WDI mini project

In this analysis we are going to demonstrate multiple basic libraries and try to keep the code as clean and tidy as possible, we are going to do some merging, formatting the columns, droping unwanted columns, dropping the missnig values, melting of the dataframes, basic and animated plots, correlation heatmap and Regional analysis.

In [2]:
```
#importing all the necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import geopandas as gpd
import seaborn as sns
```

In [3]:
```
#Loading the datasets
country = pd.read_csv("Metadata_Country.csv")
pop = pd.read_csv("country_population.csv")
fer = pd.read_csv("fertility_rate.csv")
life_exp = pd.read_csv("life_expectancy.csv")
```

In [4]:
```
country.head(2)
```

Out[4]:

	Country Code	Region	IncomeGroup	SpecialNotes	TableName	Unnamed: 5
0	ABW	Latin America & Caribbean	High income	SNA data for 2000-2011 are updated from offici...	Aruba	NaN
1	AFG	South Asia	Low income	Fiscal year end: March 20; reporting period fo...	Afghanistan	NaN

In [5]:
```
pop.head(2)
```

Out[5]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	196
0	Aruba	ABW	Population, total	SP.POP.TOTL	54211.0	55438.0	56225.0	56695.0	57032
1	Afghanistan	AFG	Population, total	SP.POP.TOTL	8996351.0	9166764.0	9345868.0	9533954.0	9731361

2 rows × 61 columns

In [6]:
```
fer.head(2)
```

Out[6]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964	1965	...	2007
0	Aruba	ABW	Fertility rate, total (births per woman)	SP.DYN.TFRT.IN	4.82	4.655	4.471	4.271	4.059	3.842	...	1.763
1	Afghanistan	AFG	Fertility rate, total (births per woman)	SP.DYN.TFRT.IN	7.45	7.450	7.450	7.450	7.450	7.450	...	6.460

2 rows × 61 columns

In [7]:
```
life_exp.head(2)
```

Out[7]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964	1965	...
0	Aruba	ABW	Life expectancy at birth, total (years)	SP.DYN.LE00.IN	65.662	66.074	66.444	66.787	67.113	67.435	...
1	Afghanistan	AFG	Life expectancy at birth, total (years)	SP.DYN.LE00.IN	32.292	32.742	33.185	33.624	34.060	34.495	...

2 rows × 61 columns

In [8]:
```
country.shape
```

Out[8]:
```
(263, 6)
```

In [9]:
```
pop.shape
```

Out[9]:
```
(264, 61)
```

In [10]:
```
fer.shape
```

Out[10]:
```
(264, 61)
```

In [11]:
```
life_exp.shape
```

Out[11]:
```
(264, 61)
```

In [12]:
```
#making the column heading to lower case for easier execution
country.columns = country.columns.str.lower().str.replace(' ','_')
```

```
pop.columns = pop.columns.str.lower().str.replace(' ','_')
fer.columns = fer.columns.str.lower().str.replace(' ','_')
life_exp.columns = life_exp.columns.str.lower().str.replace(' ','_')
```

In [13]:

```
#droping unnecessary columns
country.drop(['incomegroup','specialnotes','tablename','unnamed:_5'],axis = 1,inplace
pop.drop(['indicator_name','indicator_code'], axis = 1, inplace = True)
fer.drop(['indicator_name','indicator_code'], axis = 1, inplace = True)
life_exp.drop(['indicator_name','indicator_code'], axis = 1, inplace = True)
```

In [14]:

```
#droping the null values in each dataset
country.dropna(axis = 0,inplace = True)
pop.dropna(axis=0, inplace=True)
fer.dropna(axis=0, inplace=True)
life_exp.dropna(axis = 0, inplace = True)
```

In [15]:

```
years = [str(i) for i in range(1960,2017)]
```

In [16]:

```
#using melt to reduce the no of columns
pop = pd.melt(pop,id_vars ='country_code',value_vars = years,var_name ='year',value_na
fer = pd.melt(fer,id_vars ='country_code',value_vars = years,var_name ='year',value_na
life_exp = pd.melt(life_exp,id_vars ='country_code',value_vars = years,var_name ='year
```

In [17]:

```
#merging the dataframes
df_1= pd.merge(country,pop,how='left',on=['country_code'])
df_2= pd.merge(df_1,fer,how='left',on=['country_code', 'year'])
df = pd.merge(df_2,life_exp,how='left',on=['country_code', 'year'])
```

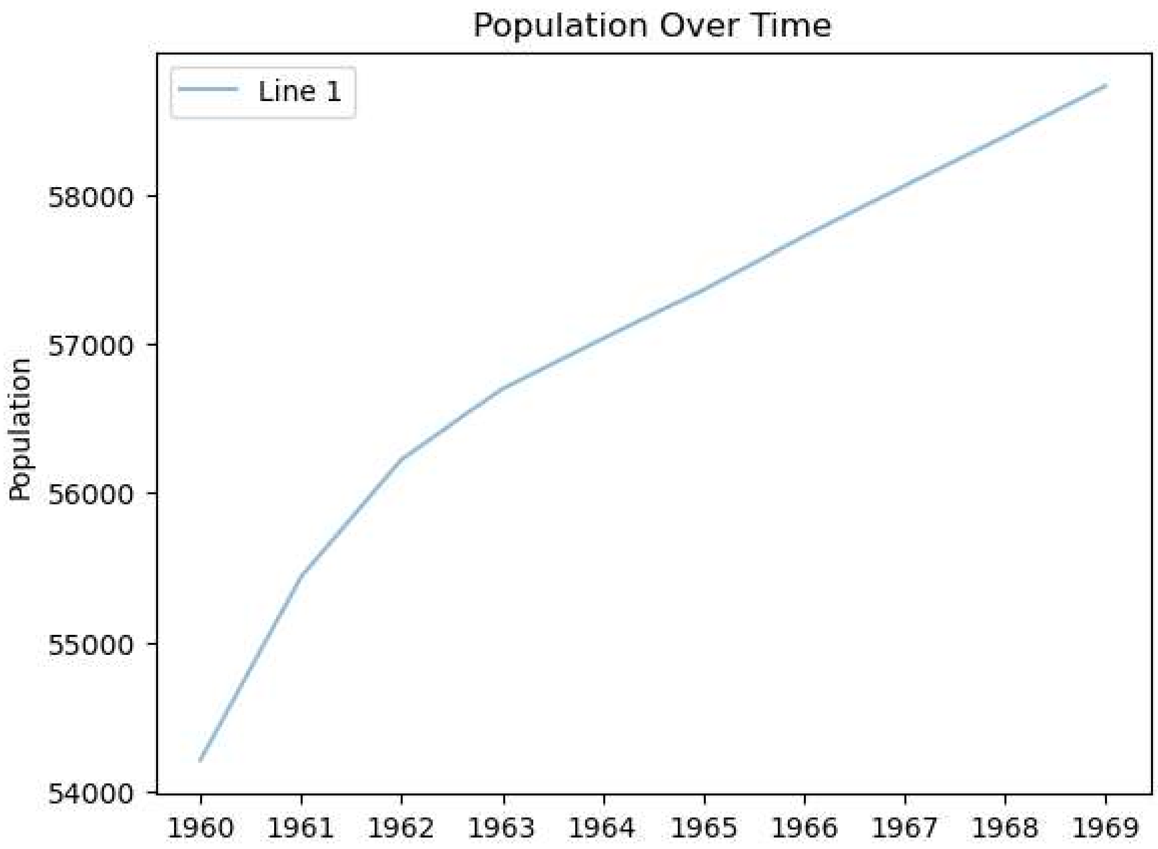
In [18]:

```
#droping the null values from the dataframe
df.dropna(axis = 0,inplace = True)
```

Year vs Population

In [19]:

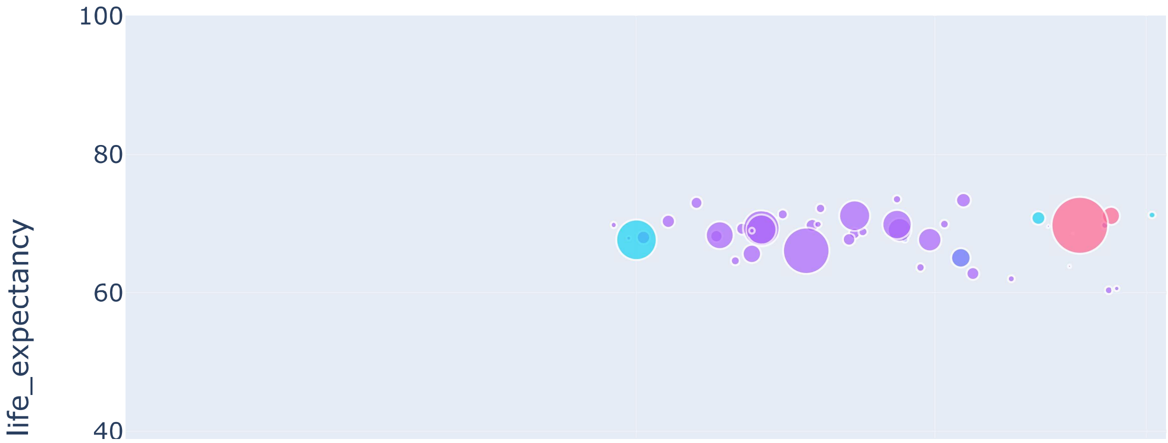
```
#line chart of year VS population
max_pop = df['population'].max()
year = df['year'][:10]
pop = df['population'][:10]
plt.plot(year, pop, alpha=0.5, label='Line 1')
plt.ylabel('Population')
plt.title('Population Over Time')
plt.legend()
plt.show()
```



Fertility rate vs Life Expectancy

In [20]:

```
px.scatter(df,
           x="fertility_rate",
           y="life_expectancy",
           animation_frame="year",
           animation_group="country_code",
           size="population",
           color="region",
           hover_name="country_code",
           log_x=True,
           size_max=55,
           range_x=[1,10],
           range_y=[10,100])
```



Correlation Heatmap

In [21]:

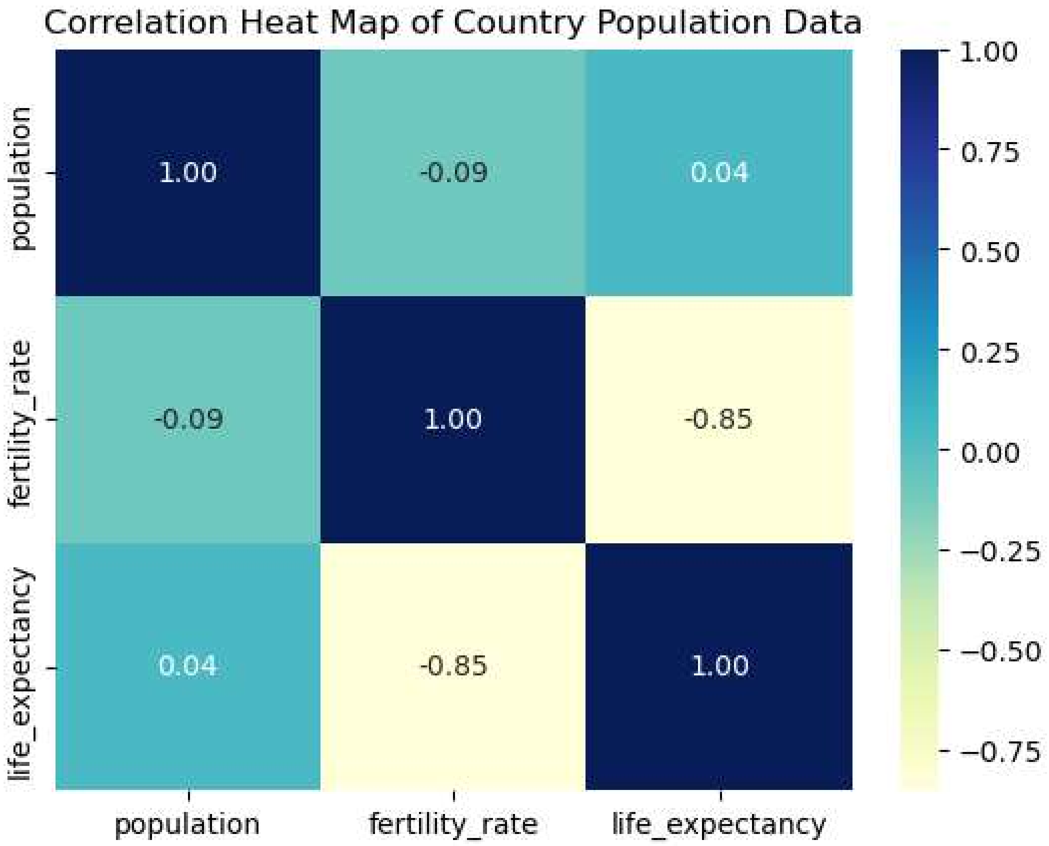
```
#correlation heatmap for the df
corr = df.corr()

sns.heatmap(corr, cmap="YlGnBu", annot=True, fmt=".2f")
plt.title("Correlation Heat Map of Country Population Data")

plt.show()
```

```
C:\Users\Mikey\AppData\Local\Temp\ipykernel_12276\2337914334.py:2: FutureWarning:

The default value of numeric_only in DataFrame.corr is deprecated. In a future versio
n, it will default to False. Select only valid columns or specify the value of numeri
c_only to silence this warning.
```



Regional Analysis

In [22]:

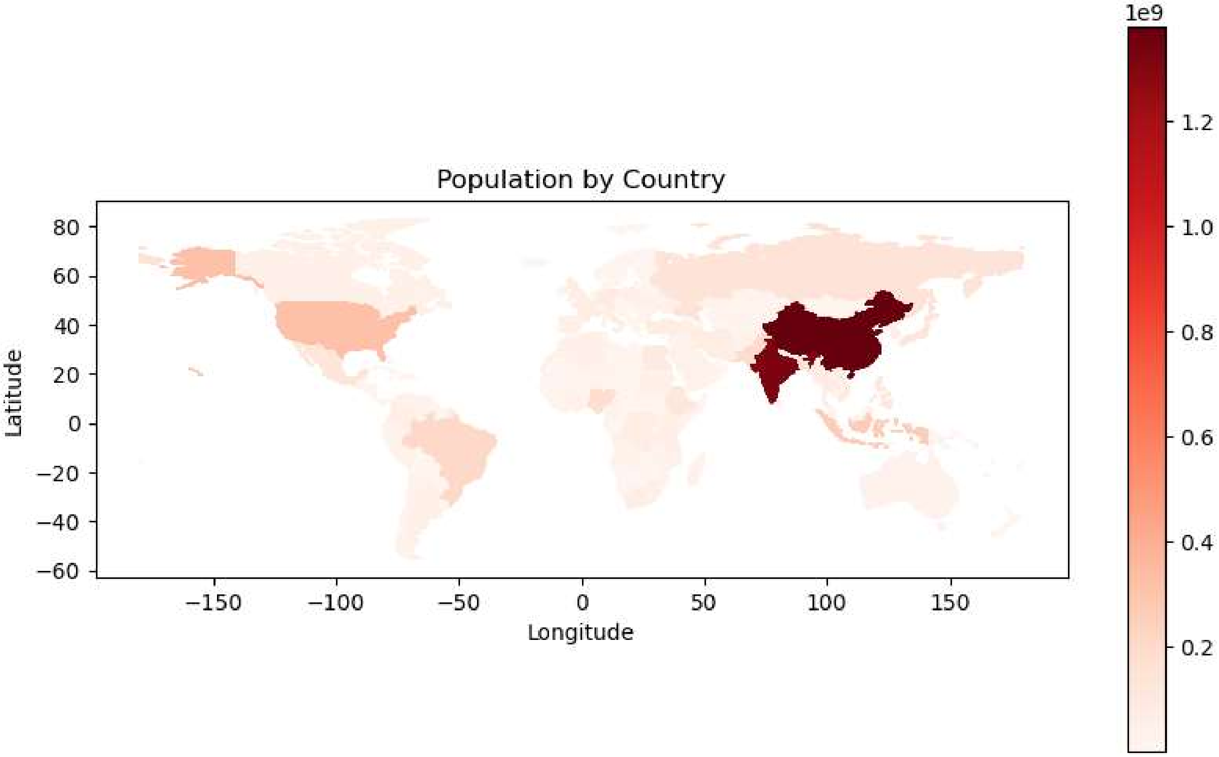
```
world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))

merged = pd.merge(world, df, left_on='iso_a3', right_on='country_code', how='left')

# Plot the regional data
merged.plot(column='population', cmap='Reds', legend=True, figsize=(10, 6))
plt.title('Population by Country')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```

```
C:\Users\Mikey\AppData\Local\Temp\ipykernel_12276\1802031940.py:1: FutureWarning:

The geopandas.dataset module is deprecated and will be removed in GeoPandas 1.0. You
can get the original 'naturalearth_lowres' data from https://www.naturalearthdata.co
m/downloads/110m-cultural-vectors/.
```


Population by Country
Latitude
80
60
40
20
0
−20
−40
−60
−150
−100
−50
0
50
100
150
Longitude
1e9
1.2
1.0
0.8
0.6
0.4
0.2

In [ ]:

In [ ]: