In [1]:

```
import pandas as pd
```

In [2]:

```
covid = pd.read_csv('covid.csv')
```

In [3]:

```
#Having a glance at some of the records
covid.head()
```

Out[3]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	total_
0	ABW	Aruba	2020-03-13	2	2	0	0	18.733	18.733	
1	ABW	Aruba	2020-03-20	4	2	0	0	37.465	18.733	
2	ABW	Aruba	2020-03-24	12	8	0	0	112.395	74.930	
3	ABW	Aruba	2020-03-25	17	5	0	0	159.227	46.831	
4	ABW	Aruba	2020-03-26	19	2	0	0	177.959	18.733	

5 rows × 32 columns

In [4]:

```
#Looking at the shape
covid.shape
```

Out[4]:

```
(19496, 32)
```

In [ ]:

```
covid.columns
```

In [5]:

```
#Looking at the different locations
covid["location"].value_counts()
```

Out[5]:

```
Canada            146
United States     146
Japan             146
Mexico            146
China             146
                 ...
Yemen              45
Western Sahara     29
Tajikistan         24
Comoros            23
Lesotho            10
Name: location, Length: 212, dtype: int64
```

In [12]:

```
#Checking if columns have null values
```

```
#Checking if columns have null values
covid.isna().any()
```

Out[12]:

```
iso_code                              True
location                             False
date                                 False
total_cases                          False
new_cases                            False
total_deaths                         False
new_deaths                           False
total_cases_per_million               True
new_cases_per_million                 True
total_deaths_per_million              True
new_deaths_per_million                True
total_tests                           True
new_tests                             True
total_tests_per_thousand              True
new_tests_per_thousand                True
new_tests_smoothed                    True
new_tests_smoothed_per_thousand       True
tests_units                           True
stringency_index                      True
population                            True
population_density                    True
median_age                            True
aged_65_older                         True
aged_70_older                         True
gdp_per_capita                        True
extreme_poverty                       True
cvd_death_rate                        True
diabetes_prevalence                   True
female_smokers                        True
male_smokers                          True
handwashing_facilities                True
hospital_beds_per_100k                True
dtype: bool
```

In [13]:

```
#Getting the sum of null values across each column
covid.isna().sum()
```

Out[13]:

```
iso_code                             64
location                              0
date                                  0
total_cases                           0
new_cases                             0
total_deaths                          0
new_deaths                            0
total_cases_per_million             377
new_cases_per_million               377
total_deaths_per_million            377
new_deaths_per_million              377
total_tests                       14332
new_tests                         14904
total_tests_per_thousand          14332
new_tests_per_thousand            14904
new_tests_smoothed                13866
new_tests_smoothed_per_thousand   13866
tests_units                       13267
stringency_index                   4500
population                           64
population_density                  850
median_age                         1743
aged_65_older                      1980
aged_70_older                      1832
gdp_per_capita                     1982
extreme_poverty                    7878
cvd_death_rate                     1817
diabetes_prevalence                1174
female_smokers                     5052
male_smokers                       5206
```

```
handwashing_facilities              11822
hospital_beds_per_100k               3160
dtype: int64
```

In [14]:

```
#Getting the cases in India
india_case=covid[covid["location"]=="India"]
```

In [15]:

```
india_case.head()
```

Out[15]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	tot
8379	IND	India	2019-12-31	0	0	0	0	0.0	0.0	
8380	IND	India	2020-01-01	0	0	0	0	0.0	0.0	
8381	IND	India	2020-01-02	0	0	0	0	0.0	0.0	
8382	IND	India	2020-01-03	0	0	0	0	0.0	0.0	
8383	IND	India	2020-01-04	0	0	0	0	0.0	0.0	

5 rows × 32 columns

In [16]:

```
india_case.tail()
```

Out[16]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	tot
8519	IND	India	2020-05-20	106750	5611	3303	140	77.355	4.066	
8520	IND	India	2020-05-21	112359	5609	3435	132	81.419	4.064	
8521	IND	India	2020-05-22	118447	6088	3583	148	85.831	4.412	
8522	IND	India	2020-05-23	125101	6654	3720	137	90.653	4.822	
8523	IND	India	2020-05-24	131868	6767	3867	147	95.556	4.904	

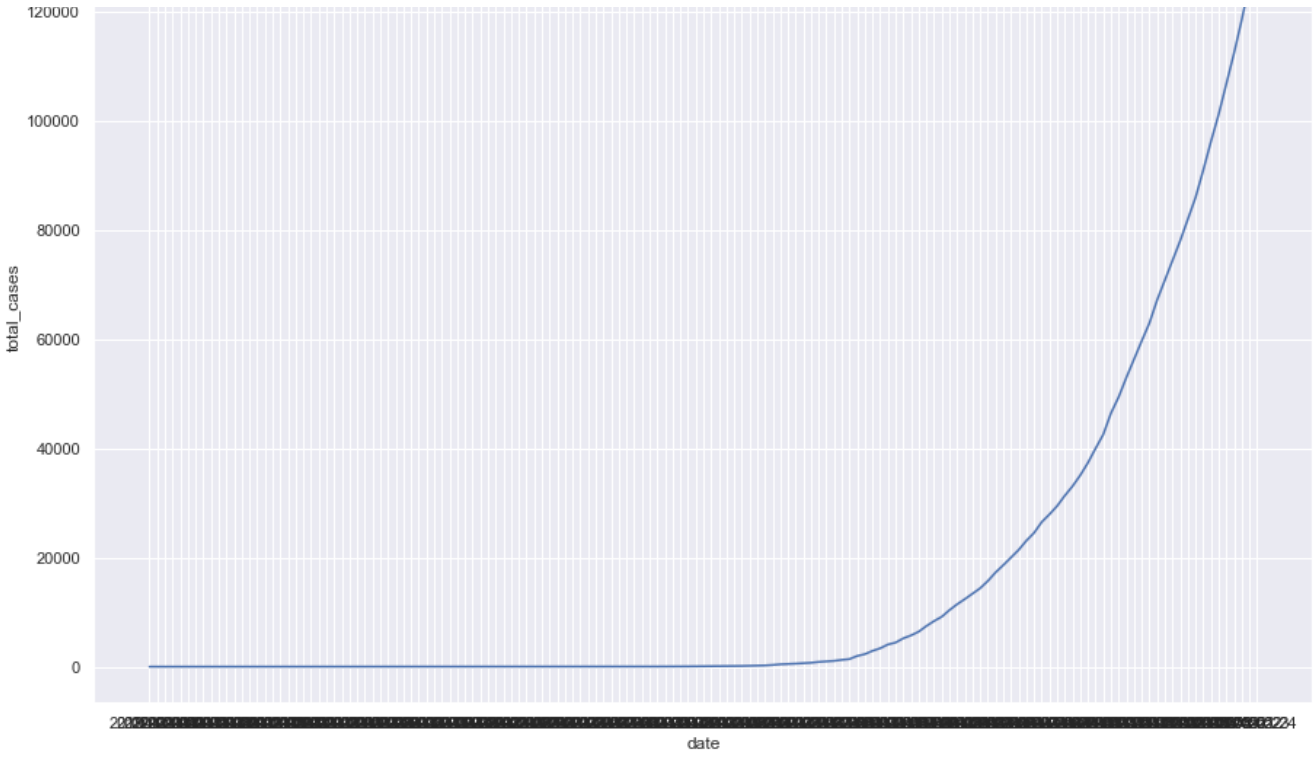
5 rows × 32 columns

In [17]:

```
import seaborn as sns
from matplotlib import pyplot as plt
```

In [18]:

```
#Total cases per day
sns.set(rc={'figure.figsize':(15,10)})
sns.lineplot(x="date",y="total_cases",data=india_case)
plt.show()
```

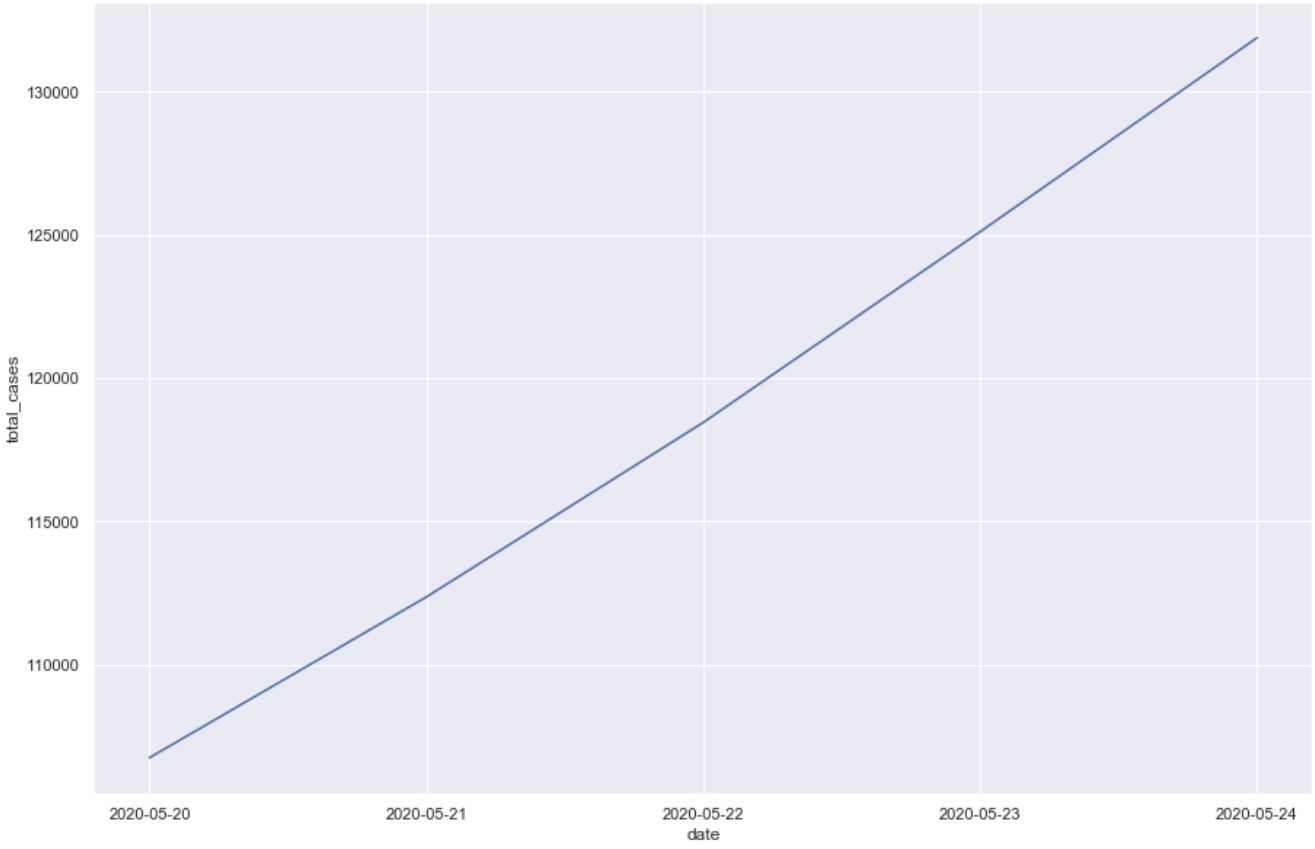


In [19]:

```
#Making a dataframe for last 5 days
india_last_5_days=india_case.tail()
```

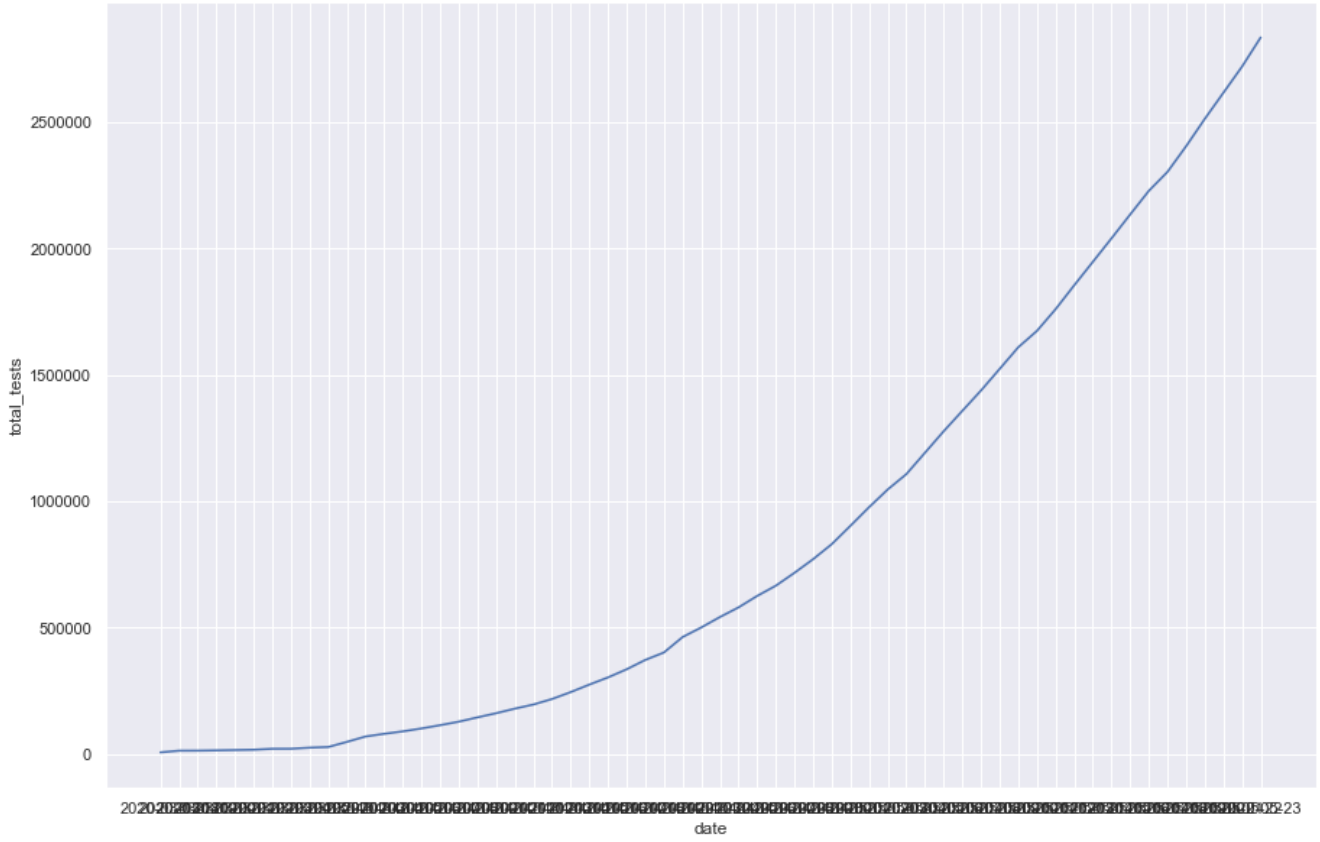
In [20]:

```
#Total cases in last 5 days
sns.set(rc={'figure.figsize':(3,3)})
sns.lineplot(x="date",y="total_cases",data=india_last_5_days)
plt.show()
```



In [21]:

```
#Total tests per day
sns.set(rc={'figure.figsize':(15,10)})
sns.lineplot(x="date",y="total_tests",data=india_case)
plt.show()
```



In [22]:

```
#Total tests in last 5 days
sns.set(rc={'figure.figsize':(15,10)})
sns.lineplot(x="date",y="total_tests",data=india_last_5_days)
plt.show()
```

In [23]:

```
#Brazil Case
brazil_case=covid[covid["location"]=="Brazil"]
brazil_case.head()
```

Out[23]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	tot
2510	BRA	Brazil	2019-12-31	0	0	0	0	0.0	0.0	
2511	BRA	Brazil	2020-01-01	0	0	0	0	0.0	0.0	
2512	BRA	Brazil	2020-01-02	0	0	0	0	0.0	0.0	
2513	BRA	Brazil	2020-01-03	0	0	0	0	0.0	0.0	
2514	BRA	Brazil	2020-01-04	0	0	0	0	0.0	0.0	

5 rows × 32 columns

In [24]:

```
brazil_case.tail()
```

Out[24]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	tot
2651	BRA	Brazil	2020-05-20	271628	17408	17971	1179	1277.892	81.897	
2652	BRA	Brazil	2020-05-21	291579	19951	18859	888	1371.753	93.861	
2653	BRA	Brazil	2020-05-22	310087	18508	20047	1188	1458.825	87.072	
2654	BRA	Brazil	2020-05-23	330890	20803	21048	1001	1556.694	97.869	
2655	BRA	Brazil	2020-05-24	347398	16508	22013	965	1634.357	77.663	

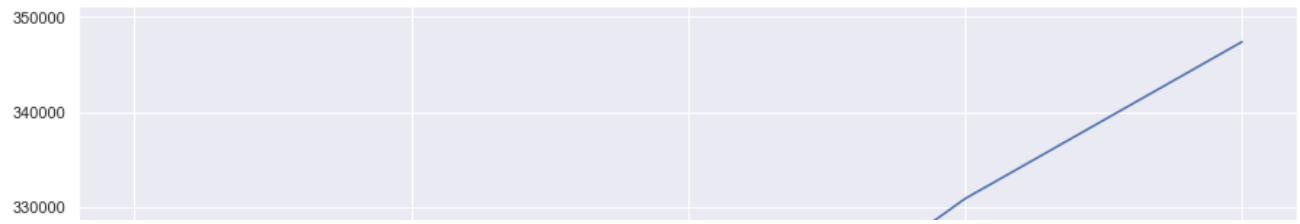
5 rows × 32 columns

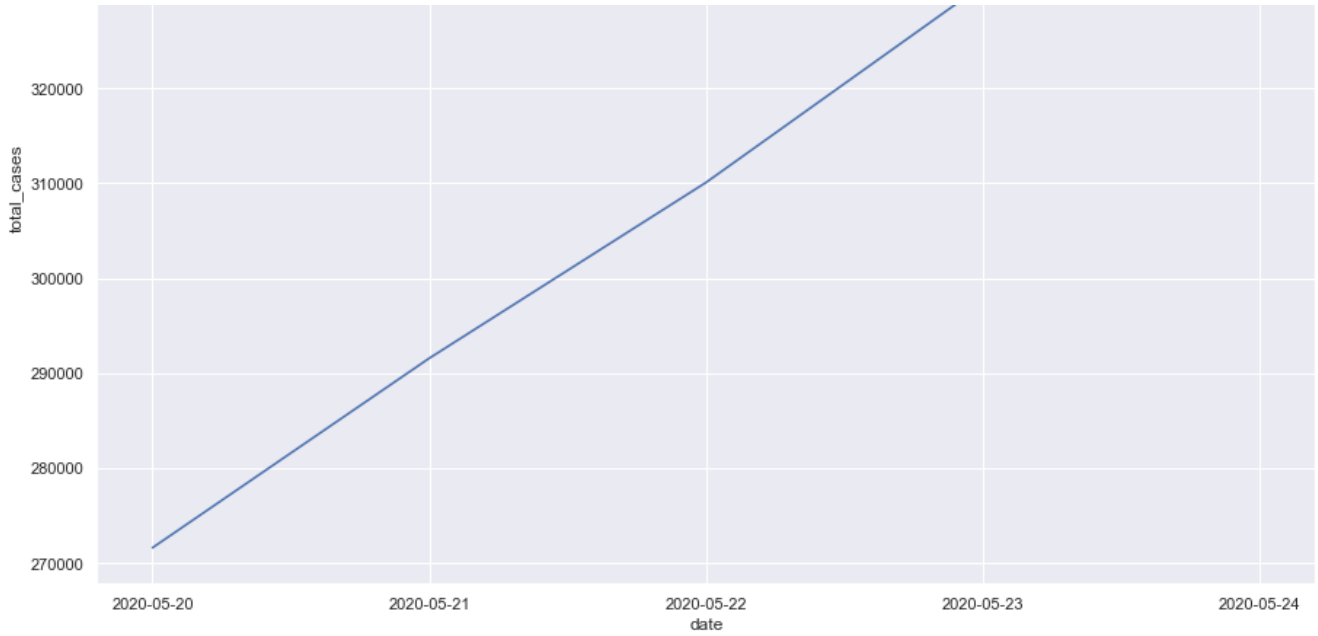
In [25]:

```
#Making a dataframe for brazil for last 5 days
brazil_last_5_days=brazil_case.tail()
```

In [26]:

```
#Total cases in last 5 days
sns.set(rc={'figure.figsize':(15,10)})
sns.lineplot(x="date",y="total_cases",data=brazil_last_5_days)
plt.show()
```



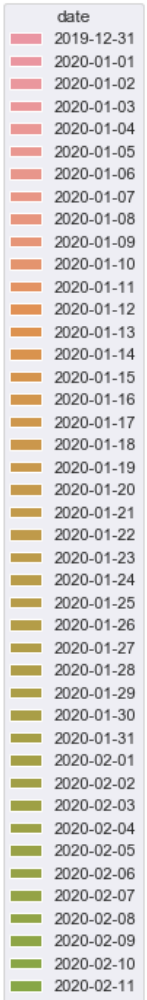


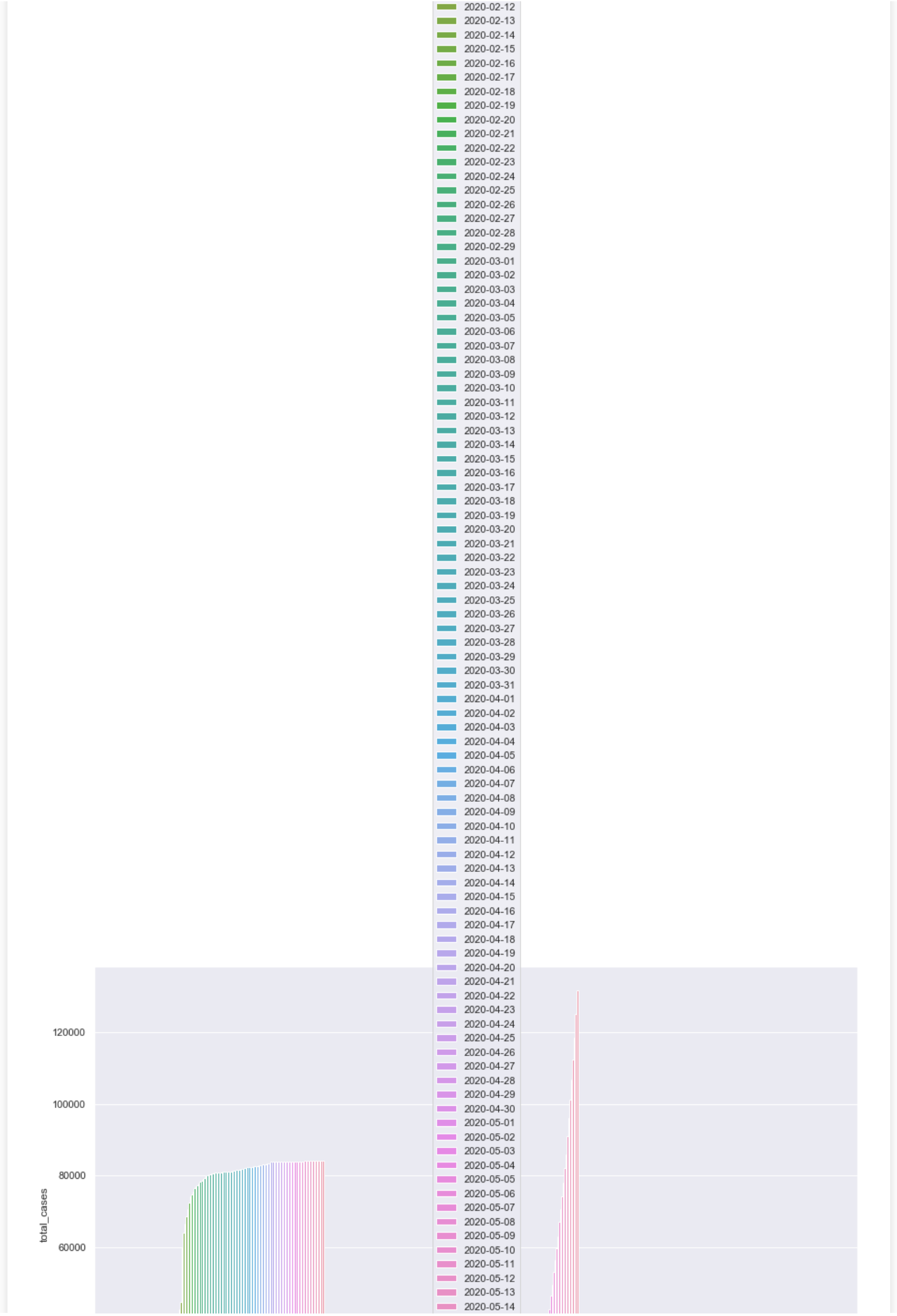
In [27]:

```
#Understanding cases of India, China and Japan
india_japan_china=covid[(covid["location"] =="India") | (covid["location"] =="China") | (covid["loc
ation"]=="Japan")]
```

In [28]:

```
#Plotting growth of cases across China, India and Japan
sns.set(rc={'figure.figsize':(15,10)})
sns.barplot(x="location",y="total_cases",data=india_japan_china,hue="date")
plt.show()
```



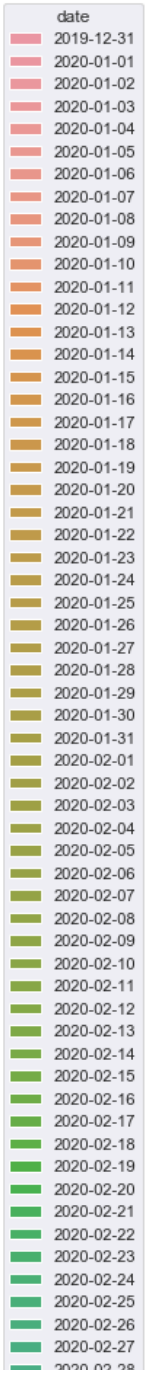

2020-02-12
2020-02-13
2020-02-14
2020-02-15
2020-02-16
2020-02-17
2020-02-18
2020-02-19
2020-02-20
2020-02-21
2020-02-22
2020-02-23
2020-02-24
2020-02-25
2020-02-26
2020-02-27
2020-02-28
2020-02-29
2020-03-01
2020-03-02
2020-03-03
2020-03-04
2020-03-05
2020-03-06
2020-03-07
2020-03-08
2020-03-09
2020-03-10
2020-03-11
2020-03-12
2020-03-13
2020-03-14
2020-03-15
2020-03-16
2020-03-17
2020-03-18
2020-03-19
2020-03-20
2020-03-21
2020-03-22
2020-03-23
2020-03-24
2020-03-25
2020-03-26
2020-03-27
2020-03-28
2020-03-29
2020-03-30
2020-03-31
2020-04-01
2020-04-02
2020-04-03
2020-04-04
2020-04-05
2020-04-06
2020-04-07
2020-04-08
2020-04-09
2020-04-10
2020-04-11
2020-04-12
2020-04-13
2020-04-14
2020-04-15
2020-04-16
2020-04-17
2020-04-18
2020-04-19
2020-04-20
2020-04-21
2020-04-22
2020-04-23
2020-04-24
2020-04-25
2020-04-26
2020-04-27
2020-04-28
2020-04-29
2020-04-30
2020-05-01
2020-05-02
2020-05-03
2020-05-04
2020-05-05
2020-05-06
2020-05-07
2020-05-08
2020-05-09
2020-05-10
2020-05-11
2020-05-12
2020-05-13
2020-05-14
120000
100000
80000
60000
total_cases

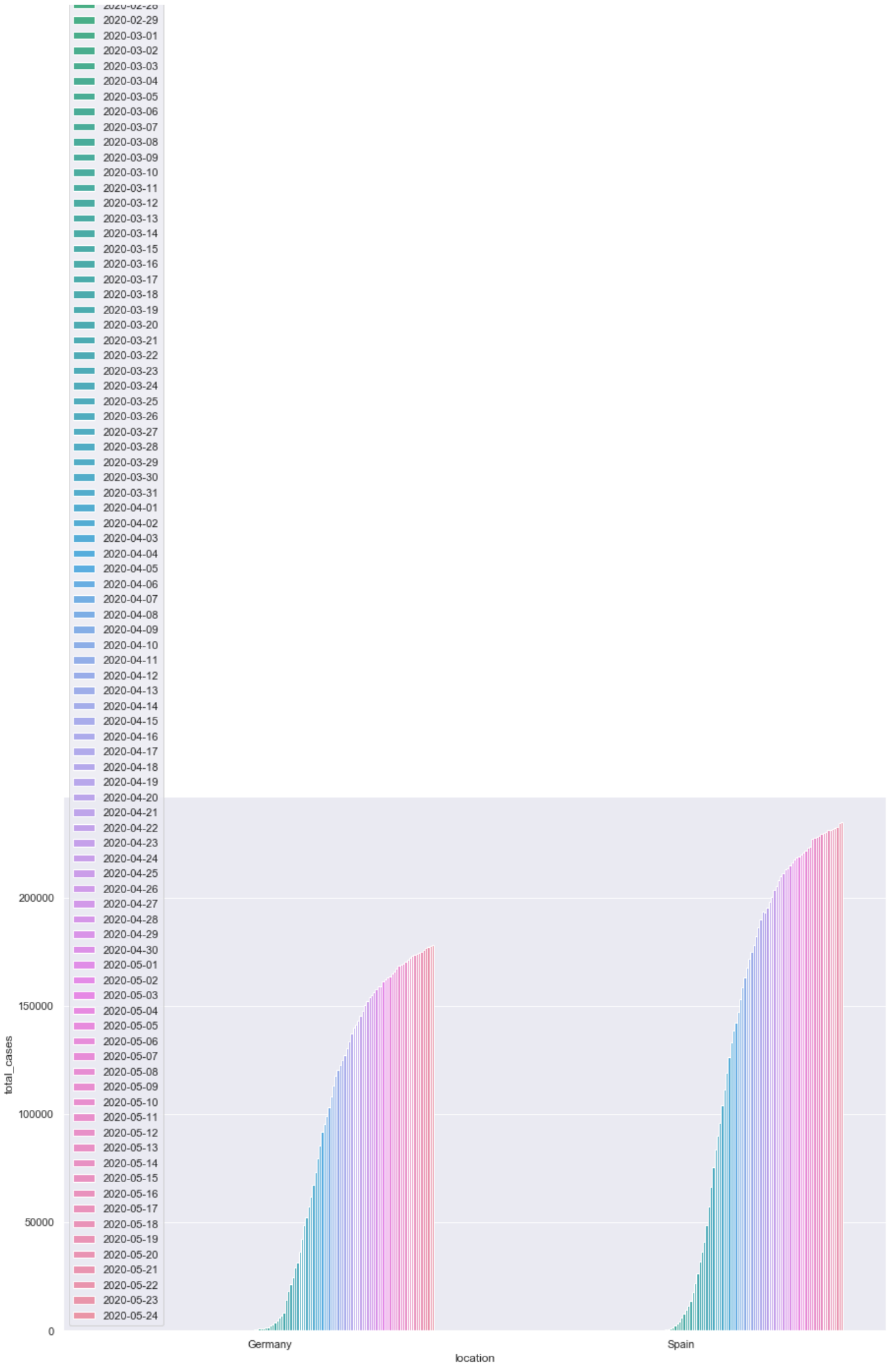
In [29]:

```
#Understanding cases of germany and spain
germany_spain=covid[(covid["location"] =="Germany") | (covid["location"] =="Spain")]
```

In [30]:

```
#Plotting growth of cases across Germany and Spain
sns.set(rc={'figure.figsize':(15,10)})
sns.barplot(x="location",y="total_cases",data=germany_spain,hue="date")
plt.show()
```




2020-02-29
2020-03-01
2020-03-02
2020-03-03
2020-03-04
2020-03-05
2020-03-06
2020-03-07
2020-03-08
2020-03-09
2020-03-10
2020-03-11
2020-03-12
2020-03-13
2020-03-14
2020-03-15
2020-03-16
2020-03-17
2020-03-18
2020-03-19
2020-03-20
2020-03-21
2020-03-22
2020-03-23
2020-03-24
2020-03-25
2020-03-26
2020-03-27
2020-03-28
2020-03-29
2020-03-30
2020-03-31
2020-04-01
2020-04-02
2020-04-03
2020-04-04
2020-04-05
2020-04-06
2020-04-07
2020-04-08
2020-04-09
2020-04-10
2020-04-11
2020-04-12
2020-04-13
2020-04-14
2020-04-15
2020-04-16
2020-04-17
2020-04-18
2020-04-19
2020-04-20
2020-04-21
2020-04-22
2020-04-23
2020-04-24
2020-04-25
2020-04-26
2020-04-27
2020-04-28
2020-04-29
2020-04-30
2020-05-01
2020-05-02
2020-05-03
2020-05-04
2020-05-05
2020-05-06
2020-05-07
2020-05-08
2020-05-09
2020-05-10
2020-05-11
2020-05-12
2020-05-13
2020-05-14
2020-05-15
2020-05-16
2020-05-17
2020-05-18
2020-05-19
2020-05-20
2020-05-21
2020-05-22
2020-05-23
2020-05-24
200000
150000
100000
50000
0
total_cases
Germany
Spain
location

In [31]:

```
#Getting latest data
last_day_cases=covid[covid["date"]=="2020-05-24"]
last_day_cases
```

Out[31]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_milli
62	ABW	Aruba	2020-05-24	101	0	3	0	945.994	0.0
198	AFG	Afghanistan	2020-05-24	9998	782	216	11	256.831	20.0
262	AGO	Angola	2020-05-24	60	0	3	0	1.826	0.0
321	AIA	Anguilla	2020-05-24	3	0	0	0	199.973	0.0
398	ALB	Albania	2020-05-24	989	8	31	0	343.665	2.7
...	...	...	...	...	...	...	...	...	
19045	YEM	Yemen	2020-05-24	212	7	39	6	7.108	0.2
19153	ZAF	South Africa	2020-05-24	21343	1218	407	10	359.863	20.5
19220	ZMB	Zambia	2020-05-24	920	0	7	0	50.044	0.0
19285	ZWE	Zimbabwe	2020-05-24	56	0	4	0	3.768	0.0
19431	OWID_WRL	World	2020-05-24	5273572	97636	341722	3633	676.550	12.5

207 rows × 32 columns

In [32]:

```
#Sorting data w.r.t total_cases
max_cases_country=last_day_cases.sort_values(by="total_cases",ascending=False)
max_cases_country
```

Out[32]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million
19431	OWID_WRL	World	2020-05-24	5273572	97636	341722	3633	676.550	12.526
18391	USA	United States	2020-05-24	1622670	21236	97087	1080	4902.287	64.157
2655	BRA	Brazil	2020-05-24	347398	16508	22013	965	1634.357	77.663
15569	RUS	Russia	2020-05-24	335882	9434	3388	139	2301.595	64.645
9396	ITA	Italy	2020-05-24	229327	669	32735	119	3792.922	11.065
...	...	...	...	...	...	...	...	...	...
18723	VGB	British Virgin Islands	2020-05-24	8	0	1	0	264.577	0.000
1645	BES	Bonaire Sint Eustatius and Saba	2020-05-24	6	0	0	0	228.824	0.000
5543	ESH	Western Sahara	2020-05-24	6	0	0	0	10.045	0.000
321	AIA	Anguilla	2020-05-24	3	0	0	0	199.973	0.000
11086	LSO	Lesotho	2020-05-24	2	1	0	0	0.934	0.467

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million

207 rows × 32 columns

In [33]:

```
#Top 5 countries with maximum cases
max_cases_country[1:6]
```

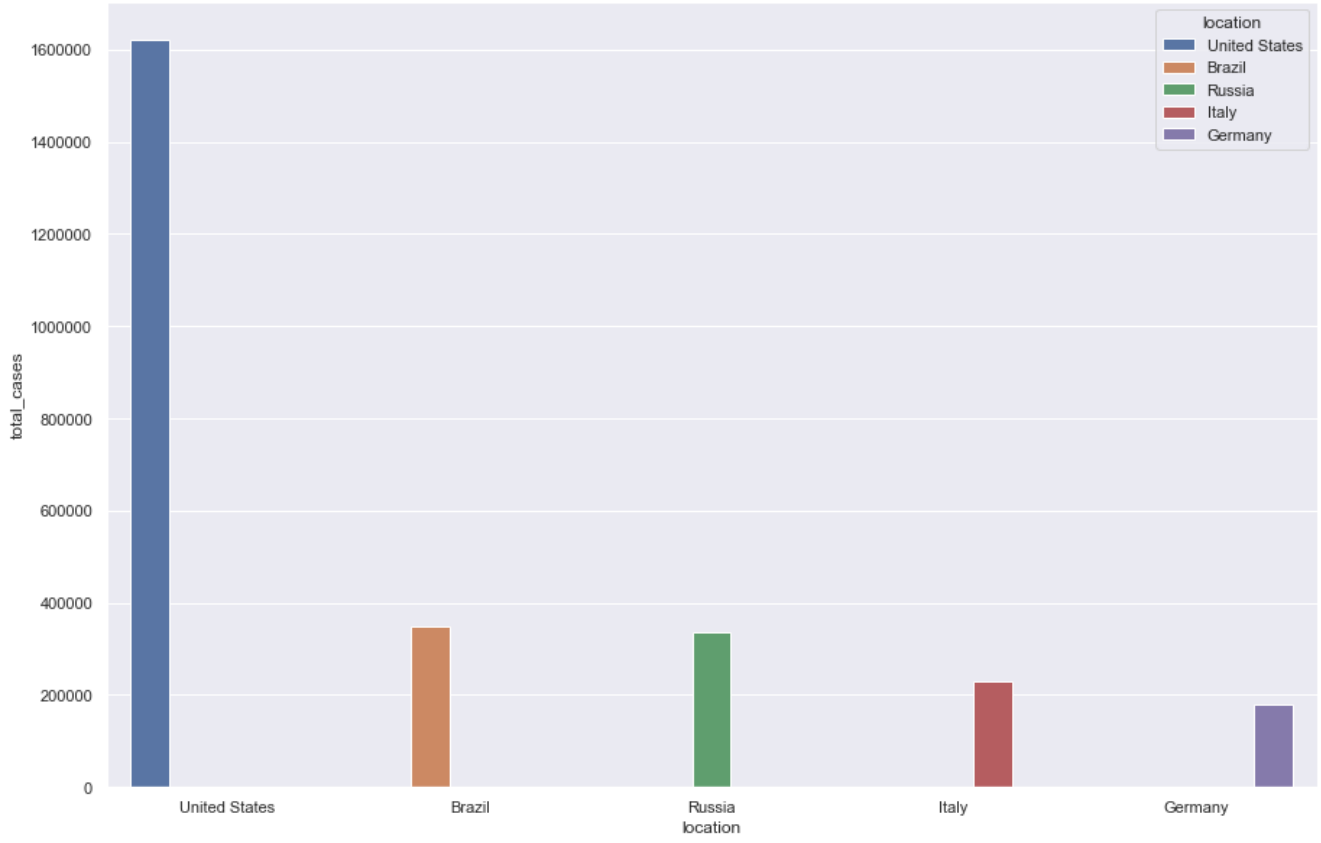
Out[33]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	t
18391	USA	United States	2020-05-24	1622670	21236	97087	1080	4902.287	64.157	
2655	BRA	Brazil	2020-05-24	347398	16508	22013	965	1634.357	77.663	
15569	RUS	Russia	2020-05-24	335882	9434	3388	139	2301.595	64.645	
9396	ITA	Italy	2020-05-24	229327	669	32735	119	3792.922	11.065	
4613	DEU	Germany	2020-05-24	178281	431	8247	31	2127.866	5.144	

5 rows × 32 columns

In [34]:

```
#Making bar-plot for countries with top cases
sns.barplot(x="location",y="total_cases",data=max_cases_country[1:6],hue="location")
plt.show()
```



In [35]:

```
india_case.head()
```

Out[35]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	tot
8379	IND	India	2019-12-31	0	0	0	0	0.0	0.0	
8380	IND	India	2020-01-01	0	0	0	0	0.0	0.0	
8381	IND	India	2020-01-02	0	0	0	0	0.0	0.0	
8382	IND	India	2020-01-03	0	0	0	0	0.0	0.0	
8383	IND	India	2020-01-04	0	0	0	0	0.0	0.0	

5 rows × 32 columns

In [36]:

```
#Linear regression
from sklearn.model_selection import train_test_split
```

In [38]:

```
#converting string date to date-time
import datetime as dt
india_case['date'] = pd.to_datetime(india_case['date'])
india_case.head()
```

```
<ipython-input-38-41c360152603>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  india_case['date'] = pd.to_datetime(india_case['date'])
```

Out[38]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	tot
8379	IND	India	2019-12-31	0	0	0	0	0.0	0.0	
8380	IND	India	2020-01-01	0	0	0	0	0.0	0.0	
8381	IND	India	2020-01-02	0	0	0	0	0.0	0.0	
8382	IND	India	2020-01-03	0	0	0	0	0.0	0.0	
8383	IND	India	2020-01-04	0	0	0	0	0.0	0.0	

5 rows × 32 columns

In [39]:

```
india_case.head()
```

Out[39]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	tot
8379	IND	India	2019-12-31	0	0	0	0	0.0	0.0	
8380	IND	India	2020-01-01	0	0	0	0	0.0	0.0	
8381	IND	India	2020-01-02	0	0	0	0	0.0	0.0	
8382	IND	India	2020-01-03	0	0	0	0	0.0	0.0	

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	tot
8383	IND	India	2020-01-04	0	0	0	0	0.0	0.0	

5 rows × 32 columns

In [40]:

```
#converting date-time to ordinal
india_case['date']=india_case['date'].map(dt.datetime.toordinal)
india_case.head()
```

```
<ipython-input-40-9eb3bcdebcda>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  india_case['date']=india_case['date'].map(dt.datetime.toordinal)
```

Out[40]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	t
8379	IND	India	737424	0	0	0	0	0.0	0.0	
8380	IND	India	737425	0	0	0	0	0.0	0.0	
8381	IND	India	737426	0	0	0	0	0.0	0.0	
8382	IND	India	737427	0	0	0	0	0.0	0.0	
8383	IND	India	737428	0	0	0	0	0.0	0.0	

5 rows × 32 columns

In [41]:

```
#getting dependent variable and inpedent variable
x=india_case['date']
y=india_case['total_cases']
```

In [42]:

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

In [43]:

```
from sklearn.linear_model import LinearRegression
```

In [44]:

```
lr = LinearRegression()
```

In [45]:

```
import numpy as np
lr.fit(np.array(x_train).reshape(-1,1),np.array(y_train).reshape(-1,1))
```

Out[45]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [46]:

```
india_case.tail()
```

Out[46]:

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	t

	iso_code	location	date	total_cases	new_cases	total_deaths	new_deaths	total_cases_per_million	new_cases_per_million	t
8519	IND	India	737565	106750	5611	3303	140	77.355	4.066	
8520	IND	India	737566	112359	5609	3435	132	81.419	4.064	
8521	IND	India	737567	118447	6088	3583	148	85.831	4.412	
8522	IND	India	737568	125101	6654	3720	137	90.653	4.822	
8523	IND	India	737569	131868	6767	3867	147	95.556	4.904	

5 rows × 32 columns

In [47]:

```
y_pred=lr.predict(np.array(x_test).reshape(-1,1))
```

In [48]:

```
from sklearn.metrics import mean_squared_error
```

In [49]:

```
mean_squared_error(x_test,y_pred)
```

Out[49]:

```
524620551377.64185
```

In [51]:

```
lr.predict(np.array([[737573]]))
```

Out[51]:

```
array([[49967.36422545]])
```

In [ ]:

```

```