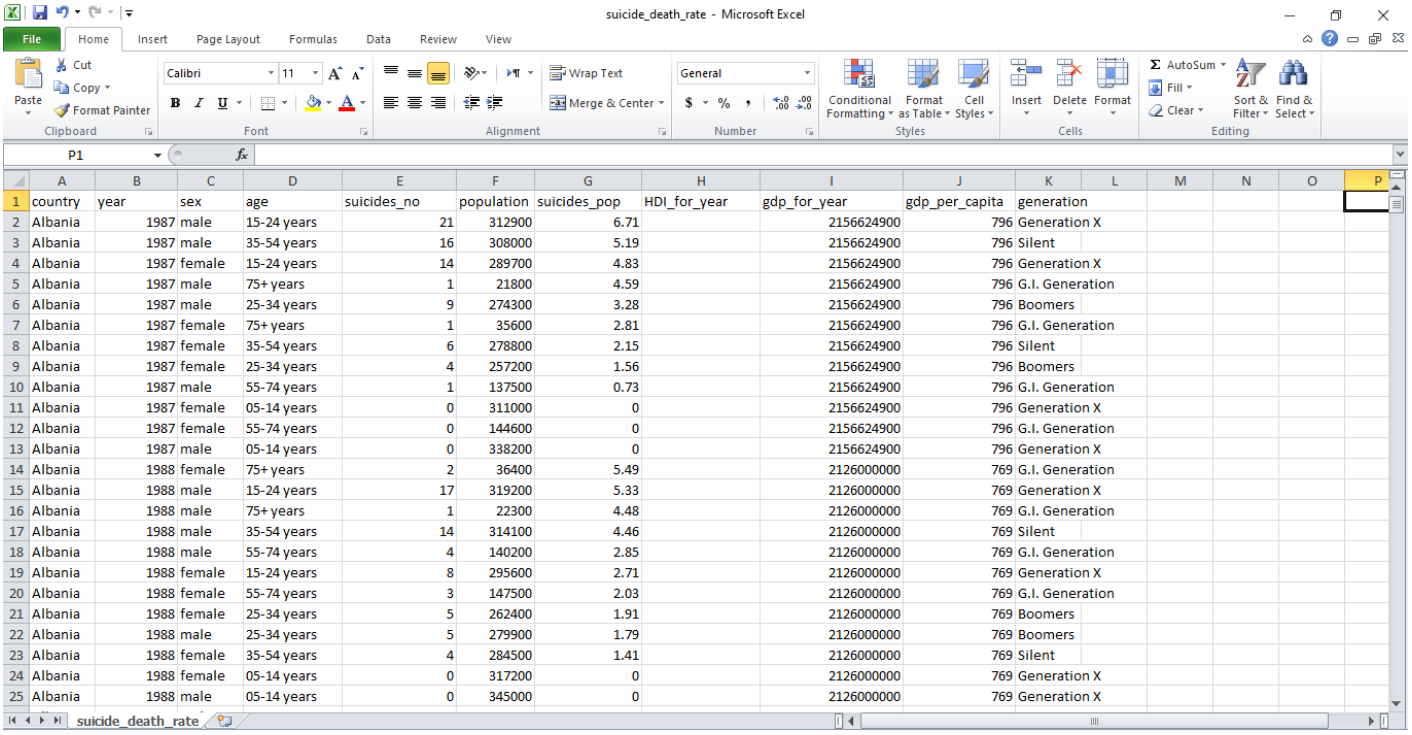
```
In [43]: df.head()
Out[43]:
```

	country	year	sex	age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_capita	generation
0	Albania	1987	male	15-24 years	21	312900	6.71	NaN	2156624900	796	Generation X
1	Albania	1987	male	35-54 years	16	308000	5.19	NaN	2156624900	796	Silent
2	Albania	1987	female	15-24 years	14	289700	4.83	NaN	2156624900	796	Generation X
3	Albania	1987	male	75+ years	1	21800	4.59	NaN	2156624900	796	G.I. Generation
4	Albania	1987	male	25-34 years	9	274300	3.28	NaN	2156624900	796	Boomers

The above figure shows after removing ',' from the data of gdp_for_year column.

```
In [44]: df.to_csv('suicide_death_rate.csv',index=False)
```

After changes are made in the data frame the data frame will be converted in a csv file and replaced with an old csv file. To replace an old csv file with a new one the to_csv method is used which takes two parameters. The first parameter is for a new csv file name which is the same as the old one and another parameter is an option for including index inside of a csv file or not.

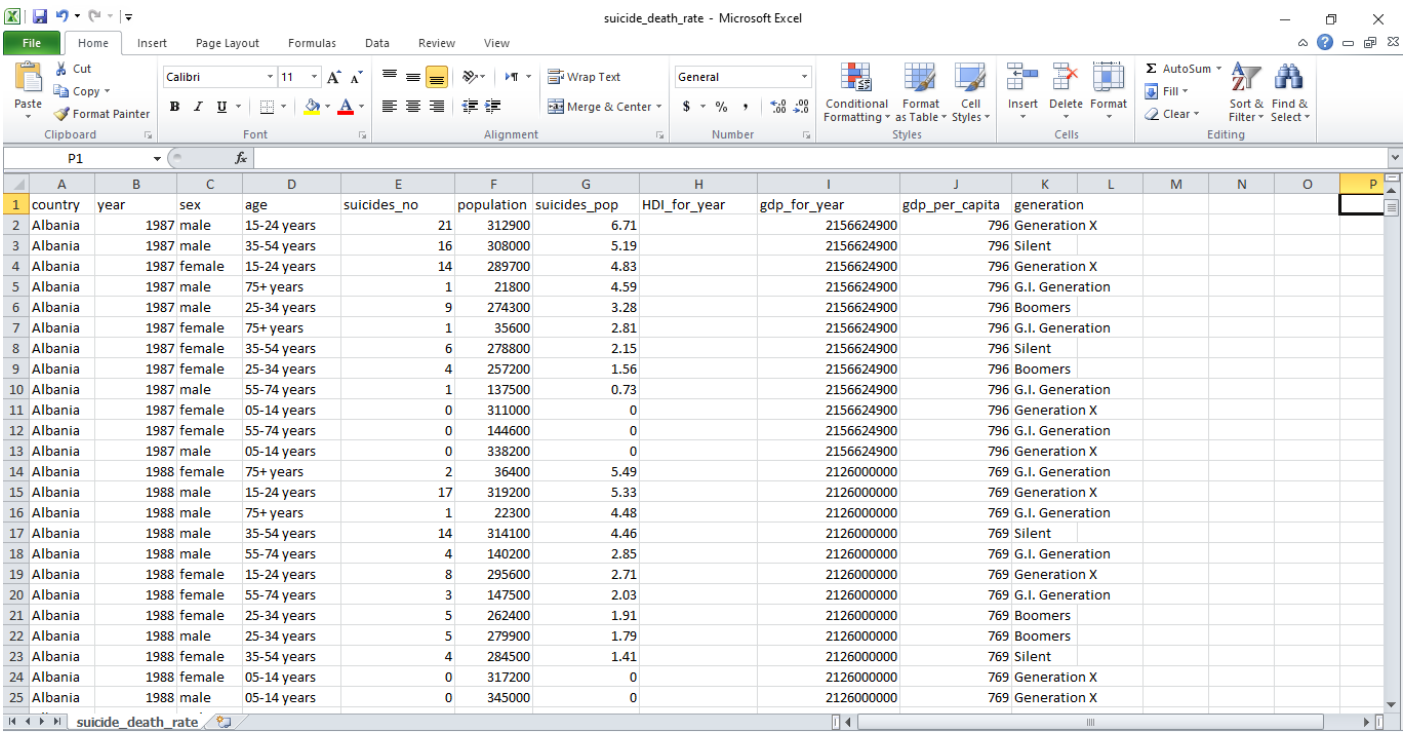


country	year	sex	age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_capita	generation
Albania	1987	male	15-24 years	21	312900	6.71		2156624900	796	Generation X
Albania	1987	male	35-54 years	16	308000	5.19		2156624900	796	Silent
Albania	1987	female	15-24 years	14	289700	4.83		2156624900	796	Generation X
Albania	1987	male	75+ years	1	21800	4.59		2156624900	796	G.I. Generation
Albania	1987	male	25-34 years	9	274300	3.28		2156624900	796	Boomers
Albania	1987	female	75+ years	1	35600	2.81		2156624900	796	G.I. Generation
Albania	1987	female	35-54 years	6	278800	2.15		2156624900	796	Silent
Albania	1987	female	25-34 years	4	257200	1.56		2156624900	796	Boomers
Albania	1987	male	55-74 years	1	137500	0.73		2156624900	796	G.I. Generation
Albania	1987	female	05-14 years	0	311000	0		2156624900	796	Generation X
Albania	1987	female	55-74 years	0	144600	0		2156624900	796	G.I. Generation
Albania	1987	male	05-14 years	0	338200	0		2156624900	796	Generation X
Albania	1988	female	75+ years	2	36400	5.49		2126000000	769	G.I. Generation
Albania	1988	male	15-24 years	17	319200	5.33		2126000000	769	Generation X
Albania	1988	male	75+ years	1	22300	4.48		2126000000	769	G.I. Generation
Albania	1988	male	35-54 years	14	314100	4.46		2126000000	769	Silent
Albania	1988	male	55-74 years	4	140200	2.85		2126000000	769	G.I. Generation
Albania	1988	female	15-24 years	8	295600	2.71		2126000000	769	Generation X
Albania	1988	female	55-74 years	3	147500	2.03		2126000000	769	G.I. Generation
Albania	1988	female	25-34 years	5	262400	1.91		2126000000	769	Boomers
Albania	1988	male	25-34 years	5	279900	1.79		2126000000	769	Boomers
Albania	1988	female	35-54 years	4	284500	1.41		2126000000	769	Silent
Albania	1988	female	05-14 years	0	317200	0		2126000000	769	Generation X
Albania	1988	male	05-14 years	0	345000	0		2126000000	769	Generation X

The above figure shows that after replacing the old csv file with a new one.

Step 4: Filling value in empty cell

Before:



country	year	sex	age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_capita	generation
Albania	1987	male	15-24 years	21	312900	6.71		2156624900	796	Generation X
Albania	1987	male	35-54 years	16	308000	5.19		2156624900	796	Silent
Albania	1987	female	15-24 years	14	289700	4.83		2156624900	796	Generation X
Albania	1987	male	75+ years	1	21800	4.59		2156624900	796	G.I. Generation
Albania	1987	male	25-34 years	9	274300	3.28		2156624900	796	Boomers
Albania	1987	female	75+ years	1	35600	2.81		2156624900	796	G.I. Generation
Albania	1987	female	35-54 years	6	278800	2.15		2156624900	796	Silent
Albania	1987	female	25-34 years	4	257200	1.56		2156624900	796	Boomers
Albania	1987	male	55-74 years	1	137500	0.73		2156624900	796	G.I. Generation
Albania	1987	female	05-14 years	0	311000	0		2156624900	796	Generation X
Albania	1987	female	55-74 years	0	144600	0		2156624900	796	G.I. Generation
Albania	1987	male	05-14 years	0	338200	0		2156624900	796	Generation X
Albania	1988	female	75+ years	2	36400	5.49		2126000000	769	G.I. Generation
Albania	1988	male	15-24 years	17	319200	5.33		2126000000	769	Generation X
Albania	1988	male	75+ years	1	22300	4.48		2126000000	769	G.I. Generation
Albania	1988	male	35-54 years	14	314100	4.46		2126000000	769	Silent
Albania	1988	male	55-74 years	4	140200	2.85		2126000000	769	G.I. Generation
Albania	1988	female	15-24 years	8	295600	2.71		2126000000	769	Generation X
Albania	1988	female	55-74 years	3	147500	2.03		2126000000	769	G.I. Generation
Albania	1988	female	25-34 years	5	262400	1.91		2126000000	769	Boomers
Albania	1988	male	25-34 years	5	279900	1.79		2126000000	769	Boomers
Albania	1988	female	35-54 years	4	284500	1.41		2126000000	769	Silent
Albania	1988	female	05-14 years	0	317200	0		2126000000	769	Generation X
Albania	1988	male	05-14 years	0	345000	0		2126000000	769	Generation X

```
In [198]: df.head(135)
```

Out[198]:

	country	year	sex	age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_capita	generation
0	Albania	1987	male	15-24 years	21	312900	6.71	NaN	2156624900	796	Generation X
1	Albania	1987	male	35-54 years	16	308000	5.19	NaN	2156624900	796	Silent
2	Albania	1987	female	15-24 years	14	289700	4.83	NaN	2156624900	796	Generation X
3	Albania	1987	male	75+ years	1	21800	4.59	NaN	2156624900	796	G.I. Generation
4	Albania	1987	male	25-34 years	9	274300	3.28	NaN	2156624900	796	Boomers
...	...	...	...	...	...	...	...	...	...	...	...
130	Albania	1999	female	05-14 years	1	365200	0.27	NaN	3414760915	1127	Millenials
131	Albania	1999	male	05-14 years	0	391300	0.00	NaN	3414760915	1127	Millenials
132	Albania	2000	male	25-34 years	17	232000	7.33	0.656	3632043908	1299	Generation X
133	Albania	2000	male	55-74 years	10	177400	5.64	0.656	3632043908	1299	Silent
134	Albania	2000	female	75+ years	2	37800	5.29	0.656	3632043908	1299	G.I. Generation

135 rows × 11 columns

After:

```
In [200]: country = df['country'].unique()
          for c in country:
              c1 = df[df['country'] == c]
              m = c1[c1['HDI_for_year'].notnull()]['HDI_for_year'].mean()
              g = float("{0:.3f}".format(m))

              count = c1['HDI_for_year'].notnull().sum()
              if count == 0:
                  df[df['HDI_for_year'].isnull(),'HDI_for_year'] = 0.000
              else:
                  df.loc[df['HDI_for_year'].isnull(),'HDI_for_year'] = g
```

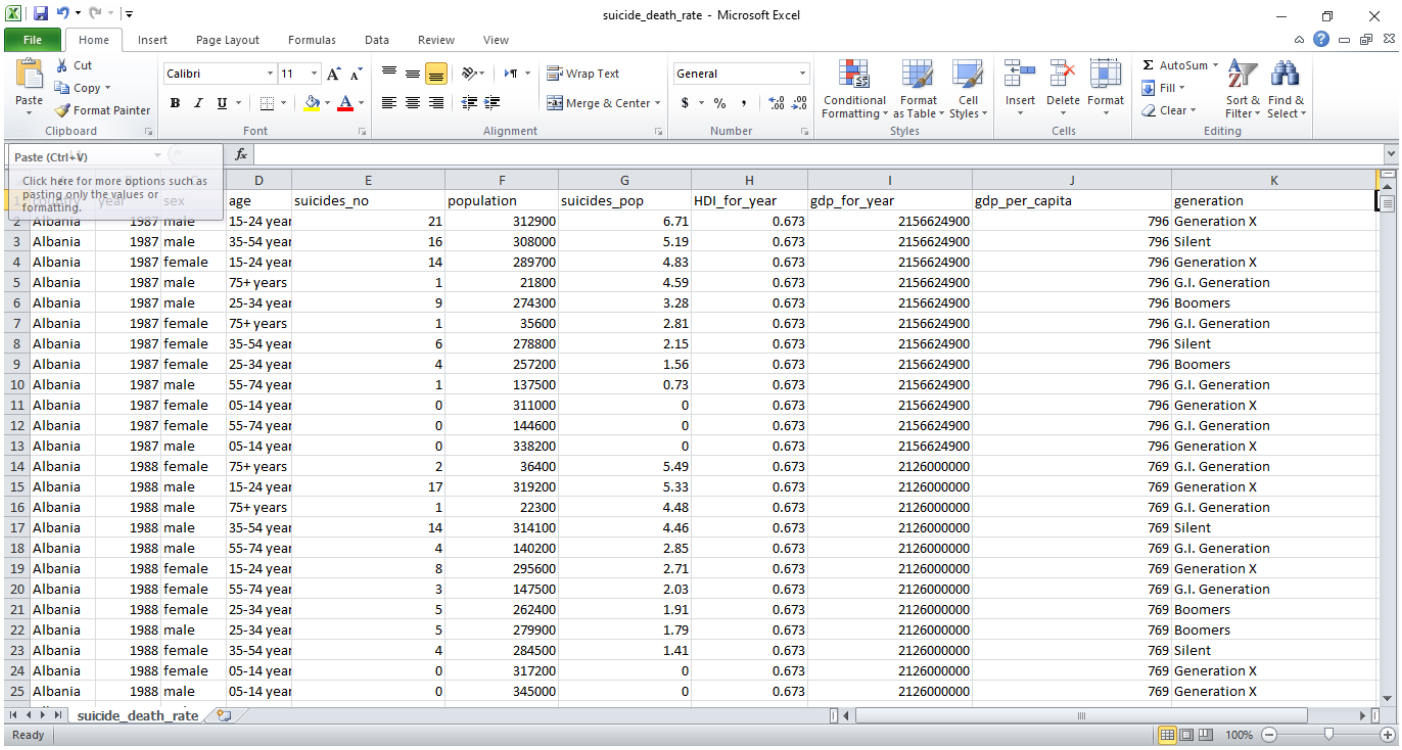
As shown in above code it fills nan value which is in HDI_for_year column with the mean of a HDI of a country.

```
In [202]: df.head(135)
```

Out[202]:

	country	year	sex	age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_capita	generation
0	Albania	1987	male	15-24 years	21	312900	6.71	0.673	2156624900	796	Generation X
1	Albania	1987	male	35-54 years	16	308000	5.19	0.673	2156624900	796	Silent
2	Albania	1987	female	15-24 years	14	289700	4.83	0.673	2156624900	796	Generation X
3	Albania	1987	male	75+ years	1	21800	4.59	0.673	2156624900	796	G.I. Generation
4	Albania	1987	male	25-34 years	9	274300	3.28	0.673	2156624900	796	Boomers
...	...	...	...	...	...	...	...	...	...	...	...
130	Albania	1999	female	05-14 years	1	365200	0.27	0.673	3414760915	1127	Millenials
131	Albania	1999	male	05-14 years	0	391300	0.00	0.673	3414760915	1127	Millenials
132	Albania	2000	male	25-34 years	17	232000	7.33	0.656	3632043908	1299	Generation X
133	Albania	2000	male	55-74 years	10	177400	5.64	0.656	3632043908	1299	Silent
134	Albania	2000	female	75+ years	2	37800	5.29	0.656	3632043908	1299	G.I. Generation

135 rows × 11 columns

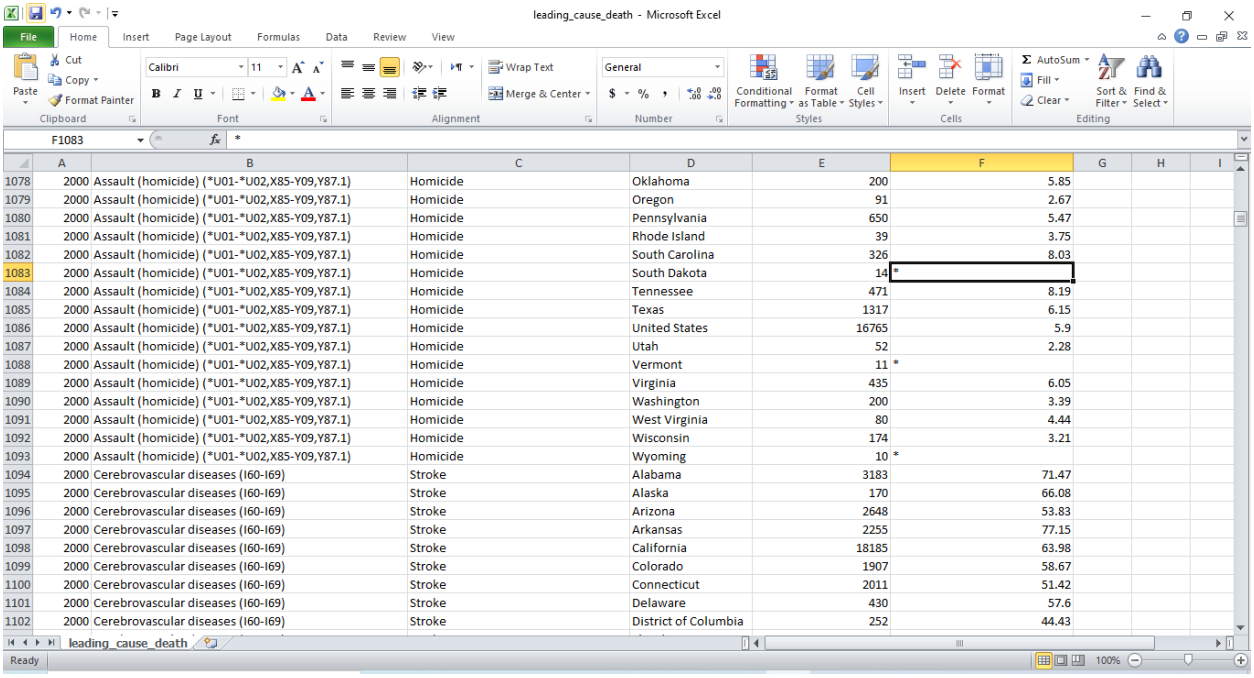


				D	E	F	G	H	I	J	K
				age	suicides_no	population	suicides_pop	HDI_for_year	gdp_for_year	gdp_per_capita	generation
2	Albania	1987	male	15-24 year	21	312900	6.71	0.673	2156624900	796	Generation X
3	Albania	1987	male	35-54 year	16	308000	5.19	0.673	2156624900	796	Silent
4	Albania	1987	female	15-24 year	14	289700	4.83	0.673	2156624900	796	Generation X
5	Albania	1987	male	75+ years	1	21800	4.59	0.673	2156624900	796	G.I. Generation
6	Albania	1987	male	25-34 year	9	274300	3.28	0.673	2156624900	796	Boomers
7	Albania	1987	female	75+ years	1	35600	2.81	0.673	2156624900	796	G.I. Generation
8	Albania	1987	female	35-54 year	6	278800	2.15	0.673	2156624900	796	Silent
9	Albania	1987	female	25-34 year	4	257200	1.56	0.673	2156624900	796	Boomers
10	Albania	1987	male	55-74 year	1	137500	0.73	0.673	2156624900	796	G.I. Generation
11	Albania	1987	female	05-14 year	0	311000	0	0.673	2156624900	796	Generation X
12	Albania	1987	female	55-74 year	0	144600	0	0.673	2156624900	796	G.I. Generation
13	Albania	1987	male	05-14 year	0	338200	0	0.673	2156624900	796	Generation X
14	Albania	1988	female	75+ years	2	36400	5.49	0.673	2126000000	769	G.I. Generation
15	Albania	1988	male	15-24 year	17	319200	5.33	0.673	2126000000	769	Generation X
16	Albania	1988	male	75+ years	1	22300	4.48	0.673	2126000000	769	G.I. Generation
17	Albania	1988	male	35-54 year	14	314100	4.46	0.673	2126000000	769	Silent
18	Albania	1988	male	55-74 year	4	140200	2.85	0.673	2126000000	769	G.I. Generation
19	Albania	1988	female	15-24 year	8	295600	2.71	0.673	2126000000	769	Generation X
20	Albania	1988	female	55-74 year	3	147500	2.03	0.673	2126000000	769	G.I. Generation
21	Albania	1988	female	25-34 year	5	262400	1.91	0.673	2126000000	769	Boomers
22	Albania	1988	male	25-34 year	5	279900	1.79	0.673	2126000000	769	Boomers
23	Albania	1988	female	35-54 year	4	284500	1.41	0.673	2126000000	769	Silent
24	Albania	1988	female	05-14 year	0	317200	0	0.673	2126000000	769	Generation X
25	Albania	1988	male	05-14 year	0	345000	0	0.673	2126000000	769	Generation X

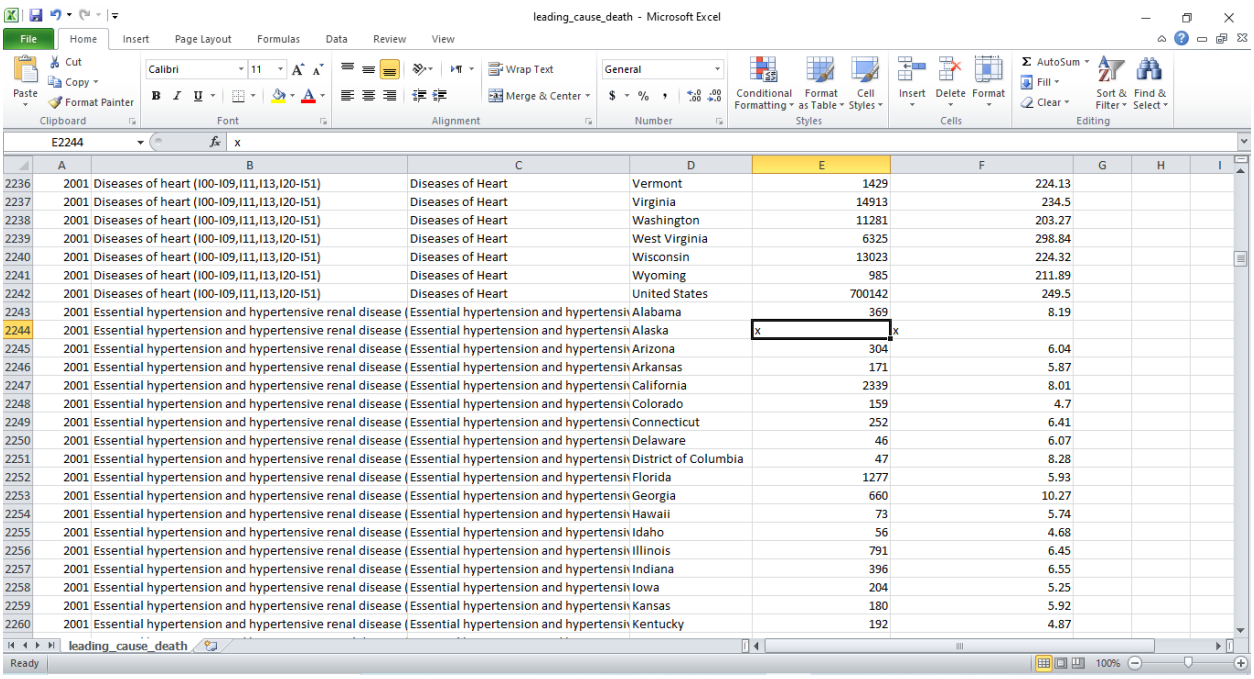
Cleaning dataset for Hadoop

Removing (*, x) from csv file

Before:



	A	B	C	D	E	F	G	H	I
1078	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Oklahoma	200	5.85			
1079	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Oregon	91	2.67			
1080	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Pennsylvania	650	5.47			
1081	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Rhode Island	39	3.75			
1082	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	South Carolina	326	8.03			
1083	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	South Dakota	14	*			
1084	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Tennessee	471	8.19			
1085	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Texas	1317	6.15			
1086	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	United States	16765	5.9			
1087	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Utah	52	2.28			
1088	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	11	*			
1089	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Virginia	435	6.05			
1090	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Washington	200	3.39			
1091	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	West Virginia	80	4.44			
1092	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wisconsin	174	3.21			
1093	2000	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	10	*			
1094	2000	Cerebrovascular diseases (I60-I69)	Stroke	Alabama	3183	71.47			
1095	2000	Cerebrovascular diseases (I60-I69)	Stroke	Alaska	170	66.08			
1096	2000	Cerebrovascular diseases (I60-I69)	Stroke	Arizona	2648	53.83			
1097	2000	Cerebrovascular diseases (I60-I69)	Stroke	Arkansas	2255	77.15			
1098	2000	Cerebrovascular diseases (I60-I69)	Stroke	California	18185	63.98			
1099	2000	Cerebrovascular diseases (I60-I69)	Stroke	Colorado	1907	58.67			
1100	2000	Cerebrovascular diseases (I60-I69)	Stroke	Connecticut	2011	51.42			
1101	2000	Cerebrovascular diseases (I60-I69)	Stroke	Delaware	430	57.6			
1102	2000	Cerebrovascular diseases (I60-I69)	Stroke	District of Columbia	252	44.43			



	A	B	C	D	E	F	G	H	I
2236	2001	Diseases of heart (I00-I09,I11,I13,I20-I51)	Diseases of Heart	Vermont	1429	224.13			
2237	2001	Diseases of heart (I00-I09,I11,I13,I20-I51)	Diseases of Heart	Virginia	14913	234.5			
2238	2001	Diseases of heart (I00-I09,I11,I13,I20-I51)	Diseases of Heart	Washington	11281	203.27			
2239	2001	Diseases of heart (I00-I09,I11,I13,I20-I51)	Diseases of Heart	West Virginia	6325	298.84			
2240	2001	Diseases of heart (I00-I09,I11,I13,I20-I51)	Diseases of Heart	Wisconsin	13023	224.32			
2241	2001	Diseases of heart (I00-I09,I11,I13,I20-I51)	Diseases of Heart	Wyoming	985	211.89			
2242	2001	Diseases of heart (I00-I09,I11,I13,I20-I51)	Diseases of Heart	United States	700142	249.5			
2243	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Alabama	369	8.19			
2244	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Alaska	x	x			
2245	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Arizona	304	6.04			
2246	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Arkansas	171	5.87			
2247	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	California	2339	8.01			
2248	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Colorado	159	4.7			
2249	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Connecticut	252	6.41			
2250	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Delaware	46	6.07			
2251	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	District of Columbia	47	8.28			
2252	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Florida	1277	5.93			
2253	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Georgia	660	10.27			
2254	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Hawaii	73	5.74			
2255	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Idaho	56	4.68			
2256	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Illinois	791	6.45			
2257	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Indiana	396	6.55			
2258	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Iowa	204	5.25			
2259	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Kansas	180	5.92			
2260	2001	Essential hypertension and hypertensive renal disease (	Essential hypertension and hypertensi	Kentucky	192	4.87			

After:

Importing pandas:

```
In [2]: import pandas as pd
        import numpy as np
```

Reading CSV file

```
In [3]: df = pd.read_csv('leading_cause_death.csv')
```

Displaying data frame rows where death column contains ‘x’:

```
In [6]: df[df['deaths'] == 'x']
Out[6]:
```

	year	cause_names	leading_causes	state	deaths	age_adjusted_death_rate
470	1999	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	x	x
1963	2001	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	x	x
2242	2001	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	x	x
2843	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	New Hampshire	x	x
2848	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	x	x
2860	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	x	x
3128	2002	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	x	x
6401	2006	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	x	x
8149	2008	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	x	x
9045	2009	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	x	x
9928	2010	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	x	x
9933	2010	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	x	x
10812	2011	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	x	x
11391	2011	Pneumonitis due to solids and liquids (J69)	Pneumonitis due to solids and liquids	Alaska	x	x
11694	2012	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	x	x

Replacing ‘x’ with 0 in deaths columns data:

```
In [7]: df.loc[df['deaths'] == 'x','deaths'] = 0
```

Displaying data frame rows where deaths columns rows contain value 0.

```
In [8]: df[df['deaths'] == 0]
```

Out[8]:

	year	cause_names	leading_causes	state	deaths	age_adjusted_death_rate
470	1999	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	0	x
1963	2001	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	0	x
2242	2001	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	0	x
2843	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	New Hampshire	0	x
2848	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	0	x
2860	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x
3128	2002	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	0	x
6401	2006	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	0	x
8149	2008	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	0	x
9045	2009	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x
9928	2010	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x
9933	2010	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	0	x
10812	2011	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x
11391	2011	Pneumonitis due to solids and liquids (J69)	Pneumonitis due to solids and liquids	Alaska	0	x
11694	2012	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x

Displaying data frame rows where age_adjusted_death_rate column contains ‘x’:

```
In [9]: df[df['age_adjusted_death_rate'] == 'x']
```

Out[9]:

	year	cause_names	leading_causes	state	deaths	age_adjusted_death_rate
470	1999	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	0	x
1963	2001	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	0	x
2242	2001	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	0	x
2843	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	New Hampshire	0	x
2848	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	0	x
2860	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x
3128	2002	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	0	x
6401	2006	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	0	x
8149	2008	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	0	x
9045	2009	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x
9928	2010	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x
9933	2010	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	0	x
10812	2011	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x
11391	2011	Pneumonitis due to solids and liquids (J69)	Pneumonitis due to solids and liquids	Alaska	0	x
11694	2012	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	x

Replacing ‘x’ with 0 in age_adjusted_death_rate column:

```
In [11]: df.loc[df['age_adjusted_death_rate'] == 'x','age_adjusted_death_rate'] = 0
```

After removing ‘x’ from deaths and age_adjusted_death_rate column.

Out[12]:

	year	cause_names	leading_causes	state	deaths	age_adjusted_death_rate
470	1999	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	0	0
1963	2001	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	0	0
2242	2001	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	0	0
2843	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	New Hampshire	0	0
2848	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	0	0
2860	2002	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	0
3128	2002	Essential hypertension and hypertensive renal ...	Essential hypertension and hypertensive renal ...	Alaska	0	0
6401	2006	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	0	0
8149	2008	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	0	0
9045	2009	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	0
9928	2010	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	0
9933	2010	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	0	0
10812	2011	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	0
11391	2011	Pneumonitis due to solids and liquids (J69)	Pneumonitis due to solids and liquids	Alaska	0	0
11694	2012	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	0	0

Displaying rows from data frame where age_adjusted_death_rate contains ‘*’:

```
In [10]: df[df['age_adjusted_death_rate'] == '*']
```

Out[10]:

	year	cause_names	leading_causes	state	deaths	age_adjusted_death_rate
190	1999	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	13	*
202	1999	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	17	*
207	1999	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	13	*
731	1999	Parkinson's disease (G20-G21)	Parkinson's disease	Alaska	15	*
783	1999	Pneumonitis due to solids and liquids (J69)	Pneumonitis due to solids and liquids	Alaska	10	*
...	...	...	...	...	...	...
12273	2012	Pneumonitis due to solids and liquids (J69)	Pneumonitis due to solids and liquids	Alaska	19	*
12566	2013	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	North Dakota	13	*
12578	2013	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Vermont	10	*
12583	2013	Assault (homicide) (*U01-*U02,X85-Y09,Y87.1)	Homicide	Wyoming	17	*
13157	2013	Pneumonitis due to solids and liquids (J69)	Pneumonitis due to solids and liquids	Alaska	16	*

90 rows × 6 columns

Replacing ‘*’ with 0 in age_adjusted_death_rate column:

```
In [17]: df.loc[df['age_adjusted_death_rate'] == '*','age_adjusted_death_rate'] = 0
```

After removing ‘*’ from age_adjusted_death_rate column:

```
In [19]: df[df['age_adjusted_death_rate'] == '*']
```

Out[19]:

	year	cause_names	leading_causes	state	deaths	age_adjusted_death_rate

Replacing 0 with mean value in age_adjusted_death_rate column:

```
In [27]: states = df['state'].unique()
         for s in states:
             mean = df[df['state']==s]['age_adjusted_death_rate'].astype(float).mean()
             df.loc[(df['state']==s) & (df['age_adjusted_death_rate'] == 0),'age_adjusted_death_rate'] = mean
```

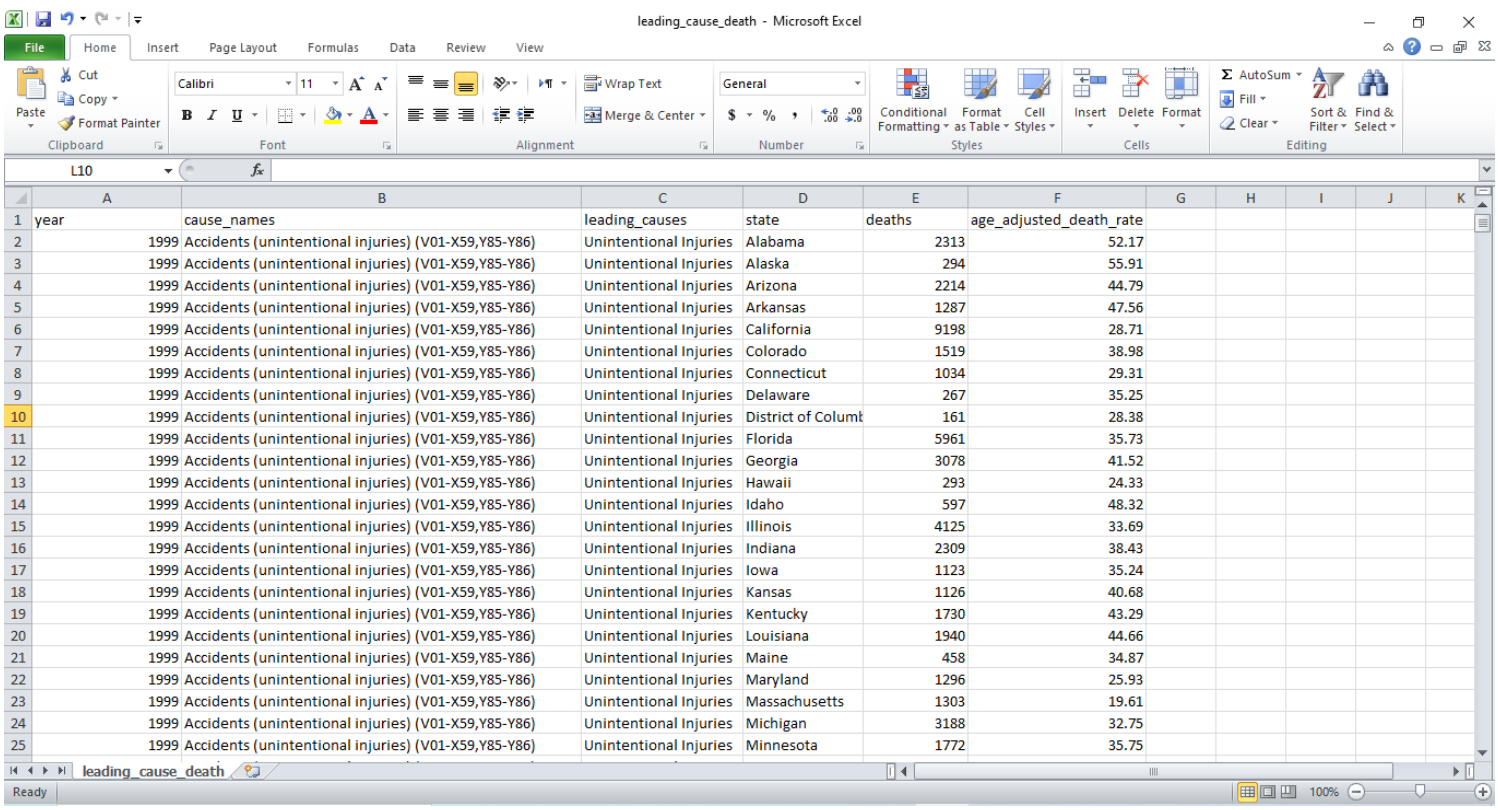
After replacing 0 with mean value in age_adjusted_death_rate column:

```
In [28]: df[df['age_adjusted_death_rate'] == 0]
Out[28]:
```

year	cause_names	leading_causes	state	deaths	age_adjusted_death_rate

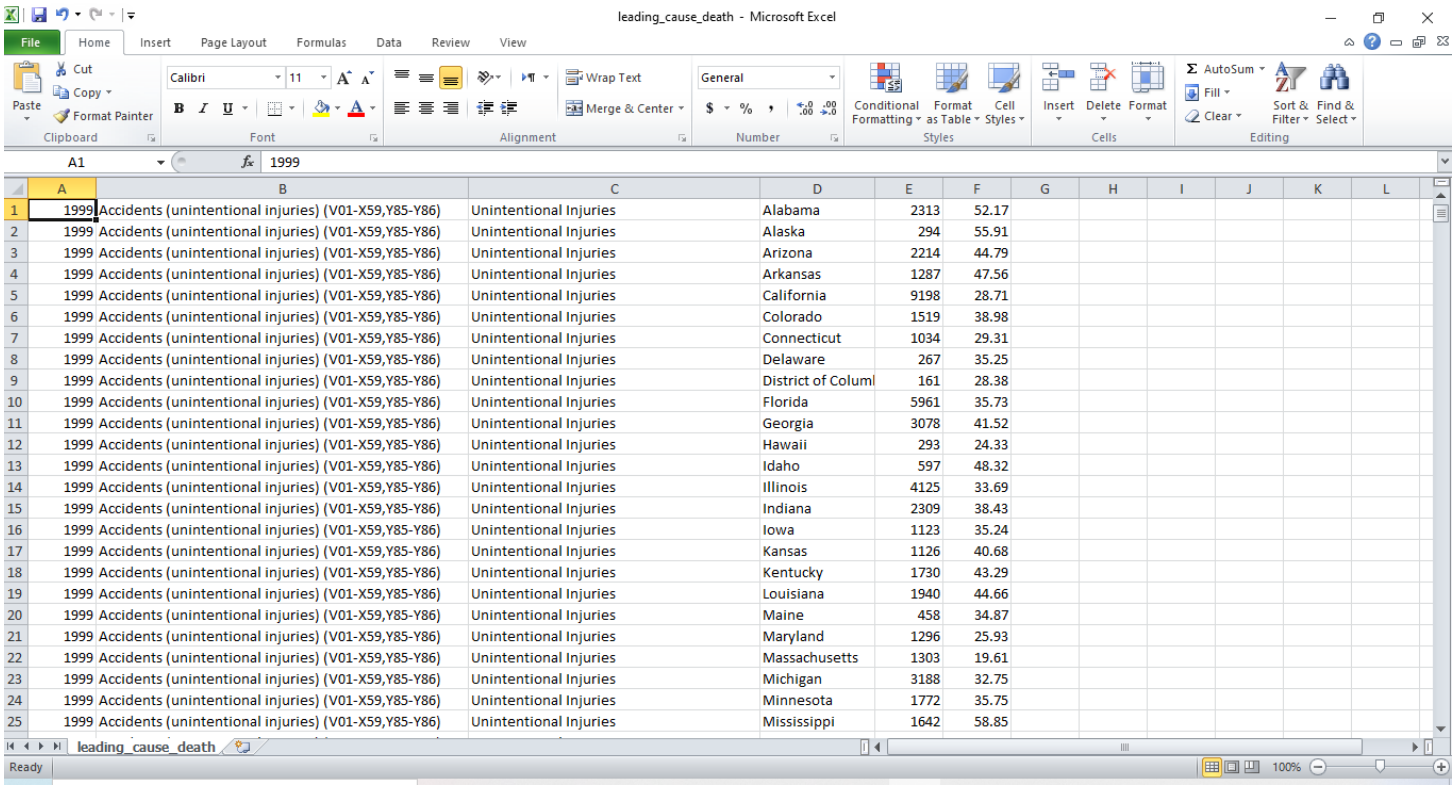
Removing all the columns

Before:



year	cause_names	leading_causes	state	deaths	age_adjusted_death_rate
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Alabama	2313	52.17
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Alaska	294	55.91
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Arizona	2214	44.79
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Arkansas	1287	47.56
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	California	9198	28.71
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Colorado	1519	38.98
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Connecticut	1034	29.31
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Delaware	267	35.25
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	District of Columb	161	28.38
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Florida	5961	35.73
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Georgia	3078	41.52
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Hawaii	293	24.33
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Idaho	597	48.32
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Illinois	4125	33.69
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Indiana	2309	38.43
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Iowa	1123	35.24
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Kansas	1126	40.68
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Kentucky	1730	43.29
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Louisiana	1940	44.66
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Maine	458	34.87
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Maryland	1296	25.93
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Massachusetts	1303	19.61
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Michigan	3188	32.75
1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Minnesota	1772	35.75

After:

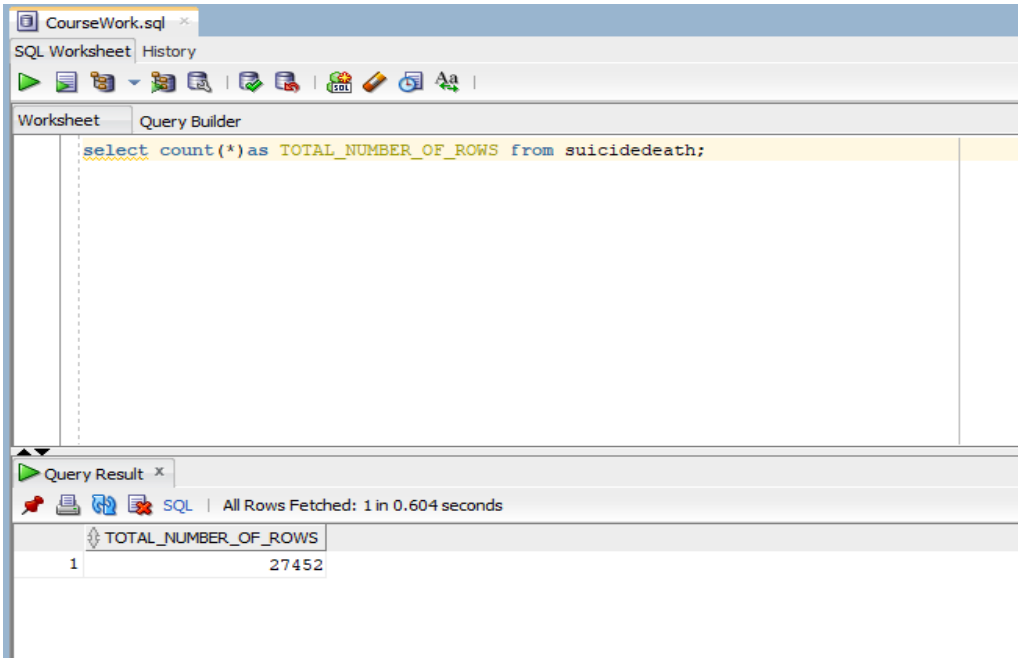


	A	B	C	D	E	F
1	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Alabama	2313	52.17
2	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Alaska	294	55.91
3	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Arizona	2214	44.79
4	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Arkansas	1287	47.56
5	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	California	9198	28.71
6	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Colorado	1519	38.98
7	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Connecticut	1034	29.31
8	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Delaware	267	35.25
9	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	District of Columl	161	28.38
10	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Florida	5961	35.73
11	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Georgia	3078	41.52
12	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Hawaii	293	24.33
13	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Idaho	597	48.32
14	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Illinois	4125	33.69
15	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Indiana	2309	38.43
16	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Iowa	1123	35.24
17	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Kansas	1126	40.68
18	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Kentucky	1730	43.29
19	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Louisiana	1940	44.66
20	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Maine	458	34.87
21	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Maryland	1296	25.93
22	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Massachusetts	1303	19.61
23	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Michigan	3188	32.75
24	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Minnesota	1772	35.75
25	1999	Accidents (unintentional injuries) (V01-X59,Y85-Y86)	Unintentional Injuries	Mississippi	1642	58.85

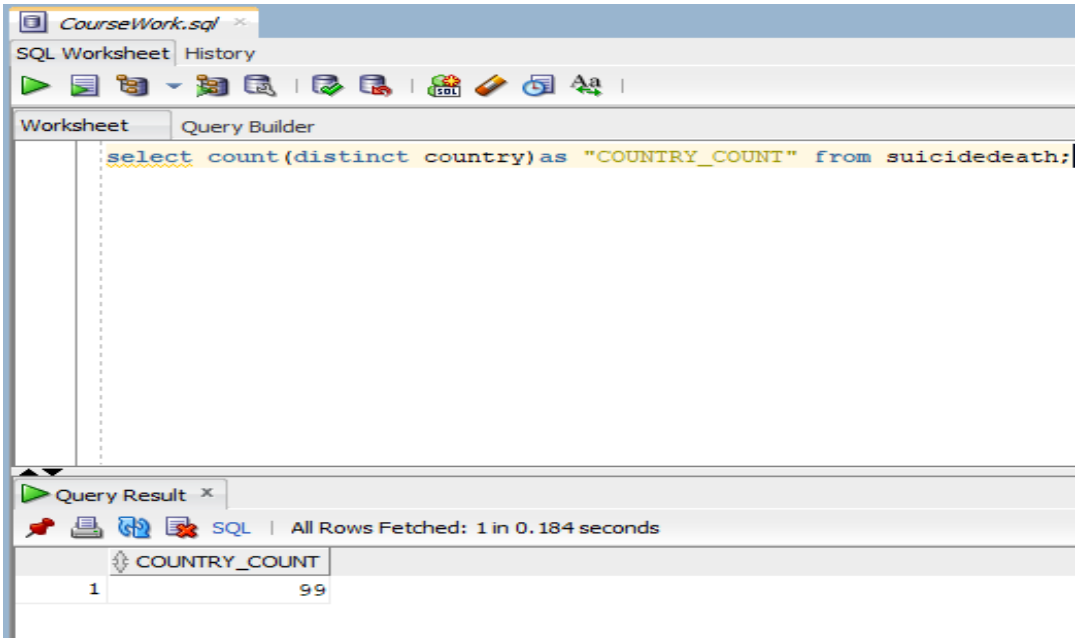
Analysis of Data

Query for oracle

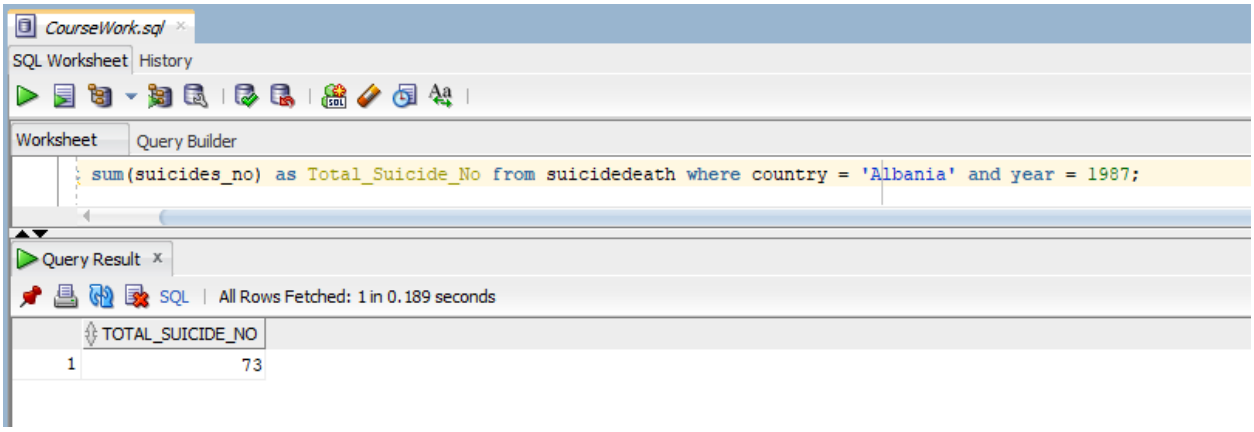
Counting total number of rows present in table.



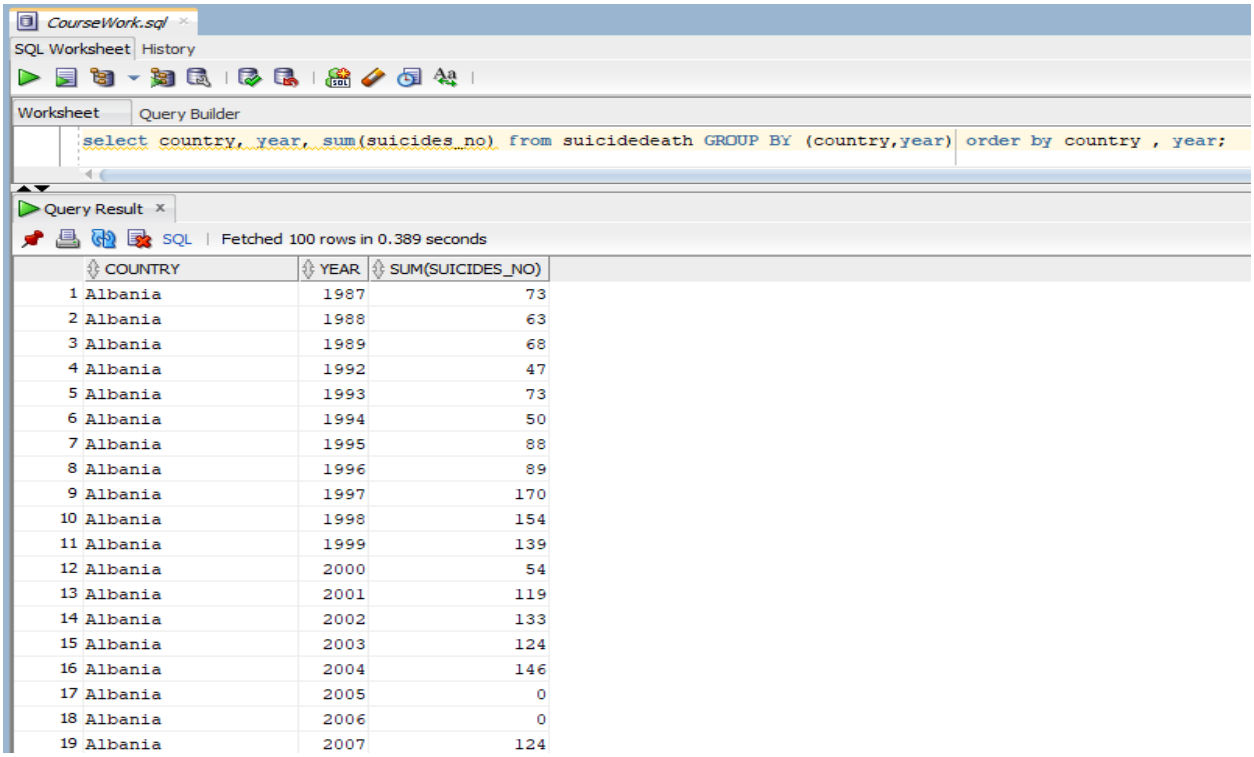
Counting total distinct countries



Summing total number of suicides happening in country Albania in Year 1987.



Grouping by country and year and adding number of suicide happening in each year.



	COUNTRY	YEAR	SUM(SUICIDES_NO)
1	Albania	1987	73
2	Albania	1988	63
3	Albania	1989	68
4	Albania	1992	47
5	Albania	1993	73
6	Albania	1994	50
7	Albania	1995	88
8	Albania	1996	89
9	Albania	1997	170
10	Albania	1998	154
11	Albania	1999	139
12	Albania	2000	54
13	Albania	2001	119
14	Albania	2002	133
15	Albania	2003	124
16	Albania	2004	146
17	Albania	2005	0
18	Albania	2006	0
19	Albania	2007	124

Group by using ROLLUP.

CourseWork.sql

SQL Worksheet | History

CourseWork

Worksheet | Query Builder

```
select country, year, sum(suicides_no) as Total_Suicides_No, sum(population) as Total_Pop, sum(suicides_pop) as Total_suicides_rate
from suicidedeath group by ROLLUP(country,year) order by country, year;
```

Query Result

SQL | Fetched 50 rows in 0.285 seconds

	COUNTRY	YEAR	TOTAL_SUICIDES_NO	TOTAL_POP	TOTAL_SUICIDES_RATE
5	Albania	1993	73	2807300	32.56
6	Albania	1994	50	2849300	32.18
7	Albania	1995	88	2903400	40.55
8	Albania	1996	89	2940200	43.62
9	Albania	1997	170	2977300	77.43
10	Albania	1998	154	3012700	66.52
11	Albania	1999	139	3029700	69.81
12	Albania	2000	54	2796300	30.7
13	Albania	2001	119	2799349	50.62
14	Albania	2002	133	2818839	62.51
15	Albania	2003	124	2843929	58.6
16	Albania	2004	146	2874991	65.39
17	Albania	2005	0	2783320	0
18	Albania	2006	0	2780176	0
19	Albania	2007	124	2770344	65.85
20	Albania	2008	160	2757059	71.05
21	Albania	2009	0	2745735	0
22	Albania	2010	96	2736025	41.66
23	Albania	(null)	1970	62325467	924.76

	COUNTRY	YEAR	TOTAL_SUICIDES_NO	TOTAL_POP	TOTAL_SUICIDES_RATE
2373	Uzbekistan	1995	1485	19599000	111.3
2374	Uzbekistan	1996	1699	19854500	125.06
2375	Uzbekistan	1997	1554	20364300	110.58
2376	Uzbekistan	1998	1620	20861200	108.3
2377	Uzbekistan	1999	1795	21329916	113.33
2378	Uzbekistan	2000	1919	21789067	120.31
2379	Uzbekistan	2001	1914	22231527	115.92
2380	Uzbekistan	2002	1576	22632256	95.28
2381	Uzbekistan	2003	1416	22982883	82.06
2382	Uzbekistan	2004	1251	23300841	68.04
2383	Uzbekistan	2005	1221	23600347	67.81
2384	Uzbekistan	2009	1399	25288102	60.86
2385	Uzbekistan	2010	1464	25651783	63.23
2386	Uzbekistan	2011	1640	25978049	67.14
2387	Uzbekistan	2012	1835	26381830	77.53
2388	Uzbekistan	2013	1950	26838924	78.86
2389	Uzbekistan	2014	2095	27313507	85.12
2390	Uzbekistan	(null)	34803	486422532	2138.17
2391	(null)	(null)	6743775	51300008217	353707.49

Group by using CUBE.

CourseWork.sql

SQL Worksheet | History

CourseWork

Worksheet | Query Builder

```
select country, year, sum(suicides_no) as Total_Suicides_No, sum(population) as Total_Pop, sum(suicides_pop) as Total_suicides_rate
from suicidedeath group by CUBE(country,year) order by country, year;
```

Query Result

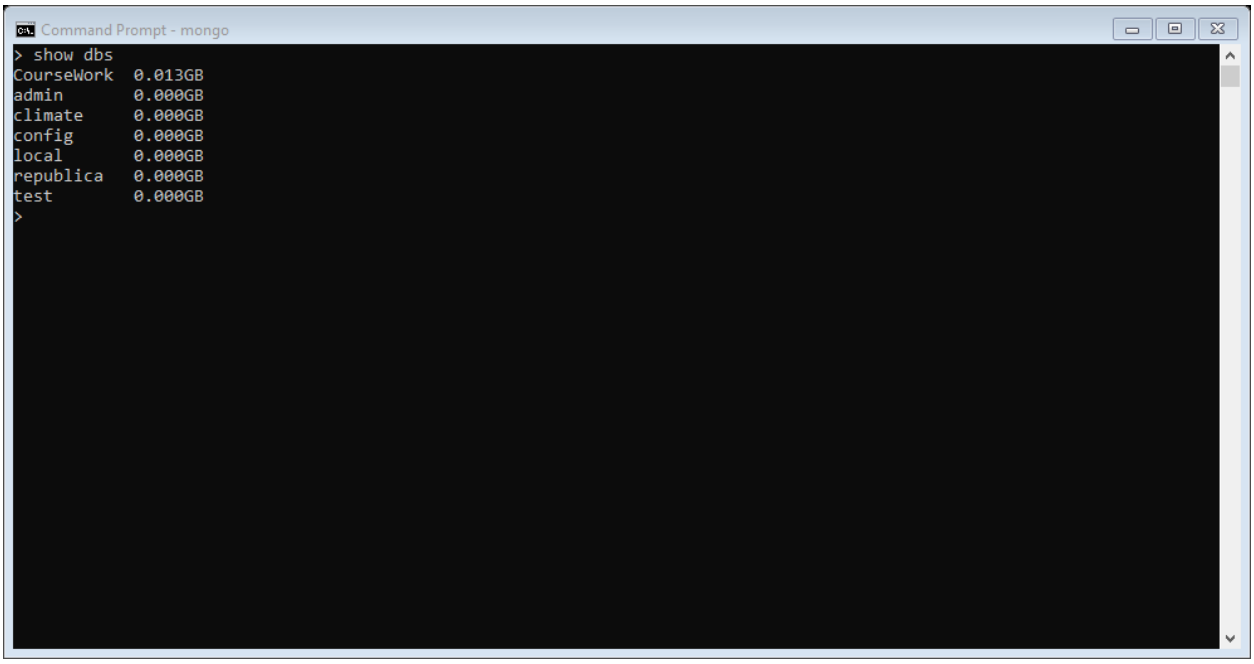
SQL | All Rows Fetched: 2423 in 9.392 seconds

	COUNTRY	YEAR	TOTAL_SUICIDES_NO	TOTAL_POP	TOTAL_SUICIDES_RATE
5	Albania	1993	73	2807300	32.56
6	Albania	1994	50	2849300	32.18
7	Albania	1995	88	2903400	40.55
8	Albania	1996	89	2940200	43.62
9	Albania	1997	170	2977300	77.43
10	Albania	1998	154	3012700	66.52
11	Albania	1999	139	3029700	69.81
12	Albania	2000	54	2796300	30.7
13	Albania	2001	119	2799349	50.62
14	Albania	2002	133	2818839	62.51
15	Albania	2003	124	2843929	58.6
16	Albania	2004	146	2874991	65.39
17	Albania	2005	0	2783320	0
18	Albania	2006	0	2780176	0
19	Albania	2007	124	2770344	65.85
20	Albania	2008	160	2757059	71.05
21	Albania	2009	0	2745735	0
22	Albania	2010	96	2736025	41.66
23	Albania	(null)	1970	62325467	924.76

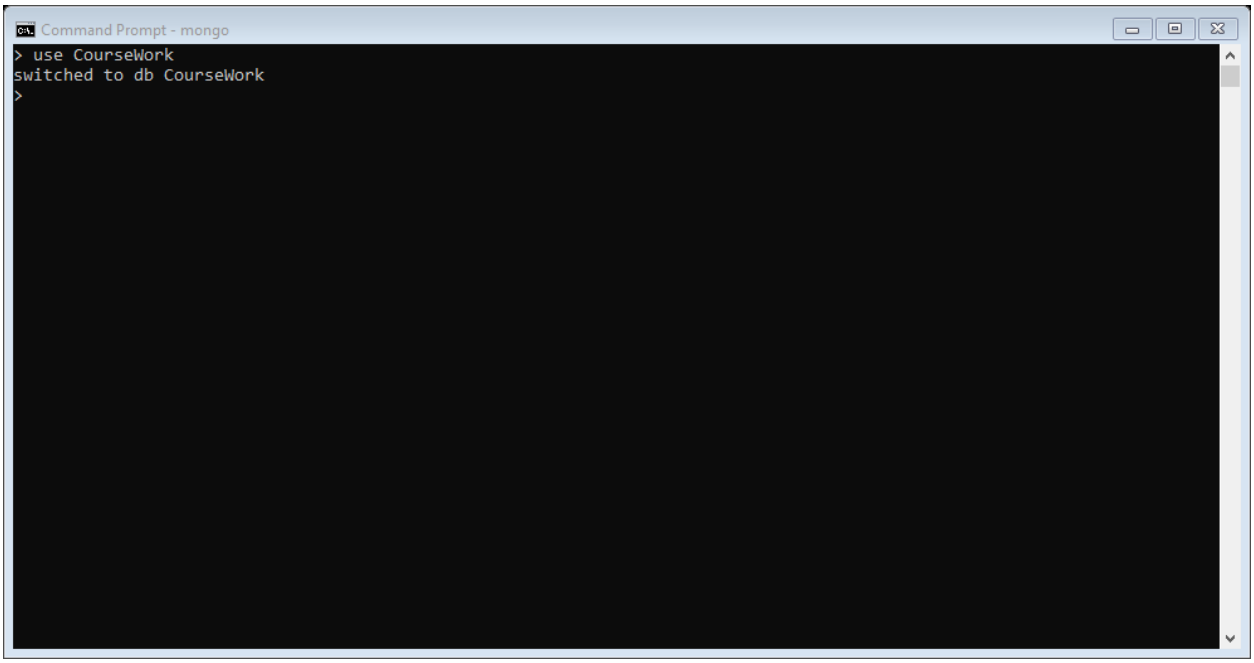
	COUNTRY	YEAR	TOTAL_SUICIDES_NO	TOTAL_POP	TOTAL_SUICIDES_RATE
2393	(null)	1987	126842	1094937690	7545.45
2394	(null)	1988	121026	1054094424	7473.13
2395	(null)	1989	160244	1225514347	8036.54
2396	(null)	1990	193356	1466525959	9806.69
2397	(null)	1991	198020	1489988384	10321.06
2398	(null)	1992	211473	1569539447	10528.88
2399	(null)	1993	221565	1530416654	10790.29
2400	(null)	1994	232063	1548749372	11483.79
2401	(null)	1995	243537	1591463129	14574.85
2402	(null)	1996	246717	1662171567	14057.15
2403	(null)	1997	240742	1702895264	13787.81
2404	(null)	1998	248121	1717905900	13649.25
2405	(null)	1999	254540	1769078447	13999.03
2406	(null)	2000	254340	1793813684	13983.77
2407	(null)	2001	250644	1755468148	14115.55
2408	(null)	2002	256088	1822055016	14024.79
2409	(null)	2003	256075	1838359728	13576.67
2410	(null)	2004	240857	1745147873	12535.34
2411	(null)	2005	234366	1734782263	12058.87

Query for MongoDB

Display store database



Using CouseWork database.



Displaying collections of CourseWork database.

```
Command Prompt - mongo
> show collections
corona
>
```

Renaming Column

Before:

```
Command Prompt - mongo
> db.corona.findOne()
{
        "_id" : ObjectId("6081070da2d36724e9fc1348"),
        "SNo" : 1,
        "ObservationDate" : "01/22/2020",
        "Province/State" : "Anhui",
        "Country/Region" : "Mainland China",
        "Last Update" : "1/22/2020 17:00",
        "Confirmed" : 1,
        "Deaths" : 0,
        "Recovered" : 0
}
>
```

After:

```
Command Prompt - mongo
> db.corona.updateMany({},{$rename:{'Province/State':'State','Country/Region':'Country','Last Update':'Last_Update'}})
{ "acknowledged" : true, "matchedCount" : 236017, "modifiedCount" : 236017 }
>
```

```
Command Prompt - mongo
> db.corona.findOne()
{
        "_id" : ObjectId("6081070da2d36724e9fc1348"),
        "SNo" : 1,
        "ObservationDate" : "01/22/2020",
        "Confirmed" : 1,
        "Deaths" : 0,
        "Recovered" : 0,
        "Country" : "Mainland China",
        "Last_Update" : "1/22/2020 17:00",
        "State" : "Anhui"
}
>
```

Displaying all the data present in corona collection.

```
Command Prompt - mongo
> db.corona.find()
{ "_id" : ObjectId("6081070da2d36724e9fc1348"), "SNo" : 1, "ObservationDate" : "01/22/2020", "Confirmed" : 1, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Anhui" }
{ "_id" : ObjectId("6081070da2d36724e9fc1349"), "SNo" : 2, "ObservationDate" : "01/22/2020", "Confirmed" : 14, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Beijing" }
{ "_id" : ObjectId("6081070da2d36724e9fc134a"), "SNo" : 3, "ObservationDate" : "01/22/2020", "Confirmed" : 6, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Chongqing" }
{ "_id" : ObjectId("6081070da2d36724e9fc134b"), "SNo" : 4, "ObservationDate" : "01/22/2020", "Confirmed" : 1, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Fujian" }
{ "_id" : ObjectId("6081070da2d36724e9fc134c"), "SNo" : 5, "ObservationDate" : "01/22/2020", "Confirmed" : 0, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Gansu" }
{ "_id" : ObjectId("6081070da2d36724e9fc134d"), "SNo" : 6, "ObservationDate" : "01/22/2020", "Confirmed" : 26, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Guangdong" }
{ "_id" : ObjectId("6081070da2d36724e9fc134e"), "SNo" : 7, "ObservationDate" : "01/22/2020", "Confirmed" : 2, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Guangxi" }
{ "_id" : ObjectId("6081070da2d36724e9fc134f"), "SNo" : 8, "ObservationDate" : "01/22/2020", "Confirmed" : 1, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Guizhou" }
{ "_id" : ObjectId("6081070da2d36724e9fc1350"), "SNo" : 9, "ObservationDate" : "01/22/2020", "Confirmed" : 4, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Hainan" }
{ "_id" : ObjectId("6081070da2d36724e9fc1351"), "SNo" : 10, "ObservationDate" : "01/22/2020", "Confirmed" : 1, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Hebei" }
{ "_id" : ObjectId("6081070da2d36724e9fc1352"), "SNo" : 11, "ObservationDate" : "01/22/2020", "Confirmed" : 0, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Heilongjiang" }
{ "_id" : ObjectId("6081070da2d36724e9fc1353"), "SNo" : 12, "ObservationDate" : "01/22/2020", "Confirmed" : 5, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Henan" }
{ "_id" : ObjectId("6081070da2d36724e9fc1354"), "SNo" : 13, "ObservationDate" : "01/22/2020", "Confirmed" : 0, "Deaths"
: 0, "Recovered" : 0, "Country" : "Hong Kong", "Last_Update" : "1/22/2020 17:00", "State" : "Hong Kong" }
{ "_id" : ObjectId("6081070da2d36724e9fc1355"), "SNo" : 14, "ObservationDate" : "01/22/2020", "Confirmed" : 444, "Deaths
" : 17, "Recovered" : 28, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc1356"), "SNo" : 15, "ObservationDate" : "01/22/2020", "Confirmed" : 4, "Deaths"
```

Displaying only one data stored in a corona collection.

```
Command Prompt - mongo
> db.corona.findOne()
{
        "_id" : ObjectId("6081070da2d36724e9fc1348"),
        "SNo" : 1,
        "ObservationDate" : "01/22/2020",
        "Confirmed" : 1,
        "Deaths" : 0,
        "Recovered" : 0,
        "Country" : "Mainland China",
        "Last_Update" : "1/22/2020 17:00",
        "State" : "Anhui"
}
>
```

Counting how many data contains in corona collection.

```
Command Prompt - mongo
> db.corona.find().count()
236017
>
```

Displaying unique countries names.

```
Command Prompt - mongo
> db.corona.distinct('Country')
[
        "('St. Martin',)",
        "Afghanistan",
        "Albania",
        "Algeria",
        "Andorra",
        "Angola",
        "Antigua and Barbuda",
        "Argentina",
        "Armenia",
        "Aruba",
        "Australia",
        "Austria",
        "Azerbaijan",
        "Bahamas",
        "Bahamas, The",
        "Bahrain",
        "Bangladesh",
        "Barbados",
        "Belarus",
        "Belgium",
        "Belize",
        "Benin",
        "Bhutan",
        "Bolivia",
        "Bosnia and Herzegovina",
        "Botswana",
        "Brazil",
        "Brunei",
```

Displaying unique countries count.

```
Command Prompt - mongo
> db.corona.distinct('Country').length
226
>
```

Displaying those data which has a country name Nepal.

```
Command Prompt - mongo
> db.corona.find({'Country':'Nepal'}).pretty()
{
        "_id" : ObjectId("6081070da2d36724e9fc1401"),
        "SNo" : 168,
        "ObservationDate" : "01/25/2020",
        "Confirmed" : 1,
        "Deaths" : 0,
        "Recovered" : 0,
        "Country" : "Nepal",
        "Last_Update" : "1/25/20 17:00",
        "State" : ""
}
{
        "_id" : ObjectId("6081070da2d36724e9fc141c"),
        "SNo" : 214,
        "ObservationDate" : "01/26/2020",
        "Confirmed" : 1,
        "Deaths" : 0,
        "Recovered" : 0,
        "Country" : "Nepal",
        "Last_Update" : "1/26/20 16:00",
        "State" : ""
}
{
        "_id" : ObjectId("6081070da2d36724e9fc144a"),
        "SNo" : 260,
        "ObservationDate" : "01/27/2020",
        "Confirmed" : 1,
        "Deaths" : 0,
        "Recovered" : 0,
```

Displaying column data ObservationDeath, Deaths, Recovered, _id which has a country name Nepal.

```
Command Prompt - mongo
> db.corona.find({'Country':'Nepal'},{'ObservationDate':1, 'Deaths':1, 'Recovered':1, _id:1})
{ "_id" : ObjectId("6081070da2d36724e9fc1401"), "ObservationDate" : "01/25/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc141c"), "ObservationDate" : "01/26/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc144a"), "ObservationDate" : "01/27/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc147d"), "ObservationDate" : "01/28/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc14b1"), "ObservationDate" : "01/29/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc14e8"), "ObservationDate" : "01/30/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc152c"), "ObservationDate" : "01/31/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc156c"), "ObservationDate" : "02/01/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc15b0"), "ObservationDate" : "02/02/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc15f5"), "ObservationDate" : "02/03/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc163c"), "ObservationDate" : "02/04/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc1681"), "ObservationDate" : "02/05/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc16c8"), "ObservationDate" : "02/06/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070da2d36724e9fc1710"), "ObservationDate" : "02/07/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070ea2d36724e9fc1758"), "ObservationDate" : "02/08/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070ea2d36724e9fc17aa"), "ObservationDate" : "02/09/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070ea2d36724e9fc17ea"), "ObservationDate" : "02/10/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070ea2d36724e9fc1831"), "ObservationDate" : "02/11/2020", "Deaths" : 0, "Recovered" : 0 }
{ "_id" : ObjectId("6081070ea2d36724e9fc187a"), "ObservationDate" : "02/12/2020", "Deaths" : 0, "Recovered" : 1 }
{ "_id" : ObjectId("6081070ea2d36724e9fc18c4"), "ObservationDate" : "02/13/2020", "Deaths" : 0, "Recovered" : 1 }
Type "it" for more
> it
{ "_id" : ObjectId("6081070ea2d36724e9fc190f"), "ObservationDate" : "02/14/2020", "Deaths" : 0, "Recovered" : 1 }
{ "_id" : ObjectId("6081070ea2d36724e9fc195a"), "ObservationDate" : "02/15/2020", "Deaths" : 0, "Recovered" : 1 }
{ "_id" : ObjectId("6081070ea2d36724e9fc19a5"), "ObservationDate" : "02/16/2020", "Deaths" : 0, "Recovered" : 1 }
{ "_id" : ObjectId("6081070ea2d36724e9fc19f0"), "ObservationDate" : "02/17/2020", "Deaths" : 0, "Recovered" : 1 }
{ "_id" : ObjectId("6081070ea2d36724e9fc1a3b"), "ObservationDate" : "02/18/2020", "Deaths" : 0, "Recovered" : 1 }
{ "_id" : ObjectId("6081070ea2d36724e9fc1a87"), "ObservationDate" : "02/19/2020", "Deaths" : 0, "Recovered" : 1 }
{ "_id" : ObjectId("6081070ea2d36724e9fc1ad3"), "ObservationDate" : "02/20/2020", "Deaths" : 0, "Recovered" : 1 }
```

Displaying data's which has confirmed corona cases greater than 1000.

```
Command Prompt - mongo
> db.corona.find({'Confirmed':{$gt:1000}})
{ "_id" : ObjectId("6081070da2d36724e9fc13f0"), "SNo" : 170, "ObservationDate" : "01/26/2020", "Confirmed" : 1058, "Deat
hs" : 52, "Recovered" : 42, "Country" : "Mainland China", "Last_Update" : "1/26/20 16:00", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc141f"), "SNo" : 217, "ObservationDate" : "01/27/2020", "Confirmed" : 1423, "Deat
hs" : 76, "Recovered" : 45, "Country" : "Mainland China", "Last_Update" : "1/27/20 23:59", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc1452"), "SNo" : 268, "ObservationDate" : "01/28/2020", "Confirmed" : 3554, "Deat
hs" : 125, "Recovered" : 80, "Country" : "Mainland China", "Last_Update" : "1/28/20 23:00", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc1487"), "SNo" : 320, "ObservationDate" : "01/29/2020", "Confirmed" : 3554, "Deat
hs" : 125, "Recovered" : 88, "Country" : "Mainland China", "Last_Update" : "1/29/20 19:30", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc14bc"), "SNo" : 374, "ObservationDate" : "01/30/2020", "Confirmed" : 4903, "Deat
hs" : 162, "Recovered" : 90, "Country" : "Mainland China", "Last_Update" : "1/30/20 16:00", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc14f6"), "SNo" : 432, "ObservationDate" : "01/31/2020", "Confirmed" : 5806, "Deat
hs" : 204, "Recovered" : 141, "Country" : "Mainland China", "Last_Update" : "1/31/2020 23:59", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc1534"), "SNo" : 494, "ObservationDate" : "02/01/2020", "Confirmed" : 7153, "Deat
hs" : 249, "Recovered" : 168, "Country" : "Mainland China", "Last_Update" : "2/1/2020 11:53", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc1577"), "SNo" : 561, "ObservationDate" : "02/02/2020", "Confirmed" : 11177, "Dea
ths" : 350, "Recovered" : 295, "Country" : "Mainland China", "Last_Update" : "2020-02-02T23:43:02", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc15cd"), "SNo" : 628, "ObservationDate" : "02/03/2020", "Confirmed" : 13522, "Dea
ths" : 414, "Recovered" : 386, "Country" : "Mainland China", "Last_Update" : "2020-02-03T23:23:03", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc15fe"), "SNo" : 696, "ObservationDate" : "02/04/2020", "Confirmed" : 16678, "Dea
ths" : 479, "Recovered" : 522, "Country" : "Mainland China", "Last_Update" : "2020-02-04T23:43:01", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc1644"), "SNo" : 766, "ObservationDate" : "02/05/2020", "Confirmed" : 19665, "Dea
ths" : 549, "Recovered" : 633, "Country" : "Mainland China", "Last_Update" : "2020-02-05T23:13:12", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc168b"), "SNo" : 837, "ObservationDate" : "02/06/2020", "Confirmed" : 22112, "Dea
ths" : 618, "Recovered" : 817, "Country" : "Mainland China", "Last_Update" : "2020-02-06T23:23:02", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc16d2"), "SNo" : 908, "ObservationDate" : "02/07/2020", "Confirmed" : 24953, "Dea
ths" : 699, "Recovered" : 1115, "Country" : "Mainland China", "Last_Update" : "2020-02-07T23:43:02", "State" : "Hubei" }

{ "_id" : ObjectId("6081070da2d36724e9fc16d3"), "SNo" : 909, "ObservationDate" : "02/07/2020", "Confirmed" : 1034, "Deat
hs" : 1, "Recovered" : 88, "Country" : "Mainland China", "Last_Update" : "2020-02-07T10:13:06", "State" : "Guangdong" }
```

Displaying data's which has confirmed corona cases less than 1000.

```
Command Prompt - mongo
> db.corona.find({'Confirmed':{$lt:1000}})
{ "_id" : ObjectId("6081070da2d36724e9fc1348"), "SNo" : 1, "ObservationDate" : "01/22/2020", "Confirmed" : 1, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Anhui" }
{ "_id" : ObjectId("6081070da2d36724e9fc1349"), "SNo" : 2, "ObservationDate" : "01/22/2020", "Confirmed" : 14, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Beijing" }
{ "_id" : ObjectId("6081070da2d36724e9fc134a"), "SNo" : 3, "ObservationDate" : "01/22/2020", "Confirmed" : 6, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Chongqing" }
{ "_id" : ObjectId("6081070da2d36724e9fc134b"), "SNo" : 4, "ObservationDate" : "01/22/2020", "Confirmed" : 1, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Fujian" }
{ "_id" : ObjectId("6081070da2d36724e9fc134c"), "SNo" : 5, "ObservationDate" : "01/22/2020", "Confirmed" : 0, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Gansu" }
{ "_id" : ObjectId("6081070da2d36724e9fc134d"), "SNo" : 6, "ObservationDate" : "01/22/2020", "Confirmed" : 26, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Guangdong" }
{ "_id" : ObjectId("6081070da2d36724e9fc134e"), "SNo" : 7, "ObservationDate" : "01/22/2020", "Confirmed" : 2, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Guangxi" }
{ "_id" : ObjectId("6081070da2d36724e9fc134f"), "SNo" : 8, "ObservationDate" : "01/22/2020", "Confirmed" : 1, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Guizhou" }
{ "_id" : ObjectId("6081070da2d36724e9fc1350"), "SNo" : 9, "ObservationDate" : "01/22/2020", "Confirmed" : 4, "Deaths" :
 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Hainan" }
{ "_id" : ObjectId("6081070da2d36724e9fc1351"), "SNo" : 10, "ObservationDate" : "01/22/2020", "Confirmed" : 1, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Hebei" }
{ "_id" : ObjectId("6081070da2d36724e9fc1352"), "SNo" : 11, "ObservationDate" : "01/22/2020", "Confirmed" : 0, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Heilongjiang" }
{ "_id" : ObjectId("6081070da2d36724e9fc1353"), "SNo" : 12, "ObservationDate" : "01/22/2020", "Confirmed" : 5, "Deaths"
: 0, "Recovered" : 0, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Henan" }
{ "_id" : ObjectId("6081070da2d36724e9fc1354"), "SNo" : 13, "ObservationDate" : "01/22/2020", "Confirmed" : 0, "Deaths"
: 0, "Recovered" : 0, "Country" : "Hong Kong", "Last_Update" : "1/22/2020 17:00", "State" : "Hong Kong" }
{ "_id" : ObjectId("6081070da2d36724e9fc1355"), "SNo" : 14, "ObservationDate" : "01/22/2020", "Confirmed" : 444, "Deaths
" : 17, "Recovered" : 28, "Country" : "Mainland China", "Last_Update" : "1/22/2020 17:00", "State" : "Hubei" }
{ "_id" : ObjectId("6081070da2d36724e9fc1356"), "SNo" : 15, "ObservationDate" : "01/22/2020", "Confirmed" : 4, "Deaths"
```

Displaying data's which has confirmed corona cases greater or equal to 30000 and less than or equal to 40000.

```
Command Prompt - mongo
> db.corona.find({'Confirmed':{$lte:40000, $gte:30000}})
{ "_id" : ObjectId("6081070ea2d36724e9fc17ab"), "SNo" : 1124, "ObservationDate" : "02/10/2020", "Confirmed" : 31728, "De
aths" : 974, "Recovered" : 2222, "Country" : "Mainland China", "Last_Update" : "2020-02-10T23:33:02", "State" : "Hubei"
}
{ "_id" : ObjectId("6081070ea2d36724e9fc17f3"), "SNo" : 1196, "ObservationDate" : "02/11/2020", "Confirmed" : 33366, "De
aths" : 1068, "Recovered" : 2639, "Country" : "Mainland China", "Last_Update" : "2020-02-11T23:33:02", "State" : "Hubei"
 }
{ "_id" : ObjectId("6081070ea2d36724e9fc183b"), "SNo" : 1269, "ObservationDate" : "02/12/2020", "Confirmed" : 33366, "De
aths" : 1068, "Recovered" : 2686, "Country" : "Mainland China", "Last_Update" : "2020-02-12T14:13:08", "State" : "Hubei"
 }
{ "_id" : ObjectId("6081070fa2d36724e9fc2b5a"), "SNo" : 6164, "ObservationDate" : "03/17/2020", "Confirmed" : 31506, "De
aths" : 2503, "Recovered" : 2941, "Country" : "Italy", "Last_Update" : "2020-03-17T18:33:02", "State" : "" }
{ "_id" : ObjectId("6081070fa2d36724e9fc2c6f"), "SNo" : 6440, "ObservationDate" : "03/18/2020", "Confirmed" : 35713, "De
aths" : 2978, "Recovered" : 4025, "Country" : "Italy", "Last_Update" : "2020-03-18T17:33:05", "State" : "" }
{ "_id" : ObjectId("6081070fa2d36724e9fc32bf"), "SNo" : 8057, "ObservationDate" : "03/23/2020", "Confirmed" : 35136, "De
aths" : 2311, "Recovered" : 3355, "Country" : "Spain", "Last_Update" : "2020-03-23 23:23:20", "State" : "" }
{ "_id" : ObjectId("6081070fa2d36724e9fc3398"), "SNo" : 8273, "ObservationDate" : "03/24/2020", "Confirmed" : 32986, "De
aths" : 157, "Recovered" : 3243, "Country" : "Germany", "Last_Update" : "2020-03-24 23:41:50", "State" : "" }
{ "_id" : ObjectId("6081070fa2d36724e9fc33eb"), "SNo" : 8357, "ObservationDate" : "03/24/2020", "Confirmed" : 39885, "De
aths" : 2808, "Recovered" : 3794, "Country" : "Spain", "Last_Update" : "2020-03-24 23:41:50", "State" : "" }
{ "_id" : ObjectId("6081070fa2d36724e9fc34c6"), "SNo" : 8575, "ObservationDate" : "03/25/2020", "Confirmed" : 37323, "De
aths" : 206, "Recovered" : 3547, "Country" : "Germany", "Last_Update" : "2020-03-25 23:37:49", "State" : "" }
{ "_id" : ObjectId("6081070fa2d36724e9fc3587"), "SNo" : 8768, "ObservationDate" : "03/25/2020", "Confirmed" : 30841, "De
aths" : 285, "Recovered" : 0, "Country" : "US", "Last_Update" : "2020-03-25 23:37:49", "State" : "New York" }
{ "_id" : ObjectId("6081070fa2d36724e9fc36bc"), "SNo" : 9075, "ObservationDate" : "03/26/2020", "Confirmed" : 37877, "De
aths" : 385, "Recovered" : 0, "Country" : "US", "Last_Update" : "2020-03-26 23:53:24", "State" : "New York" }
{ "_id" : ObjectId("6081070fa2d36724e9fc3728"), "SNo" : 9186, "ObservationDate" : "03/27/2020", "Confirmed" : 32609, "De
aths" : 1994, "Recovered" : 5700, "Country" : "France", "Last_Update" : "2020-03-27 23:27:48", "State" : "" }
{ "_id" : ObjectId("6081070fa2d36724e9fc373b"), "SNo" : 9205, "ObservationDate" : "03/27/2020", "Confirmed" : 32332, "De
aths" : 2378, "Recovered" : 11133, "Country" : "Iran", "Last_Update" : "2020-03-27 23:27:48", "State" : "" }
```

Displaying data's which has confirmed corona cases equal to 90000.

```
Command Prompt - mongo
> db.corona.find({'Confirmed':{$eq:90000}})
{ "_id" : ObjectId("6081071aa2d36724e9fd0ddd"), "SNo" : 64151, "ObservationDate" : "07/14/2020", "Confirmed" : 90000, "D
eaths" : 1571, "Recovered" : 0, "Country" : "US", "Last_Update" : "2020-07-15 04:34:39", "State" : "North Carolina" }
>
```

Displaying data's which has ObservationDate which has month 06.

```
Command Prompt - mongo
> db.corona.find({'ObservationDate':{$regex:/^06/i}})
{ "_id" : ObjectId("60810713a2d36724e9fc9300"), "SNo" : 32698, "ObservationDate" : "06/01/2020", "Confirmed" : 15753, "D
eaths" : 266, "Recovered" : 1428, "Country" : "Afghanistan", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc9301"), "SNo" : 32699, "ObservationDate" : "06/01/2020", "Confirmed" : 1143, "De
aths" : 33, "Recovered" : 877, "Country" : "Albania", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc9302"), "SNo" : 32700, "ObservationDate" : "06/01/2020", "Confirmed" : 9513, "De
aths" : 661, "Recovered" : 5894, "Country" : "Algeria", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc9303"), "SNo" : 32701, "ObservationDate" : "06/01/2020", "Confirmed" : 765, "Dea
ths" : 51, "Recovered" : 698, "Country" : "Andorra", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc9304"), "SNo" : 32702, "ObservationDate" : "06/01/2020", "Confirmed" : 86, "Deat
hs" : 4, "Recovered" : 18, "Country" : "Angola", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc9305"), "SNo" : 32703, "ObservationDate" : "06/01/2020", "Confirmed" : 26, "Deat
hs" : 3, "Recovered" : 19, "Country" : "Antigua and Barbuda", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc9306"), "SNo" : 32704, "ObservationDate" : "06/01/2020", "Confirmed" : 17415, "D
eaths" : 556, "Recovered" : 5521, "Country" : "Argentina", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc9307"), "SNo" : 32705, "ObservationDate" : "06/01/2020", "Confirmed" : 9492, "De
aths" : 139, "Recovered" : 3402, "Country" : "Armenia", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc9308"), "SNo" : 32706, "ObservationDate" : "06/01/2020", "Confirmed" : 16733, "D
eaths" : 668, "Recovered" : 15596, "Country" : "Austria", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc9309"), "SNo" : 32707, "ObservationDate" : "06/01/2020", "Confirmed" : 5662, "De
aths" : 68, "Recovered" : 3508, "Country" : "Azerbaijan", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc930a"), "SNo" : 32708, "ObservationDate" : "06/01/2020", "Confirmed" : 102, "Dea
ths" : 11, "Recovered" : 49, "Country" : "Bahamas", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc930b"), "SNo" : 32709, "ObservationDate" : "06/01/2020", "Confirmed" : 11871, "D
eaths" : 19, "Recovered" : 7070, "Country" : "Bahrain", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc930c"), "SNo" : 32710, "ObservationDate" : "06/01/2020", "Confirmed" : 49534, "D
eaths" : 672, "Recovered" : 10597, "Country" : "Bangladesh", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc930e"), "SNo" : 32711, "ObservationDate" : "06/01/2020", "Confirmed" : 92, "Deat
hs" : 7, "Recovered" : 76, "Country" : "Barbados", "Last_Update" : "2020-06-02 02:33:08", "State" : "" }
{ "_id" : ObjectId("60810713a2d36724e9fc930f"), "SNo" : 32712, "ObservationDate" : "06/01/2020", "Confirmed" : 43403, "D
```

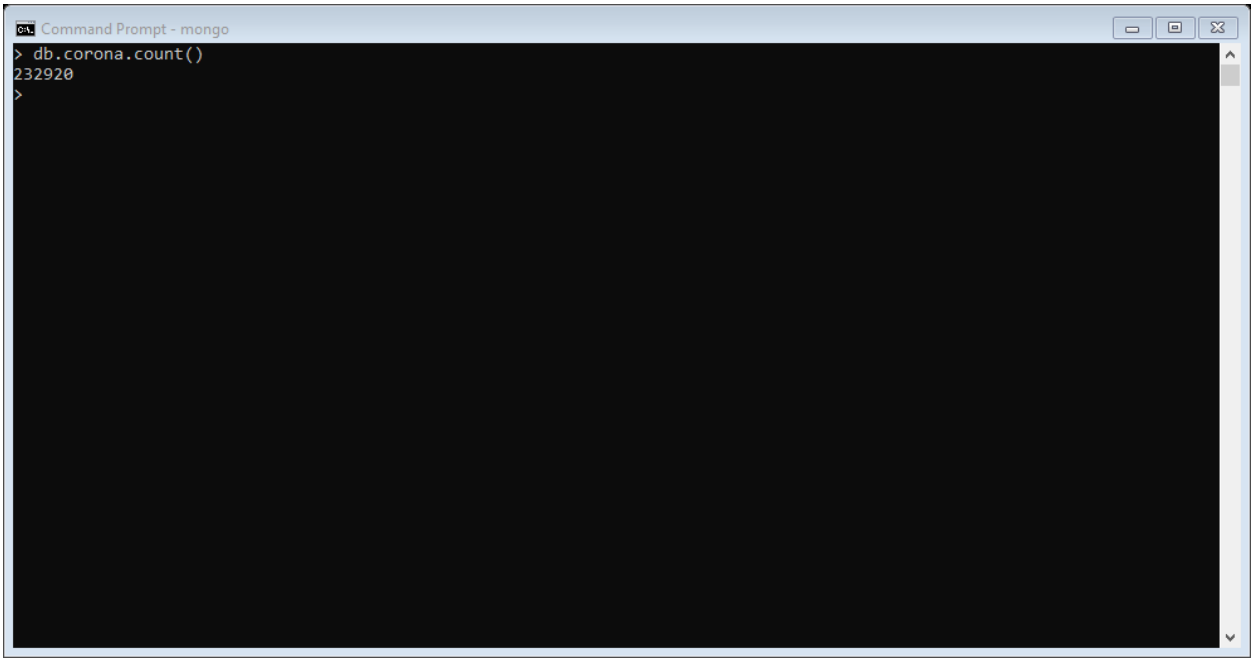
Counting those data's which has country name starts with 'a' or 'A'.

```
Command Prompt - mongo
> db.corona.find({'Country':{$regex:/^a/i}},{'Country':1}).count()
6682
>
```

Deleting data which has confirm corona cases zero.

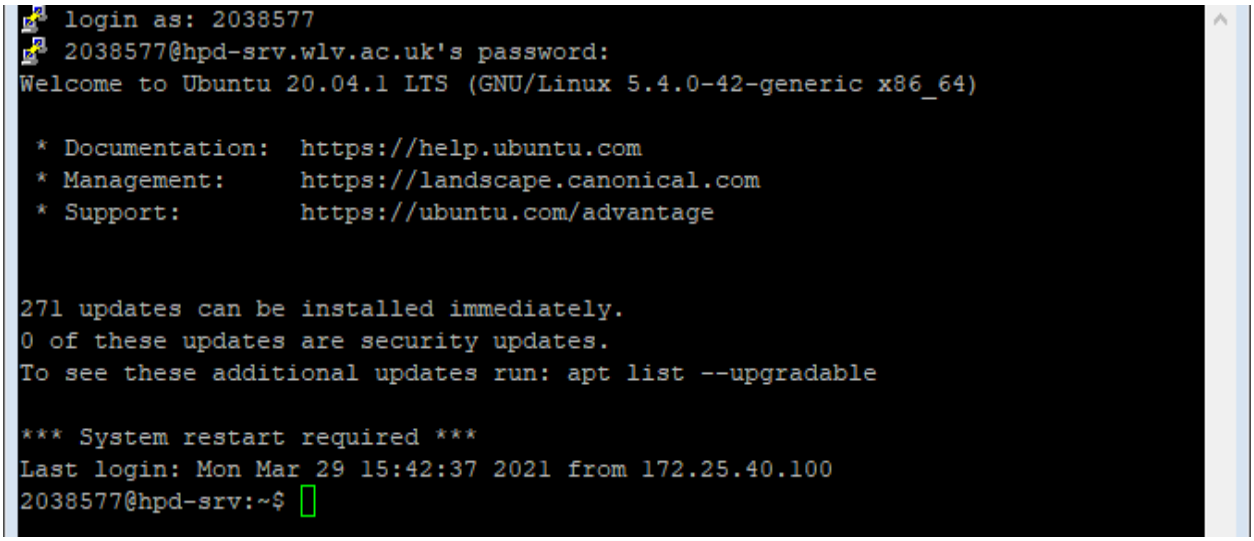
```
Command Prompt - mongo
> db.corona.deleteMany({'Confirmed':0})
{ "acknowledged" : true, "deletedCount" : 3097 }
>
```

Counting data contained in corona collection after removing certain data.



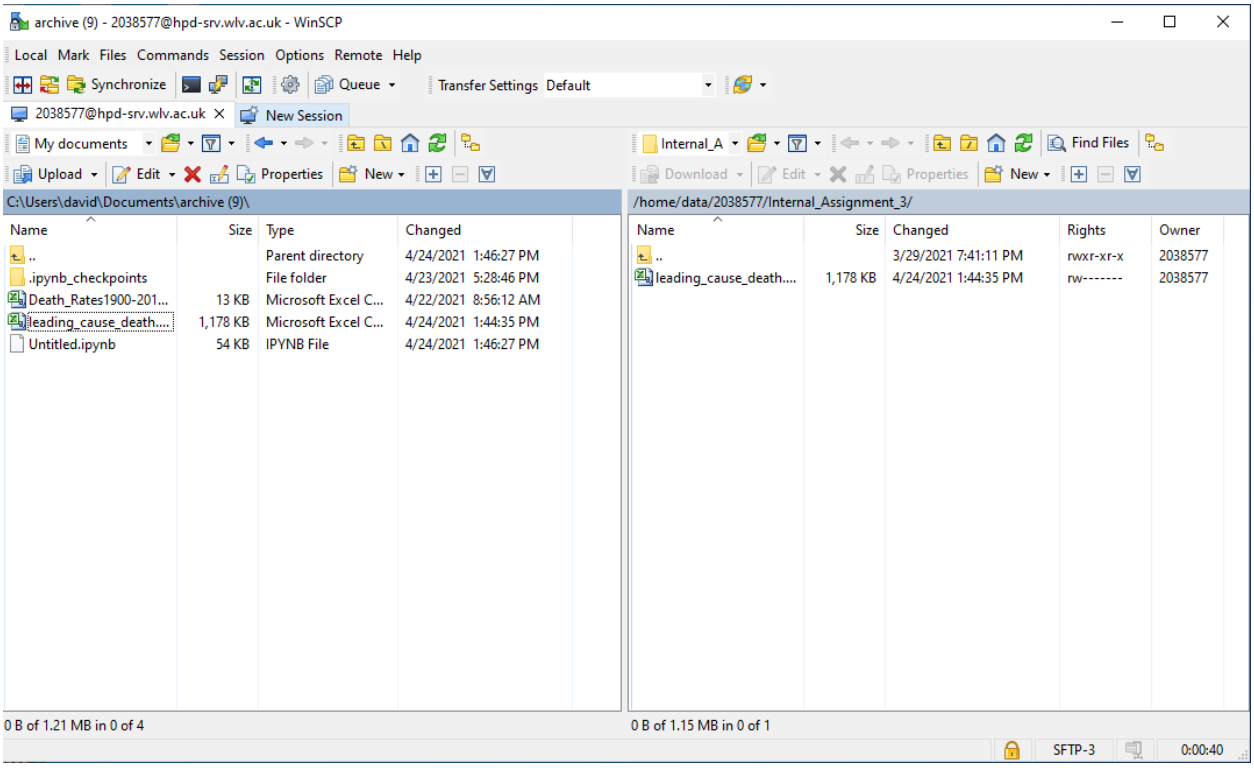
Query for Hadoop

Login using username and password in putty

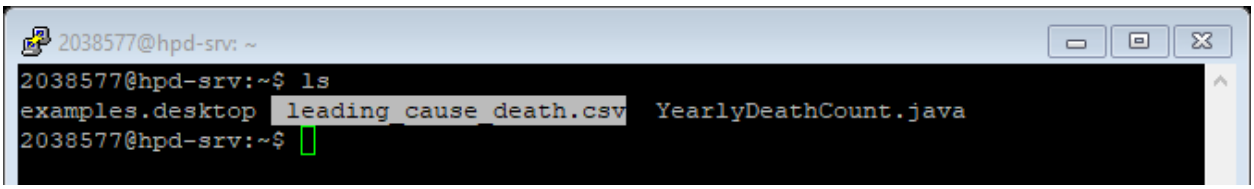


```
login as: 2038577
2038577@hpd-srv.wlv.ac.uk's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-42-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


271 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

*** System restart required ***
Last login: Mon Mar 29 15:42:37 2021 from 172.25.40.100
2038577@hpd-srv:~$
```

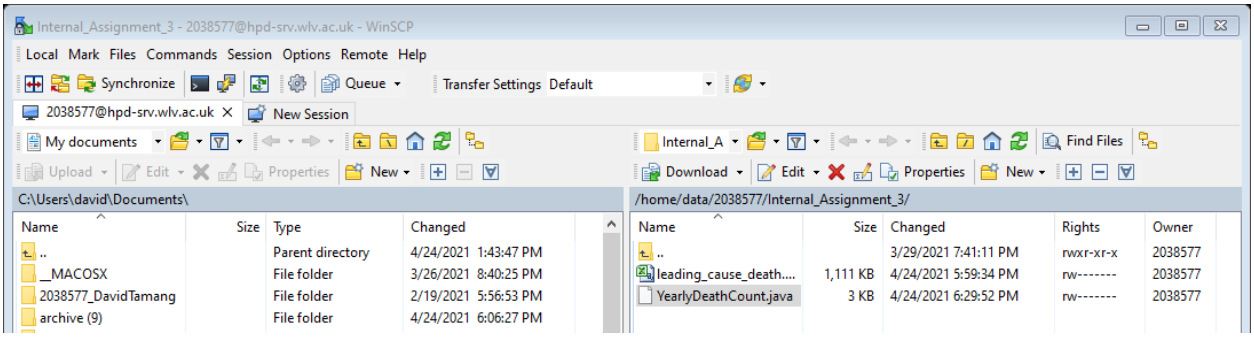
Uploading CSV file in WinSCP:



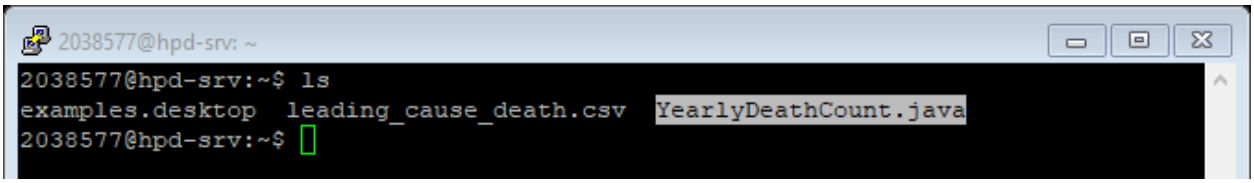
After uploading CSV file in WinSCP:



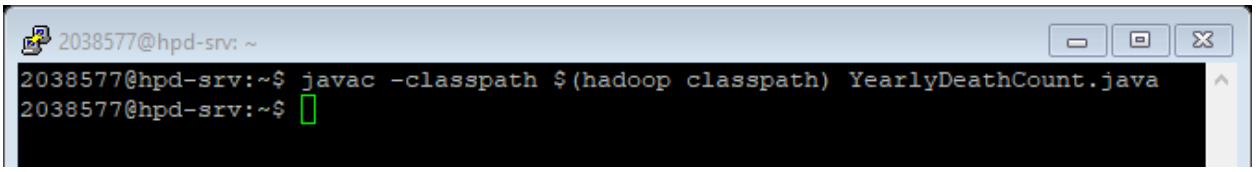
Uploading YearlyDeathCount.java in winSCP:



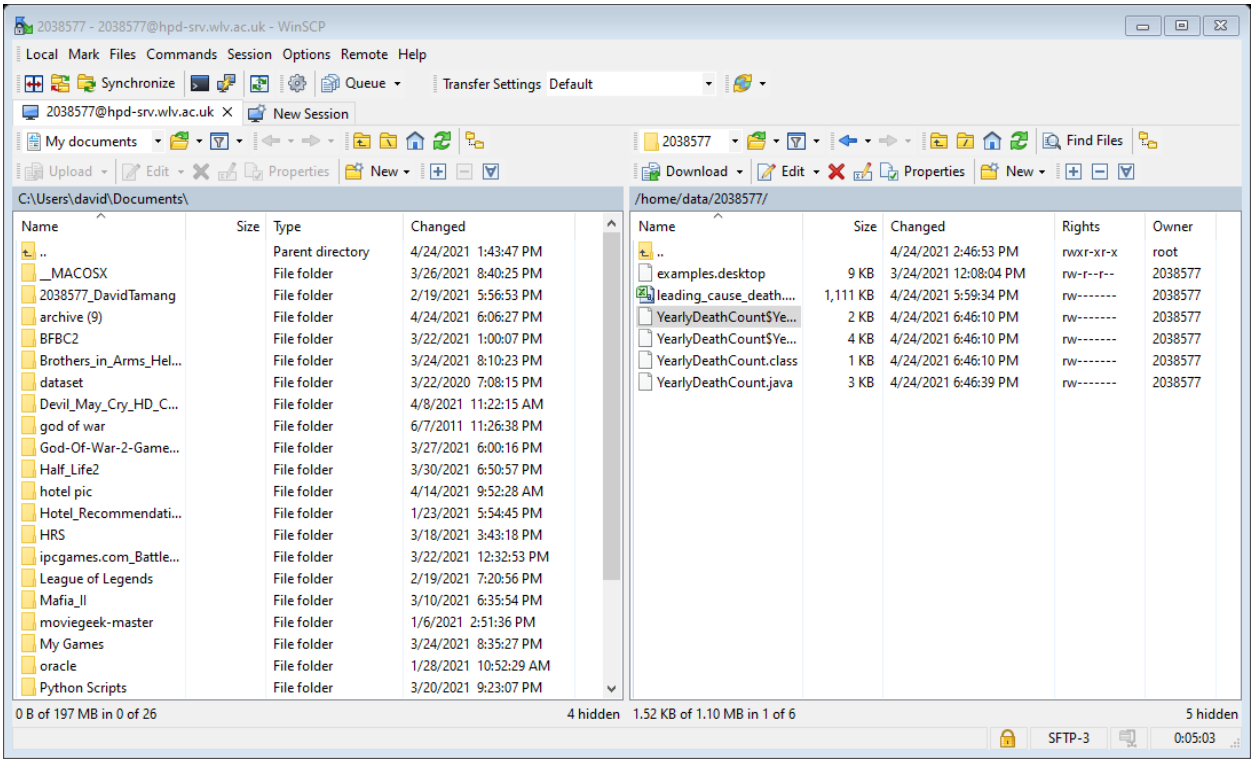
After uploading YearlyDeathCount.java in winSCP:



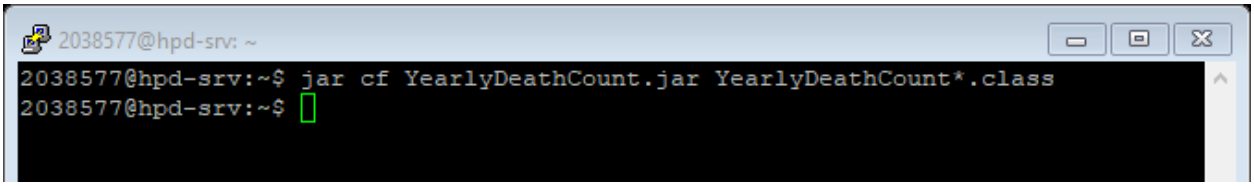
Creating YearlyDeathCount.class file using YearlyDeathCount.java:



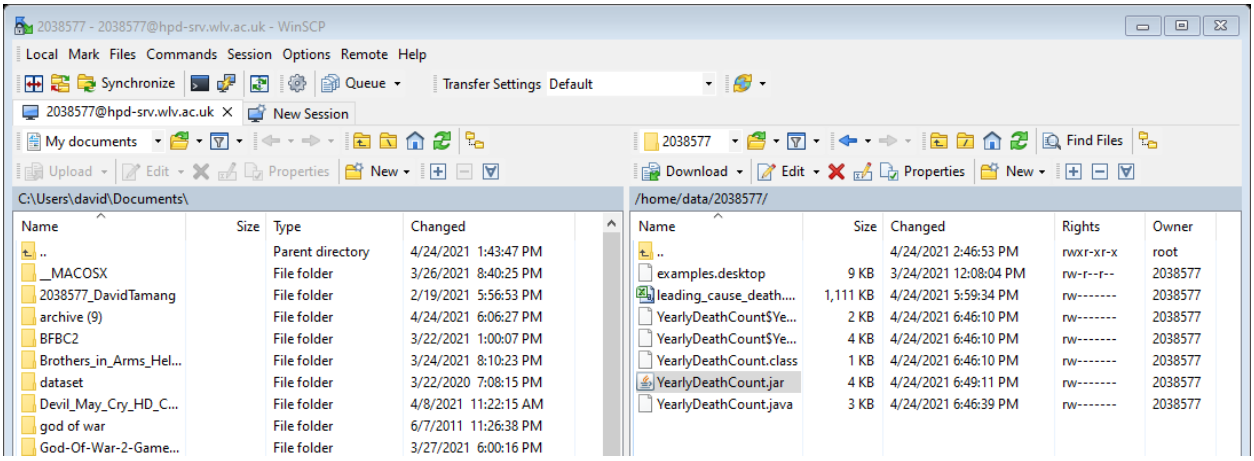
After Creating YearlyDeathCount.class file using YearlyDeathCount.java:



Creating jar file:



After Creating jar file:



Creating new folder in Hadoop:

```
2038577@hpd-srv: ~
2038577@hpd-srv:~$ hdfs dfs -mkdir input
2038577@hpd-srv:~$ hdfs dfs -ls
Found 5 items
drwxr-xr-x   - 2038577 2038577          0 2021-03-29 14:03 david
drwxr-xr-x   - 2038577 2038577          0 2021-03-24 06:25 david_output
drwxr-xr-x   - 2038577 2038577          0 2021-04-24 14:06 input
drwxr-xr-x   - 2038577 2038577          0 2021-03-29 15:11 input_david
drwxr-xr-x   - 2038577 2038577          0 2021-03-29 15:15 output_david
2038577@hpd-srv:~$
```

Putting leading_cause_death.csv file inside of input folder:

```
2038577@hpd-srv: ~
2038577@hpd-srv:~$ hdfs dfs -put leading_cause_death.csv input
2038577@hpd-srv:~$ hdfs dfs -ls input
Found 1 items
-rw-r--r--   1 2038577 2038577    1136930 2021-04-24 14:27 input/leading_cause_d
eath.csv
2038577@hpd-srv:~$
```

Using jar file to perform map-reduce operation and saving output inside of Result folder:

```
2038577@hpd-srv: ~
2038577@hpd-srv:~$ hadoop jar YearlyDeathCount.jar YearlyDeathCount input/leading_cause_death.csv Result
2021-04-24 16:16:39,511 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8050
2021-04-24 16:16:39,906 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed
. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2021-04-24 16:16:39,919 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-y
arn/staging/2038577/.staging/job_1597116737186_0332
2021-04-24 16:16:40,097 INFO input.FileInputFormat: Total input files to process : 1
2021-04-24 16:16:40,140 INFO mapreduce.JobSubmitter: number of splits:1
2021-04-24 16:16:40,252 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1597116737186_0332
2021-04-24 16:16:40,253 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-04-24 16:16:40,420 INFO conf.Configuration: resource-types.xml not found
2021-04-24 16:16:40,420 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-04-24 16:16:40,474 INFO impl.YarnClientImpl: Submitted application application_1597116737186_0332
2021-04-24 16:16:40,507 INFO mapreduce.Job: The url to track the job: http://hpd-srv.unv.wlv.ac.uk:8088/prox
y/application_1597116737186_0332/
2021-04-24 16:16:40,508 INFO mapreduce.Job: Running job: job_1597116737186_0332
2021-04-24 16:16:45,584 INFO mapreduce.Job: Job job_1597116737186_0332 running in uber mode : false
2021-04-24 16:16:45,585 INFO mapreduce.Job:  map 0% reduce 0%
2021-04-24 16:16:50,653 INFO mapreduce.Job:  map 100% reduce 0%
2021-04-24 16:16:55,685 INFO mapreduce.Job:  map 100% reduce 100%
2021-04-24 16:16:55,696 INFO mapreduce.Job: Job job_1597116737186_0332 completed successfully
2021-04-24 16:16:55,801 INFO mapreduce.Job: Counters: 53
        File System Counters
                FILE: Number of bytes read=288308
                FILE: Number of bytes written=1007419
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=1137059
                HDFS: Number of bytes written=18421
                HDFS: Number of read operations=8
```

After performing map-reduce operation:

```
2038577@hpd-srv: ~
2038577@hpd-srv:~$ hdfs dfs -ls Result
Found 2 items
-rw-r--r--   1 2038577 2038577          0 2021-04-24 16:16 Result/_SUCCESS
-rw-r--r--   1 2038577 2038577      18421 2021-04-24 16:16 Result/part-r-00000
2038577@hpd-srv:~$
```

Displaying output of map-reduce operation on csv file:

```
2038577@hpd-srv: ~
2038577@hpd-srv:~$ hdfs dfs -cat Result/part-r-00000
1999,Alabama,64184
1999,Alaska,3847
1999,Arizona,58443
1999,Arkansas,40306
1999,California,331755
1999,Chronic liver disease and cirrhosis,0
1999,Colorado,39568
1999,Connecticut,42906
1999,Delaware,9750
1999,District of Columbia,8468
1999,Florida,235737
1999,Georgia,88813
1999,Hawaii,12107
1999,I13,0
1999,I15)",0
1999,Idaho,14071
1999,Illinois,157715
1999,Indiana,80811
1999,Iowa,41857
1999,Kansas,35485
1999,Kentucky,57389
1999,Louisiana,59452
1999,Maine,18265
1999,Maryland,62964
1999,Massachusetts,82321
1999,Michigan,125832
1999,Minnesota,56685
1999,Mississippi,39931
1999,Missouri,80698
1999,Montana,12032
```

Source Code For Hadoop

```
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

 public class YearlyDeathCount {

 public static class YearlyDeathCountMapper extends Mapper <Object, Text, Text, Text>  {
        public void map(Object key, Text value, Context context)
                throws IOException, InterruptedException
                {

                String record = value.toString();
                String[] parts = record.split(",");
                if (parts.length == 6){
                context.write(new Text(parts[0]+","+parts[3]), new Text(parts[4]));}
                else{
                context.write(new Text(parts[0]+","+parts[3]), new Text("0"));}

                }
}

        public static class YearlyDeathCountReducer extends Reducer <Text, Text, Text, Text> {

                public void reduce(Text key, Iterable<Text> values, Context context)
                        throws IOException, InterruptedException
        {
                        int total = 0;
                        for (Text t : values) {
                                String parts[] = t.toString().split("\t");
                                total += Integer.parseInt(parts[0]);


                        }

                        String str = ""+total;
                        context.write(new Text(key), new Text(str));
        }
        }
```

```java
public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();

        conf.set("mapreduce.output.textoutputformat.separator", ",");

        Job job = Job.getInstance(conf, "NoQuals Count");
        job.setJarByClass(YearlyDeathCount.class);
        job.setMapperClass(YearlyDeathCountMapper.class);
        job.setReducerClass(YearlyDeathCountReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        Path outputPath = new Path(args[1]);
        FileOutputFormat.setOutputPath(job, outputPath);

        // Delete the output directory - true means if path is a directory it does recursive dele
        outputPath.getFileSystem(conf).delete(outputPath, true);
        System.exit(job.waitForCompletion(true) ? 0 : 1);
 }      // main
}
```

Query Apache Spark

Read CSV

```
2038577@hpd-srv: ~
2038577@hpd-srv:~$ pyspark
Python 2.7.18 (default, Mar  8 2021, 13:02:45)
[GCC 9.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 2.4.0
      /_/

Using Python version 2.7.18 (default, Mar  8 2021 13:02:45)
SparkSession available as 'spark'.
>>> df = spark.read.csv('leading_cause_death.csv')
>>>
```

Count how many data present in df:

```
2038577@hpd-srv: ~
>>> df.count()
13260
>>>
```

Showing 10 rows of data:

```
2038577@hpd-srv: ~
>>> df.show(10)
+----+--------------------+--------------------+--------------------+----+-----+
| _c0|                 _c1|                 _c2|                 _c3| _c4|  _c5|
+----+--------------------+--------------------+--------------------+----+-----+
|1999|Accidents (uninte...|Unintentional Inj...|             Alabama|2313|52.17|
|1999|Accidents (uninte...|Unintentional Inj...|              Alaska| 294|55.91|
|1999|Accidents (uninte...|Unintentional Inj...|             Arizona|2214|44.79|
|1999|Accidents (uninte...|Unintentional Inj...|            Arkansas|1287|47.56|
|1999|Accidents (uninte...|Unintentional Inj...|          California|9198|28.71|
|1999|Accidents (uninte...|Unintentional Inj...|            Colorado|1519|38.98|
|1999|Accidents (uninte...|Unintentional Inj...|         Connecticut|1034|29.31|
|1999|Accidents (uninte...|Unintentional Inj...|            Delaware| 267|35.25|
|1999|Accidents (uninte...|Unintentional Inj...|District of Columbia| 161|28.38|
|1999|Accidents (uninte...|Unintentional Inj...|             Florida|5961|35.73|
+----+--------------------+--------------------+--------------------+----+-----+
only showing top 10 rows

>>>
```

Selecting _c4 ,_c5 column and showing its summary:

```
2038577@hpd-srv: ~
>>> df.select('_c4','_c5').describe().show()
+-------+------------------+------------------+
|summary|               _c4|               _c5|
+-------+------------------+------------------+
|  count|             13260|             13260|
|   mean|10131.763197586726| 87.38710085470255|
| stddev| 89137.40981248594|191.80449126902192|
|    min|                 0|              1.29|
|    max|             99931|             999.1|
+-------+------------------+------------------+

>>>
```

Showing distinct values present inside of _c3 column:

```
2038577@hpd-srv: ~
>>> df.select('_c3').distinct().show()
+--------------------+
|                 _c3|
+--------------------+
|                Utah|
|              Hawaii|
|           Minnesota|
|                Ohio|
|            Arkansas|
|              Oregon|
|               Texas|
|        North Dakota|
|        Pennsylvania|
|         Connecticut|
|            Nebraska|
|             Vermont|
|       United States|
|              Nevada|
|          Washington|
|            Illinois|
|            Oklahoma|
|District of Columbia|
|            Delaware|
|              Alaska|
+--------------------+
only showing top 20 rows

>>>
```

Showing only those values which containing state ‘Alabama’:

```
2038577@hpd-srv: ~
>>> df.filter(df['_c3']=='Alabama').show()
+----+--------------------+--------------------+-------+-----+------+
| _c0|                 _c1|                 _c2|    _c3|  _c4|   _c5|
+----+--------------------+--------------------+-------+-----+------+
|1999|Accidents (uninte...|Unintentional Inj...|Alabama| 2313| 52.17|
|1999|          All Causes|          All Causes|Alabama|44806|1009.3|
|1999|Alzheimer's disea...| Alzheimer's disease|Alabama|  772|  17.8|
|1999|Assault (homicide...|            Homicide|Alabama|  438|  9.87|
|1999|Cerebrovascular d...|              Stroke|Alabama| 3148| 71.45|
|1999|Chronic liver dis...|Chronic liver dis...|Alabama|  412|  9.16|
|1999|Chronic lower res...|                CLRD|Alabama| 2179| 48.59|
|1999|Diabetes mellitus...|            Diabetes|Alabama| 1341|    30|
|1999|Diseases of heart...|   Diseases of Heart|Alabama|13419|302.96|
|1999|Essential hyperte...|Essential hyperte...|Alabama|  313|   7.1|
|1999|Influenza and pne...|Influenza and pne...|Alabama| 1228|  28.2|
|1999|Intentional self-...|             Suicide|Alabama|  555| 12.46|
|1999|Malignant neoplas...|              Cancer|Alabama| 9506| 210.9|
|1999|Nephritis, nephro...|      Kidney Disease|Alabama|  979| 22.18|
|1999|Parkinson's disea...| Parkinson's disease|Alabama|  207|  4.72|
|1999|Pneumonitis due t...|Pneumonitis due t...|Alabama|  306|  7.01|
|1999|Septicemia (A40-A41)|          Septicemia|Alabama|  691|  15.6|
|2000|Accidents (uninte...|Unintentional Inj...|Alabama| 2093| 47.02|
|2000|          All Causes|          All Causes|Alabama|45062|1004.8|
|2000|Alzheimer's disea...| Alzheimer's disease|Alabama|  895| 20.39|
+----+--------------------+--------------------+-------+-----+------+
only showing top 20 rows

>>> 
```

Showing only those values which has death greater than 1000:

```
2038577@hpd-srv: ~
>>> df.filter(df['_c4'] > 1000).show()
+----+--------------------+--------------------+-------------+----+-----+
| _c0|                 _c1|                 _c2|          _c3| _c4|  _c5|
+----+--------------------+--------------------+-------------+----+-----+
|1999|Accidents (uninte...|Unintentional Inj...|      Alabama|2313|52.17|
|1999|Accidents (uninte...|Unintentional Inj...|      Arizona|2214|44.79|
|1999|Accidents (uninte...|Unintentional Inj...|     Arkansas|1287|47.56|
|1999|Accidents (uninte...|Unintentional Inj...|   California|9198|28.71|
|1999|Accidents (uninte...|Unintentional Inj...|     Colorado|1519|38.98|
|1999|Accidents (uninte...|Unintentional Inj...|  Connecticut|1034|29.31|
|1999|Accidents (uninte...|Unintentional Inj...|      Florida|5961|35.73|
|1999|Accidents (uninte...|Unintentional Inj...|      Georgia|3078|41.52|
|1999|Accidents (uninte...|Unintentional Inj...|     Illinois|4125|33.69|
|1999|Accidents (uninte...|Unintentional Inj...|      Indiana|2309|38.43|
|1999|Accidents (uninte...|Unintentional Inj...|         Iowa|1123|35.24|
|1999|Accidents (uninte...|Unintentional Inj...|       Kansas|1126|40.68|
|1999|Accidents (uninte...|Unintentional Inj...|     Kentucky|1730|43.29|
|1999|Accidents (uninte...|Unintentional Inj...|    Louisiana|1940|44.66|
|1999|Accidents (uninte...|Unintentional Inj...|     Maryland|1296|25.93|
|1999|Accidents (uninte...|Unintentional Inj...|Massachusetts|1303|19.61|
|1999|Accidents (uninte...|Unintentional Inj...|     Michigan|3188|32.75|
|1999|Accidents (uninte...|Unintentional Inj...|    Minnesota|1772|35.75|
|1999|Accidents (uninte...|Unintentional Inj...|  Mississippi|1642|58.85|
|1999|Accidents (uninte...|Unintentional Inj...|     Missouri|2465|43.22|
+----+--------------------+--------------------+-------------+----+-----+
only showing top 20 rows

>>> 
```

References

chicco, D. & Jurman, G., 2020. Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. *BMC Medical Informatics and Decision Making,* 20(1).

DataFlair, 2019. *DataFlair.* [Online] Available at: https://data-flair.training/blogs/advantages-of-mongodb/#:~:text=A%20great%20advantage%20of%20MongoDB%20is%20that%20it,data%2C%20you%20can%20distribute%20it%20to%20several%20machines. [Accessed 6 4 20221].

DataFlair, 2019. *DataFlair.* [Online] Available at: https://data-flair.training/blogs/advantages-of-mongodb/ [Accessed 6 4 2021].

Educba, 2020. *Educba.* [Online] Available at: https://www.educba.com/what-is-oracle/ [Accessed 22 04 2021].

Guru99, 2021. *Guru99.* [Online] Available at: https://www.guru99.com/what-is-mongodb.html [Accessed 22 04 2021].

Human Development Reports, 2020. *UNITED NATIONS DEVELOPMENT PROGRAMME.* [Online] Available at: http://hdr.undp.org/en/indicators/137506 [Accessed 21 4 2021].

Juneja, A. & Das, N. N., 2019. Big Data Quality Framework: Pre-Processing Data in Weather Monitoring Application. *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon).*

Malavika, 2019. *Medium.* [Online] Available at: https://medium.com/@shireesha.tibacademy/advantages-disadvantages-of-oracle-sql-1cf77e7022d1 [Accessed 5 4 2021].

Singh, H., 2020. *The NineHertz.* [Online] Available at: https://theninehertz.com/blog/advantages-of-using-oracle-database [Accessed 6 4 2021].

Stevepaul, 2015. *Mindsmapped.* [Online] Available at: https://www.mindsmapped.com/hadoop-advantages-and-disadvantages/ [Accessed 6 4 2021].

WHO, 2018. *World Health Organization.* [Online] Available at: https://www.who.int/health-topics/suicide#tab=tab_1 [Accessed 21 04 2021].

World Bank, 2018. *DataBank World Development Indicators.* [Online] Available at: http://databank.worldbank.org/data/source/world-development-indicators# [Accessed 21 04 2021].