

Big Data Project

International Population & Health Metrics Analysis Using Hive & Spark

Team Members :

- PES2201800022 – Achyuth K Kowshik
- PES2201800116 – Aniketh D Urs
- PES2201800656 – Manoj Mahesh Patil

INDEX :

1. Content
2. Uploading the dataset into HDFS .
3. Analysis using HIVE.
 - Screenshot of the execution
 - Code
 - Output / the result
 - The inference that we make out of it
4. Analysis using SPARK .

Content :

The International Dataset provides estimates of country populations since 1950 and projections through 2050 .Specifically , the data set includes midyear population figures broken down by age and gender assignment at birth.

The csv files are :

- Age_specific_fertility_rates.csv
- Birth_death_growth_rates.csv
- Country_names_area.csv

- Midyear_population
- Midyear_population_5yr_age_sex.csv
- Midyear_population_age_country_code.csv
- Midyear_population_age_sex.csv
- Mortality_life_expectancy.csv

First Step in our Project is, we upload or Dataset into HDFS :

```

bdhadoop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal ▾ Dec 1 21:49
manoj@manoj-VirtualBox: ~ /hadoop
nanoj@manoj-VirtualBox: ~ /hadoop$ hdfs dfs -mkdir /project1
2020-12-01 21:44:02.937 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
nanoj@manoj-VirtualBox: ~ /hadoop$ hdfs dfs -ls /project1
Found 1 items:
-rw-r--r-- 1 manoj supergroup 1095587 2020-12-01 21:44 /project1/age_specific_fertility_rates.csv
2020-12-01 21:44:52.622 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
nanoj@manoj-VirtualBox: ~ /hadoop$ hdfs dfs -ls /project2
Found 1 items:
-rw-r--r-- 1 manoj supergroup 712904 2020-12-01 21:44 /project2/birth_death_growth_rates.csv
2020-12-01 21:45:34.294 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
nanoj@manoj-VirtualBox: ~ /hadoop$ hdfs dfs -ls /project3
Found 1 items:
-rw-r--r-- 1 manoj supergroup 4600 2020-12-01 21:45 /project3/country_names_area.csv
2020-12-01 21:46:20.597 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
nanoj@manoj-VirtualBox: ~ /hadoop$ hdfs dfs -ls /project4
Found 1 items:
-rw-r--r-- 1 manoj supergroup 620109 2020-12-01 21:46 /project4/midyear_population.csv
2020-12-01 21:47:07.474 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
nanoj@manoj-VirtualBox: ~ /hadoop$ hdfs dfs -ls /project5
Found 1 items:
-rw-r--r-- 1 manoj supergroup 15723103 2020-12-01 21:47 /projects5/midyear_population_5yr_age_sex.csv
2020-12-01 21:47:52.390 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:02.409 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:07.403 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:12.833 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:17.021 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:22.743 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:26.529 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:31.128 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:35.648 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:40.707 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:45.529 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:48:52.001 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2020-12-01 21:49:03.101 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
nanoj@manoj-VirtualBox: ~ /hadoop$ hdfs dfs -ls /project6

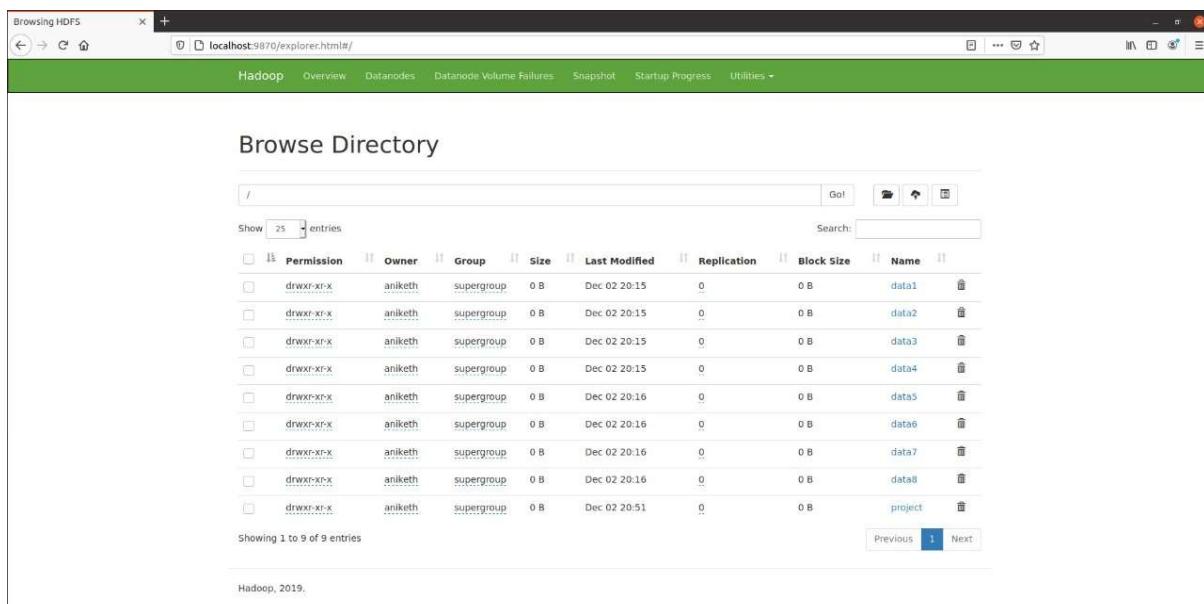
```

Code :

- hdfs dfs -mkdir /project1
- hdfs dfs -mkdir /project2
- hdfs dfs -mkdir /project3
- hdfs dfs -mkdir /project4
- hdfs dfs -mkdir /project5
- hdfs dfs -mkdir /project6
- hdfs dfs -mkdir /project7
- hdfs dfs -mkdir /project8
- hdfs dfs -put
/home/manoj/bigdataproject/age_specific_fertility_rates.csv
/project1

- hdfs dfs -put
`/home/manoj/bigdataproject/birth_death_growth_rate.csv`
`/project2`
- hdfs dfs -put
`/home/manoj/bigdataproject/country_name_area.csv` /project3
- hdfs dfs -put
`/home/manoj/bigdataproject/midyear_population.csv` /project4
- hdfs dfs -put
`/home/manoj/bigdataproject/midyear_population_5yr_age_sex.csv` /project5
- hdfs dfs -put
`/home/manoj/bigdataproject/midyear_population_age_country_code.csv` /project6
- hdfs dfs -put
`/home/manoj/bigdataproject/midyear_population_age_sex.csv` /project7
- hdfs dfs -put
`/home/manoj/bigdataproject/mortality_life_expectancy.csv`
`/project8`

Screenshot of the csv files being uploaded into the HDFS system:



The screenshot shows the HDFS Browser interface at `localhost:9870/explorehtml/`. The main title bar says "Browsing HDFS". The top navigation bar includes links for "Hadoop", "Overview", "Datanodes", "Datanode Volume Failures", "Snapshot", "Startup Progress", and "Utilities". Below the navigation bar is a search bar and some icons.

The main content area is titled "Browse Directory" and shows a table of files in the "/data" directory. The table has columns for "Permission", "Owner", "Group", "Size", "Last Modified", "Replication", "Block Size", and "Name". There are 9 entries listed, all created by "aniketh" and belonging to "supergroup". The files are named "data1" through "data8" and "project". All files have a size of 0 B and were modified on Dec 02 20:15 or 20:16. The "Replication" column shows values of 0 or 1. The "Block Size" column shows 0 B. The "Name" column shows the file names.

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	aniketh	supergroup	0 B	Dec 02 20:15	0	0 B	data1
drwxr-xr-x	aniketh	supergroup	0 B	Dec 02 20:15	0	0 B	data2
drwxr-xr-x	aniketh	supergroup	0 B	Dec 02 20:15	0	0 B	data3
drwxr-xr-x	aniketh	supergroup	0 B	Dec 02 20:15	0	0 B	data4
drwxr-xr-x	aniketh	supergroup	0 B	Dec 02 20:16	0	0 B	data5
drwxr-xr-x	aniketh	supergroup	0 B	Dec 02 20:16	0	0 B	data6
drwxr-xr-x	aniketh	supergroup	0 B	Dec 02 20:16	0	0 B	data7
drwxr-xr-x	aniketh	supergroup	0 B	Dec 02 20:16	0	0 B	data8
drwxr-xr-x	aniketh	supergroup	0 B	Dec 02 20:51	0	0 B	project

At the bottom of the page, it says "Showing 1 to 9 of 9 entries" and has "Previous" and "Next" buttons. The footer says "Hadoop, 2019."

Analysis:

1] Dataset : age_specific_fertility_rates.csv :

```
bin/hadoop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Dec 2 07:17
manoj@manoj-VirtualBox: ~/hive
hive> select avg(fertility_rate_15_19) from asfr ;
Query ID = manoj_2028120207160_0d7fd9bd-b5ca-424a-a4bb-8bedde99fb2
Total Jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running In-process (local Hadoop)
2028-12-02 07:16:52,563 Stage-1 map = 100%, reduce = 100%
Ended Job = job_local14967362_0007
Mapreduce Jobs Launched:
Stage-Stage-1: HDFS Read: 26302280 HDFS Write: 0 SUCCESS
Total Mapreduce CPU Time Spent: 0 msec
45.5045672724352
Time taken: 1.836 seconds, Fetched: 1 row(s)
hive> 
```

```
bin/hadoop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Dec 2 07:19
manoj@manoj-VirtualBox: ~/hive
hive> select avg(fertility_rate_20_24) from asfr ;
Query ID = manoj_20281202071929_a7755b17-cf17-4bb6-93ed-e19701bb2a6b
Total Jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running In-process (local Hadoop)
2028-12-02 07:17:00,563 Stage-1 map = 100%, reduce = 100%
Ended Job = job_local1493950_0010
Mapreduce Jobs Launched:
Stage-Stage-1: HDFS Read: 32875082 HDFS Write: 0 SUCCESS
Total Mapreduce CPU Time Spent: 0 msec
0
122.77988574076527
Time taken: 2.752 seconds, Fetched: 1 row(s)
hive> 
```

```
bin/hadoop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Dec 2 07:19
manoj@manoj-VirtualBox: ~/hive
hive> select avg(fertility_rate_25_29) from asfr ;
Query ID = manoj_2028120207160_474fb0d0-3ref-44b8-388d3cb973f4
Total Jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running In-process (local Hadoop)
2028-12-02 07:17:15,563 Stage-1 map = 100%, reduce = 100%
Ended Job = job_local1496542751_0009
Mapreduce Jobs Launched:
Stage-Stage-1: HDFS Read: 36684628 HDFS Write: 0 SUCCESS
Total Mapreduce CPU Time Spent: 0 msec
193.12491725542328
Time taken: 1.809 seconds, Fetched: 1 row(s)
hive> 
```

```
File: [Running] - Oracle VM VirtualBox  
Machine View Input Devices Help  
Activities Terminal Dec 2 07:23  
manoj@manojVirtualBox: ~$hive  
hive> select avg(fertility_rate_30_30) from asfr;  
Total jobs = 1  
Launching Job 1 out of 1  
Number of reduce tasks determined at compile time: 1  
In order to change the average load for a reducer (in bytes):  
  set hive.reduce.input.buffer.size=  
In order to limit the maximum number of reducers:  
  set hive.mapred.reduce.tasks=  
In order to set a constant number of reducers:  
  job running in-process (local Hadoop)  
2020-12-02 07:23:37,325 Stage-Stage: map = 100%, reduce = 100%  
  Added Job = job_local1240298672_0011  
Mapreduce Jobs Launched:  
  Total Mapreduce CPU Time spent: 0 msec  
  Total Mapreduce IO Time spent: 0 msec  
  Total Mapreduce CPU Time spent: 0 msec  
Time taken: 1.545 seconds, Fetched: 1 row(s)  
hive>
```

```
File: [Running] - Oracle VM VirtualBox  
Machine View Input Devices Help  
Activities Terminal Dec 2 07:24  
manoj@manojVirtualBox: ~$hive  
hive> select avg(fertility_rate_35_35) from asfr;  
Total jobs = 1  
Launching Job 1 out of 1  
Number of reduce tasks determined at compile time: 1  
In order to change the average load for a reducer (in bytes):  
  set hive.reduce.input.buffer.size=  
In order to limit the maximum number of reducers:  
  set hive.mapred.reduce.tasks=  
In order to set a constant number of reducers:  
  job running in-process (local Hadoop)  
2020-12-02 07:24:01,325 Stage-Stage: map = 100%, reduce = 100%  
  Added Job = job_local1240298683_0011  
Mapreduce Jobs Launched:  
  Total Mapreduce CPU Time spent: 0 msec  
  Total Mapreduce IO Time spent: 0 msec  
  Total Mapreduce CPU Time spent: 0 msec  
Time taken: 1.798 seconds, Fetched: 1 row(s)  
hive>
```

```
File: [Running] - Oracle VM VirtualBox  
Machine View Input Devices Help  
Activities Terminal Dec 2 07:24  
manoj@manojVirtualBox: ~$hive  
hive> select avg(fertility_rate_40_40) from asfr;  
Total jobs = 1  
Launching Job 1 out of 1  
Number of reduce tasks determined at compile time: 1  
In order to change the average load for a reducer (in bytes):  
  set hive.reduce.input.buffer.size=  
In order to limit the maximum number of reducers:  
  set hive.mapred.reduce.tasks=  
In order to set a constant number of reducers:  
  job running in-process (local Hadoop)  
2020-12-02 07:24:11,325 Stage-Stage: map = 100%, reduce = 100%  
  Added Job = job_local1240298693_0011  
Mapreduce Jobs Launched:  
  Total Mapreduce CPU Time spent: 0 msec  
  Total Mapreduce IO Time spent: 0 msec  
  Total Mapreduce CPU Time spent: 0 msec  
Time taken: 1.753 seconds, Fetched: 1 row(s)  
hive>
```

```
File: [Running] - Oracle VM VirtualBox  
Machine View Input Devices Help  
Activities Terminal Dec 2 07:22  
manoj@manojVirtualBox: ~$hive  
hive> select avg(fertility_rate_45_45) from asfr;  
Total jobs = 1  
Launching Job 1 out of 1  
Number of reduce tasks determined at compile time: 1  
In order to change the average load for a reducer (in bytes):  
  set hive.reduce.input.buffer.size=  
In order to limit the maximum number of reducers:  
  set hive.mapred.reduce.tasks=  
In order to set a constant number of reducers:  
  job running in-process (local Hadoop)  
2020-12-02 07:22:11,325 Stage-Stage: map = 100%, reduce = 100%  
  Added Job = job_local1240298695_0011  
Mapreduce Jobs Launched:  
  Total Mapreduce CPU Time spent: 0 msec  
  Total Mapreduce IO Time spent: 0 msec  
  Total Mapreduce CPU Time spent: 0 msec  
Time taken: 2.050 seconds, Fetched: 1 row(s)  
hive>
```

Code :

- select avg(fertility_rate_15_19) from asfr ;
- select avg(fertility_rate_20_24) from asfr ;
- select avg(fertility_rate_25_29) from asfr ;
- select avg(fertility_rate_30_34) from asfr ;
- select avg(fertility_rate_35_39) from asfr ;
- select avg(fertility_rate_40_44) from asfr ;
- select avg(fertility_rate_45_49) from asfr ;

The results come out to be :

fertility_rate_15_19 = 45.35

fertility_rate_20_24 = 122.77

fertility_rate_25_29 = 153.12

fertility_rate_30_34 = 127.087

fertility_rate_35_39 = 73.45

fertility_rate_40_44 = 31.28

fertility_rate_45_49 = 8.69

Conclusion : From the above results we can conclude that the women between **the age group 25-29** have the **highest fertility rate**.

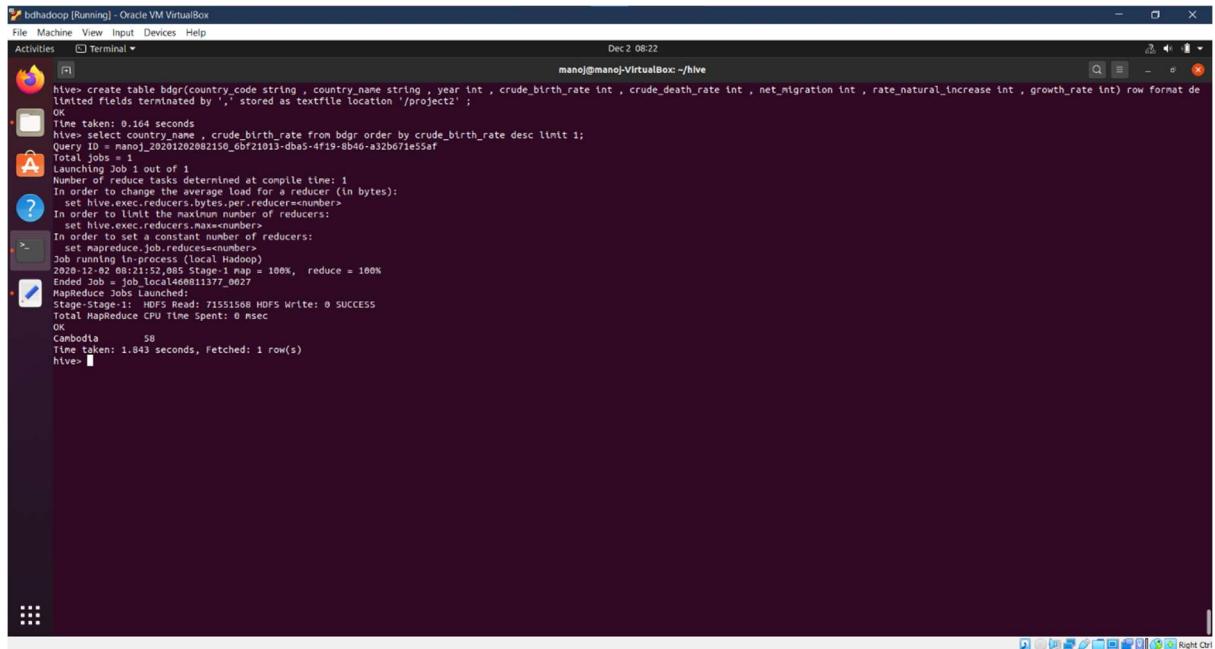
-x-

2] DATASET : birth_death_growth_rates.csv :

Crude Birth Rate : Is the number of births per 1000 divided by the length of the period in years.

Crude Death Rate : Is the number of deaths per 1000 divided by the length of the period in years.

Screenshot of execution :



```
bdhdadoop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal manoj@manoj-VirtualBox: ~/hive
Dec 2 08:22
hive> create table bdgr(country_code string , country_name string , year int , crude_birth_rate int , crude_death_rate int , net_migration int , rate_natural_increase int , growth_rate int) row format de
limited fields terminated by ',' stored as textfile location '/project2';
OK
Time taken: 0.104 seconds
hive> select country_name , crude_birth_rate from bdgr order by crude_birth_rate desc limit 1;
Query ID = manoj_20201202082150_0b721013-0ba5-4f19-0b46-a32b671e55af
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Job running In process (local Hadoop)
2020-12-02 08:21:52.085 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local1468011377_0027
MapReduce Jobs Launched:
  Stage-Stage-1: HDFS Read: 71551568 HDFS Write: 0 SUCCESS
  Total MapReduce CPU Time Spent: 0 msec
OK
Cambodia      58
Time taken: 1.843 seconds, Fetched: 1 row(s)
hive>
```

Code :

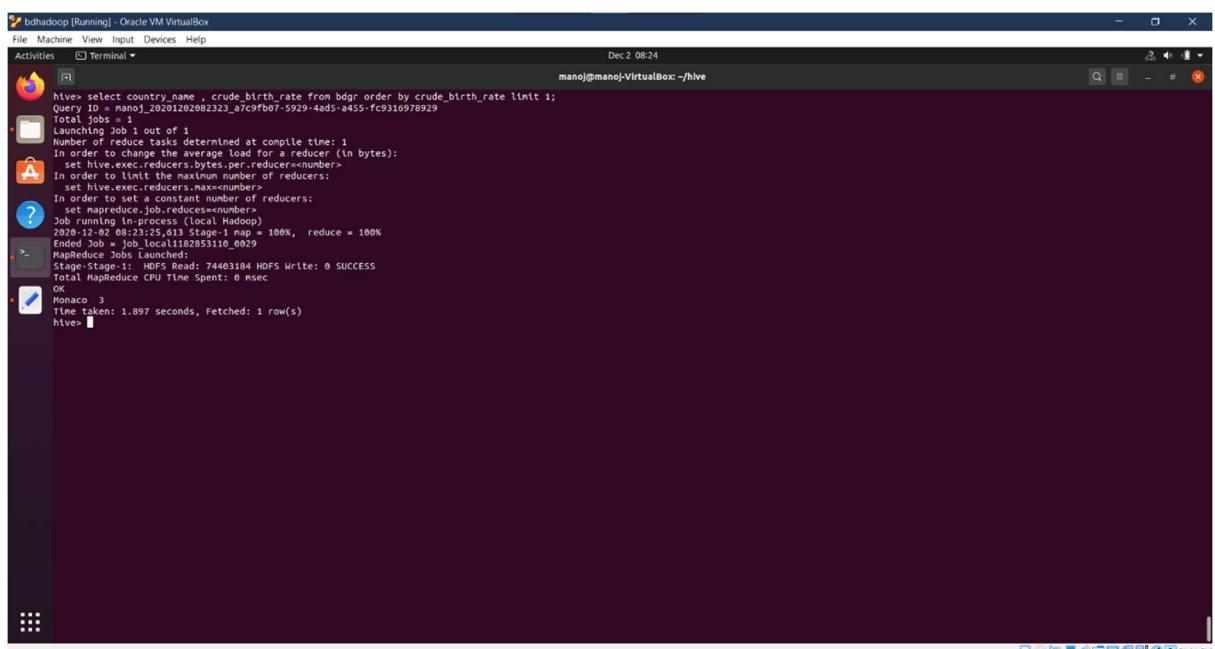
- Select country_name,crude_birth_rate from bdgr order by crude_birth_rate desc limit 1 ;

Results :

Cambodia 58

Conclusion : Cambodia has the highest crude birth rate .

-X-



```
bdhdadoop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal manoj@manoj-VirtualBox: ~/hive
Dec 2 08:24
hive> select country_name , crude_birth_rate from bdgr order by crude_birth_rate limit 1;
Query ID = manoj_20201202082323_a7c9fb07-5929-4add-a455-fc9316978929
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Set mapreduce.job.reduces=<number>
Job running In process (Local Hadoop)
2020-12-02 08:23:25.613 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local18285310_0029
MapReduce Jobs Launched:
  Stage-Stage-1: HDFS Read: 74403184 HDFS Write: 0 SUCCESS
  Total MapReduce CPU Time Spent: 0 msec
OK
Monaco      3
Time taken: 1.897 seconds, Fetched: 1 row(s)
hive>
```

Code :

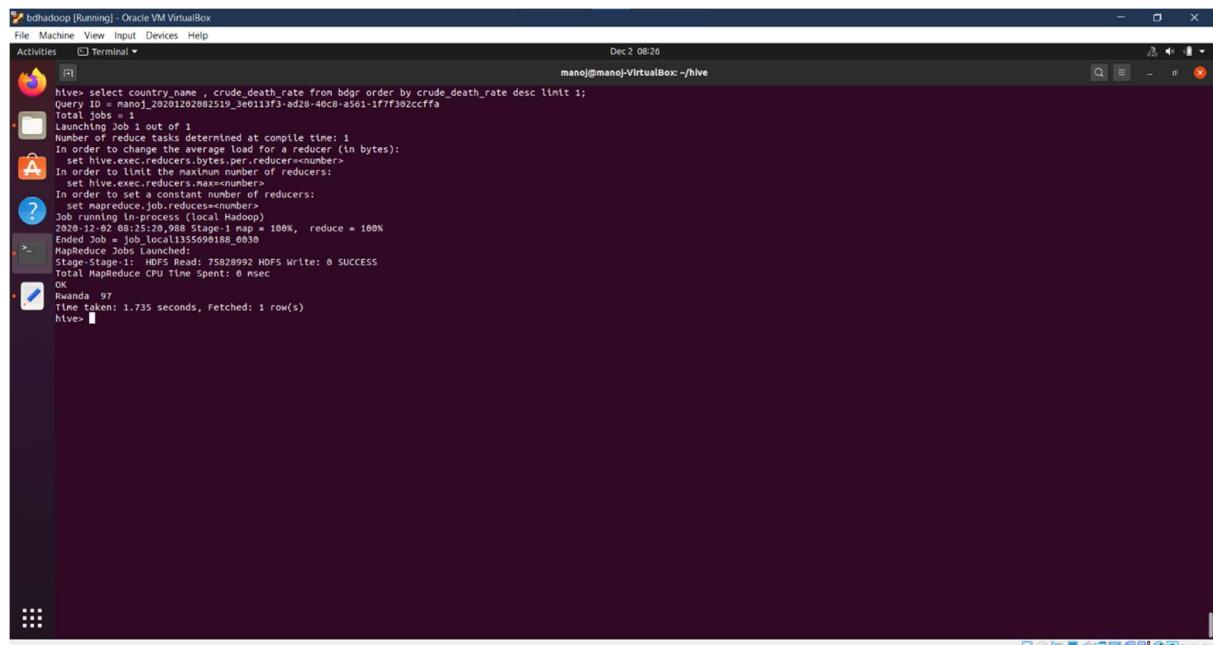
- Select country_name,crude_birth_rate from bdgr order by crude_birth_rate limit 1 ;

Results :

Monaco 3

Conclusion: Monaco has the lowest crude birth rate.

-X-



```
hive> select country_name , crude_death_rate from bdgr order by crude_death_rate desc limit 1;
Query ID : manoj_20201202082519_3e0113f3-ad28-46ce-a561-1ff7f302ccff
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducer.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2020-12-02 08:26:48 Stage-1: map = 100%,  reduce = 100%
Ended Job = job_local135509188_0030
MapReduce Jobs Launched:
Stage-Stage-1:   HDFS Read: 75828992 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Rwanda 97
Time taken: 1.735 seconds, Fetched: 1 row(s)
hive>
```

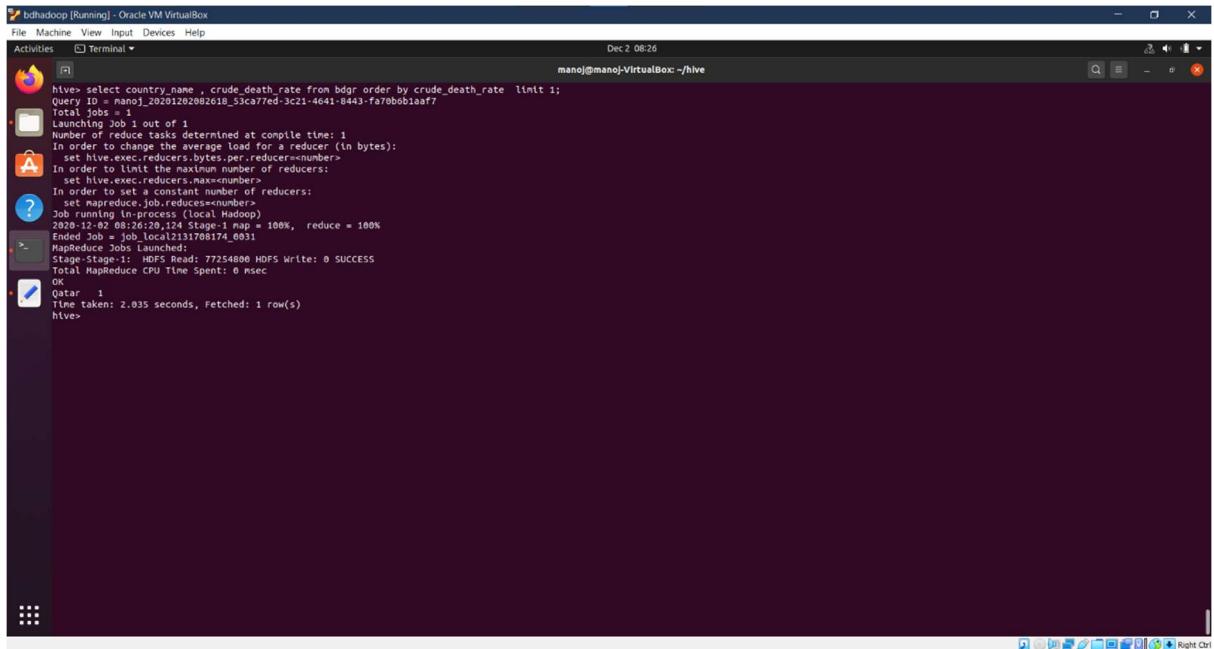
Code:

- Select country_name,crude_death_rate from bdgr order by crude_death_rate desc limit 1 ;

Results:

Rwanda 97

Conclusion: Rwanda has the highest crude death rate .



```
bdhadoop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal manoj@manoj-VirtualBox: ~/hive
hive> select country_name , crude_death_rate from bdgr order by crude_death_rate limit 1;
Query ID = manoj_20201202082618_53ca77ed-3c21-4641-8443-fa70beb1aaaf7
Total Jobs: 1
Launched Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2020-12-02 08:26:20,124 Stage-1 map = 100%, reduce = 100%
Ended Job 1 out of 1 (ID: 1768174, 003)
MapReduce Jobs Launched:
  Stage-Stage-1: HDFS Read: 77254800 HDFS Write: 0 SUCCESS
  Total MapReduce CPU Time Spent: 0 msec
OK
Qatar 1
Time taken: 2.035 seconds, Fetched: 1 row(s)
hive>
```

Code:

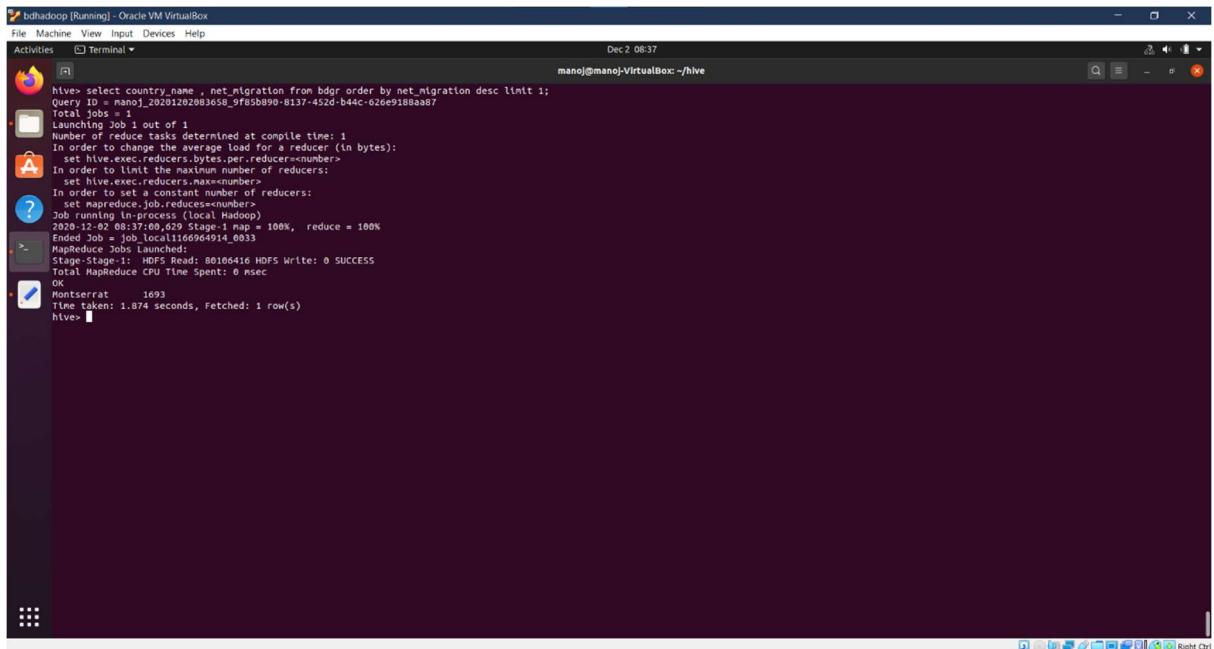
- Select country_name,crude_death_rate from bdgr order by crude_death_rate limit 1 ;

Results:

Qatar 1

Conclusion: Qatar has the highest crude death rate .

-X-



```
bdhadoop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal manoj@manoj-VirtualBox: ~/hive
hive> select country_name , net_migration from bdgr order by net_migration desc limit 1;
Query ID = manoj_20201202083658_9f85b890-8137-452d-b44c-626e9188aa87
Total Jobs: 1
Launched Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2020-12-02 08:37:10,679 Stage-1 map = 100%, reduce = 100%
Ended Job 1 out of 1 (ID: 176964914, 003)
MapReduce Jobs Launched:
  Stage-Stage-1: HDFS Read: 80166416 HDFS Write: 0 SUCCESS
  Total MapReduce CPU Time Spent: 0 msec
OK
Montserrat 1693
Time taken: 1.874 seconds, Fetched: 1 row(s)
hive>
```

Code:

- Select country_name,net_migration from bdgr order by net_migration desc limit 1 ;

Results:

Montserrat 1693.

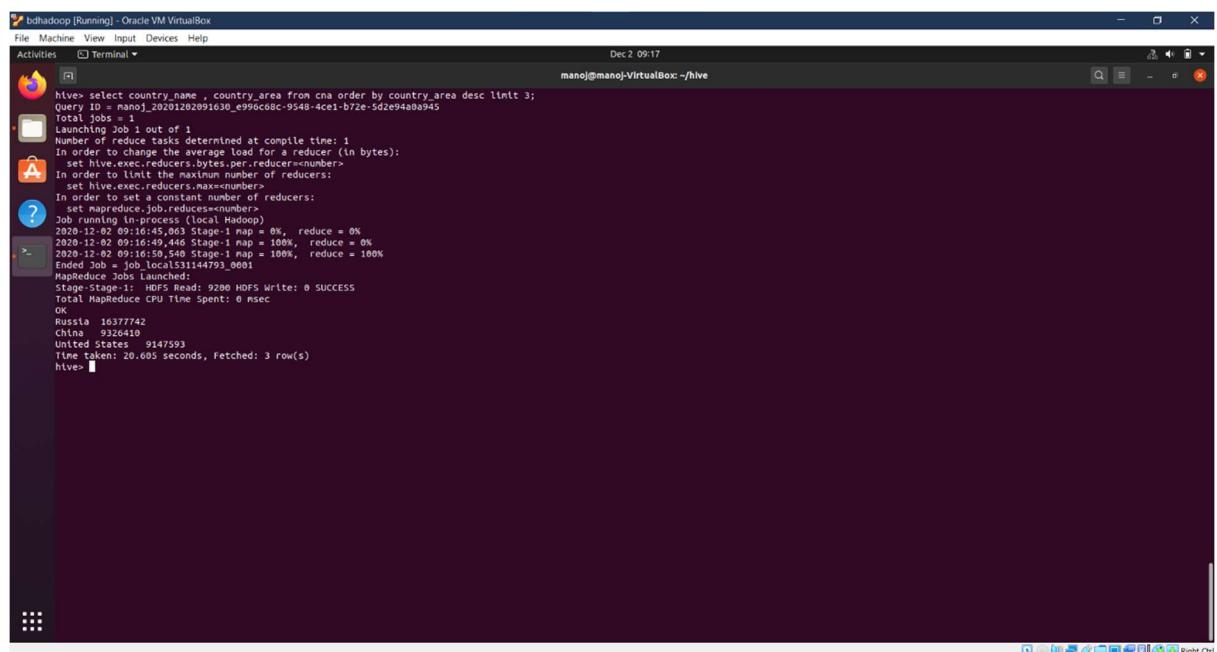
Conclusion: Net migration rate is the difference between the number of emigrants and the number of immigrants every year. The net migration rate is positive when the number of emigrants > number of immigrants .

In the country **Montserrat** , the **difference** between the number of people who leave the country and the number of people who come and settle in the country is **the highest (i.e1693)**.

-X-

3] DATASET : country_names_area.csv

Let's find the top 3 countries with the highest area (in sq. kilometers).



```
bin$ hadoop [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Dec 2 09:17
mano@mano-VirtualBox: ~/hive
hive> select country_name , country_area from cna order by country_area desc limit 3;
Total Jobs: 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducer.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in local mode (mapreduce.framework.name=Local)
2020-12-02 09:16:45,863 Stage-1 map = 0%, reduce = 0%
2020-12-02 09:16:49,446 Stage-1 map = 100%, reduce = 0%
2020-12-02 09:16:50,540 Stage-1 map = 100%, reduce = 100%
Ended Job = job_local1$11144793_0001
MapReduce Jobs Launched:
Stage-Stage-1: HDFS Read: 9200 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 nsec
OK
Russia 1637742
China 9326410
United States 9147593
Time taken: 20.605 seconds, Fetched: 3 row(s)
hive>
```

Code:

- Select country_name, country_area from cna order by country_area desc limit 3 ;

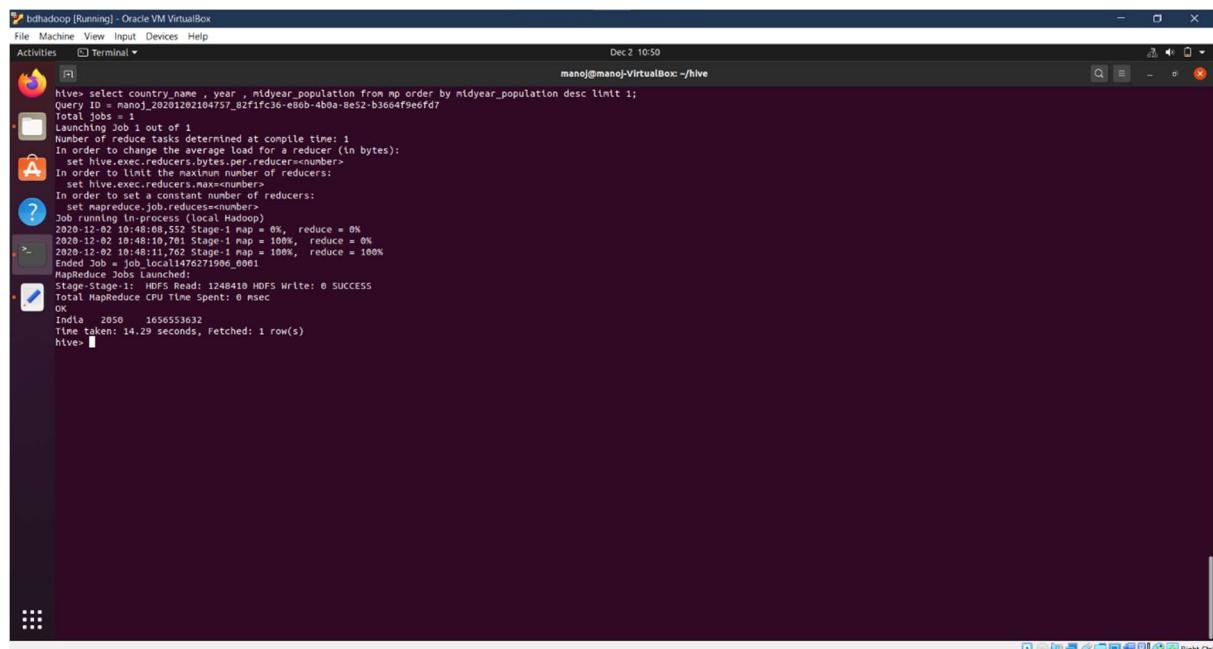
Results:

1. Russia – 16377742 sq.kms
2. China – 9326410 sq.kms
3. USA – 9147593 sq.kms

Conclusion: The top 3 countries with the highest area are **RUSSIA , CHINA , USA.**

-X-

4] DATASET : midyear_population.csv



The screenshot shows a terminal window titled "bdhadoop [Running] - Oracle VM VirtualBox". The command entered is "hive select country_name, year , midyear_population from mp order by midyear_population desc limit 1;". The output shows the execution of the query, including job launching, map/reduce stages, and the final result: "India 2050 1656553632". The terminal window has a dark background with light-colored text and icons.

Code:

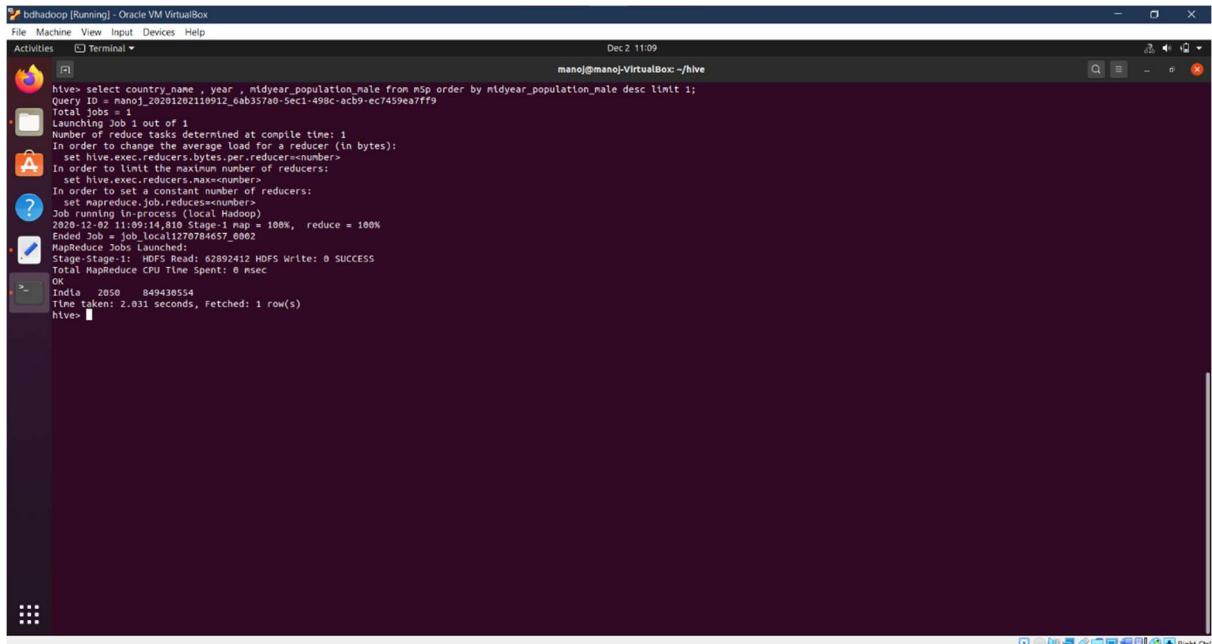
- Select country_name, year , midyear_population from mp order by midyear_population desc limit 1 ;

Results:

India – 2050 – 1656553632

Conclusion: According to the analysis , India will have the highest population in the world in the year 2050 with a population of 1.6Billion.

5] DATASET : midyear_population_5yr_age_sex



```
hive> select country_name , year , midyear_population_male from m5p order by midyear_population_male desc limit 1;
Query ID = manoj_20291202110912_6ab357a0-5ec1-498c-acb9-ec7459ea7ff9
Total jobs : 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job Running in progress (local Hadoop)
20291202110912_6ab357a0-5ec1-498c-acb9-ec7459ea7ff9
Ended Job = job_local270784657_0002
MapReduce Jobs Launched:
 Stage-Stage-1: HDFS Read: 62892412 HDFS Write: 0 SUCCESS
 Total MapReduce CPU Time Spent: 0 msec
0
India 2050 849430554
Time taken: 2.031 seconds, Fetched: 1 row(s)
hive>
```

Code:

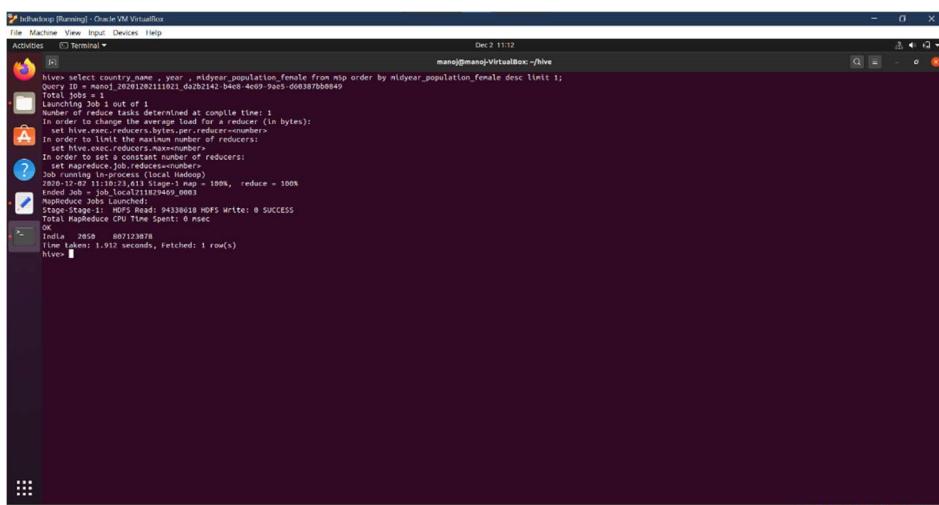
- Select country_name, year , midyear_population_male from m5p order by midyear_population_male desc limit 1 ;

Results:

India – 2050 – 849430554

Conclusion: According to the analysis , India will have the highest male population in the world in the year 2050 with a population of 0.84 Billion.

-X-



```
hive> select country_name , year , midyear_population_female from m5p order by midyear_population_female desc limit 1;
Query ID = manoj_20291202110912_d02b2442_b4e0_4e05_9e05_666387000049
Total jobs : 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job Running in progress (local Hadoop)
20291202110912_d02b2442_b4e0_4e05_9e05_666387000049
Ended Job = job_local271129469_0003
MapReduce Jobs Launched:
 Stage-Stage-1: HDFS Read: 64338018 HDFS Write: 0 SUCCESS
 Total MapReduce CPU Time Spent: 0 msec
0
India 2050 807123878
Time taken: 1.912 seconds, Fetched: 1 row(s)
hive>
```

Code:

- Select country_name, year , midyear_population_female from m5p order by midyear_population_female desc limit 1 ;

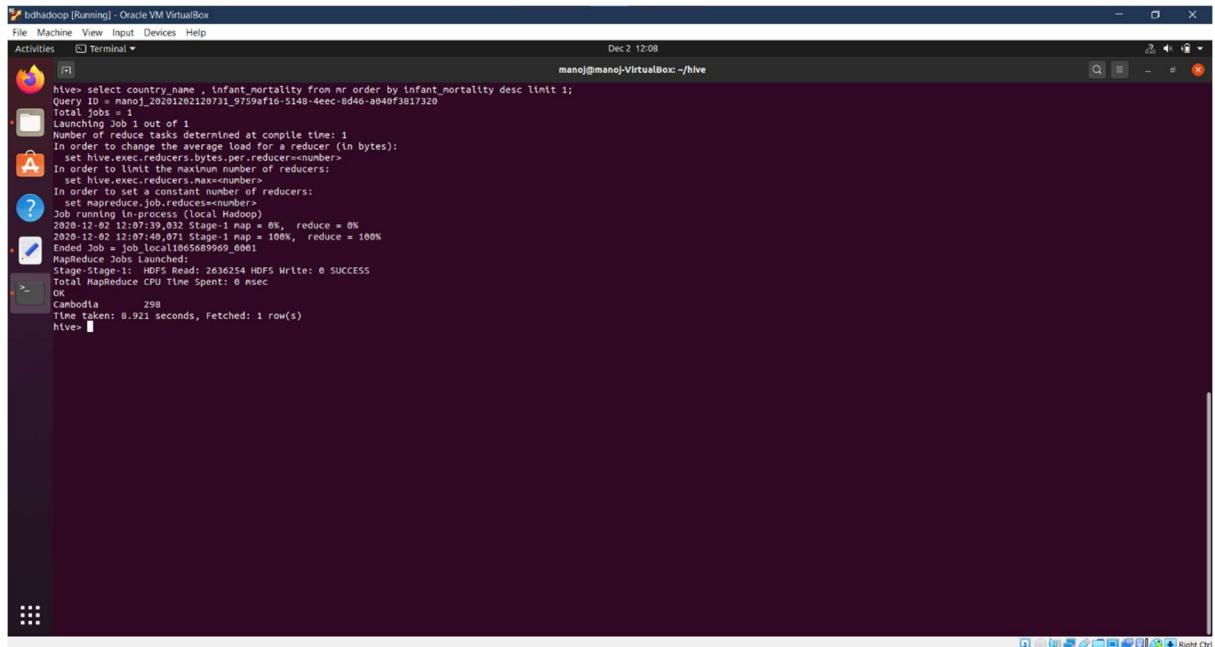
Results:

India – 2050 – 807923078

Conclusion: According to the analysis , India will have the highest female population in the world in the year 2050 with a population of 0.807Billion.

-X-

6] DATASET : mortality_life_expectancy.csv



The screenshot shows a terminal window titled "bdhadoop [Running] - Oracle VM VirtualBox". The command entered is "hive> select country_name , infant_mortality from mr order by infant_mortality desc limit 1;". The output shows a single row for Cambodia with a value of 298. The terminal window is running on a Linux desktop environment with a dark theme.

```
hive> select country_name , infant_mortality from mr order by infant_mortality desc limit 1;
Query ID = manoj_20201202120731_9759af16-5148-4ecc-8d46-a040f3817320
Total Jobs = 1
Launching Job 1 out of 1
Number of Reduce tasks determined at compile time: 1
In order to change the average load for a reducer (<n bytes>):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set the minimum number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2020-12-02 12:07:39,032 Stage-1 map = 0%,  reduce = 0%
2020-12-02 12:07:39,071 Stage-1 map = 100%,  reduce = 100%
End Job: mapred.localjob.0000000000.0001
MapReduce Jobs Launched:
Stage-Stage-1:   HDFS Read: 2636254 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Cambodia      298
Time taken: 8.921 seconds, Fetched: 1 row(s)
hive>
```

Code:

- Select country_name, year , infant_mortality from mr order by infant_mortality desc limit 1 ;

Results:

Cambodia - 298

Conclusion: Infant Mortality Rate means the number of deaths per 1000 that happen before the child is born. From our analysis,

In Cambodia , for every 1000 births , 298 babies die .

The screenshot shows a terminal window titled "bdhdoo [Running] - Oracle VM VirtualBox". The command entered is "hive> select country_name , infant_mortality_male from mr order by infant_mortality_male desc limit 1;". The output shows a single row: "Cambodia 314". The terminal window has a dark background with light-colored text. The title bar includes the application name, window controls, and the date/time "Dec 2 12:09". The bottom of the window shows various icons for file operations.

Code:

- Select country_name, year , infant_mortality_male from mr order by infant_mortality_male desc limit 1 ;

Results:

Cambodia - 314

Conclusion: In Cambodia , for every 1000 births , 314 male babies die

-X-

The screenshot shows a terminal window titled "bdhdoo [Running] - Oracle VM VirtualBox". The command entered is "hive> select country_name , infant_mortality_female from mr order by infant_mortality_female desc limit 1;". The output shows a single row: "Cambodia 281". The terminal window has a dark background with light-colored text. The title bar includes the application name, window controls, and the date/time "Dec 2 12:10". The bottom of the window shows various icons for file operations.

Code:

- Select country_name, year , infant_mortality_female from mr order by infant_mortality_female desc limit 1 ;

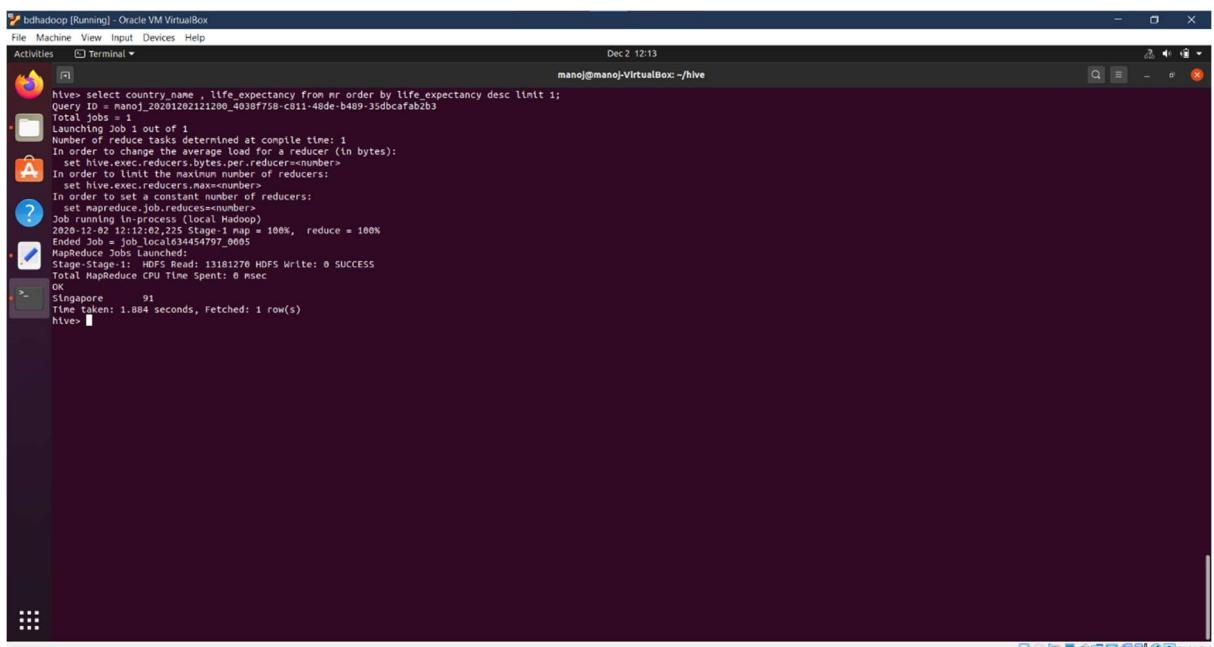
Results:

Cambodia - 281

Conclusion: From our analysis , we conclude that

In Cambodia , for every 1000 births , 281 female babies die .

-X-



```
hive> select country_name , life_expectancy from mr order by life_expectancy desc limit 1;
Query ID = manoj_20201202121200_4038f758-c811-48de-b489-35dbcab2b3
Total jobs = 1
Launching job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to limit the minimum number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2020-12-02 12:12:02,225 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local1634454797_0005
MapReduce Jobs completed:
Stage-Stage1: HDFS Read: 13181278 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Singapore      91
Time taken: 1.884 seconds, Fetched: 1 row(s)
hive>
```

Code:

- Select country_name, year , life_expectancy from mr order by life_expectancy desc limit 1 ;

Results:

Singapore - 91

Conclusion: Life expectancy means the average number of years the person in a country can live.

From our Analysis , we conclude that

In Singapore , maximum number of people on average live upto the 91 years !

```
hive> select country_name , life_expectancy_male from mr order by life_expectancy_male desc limit 1;
Query ID = manoj_20201202121330_8e7b6ed1-0d46-4043-8be0-ddzab9b38889
Total Jobs : 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2020-12-02 12:13:32,307 Stage-1 map = 100%,  reduce = 100%
End Job  JobID: 20201202121330_8e7b6ed1-0d46-4043-8be0-ddzab9b38889
MapReduce Jobs Launched:
  Stage-Stage-1: HDFS Read: 15817524 HDFS Write: 0 SUCCESS
  Total MapReduce CPU Time Spent: 0 msec
OK
Singapore      88
Time taken: 1.614 seconds, Fetched: 1 row(s)
hive>
```

Code:

- Select country_name, year , life_expectancy_male from mr order by life_expectancy_male desc limit 1 ;

Results:

Singapore - 88

Conclusion: From our Analysis , we conclude that

In Singapore , maximum number of men on average upto the age of 88 years !

-X-

```
hive> select country_name , life_expectancy_female from mr order by life_expectancy_female desc limit 1;
Query ID = manoj_20201202121409_43962d5c-a35d-4829-b8eb-b12347bc88fc
Total Jobs : 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2020-12-02 12:14:10,047 Stage-1 map = 100%,  reduce = 100%
End Job  JobID: 20201202121409_43962d5c-a35d-4829-b8eb-b12347bc88fc
MapReduce Jobs Launched:
  Stage-Stage-1: HDFS Read: 18453778 HDFS Write: 0 msec
  Total MapReduce CPU Time Spent: 0 msec
OK
Monaco      94
Time taken: 1.645 seconds, Fetched: 1 row(s)
hive>
```

Code:

- Select country_name, year , life_expectancy_female from mr order by life_expectancy_female desc limit 1 ;

Results:

Monaco - 94

Conclusion: From our Analysis , we conclude that

In Monaco , maximum number of women on average upto the age of 94 years !

-X-

7] DATASET : midyear_population_age_country_code:

Let's see what the population of India will be in the year 2026.

Code:

```
hive> select distinct country_name, year, (population_age_0+population_age_1+population_age_2+population_age_3+population_age_4+population_age_5+population_age_6+population_age_7+population_age_8+population_age_9+population_age_10+population_age_11+population_age_12+population_age_13+population_age_14+population_age_15+population_age_16+population_age_17+population_age_18+population_age_19+population_age_20) as sum, (population_age_21+population_age_22+population_age_23+population_age_24+population_age_25+population_age_26+population_age_27+population_age_28+population_age_29+population_age_30+population_age_31+population_age_32+population_age_33+population_age_34+population_age_35+population_age_36+population_age_37+population_age_38+population_age_39+population_age_40) as sum1, (population_age_41+population_age_42+population_age_43+population_age_44+population_age_45+population_age_46+population_age_47+population_age_48+population_age_49+population_age_50+population_age_51+population_age_52+population_age_53+population_age_54+population_age_55+population_age_56+population_age_57+population_age_58+population_age_59+population_age_60) as sum2, (population_age_61+population_age_62+population_age_63+population_age_64+population_age_65+population_age_66+population_age_67+population_age_68+population_age_69+population_age_70+population_age_71+population_age_72+population_age_73+population_age_74+population_age_75+population_age_76+population_age_77+population_age_78+population_age_79+population_age_80) as sum3, (population_age_81+population_age_82+population_age_83+population_age_84+population_age_85+population_age_86+population_age_87+population_age_88+population_age_89+population_age_90+population_age_91+population_age_92+population_age_93+population_age_94+population_age_95+population_age_96+population_age_97+population_age_98+population_age_99+population_age_100) as sum4, (population_age_0+population_age_1+population_age_2+population_age_3+population_age_4+population_age_5+population_age_6+population_age_7+population_age_8+population_age_9+population_age_10+population_age_11+population_age_12+population_age_13+population_age_14+population_age_15+population_age_16+population_age_17+population_age_18+population_age_19+population_age_20+population_age_21+population_age_22+population_age_23+population_age_24+population_age_25+population_age_26+population_age_27+population_age_28+population_age_29+population_age_30+population_age_31+population_age_32+population_age_33+population_age_34+population_age_35+population_age_36+population_age_37+population_age_38+population_age_39+population_age_40+population_age_41+population_age_42+population_age_43+population_age_44+population_age_45+population_age_46+population_age_47+population_age_48+population_age_49+population_age_50+population_age_51+population_age_52+population_age_53+population_age_54+population_age_55+population_age_56+population_age_57+population_age_58+population_age_59+population_age_60) as sum5 from midyear_population_age_country_code where year=2036 and country_name='India' order by country_name desc limit 1;
query OK
Time taken: 0.001 seconds, Fetched: 1 row(s)
Total jobs: 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 3
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to set the number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2026-12-04 03:50:03,788 Stage-1 map = 0%, reduce = 0%
2026-12-04 03:50:04,788 Stage-1 map = 100%, reduce = 0%
2026-12-04 03:50:08,709 Stage-1 map = 100%, reduce = 100%
Ended Job = job_local095964774_0018
MapReduce Jobs Launched:
Stage-Stage-1: HDFS Read: 58382030192 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
India 2036 227588702 214539462 182701858 101983940 13395251
Time taken: 0.512 seconds, Fetched: 1 row(s)
```

Result:

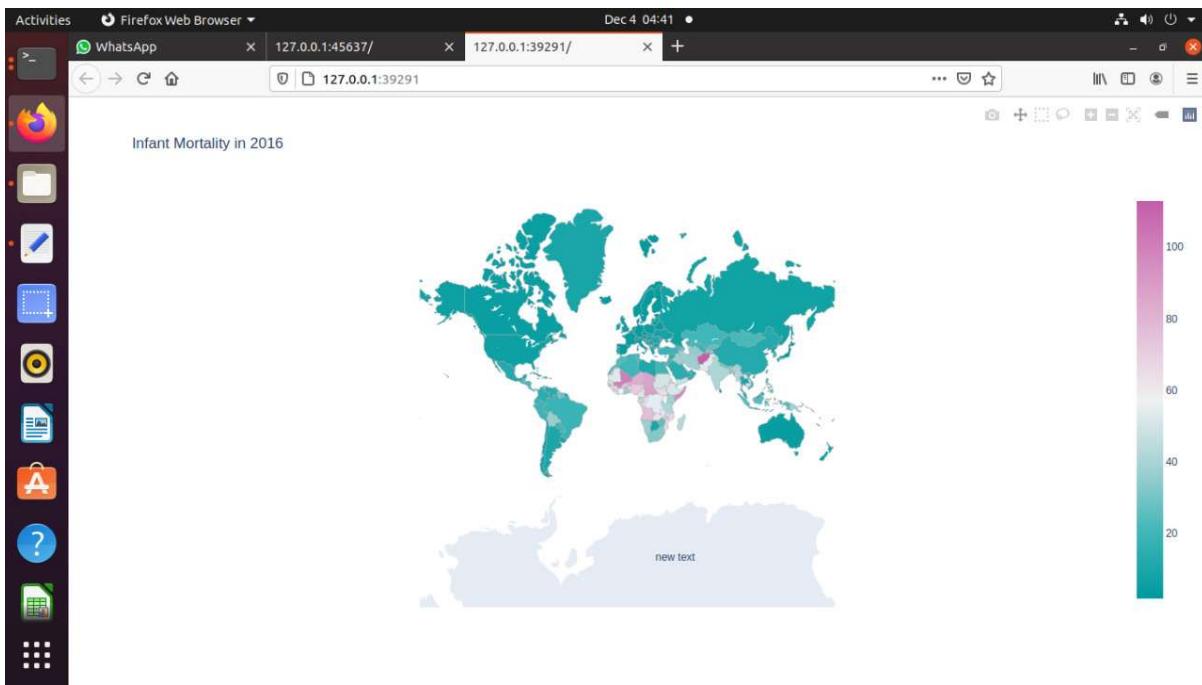
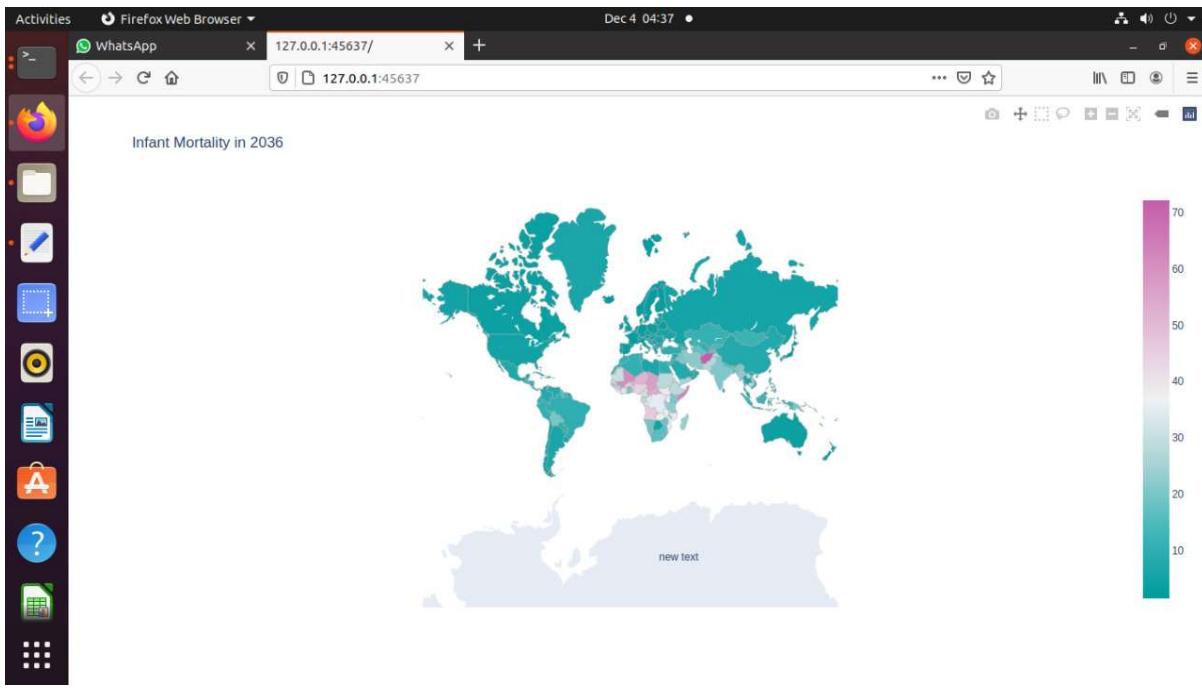
Total MapReduce CPU Time Spent: 0 msec								
OK								
Afghanistan	2036	12124289	7870651	3952910	1285414	91393		
Afghanistan	2036	12535530	8050863	3981785	1144932	59629		
Albania	2036	340090	347321	461342	324348	82800		
Albania	2036	363254	380111	433749	267450	49946		
Algeria	2036	7772104	6781490	6512027	3356350	550976		
Algeria	2036	8160969	7111035	6631157	3206951	411262		
American Samoa	2036	6499	5124	8217	4644	403		
American Samoa	2036	6712	5720	7495	5458	798		
Andorra	2036	6283	8211	9965	13313	3327		
Andorra	2036	6687	8663	10207	13193	2826		
Angola	2036	8513827	4862680	2364129	831339	78967		
Angola	2036	8855527	4991014	2400433	740607	55382		
Anguilla	2036	3075	2998	3430	2638	399		
Anguilla	2036	3155	2805	2463	1931	317		
Antigua and Barbuda	2036	15961	16217	15487	11291	2010		
Antigua and Barbuda	2036	16393	15029	12620	8277	1145		
Argentina	2036	7192005	6853090	6333842	4057578	1220248		
Argentina	2036	7645642	7172201	6258038	3478540	653943		
Armenia	2036	274936	297625	416983	334181	76322		
Armenia	2036	287489	318673	445246	283989	41504		
Aruba	2036	15805	16849	19694	16366	4546		
Aruba	2036	15972	16473	17908	12763	2104		
Australia	2036	3091938	3383234	3365701	2697514	990088		
Australia	2036	3271899	3568367	3525515	2538752	660200		
Austria	2036	874228	1000854	1206237	1190328	398575		
Austria	2036	916399	1021951	1204990	1067316	244761		
Azerbaijan	2036	1357308	1353427	1612259	1080665	153179		
Azerbaijan	2036	1442545	1536432	1601982	798031	70681		
Bahamas The	2036	48825	49118	46594	34633	7080		
Bahamas The	2036	50231	50437	46938	28599	3264		
Bahrain	2036	247565	172025	74297	10411			
Bahrain	2036	213012	421376	279029	84303	8119		
Bangladesh	2036	28149337		26908621	24280862	13023884	2062542	
Bangladesh	2036	29157333		26000677	21847502	11030444	1339113	
Barbados	2036	32831	34146	37765	31897	5275		
Barbados	2036	32974	35224	38742	37298	9753		
Belarus	2036	847450	952712	1399927	1221128	349356		
Belarus	2036	897317	1001291	1344781	796131	108872		
Belgium	2036	1382988	1491618	1560060	1389461	510697		
Belgium	2036	1448511	1529094	1568849	1273025	315330		
Belize	2036	86196	73941	48735	22321	3239		
Belize	2036	90090	77881	50932	19363	2044		
Benin	2036	4037757	2660102	1333103	485859	54645		
Benin	2036	4204758	2731878	1346838	433445	28044		
Bermuda	2036	8178	8304	8142	9599	3699		
Bermuda	2036	8345	8464	8183	8194	2087		
Bhutan	2036	120970	129960	125052	48107	8719		

Conclusion:

The above screenshot show the population of different countries of different age groups (i.e 0-20 , 21-40 , 41-60 , 61-80 , 81-100).

SPARK ANALYSIS:

1] Infant Mortality Rate :



Code :

```
from pyspark import SparkContext, SparkConf  
  
from pyspark.sql import SparkSession  
  
sc=SparkContext('local')
```

```
spark=SparkSession(sc)
```

```
import pandas as pd  
import pydoop.hdfs as hd  
import plotly.graph_objects as go  
from mpl_toolkits.mplot3d import Axes3D  
from sklearn.preprocessing import StandardScaler  
import matplotlib.pyplot as plt # plotting  
import numpy as np # linear algebra
```

```
t3 = spark.read.option('header','true').csv("hdfs://localhost:9000/project/mortality_life_expectancy.csv")  
  
with hd.open("/project/mortality_life_expectancy.csv") as b:  
  
    df2 = pd.read_csv(b)  
  
    df2.dataframeName = 'mortality_life_expectancy.csv'  
  
countries={'Afghanistan':'AFG',  
  
'Aland Islands':'ALA','Albania':'ALB','Algeria':'DZA','American  
Samoa':'ASM','Andorra':'AND','Angola':'AGO','Anguilla':'AIA','Antigua and Barbuda':'ATG',  
  
'Antarctica':'ATA','Argentina':'ARG','Armenia':'ARM','Aruba':'ABW','Australia':'AUS','Austria':'AUT','Azerbaijan':'AZE','Azerbaijd  
an':'AZE',  
  
'Bahrain':'BHR','Bahamas The':'BHS','Bangladesh':'BGD','Barbados':'BRB','Belarus':'BLR',  
  
'Belgium':'BEL','Belize':'BLZ','Benin':'BEN','Bermuda':'BMU','Bhutan':'BTN','Bolivia':'BOL','Bosnia and  
Herzegovina':'BIH','Bosnia-Herzegovina':'BIH',  
  
'Botswana':'BWA','Bouvet Island':'BVT','Brazil':'BRA','British Virgin Islands':'VGB','British Indian Ocean Territory':'IOT',  
  
'Brunei':'BRN','Brunei Darussalam':'BRN','Bulgaria':'BGR','Burkina Faso':'BFA','Burma':'MMR',  
  
'Burundi':'BDI','Cabo Verde':'CPV','Cape Verde':'CPV','Cambodia':'KHM','Cameroon':'CMR',  
  
'Canada':'CAN','Cayman Islands':'CYM','Central African Republic':'CAF','Chad':'TCD','Chile':'CHL',  
  
'Christmas Island':'CHR','China':'CHN','Colombia':'COL','Comoros':'COM','Congo (Kinshasa)':'COG','Cook Islands':'COK','Costa  
Rica':'CRI','Cote d'Ivoire':'CIV',  
  
"Ivory Coast (Cote D'Ivoire)":'CIV','Croatia':'HRV','Cuba':'CUB','Curacao':'CUW','Cyprus':'CYP',  
  
'Czech Republic':'CZE','Denmark':'DNK','Djibouti':'DJI','Dominica':'DMA','Dominican Republic':'DOM',  
  
'Ecuador':'ECU','Egypt':'EGY','El Salvador':'SLV','Equatorial Guinea':'GNQ','Eritrea':'ERI','Estonia':'EST',  
  
'Ethiopia':'ETH','Falkland Islands (Islas Malvinas)':'FLK','Falkland Islands':'FLK','Faroe Islands':'FRO',  
  
'Fiji':'FJI','Finland':'FIN','France':'FRA','French Polynesia':'PYF','Gabon':'GAB',  
  
'Gambia The':'GMB','Georgia':'GEO','Germany':'DEU','Ghana':'GHA','Gibraltar':'GIB',  
  
'Greece':'GRC','Greenland':'GRL','Grenada':'GRD','Guam':'GUM','Guatemala':'GTM',  
  
'Guernsey':'GGY','Guinea-Bissau':'GNB','Guinea':'GIN','Guyana':'GUY','French Guyana':'GUY','Haiti':'HTI',
```

'Honduras':'HND','Heard and McDonald Islands':'HMD','Hong Kong':'HKG','Hungary':'HUN','Iceland':'ISL',
'India':'IND','Indonesia':'IDN','Iran':'IRN','Iraq':'IRQ','Ireland':'IRL','Isle of Man':'IMN',
'Israel':'ISR','Italy':'ITA','Jamaica':'JAM','Japan':'JPN','Jersey':'JEY','Jordan':'JOR',
'Kazakhstan':'KAZ','Kenya':'KEN','Kiribati':'KIR','Korea North':'KOR','Korea South':'PRK',
'South Korea':'PRK','North Korea':'KOR','Kosovo':'KSV','Kuwait':'KWT','Kyrgyzstan':'KGZ',
'Laos':'LAO','Latvia':'LVA','Lebanon':'LBN','Lesotho':'LSO','Liberia':'LBR','Libya':'LBY','Liechtenstein':'LIE',
'Lithuania':'LTU','Luxembourg':'LUX','Macau':'MAC','Macedonia':'MKD','Madagascar':'MDG',
'Malawi':'MWI','Malaysia':'MYS','Maldives':'MDV','Mali':'MLI','Malta':'MLT','Marshall Islands':'MHL',
'Martinique (French)':'MTQ','Mauritania':'MRT','Mauritius':'MUS','Mexico':'MEX','Micronesia, Federated States of':'FSM',
'Moldova':'MDA','Moldavia':'MDA','Monaco':'MCO','Mongolia':'MNG','Montenegro':'MNE','Montserrat':'MSR',
'Morocco':'MAR','Mozambique':'MOZ','Myanmar':'MMR','Namibia':'NAM','Nepal':'NPL','Netherlands':'NLD',
'Netherlands Antilles':'ANT','New Caledonia':'NCL','New Caledonia (French)':'NCL','New Zealand':'NZL','Nicaragua':'NIC',
'Nigeria':'NGA','Niger':'NER','Niue':'NIU','Northern Mariana Islands':'MNP','Norway':'NOR','Oman':'OMN',
'Pakistan':'PAK','Palau':'PLW','Panama':'PAN','Papua New Guinea':'PNG','Paraguay':'PRY','Peru':'PER',
'Philippines':'PHL','Pitcairn Island':'PCN','Poland':'POL','Polynesia (French)':'PYF','Portugal':'PRT',
'Puerto Rico':'PRI','Qatar':'QAT','Reunion (French)':'REU','Romania':'ROU','Russia':'RUS','Russian Federation':'RUS',
'Rwanda':'RWA','Saint Kitts and Nevis':'KNA','Saint Lucia':'LCA','Saint Martin':'MAF','Saint Pierre and Miquelon':'SPM',
'Saint Vincent and the Grenadines':'VCT','Saint Vincent & Grenadines':'VCT','S. Georgia & S. Sandwich
Isls.':'SGS','Samoa':'WSM',
'San Marino':'SMR','Saint Helena':'SHN','Sao Tome and Principe':'STP','Saudi Arabia':'SAU','Senegal':'SEN',
'Serbia':'SRB','Seychelles':'SYC','Sierra Leone':'SLE','Singapore':'SGP','Sint Maarten':'SXM',
'Slovakia':'SVK','Slovak Republic':'SVK','Slovenia':'SVN','Solomon Islands':'SLB','Somalia':'SOM','South Africa':'ZAF',
'South Sudan':'SSD','Spain':'ESP','Sri Lanka':'LKA','Sudan':'SDN','Suriname':'SUR',
'Swaziland':'SWZ','Sweden':'SWE','Switzerland':'CHE','Syria':'SYR','Taiwan':'TWN','Tajikistan':'TJK',
'Tadzhikistan':'TJK','Tanzania':'TZA','Thailand':'THA','Timor-Leste':'TLS','Togo':'TGO','Tonga':'TON',
'Trinidad and Tobago':'TTO','Tunisia':'TUN','Turkey':'TUR','Turkmenistan':'TKM','Tuvalu':'TUV',
'Uganda':'UGA','Ukraine':'UKR','United Arab Emirates':'ARE','United Kingdom':'GBR','United States':'USA',
'U.S. Minor Outlying Islands':'UMI','Uruguay':'URY','Uzbekistan':'UZB','Vanuatu':'VUT',
'Vatican City State':'VAT','Venezuela':'VEN','Vietnam':'VNM','Virgin Islands':'VGB',
'Virgin Islands U.S.':'VIR','Virgin Islands British':'VGB','West Bank':'WBG','Yemen':'YEM',
'Zaire':'ZAR','Zambia':'ZMB','Zimbabwe':'ZWE','Saint Barthelemy':'BLM','Nauru':'NRU',
'Western Sahara':'ESH','Wallis and Futuna':'WLF','Gaza Strip':'GZA','Micronesia Federated States
of':'FSM','Czechia':'CZE','Congo (Brazzaville)':'COG','Turks and Caicos Islands':'TCA'}

codes = [countries[country] if country != 'I prefer not to say' else None for country in df2['country_name']]
df2['code']=codes

```

sbd = df2[['code','infant_mortality','year']]

sbdgroup36 = sbd[sbd['year']==2036].drop('year',axis=1).groupby('code').mean()

sbdgroup16 = sbd[sbd['year']==2016].drop('year',axis=1).groupby('code').mean()

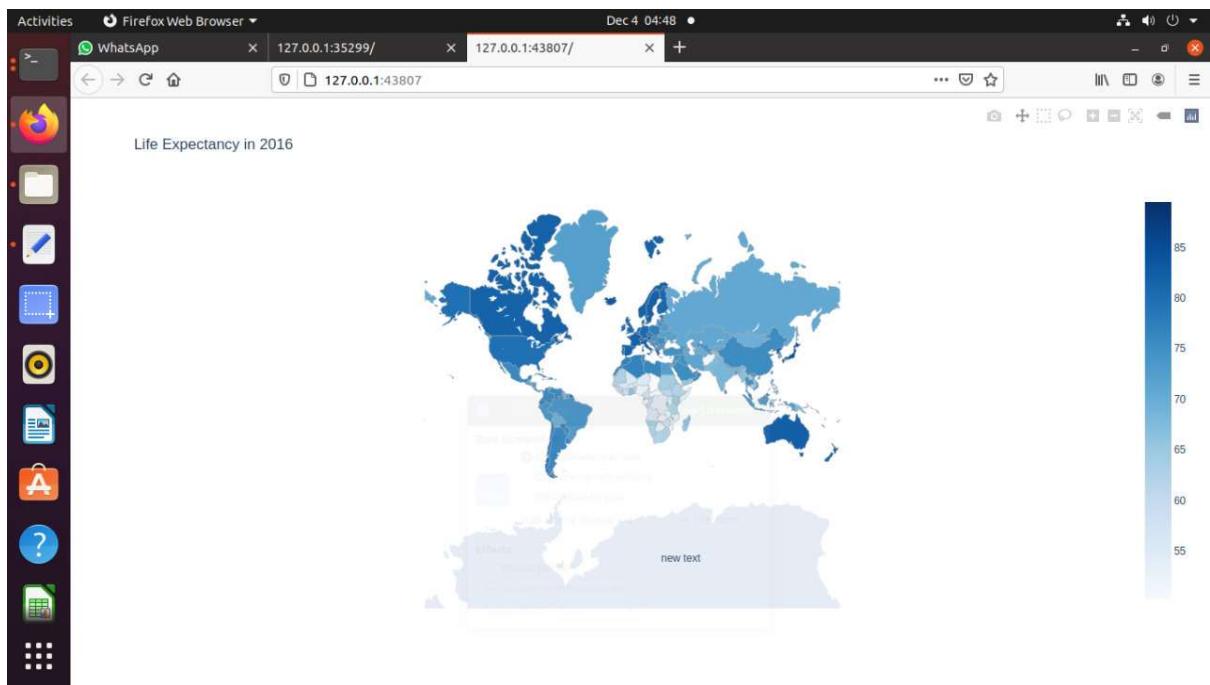
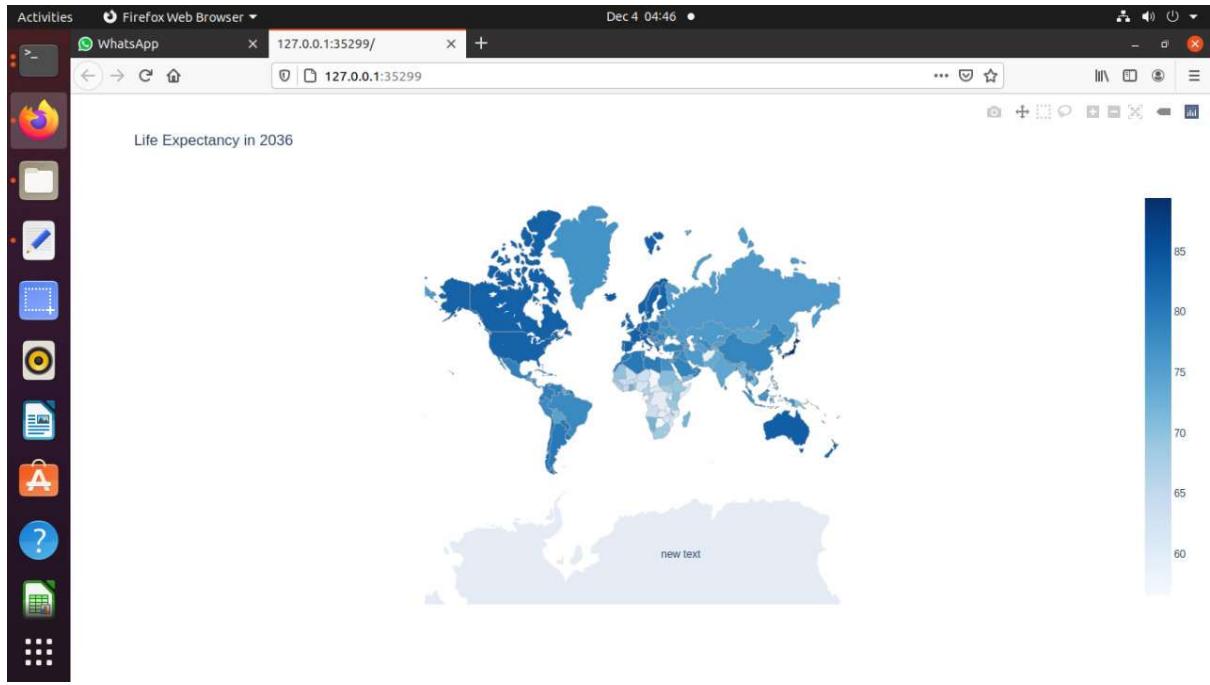
fig = go.Figure(data=go.Choropleth(
    locations = sbdgroup16.index,
    z = sbdgroup16['infant_mortality'],
    text = sbdgroup16.index,
    colorscale = 'tropic',
    autocolorscale=False,
    reversescale=False,
    marker_line_color='darkgray',
    marker_line_width=0.5,
    colorbar_tickprefix = '',
    colorbar_title = '',
))

fig.update_layout(
    title_text='Infant Mortality in 2016',
    geo=dict(
        showframe=False,
        showcoastlines=False,
        projection_type='mercator'
    ),
    annotations = [dict(
        x=0.55,
        y=0.1,
        xref='paper',
        yref='paper',
        showarrow = False
    )]
)

fig.show()

```

2] Life Expectancy :



Code :

```
from pyspark import SparkContext, SparkConf  
  
from pyspark.sql import SparkSession  
  
sc=SparkContext('local')  
  
spark=SparkSession(sc)
```

```
import pandas as pd

import pydoop.hdfs as hd

import plotly.graph_objects as go

from mpl_toolkits.mplot3d import Axes3D

from sklearn.preprocessing import StandardScaler

import matplotlib.pyplot as plt # plotting

import numpy as np # linear algebra

t3 = spark.read.option('header','true').csv("hdfs://localhost:9000/project/mortality_life_expectancy.csv")

with hd.open("/project/mortality_life_expectancy.csv") as b:

    df2 = pd.read_csv(b)

df2.dataframeName = 'mortality_life_expectancy.csv'

countries={'Afghanistan':'AFG',

'Aland Islands':'ALA','Albania':'ALB','Algeria':'DZA','American Samoa':'ASM','Andorra':'AND','Angola':'AGO','Anguilla':'AIA','Antigua and Barbuda':'ATG','Antarctica':'ATA','Argentina':'ARG','Armenia':'ARM','Aruba':'ABW','Australia':'AUS','Austria':'AUT','Azerbaijan':'AZE','Azerbaijan':'AZE','Bahrain':'BHR','Bahamas The':'BHS','Bangladesh':'BGD','Barbados':'BRB','Belarus':'BLR','Belgium':'BEL','Belize':'BLZ','Benin':'BEN','Bermuda':'BMU','Bhutan':'BTN','Bolivia':'BOL','Bosnia and Herzegovina':'BIH','Bosnia-Herzegovina':'BIH','Botswana':'BWA','Bouvet Island':'BVT','Brazil':'BRA','British Virgin Islands':'VGB','British Indian Ocean Territory':'IOT','Brunei':'BRN','Brunei Darussalam':'BRN','Bulgaria':'BGR','Burkina Faso':'BFA','Burma':'MMR','Burundi':'BDI','Cabo Verde':'CPV','Cape Verde':'CPV','Cambodia':'KHM','Cameroon':'CMR','Canada':'CAN','Cayman Islands':'CYM','Central African Republic':'CAF','Chad':'TCD','Chile':'CHL','Christmas Island':'CHR','China':'CHN','Colombia':'COL','Comoros':'COM','Congo (Kinshasa)':'COG','Cook Islands':'COK','Costa Rica':'CRI','Cote d'Ivoire':'CIV','Ivory Coast (Cote D'Ivoire)':'CIV','Croatia':'HRV','Cuba':'CUB','Curacao':'CUW','Cyprus':'CYP','Czech Republic':'CZE','Denmark':'DNK','Djibouti':'DJI','Dominica':'DMA','Dominican Republic':'DOM','Ecuador':'ECU','Egypt':'EGY','El Salvador':'SLV','Equatorial Guinea':'GNQ','Eritrea':'ERI','Estonia':'EST','Ethiopia':'ETH','Falkland Islands (Islas Malvinas)':'FLK','Falkland Islands':'FLK','Faroe Islands':'FRO','Fiji':'FJI','Finland':'FIN','France':'FRA','French Polynesia':'PYF','Gabon':'GAB','Gambia The':'GMB','Georgia':'GEO','Germany':'DEU','Ghana':'GHA','Gibraltar':'GIB','Greece':'GRC','Greenland':'GRL','Grenada':'GRD','Guam':'GUM','Guatemala':'GTM','Guernsey':'GGY','Guinea-Bissau':'GNB','Guinea':'GIN','Guyana':'GUY','French Guyana':'GUY','Haiti':'HTI','Honduras':'HND','Heard and McDonald Islands':'HMD','Hong Kong':'HKG','Hungary':'HUN','Iceland':'ISL',
```

'India':'IND','Indonesia':'IDN','Iran':'IRN','Iraq':'IRQ','Ireland':'IRL','Isle of Man':'IMN',
'Israel':'ISR','Italy':'ITA','Jamaica':'JAM','Japan':'JPN','Jersey':'JEY','Jordan':'JOR',
'Kazakhstan':'KAZ','Kenya':'KEN','Kiribati':'KIR','Korea North':'KOR','Korea South':'PRK',
'South Korea':'PRK','North Korea':'KOR','Kosovo':'KSV','Kuwait':'KWT','Kyrgyzstan':'KGZ',
'Laos':'LAO','Latvia':'LVA','Lebanon':'LBN','Lesotho':'LSO','Liberia':'LBR','Libya':'LYB','Liechtenstein':'LIE',
'Lithuania':'LTU','Luxembourg':'LUX','Macau':'MAC','Macedonia':'MKD','Madagascar':'MDG',
'Malawi':'MWI','Malaysia':'MYS','Maldives':'MDV','Mali':'MLI','Malta':'MLT','Marshall Islands':'MHL',
'Martinique (French)':'MTQ','Mauritania':'MRT','Mauritius':'MUS','Mexico':'MEX','Micronesia, Federated States of':'FSM',
'Moldova':'MDA','Moldavia':'MDA','Monaco':'MCO','Mongolia':'MNG','Montenegro':'MNE','Montserrat':'MSR',
'Morocco':'MAR','Mozambique':'MOZ','Myanmar':'MMR','Namibia':'NAM','Nepal':'NPL','Netherlands':'NLD',
'Netherlands Antilles':'ANT','New Caledonia':'NCL','New Caledonia (French)':'NCL','New Zealand':'NZL','Nicaragua':'NIC',
'Nigeria':'NGA','Niger':'NER','Niue':'NIU','Northern Mariana Islands':'MNP','Norway':'NOR','Oman':'OMN',
'Pakistan':'PAK','Palau':'PLW','Panama':'PAN','Papua New Guinea':'PNG','Paraguay':'PRY','Peru':'PER',
'Philippines':'PHL','Pitcairn Island':'PCN','Poland':'POL','Polynesia (French)':'PYF','Portugal':'PRT',
'Puerto Rico':'PRI','Qatar':'QAT','Reunion (French)':'REU','Romania':'ROU','Russia':'RUS','Russian Federation':'RUS',
'Rwanda':'RWA','Saint Kitts and Nevis':'KNA','Saint Lucia':'LCA','Saint Martin':'MAF','Saint Pierre and Miquelon':'SPM',
'Saint Vincent and the Grenadines':'VCT','Saint Vincent & Grenadines':'VCT','S. Georgia & S. Sandwich
Isls.':'SGS','Samoa':'WSM',
'San Marino':'SMR','Saint Helena':'SHN','Sao Tome and Principe':'STP','Saudi Arabia':'SAU','Senegal':'SEN',
'Serbia':'SRB','Seychelles':'SYC','Sierra Leone':'SLE','Singapore':'SGP','Sint Maarten':'SXM',
'Slovakia':'SVK','Slovak Republic':'SVK','Slovenia':'SVN','Solomon Islands':'SLB','Somalia':'SOM','South Africa':'ZAF',
'South Sudan':'SSD','Spain':'ESP','Sri Lanka':'LKA','Sudan':'SDN','Suriname':'SUR',
'Swaziland':'SWZ','Sweden':'SWE','Switzerland':'CHE','Syria':'SYR','Taiwan':'TWN','Tajikistan':'TJK',
'Tadjikistan':'TJK','Tanzania':'TZA','Thailand':'THA','Timor-Leste':'TLS','Togo':'TGO','Tonga':'TON',
'Trinidad and Tobago':'TTO','Tunisia':'TUN','Turkey':'TUR','Turkmenistan':'TKM','Tuvalu':'TUV',
'Uganda':'UGA','Ukraine':'UKR','United Arab Emirates':'ARE','United Kingdom':'GBR','United States':'USA',
'U.S. Minor Outlying Islands':'UMI','Uruguay':'URY','Uzbekistan':'UZB','Vanuatu':'VUT',
'Vatican City State':'VAT','Venezuela':'VEN','Vietnam':'VNM','Virgin Islands':'VGB',
'Virgin Islands U.S.':'VIR','Virgin Islands British':'VGB','West Bank':'WBG','Yemen':'YEM',
'Zaire':'ZAR','Zambia':'ZMB','Zimbabwe':'ZWE','Saint Barthelemy':'BLM','Nauru':'NRU',
'Western Sahara':'ESH','Wallis and Futuna':'WLF','Gaza Strip':'GZA','Micronesia Federated States
of':'FSM','Czechia':'CZE','Congo (Brazzaville)':'COG','Turks and Caicos Islands':'TCA'}

```

codes = [countries[country] if country != 'I prefer not to say' else None for country in df2['country_name']]

df2['code']=codes

sbd = df2[['code','life_expectancy','year']]

sbdgroup36 = sbd[sbd['year']==2036].drop('year',axis=1).groupby('code').mean()

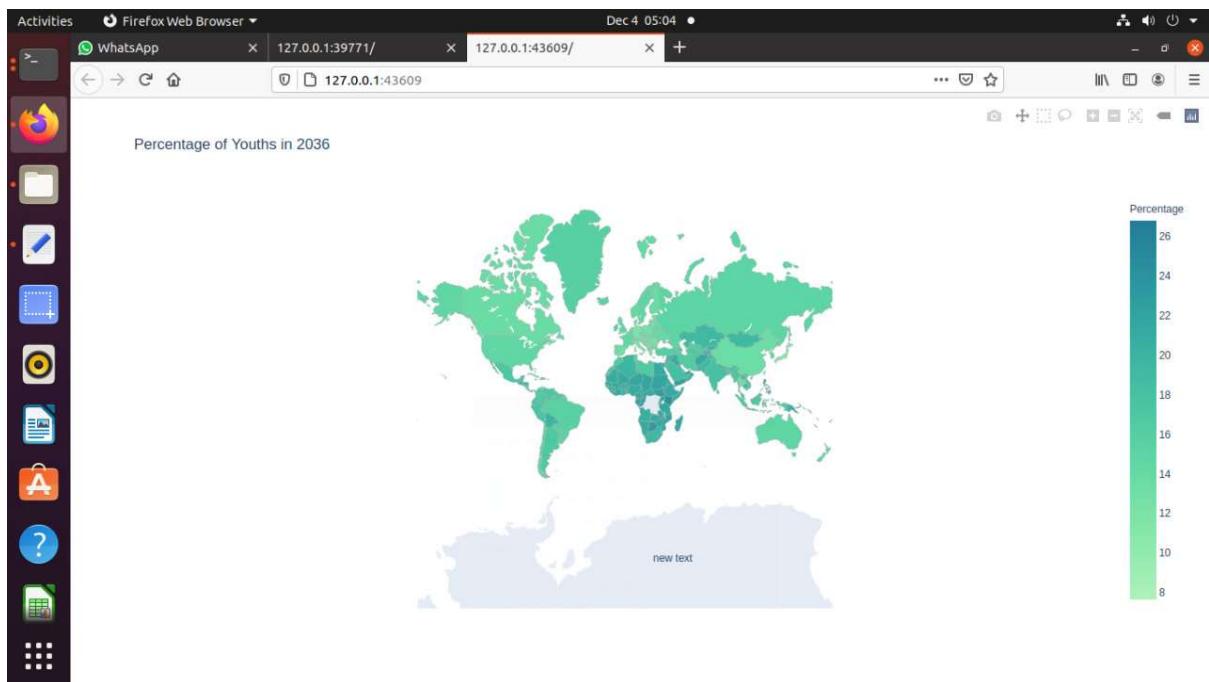
sbdgroup16 = sbd[sbd['year']==2016].drop('year',axis=1).groupby('code').mean()

fig = go.Figure(data=go.Choropleth(
    locations = sbdgroup16.index,
    z = sbdgroup16['life_expectancy'],
    text = sbdgroup16.index,
    colorscale = 'Blues',
    autocolorscale=False,
    reversescale=False,
    marker_line_color='darkgray',
    marker_line_width=0.5,
    colorbar_tickprefix = '',
    colorbar_title = '',
))

fig.update_layout(
    title_text='Life Expectancy in 2016',
    geo=dict(
        showframe=False,
        showcoastlines=False,
        projection_type='mercator'
    ),
    annotations = [dict(
        x=0.55,
        y=0.1,
        xref='paper',
        yref='paper',
        showarrow = False
    )]
)
fig.show()

```

3] Percentage of Youths in Countries:



CODE:

```
from pyspark import SparkContext, SparkConf  
  
from pyspark.sql import SparkSession  
  
sc=SparkContext('local')  
  
spark=SparkSession(sc)
```

```

import pandas as pd

import pydoop.hdfs as hd

import plotly.graph_objects as go

from mpl_toolkits.mplot3d import Axes3D

from sklearn.preprocessing import StandardScaler

import matplotlib.pyplot as plt # plotting

import numpy as np # linear algebra

t3 = spark.read.option('header','true').csv("hdfs://localhost:9000/project/midyear_population_age_sex.csv")

with hd.open("/project/midyear_population_age_sex.csv") as b:

    df = pd.read_csv(b)

    df.dataframeName = 'midyear_population_age_sex.csv'

K=[]

Y=[]

M=[]

O=[]

for i in range(len(df)):

    kids=df.loc[i,'population_age_0':'population_age_18']

    K.append(sum(kids))

    youths=df.loc[i,'population_age_19':'population_age_30']

    Y.append(sum(youths))

    middle_age=df.loc[i,'population_age_31':'population_age_60']

    M.append(sum(middle_age))

    old=df.loc[i,'population_age_61':'population_age_100']

    O.append(sum(old))

df['kids']=K

df['youths']=Y

df['middle_aged']=M

df['old_aged']=O

```

countries={'Afghanistan':'AFG',
'Aland Islands':'ALA','Albania':'ALB','Algeria':'DZA','American
Samoa':'ASM','Andorra':'AND','Angola':'AGO','Anguilla':'AIA','Antigua and Barbuda':'ATG',
'Antarctica':'ATA','Argentina':'ARG','Armenia':'ARM','Aruba':'ABW','Australia':'AUS','Austria':'AUT','Azerbaijan':'AZE','Azerba
idjan':'AZE',
'Bahrain':'BHR','Bahamas The':'BHS','Bangladesh':'BGD','Barbados':'BRB','Belarus':'BLR',
'Belgium':'BEL','Belize':'BLZ','Benin':'BEN','Bermuda':'BMU','Bhutan':'BTN','Bolivia':'BOL','Bosnia and
Herzegovina':'BIH','Bosnia-Herzegovina':'BIH',
'Botswana':'BWA','Bouvet Island':'BVT','Brazil':'BRA','British Virgin Islands':'VGB','British Indian Ocean Territory':'IOT',
'Brunei':'BRN','Brunei Darussalam':'BRN','Bulgaria':'BGR','Burkina Faso':'BFA','Burma':'MMR',
'Burundi':'BDI','Cabo Verde':'CPV','Cape Verde':'CPV','Cambodia':'KHM','Cameroon':'CMR',
'Canada':'CAN','Cayman Islands':'CYM','Central African Republic':'CAF','Chad':'TCD','Chile':'CHL',
'Christmas Island':'CHR','China':'CHN','Colombia':'COL','Comoros':'COM','Congo (Kinshasa)':'COG','Cook Islands':'COK','Costa
Rica':'CRI','Cote d\Ivoire':'CIV',
'Ivory Coast (Cote D'Ivoire)':'CIV','Croatia':'HRV','Cuba':'CUB','Curacao':'CUW','Cyprus':'CYP',
'Czech Republic':'CZE','Denmark':'DNK','Djibouti':'DJI','Dominica':'DMA','Dominican Republic':'DOM',
'Ecuador':'ECU','Egypt':'EGY','El Salvador':'SLV','Equatorial Guinea':'GNQ','Eritrea':'ERI','Estonia':'EST',
'Ethiopia':'ETH','Falkland Islands (Islas Malvinas)':'FLK','Falkland Islands':'FLK','Faroe Islands':'FRO',
'Fiji':'FJI','Finland':'FIN','France':'FRA','French Polynesia':'PYF','Gabon':'GAB',
'Gambia The':'GMB','Georgia':'GEO','Germany':'DEU','Ghana':'GHA','Gibraltar':'GIB',
'Greece':'GRC','Greenland':'GRL','Grenada':'GRD','Guam':'GUM','Guatemala':'GTM',
'Guernsey':'GGY','Guinea-Bissau':'GNB','Guinea':'GIN','Guyana':'GUY','French Guyana':'GUY','Haiti':'HTI',
'Honduras':'HND','Heard and McDonald Islands':'HMD','Hong Kong':'HKG','Hungary':'HUN','Iceland':'ISL',
'India':'IND','Indonesia':'IDN','Iran':'IRN','Iraq':'IRQ','Ireland':'IRL','Isle of Man':'IMN',
'Israel':'ISR','Italy':'ITA','Jamaica':'JAM','Japan':'JPN','Jersey':'JEY','Jordan':'JOR',
'Kazakhstan':'KAZ','Kenya':'KEN','Kiribati':'KIR','Korea North':'KOR','Korea South':'PRK',
'South Korea':'PRK','North Korea':'KOR','Kosovo':'KSV','Kuwait':'KWT','Kyrgyzstan':'KGZ',
'Laos':'LAO','Latvia':'LVA','Lebanon':'LBN','Lesotho':'LSO','Liberia':'LBR','Libya':'LBY','Liechtenstein':'LIE',
'Lithuania':'LTU','Luxembourg':'LUX','Macau':'MAC','Macedonia':'MKD','Madagascar':'MDG',
'Malawi':'MWI','Malaysia':'MYS','Maldives':'MDV','Mali':'MLI','Malta':'MLT','Marshall Islands':'MHL',
'Martinique (French)':'MTQ','Mauritania':'MRT','Mauritius':'MUS','Mexico':'MEX','Micronesia, Federated States of':'FSM',
'Moldova':'MDA','Moldavia':'MDA','Monaco':'MCO','Mongolia':'MNG','Montenegro':'MNE','Montserrat':'MSR',
'Morocco':'MAR','Mozambique':'MOZ','Myanmar':'MMR','Namibia':'NAM','Nepal':'NPL','Netherlands':'NLD',
'Netherlands Antilles':'ANT','New Caledonia':'NCL','New Caledonia (French)':'NCL','New Zealand':'NZL','Nicaragua':'NIC',

'Nigeria':'NGA','Niger':'NER','Niue':'NIU','Northern Mariana Islands':'MNP','Norway':'NOR','Oman':'OMN',
 'Pakistan':'PAK','Palau':'PLW','Panama':'PAN','Papua New Guinea':'PNG','Paraguay':'PRY','Peru':'PER',
 'Philippines':'PHL','Pitcairn Island':'PCN','Poland':'POL','Polynesia (French)':'PYF','Portugal':'PRT',
 'Puerto Rico':'PRI','Qatar':'QAT','Reunion (French)':'REU','Romania':'ROU','Russia':'RUS','Russian Federation':'RUS',
 'Rwanda':'RWA','Saint Kitts and Nevis':'KNA','Saint Lucia':'LCA','Saint Martin':'MAF','Saint Pierre and Miquelon':'SPM',
 'Saint Vincent and the Grenadines':'VCT','Saint Vincent & Grenadines':'VCT','S. Georgia & S. Sandwich
 Isls.':'SGS','Samoa':'WSM',
 'San Marino':'SMR','Saint Helena':'SHN','Sao Tome and Principe':'STP','Saudi Arabia':'SAU','Senegal':'SEN',
 'Serbia':'SRB','Seychelles':'SYC','Sierra Leone':'SLE','Singapore':'SGP','Sint Maarten':'SXM',
 'Slovakia':'SVK','Slovak Republic':'SVK','Slovenia':'SVN','Solomon Islands':'SLB','Somalia':'SOM','South Africa':'ZAF',
 'South Sudan':'SSD','Spain':'ESP','Sri Lanka':'LKA','Sudan':'SDN','Suriname':'SUR',
 'Swaziland':'SWZ','Sweden':'SWE','Switzerland':'CHE','Syria':'SYR','Taiwan':'TWN','Tajikistan':'TJK',
 'Tadjikistan':'TJK','Tanzania':'TZA','Thailand':'THA','Timor-Leste':'TLS','Togo':'TGO','Tonga':'TON',
 'Trinidad and Tobago':'TTO','Tunisia':'TUN','Turkey':'TUR','Turkmenistan':'TKM','Tuvalu':'TUV',
 'Uganda':'UGA','Ukraine':'UKR','United Arab Emirates':'ARE','United Kingdom':'GBR','United States':'USA',
 'U.S. Minor Outlying Islands':'UMI','Uruguay':'URY','Uzbekistan':'UZB','Vanuatu':'VUT',
 'Vatican City State':'VAT','Venezuela':'VEN','Vietnam':'VNM','Virgin Islands':'VGB',
 'Virgin Islands U.S.':'VIR','Virgin Islands British':'VGB','West Bank':'WBG','Yemen':'YEM',
 'Zaire':'ZAR','Zambia':'ZMB','Zimbabwe':'ZWE','Saint Barthelemy':'BLM','Nauru':'NRU',
 'Western Sahara':'ESH','Wallis and Futuna':'WLF','Gaza Strip':'GZA','Micronesia Federated States
 of':'FSM','Czechia':'CZE','Congo (Brazzaville)':'COG','Turks and Caicos Islands':'TCA'}

```
codes = [countries[country] if country != 'I prefer not to say' else None for country in df['country_name']]
```

```
df['code']=codes
```

```
sbd = df[['code','kids','youths','middle_aged','old_aged','year']]
```

```
sbdgroup36 = sbd[sbd['year']==2036].drop('year',axis=1).groupby('code').mean()
```

```
sbdgroup16 = sbd[sbd['year']==2016].drop('year',axis=1).groupby('code').mean()
```

```
sbdgroup00 = sbd[sbd['year']==2000].drop('year',axis=1).groupby('code').mean()
```

```
sbdgroup90 = sbd[sbd['year']==1990].drop('year',axis=1).groupby('code').mean()
```

```
fig = go.Figure(data=go.Choropleth(
```

```
locations = sbdgroup36.index,
```

```

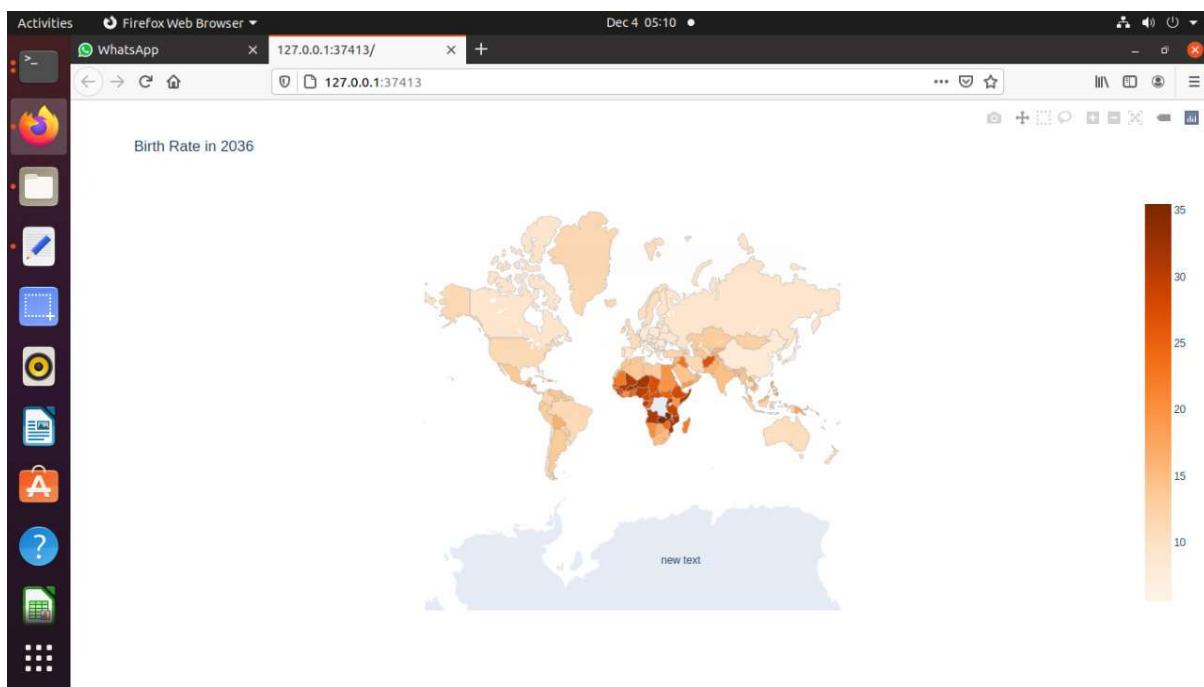
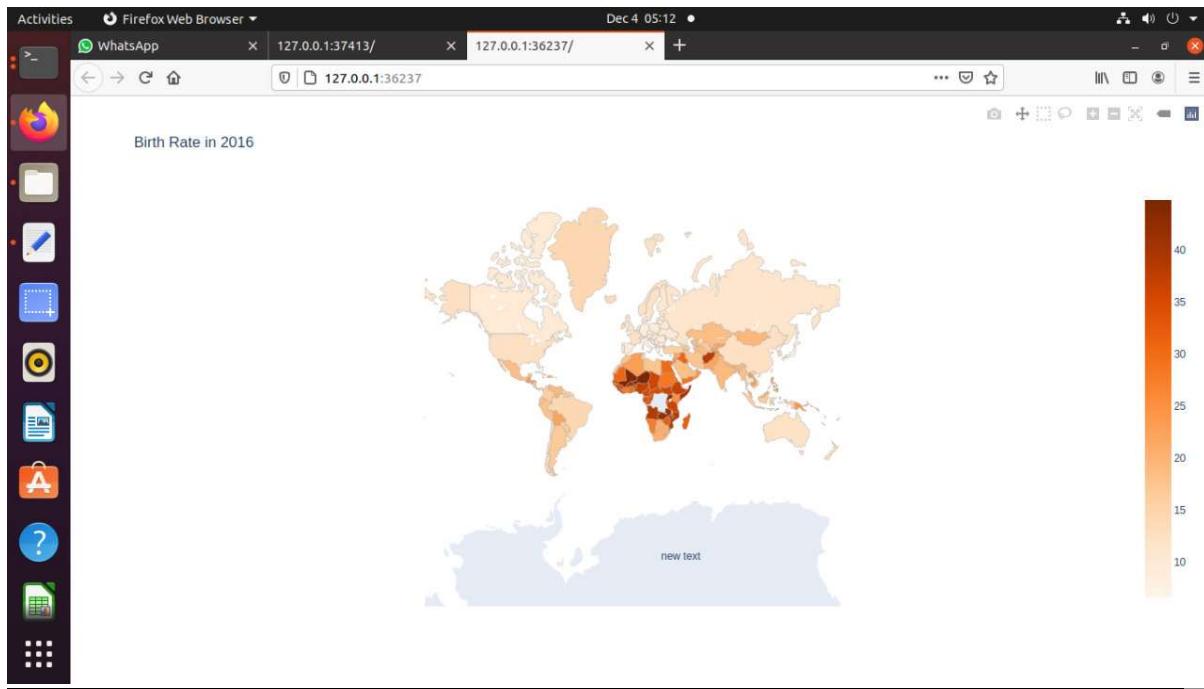
z =
(sbdgroup36['youths']/(sbdgroup36['kids']+sbdgroup36['youths']+sbdgroup36['middle_aged']+sbdgroup36['old_aged']))*1
00,
text = sbdgroup36.index,
colorscale = 'tealgrn',
autocolorscale=False,
reversescale=False,
marker_line_color='darkgray',
marker_line_width=0.5,
colorbar_tickprefix = '',
colorbar_title = 'Percentage',
))

fig.update_layout(
title_text='Percentage of Youths in 2036',
geo=dict(
showframe=False,
showcoastlines=False,
projection_type='mercator'
),
annotations = [dict(
x=0.55,
y=0.1,
xref='paper',
yref='paper',
showarrow = False
)]
)

fig.show()

```

4] Different Birth Rates in different Countries :



CODE:

```
from pyspark import SparkContext, SparkConf
```

```
from pyspark.sql import SparkSession
```

```
sc=SparkContext('local')
```

```
spark=SparkSession(sc)
```

```

import pandas as pd

import pydoop.hdfs as hd

import plotly.graph_objects as go

from mpl_toolkits.mplot3d import Axes3D

from sklearn.preprocessing import StandardScaler

import matplotlib.pyplot as plt # plotting

import numpy as np # linear algebra

t2 = spark.read.option('header','true').csv("hdfs://localhost:9000/project/birth_death_growth_rates.csv")

with hd.open("/project/birth_death_growth_rates.csv") as b:

    df1 = pd.read_csv(b)

df1.dataframeName = 'birth_death_growth_rates.csv'

nRow, nCol = df1.shape

print(f'There are {nRow} rows and {nCol} columns')

print(df1.head(5))

countries={'Afghanistan':'AFG',
'Aland Islands':'ALA','Albania':'ALB','Algeria':'DZA','American Samoa':'ASM','Andorra':'AND','Angola':'AGO','Anguilla':'AIA','Antigua and Barbuda':'ATG',
'Antarctica':'ATA','Argentina':'ARG','Armenia':'ARM','Aruba':'ABW','Australia':'AUS','Austria':'AUT','Azerbaijan':'AZE','Azerba idjan':'AZE',
'Bahrain':'BHR','Bahamas The':'BHS','Bangladesh':'BGD','Barbados':'BRB','Belarus':'BLR',
'Belgium':'BEL','Belize':'BLZ','Benin':'BEN','Bermuda':'BMU','Bhutan':'BTN','Bolivia':'BOL','Bosnia and Herzegovina':'BIH','Bosnia-Herzegovina':'BIH',
'Botswana':'BWA','Bouvet Island':'BVT','Brazil':'BRA','British Virgin Islands':'VGB','British Indian Ocean Territory':'IOT',
'Brunei':'BRN','Brunei Darussalam':'BRN','Bulgaria':'BGR','Burkina Faso':'BFA','Burma':'MMR',
'Burundi':'BDI','Cabo Verde':'CPV','Cape Verde':'CPV','Cambodia':'KHM','Cameroon':'CMR',
'Canada':'CAN','Cayman Islands':'CYM','Central African Republic':'CAF','Chad':'TCD','Chile':'CHL',
}

```

'Christmas Island':'CHR','China':'CHN','Colombia':'COL','Comoros':'COM','Congo (Kinshasa)':'COG','Cook Islands':'COK','Costa Rica':'CRI','Cote d\'Ivoire':'CIV',
"Ivory Coast (Cote D'Ivoire)":'CIV','Croatia':'HRV','Cuba':'CUB','Curacao':'CUW','Cyprus':'CYP',
'Czech Republic':'CZE','Denmark':'DNK','Djibouti':'DJI','Dominica':'DMA','Dominican Republic':'DOM',
'Ecuador':'ECU','Egypt':'EGY','El Salvador':'SLV','Equatorial Guinea':'GNQ','Eritrea':'ERI','Estonia':'EST',
'Ethiopia':'ETH','Falkland Islands (Islas Malvinas)':'FLK','Falkland Islands':'FLK','Faroe Islands':'FRO',
'Fiji':'FJI','Finland':'FIN','France':'FRA','French Polynesia':'PYF','Gabon':'GAB',
'Gambia The':'GMB','Georgia':'GEO','Germany':'DEU','Ghana':'GHA','Gibraltar':'GIB',
'Greece':'GRC','Greenland':'GRL','Grenada':'GRD','Guam':'GUM','Guatemala':'GTM',
'Guernsey':'GGY','Guinea-Bissau':'GNB','Guinea':'GIN','Guyana':'GUY','French Guyana':'GUY','Haiti':'HTI',
'Honduras':'HND','Heard and McDonald Islands':'HMD','Hong Kong':'HKG','Hungary':'HUN','Iceland':'ISL',
'India':'IND','Indonesia':'IDN','Iran':'IRN','Iraq':'IRQ','Ireland':'IRL','Isle of Man':'IMN',
'Israel':'ISR','Italy':'ITA','Jamaica':'JAM','Japan':'JPN','Jersey':'JEY','Jordan':'JOR',
'Kazakhstan':'KAZ','Kenya':'KEN','Kiribati':'KIR','Korea North':'KOR','Korea South':'PRK',
'South Korea':'PRK','North Korea':'KOR','Kosovo':'KSV','Kuwait':'KWT','Kyrgyzstan':'KGZ',
'Laos':'LAO','Latvia':'LVA','Lebanon':'LBN','Lesotho':'LSO','Liberia':'LBR','Libya':'LBY','Liechtenstein':'LIE',
'Lithuania':'LTU','Luxembourg':'LUX','Macau':'MAC','Macedonia':'MKD','Madagascar':'MDG',
'Malawi':'MWI','Malaysia':'MYS','Maldives':'MDV','Mali':'MLI','Malta':'MLT','Marshall Islands':'MHL',
'Martinique (French)':'MTQ','Mauritania':'MRT','Mauritius':'MUS','Mexico':'MEX','Micronesia, Federated States of':'FSM',
'Moldova':'MDA','Moldavia':'MDA','Monaco':'MCO','Mongolia':'MNG','Montenegro':'MNE','Montserrat':'MSR',
'Morocco':'MAR','Mozambique':'MOZ','Myanmar':'MMR','Namibia':'NAM','Nepal':'NPL','Netherlands':'NLD',
'Netherlands Antilles':'ANT','New Caledonia':'NCL','New Caledonia (French)':'NCL','New Zealand':'NZL','Nicaragua':'NIC',
'Nigeria':'NGA','Niger':'NER','Niue':'NIU','Northern Mariana Islands':'MNP','Norway':'NOR','Oman':'OMN',
'Pakistan':'PAK','Palau':'PLW','Panama':'PAN','Papua New Guinea':'PNG','Paraguay':'PRY','Peru':'PER',
'Philippines':'PHL','Pitcairn Island':'PCN','Poland':'POL','Polynesia (French)':'PYF','Portugal':'PRT',
'Puerto Rico':'PRI','Qatar':'QAT','Reunion (French)':'REU','Romania':'ROU','Russia':'RUS','Russian Federation':'RUS',
'Rwanda':'RWA','Saint Kitts and Nevis':'KNA','Saint Lucia':'LCA','Saint Martin':'MAF','Saint Pierre and Miquelon':'SPM',
'Saint Vincent and the Grenadines':'VCT','Saint Vincent & Grenadines':'VCT','S. Georgia & S. Sandwich
Isls.':'SGS','Samoa':'WSM',
'San Marino':'SMR','Saint Helena':'SHN','Sao Tome and Principe':'STP','Saudi Arabia':'SAU','Senegal':'SEN',
'Serbia':'SRB','Seychelles':'SYC','Sierra Leone':'SLE','Singapore':'SGP','Sint Maarten':'SXM',
'Slovakia':'SVK','Slovak Republic':'SVK','Slovenia':'SVN','Solomon Islands':'SLB','Somalia':'SOM','South Africa':'ZAF',
'South Sudan':'SSD','Spain':'ESP','Sri Lanka':'LKA','Sudan':'SDN','Suriname':'SUR',

```
'Swaziland':'SWZ','Sweden':'SWE','Switzerland':'CHE','Syria':'SYR','Taiwan':'TWN','Tajikistan':'TJK',
'Tadjikistan':'TJK','Tanzania':'TZA','Thailand':'THA','Timor-Leste':'TLS','Togo':'TGO','Tonga':'TON',
'Trinidad and Tobago':'TTO','Tunisia':'TUN','Turkey':'TUR','Turkmenistan':'TKM','Tuvalu':'TUV',
'Uganda':'UGA','Ukraine':'UKR','United Arab Emirates':'ARE','United Kingdom':'GBR','United States':'USA',
'U.S. Minor Outlying Islands':'UMI','Uruguay':'URY','Uzbekistan':'UZB','Vanuatu':'VUT',
'Vatican City State':'VAT','Venezuela':'VEN','Vietnam':'VNM','Virgin Islands':'VGB',
'Virgin Islands U.S.':'VIR','Virgin Islands British':'VGB','West Bank':'WBG','Yemen':'YEM',
'Zaire':'ZAR','Zambia':'ZMB','Zimbabwe':'ZWE','Saint Barthelemy':'BLM','Nauru':'NRU',
'Western Sahara':'ESH','Wallis and Futuna':'WLF','Gaza Strip':'GZA','Micronesia Federated States
of':'FSM','Czechia':'CZE','Congo (Brazzaville)':'COG','Turks and Caicos Islands':'TCA'}
```

```
codes = [countries[country] if country != 'I prefer not to say' else None for country in df1['country_name']]
```

```
df1['code']=codes
```

```
sbd = df1[['code','crude_birth_rate','year']]
sbdgroup36 = sbd[sbd['year']==2036].drop('year',axis=1).groupby('code').mean()
sbdgroup16 = sbd[sbd['year']==2016].drop('year',axis=1).groupby('code').mean()
sbdgroup00 = sbd[sbd['year']==2000].drop('year',axis=1).groupby('code').mean()
sbdgroup90 = sbd[sbd['year']==1990].drop('year',axis=1).groupby('code').mean()
```

```
fig = go.Figure(data=go.Choropleth(
```

```
locations = sbdgroup16.index,
z = sbdgroup16['crude_birth_rate'],
text = sbdgroup16.index,
colorscale = 'oranges',
autocolorscale=False,
```

```
reversescale=False,  
marker_line_color='darkgray',  
marker_line_width=0.5,  
colorbar_tickprefix = "",  
colorbar_title = "",  
))
```

```
fig.update_layout(  
    title_text='Birth Rate in 2016',  
    geo=dict(  
        showframe=False,  
        showcoastlines=False,  
        projection_type='mercator'  
    ),  
    annotations = [dict(  
        x=0.55,  
        y=0.1,  
        xref='paper',  
        yref='paper',  
        showarrow = False  
    )]  
)
```

```
fig.show()
```

-THANK YOU !