2. Prepare datasets

In this phase, I will navigate a few tools to clean and prepare the dataset for analysis.

1 CLEANING DATASETS

The data cleaning process will be conducted using Python and the Pandas library. This approach will enable us to perform various tasks, including the removal of unnecessary columns, the renaming of columns, the rounding of numerical values, and the sorting of data based on specific columns. Additionally, we will export the cleaned files for further analysis and use.

1.1 AVERAGE AGE WHEN WOMEN MARRY BY COUNTRY IN THE LAST 20 YEARS.

This dataset examines the average age at which women enter into marriage. The analysis process involves importing a CSV file, renaming several columns for clarity, filtering the data to include only records from the year 2000 onward, rounding the ages to the nearest whole number, and sorting the information in ascending order by year. The final results will be exported to an Excel file for further review and utilization.

```
1. import pandas as pd
2.
3.
4. #Load dataset

5.
6.  df  =  pd.read_csv('C:/Users/Admin/Desktop/GOOGLE  DATA  CERTIFICATE/Capstone/database_europe
birthrate/DATABASE/avg age woman marriage/age-at-marriage-women.csv')
7.
8. #Renamed some columns
9.
10.  df.rename(columns={'Estimated  average  age  at  marriage,  women':  'Women  Avg  Married
Age','Entity':'Country'}, inplace=True)
11.
12. #Creating new df to keep just the values where year >=2000
13.
14. df_filtered = df[df['Year'] >=2000]
15.
16. #Round the ages to a whole number
17.
18. df_filtered['Women Avg Married Age'] = df_filtered['Women Avg Married Age'].round()
19.
20. #Sort by year asc
21.
22. df_sorted_by_year = df_filtered.sort_values(by=['Year','Country'], ascending=[True,True])
23.
24. print(df.head())
25.
26. #Export to an Excel on a specific path
27.   path_to   =   'C:/Users/Admin/Desktop/GOOGLE   DATA   CERTIFICATE/Capstone/database_europe
birthrate/DATABASE/avg age woman marriage/Women Avg Married Age.xlsx'
28. df_sorted_by_year.to_excel(path_to, index=False)
```

1.2 BIRTHRATE AROUND THE WORLD BETWEEN 2000-2022

This dataset provides an opportunity to examine global birth rates from the year 2000 to 2022. The analysis involves importing the relevant CSV file, eliminating unnecessary columns, filtering the data to retain only those entries from the year 2000 onwards, rounding the values in the year columns, and subsequently exporting the refined data to an Excel file.

```
 1. import pandas as pd
 2. import os
 3.
 4. # Load the dataset
 5. file_path = r'C:\Users\Admin\Desktop\GOOGLE DATA
CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\birthrate\birthrate per
1k_eurostat.csv'
 6. df = pd.read_csv(file_path)
 7.
 8. # Drop 'Indicator Name' and 'Indicator Code' columns
 9. columns_to_drop = ['Indicator Name', 'Indicator Code','2023']
10. df.drop(columns=columns_to_drop, inplace=True)
11.
12.
13. # Filter columns to keep only 'Country Name' and years >= 2000
14. columns_to_keep = ['Country Name'] +['Country Code']+ [col for col in df.columns if
col.isdigit() and int(col) >= 2000]
15. df_filtered = df[columns_to_keep]
16.
17. # Round the values in year columns
18. year_columns = [col for col in df_filtered.columns if col.isdigit()]
19. df_filtered[year_columns] = df_filtered[year_columns].round(2)
20.
21. # Define the output path
22. output_path = r'C:\Users\Admin\Desktop\GOOGLE DATA
CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\birthrate
\Birthrate_clean_TO_USE.xlsx'
23.
24. # Save the cleaned dataset to a new file
25. df_filtered.to_excel(output_path, index=False)
```

1.3 DIVORCE RATE DATASET

For this dataset, I have selected Jupyter Notebook as the primary analytical tool. I established a variable for the file path and proceeded to drop and split several columns. Additionally, I reordered and sorted the data while removing any rows that were not pertinent to the analysis.

```
For this we explore use a Jupyter Notebook to the process # %% [markdown]
# '''
# Divorce rate on europe per every 100 marriages
# '''

# %%
#Import Library
import pandas as pd

# %%
#Create a Varible for the file path
file_path= r'C:\Users\Admin\Desktop\GOOGLE DATA
CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe
birthrate\DATABASE\divorce\divorce_per_100Marriage.csv'

# %%
#Reading the file
df = pd.read_csv(file_path)

# %%
#Drop columns we dont need
drop_cols=['DATAFLOW','LAST UPDATE','freq','indic_de','OBS_FLAG']
df= df.drop(columns=drop_cols,errors='ignore')

#Rename The columns
df.rename(columns={'geo':'Country','TIME_PERIOD':'Year','OBS_VALUE':'Divorce per 100
Marriages'},inplace=True)
df.head()

# %%
#Split the Country Column on Country Code and Country Name
df[['Country Code', 'Country Name']] = df['Country'].str.split(':', expand=True)
df.head()

# %%
#Drop old country column
df = df.drop(columns=['Country'], errors='ignore')
df.head()

# %%
#Reorder the columns and sorted by year
Column_order=['Country Name','Country Code','Year','Divorce per 100 Marriages']
df= df[Column_order]

df.head()

# %%
#Sort data set by year
df= df.sort_values(by=['Year','Country Name'], ascending=[True,True])
df.head(20)

# %%
#Remove the row where Country Code contain at least one number,
#this because this columns are totals that we dont need
df = df[~df['Country Code'].str.contains(r'\d', na=False)]

# %%
# Remove rows where 'Country Code' contains 'DE_TOT'
df = df[~df['Country Code'].str.contains('DE_TOT', na=False)]

```

```
df.head(50)

# %%
#Export to an Excel
output_path= r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\divorce\Divorce_Per_100_Marriage-CLEAN.xlsx'
df.to_excel(output_path,index=False)

```

1.4 EDUCATION

This dataset focuses on the analysis of women with tertiary education, utilizing Jupyter Notebook as the primary tool. The data preparation process included actions such as the removal of specific columns, the renaming of columns, as well as the reordering and sorting of the dataset to enhance clarity and accessibility.

```
# %%
'''
IN THIS DATASET WE WILL EXPLORE THE AVG OF WOMEN WITH TERTIARY EDUCATION
'''

# %%
import pandas as pd

# %%
df=pd.read_csv(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\education\BL_v3_F1564.csv')

# %%
#Drop columns we dont need
#
drop_cols=['BLcode','agefrom','agefrom','sex', 'ageto', 'lu', 'lp','lpc', 'ls', 'lsc','yr_sch', 'yr_sch_pri', 'yr_sch_sec',
        'yr_sch_ter']
df= df.drop(columns=drop_cols,errors='ignore')
df.columns

# %%
#Rename The columns
#lh  -Percentage of Tertiary Schooling Attained in Pop.
#lhc -Percentage of Complete Tertiary Schooling Attained in Pop.

df.rename(columns={'country':'Country','year':'Year','Tertiary Schooling Attaine':'Tertiary Schooling Attained','lhc':'Complete Tertiary Schooling','WBcode':'Country Code','region_code':'Region Code', 'Population':'Population(1,000)'},inplace=True)
df.head()

# %%
Column_order=['Country','Country Code','Region Code','Year','Tertiary Schooling Attained','Complete Tertiary Schooling','Population(1,000)']
df= df[Column_order]
df.head()

# %%
#Sort data set by year
df= df.sort_values(by=['Year','Country'], ascending=[True,True])

# %%
#Export to an Excel
output_path= r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\education\Women_with_Tertiary_Education_CLEAN.xlsx'
df.to_excel(output_path,index=False)
```

1.5 POPULATION (URBAN & RURAL)

The population database comprises three datasets: total population, urban population, and rural population. Given that the structure of these datasets is identical, this script is applicable to all three.

```
# %%
import pandas as pd

# %%
df= pd.read_csv(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\europe population\european-union-TOTAL-population.csv')
df.head(3)

# %%
#Drop Indicator Name & Indicator Code columns
drop_cols=['Indicator Code',]
df = df.drop(columns=drop_cols)
df.head(3)

# %%
#Drop all columns where year < 2000's
columns_to_keep = ['Country Name'] + ['Indicator Name'] + ['Country Code'] + [col for col in df.columns if col.isdigit() and int(col) >= 2000]
df = df[columns_to_keep]
df.head(3)

# %%
#Reorder the columns
Col_order =['Country Name','Country Code','Indicator Name']+[col for col in df.columns if col.isdigit()]
df = df[Col_order]

# %%
#Million format to the year columns
years_col=[col for col in df.columns if col.isdigit()]
df[years_col]= df[years_col].applymap(lambda x: f"{x:,.0f}" if pd.notnull(x) else x)
df.head(3)

# %%
#Export to an Excel
output_path= r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\europe population\TOTALPopulation_CLEAN.xlsx'
df.to_excel(output_path,index=False)
```

1.6 FERTILITY RATE

This dataset presents the average fertility rate by country on an annual basis. Demographers utilize the “total fertility rate” to assess the average number of children a woman is expected to have during her reproductive years. To analyze this data, we import a CSV file, rename certain columns, round the fertility rates for accuracy, and subsequently export the refined information to an Excel file.

```
# %% [markdown]
# On this part we go to clean the dataset about Fertility Rate

# %%
import pandas as pd

# %%
df= pd.read_csv(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\fertility rate\birth per year\children-per-woman-un.csv')
df.head(1)

# %% [markdown]
# Rename fertility_rate__sex_all__age_all__variant_estimates column
#

# %%
df.rename(columns={'fertility_rate__sex_all__age_all__variant_estimates':'Fertility_Rate'}, inplace=True)
df.head(1)

# %% [markdown]
# Round the Fertility Rate

# %%
df=df.round({'Fertility_Rate':2})
df.head(1)

# %%
#Export to excel
output_path= r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\fertility rate\Fertility_Rate_Clean.xlsx'
df.to_excel(output_path,index=False)
```

1.7 GDP AROUND THE WORLD

This dataset pertains to the cleaning and formatting of worldwide GDP values. The process commences with the reading of a CSV file that contains GDP information for various countries across multiple years. The code subsequently selects pertinent columns, including country names, country codes, and specific years' data (presumably from the year 2000 onward). Following this, the GDP values are rounded to two decimal places, and a comma-separated format is applied to enhance readability. Ultimately, the cleaned and formatted data is exported to an Excel file.

```
# %% [markdown]
# lets check the GDP worldwide

# %%
import pandas as pd

# %%
#Read CSV file

df= pd.read_csv(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\GDP around theworld\GDP.csv')
df.head(1)

# %%
#Filtering the columns need it

df = df.filter(items=['Country Name', 'Country Code']).join(df.filter(regex=r'^20[0-9]{2}$'))
df.head(5)


# %%
# Identify columns with year values (2000 and onwards)
year_columns = df.columns[df.columns.str.match(r'^20[0-9]{2}$')]

# Round the GDP values in those columns
df[year_columns] = df[year_columns].round(2)

# Apply formatting to add commas
df[year_columns] = df[year_columns].applymap(lambda x: f"{x:,.2f}")

# Display the first 5 rows to verify
df.head(5)

# %%
#Export to excel
output_path= r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\GDP around theworld\GDP_Per_Capita_CLEAN.xlsx'
df= df.to_excel(output_path)

# %%
```

1.8 HEALTHCARE

This dataset focuses on government investments in healthcare as a percentage of their Gross Domestic Product (GDP). The associated script begins by reading a CSV file that contains the raw data. Subsequently, it undertakes several data cleaning and preparation procedures, which include renaming columns to enhance readability and removing any extraneous columns, such as "Unnamed: 4." Ultimately, the refined dataset is exported to an Excel file for further analysis and dissemination.

```
# %% [markdown]
# Government health expenditure as a share of GDP

# %%
import pandas as pd

# %%
df = pd.read_csv(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\HEALTHCARE\public-health-expenditure-share-gdp.csv')
df.head(2)


# %%
df.rename(columns={"public_health_expenditure_pc_gdp": "Health Expenditure",'Entity':'Country'}, inplace=True)
df.head(2)

# %%
#Drop column Unnamed: 4
df.drop(columns=['Unnamed: 4'],inplace=True)
df.head(2)



# %%
#Export to excel
Output_Path=r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\HEALTHCARE\health_expenditure_by_goverments.xlsx'
df=df.to_excel(Output_Path)

```

1.9 MORTALITY RATE

This dataset examines the average child mortality rate for individuals under five years of age across various regions globally. The analysis commences with the ingestion of a CSV file containing the raw data. Following this, the code executes data cleaning and preparation procedures, which include renaming the column titled "under_five_mortality_selected" to "under_five_mortality" for enhanced clarity. Furthermore, the under-five mortality values are rounded to two decimal places to improve the presentation. Ultimately, the cleaned and prepared dataset is exported to an Excel file for further analysis and reporting.

```
# %% [markdown]
# Mortality under 5 years old

# %%
import pandas as pd

# %%
df=pd.read_csv(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\mortality rate\child-mortality.csv')
df.head(1)

# %%
#Rename column under_five_mortality_selected
df.rename(columns={"under_five_mortality_selected": "under_five_mortality"}, inplace=True)
df.head(2)

# %%
#Round under five mortality column
df['under_five_mortality']=df['under_five_mortality'].round(2)
df.head(3)

# %%
#Export to excel
Output_Path=r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\mortality rate\Mortality_under_5yo.xlsx'
df.to_excel(Output_Path)

```

1.10 UNEMPLOYMENT

This dateset brings the average of people of labor force stage that are unemployment. The code snippet cleans a dataset related to unemployment rates in the European Union. It begins by reading a CSV file, explicitly specifying that the first row contains the column headers. The code then identifies and drops any columns with names containing "Unnamed," which typically indicates columns without proper headers. Subsequently, it filters the DataFrame to include only columns representing years from 2000 onwards. Finally, the cleaned dataset is exported to an Excel file.

```
# %% [markdown]
# Lets try to clean this dataset droping the columns without headers, and filtering the >=2000's

# %%
import pandas as pd

# %%
df=pd.read_csv(r"C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\unemplyment\unioneurope_unem.csv", header=0)
df.head(2)

# %%
#DROP all columns without headers
df.drop(columns=[col for col in df.columns if 'Unnamed' in col], inplace=True)
df.head(2)

# %%
#Drop all columns where year < 2000's
columns_to_keep = ['Country'] + [col for col in df.columns if col.isdigit() and int(col) >= 2000]
df = df[columns_to_keep]
df.head(3)

# %%
Output_Path=(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\unemplyment\Unemployment_CLEAN.xlsx')
df.to_excel(Output_Path)

```

1.11 INFLATION RATE

This dataset provides a comprehensive historical overview of the inflation rates of countries worldwide. The process begins with the reading of a CSV file, followed by the removal of the first row, which is identified as empty. The column titled "Inflation rate, average consumer prices (Annual percent change)" is subsequently renamed to "Inflation Rate" to enhance readability. The DataFrame is then filtered to retain only the data for the years spanning from 2000 to 2023. Following this, all instances of "no data" are replaced with empty strings to ensure data consistency. Finally, the cleaned dataset is exported to an Excel file.

```
# %% [markdown]
# Let's clean the inflation rate dataset
# average consumer prices Annual percent change

# %%
#Import Library
import pandas as pd

# %%
#Import CSV
df= pd.read_csv(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\europe inflation\Inflation_rates_worldwide.csv')
df.head(2)

# %%
#Drop first row because is empty
df.drop(index=0, inplace=True)
df.head(2)

# %%
#Rename Inflation rate, average consumer prices (Annual percent change) column
df.rename(columns={'Inflation rate, average consumer prices (Annual percent change)': 'Inflation Rate'}, inplace=True)
df.head(2)



# %%
#Filter the years between 2000 & 2023
columns_to_keep= ['Inflation Rate']+[col for col in df.columns if col.isdigit() and 2000 <= int(col) <=2023]
df = df[columns_to_keep]
df.head(2)

# %%
#Rename all no data values with a blank
df.replace('no data','',inplace=True)
df.head(2)

# %%
#Export Excel
Output_Path= (r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\europe inflation\Inflation_worldwide.xlsx')
df.to_excel(Output_Path)
```

2 STANDARD LONG FORMAT

Since some dataset are on horizontal format let homogenize all the data set to vertical format.

From:

Country	Country Code	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022
Aruba	ABW	14.43	13.74	12.99	12.62	11.92	12.35	13.06	12.96	12.75	12.35	12.19	12.25	12.72	13.32	13.53	12.43	12.3	11.53	9.88	9.14	8.1	7.19	7.18
Africa Eastern and Southern	AFE	40.53	40.34	40.05	39.75	39.58	39.41	39.24	39	38.85	38.36	37.94	37.48	36.92	36.45	36.03	35.61	35.19	34.89	34.61	34.34	33.92	33.55	33.14
Afghanistan	AFG	49.66	48.98	48.2	47.35	46.33	45.26	44.72	43.86	41.51	41.16	40.6	39.85	40.01	39.6	39.1	38.8	37.94	37.34	36.93	36.47	36.05	35.84	35.14

To:

Country	Country Code	Year	Divorce per 100 Marriages
Austria	AUT	2013	44.2
Azerbaijan	AZE	2013	13.5
Belarus	BLR	2013	41.4
Belgium	BEL	2013	65.7

The objective of this script is to transpose a dataset that contains GDP per capita data for various countries. The process begins with the loading of the primary dataset along with a reference file that includes country codes. Subsequently, the code merges these datasets to ensure that all countries possess corresponding country codes. Following this, the columns are reordered to position the "Country Code" column as desired. The key functionality of the code is derived from the pd.melt() function, which adeptly transforms the data from a wide format—where each year is represented as a distinct column—to a long format. In the long format, each row corresponds to a specific country-year combination, with the pertinent columns including "Country," "Country Code," "Year," and "GDP Per Capita." Ultimately, the transposed dataset is exported to an Excel file for further analysis.

We utilize this script for all the datasets that are required.

```
# %% [markdown]
# Lets transpose the dataset to keep a clean and standard database

# %%
#Import Library
import pandas as pd

# %%
#Import files
file_path=(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\CLEAN_DATASETS\GDP_Per_Capita_CLEAN.xlsx')
reference_file_path=(r'C:\Users\Admin\Desktop\GOOGLE DATA CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe birthrate\DATABASE\Regions_weather_Geo.xlsx')

#Load file to df
df = pd.read_excel(file_path)
Country_Codes = pd.read_excel(reference_file_path)

# %%
#Merge the Country code to those that dont have it
df = df.merge(Country_Codes[["Country", "Country Code"]], on="Country", how="left", suffixes=("", "_from_reference"))
df.drop(columns=["Country Code_from_reference"], inplace=True)
df.head(2)

# %%
#Reorder the columns
cols = df.columns.tolist()

# Move "Country Code" to second position
cols.insert(1, cols.pop(cols.index("Country Code")))
```

```
df = df[cols]
df.head(4)

# %%
# Proceed with the transposing step
df= pd.melt(df, id_vars=["Country", "Country Code"],
                    value_vars=[str(year) for year in range(2000, 2023)],
                    var_name="Year", value_name="GDP Per Capita")
df.reset_index(drop=True, inplace=True)
df.head(4)

# %%
Output_path= (r'C:\Users\Admin\Desktop\GOOGLE DATA
CERTIFICATE\CAPSTONE_GOOGLE_CERT\database_europe
birthrate\CLEAN_DATASETS\GDP_Per_Capita_CLEAN1.xlsx')
df.to_excel(Output_path,index=False)

```

3 FILTER BY EUROPEAN COUNTRIES

Considering that a considerable portion of our dataset encompasses global information, we will utilize various SQL joins to filter and extract data specifically related to European countries. This same query methodology will be applicable across all datasets.

```
---FILTERING Birth Rate DATASET

SELECT b.Country, b.CountryCode,b.Year, b.BirthRate
FROM Birth_Rate_CLEAN AS b
INNER JOIN Regions_weather_Geo AS r
ON b.CountryCode = r.CountryCode;

```