Alexei Kosyhin

(a.kosyhin@yahoo.com, aleksksn@yahoo.co.uk)

REPORT on number of suicides on country, gender, age group, generation, year and other parameters.

Submitted for Quantum Group as a data analyst home challenge.

Task 1

RESULT/OUTPUT: " **task_1.csv"** in zip folder **REPORT** , I used python code **task_1.py**

0 Albania
1 Antigua and Barbuda
2 Argentina
3 Armenia
4 Aruba
5 Australia
6 Austria
7 Azerbaijan
…
…

All countries are listed in alphabetical order, there are 101 countries.

For data analysis I will use Python and Pandas Data Frame, which is a Python Library.

To list an unique list of countries I took the following steps:

- Downloaded the original file "*who_suicide_statistics_modified3*", placed it in the same folder as a python file, task_1.py,
- Followed some formal procedures and imported panda library into python file: *"import pandas as pd*"
- Downloaded the file "*who_suicide_statistics_modified3*" and saved as a Pandas Data Frame.
- Using Pandas syntax I extracted a list of all countries into data countries, using Pandas method unique I made the countries list unique (every country appears in the list only once)
- I numerate the list using python method "enumerate"
- I plotted the list to be sure it is correct:
  "for num, country in enumerate(df1['country'].unique()):
  print(num, country)"
- I exported the result as a csv file "**task_1.csv**" . For this I used Pandas method:
  task_1.to_csv('task_1.csv')

Task 2

a) Data in columns “*year*”, “*suicides_no*”, “*population*”, “*gdp_for_year($)”* are represented in text/ strings, not numbers/ integers or float. At first instance we need to transfer it in numbers in integers/ l or float python types.

b) In column “*suicides_no*”, there empty spaces or “*NULL*”. Probably some data is missed. I can assume there are no records of suicides, or suicides didn’t happen. For simplicity I decided to replace this data with integer (0).

c) In the column “*age*” the data is represented by text/strings, which is actually no problem. However, I decided to replace data of “ 05-14 years” by “_5-14 years” to able to sort out the data in ascending or descending orders.

Task 3

Readable and 'cleaned' data is represented in file **task_3.csv**

For this task I used python code **task_3.py**

Some numbers in the csv file were presented with 2 or few decimals. I manually changed format of these numbers to numbers with no decimals after a dot. For example, before it was "14.00". So I changed to "14", which actually is the same number, but python code would recognise it as integer.

Then I used python code:

```
# Cleaning data - transferming strings and spaces  into integers
for i in range(0,len(data)):
    data[i][1] = int(data[i][1])
    data[i][5] = int(data[i][5])
    data[i][7] = int(data[i][7])
```

Where 1, 5, 7 are columns "year", "population", "gdp_for_year($)"

To replace strings and missing data in column "suicides_no", which is column 4, I used this code:

```
for i in range(0,len(data)):
    if data[i][4].isdigit() == True:
        data[i][4] = int(data[i][4])
    else:
        data[i][4] = 0
```

I used looping over all data in range of data length.

I replaced the string "5-14 years" by string "_5-14 years" using following code:

```
30  for i in range(0,len(data)):
31      if data[i][3] == '5-14 years':
32          data[i][3] = ' 5-14 years'
```

I exported results to csv file using the code:

```
40  df1.to_csv('task_3.csv')
```

- I renamed the file "*who_suicide_statistics_modified3.csv*" as "who.csv"

Task 4

Results are shown in file **task_4.csv**

	country	year	sex	...	HDI	gdp_for_year	suicide_person
0	Albania	1987	male	...		2156624900	6.711409
1	Albania	1987	male	...		2156624900	5.194805
2	Albania	1987	female	...		2156624900	4.832585
3	Albania	1987	male	...		2156624900	4.587156
4	Albania	1987	male	...		2156624900	3.281079
...	...	...	...	...	...	...	...
27835	Belgium	2011	female	...	0.886	527008453887	0.848015
27836	Thailand	2016	male	...		411755164833	13.522506

Method:

I used python code **task_4.py.**

```
34 for i in range(0,len(data)):
35     data[i].append(data[i][4]*100000/data[i][5])
```

I added a heading “suicide_person” of csv file via pandas data frame.

The name of new column is “suicide_person”

I used Pandas code to visualise/ print data and export to a csv file.

Task 5

Results can be found in file **task_5.csv.** I used python code **task_5.py**

I created a table of generations according their birth year in a csv file **generation.csv**

Then I downloaded this file into python code in the same way as I did for previous data.

Then I added a new column “generation”.

I used two loops and python ifs conditions to fill right data in the column.

```
48 for i in range(0,len(data)):
49     for n in range(0, len(gen)):
50         dob = data[i][1] - int(data[i][3][0:2])
51         if  (dob >= int(gen[n][1]) and dob <=int(gen[n][2])):
52             data[i].append(gen[n][0])
```

I created a new variable/ integer “dob”.

For example, for this in line 50 I sliced a string data[i][3] (for example ’15 – 25 years’[0:2]

and left first 2 characters (for example ’15 – 25 years’[0:2] = ‘15’)

I converted these 2 characters into integer (int(‘15’) = 15))

Then I subtracted 2 integers data[i][1] from int(data[3][0:2]) and obtained an integer.

```
50         dob = data[i][1] - int(data[i][3][0:2])
```

Finally I compared 2 integers in python ifs condition (line 51)

In line 52 I generated / append a new data in column “Generation”

Then using panda’s data frame I printed/ visualised and correctness of results.

Then I exported the results to the csv file.

```
84  # pop is for calculating population and suicide number for the year,
85  pop= year.groupby(['country'],as_index=False)['suicide_no','population'].sum()
86
87  # gdp is for finding a mean gdp_per_year for each year
88  gdp =year.groupby(['country'],as_index=False)['gdp_for_year'].mean()
89
90  #creating a new database which has the same dimension as pop and gdp
91  table =pd.DataFrame(pop)
92
93  # adding a new collumn 'gdp_for_year, $' and filling this column
94  table['gdp_for_year, $'] = gdp['gdp_for_year']
95
96  # adding a new column 'gdp_per_capita_per_year,$' and filling it.
97  table['gdp_per_capita_per_year,$'] = gdp['gdp_for_year']/pop['population']
```

Task 6

Results are shown in file **task_6.csv.** I used python code **task_6.py** This result is for year **2002.** The similar data can be calculated for every yes year by replacing number '2002'

	country	suicide_no	population	gdp_for_year, $	gdp_per_capita_per_year,$
0	Albania	133	2818839	4435078648	1573
1	Antigua and	0	77588	814615333	10499
2	Argentina	3162	34407507	97724004252	2840
3	Armenia	74	3014844	2376335048	788
4	Aruba	9	88284	1941094972	21987
5	Australia	2139	18370058	3.94487E+11	21474
6	Austria	1364	7683162	2.13378E+11	27772
7	Azerbaijan	92	7605700	6235795104	820
8	Bahamas	3	282555	8881160000	31432
9	Bahrain	25	661010	9632155053	14572
10	Barbados	0	253353	3169600000	12511
11	Belarus	2733	9469234	14594249023	1541
12	Belgium	1986	9758570	2.5886E+11	26526

Method:

It's not possible to add gdp_per_capita for the whole data, for every year.
It's because population and gdp for each year is different and gdp_per_capita will be different for each year. So, from this point it's better to look at results for every particular year. It was not possible to keep all columns ('sex', 'generation', etc) in new results.
- I selected the year 2002. The year can be changed by assigning "year":

```
79  # Just selecting a year, for example year = 2002
80  year= 2002
81  year = df1[df1['year']== year]
```

- I used pandas functions as groupby with combination "mean()" and "sum()"
- I assigned a new variable "table" where I generated the obtained results.

```
84  # pop is for calculating population and suicide number for the year,
85  pop= year.groupby(['country'],as_index=False)['suicide_no','population'].sum()
86
87  # gdp is for finding a mean gdp_per_year for each year
88  gdp =year.groupby(['country'],as_index=False)['gdp_for_year'].mean()
89
90  #creating a new database which has the same dimension as pop and gdp
91  table =pd.DataFrame(pop)
92
93  # adding a new collumn 'gdp_for_year, $' and filling this column
94  table['gdp_for_year, $'] = gdp['gdp_for_year']
95
96  # adding a new column 'gdp_per_capita_per_year,$' and filling it.
97  table['gdp_per_capita_per_year,$'] = gdp['gdp_for_year']/pop['population']
```

- I created an option to sort data on its variables, (suicide_no, gdp_per year_, population, etc)

```
99  #sorting data according 'gdp_per_capita' or a required parametor.
100 table.sort_values(by = ['gdp_for_year, $'])
```

Task 7

The result in shown in csv file **task_7.csv** . I used python and pandas code **task_7.py**

country	suicide_no	suicide_rank
Dominica	0	1.5
Saint Kitts and Nevis	0	1.5
San Marino	4	3.0
Antigua and Barbuda	11	4.0
Maldives	20	5.0
...	...	...
Ukraine	238061	97.0
France	240432	98.0
Japan	635785	99.0
United States	759837	100.0
Russian Federation	815965	101.0

Method:

I used a code in Pandas Data Frame.

I calculated total number of suicides for each country using Pandas function "*groupby*"

```
85  # calculating total numbers by country
86  country_rank  = df1.groupby(['country'])['suicide_no'].sum()
```

Finally I ranked the list using pandas function "*rank*"

```
94  # ranking countries according total suicides in ascending order.
95  country_rank['suicide_rank'] = country_rank["suicide_no"].rank(ascending=True)
```

I visualised the result and wrote it into a csv file.

```
97  # printing and writing to a csv file
98  print(country_rank)
99  country_rank.to_csv('task_7.csv')
```

Task 8

Results are shown in files **task_8.csv** , **task_8.png** .I used a python code

task_8.py

This result shows number of suicides for each continent:

"continent"	*"suicide_no"*
Africa	10089
Asia	1150065
Europe	2539928
North America	1004670
Oceania	72379
South America	353165

Method:

- I downloaded csv file "continent_mapping.csv" and placed in the folder with the python code.
- I downloaded data from the file into the python code using the same method/ code as I did it in previous tasks.
- I created a new column "continents" and generated data to this column using the python code:

```
65 for i in range(0,len(data)):
66     for n in range(0, len(continents)):
67         if  data[i][0] == continents[n][0]:
68             data[i].append(continents[n][1])
```

I was working on 2 two dimensional python lists to combine them into 1 two dimensial python list. I used python method to add a new column. I was comparing 2 strings using python ifs condition and looped the data two times.

- I transferred the data file from python to Pandas Data Frame.

```
73 df1= pd.DataFrame(data) # PANDAS DATA FRAME
```

- I printed data to check and be sure it's correct.

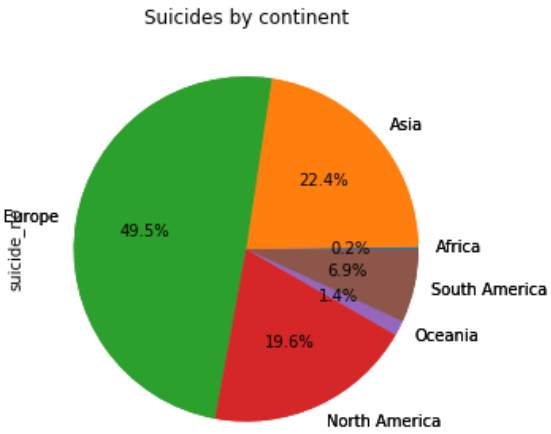
```
79 print (data[72][1] )
```

- I used Pandas function to groupby and sum up. I named the new data "cont"

```
83 cont = df1.groupby(['continent'])['suicide_no'].sum()
```

- When printed the results I obtained the same results shown above.

- I plotted the result into a pie chart:



Task 9

Analysis: on finding correlations between suicides, GBR per capita and population.

For this analysis I will be using obtained in task_6.csv data presented in a table:

	country	suicide_no	population	gdp_for_year, $	gdp_per_capita_per_year,$
0	Albania	133	2,818,839	4,435,078,648	1,573
1	Antigua ar	0	77,588	814,615,333	10,499
2	Argentina	3,162	34,407,507	97,724,004,252	2,840
3	Armenia	74	3,014,844	2,376,335,048	788
4	Aruba	9	88,284	1,941,094,972	21,987
5	Australia	2,139	18,370,058	394,486,709,920	21,474
6	Austria	1,364	7,683,162	213,377,771,504	27,772
7	Azerbaijar	92	7,605,700	6,235,795,104	820
8	Bahamas	3	282,555	8,881,160,000	31,432
9	Bahrain	25	661,010	9,632,155,053	14,572
10	Barbados	0	253,353	3,169,600,000	12,511
11	Belarus	2,733	9,469,234	14,594,249,023	1,541
12	Belgium	1,986	9,758,570	258,860,436,665	26,526
13	Belize	14	224,136	932,551,850	4,161
14	Brazil	7,157	162,761,414	507,962,487,700	3,121
15	Bulgaria	1,218	7,544,410	16,276,456,428	2,157
16	Canada	3,065	29,642,114	757,950,678,647	25,570

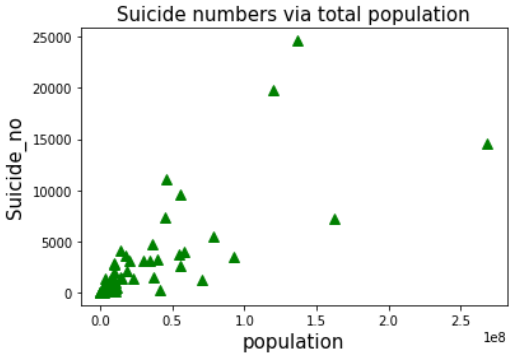
By sorting data in the table we can find a correlation between the parameters.

For example, we can look at Belarus and Belgium

13	11	Belarus	2,733	9,469,234	14,594,249,023	1,541
14	12	Belgium	1,986	9,758,570	258,860,436,665	26,526

These countries have nearly the same population, and the similar numbers of suicides in year 2002, however, GDPs for these countries are very significant. There is a very weak correlation between the GDP and suicide rate.

- Regarding number of suicides and total country population, of course, there is a correlation. It is obvious that the more people the more they commit suicides But suicide rate or suicide number per 100K very different, this is why is not a straight line on the plot below.



Task 10

Results for analysis are stored in **task_10.csv.** I used python code with its module pandas data frame **task_10.py.** Plots are in png files **task_10a.png** and **task_10b.png**

Method:

I used a pandas data frame groupup and sum up functions to calculate total number of suicides over years.

```
79   # groupby and sumup for totla suicide numbers over years
80   suic_year = df1.groupby(['year'],as_index=False)['pop','suicide_no'].sum()
```

It's very important to stress that we need to write this option `,as_index=False` Otherwise key argument 'year' will become index and it won't be possible to use this argument in code again.

Then using matplotlib python library I plotted results and saved in a png picture.

Then I calculated total population over years:

```
93   #calculating total population by year
94   country_pop = df1.groupby(['year','country' ],as_index=False)['pop'].sum()
95   year_pop = country_pop.groupby(['year' ],as_index=False)['pop'].sum()
```

Knowing total population and total suicides per year, I calculated suicide per 100,000 people over years.

```
97   # calculating suicide per 100K of population over years.
98   suic_year['suicide_per_100k'] = suic_year['suicide_no']*100000/year_pop['pop']
```

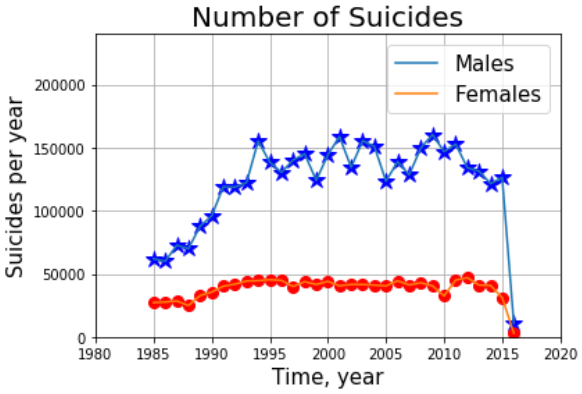
I plotted this data and added to data base column.

Task 11

Analysis

Comparing suicides by gender over years

Results were represented in **task_11.csv, task_11.png, task_11b.png**



Method:

I created a new data base represented in a table using pandas code **task_11.py**

	year	suicide_no	male	female	ratio
0	1985	89009	61894	27115	2.282648
1	1986	87541	59935	27606	2.171086
2	1987	101102	72648	28454	2.553174
3	1988	94701	69876	24825	2.814743
4	1989	120481	87724	32757	2.678023
5	1990	131333	95804	35529	2.696501
6	1991	159158	119049	40109	2.968137
7	1992	160538	118716	41822	2.838602
8	1993	166539	122300	44239	2.764529
9	1994	199798	155259	44539	3.485911
10	1995	183723	138444	45279	3.057576
11	1996	175193	130429	44764	2.913703
12	1997	179995	140080	39915	3.509458

Full data can be viewed in **task_11.csv file.**

This is the code for calculating these new values.

```
df1.columns= ['country', 'year','sex', 'age','suicide_no','pop', 'HDI', 'gdp_for_year','s
# groupby and sumup for totla suicide numbers over years
# calculating total number suicedes for bth males and females by year
suic_sex = df1.groupby(['year'],as_index=False)['suicide_no'].sum()
# selecting data only males
male= df1[df1['sex']== 'male']
#  grouping by year
suic_male = male.groupby(['year'],as_index=False)['suicide_no'].sum()
# selecting data only females
female= df1[df1['sex']== 'female']
#  grouping by yea
suic_female = female.groupby(['year'],as_index=False)['suicide_no'].sum()
# adding a new column 'male' and filling it
suic_sex['male']  = suic_male['suicide_no']
# adding a new column 'female' and filling it
suic_sex['female']  = suic_female['suicide_no']
# visualising data and checking it
print(suic_sex)
```

Then I plotted this data using python module matplotlib:

```
#  visualising data  via printing using matplotlib
plt.plot(suic_sex['year'], suic_sex['male'],   'b*',  markersize =12)
plt.plot(suic_sex['year'], suic_sex['female'],  'ro',  markersize =8)
plt.xlim((1980, 2020))
plt.ylim((0, 240000))
plt.title ('Number of Suicides',fontsize = 20 )
plt.xlabel ('Time, year', fontsize = 15)
plt.ylabel ('Suicides per year', fontsize =15)
plt.plot (suic_sex['year'], suic_sex['male'], label="Males")
plt.plot (suic_sex['year'], suic_sex['female'], label="Females")
plt.legend(loc=1,    prop={'size': 15})
plt.grid(which='major')
plt.show()
```

Then I calculated the ratio of suicides for males to females.

```
# calculating ratio of suicides of males to females
suic_sex['ratio'] = suic_sex['male']/ suic_sex['female']
print(suic_sex)
```

Finally I plotted this ratio.

I wrote the results to a csv file

```
#writing the data to a csv file
suic_sex.to_csv('task_11.csv')
```

Conclusion

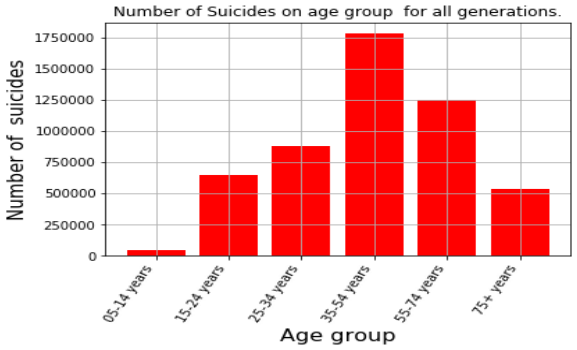
1) Males' suicides for all total countries exceed females' about in 3 times.
2) The ratio of suicides for males to females is slightly increasing over years (from 2.5 times in 1985 to 3.5 times in 2015)
3) Data for 2016 is incomplete and we can't rely on it. It should be ignored for analysis.

Task 12

Results are shown in files **task_12_a.csv, task_12_g.csv, task_12_1.csv, task_2.csv, task_12_a.png, task_12_g.png**

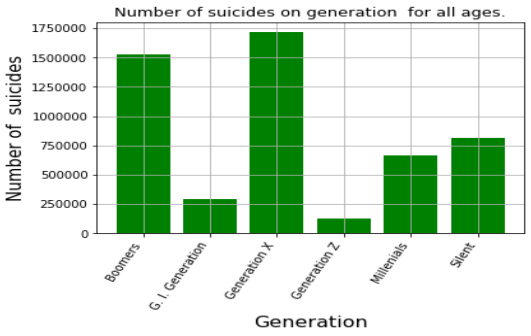
a) Number of suicides on age group for all generations:

	age	suicide_no
0	05-14 years	46107
1	15-24 years	642120
2	25-34 years	879553
3	35-54 years	1780196
4	55-74 years	1245426
5	75+ years	536894



b) Number of suicides on generation for all age groups:

	gen	suicide_no
0	Boomers	1526821
1	G. I. Generation	289294
2	Generation X	1715461
3	Generation Z	123790
4	Millenials	663949
5	Silent	810981



c) Number of suicides on age group for every generation, and on generation for every age group:

age	gen	
05-14 years	Generation X	750
	Generation Z	24450
	Millenials	20907
15-24 years	Generation X	187110
	Generation Z	99340
	Millenials	355670
25-34 years	Boomers	93915
	Generation X	503393
	Millenials	282245
35-54 years	Boomers	750861
	Generation X	1024208
	Millenials	5127
55-74 years	Boomers	682045
	Silent	563381
75+ years	G. I. Generation	289294
	Silent	247600

gen	age	
Boomers	25-34 years	93915
	35-54 years	750861
	55-74 years	682045
G. I. Generation	75+ years	289294
Generation X	05-14 years	750
	15-24 years	187110
	25-34 years	503393
	35-54 years	1024208
Generation Z	05-14 years	24450
	15-24 years	99340
Millenials	05-14 years	20907
	15-24 years	355670
	25-34 years	282245
	35-54 years	5127
Silent	55-74 years	563381
	75+ years	247600

d) Sorted data according number of suicides

where I sorted number of suicides in ascending order.

gen	age	suicide_no
Generation X	05-14 years	750
Millenials	35-54 years	5127
Millenials	05-14 years	20907
Generation Z	05-14 years	24450
Boomers	25-34 years	93915
Generation Z	15-24 years	99340
Generation X	15-24 years	187110
Silent	75+ years	247600
Millenials	25-34 years	282245
G. I. Generation	75+ years	289294
Millenials	15-24 years	355670
Generation X	25-34 years	503393
Silent	55-74 years	563381
Boomers	55-74 years	682045
Boomers	35-54 years	750861
Generation X	35-54 years	1024208

Findings and conclusion

- There are no records on suicides for 'Lost Generation';
- Children aged from 5 to 14 very rarely commit suicides;
- The most records of suicides come from people from Generation X aged of age of 35 to 54;
- It is very important to point that these results show absolute numbers of suicides per strata, not number of suicides by number of population (suicides_per 100K) This results don't show vulnerability people of each strata.

Acknowledgement: I would like to thank Quantum Group for giving me such an interesting problem and motivation to learn panda data frame software.

Date: 3 September 2020