In [ ]:

```

```

In [21]:

```
# IMPORT LIBRARIES
import numpy  as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [22]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\18_world-data-2023.csv")
a
```

Out[22]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Call Cc
0	Afghanistan	60	AF	58.10%	652,230	323,000	32.49	9
1	Albania	105	AL	43.10%	28,748	9,000	11.78	35
2	Algeria	18	DZ	17.40%	2,381,741	317,000	24.28	21
3	Andorra	164	AD	40.00%	468	NaN	7.20	37
4	Angola	26	AO	47.50%	1,246,700	117,000	40.73	24
...	...	...	...	...	...	...	...	
190	Venezuela	32	VE	24.50%	912,050	343,000	17.88	5
191	Vietnam	314	VN	39.30%	331,210	522,000	16.75	8
192	Yemen	56	YE	44.60%	527,968	40,000	30.45	96
193	Zambia	25	ZM	32.10%	752,618	16,000	36.19	26
194	Zimbabwe	38	ZW	41.90%	390,757	51,000	30.68	26

195 rows × 35 columns

In [23]:

```
a=a.head(10)
a
```

Out[23]:

Land rea(Km2)	Armed Forces size	Birth Rate	Calling Code	Capital/Major City	Co2-Emissions	...	Out of pocket health expenditure	Physicians per thousand	Popula
652,230	323,000	32.49	93.0	Kabul	8,672	...	78.40%	0.28	38,041
28,748	9,000	11.78	355.0	Tirana	4,536	...	56.90%	1.20	2,854
2,381,741	317,000	24.28	213.0	Algiers	150,006	...	28.10%	1.72	43,053
468	NaN	7.20	376.0	Andorra la Vella	469	...	36.40%	3.33	77
1,246,700	117,000	40.73	244.0	Luanda	34,693	...	33.40%	0.21	31,825
443	0	15.33	1.0	St. John's, Saint John	557	...	24.30%	2.76	97
2,780,400	105,000	17.02	54.0	Buenos Aires	201,348	...	17.60%	3.96	44,938
29,743	49,000	13.99	374.0	Yerevan	5,156	...	81.60%	4.40	2,957
7,741,220	58,000	12.60	61.0	Canberra	375,908	...	19.60%	3.68	25,766
83,871	21,000	9.70	43.0	Vienna	61,448	...	17.90%	5.17	8,877

In [24]:

```
# to find
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 35 columns):
 #   Column                                      Non-Null Count  Dtype  
---  ------                                      --------------  -----  
 0   Country                                     10 non-null     object 
 1   Density
(P/Km2)                              10 non-null     object 
 2   Abbreviation                                10 non-null     object 
 3   Agricultural Land( %)                       10 non-null     object 
 4   Land Area(Km2)                              10 non-null     object 
 5   Armed Forces size                           9 non-null      object 
 6   Birth Rate                                  10 non-null     float64
 7   Calling Code                                10 non-null     float64
 8   Capital/Major City                          10 non-null     object 
 9   Co2-Emissions                               10 non-null     object 
 10  CPI                                         9 non-null      object 
 11  CPI Change (%)                              9 non-null      object 
 12  Currency-Code                               10 non-null     object 
 13  Fertility Rate                              10 non-null     float64
 14  Forested Area (%)                           10 non-null     object 
 15  Gasoline Price                              10 non-null     object 
 16  GDP                                         10 non-null     object 
 17  Gross primary education enrollment (%)      10 non-null     object 
 18  Gross tertiary education enrollment (%)     9 non-null      object 
 19  Infant mortality                            10 non-null     float64
 20  Largest city                                10 non-null     object 
 21  Life expectancy                             9 non-null      float64
 22  Maternal mortality ratio                    9 non-null      float64
 23  Minimum wage                                9 non-null      object 
 24  Official language                           10 non-null     object 
 25  Out of pocket health expenditure            10 non-null     object 
 26  Physicians per thousand                     10 non-null     float64
 27  Population                                  10 non-null     object 
 28  Population: Labor force participation (%)   8 non-null      object 
 29  Tax revenue (%)                             9 non-null      object 
 30  Total tax rate                              9 non-null      object 
 31  Unemployment rate                           8 non-null      object 
 32  Urban_population                            10 non-null     object 
 33  Latitude                                    10 non-null     float64
 34  Longitude                                   10 non-null     float64
dtypes: float64(9), object(26)
memory usage: 2.9+ KB
```

In [25]:

```
# to display summary of statastic
a.describe()
```

Out[25]:

	Birth Rate	Calling Code	Fertility Rate	Infant mortality	Life expectancy	Maternal mortality ratio	Physicians per thousand	L
count	10.000000	10.000000	10.000000	10.000000	9.000000	9.000000	10.000000	10.
mean	18.512000	181.400000	2.512000	16.090000	74.788889	124.888889	2.671000	17.
std	10.754729	149.167467	1.416622	18.504321	7.376897	206.621904	1.738387	31.
min	7.200000	1.000000	1.270000	2.700000	60.800000	5.000000	0.210000	-38.
25%	11.985000	55.750000	1.650000	3.575000	74.900000	15.000000	1.330000	-4.
50%	14.660000	153.000000	1.875000	8.300000	76.700000	39.000000	3.045000	30.
75%	22.465000	327.250000	2.830000	17.825000	78.500000	112.000000	3.890000	40.
max	40.730000	376.000000	5.520000	51.600000	82.700000	638.000000	5.170000	47.

In [26]:

```
# to display colum heading
a.columns
```

Out[26]:

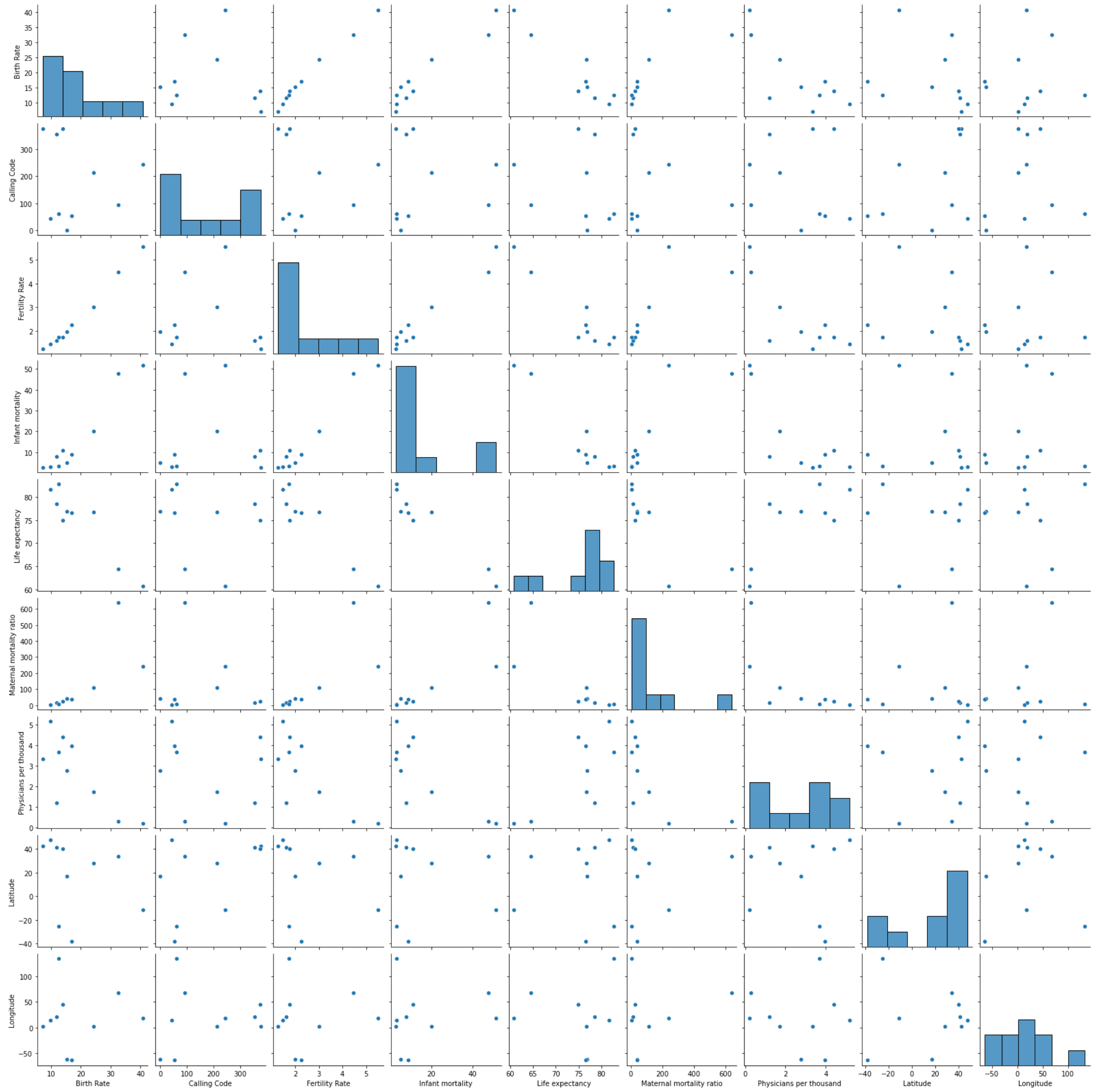
```
Index(['Country', 'Density\n(P/Km2)', 'Abbreviation', 'Agricultural Land(
%)',
       'Land Area(Km2)', 'Armed Forces size', 'Birth Rate', 'Calling Cod
e',
       'Capital/Major City', 'Co2-Emissions', 'CPI', 'CPI Change (%)',
       'Currency-Code', 'Fertility Rate', 'Forested Area (%)',
       'Gasoline Price', 'GDP', 'Gross primary education enrollment (%)',
       'Gross tertiary education enrollment (%)', 'Infant mortality',
       'Largest city', 'Life expectancy', 'Maternal mortality ratio',
       'Minimum wage', 'Official language', 'Out of pocket health expendit
ure',
       'Physicians per thousand', 'Population',
       'Population: Labor force participation (%)', 'Tax revenue (%)',
       'Total tax rate', 'Unemployment rate', 'Urban_population', 'Latitud
e',
       'Longitude'],
      dtype='object')
```

In [27]:

```
sns.pairplot(a)
```

Out[27]:

<seaborn.axisgrid.PairGrid at 0x198b6d6dee0>

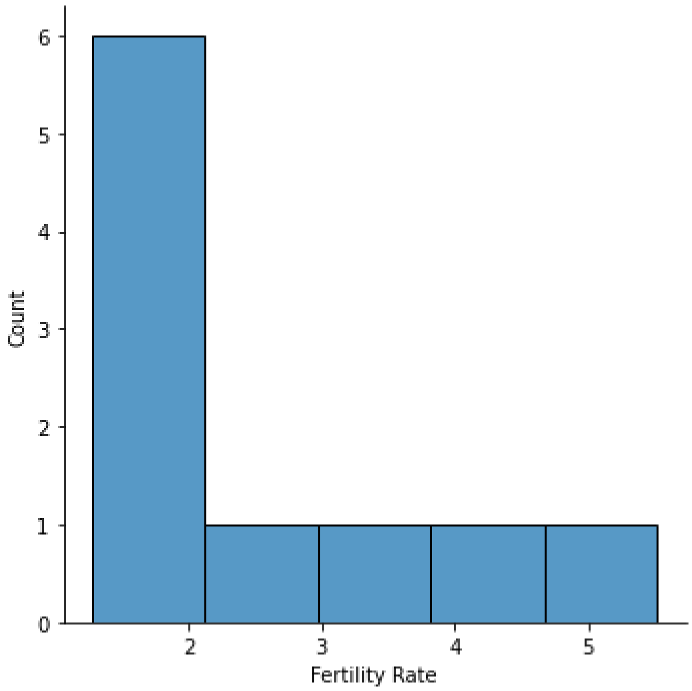


In [28]:

```
sns.displot(a["Fertility Rate"])
```

Out[28]:

```
<seaborn.axisgrid.FacetGrid at 0x198ba15df40>
```



In [29]:

```
b=a[['Density\n(P/Km2)', 'Abbreviation', 'Agricultural Land( %)',
      'Land Area(Km2)', 'Armed Forces size', 'Birth Rate', 'Calling Code' ]]
b
```

Out[29]:

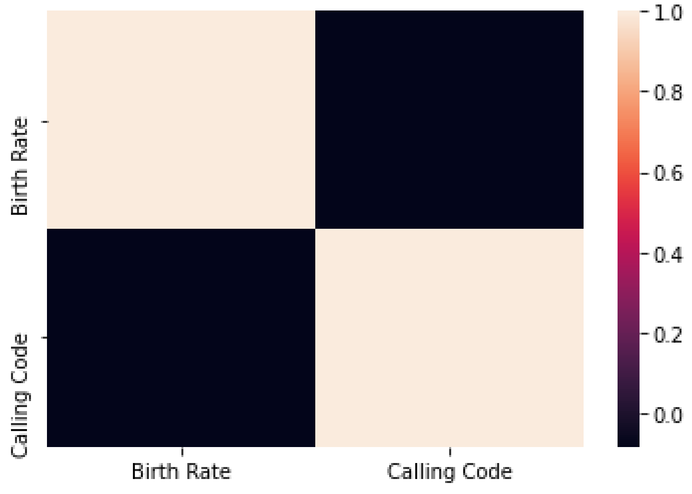
	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code
0	60	AF	58.10%	652,230	323,000	32.49	93.0
1	105	AL	43.10%	28,748	9,000	11.78	355.0
2	18	DZ	17.40%	2,381,741	317,000	24.28	213.0
3	164	AD	40.00%	468	NaN	7.20	376.0
4	26	AO	47.50%	1,246,700	117,000	40.73	244.0
5	223	AG	20.50%	443	0	15.33	1.0
6	17	AR	54.30%	2,780,400	105,000	17.02	54.0
7	104	AM	58.90%	29,743	49,000	13.99	374.0
8	3	AU	48.20%	7,741,220	58,000	12.60	61.0
9	109	AT	32.40%	83,871	21,000	9.70	43.0

In [30]:

```
sns.heatmap(b.corr())
```

Out[30]:

```
<AxesSubplot:>
```



In [46]:

```
x=a[['Density\n(P/Km2)','Birth Rate', 'Calling Code']]
y=a['Density\n(P/Km2)']
```

In [47]:

```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [48]:

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[48]:

```
LinearRegression()
```

In [49]:

```
lr.intercept_
```

Out[49]:

```
-5.684341886080802e-14
```

In [50]:

```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[50]:

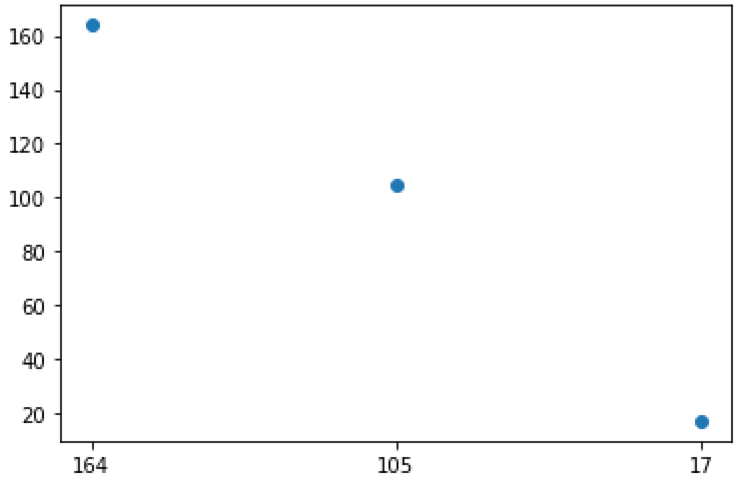
	Co-efficient
Density\n(P/Km2)	1.000000e+00
Birth Rate	4.002689e-17
Calling Code	-4.293771e-18

In [51]:

```
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[51]:

```
<matplotlib.collections.PathCollection at 0x198bad7fd30>
```



In [52]:

```
lr.score(x_test,y_test)
```

Out[52]:

```
1.0
```

In [53]:

```
lr.score(x_train,y_train)
```

Out[53]:

```
1.0
```

In [54]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [55]:

```
rr=Ridge(alpha=10)
rr.fit(x_test,y_test)
```

Out[55]:

```
Ridge(alpha=10)
```

In [56]:

```
rr.score(x_test,y_test)
```

Out[56]:

```
0.9999931833661889
```

In [57]:

```
la=Lasso(alpha=10)
la.fit(x_test,y_test)
```

Out[57]:

```
Lasso(alpha=10)
```

In [58]:

```
la.score(x_test,y_test)
```

Out[58]:

```
0.9999771729246785
```

In [63]:

```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[63]:

```
ElasticNet()
```

In [66]:

```
en.coef_
```

Out[66]:

```
array([ 9.99796728e-01, -0.00000000e+00, -5.38771499e-06])
```

In [67]:

```
en.intercept_
```

Out[67]:

```
0.016560087217570185
```

In [74]:

```
prediction=en.predict(x_test)
prediction
```

Out[74]:

```
array([163.98119771, 104.9933039 ,  17.01281353])
```

In [75]:

```
en.score(x_test,y_test)
```

Out[75]:

```
0.999999948600505
```

EVALUATION METRICS

In [76]:

```
from sklearn import metrics
```

In [78]:

```
print("Mean Absolute Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute Error: 0.012770639772059647
```

In [80]:

```
print("Mean Squared Error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared Error 0.0001875167799535862
```

In [81]:

```
print("Root Mean Squared Error",np.sqrt(metrics.mean_squared_error(y_test,prediction)))
```

```
Root Mean Squared Error 0.013693676641194147
```

In [ ]:

```

```