Importing Libraries

In [262]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings("ignore")
```

Importing Pakistan Population Dataset

The dataset is taken from this link:
https://databank.worldbank.org/country/PAK/556d8fa6/Popular_countries
(https://databank.worldbank.org/country/PAK/556d8fa6/Popular_countries)

In [263]:
```
data = pd.read_csv(r"C:/Users/zahid/Downloads/PakistanDataset/API_PAK_DS2_en_csv_v2_4
```

Introductory Details

In [264]:
```
data.head()
```

Out[264]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	
0	Pakistan	PAK	Internally displaced persons, total displaced ...	VC.IDP.TOCV	NaN	NaN	NaN	
1	Pakistan	PAK	Travel services (% of commercial service exports)	TX.VAL.TRVL.ZS.WT	NaN	NaN	NaN	
2	Pakistan	PAK	Commercial service exports (current US$)	TX.VAL.SERV.CD.WT	NaN	NaN	NaN	
3	Pakistan	PAK	Merchandise exports by the reporting economy (...	TX.VAL.MRCH.WL.CD	3.918000e+08	3.939000e+08	4.185000e+08	4.59
4	Pakistan	PAK	Merchandise exports to low- and middle-income ...	TX.VAL.MRCH.R4.ZS	3.241450e+00	4.417365e+00	3.966547e+00	2.72

5 rows × 66 columns

```
In [265]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1446 entries, 0 to 1445
Data columns (total 66 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Country Name    1442 non-null   object 
 1   Country Code    1442 non-null   object 
 2   Indicator Name  1442 non-null   object 
 3   Indicator Code  1442 non-null   object 
 4   1960            258 non-null    float64
 5   1961            296 non-null    float64
 6   1962            306 non-null    float64
 7   1963            312 non-null    float64
 8   1964            317 non-null    float64
 9   1965            322 non-null    float64
 10  1966            317 non-null    float64
 11  1967            324 non-null    float64
 12  1968            324 non-null    float64
 13  1969            329 non-null    float64
 14  1970            402 non-null    float64
 15  1971            458 non-null    float64
 16  1972            471 non-null    float64
 17  1973            473 non-null    float64
 18  1974            474 non-null    float64
 19  1975            495 non-null    float64
 20  1976            563 non-null    float64
 21  1977            564 non-null    float64
 22  1978            563 non-null    float64
 23  1979            568 non-null    float64
 24  1980            589 non-null    float64
 25  1981            593 non-null    float64
 26  1982            579 non-null    float64
 27  1983            577 non-null    float64
 28  1984            576 non-null    float64
 29  1985            583 non-null    float64
 30  1986            602 non-null    float64
 31  1987            592 non-null    float64
 32  1988            591 non-null    float64
 33  1989            597 non-null    float64
 34  1990            703 non-null    float64
 35  1991            724 non-null    float64
 36  1992            713 non-null    float64
 37  1993            691 non-null    float64
 38  1994            684 non-null    float64
 39  1995            741 non-null    float64
 40  1996            717 non-null    float64
 41  1997            752 non-null    float64
 42  1998            788 non-null    float64
 43  1999            759 non-null    float64
 44  2000            876 non-null    float64
 45  2001            848 non-null    float64
 46  2002            829 non-null    float64
 47  2003            819 non-null    float64
 48  2004            912 non-null    float64
 49  2005            956 non-null    float64
 50  2006            978 non-null    float64
 51  2007            1050 non-null   float64
 52  2008            1002 non-null   float64
 53  2009            994 non-null    float64
 54  2010            1031 non-null   float64
 55  2011            1078 non-null   float64
 56  2012            965 non-null    float64
 57  2013            1142 non-null   float64
 58  2014            1046 non-null   float64
 59  2015            1059 non-null   float64
 60  2016            983 non-null    float64
 61  2017            970 non-null    float64
 62  2018            1111 non-null   float64
 63  2019            963 non-null    float64
 64  2020            717 non-null    float64
 65  2021            560 non-null    float64
dtypes: float64(62), object(4)
memory usage: 745.7+ KB
```

```
In [266]: data.describe
```

```
Out[266]: <bound method NDFrame.describe of      Country Name Country Code  \
          0       Pakistan          PAK
          1       Pakistan          PAK
          2       Pakistan          PAK
          3       Pakistan          PAK
          4       Pakistan          PAK
          ...          ...          ...
          1441    Pakistan          PAK
          1442    Pakistan          PAK
          1443    Pakistan          PAK
          1444    Pakistan          PAK
          1445    Pakistan          PAK

                                                  Indicator Name     Indicator Code  \
          0     Internally displaced persons, total displaced ...        VC.IDP.TOCV
          1     Travel services (% of commercial service exports)  TX.VAL.TRVL.ZS.WT
          2              Commercial service exports (current US$)  TX.VAL.SERV.CD.WT
          3     Merchandise exports by the reporting economy (...  TX.VAL.MRCH.WL.CD
          4     Merchandise exports to low- and middle-income ...  TX.VAL.MRCH.R4.ZS
          ...                                                 ...                ...
          1441                              Expense (current LCU)     GC.XPN.TOTL.CN
          1442                      Interest payments (% of revenue)  GC.XPN.INTP.RV.ZS
          1443             Compensation of employees (% of expense)     GC.XPN.COMP.ZS
          1444  Taxes on income, profits and capital gains (cu...     GC.TAX.YPKG.CN
          1445                          Other taxes (current LCU)     GC.TAX.OTHR.CN

                        1960          1961          1962          1963          1964  \
          0              NaN           NaN           NaN           NaN           NaN
          1              NaN           NaN           NaN           NaN           NaN
          2              NaN           NaN           NaN           NaN           NaN
          3     3.918000e+08  3.939000e+08  4.185000e+08  4.595000e+08  4.889000e+08
          4     3.241450e+00  4.417365e+00  3.966547e+00  2.720348e+00  2.863571e+00
          ...            ...           ...           ...           ...           ...
          1441           NaN           NaN           NaN           NaN           NaN
          1442           NaN           NaN           NaN           NaN           NaN
          1443           NaN           NaN           NaN           NaN           NaN
          1444           NaN           NaN           NaN           NaN           NaN
          1445           NaN           NaN           NaN           NaN           NaN

                        1965  ...          2012          2013          2014  \
          0              NaN  ...  7.580000e+05  7.470000e+05  1.900000e+06
          1              NaN  ...  1.057722e+01  8.703536e+00  7.939189e+00
          2              NaN  ...  3.205000e+09  3.309000e+09  3.552000e+09
          3     5.237000e+08  ...  2.478405e+10  2.516280e+10  2.473126e+10
          4     2.253198e+00  ...  2.861330e+00  2.514726e+00  2.501451e+00
          ...            ...  ...           ...           ...           ...
          1441           NaN  ...           NaN           NaN           NaN
          1442           NaN  ...           NaN           NaN           NaN
          1443           NaN  ...           NaN           NaN           NaN
          1444           NaN  ...           NaN           NaN           NaN
          1445           NaN  ...           NaN           NaN           NaN

                        2015          2016          2017          2018          2019  \
          0     1.459000e+06  4.640000e+05  2.490000e+05  1.190000e+05  1.060000e+05
          1     9.164498e+00  8.735257e+00  7.823196e+00  8.306904e+00  1.060962e+01
          2     3.459000e+09  3.686211e+09  4.499440e+09  4.694890e+09  4.656150e+09
          3     2.213954e+10  2.054714e+10  2.150368e+10  2.316338e+10  2.333783e+10
          4     2.102910e+00  2.015183e+00  1.680216e+00  1.604904e+00  1.752165e+00
          ...            ...           ...           ...           ...           ...
          1441           NaN           NaN           NaN           NaN           NaN
          1442           NaN           NaN           NaN           NaN           NaN
          1443           NaN           NaN           NaN           NaN           NaN
          1444           NaN           NaN           NaN           NaN           NaN
          1445           NaN           NaN           NaN           NaN           NaN

                        2020          2021
          0     1.040000e+05  1.040000e+05
          1     9.943217e+00  1.022521e+01
          2     4.415070e+09  5.466880e+09
          3     2.223546e+10           NaN
          4     1.486735e+00           NaN
          ...            ...           ...
          1441           NaN           NaN
          1442           NaN           NaN
          1443           NaN           NaN
          1444           NaN           NaN
          1445           NaN           NaN
```

```
[1446 rows x 66 columns]>
```

```
In [267]: data["Country Name"].unique()
```

```
Out[267]: array(['Pakistan', nan], dtype=object)
```

```
In [268]: data.columns
```

```
Out[268]: Index(['Country Name', 'Country Code', 'Indicator Name', 'Indicator Code',
                 '1960', '1961', '1962', '1963', '1964', '1965', '1966', '1967', '1968',
                 '1969', '1970', '1971', '1972', '1973', '1974', '1975', '1976', '1977',
                 '1978', '1979', '1980', '1981', '1982', '1983', '1984', '1985', '1986',
                 '1987', '1988', '1989', '1990', '1991', '1992', '1993', '1994', '1995',
                 '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004',
                 '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013',
                 '2014', '2015', '2016', '2017', '2018', '2019', '2020', '2021'],
                dtype='object')
```

```
In [269]: data.index
```

```
Out[269]: RangeIndex(start=0, stop=1446, step=1)
```

Removing Null values

```
In [270]: print("No. of Null values in the data set :", data.isnull().sum().sum())
```

```
No. of Null values in the data set : 47492
```

```
In [271]: # Remove null values from the DataFrame
          data = data.dropna()
```

```
In [272]: # Verify that null values have been removed
          print("No. of Null values in the data set after removal:", data.isnull().sum().sum()
```

```
No. of Null values in the data set after removal: 0
```

```
In [273]: data.columns
```

```
Out[273]: Index(['Country Name', 'Country Code', 'Indicator Name', 'Indicator Code',
                 '1960', '1961', '1962', '1963', '1964', '1965', '1966', '1967', '1968',
                 '1969', '1970', '1971', '1972', '1973', '1974', '1975', '1976', '1977',
                 '1978', '1979', '1980', '1981', '1982', '1983', '1984', '1985', '1986',
                 '1987', '1988', '1989', '1990', '1991', '1992', '1993', '1994', '1995',
                 '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004',
                 '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013',
                 '2014', '2015', '2016', '2017', '2018', '2019', '2020', '2021'],
                dtype='object')
```

In [274]:
```
data
```

Out[274]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	
28	Pakistan	PAK	Population, female (% of total population)	SP.POP.TOTL.FE.ZS	4.604375e+01	4.610449e+01	4.616030e+01	4.6
30	Pakistan	PAK	Age dependency ratio (% of working-age populat...	SP.POP.DPND	7.971747e+01	8.008104e+01	8.073835e+01	8.1
31	Pakistan	PAK	Population ages 75-79, male (% of male populat...	SP.POP.7579.MA.5Y	7.319649e-01	7.134462e-01	6.973860e-01	6.
32	Pakistan	PAK	Population ages 65 and above (% of total popul...	SP.POP.65UP.TO.ZS	3.720856e+00	3.624116e+00	3.539410e+00	3.4
33	Pakistan	PAK	Population ages 65 and above, female (% of fem...	SP.POP.65UP.FE.ZS	3.359463e+00	3.253534e+00	3.162808e+00	3.0
...	...	...	...	...	...	...	...	
1398	Pakistan	PAK	Households and NPISHs final consumption expend...	NE.CON.PRVT.ZS	8.224488e+01	8.387804e+01	8.206577e+01	7.8
1401	Pakistan	PAK	General government final consumption expenditu...	NE.CON.GOVT.KN	1.646390e+11	1.628370e+11	1.770990e+11	1.9
1402	Pakistan	PAK	General government final consumption expenditu...	NE.CON.GOVT.CD	3.849223e+08	4.034019e+08	4.351113e+08	4.7
1403	Pakistan	PAK	Military expenditure (current LCU)	MS.MIL.XPND.CN	9.945000e+08	1.000500e+09	9.540000e+08	1.0
1408	Pakistan	PAK	Fixed telephone subscriptions	IT.MLT.MAIN	6.677600e+04	6.677600e+04	6.677600e+04	6.6

194 rows × 66 columns

insights

In [275]:
```
data.mean()
```

Out[275]:
```
1960    6.121543e+10
1961    6.490360e+10
1962    6.816337e+10
1963    7.670173e+10
1964    8.240299e+10
            ...
2017    2.253538e+12
2018    2.463886e+12
2019    2.666158e+12
2020    2.785241e+12
2021    3.174667e+12
Length: 62, dtype: float64
```

In [276]:
```
data.median()
```

Out[276]:
```
1960     21459.871781
1961     55396.833215
1962     22682.891187
1963     24069.772453
1964     25273.609774
             ...
2017    163093.223000
2018    176991.732350
2019    194432.939100
2020    299383.500000
2021    246213.146800
Length: 62, dtype: float64
```

In [307]:
```
def diff_max_min(x):
    return x['2010'].max() - x['2010'].min()

grouped_data.apply(diff_max_min)
```

Out[307]:
```
1960           2021
-1.075782e+10   6.254680e+11    0.0
-1.079000e+09  -4.984620e+12    0.0
-2.265855e+08  -3.111285e+10    0.0
-2.599794e+07  -7.049290e+11    0.0
-5.459457e+06  -4.400000e+09    0.0
                                ...
 1.076970e+12   8.420710e+12    0.0
 1.103670e+12   8.414660e+12    0.0
 1.798970e+12   3.657260e+13    0.0
 1.954490e+12   3.909180e+13    0.0
 2.382600e+12   3.938280e+13    0.0
Length: 198, dtype: float64
```

In [308]:
```
Population_by_Year = {
    'Year': ['1960', '1990', '2000', '2010', '2021'],
    'Population': ['1.745840e+09', '1.371820e+09', '675659973.1', '2.933210e+09', '2
df = pd.DataFrame(Population_by_Year)
print("Mode of the Populations by Year wise \n", df)
```

```
Mode of the Populations by Year wise
     Year    Population
0  1960  1.745840e+09
1  1990  1.371820e+09
2  2000   675659973.1
3  2010  2.933210e+09
4  2021  2.698300e+09
```

In [279]:
```
data['2021'].describe
```

Out[279]:
```
<bound method NDFrame.describe of 28      4.951824e+01
30      6.995679e+01
31      6.991722e-01
32      4.221414e+00
33      4.548844e+00
            ...
1398    8.335130e+01
1401    4.161030e+12
1402    3.809136e+10
1403    1.836450e+12
1408    2.989133e+06
Name: 2021, Length: 194, dtype: float64>
```

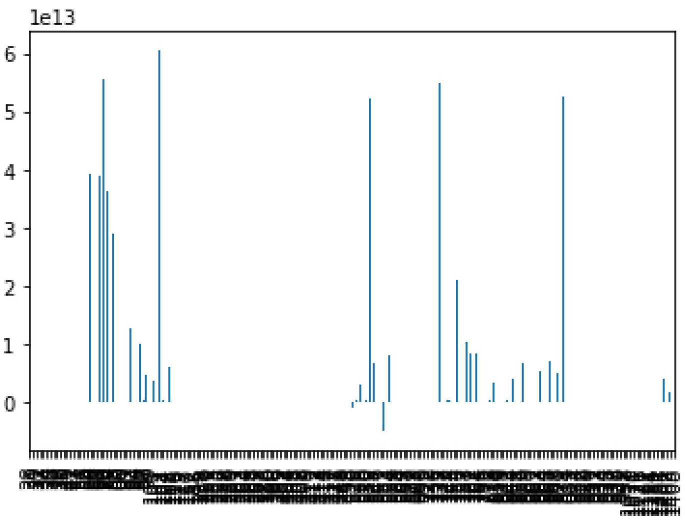
In [280]:
```
data.columns
```

Out[280]:
```
Index(['Country Name', 'Country Code', 'Indicator Name', 'Indicator Code',
       '1960', '1961', '1962', '1963', '1964', '1965', '1966', '1967', '1968',
       '1969', '1970', '1971', '1972', '1973', '1974', '1975', '1976', '1977',
       '1978', '1979', '1980', '1981', '1982', '1983', '1984', '1985', '1986',
       '1987', '1988', '1989', '1990', '1991', '1992', '1993', '1994', '1995',
       '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004',
       '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013',
       '2014', '2015', '2016', '2017', '2018', '2019', '2020', '2021'],
      dtype='object')
```

Visualization

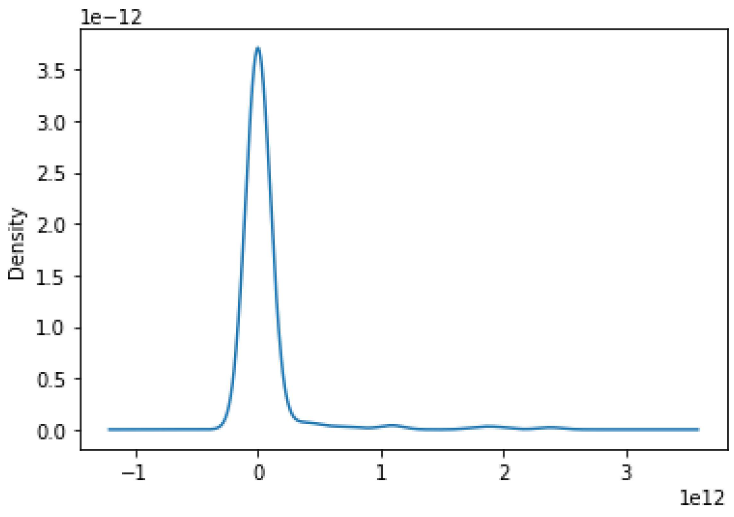
```
In [281]: data['2021'].plot(kind='bar')
```

```
Out[281]: <AxesSubplot:>
```

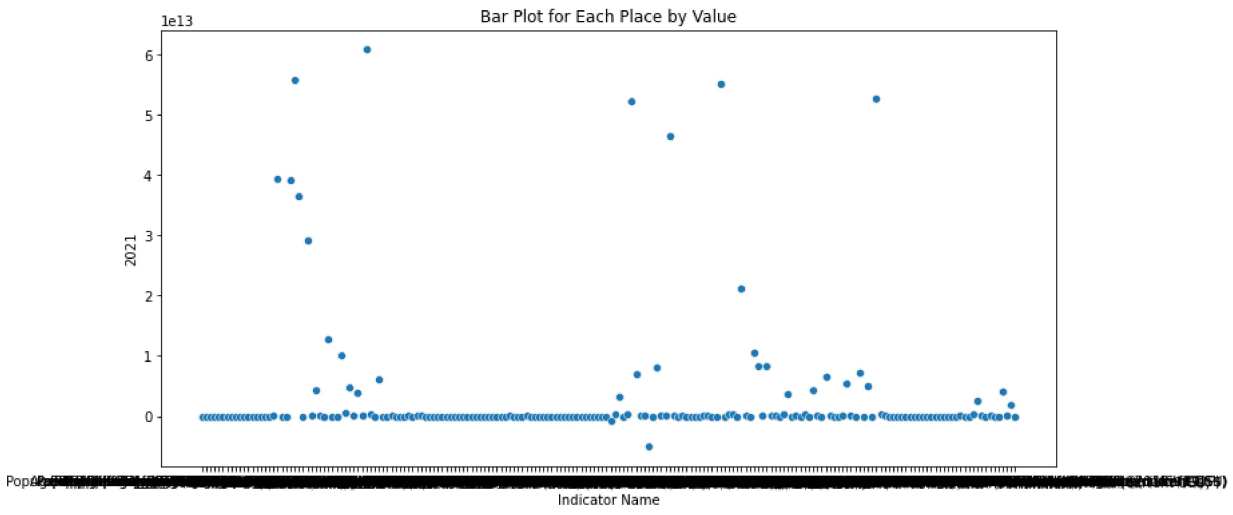


```
In [282]: data['1960'].plot(kind='kde')
```

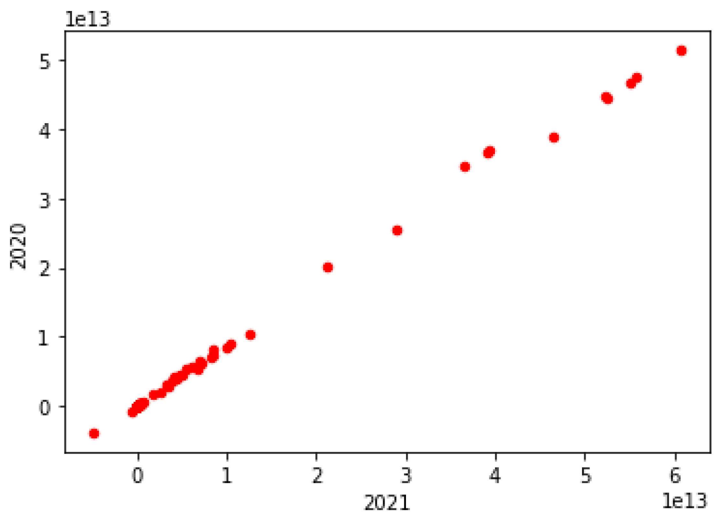
```
Out[282]: <AxesSubplot:ylabel='Density'>
```



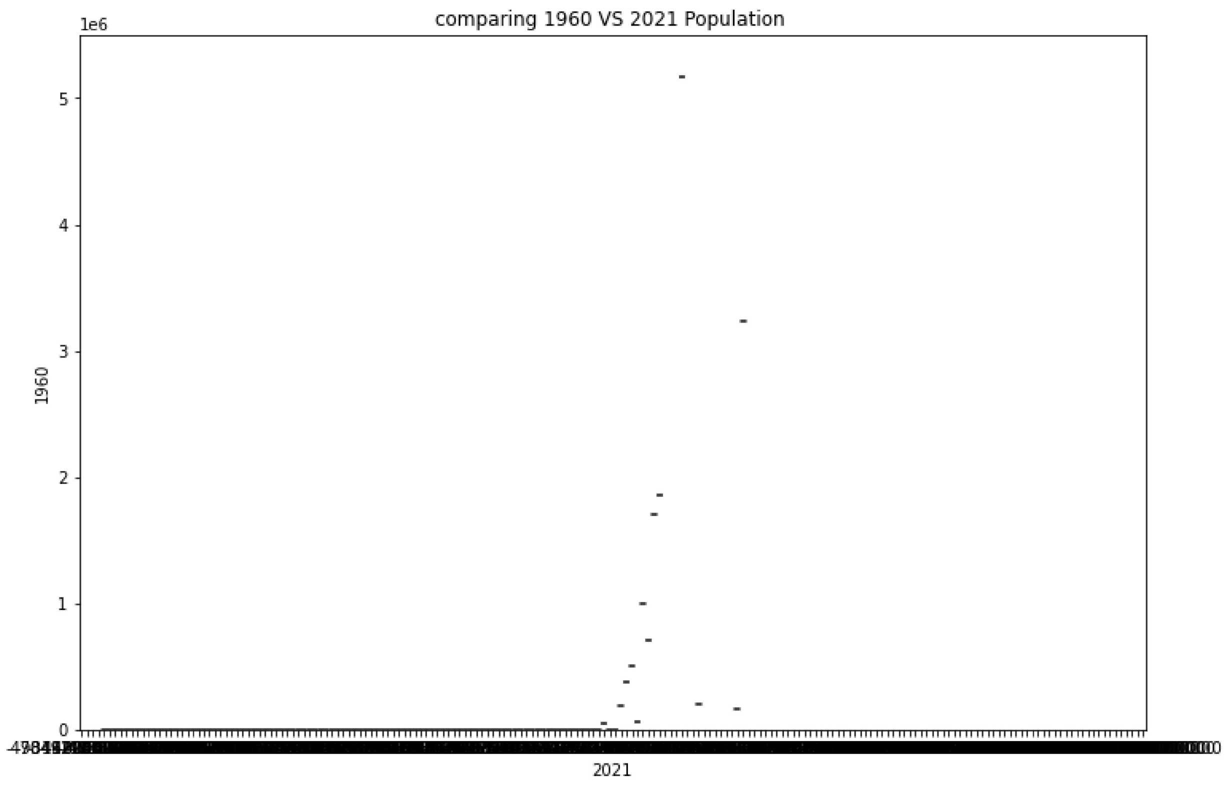
```
In [283]: plt.figure(figsize=(12, 6))
          sns.scatterplot(data = data, x = "Indicator Name", y = "2021")
          plt.title("Bar Plot for Each Place by Value")
          plt.show()
```



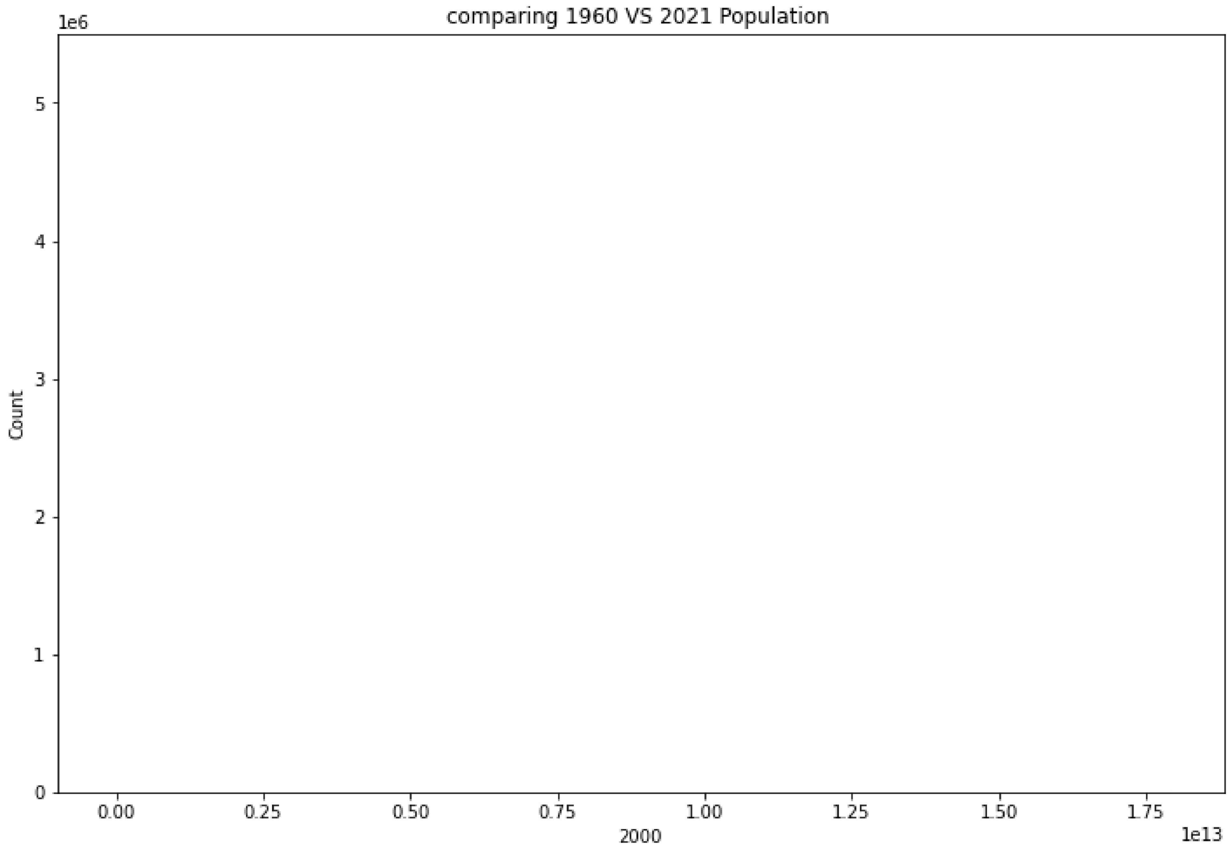
```
In [284]: data.plot.scatter('2021', '2020', color = 'red')

Out[284]: <AxesSubplot:xlabel='2021', ylabel='2020'>
```



```
In [285]: plt.figure(figsize=(12, 8))
          sns.boxplot(data = data, x = "2021", y = "1960")
          plt.title("comparing 1960 VS 2021 Population ")
          plt.ylim([0, 5_500_000])
          plt.show()
```

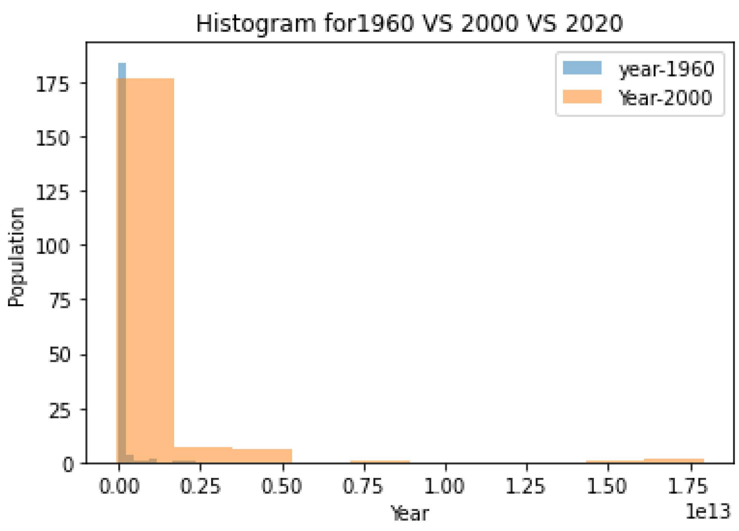


In [286]:
```
plt.figure(figsize=(12, 8))
sns.histplot(data = data, x = "2000")
plt.title("comparing 1960 VS 2021 Population ")
plt.ylim([0, 5_500_000])
plt.show()
```

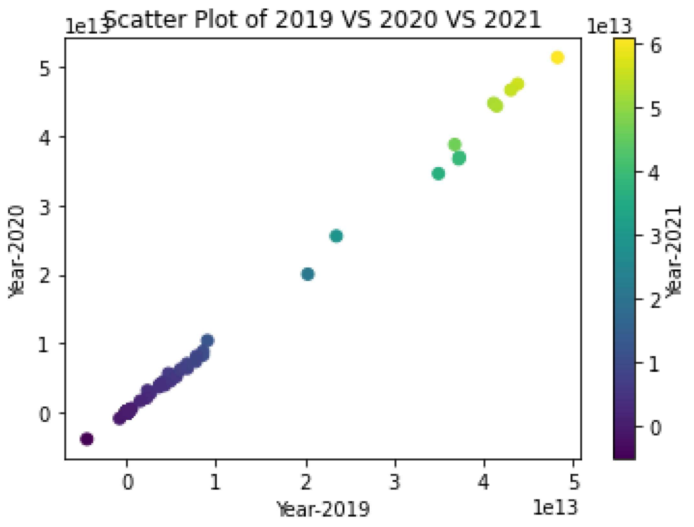


In [287]:
```
# Create a figure and axis object
fig, ax = plt.subplots()

# Create a histogram plot for the 1960 of data
ax.hist(data['1960'], bins=10, alpha=0.5, label='year-1960')

# Create a histogram plot for the 2000 of data
ax.hist(data['2000'], bins=10, alpha=0.5, label='Year-2000')

#  Histogram plot for Year-2020
# ax.hist(data['2020'], bins=10, alpha=0.5, label='Year-2020')

# Set axis labels and title
ax.set_xlabel('Year')
ax.set_ylabel('Population')
ax.set_title('Histogram for1960 VS 2000 VS 2020')

# Add legend
ax.legend()
```

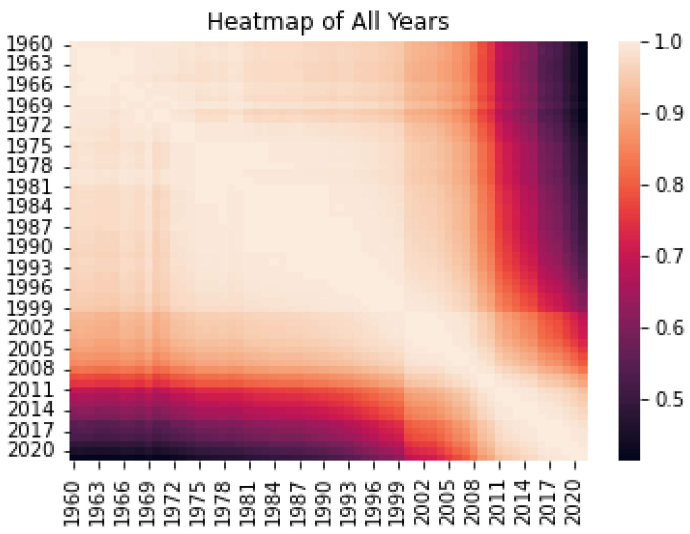
Out[287]:
```
<matplotlib.legend.Legend at 0x216d1484b80>
```



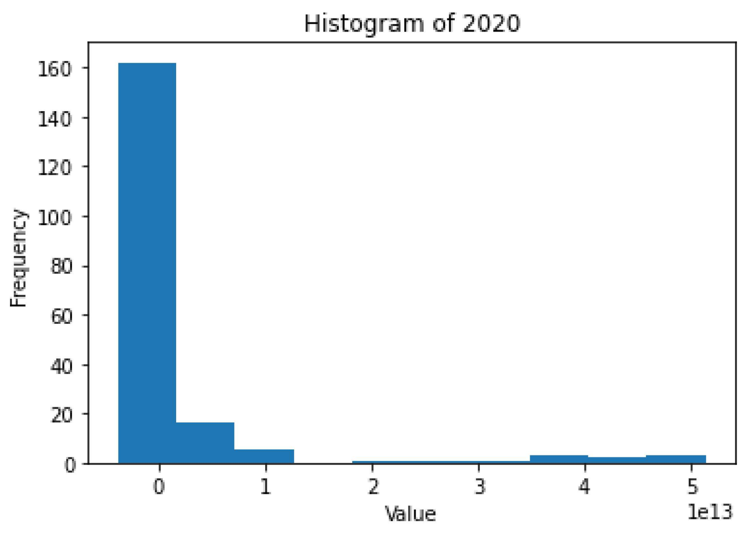
In [288]:
```
# Create a scatter plot with three columns
plt.scatter(data['2019'], data['2020'], c=data['2021'])

# Add labels and title to the plot
plt.xlabel('Year-2019')
plt.ylabel('Year-2020')
plt.title('Scatter Plot of 2019 VS 2020 VS 2021')

# Add a colorbar for the third column
cbar = plt.colorbar()
cbar.set_label('Year-2021')

# Display the plot
plt.show()
```



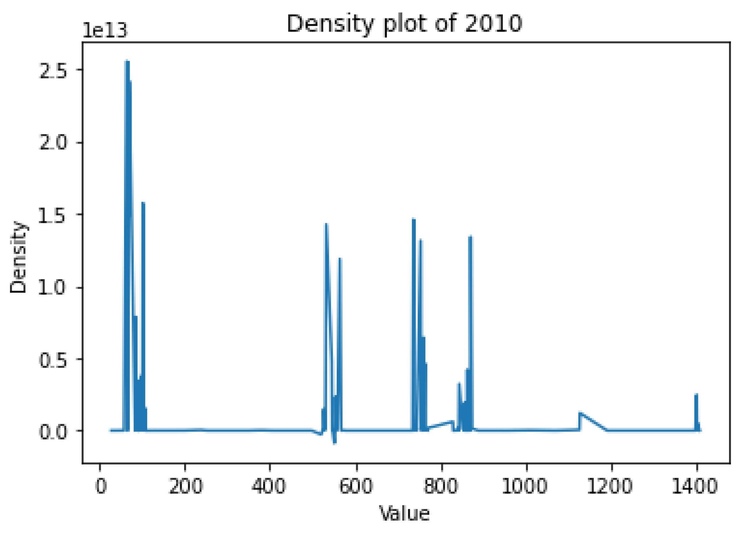
In [289]:
```
# Create a heatmap with all columns
sns.heatmap(data.corr())

# Add a title to the plot
plt.title('Heatmap of All Years')

# Display the plot
plt.show()
```



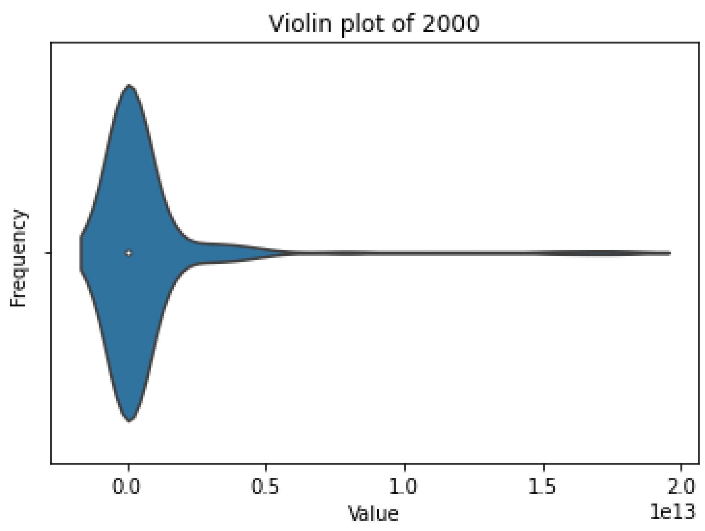
```
In [290]: # select the column to plot
          col_name = '2020'

          # create a histogram plot
          fig, ax = plt.subplots()
          ax.hist(data[col_name], bins=10)

          # set the title and labels for the plot
          ax.set_title('Histogram of {}'.format(col_name))
          ax.set_xlabel('Value')
          ax.set_ylabel('Frequency')

          # show the plot
          plt.show()
```



```
In [291]: # select the column to plot
          col_name = '2010'

          # create a density plot
          fig, ax = plt.subplots()
          ax.plot(data[col_name])

          # set the title and labels for the plot
          ax.set_title('Density plot of {}'.format(col_name))
          ax.set_xlabel('Value')
          ax.set_ylabel('Density')

Out[291]: Text(0, 0.5, 'Density')
```



In [292]:
```python
# select the column to plot
col_name = '2000'

# create a violin plot
fig, ax = plt.subplots()
sns.violinplot(data[col_name], ax=ax)

# set the title and labels for the plot
ax.set_title('Violin plot of {}'.format(col_name))
ax.set_xlabel('Value')
ax.set_ylabel('Frequency')

# show the plot
plt.show()
```



Follow for More:

Github: https://www.github.com/ZaidArman (https://www.github.com/ZaidArman)

Linkedin: https://www.linkedin.com/in/zaid-ullah07 (https://www.linkedin.com/in/zaid-ullah07)

Twitter: https://www.twitter.com/ZaidArman (https://www.twitter.com/ZaidArman)_

Instagram: https://www.instagram.com/zaid__arman7 (https://www.instagram.com/zaid__arman7)

Facebook: https://www.facebook.com/profile.php?id=100011010551170 (https://www.facebook.com/profile.php?id=100011010551170)

Made by ❤️ Zaid Ullah ❤️