In [76]:
```
# import libaries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [646]:
```
x=pd.read_csv(r"C:\Users\user\Downloads\22_countries - 22_countries.csv")
```

Out[646]:

| | id | name | iso3 | iso2 | numeric_code | phone_code | capital | currency | currency_na |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | Afghanistan | AFG | AF | 4 | 93 | Kabul | AFN | Afghan afgh |
| **1** | 2 | Aland Islands | ALA | AX | 248 | +358-18 | Mariehamn | EUR | E |
| **2** | 3 | Albania | ALB | AL | 8 | 355 | Tirana | ALL | Albanian |
| **3** | 4 | Algeria | DZA | DZ | 12 | 213 | Algiers | DZD | Algerian d |
| **4** | 5 | American Samoa | ASM | AS | 16 | +1-684 | Pago Pago | USD | US Dc |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **245** | 243 | Wallis And Futuna Islands | WLF | WF | 876 | 681 | Mata Utu | XPF | CFP fr |
| **246** | 244 | Western Sahara | ESH | EH | 732 | 212 | El-Aaiun | MAD | Moroc Dirh |
| **247** | 245 | Yemen | YEM | YE | 887 | 967 | Sanaa | YER | Yemeni |
| **248** | 246 | Zambia | ZMB | ZM | 894 | 260 | Lusaka | ZMW | Zamb kwa |
| **249** | 247 | Zimbabwe | ZWE | ZW | 716 | 263 | Harare | ZWL | Zimbab Dc |

250 rows × 19 columns

In [647]:
```
x=x.head(10)
```

Out[647]:

| | id | name | iso3 | iso2 | numeric_code | phone_code | capital | currency | currency_name |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | Afghanistan | AFG | AF | 4 | 93 | Kabul | AFN | Afghan afghani |
| **1** | 2 | Aland Islands | ALA | AX | 248 | +358-18 | Mariehamn | EUR | Euro |
| **2** | 3 | Albania | ALB | AL | 8 | 355 | Tirana | ALL | Albanian lek |
| **3** | 4 | Algeria | DZA | DZ | 12 | 213 | Algiers | DZD | Algerian dinar |
| **4** | 5 | American Samoa | ASM | AS | 16 | +1-684 | Pago Pago | USD | US Dollar |
| **5** | 6 | Andorra | AND | AD | 20 | 376 | Andorra la Vella | EUR | Euro |
| **6** | 7 | Angola | AGO | AO | 24 | 244 | Luanda | AOA | Angolan kwanza |
| **7** | 8 | Anguilla | AIA | AI | 660 | +1-264 | The Valley | XCD | East Caribbean dollar |
| **8** | 9 | Antarctica | ATA | AQ | 10 | 672 | NaN | AAD | Antarctican dollar |
| **9** | 10 | Antigua And Barbuda | ATG | AG | 28 | +1-268 | St. John's | XCD | Eastern Caribbean dollar |

In [648]:

```
info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 19 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   id               10 non-null     int64  
 1   name             10 non-null     object 
 2   iso3             10 non-null     object 
 3   iso2             10 non-null     object 
 4   numeric_code     10 non-null     int64  
 5   phone_code       10 non-null     object 
 6   capital          9 non-null      object 
 7   currency         10 non-null     object 
 8   currency_name    10 non-null     object 
 9   currency_symbol  10 non-null     object 
 10  tld              10 non-null     object 
 11  native           10 non-null     object 
 12  region           10 non-null     object 
 13  subregion        9 non-null      object 
 14  timezones        10 non-null     object 
 15  latitude         10 non-null     float64
 16  longitude        10 non-null     float64
 17  emoji            10 non-null     object 
 18  emojiU           10 non-null     object 
dtypes: float64(2), int64(2), object(15)
memory usage: 1.6+ KB
```

In [649]:

```
[illegible]
```

Out[649]:

```
Index(['id', 'name', 'iso3', 'iso2', 'numeric_code', 'phone_code', 'capital',
       'currency', 'currency_name', 'currency_symbol', 'tld', 'native',
       'region', 'subregion', 'timezones', 'latitude', 'longitude', 'emoji',
       'emojiU'],
      dtype='object')
```

In [669]:
```
d=x[['id', 'name', 'iso3', 'iso2',]]
```

Out[669]:

|   | id | name | iso3 | iso2 |
|---|---|---|---|---|
| **0** | 1 | Afghanistan | AFG | AF |
| **1** | 2 | Aland Islands | ALA | AX |
| **2** | 3 | Albania | ALB | AL |
| **3** | 4 | Algeria | DZA | DZ |
| **4** | 5 | American Samoa | ASM | AS |
| **5** | 6 | Andorra | AND | AD |
| **6** | 7 | Angola | AGO | AO |
| **7** | 8 | Anguilla | AIA | AI |
| **8** | 9 | Antarctica | ATA | AQ |
| **9** | 10 | Antigua And Barbuda | ATG | AG |

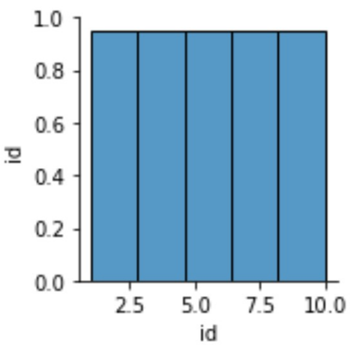
In [670]:
```
[illegible]
```

Out[670]:

|   | id |
|---|---|
| **count** | 10.00000 |
| **mean** | 5.50000 |
| **std** | 3.02765 |
| **min** | 1.00000 |
| **25%** | 3.25000 |
| **50%** | 5.50000 |
| **75%** | 7.75000 |
| **max** | 10.00000 |

In [671]:
```
[illegible]
```

Out[671]:
```
<seaborn.axisgrid.PairGrid at 0x190d7e4b2e0>
```



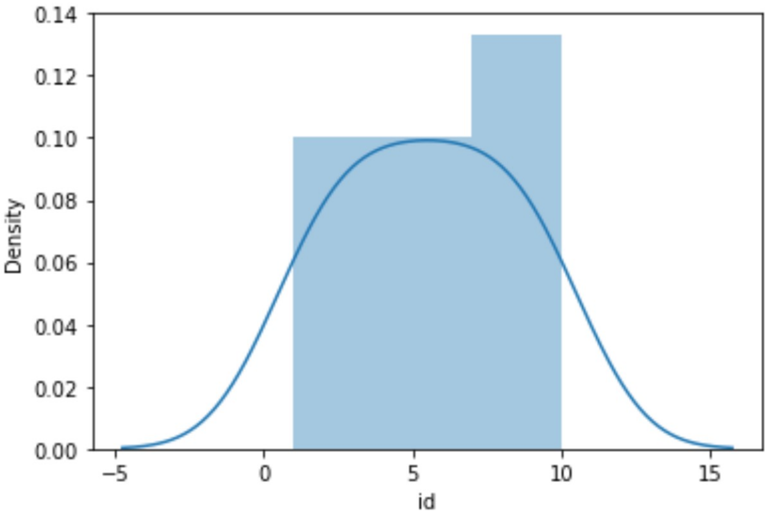
In [673]:

```
sns.distplot(x['id'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level fu
nction with similar flexibility) or `histplot` (an axes-level function for hi
stograms).
  warnings.warn(msg, FutureWarning)
```

Out[673]:

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



In [675]:

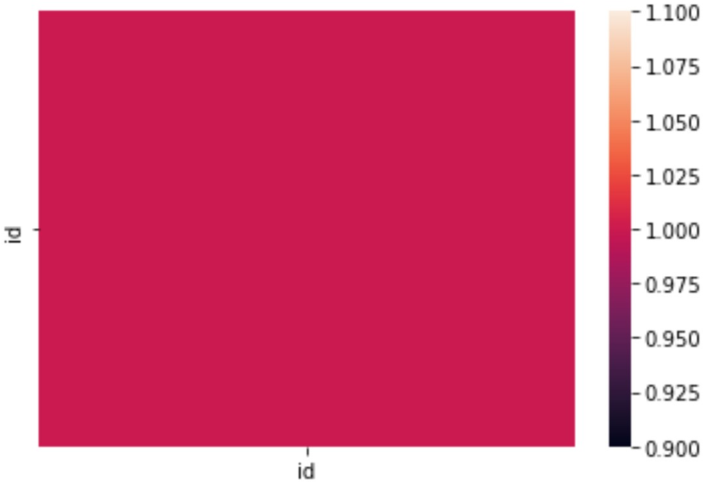
```
x1=x[['id']]
```

In [676]:

```
sns.heatmap(x1.corr())
```

Out[676]:

```
<AxesSubplot:>
```



In [679]:

```
x=x1[['id']]
y=x1['id']
```

In [680]:
```
# to split my dataset into traning and test date

from sklearn.model_selection import train_test_split
```

In [681]:
```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
```

Out[681]:
```
LinearRegression()
```

In [682]:

```
2.6645352591003757e-15
```

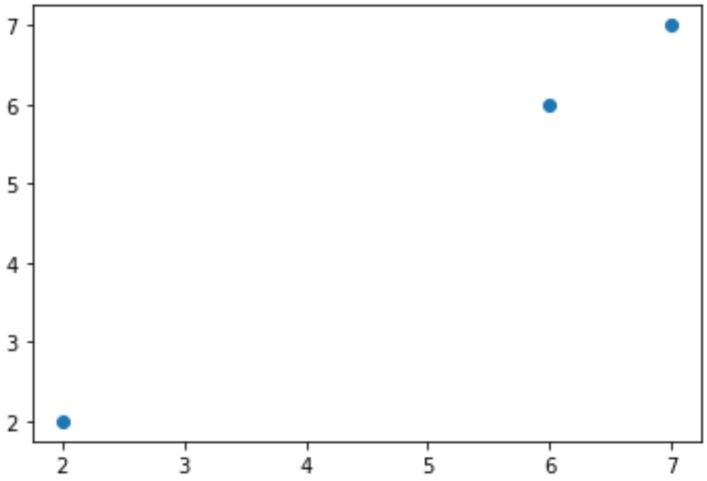
In [683]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[683]:

|  | Co-efficient |
|---|---|
| **id** | 1.0 |

In [684]:
```
prediction=lr.predict(x_test)
```

Out[684]:
```
<matplotlib.collections.PathCollection at 0x190d800e430>
```



In [685]:

Out[685]:
```
1.0
```

In [686]:

Out[686]:
```
1.0
```

In [687]:

```
In [688]: rr=Ridge(alpha=10)
          rr.fit(x_train,y_train)

Out[688]: 0.9814963226068739

In [689]: la=Lasso(alpha=10)

Out[689]: Lasso(alpha=10)

In [690]:

Out[690]: -0.10932944606413986

In [691]: from sklearn.linear_model import ElasticNet
          en=ElasticNet()

Out[691]: ElasticNet()

In [692]:

Out[692]: array([0.90130916])

In [693]:

Out[693]: array([5.97180262, 6.87311178, 2.36656596])

In [694]:

Out[694]: 0.5639476334340383

In [695]:

Out[695]: 0.9891952631157275

In [696]:

In [697]:

          Mean Absolute Error 8.881784197001252e-16

In [698]:

          Mean Squared Error 1.314768175368353e-30

In [699]:

          Root Mean Squared Error 1.1466334093198022e-15

In [700]:

In [701]: filename="prediction"
          pickle.dump(lr,open(filename,'wb'))
```

```
In [702]: import pandas as pd
```

```
In [703]: filename="prediction"
```

```
In [706]: real=[[102],[505]]
```

```
In [707]:

Out[707]: array([102., 505.])
```

```
In [ ]:
```