In [1]:

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

In [2]:

```
data = pd.read_csv("C:/Users/AKASH JOY/Documents/Verzeo DataSciene Major/covid_19_india.csv
```

In [3]:

```
data.columns
```

Out[3]:

```
Index(['Sno', 'Date', 'Time', 'State/UnionTerritory',
       'ConfirmedIndianNational', 'ConfirmedForeignNational', 'Cured',
       'Deaths', 'Confirmed'],
      dtype='object')
```

In [4]:

```
data.head()
```

Out[4]:

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	C
0	1	2020-01-30	6:00 PM	Kerala	1	0	
1	2	2020-01-31	6:00 PM	Kerala	1	0	
2	3	2020-02-01	6:00 PM	Kerala	2	0	
3	4	2020-02-02	6:00 PM	Kerala	3	0	
4	5	2020-02-03	6:00 PM	Kerala	3	0	

In [5]:

```
data.describe()
```

Out[5]:

	Sno	Cured	Deaths	Confirmed
count	15806.000000	1.580600e+04	15806.000000	1.580600e+04
mean	7903.500000	1.986514e+05	3004.846324	2.204181e+05
std	4562.943513	4.299306e+05	7919.358996	4.781429e+05
min	1.000000	0.000000e+00	0.000000	0.000000e+00
25%	3952.250000	2.082000e+03	15.000000	3.291250e+03
50%	7903.500000	2.293950e+04	385.000000	2.927950e+04
75%	11854.750000	2.208698e+05	2690.500000	2.472700e+05
max	15806.000000	5.564348e+06	100470.000000	5.842000e+06

In [6]:

```
data.isnull().sum()
```

Out[6]:

```
Sno                          0
Date                         0
Time                         0
State/UnionTerritory         0
ConfirmedIndianNational      0
ConfirmedForeignNational     0
Cured                        0
Deaths                       0
Confirmed                    0
dtype: int64
```

In [7]:

```
data1 = data
```

In [8]:

```
dates_1 = data1[data1['Sno']>5]
```

In [9]:

```
data1.Date
```

Out[9]:

```
0        2020-01-30
1        2020-01-31
2        2020-02-01
3        2020-02-02
4        2020-02-03
            ...
15801    2021-06-08
15802    2021-06-08
15803    2021-06-08
15804    2021-06-08
15805    2021-06-08
Name: Date, Length: 15806, dtype: object
```

In [10]:

```
data1.Date = pd.to_datetime(data1.Date,utc = True)
```

In [11]:

```
data1['Date']
```

Out[11]:

```
0       2020-01-30 00:00:00+00:00
1       2020-01-31 00:00:00+00:00
2       2020-02-01 00:00:00+00:00
3       2020-02-02 00:00:00+00:00
4       2020-02-03 00:00:00+00:00
                   ...
15801   2021-06-08 00:00:00+00:00
15802   2021-06-08 00:00:00+00:00
15803   2021-06-08 00:00:00+00:00
15804   2021-06-08 00:00:00+00:00
15805   2021-06-08 00:00:00+00:00
Name: Date, Length: 15806, dtype: datetime64[ns, UTC]
```

In [12]:

```
ds=data1[(data1['Date'] > '2020-03-26') & (data1['Date'] <= '2020-04-30')]
```

In [13]:

```
ds
```

Out[13]:

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForei
392	393	2020-03-27 00:00:00+00:00	10:00 AM	Andaman and Nicobar Islands	1	
393	394	2020-03-27 00:00:00+00:00	10:00 AM	Andhra Pradesh	12	
394	395	2020-03-27 00:00:00+00:00	10:00 AM	Bihar	6	
395	396	2020-03-27 00:00:00+00:00	10:00 AM	Chandigarh	7	
396	397	2020-03-27 00:00:00+00:00	10:00 AM	Chhattisgarh	6	
...	...	...	...	...	...	
1473	1474	2020-04-30 00:00:00+00:00	5:00 PM	Telengana	-	
1474	1475	2020-04-30 00:00:00+00:00	5:00 PM	Tripura	-	
1475	1476	2020-04-30 00:00:00+00:00	5:00 PM	Uttarakhand	-	
1476	1477	2020-04-30 00:00:00+00:00	5:00 PM	Uttar Pradesh	-	
1477	1478	2020-04-30 00:00:00+00:00	5:00 PM	West Bengal	-	

1086 rows × 9 columns

In [ ]:

```
```

In [14]:

```
ds
```

Out[14]:

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForei
392	393	2020-03-27 00:00:00+00:00	10:00 AM	Andaman and Nicobar Islands	1	
393	394	2020-03-27 00:00:00+00:00	10:00 AM	Andhra Pradesh	12	
394	395	2020-03-27 00:00:00+00:00	10:00 AM	Bihar	6	
395	396	2020-03-27 00:00:00+00:00	10:00 AM	Chandigarh	7	
396	397	2020-03-27 00:00:00+00:00	10:00 AM	Chhattisgarh	6	
...	...	...	...	...	...	
1473	1474	2020-04-30 00:00:00+00:00	5:00 PM	Telengana	-	
1474	1475	2020-04-30 00:00:00+00:00	5:00 PM	Tripura	-	
1475	1476	2020-04-30 00:00:00+00:00	5:00 PM	Uttarakhand	-	
1476	1477	2020-04-30 00:00:00+00:00	5:00 PM	Uttar Pradesh	-	
1477	1478	2020-04-30 00:00:00+00:00	5:00 PM	West Bengal	-	

1086 rows × 9 columns

In [15]:

```
ds.shape
```

Out[15]:

```
(1086, 9)
```

In [16]:

```
ds.dtypes
```

Out[16]:

```
Sno                                 int64
Date                datetime64[ns, UTC]
Time                             object
State/UnionTerritory             object
ConfirmedIndianNational          object
ConfirmedForeignNational         object
Cured                             int64
Deaths                            int64
Confirmed                         int64
dtype: object
```

Sno O Date I Time I State/UnionTerritory N ConfirmedIndianNational N ConfirmedForeignNational N Cured N Deaths N Confirmed N

Labels are:-

1.Cured

2.Deaths

3.Confirmed

Features are:-

1. State/Union Territory

2. ConfirmedIndianNational

3. ConfirmedForeignNational

In [17]:

```
ds.isnull().sum()
```

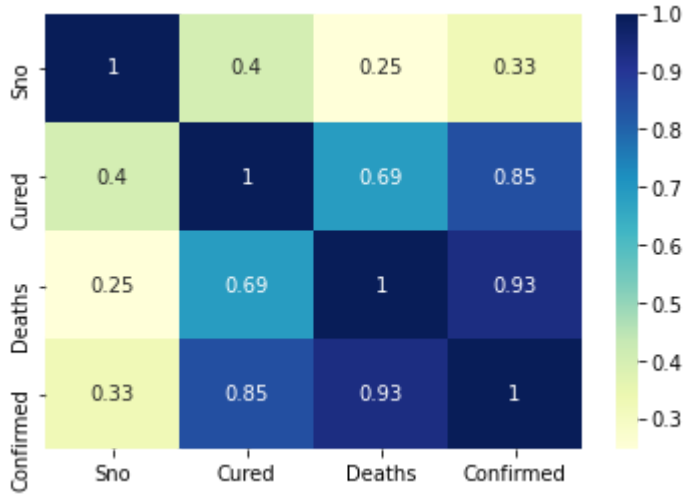
Out[17]:

```
Sno                          0
Date                         0
Time                         0
State/UnionTerritory         0
ConfirmedIndianNational      0
ConfirmedForeignNational     0
Cured                        0
Deaths                       0
Confirmed                    0
dtype: int64
```

The given DataSet does not have any null values

In [18]:

```
dataplot = sb.heatmap(ds.corr(),cmap="YlGnBu",annot = True)
```



from the above correlation its gives and insight:-

Cured is related to Confirmed COVID-19 Cases

Deaths is highly related to Confirmed COVID-19 Cases

Also the number of cases confirmed increased gets the deaths rate increased too

In [19]:

```
ds.drop(['Time','Sno'],axis = 1,inplace =True)
```

```
C:\anaconda\lib\site-packages\pandas\core\frame.py:4163: SettingWithCopyWarn
ing:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  return super().drop(
```

In [20]:

```
ds.columns
```

Out[20]:

```
Index(['Date', 'State/UnionTerritory', 'ConfirmedIndianNational',
       'ConfirmedForeignNational', 'Cured', 'Deaths', 'Confirmed'],
      dtype='object')
```

In [21]:

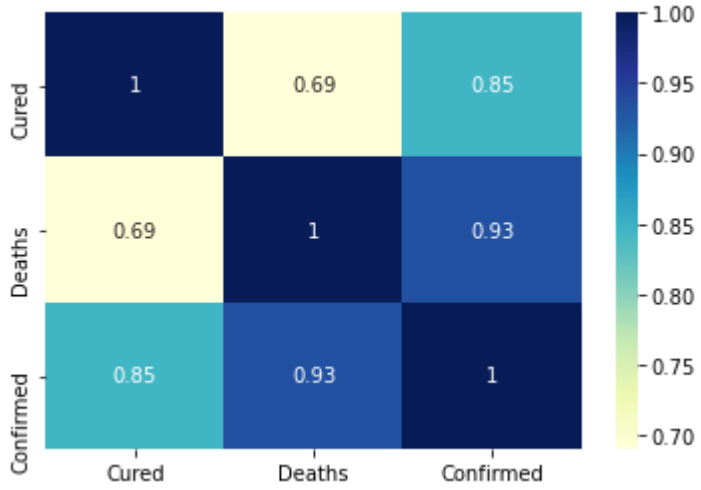
```
ds.shape
```

Out[21]:

```
(1086, 7)
```

In [22]:

```
dataplot = sb.heatmap(ds.corr(),cmap="YlGnBu",annot = True)
```



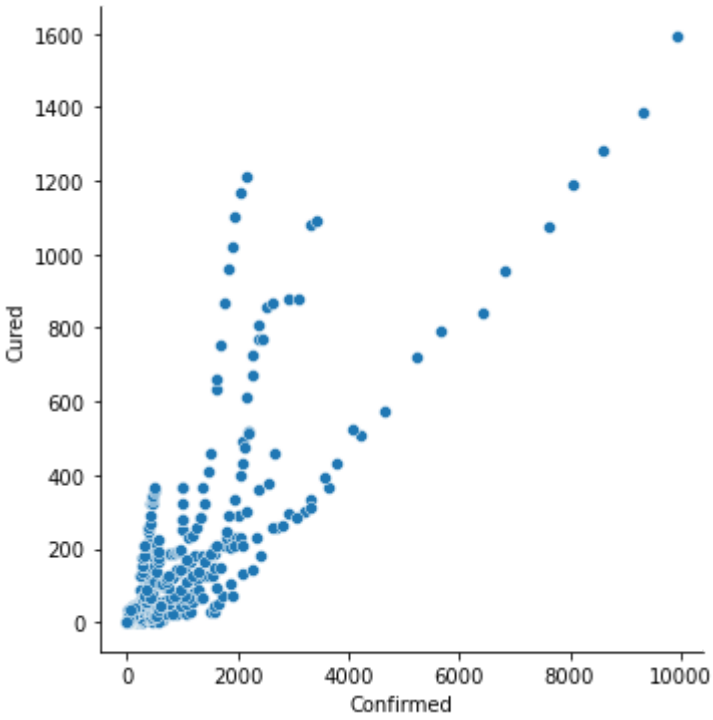
In [ ]:

In [23]:

```
sb.relplot(x="Confirmed",y = "Cured", data = ds)
```

Out[23]:

```
<seaborn.axisgrid.FacetGrid at 0xab994a8>
```

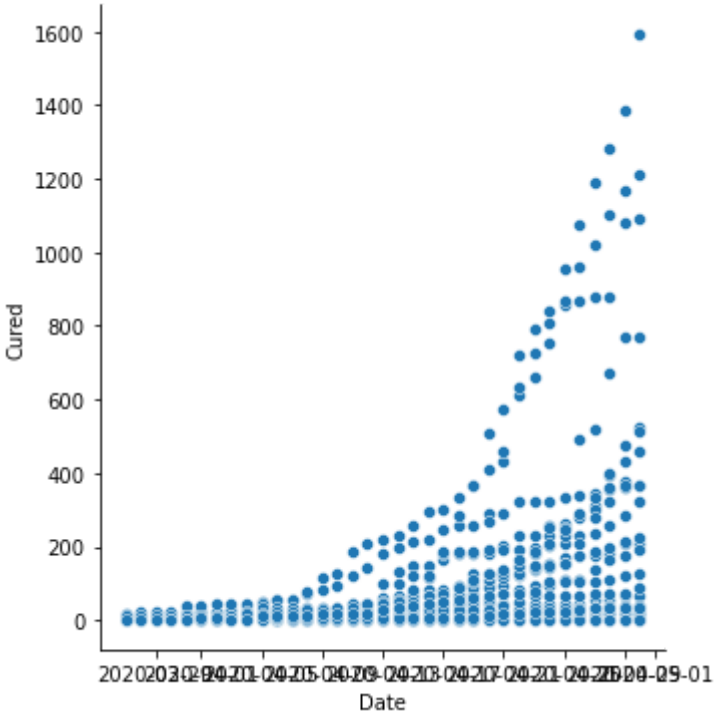


In [24]:

```
sb.relplot(x="Date",y = "Cured", data = ds)
```

Out[24]:

```
<seaborn.axisgrid.FacetGrid at 0xab783d0>
```

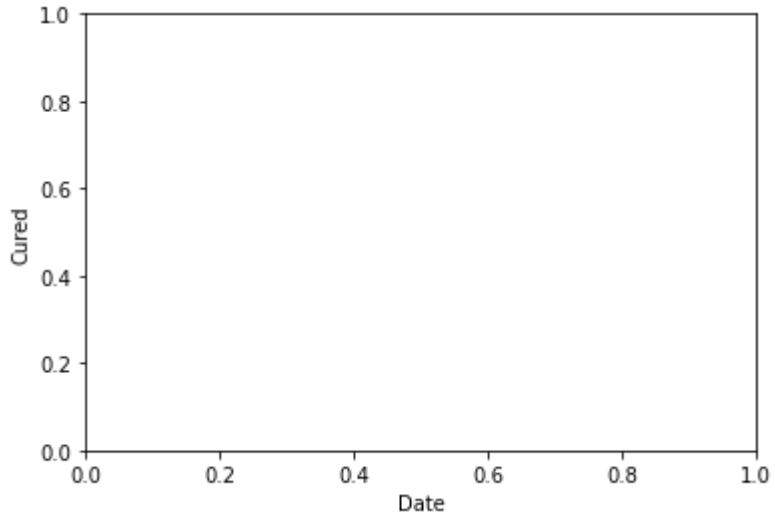


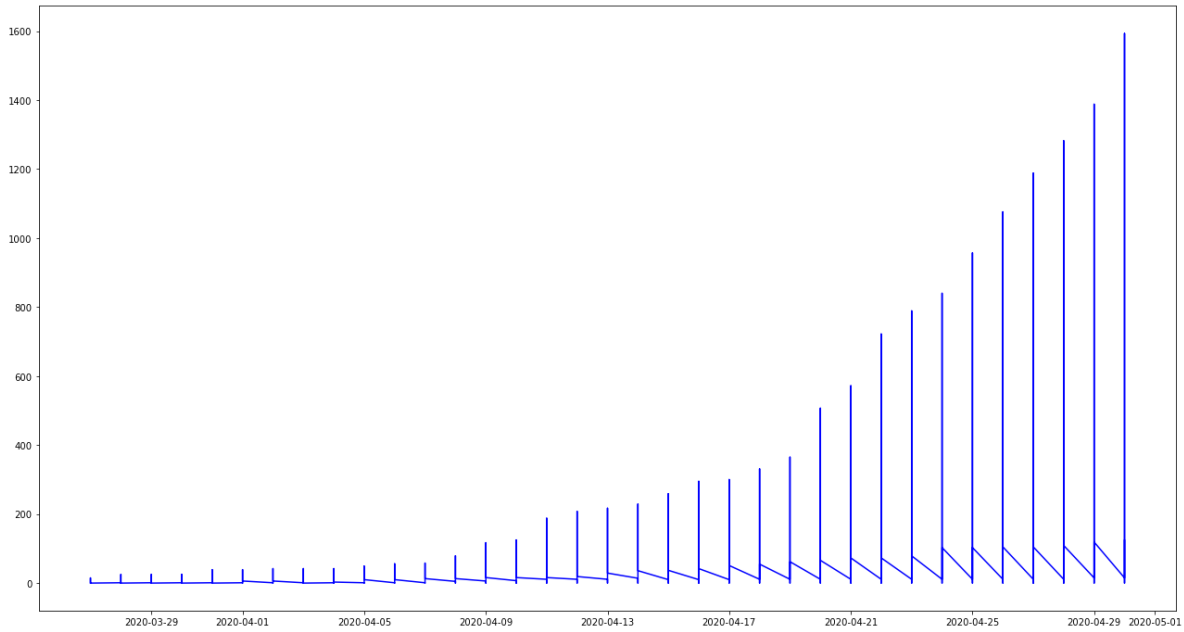
In [44]:

```
X = ds['Date']
Y = ds['Cured']
plt.xlabel("Date")
plt.ylabel("Cured")
fig,ax = plt.subplots(figsize = (22,12))
ax.plot(X,Y,color ='blue',label='Date')
```

Out[44]:

```
[<matplotlib.lines.Line2D at 0xa90d9b8>]
```



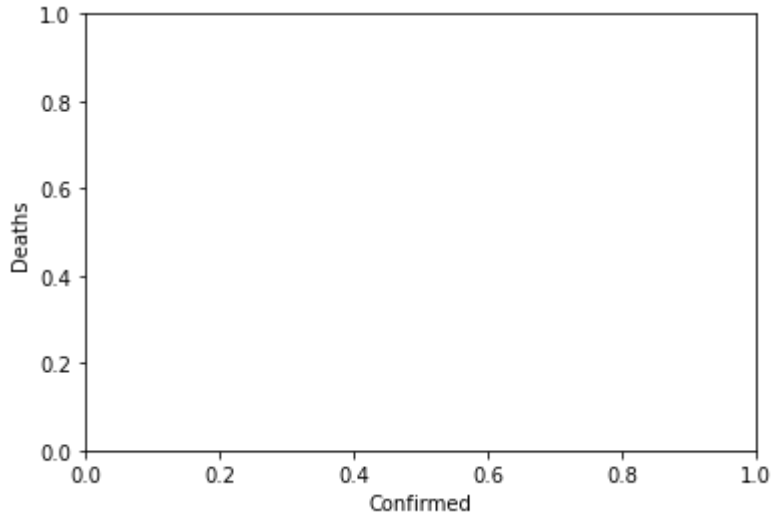


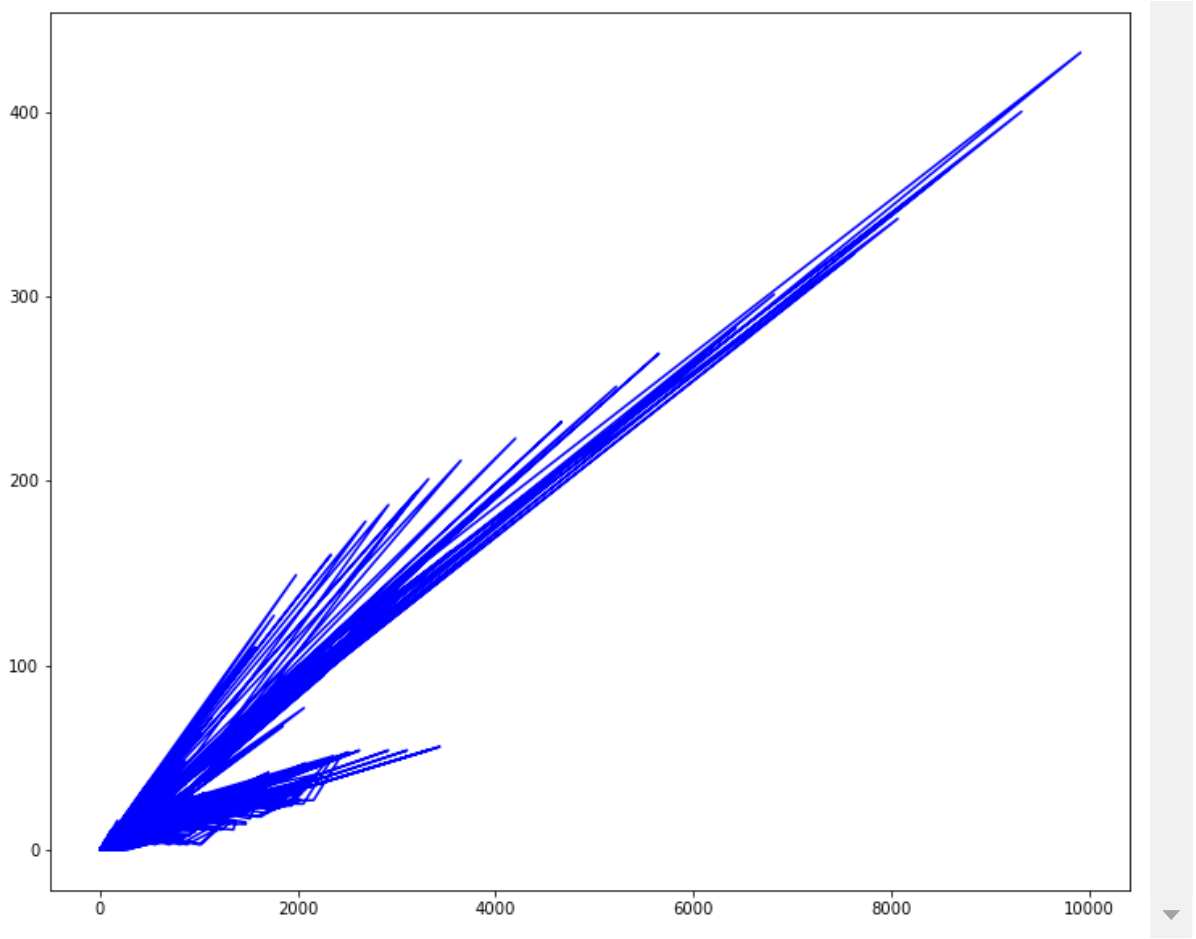
In [45]:

```
X = ds['Confirmed']
Y = ds['Deaths']
plt.xlabel("Confirmed")
plt.ylabel("Deaths")
fig,ax = plt.subplots(figsize = (12,10))
ax.plot(X,Y,color ='blue',label='Date')
```

Out[45]:

```
[<matplotlib.lines.Line2D at 0xb05640>]
```



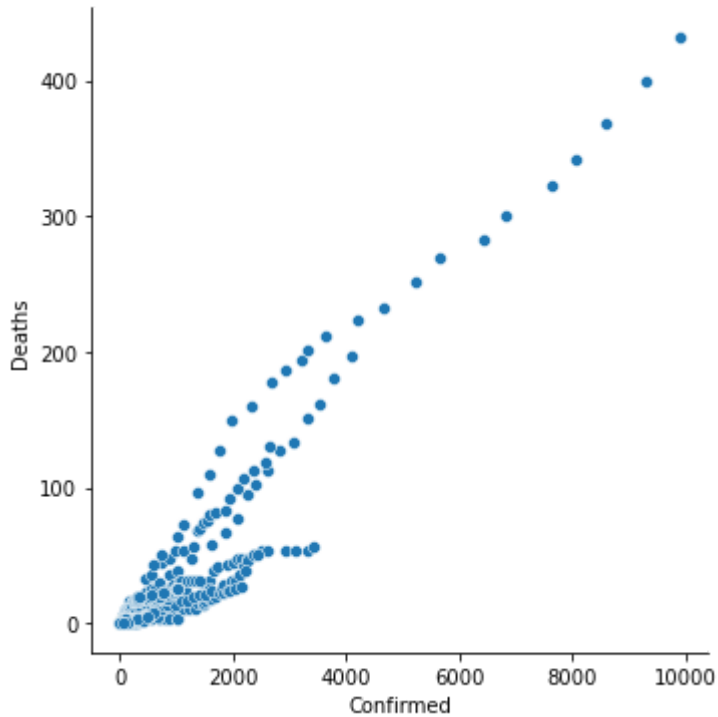


In [46]:

```
sb.relplot(x="Confirmed",y = "Deaths", data = ds)
```

Out[46]:

```
<seaborn.axisgrid.FacetGrid at 0xa91b400>
```

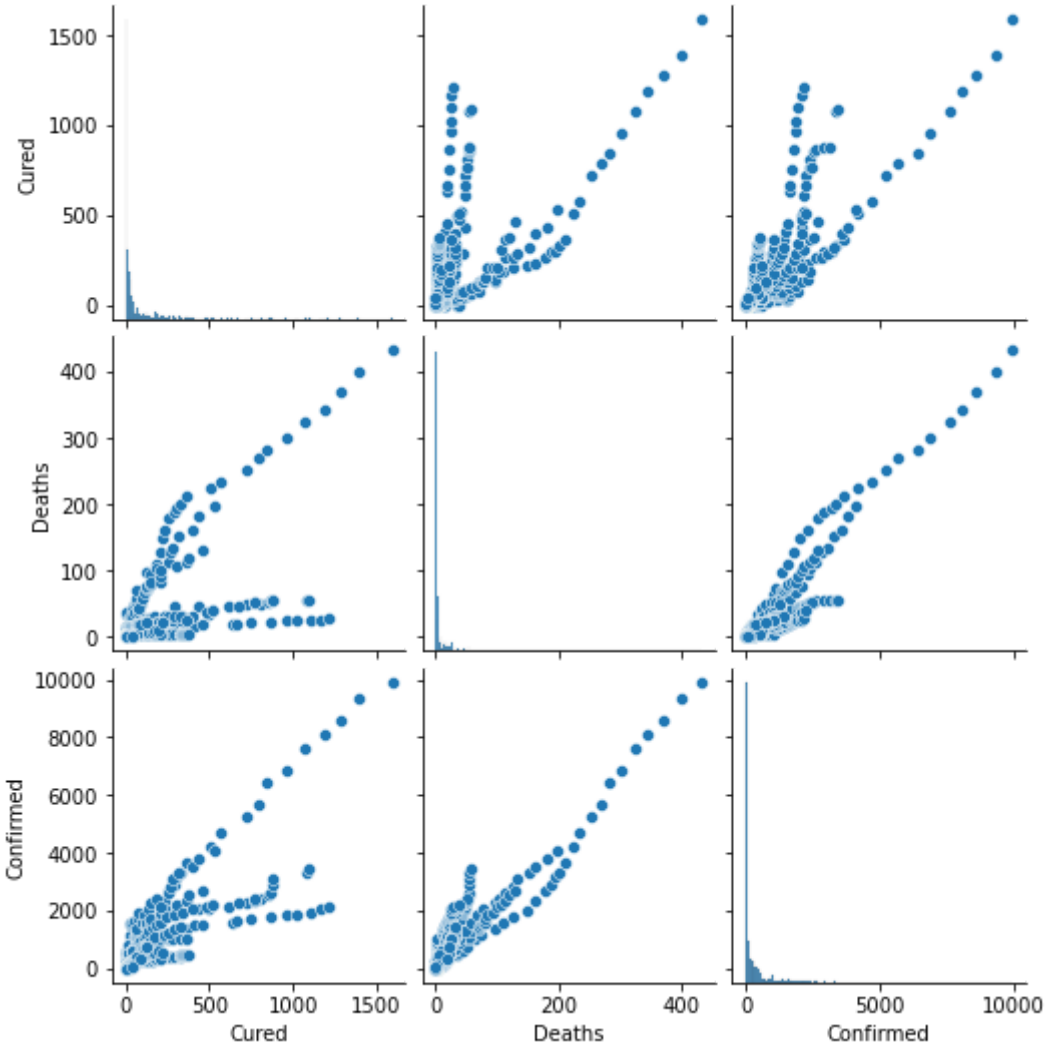


In [47]:

```
sb.pairplot(ds)
```

Out[47]:

```
<seaborn.axisgrid.PairGrid at 0xa91b6d0>
```



In [48]:

```
ds.columns
```

Out[48]:

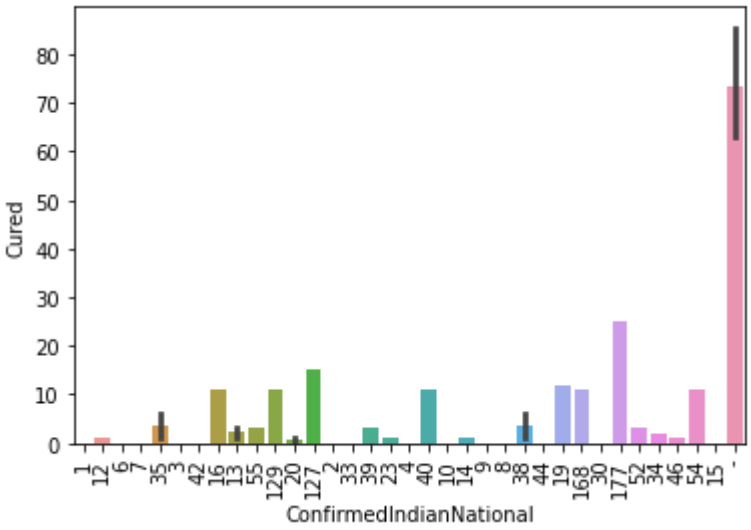
```
Index(['Date', 'State/UnionTerritory', 'ConfirmedIndianNational',
       'ConfirmedForeignNational', 'Cured', 'Deaths', 'Confirmed'],
      dtype='object')
```

In [49]:

```
ax=sb.barplot(x =ds['ConfirmedIndianNational'],y = ds['Cured'])
plt.xticks(rotation = 90)
plt.figure(figsize = (42,12))
```

Out[49]:

```
<Figure size 3024x864 with 0 Axes>
```



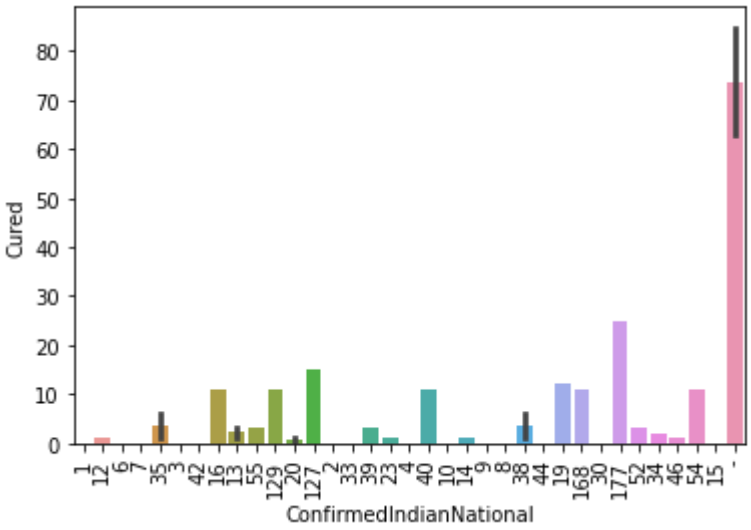

```
<Figure size 3024x864 with 0 Axes>
```

In [50]:

```
ax=sb.barplot(x =ds['ConfirmedIndianNational'],y = ds['Cured'])
plt.xticks(rotation = 90)
plt.figure(figsize = (42,12))
```

Out[50]:

```
<Figure size 3024x864 with 0 Axes>
```



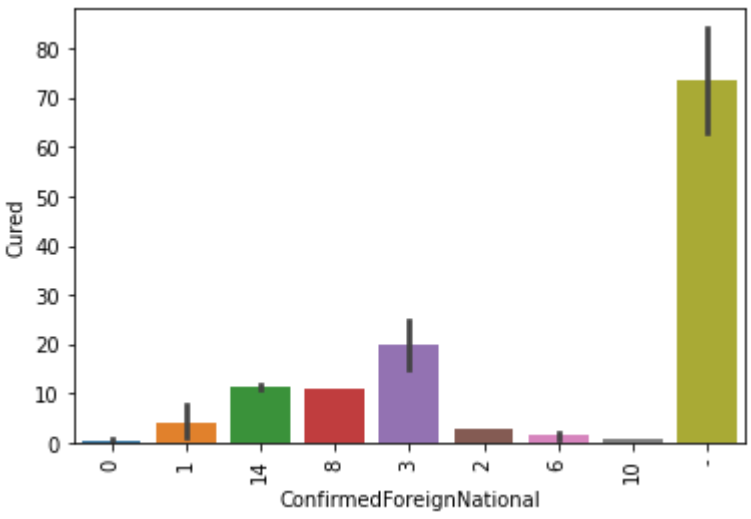

```
<Figure size 3024x864 with 0 Axes>
```

In [51]:

```
ax=sb.barplot(x =ds['ConfirmedForeignNational'],y = ds['Cured'])
plt.xticks(rotation = 90)
plt.figure(figsize = (42,12))
```

Out[51]:

```
<Figure size 3024x864 with 0 Axes>
```



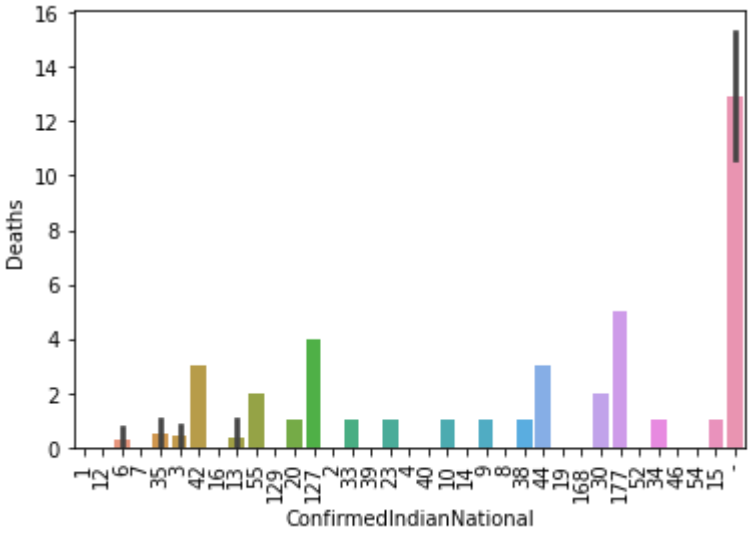

```
<Figure size 3024x864 with 0 Axes>
```

In [52]:

```
ax=sb.barplot(x =ds['ConfirmedIndianNational'],y = ds['Deaths'])
plt.xticks(rotation = 90)
plt.figure(figsize = (42,12))
```

Out[52]:

```
<Figure size 3024x864 with 0 Axes>
```



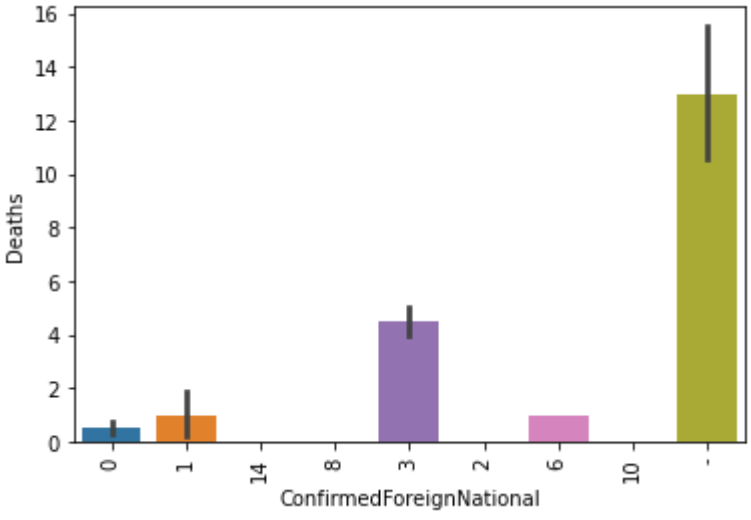

```
<Figure size 3024x864 with 0 Axes>
```

In [53]:

```
ax=sb.barplot(x =ds['ConfirmedForeignNational'],y = ds['Deaths'])
plt.xticks(rotation = 90)
plt.figure(figsize = (42,12))
```

Out[53]:

```
<Figure size 3024x864 with 0 Axes>
```




```
<Figure size 3024x864 with 0 Axes>
```

In [54]:

```
ds['State/UnionTerritory']
```

Out[54]:

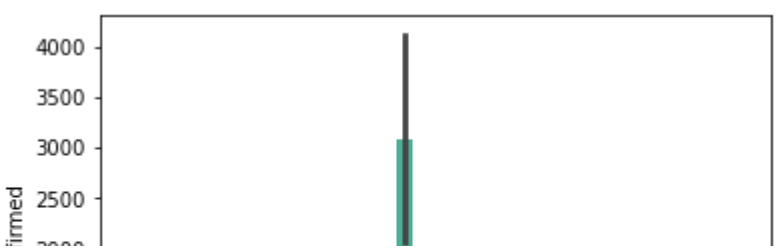
```
392     Andaman and Nicobar Islands
393                  Andhra Pradesh
394                           Bihar
395                      Chandigarh
396                    Chhattisgarh
                  ...
1473                       Telengana
1474                         Tripura
1475                     Uttarakhand
1476                   Uttar Pradesh
1477                     West Bengal
Name: State/UnionTerritory, Length: 1086, dtype: object
```

In [55]:

```
sb.barplot(x = 'State/UnionTerritory',y = 'Confirmed',data = ds)
plt.xticks(rotation = 90)
```

Out[55]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 3
3]),
 [Text(0, 0, 'Andaman and Nicobar Islands'),
  Text(1, 0, 'Andhra Pradesh'),
  Text(2, 0, 'Bihar'),
  Text(3, 0, 'Chandigarh'),
  Text(4, 0, 'Chhattisgarh'),
  Text(5, 0, 'Delhi'),
  Text(6, 0, 'Goa'),
  Text(7, 0, 'Gujarat'),
  Text(8, 0, 'Haryana'),
  Text(9, 0, 'Himachal Pradesh'),
  Text(10, 0, 'Jammu and Kashmir'),
  Text(11, 0, 'Karnataka'),
  Text(12, 0, 'Kerala'),
  Text(13, 0, 'Ladakh'),
  Text(14, 0, 'Madhya Pradesh'),
  Text(15, 0, 'Maharashtra'),
  Text(16, 0, 'Manipur'),
  Text(17, 0, 'Mizoram'),
  Text(18, 0, 'Odisha'),
  Text(19, 0, 'Puducherry'),
  Text(20, 0, 'Punjab'),
  Text(21, 0, 'Rajasthan'),
  Text(22, 0, 'Tamil Nadu'),
  Text(23, 0, 'Telengana'),
  Text(24, 0, 'Uttarakhand'),
  Text(25, 0, 'Uttar Pradesh'),
  Text(26, 0, 'West Bengal'),
  Text(27, 0, 'Unassigned'),
  Text(28, 0, 'Assam'),
  Text(29, 0, 'Jharkhand'),
  Text(30, 0, 'Arunachal Pradesh'),
  Text(31, 0, 'Tripura'),
  Text(32, 0, 'Nagaland'),
  Text(33, 0, 'Meghalaya')])
```



From The Above Bar Chart:-

We can observe that the states like Maharashtra has more than 500 Cases!

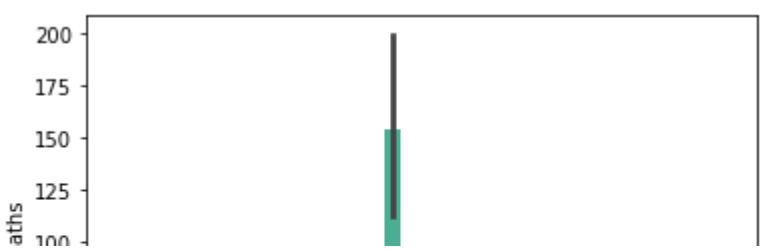
while there are States/Union-Territory like Meghalaya,Nagaland,Tripura,Arunachal Pradesh that has very less Covid-19 Cases!

In [56]:

```
sb.barplot(x = 'State/UnionTerritory',y = 'Deaths',data = ds)
plt.xticks(rotation = 90)
```

Out[56]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 3
3]),
 [Text(0, 0, 'Andaman and Nicobar Islands'),
  Text(1, 0, 'Andhra Pradesh'),
  Text(2, 0, 'Bihar'),
  Text(3, 0, 'Chandigarh'),
  Text(4, 0, 'Chhattisgarh'),
  Text(5, 0, 'Delhi'),
  Text(6, 0, 'Goa'),
  Text(7, 0, 'Gujarat'),
  Text(8, 0, 'Haryana'),
  Text(9, 0, 'Himachal Pradesh'),
  Text(10, 0, 'Jammu and Kashmir'),
  Text(11, 0, 'Karnataka'),
  Text(12, 0, 'Kerala'),
  Text(13, 0, 'Ladakh'),
  Text(14, 0, 'Madhya Pradesh'),
  Text(15, 0, 'Maharashtra'),
  Text(16, 0, 'Manipur'),
  Text(17, 0, 'Mizoram'),
  Text(18, 0, 'Odisha'),
  Text(19, 0, 'Puducherry'),
  Text(20, 0, 'Punjab'),
  Text(21, 0, 'Rajasthan'),
  Text(22, 0, 'Tamil Nadu'),
  Text(23, 0, 'Telengana'),
  Text(24, 0, 'Uttarakhand'),
  Text(25, 0, 'Uttar Pradesh'),
  Text(26, 0, 'West Bengal'),
  Text(27, 0, 'Unassigned'),
  Text(28, 0, 'Assam'),
  Text(29, 0, 'Jharkhand'),
  Text(30, 0, 'Arunachal Pradesh'),
  Text(31, 0, 'Tripura'),
  Text(32, 0, 'Nagaland'),
  Text(33, 0, 'Meghalaya')])
```



From The Above Bar Chart:-

We can observe that the states like Maharashtra has more than 500 Death Cases!

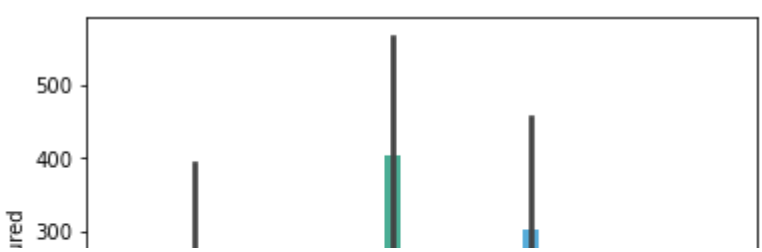
while there are States/Union-Territory like Meghalaya,Nagaland,Tripura,Arunachal Pradesh that has very less Death in Covid-19 Cases!

In [57]:

```
sb.barplot(x = 'State/UnionTerritory',y = 'Cured',data = ds)
plt.xticks(rotation = 90)
```

Out[57]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 3
3]),
 [Text(0, 0, 'Andaman and Nicobar Islands'),
  Text(1, 0, 'Andhra Pradesh'),
  Text(2, 0, 'Bihar'),
  Text(3, 0, 'Chandigarh'),
  Text(4, 0, 'Chhattisgarh'),
  Text(5, 0, 'Delhi'),
  Text(6, 0, 'Goa'),
  Text(7, 0, 'Gujarat'),
  Text(8, 0, 'Haryana'),
  Text(9, 0, 'Himachal Pradesh'),
  Text(10, 0, 'Jammu and Kashmir'),
  Text(11, 0, 'Karnataka'),
  Text(12, 0, 'Kerala'),
  Text(13, 0, 'Ladakh'),
  Text(14, 0, 'Madhya Pradesh'),
  Text(15, 0, 'Maharashtra'),
  Text(16, 0, 'Manipur'),
  Text(17, 0, 'Mizoram'),
  Text(18, 0, 'Odisha'),
  Text(19, 0, 'Puducherry'),
  Text(20, 0, 'Punjab'),
  Text(21, 0, 'Rajasthan'),
  Text(22, 0, 'Tamil Nadu'),
  Text(23, 0, 'Telengana'),
  Text(24, 0, 'Uttarakhand'),
  Text(25, 0, 'Uttar Pradesh'),
  Text(26, 0, 'West Bengal'),
  Text(27, 0, 'Unassigned'),
  Text(28, 0, 'Assam'),
  Text(29, 0, 'Jharkhand'),
  Text(30, 0, 'Arunachal Pradesh'),
  Text(31, 0, 'Tripura'),
  Text(32, 0, 'Nagaland'),
  Text(33, 0, 'Meghalaya')])
```



From The Above Bar Chart:-

We can observe that the states like Maharashtra has more than 500 cases Cured!

while there are States/Union-Territory like Meghalaya,Nagaland,Tripura,Arunachal Pradesh that has very less cured level of Covid Cases!

So from the above Visualization/Chart we can conclude that the states like Maharashtra,Delhi,Goa need higher precauntionary measures against Covid-19

While the states like Meghalaya,Nagaland,Tripura,Arunachal Pradesh are stable/Green regions so no higher precauntionary measures against Covid-19 is required

In [ ]:

```
```

In [ ]:

```
ds.columns
```

In [26]:

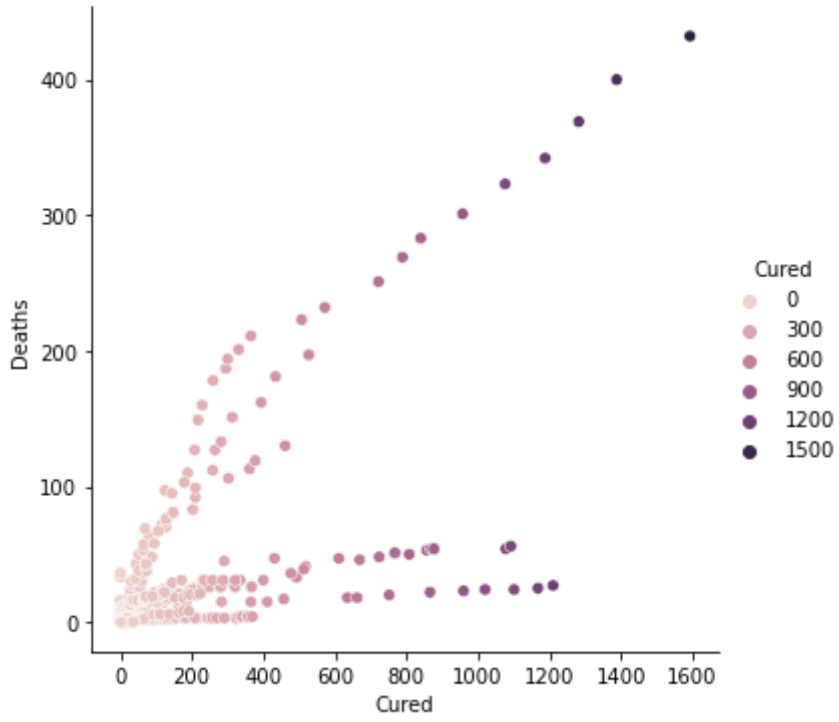
```
X = ds.drop(['Cured'],axis = 1)
Y = ds['Cured']
```

In [27]:

```
sb.relplot(x="Cured",y = "Deaths", hue = "Cured",data = ds)
```

Out[27]:

```
<seaborn.axisgrid.FacetGrid at 0x5cddf88>
```



So, The Number of Death Cases is also Rising with the covid-19 Cases

But there is also a Rise in Cure Cases of Covid-19 which is a point to be Observed!!!

In [28]:

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

In [29]:

```
X = ds.drop(['Date','Cured','State/UnionTerritory','ConfirmedIndianNational','ConfirmedFore
Y = ds['Cured']
```

In [30]:

```
ds.dtypes
```

Out[30]:

```
Date                        datetime64[ns, UTC]
State/UnionTerritory                     object
ConfirmedIndianNational                  object
ConfirmedForeignNational                 object
Cured                                     int64
Deaths                                    int64
Confirmed                                 int64
dtype: object
```

In [31]:

```
X
```

Out[31]:

	Deaths	Confirmed
392	0	1
393	0	12
394	1	6
395	0	7
396	0	6
...	...	...
1473	26	1012
1474	0	2
1475	0	55
1476	39	2203
1477	22	758

1086 rows × 2 columns

In [32]:

```
Y
```

Out[32]:

```
392       0
393       1
394       0
395       0
396       0
       ...
1473    367
1474      2
1475     36
1476    513
1477    124
Name: Cured, Length: 1086, dtype: int64
```

In [33]:

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.3)
```

In [34]:

```
print(X_train.shape)
print(Y_train.shape)
print(X_test.shape)
print(Y_test.shape)
```

```
(760, 2)
(760,)
(326, 2)
(326,)
```

In [35]:

```
print(X)
```

```
      Deaths  Confirmed
392        0          1
393        0         12
394        1          6
395        0          7
396        0          6
...      ...        ...
1473      26       1012
1474       0          2
1475       0         55
1476      39       2203
1477      22        758

[1086 rows x 2 columns]
```

In [36]:

```
from sklearn.linear_model import RidgeCV
model = RidgeCV(cv=2)
model.fit(X_train, Y_train)
```

Out[36]:

```
RidgeCV(alphas=array([ 0.1,  1. , 10. ]), cv=2)
```

In [40]:

```
predict = model.predict(X_test)
print("{}".format(np.linalg.norm(predict - Y_test, 1)*2/len(Y_test)))
```

```
70.366441049453
```

The number of each cases were approx 70 new cases each day starting from March 26th to April 30th

In [ ]:

```

```