In [1]:

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

In [2]:

```
df1=pd.read_csv(r'C:\Users\user\Downloads\21_cities.csv')
df1
```

Out[2]:

	id	name	state_id	state_code	state_name	country_id	country_code	cou
0	52	Ashkāsham	3901	BDS	Badakhshan	1	AF	A
1	68	Fayzabad	3901	BDS	Badakhshan	1	AF	A
2	78	Jurm	3901	BDS	Badakhshan	1	AF	A
3	84	Khandūd	3901	BDS	Badakhshan	1	AF	A
4	115	Rāghistān	3901	BDS	Badakhshan	1	AF	A
...	...	...	...	...	...	...	...	
150449	131496	Redcliff	1957	MI	Midlands Province	247	ZW	
150450	131502	Shangani	1957	MI	Midlands Province	247	ZW	
150451	131503	Shurugwi	1957	MI	Midlands Province	247	ZW	
150452	131504	Shurugwi District	1957	MI	Midlands Province	247	ZW	
150453	131508	Zvishavane District	1957	MI	Midlands Province	247	ZW	

150454 rows × 11 columns

In [3]:

```
df=df1.head(30)
df
```

Out[3]:

	id	name	state_id	state_code	state_name	country_id	country_code	country_nar
0	52	Ashkāsham	3901	BDS	Badakhshan	1	AF	Afghanist
1	68	Fayzabad	3901	BDS	Badakhshan	1	AF	Afghanist
2	78	Jurm	3901	BDS	Badakhshan	1	AF	Afghanist
3	84	Khandūd	3901	BDS	Badakhshan	1	AF	Afghanist
4	115	Rāghistān	3901	BDS	Badakhshan	1	AF	Afghanist
5	131	Wākhān	3901	BDS	Badakhshan	1	AF	Afghanist
6	72	Ghormach	3871	BDG	Badghis	1	AF	Afghanist
7	108	Qala i Naw	3871	BDG	Badghis	1	AF	Afghanist
8	54	Baghlān	3875	BGL	Baghlan	1	AF	Afghanist
9	140	Ḩukūmatī Dahanah-ye Ghōrī	3875	BGL	Baghlan	1	AF	Afghanist
10	101	Nahrīn	3875	BGL	Baghlan	1	AF	Afghanist
11	105	Pul-e Khumrī	3875	BGL	Baghlan	1	AF	Afghanist
12	55	Balkh	3884	BAL	Balkh	1	AF	Afghanist
13	65	Dowlatābād	3884	BAL	Balkh	1	AF	Afghanist
14	85	Khulm	3884	BAL	Balkh	1	AF	Afghanist
15	91	Lab-Sar	3884	BAL	Balkh	1	AF	Afghanist
16	97	Mazār-e Sharīf	3884	BAL	Balkh	1	AF	Afghanist
17	112	Qarchī Gak	3884	BAL	Balkh	1	AF	Afghanist
18	57	Bāmyān	3872	BAM	Bamyan	1	AF	Afghanist
19	104	Panjāb	3872	BAM	Bamyan	1	AF	Afghanist
20	102	Nīlī	3892	DAY	Daykundi	1	AF	Afghanist
21	66	Farah	3899	FRA	Farah	1	AF	Afghanist
22	50	Andkhoy	3889	FYB	Faryab	1	AF	Afghanist
23	96	Maymana	3889	FYB	Faryab	1	AF	Afghanist
24	71	Ghazni	3870	GHA	Ghazni	1	AF	Afghanist
25	67	Fayrōz Kōh	3888	GHO	Ghōr	1	AF	Afghanist
26	121	Shahrak	3888	GHO	Ghōr	1	AF	Afghanist
27	141	ʻAlāqahdārī Dīshū	3873	HEL	Helmand	1	AF	Afghanist
28	70	Gereshk	3873	HEL	Helmand	1	AF	Afghanist
29	93	Lashkar Gāh	3873	HEL	Helmand	1	AF	Afghanist

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   id            30 non-null     int64
 1   name          30 non-null     object
 2   state_id      30 non-null     int64
 3   state_code    30 non-null     object
 4   state_name    30 non-null     object
 5   country_id    30 non-null     int64
 6   country_code  30 non-null     object
 7   country_name  30 non-null     object
 8   latitude      30 non-null     float64
 9   longitude     30 non-null     float64
 10  wikiDataId    30 non-null     object
dtypes: float64(2), int64(3), object(6)
memory usage: 2.7+ KB
```

In [5]:

```
df.describe()
```

Out[5]:

	id	state_id	country_id	latitude	longitude
count	30.000000	30.000000	30.0	30.000000	30.000000
mean	88.366667	3884.333333	1.0	35.385490	67.218887
std	26.284233	11.158214	0.0	1.887819	2.876642
min	50.000000	3870.000000	1.0	30.432060	62.116380
25%	67.250000	3873.500000	1.0	34.420842	64.868647
50%	88.000000	3884.000000	1.0	35.985460	66.960235
75%	104.750000	3891.250000	1.0	36.837665	68.713413
max	141.000000	3901.000000	1.0	37.660790	73.349280

In [6]:

```
df.columns
```

Out[6]:

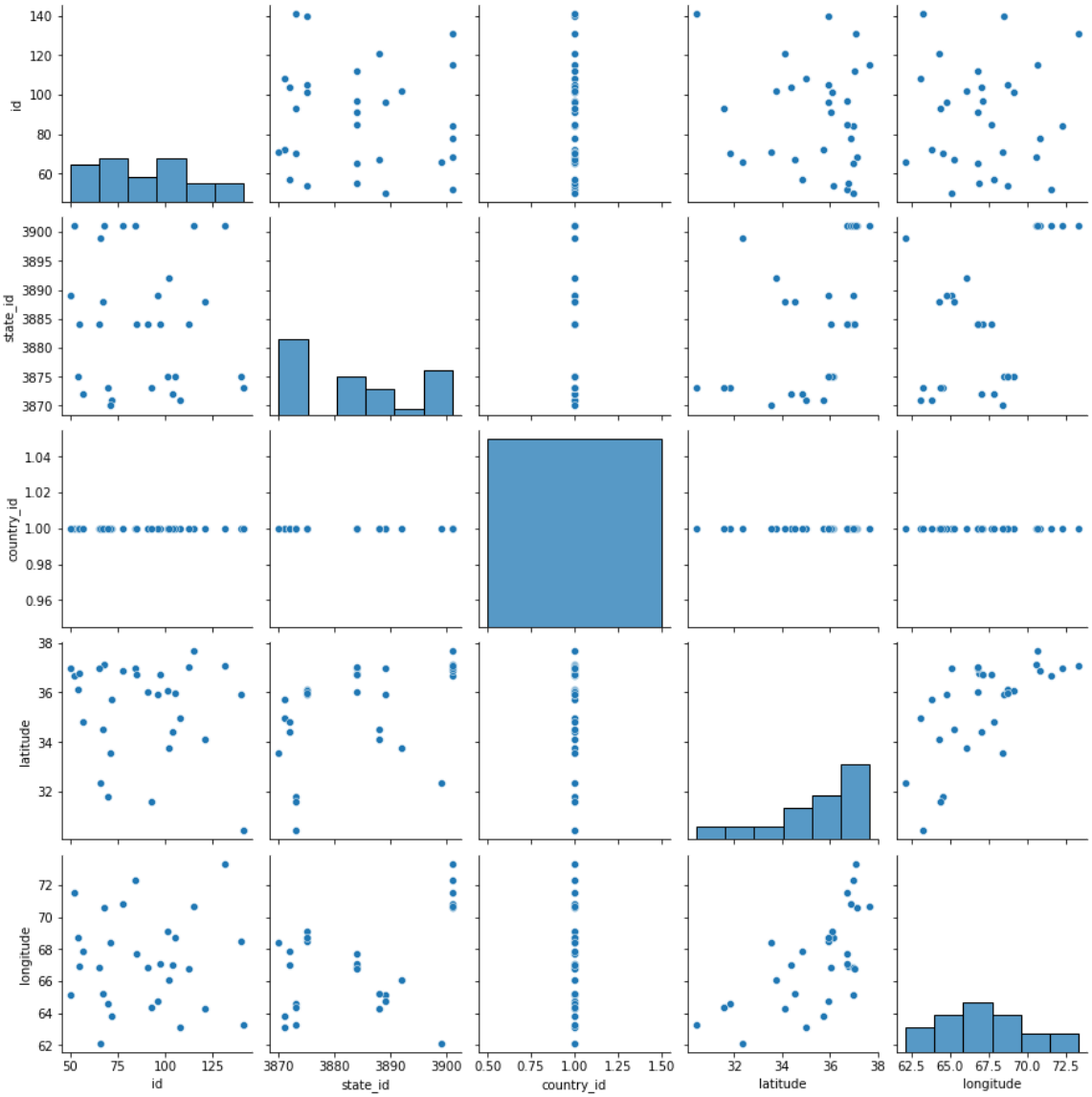
```
Index(['id', 'name', 'state_id', 'state_code', 'state_name', 'country_id',
       'country_code', 'country_name', 'latitude', 'longitude', 'wikiDataI
d'],
      dtype='object')
```

In [7]:

```
sns.pairplot(df)
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x1eb07508d30>
```



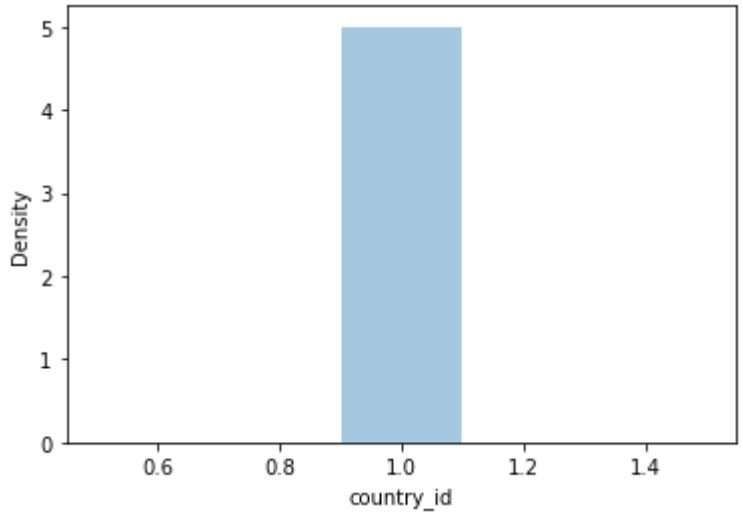
In [8]:

```
sns.distplot(df['country_id'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning: `distplot` is a deprecated function and will be removed in
a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level fun
ction for histograms).
  warnings.warn(msg, FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:306: U
serWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
```

Out[8]:

```
<AxesSubplot:xlabel='country_id', ylabel='Density'>
```

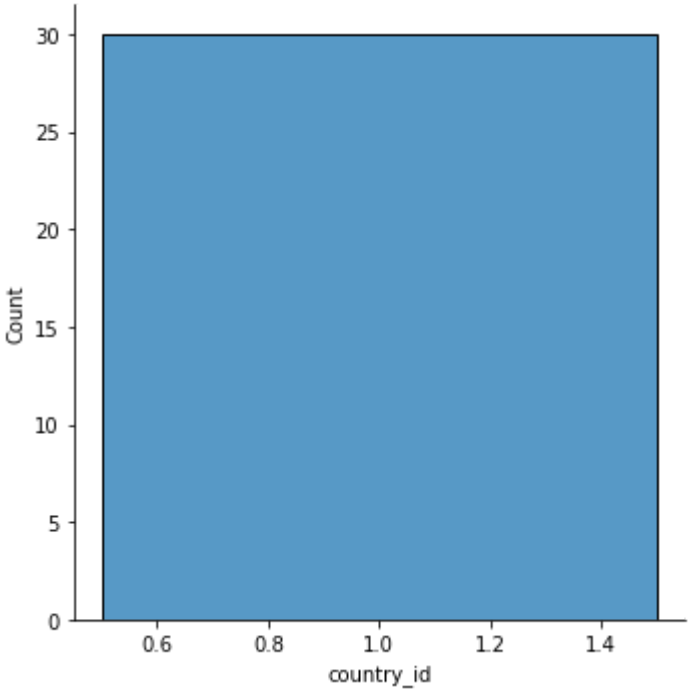


In [9]:

```
sns.displot(df["country_id"])
```

Out[9]:

```
<seaborn.axisgrid.FacetGrid at 0x1eb094d8f70>
```



In [10]:

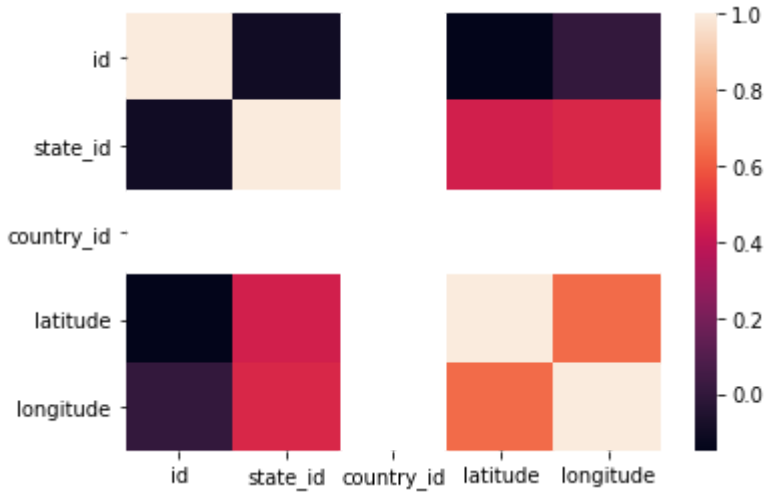
```
df1=df[['id', 'name', 'state_id', 'state_code', 'state_name', 'country_id',
       'country_code', 'country_name', 'latitude', 'longitude', 'wikiDataId']]
```

In [11]:

```
sns.heatmap(df1.corr())
```

Out[11]:

```
<AxesSubplot:>
```



In [12]:

```
x=df1[['id', 'state_id', 'country_id', 'latitude', 'longitude']]
y=df1[['country_id']]
```

In [13]:

```
from sklearn.model_selection import train_test_split
```

In [14]:

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

In [15]:

```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)#ValueError: Input contains NaN, infinity or a value too large for
```

Out[15]:

```
LinearRegression()
```

In [16]:

```
print(lr.intercept_)
```

```
[1.]
```

In [17]:

```
coef= pd.DataFrame(lr.coef_)
coef
```

Out[17]:

	0	1	2	3	4
0	0.0	0.0	0.0	0.0	0.0

In [18]:

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

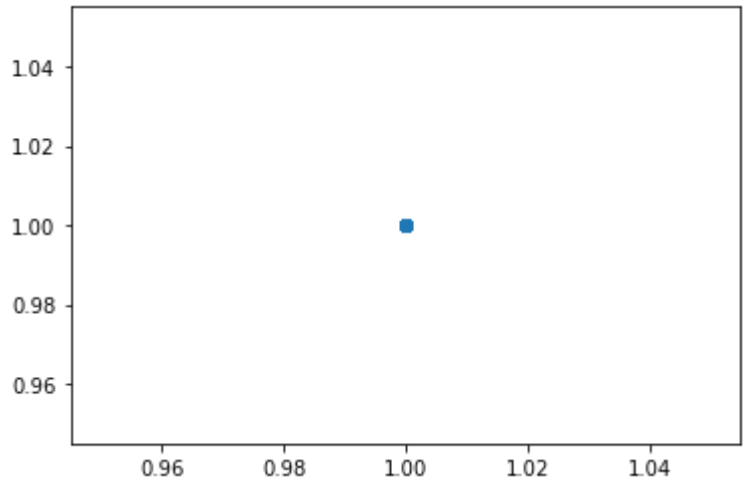
```
1.0
```

In [19]:

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

Out[19]:

```
<matplotlib.collections.PathCollection at 0x1eb09dce520>
```



In [20]:

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

Out[20]:

```
1.0
```

In [21]:

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

Out[21]:

```
1.0
```

In [22]:

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

In [23]:

```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[23]:

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

In [24]:

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

Out[24]:

```
1.0
```

In [25]:

```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinat
e_descent.py:530: ConvergenceWarning: Objective did not converge. You migh
t want to increase the number of iterations. Duality gap: 0.0, tolerance:
0.0
  model = cd_fast.enet_coordinate_descent(
```

Out[25]:

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

In [26]:

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

Out[26]:

```
1.0
```

Elastic Net

In [27]:

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

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinat
e_descent.py:530: ConvergenceWarning: Objective did not converge. You migh
t want to increase the number of iterations. Duality gap: 0.0, tolerance:
0.0
  model = cd_fast.enet_coordinate_descent(
```

Out[27]:

```
ElasticNet()
```

In [28]:

```
print(en.coef_)
```

```
[0. 0. 0. 0. 0.]
```

In [29]:

```
print(en.intercept_)
```

```
[1.]
```

In [30]:

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

```
[1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

In [31]:

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

```
1.0
```

Evaluation Metrics

In [32]:

```
from sklearn import metrics
```

In [33]:

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

```
Mean Absolute Error: 0.0
```

In [34]:

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

```
Mean Squared Error: 0.0
```

In [35]:

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

```
Root Mean Squared Error: 0.0
```

Model Saving

In [36]:

```
import pickle
```

In [37]:

```
filename="prediction2"
pickle.dump(lr,open(filename,'wb'))
```