DATA1002-CC-L14-G1 Stage 1 Report

SID and Unikeys

SID: 520 423 045 Unikey:slin7370
SID: 520 102 934 Unikey:pwoo5799
SID: 520 246 478 Unikey:obow2075
SID: 520 463 178 Unikey:sspi3010

Section 1: Identifying the topic

How does a country's GDP affect their education rate, crime rate and life expectancy?

We have collected data regarding a country's GDP, education rate, crime rate, and life expectancy in the years 2005 and 2010. These data contribute to a country's quality of life and we are interested to see if a change in a country's GDP has any effects on the other indicators of quality of life.

Data regarding education is a good indicator of a nation's quality of life as it can be correlated to many other factors such as life expectancy, by providing more access to higher quality healthcare, and also increases in the potential for a country to innovate and develop their economy. Crime rates are correlated with cost of living to society and the idea of safety. Governments may have to allocate their budget differently based on the need to provide security to their citizens. This poses an opportunity cost and can be very costly and affect a country's quality of life. Life expectancy is an important indicator as it can be correlated to many other factors such as increasing living standards, access to high quality healthcare, and overall health and wellbeing of the society. To improve these factors, investments need to be made by the government to support their improvement. One of the main methods for a government to support the development of these indicators is through financial support. As GDP is a commonly used method to determine a country's wealth and financial status, we were interested to see if a change in GDP would affect a country's education rate, crime rate, and life expectancy.

Stakeholders and how our data can be used:

There are various stakeholders that can be impacted by our findings. Major stakeholders include the reigning government body of the country and their departments (department of justice, education etc), and also potential immigrants to the respective country.

The departments of education, health, and justice will be able to more thoroughly understand and become aware of their operation position and effectiveness compared to other countries. They can utilise this knowledge to determine whether there is an issue to be solved and they can use the date to choose countries to refer to in terms of policy, strategy and implementation. The departments can then provide guidance to the reigning government body and also institutions that have the ability to influence and execute decisions.

Potential immigrants are also likely to be interested in the data we find. Majority of immigration occurs to seek a better standard of living. People looking to immigrate can make their decisions based on the data they are presented when researching potential places to immigrate to. Immigration is important for a country as it has potential to increase the size of their labour force, increase productivity and these factors contribute to the wellbeing of an economy and has flow on effects to better standard of living (a larger workforce can lead to higher tax income for a nation and hence more public spending).

Our aggregated dataset can be used by a government to determine the impact of a change in GDP on education rate, crime rates, and also life expectancy. They can use this data to assist them make decisions regarding the wellbeing of the society and also the economy. They can compare the effectiveness of their budget allocation to a country of similar status to determine whether there are inefficiencies or improvements that can be made to improve the socio-economic problems in their country.

Our data can also provide information to international organisations such as the United Nations and the World Bank regarding the current state of certain nations, in particular those that are still developing and these international organisations may be better informed to make decisions regarding the provision of financial aid and support to such couintries.

Education Rate Data (SID: 520423045; Unikey: slin7370)

Metadata, Data Structure and Data Dictionary:

This data set is retrieved from the Barro Lee Educational Attainment Dataset and is composed of a list of countries with data regarding percentages of population in each level of education (primary, secondary, tertiary) and also average years of schooling attained in each level. This dataset uses a percentile scale for the percentage of population and an absolute scale for the average years. The original dataset had 28470 rows and 19 columns (540930 entries) as it contains data from 1950 to 2010. We isolated out the most recent years available in the dataset (2005, 2010)

Data Provenance:

This dataset was obtained from http://www.barrolee.com/, the source of the data collection includes UNESCO, which collected their data from different origins such as:

- Administrative data- Information already available from educational management information systems (EMIS), from the respective department of education
- Household surveys- which is also linked to EMIS
- Population census - census conducted by government of the respective country
- And many more

The Barrolee data collection grants permission to use their collection under https://github.com/barrolee/BarroLeeDataSet/blob/master/LICENSE.

UNESCO and the UN demographic yearbook also grants permission to use their data under https://data.un.org/Host.aspx?Content=UNdataUse, https://datacatalog.worldbank.org/public-licenses#cc-by.

Also according to their website, the UN and UNESCO have declared its data as publicly available under CC-BY 4.0, giving users the right to use their data.

The data dictionary is listed as following and have been accounted for in the cleaning process:

Variable	Description
BLcode	Barro-Lee Country Code
WBcode	Three-letter Country Code
region_code	Region Classification
country	Country Name
year	Year
sex	Sex
agefrom	Starting Age
ageto	Finishing Age
lu	Percentage of No Schooling Attained in Pop.
lp	Percentage of Primary Schooling Attained in Pop.
lpc	Percentage of Complete Primary Schooling Attained in Pop.
ls	Percentage of Secondary Schooling Attained in Pop.
lsc	Percentage of Complete Secondary Schooling Attained in Pop.
lh	Percentage of Tertiary Schooling Attained in Pop.
lhc	Percentage of Complete Tertiary Schooling Attained in Pop.
yr_sch	Average Years of Schooling Attained
yr_sch_pri	Average Years of Primary Schooling Attained
yr_sch_sec	Average Years of Secondary Schooling Attained
yr_sch_ter	Average Years of Tertirary Schooling Attained
pop	Population (1000s)
pop15	Total Population over 15 (1000s)
pop25	Total Population over 25 (1000s)

Source: Barro, Robert and Jong-Wha Lee, 2013, "A New Data Set of Educational Attainment in the World, 1950-2010." Journal of Development Economics, vol 104, pp.184-198.

Weaknesses of Dataset:

- This dataset spans from 1950 to 2010 so the data in the years we have collected may be considered out of date and not applicable to current socio-economic states (COVID, War, etc)
- The original dataset had unclear headings which required the need of a data dictionary and was added to the cleaning process.
- Percentages and years in schooling were unable to determine the quality of schooling received
- Although education years and percentages were recorded, we are unaware and unable to determine how the people who received education applied their knowledge. For example, they could have completed education but went on to do a job that has no correlation with education eg manual labour

Strengths of Dataset:

- The dataset has information for many different years which gives us more evidence if we were to plot trends and extrapolate to future years
- The dataset has various columns which breaks down education attainment into different types (tertiary, secondary, primary) which provides more details that acting bodies can base their decisions on

- The dataset included a large variety of countries from different stages of development and population levels to increase our ability to compare between them

Data Cleaning:

Data cleaning was conducted through Python. Although this dataset from Barrolee is likely to have already been cleaned, it is still in our favour to clean it again to ensure the quality of the data that exists.

```
inputs = open("barrolee.csv", "r")      #opening dataset to read

lines = inputs.readlines()      #extracting date from input file (dataset)
is_first_line = True
cleaned = open("education_cleaned.csv", "w")      #opening output file to write to

for i in lines:
    if is_first_line:
        is_first_line = False
        a = i.replace(",yr_sch,", ",Average Years of Schooling Attained,")
        a = a.replace(",yr_sch_pri,", ",Average Years of Primary Schooling Attained,")
        a = a.replace(",yr_sch_sec,", ",Average Years of Secondary Schooling Attained,")
        a = a.replace(",yr_sch_ter,", ",Average Years of Tertirary Schooling Attained,")
        a = a.replace(",pop,", ",Population (1000s),")
        a = a.replace(",lu,", ",Percentage of No Schooling Attained in Pop,")
        a = a.replace(",lp," , ",Percentage of Primary Schooling Attained in Pop,")
        a = a.replace(",lpc,", ",Percentage of Complete Primary Schooling Attained in Pop,")
        a = a.replace(",ls," , ",Percentage of Secondary Schooling Attained in Pop,")
        a = a.replace(",lsc,", ",Percentage of Complete Secondary Schooling Attained in Pop,")
        a = a.replace(",lh," , ",Percentage of Tertiary Schooling Attained in Pop,")
        a = a.replace(",lhc," , ",Percentage of Complete Tertiary Schooling Attained in Pop,")
        cleaned.write(a)          #Extracting headline and replacing abbreviation with actual meaning

    else:
        try:        #try to convert data to type float, if you can't it implies it is qualitative
            dataSplit = i.split(",")
            for j in dataSplit:
                float(j)
                if float(j) < 0:        #As our numerical data is based on number of years or percentages, if it is less than 0,
                                        #it implies there is an error. We set these values to -1 as we can not get negative years or percentages
                    j = -1
                elif j == "":       #If a position is missing a data value, we set it as 'NULL'
                    j = "NULL"
                if dataSplit[4] == "15" and ("2010" == dataSplit[2] or "2005" == dataSplit[2]) and float(dataSplit[5]) <30 :       #Extracting data from years 2005 and 2010, and for ages under 30
                    cleaned.write(",".join(dataSplit))      #Writing the row to the output file

        except:
            pass

inputs.close()      #Finished cleaning
cleaned.close()     #Finished writing
```

The first part of cleaning was to change the columns from abbreviations to statements that can be easily understood without the need of a data dictionary.

I then split the data by its delimiter “,” and iterated through the values. As the majority of the data was numeric but there was also qualitative data, I implemented a try, except bracket.

If the value was numeric, it would be turned into a float. I then used a conditional to check the value of the float (as percentages and average years can not be less than 0), to see if it was less than 0. If it was less than 0, it implied the data value was wrong and I set it to -1 which is an obviously wrong value.

If the value was non-numeric, an error would not be raised due to the try-except. It would then determine if the value was empty, by testing if it equaled an empty string. If the value was indeed an empty string, i.e. missing, it was set to “NULL”.

Since a list is mutable, we could change the values in the original list ‘dataSplit’. After accounting for wrong numeric values and missing values, I selected the entries in years 2005, and 2010 and for ages less than 30 to be written to my cleaned data file.

Data Analysis:

Code:

```
cleaned = open("education_cleaned.csv", "r")      #Opening output file to extract information regarding entire dataset - This time open as "r"

is_first_line = True

avg_years_2005= []
avg_years_2010= []
avg_years_school = []       #Storing information regarding average years of education in various years and overall

data = cleaned.readlines()
for i in data:
    if is_first_line:
        is_first_line = False
    else:
        info = i.split(",")
        avg_years_school = avg_years_school + [float(info[13])]     #Collecting data regarding average years of schooling (Ages 15-19) and storing into a list
        if info[2] == "2005":
            avg_years_2005 += [float(info[13])]
        elif info[2] == "2010":
            avg_years_2010 += [float(info[13])]



print("Highest average years of schooling attained: " + str(max(avg_years_school)))     #Determining highest average year and lowest and average.
print("Lowest average years of schooling attained: " + str(round(min(avg_years_school),2)))
print("Average of average years of schooling attained: " + str(round(sum(avg_years_school)/len(avg_years_school),2)))        #Average school years - not weighted by population size

print("Highest average years of schooling attained in 2005: " + str(round(max(avg_years_2005),2)))
print("Lowest average years of schooling attained in 2005: " + str(min(avg_years_2005)))
print("Average of average years of schooling attained in 2005: " + str(round(sum(avg_years_2005)/len(avg_years_2005),2)))      #Average school years not weighted by population in 2005

print("Highest average years of schooling attained in 2010: " + str(round(max(avg_years_2010),2)))
print("Lowest average years of schooling attained in 2010: " + str(round(min(avg_years_2010),2)))
print("Average average years of schooling attained in 2010: " + str(round(sum(avg_years_2010)/len(avg_years_2010),2)))     #Average school years not weighted by population in 2010

is_first_line = True
for i in data:
    if is_first_line:
        is_first_line = False

    else:
        info = i.split(",")
        if float(info[13]) == max(avg_years_school):
            print("Country of max average years: " + info[1])

        if float(info[13]) == min(avg_years_school):
            print("Country of min average years: " + info[1])
```

Output:

```
Highest average years of schooling attained: 12.46
Lowest average years of schooling attained: 1.49
Average of average years of schooling attained: 7.92
Highest average years of schooling attained in 2005: 11.85
Lowest average years of schooling attained in 2005: 1.49
Average of average years of schooling attained in 2005: 7.74
Highest average years of schooling attained in 2010: 12.46
Lowest average years of schooling attained in 2010: 2.9
Average average years of schooling attained in 2010: 8.1
Country of min average years: "Mozambique"
Country of max average years: "Republic of Korea"
```

My data analysis was also conducted through python. First I opened the cleaned dataset as “r” instead of “w” this time as I am reading through its contents.

I then made lists for the different years involved in my dataset. Since I was reading the csv file as a text file, I had to convert the value to a float before I appended it to my list (by default it is a string).

I used conditional statements to choose the years I wanted and added that information to the list. For example, if the year was 2005, I added the average years in school for that country to the list for 2005. I also had a list for all of the values regardless of the year.

Then I used inbuilt list functions such as min() and max() to determine the max and min for the respective year and also for the combined of both years. I used list functions to find the mean of all 3 classifications of dates I had by using the sum() function to add up all the values in the list (as I converted them to a float before appending to a list) and then dividing by the length of the list (as the length is equal to the number of entries for that year). I rounded the value to 2d.p before outputting it.

I also iterated through again to find the country with the highest average years in schooling and lowest average years in schooling.

It is important to note that the average of 2005, 2010 and the combined are not weighted to population.

Section 2: Datasets and Metadata

GDP per Capita Data (SID: 520463178, UniKey: sspi3010)

The dataset used to obtain GDP per capita data from years 2005 and 2010 was sourced from the World Bank database. GDP per capita is defined as the total Gross Domestic Product (GDP) divided by the total population of a country. The data used for the group's research was extracted from World Bank national accounts data and OECD National Accounts data files covering a total of 165 countries. The raw dataset contained GDP per capita statistics from 1960 to most recently in 2021, meaning that many columns had to be filtered out to produce the final 'cleaned' dataset. The uncleaned raw dataset had 66 columns and 271 rows, which was brought down to 4 columns and 267 rows after trimming down. As the data is provided by the World Bank, access is free and public under the Creative Commons Attribution 4.0 International License, which states that "You must give appropriate credit, provide a link to the licence, and indicate if changes were made." A positive aspect of the data within this dataset is that the values are to seven significant figures, which provides figures as close to the true value as possible for each country provided. However, a limitation to this dataset is that there are several missing data values within some of its rows and columns, particularly in smaller, lesser-known countries and in earlier years. However, since more recent years were taken (2005 and 2010), the amount of missing values was minimised making the dataset more valid in the latter years of data.

Homicide Rate Data (SID:520102934; Unikey:pwoo5799)

The data is about the homicide rate collected from different countries from 1990 to 2019. In this data, the homicide rate is defined as deaths from interpersonal violence including all ages and both sexes. The rate measures the number of deaths per 100,000 people in a given country or region. The data is published by Global Burden of Disease Collaborative Network, Global Burden of Disease Study 2019 (GBD 2019) Results, and Seattle, United States: Institute for Health Metrics and Evaluation (IHME), 2021. The data publisher's source is from Institute for Health Metrics and Evaluation, Global Burden of Disease (2019). The data was retrieved on September 22nd, 2021 from the source by Roser and Ritchie (2019). I obtained the dataset from Our World in Data website with the title of the dataset as "Deaths - Interpersonal Violence - Sex: Both - Age: All Ages (Rate)". The authors have noted that anyone can use their works freely as long as the authors and the source are cited.

Since I am using a dataset containing homicide rates around the world, it must be noted that it only includes deaths caused by interpersonal violence. If we are to look at crime rates as mentioned in our topic, we would also need to include rates of more common crimes such as theft crimes, sex crimes, or violent crimes. Hence it is limited in some areas to determine how safe the country is. However, one of the strengths of this dataset is that it includes a lot of countries and has the death rates for each year of each country without fail.

The data file is formatted as a CSV file. The data shows the homicide rates of different countries over time. The data has four columns: country name (str), official country

code (str), year(numeric), and death rate from interpersonal violence (numeric). Each country has 20 rows to present the death rate of each year from 1990 to 2019. It has 6841 rows in total and 27364 entries.

Life Expectancy Data (SID: 520246478; Unikey: obow2075)

Life expectancy data was retrieved from The World Bank database (*Life Expectancy at Birth, Total (Years), The World Bank*), a compilation sourcing from, United Nations Population Division , Census reports from national statistical offices, Eurostat: Demographic Statistics United Nations Statistical Division, US Census Bureau, and Secretariat of the Pacific Community. The indicator of interest selected from this dataset was "Life expectancy at birth, total (years)", specifically the number of years a newborn was expected to live given the mortality patterns at its time of birth, and to presume that these were to remain constant throughout its lifetime. Therefore these entries are weighted averages of their respective populations. It must be noted, due to this type of aggregation, threshold type mortality (referring to life expectancy increasing after reaching a certain age) is not represented, and is one limitation of this dataset. Additional limitations include the use of interpolated data from 5-year period data and therefore, measurements recorded on other intervals may be inaccurate.The structure follows: country name (str), country code (str), indicator type (str), indicator code (str), life expectancy (years numeric) from years 1960 to 2020.

Section 3: Ensuring Data Quality

GDP per Capita Data (SID: 520463178, UniKey: sspi3010)

The GDP per capita data was obtained from the World Bank database for years 2005, 2010, which were decided upon by group consensus. This data was sourced from World Bank national accounts data and OECD National Accounts data files covering a total of 165 countries, and was obtained from the following link: https://data.worldbank.org/indicator/NY.GDP.PCAP.CD

The raw dataset contains GDP per capita split into columns from 1960 till 2021 which is measured in current US dollars to adjust for inflation of the currency over time, to 9 significant figures. In addition, the columns include the “Country Name”, “Country Code”, or three letter abbreviations of each country, “Indicator name” given by ‘GDP per capita (current US$)’, and “Indicator Code”, given by the dataset ID ‘NY.GDP.PCAP.CD’. Since only the years 2005 and 2010 were going to be utilised for the group’s research, it was necessary to clean out the unnecessary data using python, as shown in the figure below.

```
import pandas as pd
import numpy as np

f = open("API_NY.GDP.PCAP.CD_DS2_en_csv_v2_4473274.csv")
dataset = f.read()
dataset = dataset.split("\n")
dataset = dataset[4:]
dat=[]
for row in dataset:
    row = row.replace("\"", "")
    dat.append(row.split(","))
df = pd.DataFrame(data = dat[1:], columns= dat[0] + ["1"])
df = df.filter(["Country Name", "Country Code", "2005", "2010"])
df = df.drop(index=266)
df = df.replace("", np.NAN)
df = df.rename(columns={"2005": "GDP_2005", "2010":"GDP_2010"})

df.to_csv("GDP_clean.csv", index= False)
```

Lines 1-2: The “import pandas” section of the code signals Python to bring the pandas data analysis library into the current environment. The “as pd” part allocates “pandas” as an abbreviation “pd”. “Import numpy” does the same action as the previous import chunk, telling Python to bring the NumPy library into the current environment. “As np” gives a shortened alias of NumPy.
Line 4: This line is assigning a variable “f” to open the raw excel data into Python as a csv file.
Line 5: This line reads the raw data’s contents with the “f.read()” function while assigning this as a variable “dataset”.
Line 6: The variable “dataset” is by newline character using the “string(“\n”)” function, which returns a list of strings resulting from splitting the original string at the newline “\n”.

Line 7: This step [4:] takes the dataset from rows 4 onwards, as rows 1-3 were filler and unnecessary to the actual dataset
Line 8: “dat” is assigned as the variable to the list produced from [].
Line 9: This command is used to repeat a sequence a set amount of times, which in this case is “row”.
Line 10: This line assigns row as a variable and replaces all occurrences of “\”” with substring “”.
Line 11: The “append” function adds an item at the end of the “dat” list, splitting the rows by a comma (,) in the outermost brackets
Line 12: This command assigns “df” as an abbreviation for the DataFrame function which produces a labelled data structure with columns. Inside the brackets the dat command records the data from 1 onwards of the new dataset and is assigned to the variable “data”. The columns are also split starting from index 0 to signal the left-most column with an increment of “1”.
Line 13: In this line the filter command is used on “df” to only keep “Country Name”, “Country Code”, “2005” and “2010” columns.
Line 14: This line drops the index 266 or value in row 266 which contains a missing value.
Line 15: The command “replace” which replaces the substring “” with “np.NAN” which represents entries that are undefined.
Line 16: This line uses the command “rename” to change the names of the “2005” and “2010” columns to “GDP_2005” and “GDP_2010”, respectively.
Line 18: This line converts “df”, or the dataframe, to a csv file naming it “GDP_clean.csv” and “index= False” removes the row numbers in the first column

Homicide Rate Data (SID:520102934; Unikey:pwoo5799)

For this dataset, the only missing data it had was country codes for some rows. Since the missing country codes were for countries from the United Kingdom, there was no official country code assigned to these countries. Hence, I put -1 in those missing country codes. There also didn't seem to be outliers as the lowest value and the highest value were believable for their respective countries.

Since we were only looking at data from 2005 and 2010, I filtered the dataset using python to only include rows from 2005 and 2010. Also, my dataset included regions and our group decided to only include countries so I removed the rows representing regions. To make sure I didn't accidentally alter the dataset, I wrote a new file containing rows from 2005 and 2010, with -1 in place of missing data for country codes.

```python
first_line = True
f = open("homicide-rate-cleaned.csv", "w")

for line in open("homicide-rate.csv"):
  #  line = line.rstrip()
  #  line = line.split(",")
    if first_line == True:
        line = line.rstrip()
        line += ",2005-1010 crime rate ratio\n"
        f.write(line)
        first_line = False
    else:
        line = line.rstrip()
        line = line.split(",")

        # get rid of regions
        if line[0] == "African Region (WHO)":
            line = line
        elif line[0] == "East Asia & Pacific (WB)":
            line = line
        elif line[0] == "Eastern Mediterranean Region (WHO)":
            line = line
        elif line[0] == "Europe & Central Asia (WB)":
            line = line
        elif line[0] == "European Region (WHO)":
            line = line
        elif line[0] == "G20":
            line = line
        elif line[0] == "Latin America & Caribbean (WB)":
            line = line
        elif line[0] == "Middle East & North Africa (WB)":
            line = line
        elif line[0] == "North America (WB)":
            line = line
        elif line[0] == "OECD Countries":
            line = line
        elif line[0] == "Region of the Americas (WHO)":
            line = line
        elif line[0] == "South Asia (WB)":
            line = line
        elif line[0] == "South-East Asia Region (WHO)":
            line = line
        elif line[0] == "Sub-Saharan Africa (WB)":
            line = line
        elif line[0] == "Western Pacific Region (WHO)":
            line = line
        elif line[0] == "World":
            line = line
        elif line[0] == "World Bank High Income":
            line = line
        elif line[0] == "World Bank Low Income":
            line = line
        elif line[0] == "World Bank Lower Middle Income":
            line = line
        elif line[0] == "World Bank Upper Middle Income":
            line = line

        else:
            # get 2005 , 2010
            # getting the ratio
            if line[2] == "2005":
                rate05 = float(line[3])
                line.append("NA")
            elif line[2] == "2010":
                rate10 = float(line[3])
                rate = rate10 / rate05
                rate = str(rate)
                line.append(rate)

            # writing it in new file
            if line[2] == "2005":
                if line[1] == "":
                    line[1] = "-1"
                line = ",".join(line)
                line += "\n"
                f.write(line)
            elif line[2] == "2010":
                if line[1] == "":
                    line[1] = "-1"
                line = ",".join(line)
                line += "\n"
                f.write(line)


f.close()
```

Life Expectancy Data (SID: 520246478; Unikey: obow2075)

The data was cleaned using the python Pandas package. After it was read in the data was separated by line to clear the heading of the data. Before the data was entered into a dataframe there were some issues in the formatting of the column “Life expectancy at birth, total (years)”, as it contained a comma. These entries were then replaced with a temporary variable “tmp” as they could be discarded from the dataset. The next rows are then split by comma and fed into a pandas dataframe, with column headings being the csv column headings. Then the data frame was filtered to extract the rows of interest, those being Country Name, Country Code, 2005, and 2010 (for life expectancy). Columns 2005 and 2010 were then named LE_2005 and LE_2010 to preserve meaning when the datasets were combined. Lastly all empty cells were replaced with the Numpy nan value and the data frame was exported to csv.

```python
import pandas as pd
import numpy as np

f = open("API_SP.DYN.LE00.IN_DS2_en_csv_v2_4330149.csv")
dat = f.read()
dat = dat.split("\n")[4:]
x=0
while x < len(dat):
    dat[x] = dat[x].replace("\"Life expectancy at birth, total (years)\"", "tmp")
    dat[x] = dat[x].replace("\"", "")
    x+=1
dat = [x.split(",") for x in dat]

df = pd.DataFrame(data = dat[1:], columns = dat[0]+["placehold"])
df = df.filter(["Country Name", "Country Code", "2005", "2010"], axis=1)
df = df.rename(columns={"2005": "LE_2005", "2010": "LE_2010"})
df = df.drop(index =266)
df = df.replace("", np.nan, regex=True)

df.to_csv("LE_clean.csv", index = False)
```

Summaries for this dataset were created using excel functions (MIN, AVERAGE, MAX) on this cleaned output file:

Section 4: Data Summaries

GDP per Capita Data Summary (SID: 520463178, UniKey: sspi3010)

Year	Min	Av	Max
2005	151.6815663	11518.40784	124197.2754
2010	234.2355389	14671.81423	150737.8925

The data summary for GDP per Capita data contained the minimum, average and maximum values for each of the columns for 2005 and 2010 data. These values were found using the =MIN, =AVERAGE and =MAX commands and highlighting the required columns. A trend present in the years given above is that each values, the minimum, average and maximum values of GDP per capita have all increased in the 5 year timeframe, the minimum increasing by approximately 35.2% (3 s.f.), the average increasing by approximately 21.5% (3 s.f.) and the maximum increasing by around 17.6% (3 s.f.).

Homicide Rate Data (SID:520102934; Unikey:pwoo5799)

In the new dataset formed from writing rows only including “2005” and “2010”, I added another column representing the ratio of homicide rate in 2010 to homicide rate in 2005 for each country. I put “NA” for rows with 2005 and added the ratio value to rows with 2010.

Homicide_Data_Summary

Category	Country	Homicide-Rate-Per-100000-People
Average-Homicide-Rate-In-2005		7.99
Average-Homicide-Rate-In-2010		7.76
Maximum-Homicide-Rate-In-2005	Guatemala	54.88
Minimum-Homicide-Rate-In-2005	San Marino	0.5
Maximum-Homicide-Rate-In-2010	El Salvador	53.13
Minimum-Homicide-Rate-In-2010	San Marino	0.49

I also created another CSV file to show the summary to my dataset. In the file, I put rows presenting: average homicide rate in 2005, average homicide rate in 2010, maximum homicide rate in 2005, minimum homicide rate in 2005, maximum homicide rate in 2010, and minimum homicide rate in 2010. I also put a column to show the countries with the maximum and minimum rates and another column answering the categories from each row.

From looking at the summary, we can see that overall, the homicide rate has gone down for all average, maximum, and minimum. San Marino continued to have the lowest

homicide rate of 0.49 per 100,000 people in 2010 and El Salvador became the country with the highest homicide rate in 2010.

Life Expectancy Summaries (SID: 520246478; Unikey: obow2075)

Year	Min	Av	Max
2005	42.52	68.13	81.93
2010	45.10	70.06	82.84

Life Expectancy summaries were generated using the cleaned csv file in excel, creating a min, max, and average across all countries for both the years of interest: 2005, 2010. As displayed in this table, all values have increased over the 5 year period, with minimum showing the largest increase, average showing the second largest increase and, maximum increasing the least.

Integration Section

Integration was also carried out using the python pandas library. All cleaned datasets were read in; however education and homicide datasets used a year column in a longer data format, where life expectancy and gdp datasets contained a wider format. As the education and homicide datasets contained significantly more values, it was decided that gdp and life expectancy would be changed to this format during integration. This was done for each dataset individually, where entries were moved to a new data frame and the year variable would be introduced depending on which column the value was taken from e.g. 2005 column would have a new value in the year column (2005). This was carried out for both life expectancy and gdp without issue, and datasets were merged with pandas, using an inner join and using Country Code as the index. Finally, some of the column labels were transformed into their long form to provide ample information when analysing the final document.

Importing and Reformat of GDP and Life Expectancy Datasets (cleaned)

```python
import pandas as pd
import numpy as np

homocide = pd.read_csv("homicide-rate-cleaned.csv")
gdp = pd.read_csv("GDP_clean.csv")
education = pd.read_csv("education_cleaned.csv")
lifeExp= pd.read_csv("LE_clean.csv")

new_df= pd.DataFrame(columns=["Country Name", "Country Code", "Year", "GDP"])
n=0
for index, row  in gdp.iterrows():
    new_df.loc[n]=[row["Country Name"], row["Country Code"], 2005, row["GDP_2005"]]
    n+=1
    new_df.loc[n]=[row["Country Name"], row["Country Code"], 2010, row["GDP_2010"]]
    n+=1
gdp = new_df

new_df= pd.DataFrame(columns=["Country Name", "Country Code", "Year", "Life Expectancy"])
n=0
for index, row  in lifeExp.iterrows():
    new_df.loc[n]=[row["Country Name"], row["Country Code"], 2005, row["LE_2005"]]
    n+=1
    new_df.loc[n]=[row["Country Name"], row["Country Code"], 2010, row["LE_2010"]]
    n+=1
lifeExp = new_df

```

Dataset Merging and Export to csv

```python
df = pd.merge(lifeExp, gdp,
    how="inner",
    left_on=["Country Name", "Country Code", "Year"],
    right_on=["Country Name", "Country Code", "Year"])

df = pd.merge(df, homocide,
    how="inner",
    left_on=["Country Name", "Country Code", "Year"],
    right_on=["Entity", "Code", "Year"]).drop(["Entity", "Code"], axis=1)

df = pd.merge(df, education,
    how="inner",
    left_on=["Country Name", "Country Code", "Year"],
    right_on=["country", "WBcode", "year"]).drop(["country", "WBcode", "year"], axis=1)


df = df.rename(columns={"GDP": "GDP/capita (USD)",
    "Life Expectancy": "Life Expectancy (Years)",
    "2005-2010 crime rate ratio":"5 year crimte rate increase (ratio)",
    "Deaths - Interpersonal violence - Sex: Both - Age: All Ages (Rate)":"Deaths - Interpersonal violence - Sex:
df.to_csv("intergrated.csv", index = False)
```

References

Life expectancy at birth, total (years), The World Bank, Retrieved September 9, 2022, from: Life expectancy at birth, total (years) | Data

Roser, M., & Ritchie, H. (2019). *Deaths - Interpersonal Violence - Sex: Both - Age: All Ages (Rate)* [Data Set]. Our World in Data. https://ourworldindata.org/homicides