RASHI JAIN , PGID: 11920010

Q.6. The following functions are all intended to check whether a string contains any uppercase letters, but at least some of them are wrong. For each function, describe what the function actually does, assuming that the parameter is a string.

a. Problem with this function: It is just checking the first character of the string then returning true/false.

b. Problem with this function: It will return false every time. As the function is checking whether small character 'c' is in uppercase or not. This condition will never be true. So, function will always be false.

c. Problem with this function: Function is checking whether the last character in the string is uppercase or not. If not, then assigning false to flag.

d. Problem with this function: It will return true if any character in the string is in uppercase.

e. This function is correct. It is checking whether all the characters in the string is in uppercase or not. If they are in uppercase then returns true.

Q.11. Analysis on -

Univariate analysis to count the number of cities in each state

Most populated city in the State

Sex ratio according to state

Female and Male literacy rate in state

Literacy Rate Affecting Sex Ratio

Female and Male graduation rate in state

In [280]:
```
import numpy as np
import pandas as pd
import seaborn as sns

import matplotlib.pyplot as plt
```

In [281]:
```
file = pd.read_csv("Indian_cities.csv")
file.head()
```

Out[281]:

	name_of_city	state_code	state_name	dist_code	population_total	population_male	populat
0	Abohar	3	PUNJAB	9	145238	76840	
1	Achalpur	27	MAHARASHTRA	7	112293	58256	
2	Adilabad	28	ANDHRA PRADESH	1	117388	59232	
3	Adityapur	20	JHARKHAND	24	173988	91495	
4	Adoni	28	ANDHRA PRADESH	21	166537	82743	

5 rows × 22 columns

In [282]:
```
print(file.info())
print(file.columns)
```

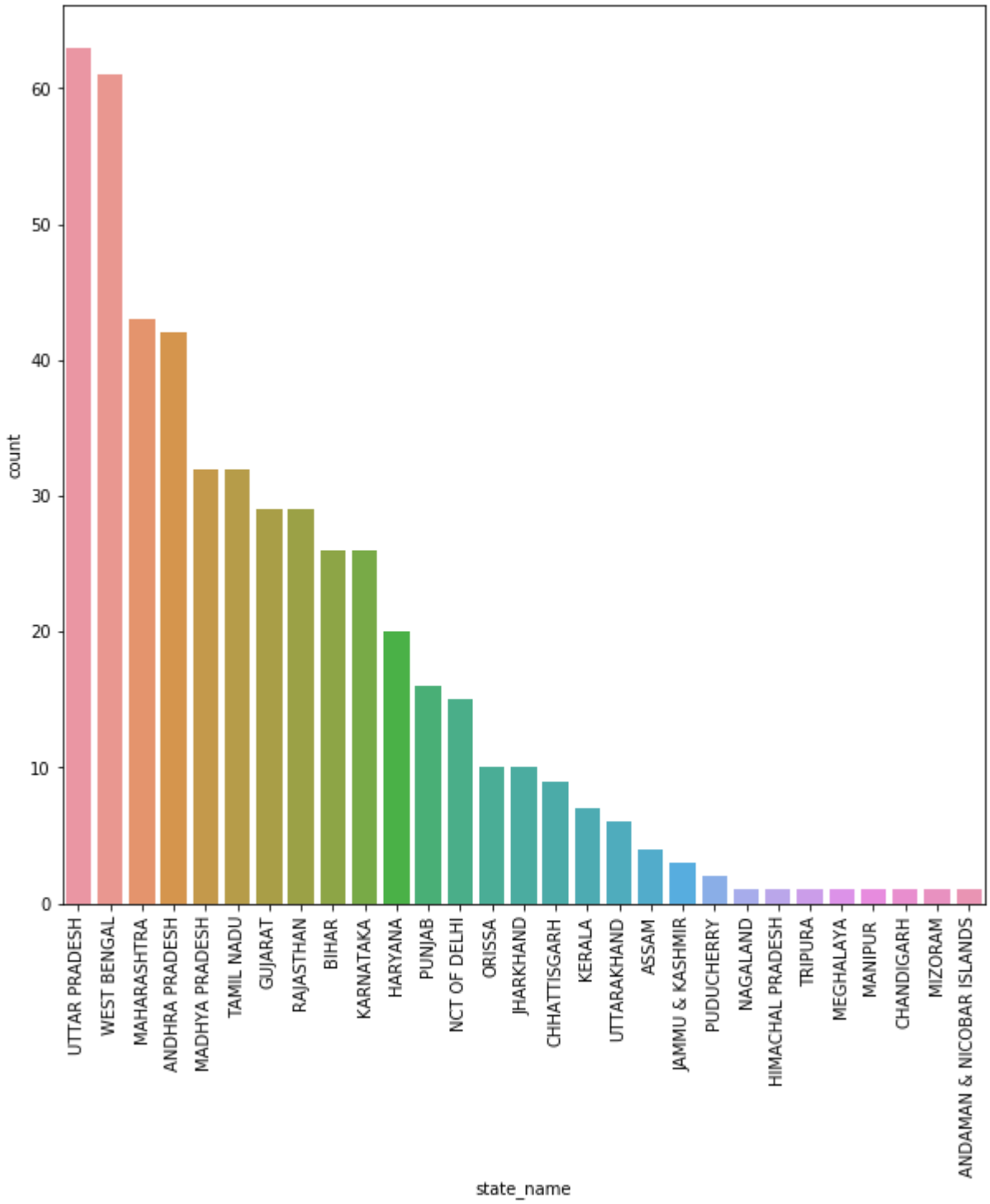
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 493 entries, 0 to 492
Data columns (total 22 columns):
name_of_city                        493 non-null object
state_code                          493 non-null int64
state_name                          493 non-null object
dist_code                           493 non-null int64
population_total                    493 non-null int64
population_male                     493 non-null int64
population_female                   493 non-null int64
0-6_population_total                493 non-null int64
0-6_population_male                 493 non-null int64
0-6_population_female               493 non-null int64
literates_total                     493 non-null int64
literates_male                      493 non-null int64
literates_female                    493 non-null int64
sex_ratio                           493 non-null int64
child_sex_ratio                     493 non-null int64
effective_literacy_rate_total       493 non-null float64
effective_literacy_rate_male        493 non-null float64
effective_literacy_rate_female      493 non-null float64
location                            493 non-null object
total_graduates                     493 non-null int64
male_graduates                      493 non-null int64
female_graduates                    493 non-null int64
dtypes: float64(3), int64(16), object(3)
memory usage: 84.8+ KB
None
Index(['name_of_city', 'state_code', 'state_name', 'dist_code',
       'population_total', 'population_male', 'population_female',
       '0-6_population_total', '0-6_population_male', '0-6_population_female',
       'literates_total', 'literates_male', 'literates_female', 'sex_ratio',
       'child_sex_ratio', 'effective_literacy_rate_total',
       'effective_literacy_rate_male', 'effective_literacy_rate_female',
       'location', 'total_graduates', 'male_graduates', 'female_graduates'],
      dtype='object')
```

Univariate Analysis

In [283]:
```
#Univariate Analysis is only meaningful in State
#We can only count the number of cities in each state

plt.figure(figsize=(10,10))

sns.countplot(x="state_name", data=file, order = file["state_name"].value_counts
plt.xticks(rotation=90)
plt.show()
```



From the above graph, It is clear that uttar Pradesh has the highest number of cities. After UttarPradesh, West Bengal is the second state having most cities.

Segmented Univariate Analysis

Most populated city in the State

In [284]:
```
states=file['state_name'].unique()
ans = pd.DataFrame({'state_name' : [],'name_of_city':[],'population_total':[]})
for i in states:
    temp = file[file['state_name']==i]
    max_pop_index = temp.index[temp['population_total']==temp['population_total'
    row = temp.loc[max_pop_index,['state_name','name_of_city','population_total'
    ans = ans.append(row.iloc[0], ignore_index=True)
print(ans)
```

	state_name	name_of_city	population_total
0	PUNJAB	Ludhiana	1613878.0
1	MAHARASHTRA	Greater Mumbai	12478447.0
2	ANDHRA PRADESH	Greater Hyderabad	6809970.0
3	JHARKHAND	Dhanbad	1161561.0
4	TRIPURA	Agartala	399688.0
5	UTTAR PRADESH	Lucknow	2815601.0
6	GUJARAT	Ahmadabad	5570585.0
7	MIZORAM	Aizawl	291822.0
8	RAJASTHAN	Jaipur	3073350.0
9	TAMIL NADU	Chennai	4681087.0
10	KERALA	Thiruvananthapuram	752490.0
11	HARYANA	Faridabad	1404653.0
12	CHHATTISGARH	Raipur	1010087.0
13	JAMMU & KASHMIR	Srinagar	1192792.0
14	BIHAR	Patna	1683200.0
15	WEST BENGAL	Kolkata	4486679.0
16	KARNATAKA	Bengaluru	8425970.0
17	ORISSA	Bhubaneswar Town	837737.0
18	MADHYA PRADESH	Indore	1960631.0
19	NCT OF DELHI	Delhi	11007835.0
20	CHANDIGARH	Chandigarh	960787.0
21	UTTARAKHAND	Dehradun	578420.0
22	ASSAM	Guwahati	963429.0
23	NAGALAND	Dimapur	123777.0
24	MANIPUR	Imphal	264986.0
25	PUDUCHERRY	Ozhukarai	300028.0
26	ANDAMAN & NICOBAR ISLANDS	Port Blair	100608.0
27	MEGHALAYA	Shillong	143007.0
28	HIMACHAL PRADESH	Shimla	169758.0

From data of most populated city in the state, we can figure out that, most of these cities are metropolitian or have some industry.

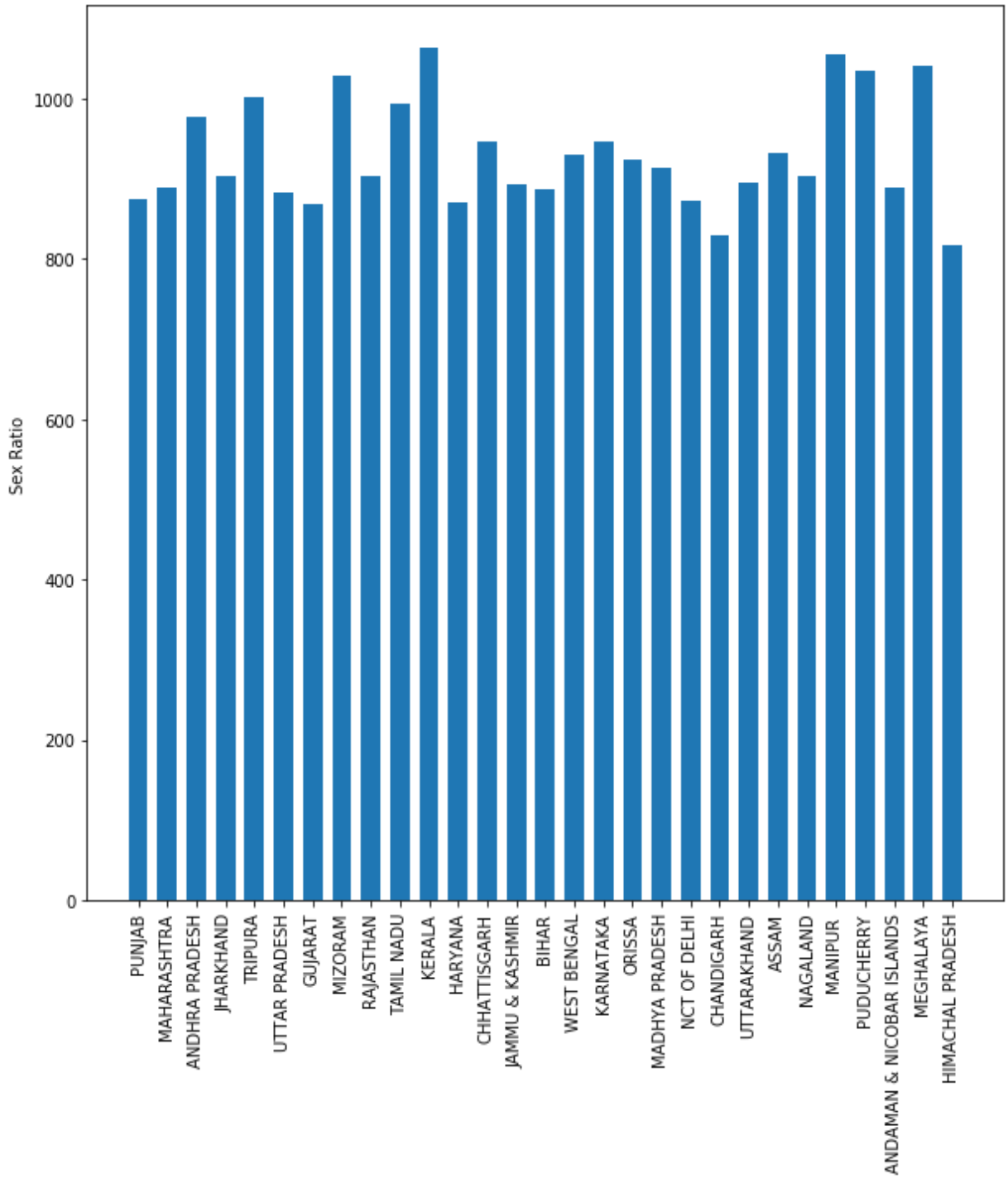
Due to Urbanisation and Industrialisation, people have migrated to cities over the period of time and this number is increasing with time.

Due to higher population, these cities are facing water scarcity and higher pollution index in comparison to other cities of state. Also, cost of living is high in these cities

Sex ratio according to state

In [285]:
```
sexratio = []
for i in states:
    temp = file[file['state_name']==i]
    male = temp['population_male'].sum()
    ratio = (temp['population_female'].sum())/male
    sexratio.append(ratio*1000)

plt.figure(figsize=(10,10))
width = 1/1.5
plt.bar(states, sexratio, width)
plt.xticks(rotation=90)
plt.ylabel('Sex Ratio')
plt.show()
```



If we look at the graph, Kerela has the highest female to male ratio. Number of females are more than number of males. This denotes crimes like foeticide and infanticide are less.

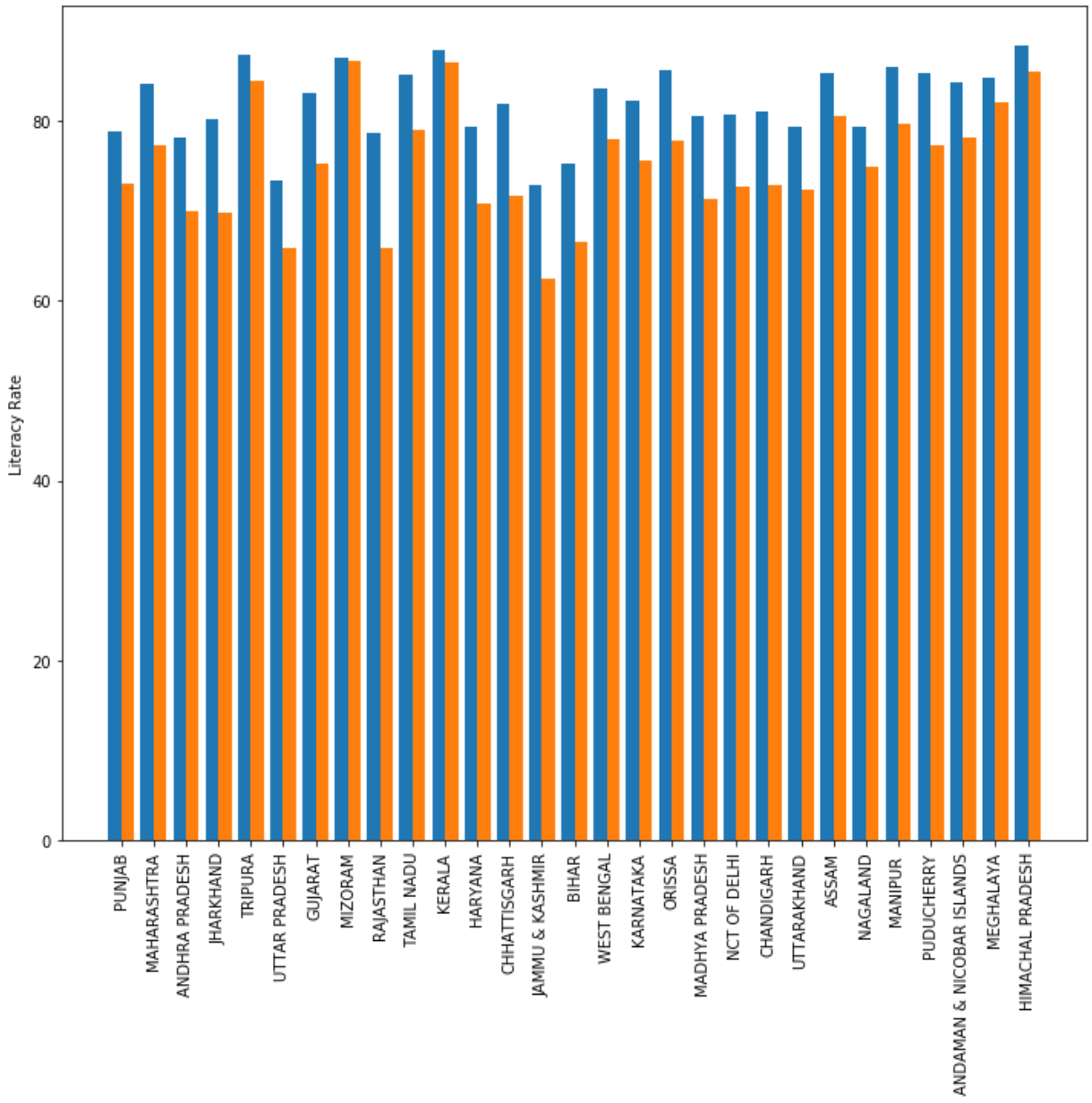
At the same time, if you look at the sexratio of Uttarpradesh, MadhyaPradesh, Haryana and Rajasthan, Ratio is approx 850 per thousand males. This denotes crimes like foeticide and infanticide are very frequent in these states.

FEMALE AND MALE LITERACY RATE IN STATE

In [286]:
```
maleRatio = []
femaleRatio = []

for i in states:
    temp = file[file['state_name']==i]
    male = temp['literates_male'].sum()/temp['population_male'].sum()
    female = temp['literates_female'].sum()/temp['population_female'].sum()
    maleRatio.append(male*100)
    femaleRatio.append(female*100)

plt.figure(figsize=(12,10))
states_ = np.arange(len(states))
plt.bar(states_ - 0.2, maleRatio, 0.4)
plt.bar(states_ + 0.2, femaleRatio, 0.4)
plt.xticks(states_, states,rotation=90) # set labels manually
plt.ylabel('Literacy Rate')
plt.show()
```



Kerela, Himachal Pradesh and Mizoram have the highest number of literate people. Both men and women are almost equally literate in these states. This denotes, people don't discriminate based on gender for giving education to their childern.

At the same time, If we compare the literacy rate of men and women in Rajasthan as well in J&K, women literacy rate is quite low. People discriminate on the basis of gender and believe in giving education to males

Literacy Rate Affecting Sex Ratio

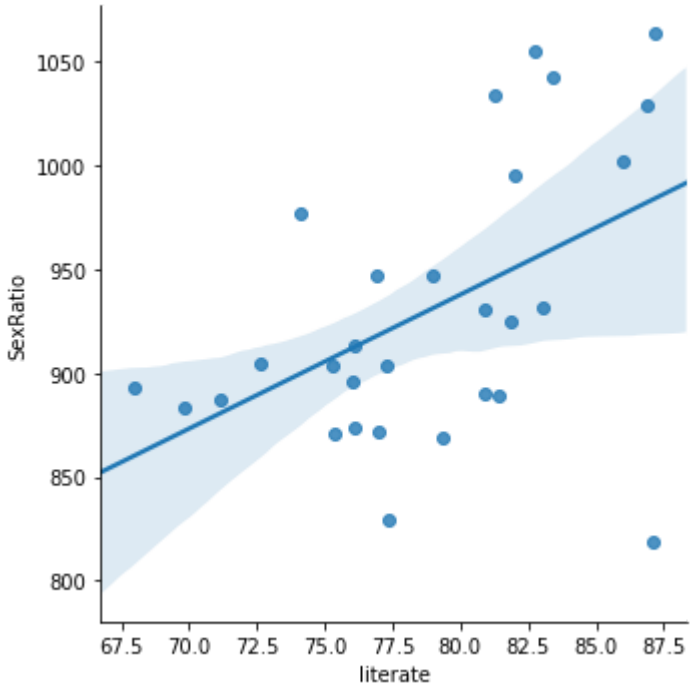
In [287]:
```
Ratio = []

for i in states:
    temp = file[file['state_name']==i]
    liter = temp['literates_total'].sum()/temp['population_total'].sum()
    Ratio.append(liter*100)

df = pd.DataFrame({'STATE': states, 'literate': Ratio, 'SexRatio': sexratio})
print(df)

print(df['literate'].corr(df['SexRatio']))
sns.lmplot(x="literate", y="SexRatio", data=df)
plt.show()
```

```
                        STATE   literate     SexRatio
0                      PUNJAB  76.110828   874.111251
1                 MAHARASHTRA  80.892233   889.722217
2              ANDHRA PRADESH  74.092308   976.815441
3                   JHARKHAND  75.264892   903.293370
4                     TRIPURA  85.975311  1002.284386
5                UTTAR PRADESH  69.831484   883.587431
6                     GUJARAT  79.364671   869.171262
7                     MIZORAM  86.898178  1029.317886
8                   RAJASTHAN  72.589461   904.106687
9                  TAMIL NADU  82.033115   994.951030
10                     KERALA  87.140077  1063.448673
11                    HARYANA  75.344556   870.610578
12                CHHATTISGARH  76.913259   946.576312
13             JAMMU & KASHMIR  67.934838   893.425100
14                       BIHAR  71.165070   886.777332
15                 WEST BENGAL  80.861808   930.320524
16                   KARNATAKA  78.956051   947.517269
17                      ORISSA  81.871143   924.628130
18               MADHYA PRADESH  76.125889   913.351396
19                 NCT OF DELHI  76.984337   872.155835
20                   CHANDIGARH  77.358874   829.283013
21                  UTTARAKHAND  76.050528   896.057510
22                        ASSAM  83.015324   931.804169
23                     NAGALAND  77.272837   903.763631
24                      MANIPUR  82.744371  1055.254361
25                   PUDUCHERRY  81.233700  1034.100594
26  ANDAMAN & NICOBAR ISLANDS  81.413009   889.458561
27                    MEGHALAYA  83.431580  1042.140287
28            HIMACHAL PRADESH  87.064527   818.238293
0.48732152064363043
```

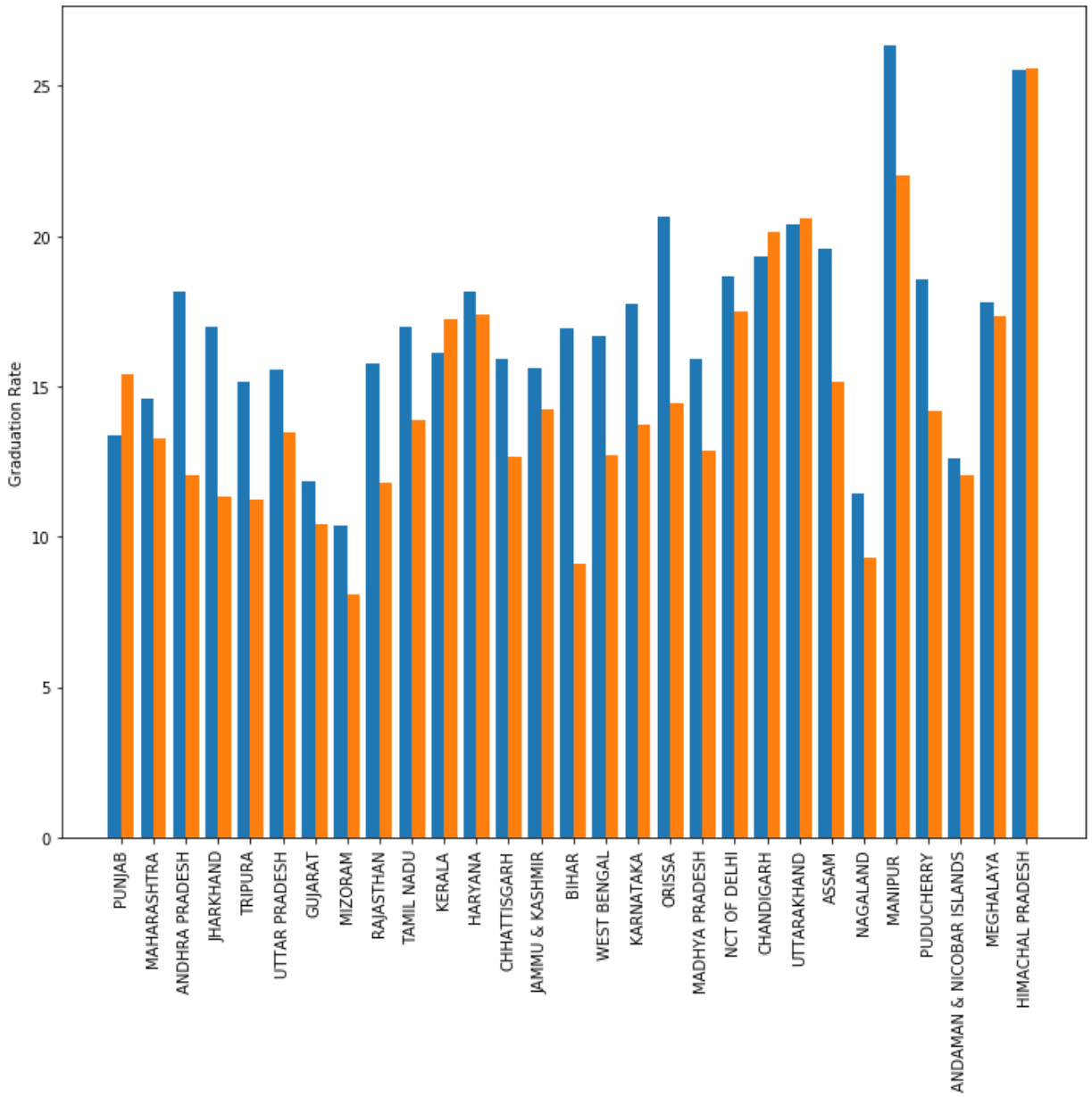


There is some correlation between sex ratio and literacy rate of state. If people get more literate then sex ratio can slightly improve

FEMALE AND MALE GRADUATION RATE IN STATE

In [288]:
```
m_grad_ratio = []
f_grad_ratio = []
for i in states:
    temp = file[file['state_name']==i]
    male = temp['male_graduates'].sum()/temp['population_male'].sum()
    female = temp['female_graduates'].sum()/temp['population_female'].sum()
    m_grad_ratio.append(male*100)
    f_grad_ratio.append(female*100)

plt.figure(figsize=(12,10))
states_ = np.arange(len(states))
plt.bar(states_ - 0.2, m_grad_ratio, 0.4)
plt.bar(states_ + 0.2, f_grad_ratio, 0.4)
plt.xticks(states_, states,rotation=90) # set labels manually
plt.ylabel('Graduation Rate')
plt.show()
```



Although, kerela has higest number of literate people but they don't do higher education. Himachal Pradesh has a perfect balance in terms of graduation. Number of males graduate is equal to female graduates. Manipur has higher male graduates than female.

Q.10 Earthquake Analysis

Couting number of Earthquake in the year and from which year, it is increasing around the world

Magnitude of Earthquake and analysis about damage it cause

Reason behind earthquake with more than 6 magnitude(on average)

In [289]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import datetime as dt
import matplotlib.pyplot as plt
```

In [290]:
```
file2 = pd.read_csv("earthquake.csv")
file2.head()
```

Out[290]:

	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations	Magnitude	Magni
0	01/02/1965	13:44:18	19.246	145.616	Earthquake	131.6	NaN	NaN	6.0	
1	01/04/1965	11:29:49	1.863	127.352	Earthquake	80.0	NaN	NaN	5.8	
2	01/05/1965	18:05:58	-20.579	-173.972	Earthquake	20.0	NaN	NaN	6.2	
3	01/08/1965	18:49:43	-59.076	-23.557	Earthquake	15.0	NaN	NaN	5.8	
4	01/09/1965	13:32:50	11.938	126.427	Earthquake	15.0	NaN	NaN	5.8	

5 rows × 21 columns

In [291]:
```
print(file2.info())

print(file2.columns)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23412 entries, 0 to 23411
Data columns (total 21 columns):
Date                            23412 non-null object
Time                            23412 non-null object
Latitude                        23412 non-null float64
Longitude                       23412 non-null float64
Type                            23412 non-null object
Depth                           23412 non-null float64
Depth Error                     4461 non-null float64
Depth Seismic Stations          7097 non-null float64
Magnitude                       23412 non-null float64
Magnitude Type                  23409 non-null object
Magnitude Error                 327 non-null float64
Magnitude Seismic Stations      2564 non-null float64
Azimuthal Gap                   7299 non-null float64
Horizontal Distance             1604 non-null float64
Horizontal Error                1156 non-null float64
Root Mean Square                17352 non-null float64
ID                              23412 non-null object
Source                          23412 non-null object
Location Source                 23412 non-null object
Magnitude Source                23412 non-null object
Status                          23412 non-null object
dtypes: float64(12), object(9)
memory usage: 3.8+ MB
None
Index(['Date', 'Time', 'Latitude', 'Longitude', 'Type', 'Depth', 'Depth Error',
       'Depth Seismic Stations', 'Magnitude', 'Magnitude Type',
       'Magnitude Error', 'Magnitude Seismic Stations', 'Azimuthal Gap',
       'Horizontal Distance', 'Horizontal Error', 'Root Mean Square', 'ID',
       'Source', 'Location Source', 'Magnitude Source', 'Status'],
      dtype='object')
```

In [292]:
```
#Finding the percentage null value
perc_null=(file2.isnull().sum()/len(file2.index))*100
perc_null
```

Out[292]:
```
Date                          0.000000
Time                          0.000000
Latitude                      0.000000
Longitude                     0.000000
Type                          0.000000
Depth                         0.000000
Depth Error                  80.945669
Depth Seismic Stations       69.686486
Magnitude                     0.000000
Magnitude Type                0.012814
Magnitude Error              98.603280
Magnitude Seismic Stations   89.048351
Azimuthal Gap                68.823680
Horizontal Distance          93.148813
Horizontal Error             95.062361
Root Mean Square             25.884162
ID                            0.000000
Source                        0.000000
Location Source               0.000000
Magnitude Source              0.000000
Status                        0.000000
dtype: float64
```

In [293]:
```
 NULL value
rror', 'Depth Seismic Stations','Magnitude Error', 'Magnitude Seismic Stations','
```

In [294]:
```
file2.head()
```

Out[294]:

	Date	Time	Latitude	Longitude	Type	Depth	Magnitude	Magnitude Type	
0	01/02/1965	13:44:18	19.246	145.616	Earthquake	131.6	6.0	MW	ISCGEM860
1	01/04/1965	11:29:49	1.863	127.352	Earthquake	80.0	5.8	MW	ISCGEM860
2	01/05/1965	18:05:58	-20.579	-173.972	Earthquake	20.0	6.2	MW	ISCGEM860
3	01/08/1965	18:49:43	-59.076	-23.557	Earthquake	15.0	5.8	MW	ISCGEM860
4	01/09/1965	13:32:50	11.938	126.427	Earthquake	15.0	5.8	MW	ISCGEM860

In [295]:

```
# convert the 'Date' column to datetime format
file2['Date']= pd.to_datetime(file2['Date'],utc =True)
file2['year']=file2['Date'].dt.year
file2.head()
```

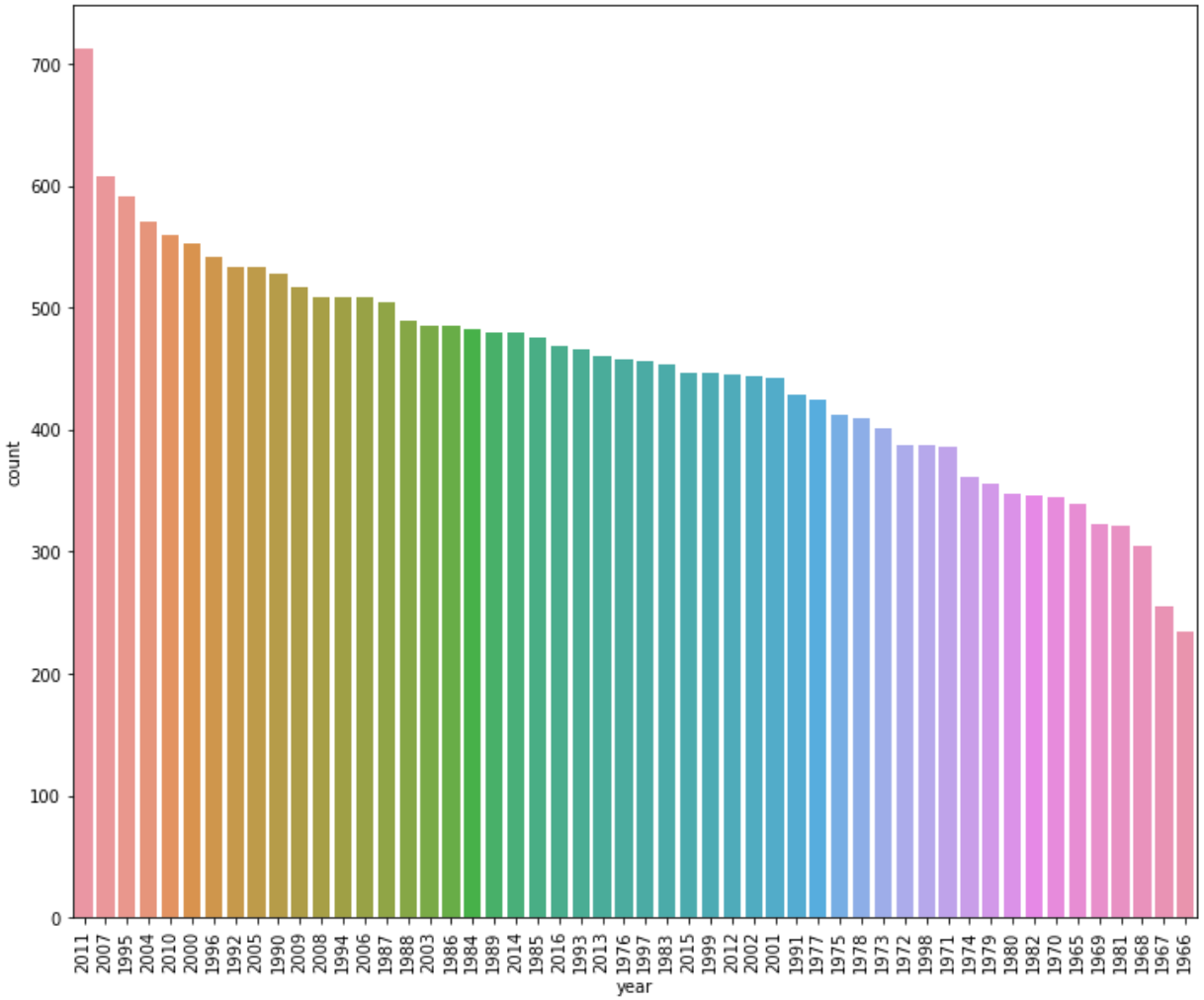
Out[295]:

	Date	Time	Latitude	Longitude	Type	Depth	Magnitude	Magnitude Type	
0	1965-01-02 00:00:00+00:00	13:44:18	19.246	145.616	Earthquake	131.6	6.0	MW	ISCGEM
1	1965-01-04 00:00:00+00:00	11:29:49	1.863	127.352	Earthquake	80.0	5.8	MW	ISCGEM
2	1965-01-05 00:00:00+00:00	18:05:58	-20.579	-173.972	Earthquake	20.0	6.2	MW	ISCGEM
3	1965-01-08 00:00:00+00:00	18:49:43	-59.076	-23.557	Earthquake	15.0	5.8	MW	ISCGEM
4	1965-01-09 00:00:00+00:00	13:32:50	11.938	126.427	Earthquake	15.0	5.8	MW	ISCGEM

Couting number of Earthquake in the year

In [296]:
```
plt.figure(figsize=(12,10))

sns.countplot(x="year", data=file2, order = file2["year"].value_counts().index)
plt.xticks(rotation=90)
plt.show()
```



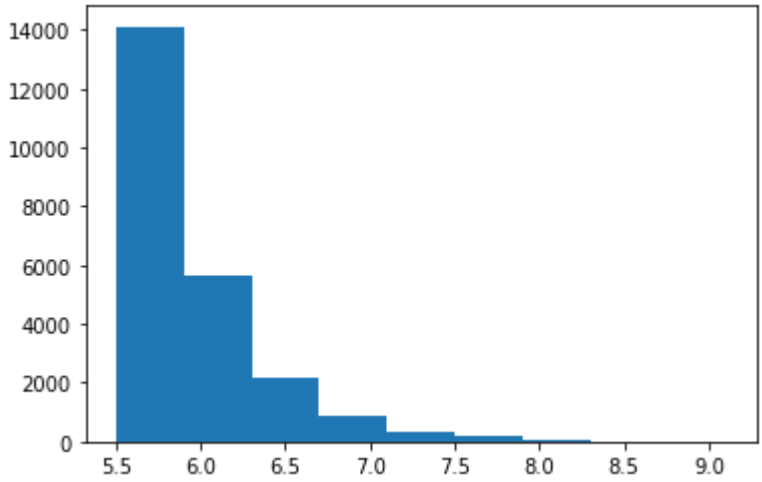
By looking at this graph, we can figure out-

In 2011, There were maximum number of earthquakes.

From 1995, number of earthquakes in the given data, is approximately more than 450

Magnitude of Earthquake

In [297]:
```
x = file2["Magnitude"]
plt.hist(x, bins = 9)
plt.show()
```



Most of the earthquake is of 5.5 to 6 magnitude.

5.5 to 6.0 - Cause only Slight damage to buildings and other structures

6.1 to 6.9 - May cause a lot of damage in very populated areas.

7.0 to 7.9 - Major earthquake. Serious damage.

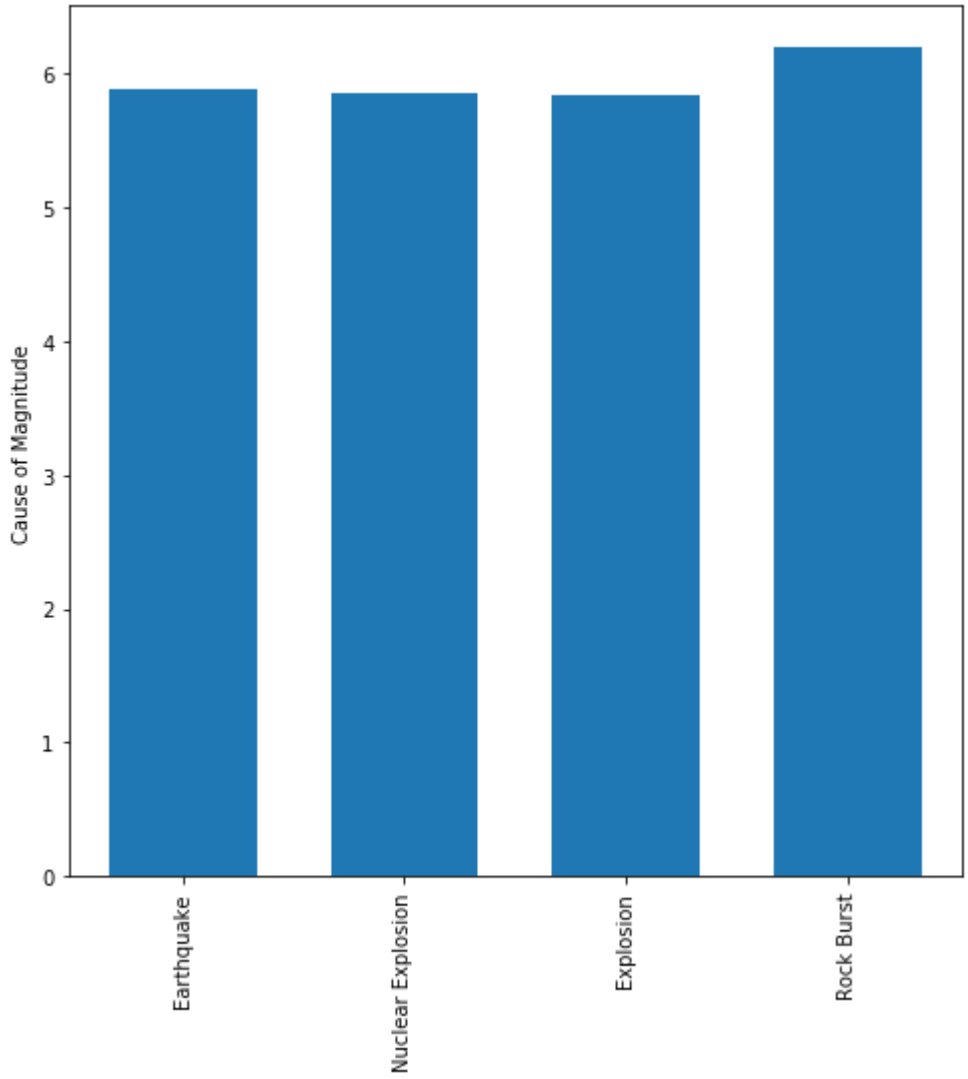
8.0 or greater - Great earthquake. It can totally destroy communities near the epicenter

Reason behind earthquake with more than 6 magnitude(on average)

In [298]:
```
EarquakeType = file2['Type'].unique()
print(EarquakeType)
Magnitude = []
for i in EarquakeType:
    temp = file2[file2['Type']==i]
    avgr  = temp['Magnitude'].mean()
    Magnitude.append(avgr)

plt.figure(figsize=(8,8))
width = 1/1.5
plt.bar(EarquakeType, Magnitude, width)
plt.xticks(rotation=90)
plt.ylabel('Cause of Magnitude')
plt.show()
```

```
['Earthquake' 'Nuclear Explosion' 'Explosion' 'Rock Burst']
```



According to data, RockBurst is the major reason behind Earthquake of high magnitude

Furthermore, we can plot earthquake prone zones on the worldmap