In [1]:

```
import pandas as pd
# Load the CSV file into a DataFrame
df = pd.read_csv(
"https://raw.githubusercontent.com/xscientisttech/detaset/main/india-state-wise-data-anal
)
```

In [2]:

```
display(df)
```

	State & District	Population	Male	Female	Literate	Male_Literate	Female_l
0	district_code:1, state_name:JAMMU AND KASHMIR,...	870354	474190	396164	439654	282823	
1	district_code:2, state_name:JAMMU AND KASHMIR,...	753745	398041	355704	335649	207741	
2	district_code:3, state_name:JAMMU AND KASHMIR,...	133487	78971	54516	93770	62834	
3	district_code:4, state_name:JAMMU AND KASHMIR,...	140802	77785	63017	86236	56301	
4	district_code:5, state_name:JAMMU AND KASHMIR,...	476835	251899	224936	261724	163333	
...	...	...	...	...	...	...	
635	district_code:636, state_name:PONDICHERRY, dis...	41816	19143	22673	36470	16610	
636	district_code:637, state_name:PONDICHERRY, dis...	200222	97809	102413	154916	79903	
637	district_code:638, state_name:ANDAMAN AND NICO...	36842	20727	16115	25332	15397	
638	district_code:639, state_name:ANDAMAN AND NICO...	105597	54861	50736	78683	43186	
639	district_code:640, state_name:ANDAMAN AND NICO...	238142	127283	110859	190266	105794	

640 rows × 51 columns

#This are questions need to be ask How many rows does our dataframe have?

How many columns does it have?

What are the labels for the columns? Do the columns have names?

Are there any missing values in our dataframe? Does our dataframe contain any bad data?

In [3]:

```
df.head()
```

Out[3]:

	State & District	Population	Male	Female	Literate	Male_Literate	Female_Literate
0	district_code:1, state_name:JAMMU AND KASHMIR,...	870354	474190	396164	439654	282823	156831
1	district_code:2, state_name:JAMMU AND KASHMIR,...	753745	398041	355704	335649	207741	127908
2	district_code:3, state_name:JAMMU AND KASHMIR,...	133487	78971	54516	93770	62834	30936
3	district_code:4, state_name:JAMMU AND KASHMIR,...	140802	77785	63017	86236	56301	29935
4	district_code:5, state_name:JAMMU AND KASHMIR,...	476835	251899	224936	261724	163333	98391

5 rows × 51 columns

In [4]:

```
df.head()# by default return first 5 rows
```

Out[4]:

	State & District	Population	Male	Female	Literate	Male_Literate	Female_Literate
0	district_code:1, state_name:JAMMU AND KASHMIR,...	870354	474190	396164	439654	282823	156831
1	district_code:2, state_name:JAMMU AND KASHMIR,...	753745	398041	355704	335649	207741	127908
2	district_code:3, state_name:JAMMU AND KASHMIR,...	133487	78971	54516	93770	62834	30936
3	district_code:4, state_name:JAMMU AND KASHMIR,...	140802	77785	63017	86236	56301	29935
4	district_code:5, state_name:JAMMU AND KASHMIR,...	476835	251899	224936	261724	163333	98391

5 rows × 51 columns

In [5]:

```
df.tail() #by default return last five last rows
```

Out[5]:

	State & District	Population	Male	Female	Literate	Male_Literate	Female_l
635	district_code:636, state_name:PONDICHERRY, dis...	41816	19143	22673	36470	16610	
636	district_code:637, state_name:PONDICHERRY, dis...	200222	97809	102413	154916	79903	
637	district_code:638, state_name:ANDAMAN AND NICO...	36842	20727	16115	25332	15397	
638	district_code:639, state_name:ANDAMAN AND NICO...	105597	54861	50736	78683	43186	
639	district_code:640, state_name:ANDAMAN AND NICO...	238142	127283	110859	190266	105794	

5 rows × 51 columns

In [6]:

```
df.tail(3) # last 3 rows
```

Out[6]:

	State & District	Population	Male	Female	Literate	Male_Literate	Female_Litera
637	district_code:638, state_name:ANDAMAN AND NICO...	36842	20727	16115	25332	15397	99
638	district_code:639, state_name:ANDAMAN AND NICO...	105597	54861	50736	78683	43186	354
639	district_code:640, state_name:ANDAMAN AND NICO...	238142	127283	110859	190266	105794	844

3 rows × 51 columns

In [7]:

```
df.shape #use to getnumber of  row and cloumn
```

Out[7]:

```
(640, 51)
```

In [8]:

```
df.columns #to get column names
```

Out[8]:

```
Index(['State & District', 'Population', 'Male', 'Female', 'Literate',
       'Male_Literate', 'Female_Literate', 'SC', 'Male_SC', 'Female_SC',
'ST',
       'Male_ST', 'Female_ST', 'Workers', 'Male_Workers', 'Female_Worker
s',
       'Main_Workers', 'Marginal_Workers', 'Non_Workers', 'Cultivator_Work
ers',
       'Agricultural_Workers', 'Household_Workers', 'Other_Workers', 'Hind
us',
       'Muslims', 'Christians', 'Sikhs', 'Buddhists', 'Jains',
       'Others_Religions', 'Religion_Not_Stated', 'LPG_or_PNG_Households',
       'Housholds_with_Electric_Lighting', 'Households_with_Internet',
       'Households_with_Computer', 'Rural_Households', 'Urban_Households',
       'Households', 'Below_Primary_Education', 'Primary_Education',
       'Middle_Education', 'Secondary_Education', 'Higher_Education',
       'Graduate_Education', 'Other_Education', 'Literate_Education',
       'Illiterate_Education', 'Total_Education', 'Age_Group_0_29',
       'Age_Group_30_49', 'Age_Group_50'],
      dtype='object')
```

In [9]:

```
df.info()# to get column with null values and data types
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 640 entries, 0 to 639
Data columns (total 51 columns):
 #   Column                            Non-Null Count  Dtype 
---  ------                            --------------  ----- 
 0   State & District                  640 non-null    object
 1   Population                        640 non-null    int64 
 2   Male                              640 non-null    int64 
 3   Female                            640 non-null    int64 
 4   Literate                          640 non-null    int64 
 5   Male_Literate                     640 non-null    int64 
 6   Female_Literate                   640 non-null    int64 
 7   SC                                640 non-null    int64 
 8   Male_SC                           640 non-null    int64 
 9   Female_SC                         640 non-null    int64 
 10  ST                                640 non-null    int64 
 11  Male_ST                           640 non-null    int64 
 12  Female_ST                         640 non-null    int64 
 13  Workers                           640 non-null    int64 
 14  Male_Workers                      640 non-null    int64 
 15  Female_Workers                    640 non-null    int64 
 16  Main_Workers                      640 non-null    int64 
 17  Marginal_Workers                  640 non-null    int64 
 18  Non_Workers                       640 non-null    int64 
 19  Cultivator_Workers                640 non-null    int64 
 20  Agricultural_Workers              640 non-null    int64 
 21  Household_Workers                 640 non-null    int64 
 22  Other_Workers                     640 non-null    int64 
 23  Hindus                            640 non-null    int64 
 24  Muslims                           640 non-null    int64 
 25  Christians                        640 non-null    int64 
 26  Sikhs                             640 non-null    int64 
 27  Buddhists                         640 non-null    int64 
 28  Jains                             640 non-null    int64 
 29  Others_Religions                  640 non-null    int64 
 30  Religion_Not_Stated               640 non-null    int64 
 31  LPG_or_PNG_Households             640 non-null    int64 
 32  Housholds_with_Electric_Lighting  640 non-null    int64 
 33  Households_with_Internet          640 non-null    int64 
 34  Households_with_Computer          640 non-null    int64 
 35  Rural_Households                  640 non-null    int64 
 36  Urban_Households                  640 non-null    int64 
 37  Households                        640 non-null    int64 
 38  Below_Primary_Education           640 non-null    int64 
 39  Primary_Education                 640 non-null    int64 
 40  Middle_Education                  640 non-null    int64 
 41  Secondary_Education               640 non-null    int64 
 42  Higher_Education                  640 non-null    int64 
 43  Graduate_Education                640 non-null    int64 
 44  Other_Education                   640 non-null    int64 
 45  Literate_Education                640 non-null    int64 
 46  Illiterate_Education              640 non-null    int64 
 47  Total_Education                   640 non-null    int64 
 48  Age_Group_0_29                    640 non-null    int64 
 49  Age_Group_30_49                   640 non-null    int64 
 50  Age_Group_50                      640 non-null    int64 
dtypes: int64(50), object(1)
memory usage: 255.1+ KB
```

In [10]:

```
df.describe() #used to find statistical  information of each column
```

Out[10]:

	Population	Male	Female	Literate	Male_Literate	Female_Literat
count	6.400000e+02	6.400000e+02	6.400000e+02	6.400000e+02	6.400000e+02	6.400000e+0
mean	1.891961e+06	9.738598e+05	9.181011e+05	1.193186e+06	6.793182e+05	5.138675e+0
std	1.544380e+06	8.007785e+05	7.449864e+05	1.068583e+06	5.924144e+05	4.801816e+0
min	8.004000e+03	4.414000e+03	3.590000e+03	4.436000e+03	2.614000e+03	1.822000e+0
25%	8.178610e+05	4.171682e+05	4.017458e+05	4.825982e+05	2.764365e+05	2.008920e+0
50%	1.557367e+06	7.986815e+05	7.589200e+05	9.573465e+05	5.483525e+05	4.038590e+0
75%	2.583551e+06	1.338604e+06	1.264277e+06	1.602260e+06	9.188582e+05	6.641550e+0
max	1.106015e+07	5.865078e+06	5.195070e+06	8.227161e+06	4.591396e+06	3.635765e+0

8 rows × 50 columns

In [11]:

```
#Data Analyst what all things we do
#Missing values
#Explore about numerical variables
#Explore about categorical variables
#Finding relationship between features
```

In [12]:

```
df.isnull().sum() # It check Missing values or df.isna().sum() also used
```

Out[12]:

```
State & District                     0
Population                           0
Male                                 0
Female                               0
Literate                             0
Male_Literate                        0
Female_Literate                      0
SC                                   0
Male_SC                              0
Female_SC                            0
ST                                   0
Male_ST                              0
Female_ST                            0
Workers                              0
Male_Workers                         0
Female_Workers                       0
Main_Workers                         0
Marginal_Workers                     0
Non_Workers                          0
Cultivator_Workers                   0
Agricultural_Workers                 0
Household_Workers                    0
Other_Workers                        0
Hindus                               0
Muslims                              0
Christians                           0
Sikhs                                0
Buddhists                            0
Jains                                0
Others_Religions                     0
Religion_Not_Stated                  0
LPG_or_PNG_Households                0
Housholds_with_Electric_Lighting     0
Households_with_Internet             0
Households_with_Computer             0
Rural_Households                     0
Urban_Households                     0
Households                           0
Below_Primary_Education              0
Primary_Education                    0
Middle_Education                     0
Secondary_Education                  0
Higher_Education                     0
Graduate_Education                   0
Other_Education                      0
Literate_Education                   0
Illiterate_Education                 0
Total_Education                      0
Age_Group_0_29                       0
Age_Group_30_49                      0
Age_Group_50                         0
dtype: int64
```

In [12]:

```
```

In [13]:

```
[i for i in df.columns if df[i].isnull().sum()>0] # to get column that contain Missing  v
```

Out[13]:

```
[]
```

In [14]:

```
#There is no Missing Value in any column so no need to fill null values with inputation m
#or no need to drop column with null values
```

In [15]:

```
df.duplicated()# used to find duplicates
```

Out[15]:

```
0      False
1      False
2      False
3      False
4      False
       ...  
635    False
636    False
637    False
638    False
639    False
Length: 640, dtype: bool
```

In [16]:

```
#here there is need to  split coulmn state and district code
#because the feature 'State & District' is in the form of:'district_code:[District_Code]
#so by using str.split() method we can extract District_Code, State_Name, District_Name F
```

In [17]:

```
df['State & District']
```

Out[17]:

```
0      district_code:1, state_name:JAMMU AND KASHMIR,...
1      district_code:2, state_name:JAMMU AND KASHMIR,...
2      district_code:3, state_name:JAMMU AND KASHMIR,...
3      district_code:4, state_name:JAMMU AND KASHMIR,...
4      district_code:5, state_name:JAMMU AND KASHMIR,...
                             ...                        
635    district_code:636, state_name:PONDICHERRY, dis...
636    district_code:637, state_name:PONDICHERRY, dis...
637    district_code:638, state_name:ANDAMAN AND NICO...
638    district_code:639, state_name:ANDAMAN AND NICO...
639    district_code:640, state_name:ANDAMAN AND NICO...
Name: State & District, Length: 640, dtype: object
```

In [18]:

```
df[['District_code','State_name','District_name']]=df['State & District'].str.split(',',e
```

In [19]:

```
df
```

Out[19]:

	State & District	Population	Male	Female	Literate	Male_Literate	Female_l
0	district_code:1, state_name:JAMMU AND KASHMIR,...	870354	474190	396164	439654	282823	
1	district_code:2, state_name:JAMMU AND KASHMIR,...	753745	398041	355704	335649	207741	
2	district_code:3, state_name:JAMMU AND KASHMIR,...	133487	78971	54516	93770	62834	
3	district_code:4, state_name:JAMMU AND KASHMIR,...	140802	77785	63017	86236	56301	
4	district_code:5, state_name:JAMMU AND KASHMIR,...	476835	251899	224936	261724	163333	
...	...	...	...	...	...	...	
635	district_code:636, state_name:PONDICHERRY, dis...	41816	19143	22673	36470	16610	
636	district_code:637, state_name:PONDICHERRY, dis...	200222	97809	102413	154916	79903	
637	district_code:638, state_name:ANDAMAN AND NICO...	36842	20727	16115	25332	15397	
638	district_code:639, state_name:ANDAMAN AND NICO...	105597	54861	50736	78683	43186	
639	district_code:640, state_name:ANDAMAN AND NICO...	238142	127283	110859	190266	105794	

640 rows × 54 columns

In [20]:

```
df['District_code'] = df['State & District'].str.extract('district_code:(\d+)')
```

In [21]:

```python
df.head()
```

Out[21]:

	State & District	Population	Male	Female	Literate	Male_Literate	Female_Literate
0	district_code:1, state_name:JAMMU AND KASHMIR,...	870354	474190	396164	439654	282823	156831
1	district_code:2, state_name:JAMMU AND KASHMIR,...	753745	398041	355704	335649	207741	127908
2	district_code:3, state_name:JAMMU AND KASHMIR,...	133487	78971	54516	93770	62834	30936
3	district_code:4, state_name:JAMMU AND KASHMIR,...	140802	77785	63017	86236	56301	29935
4	district_code:5, state_name:JAMMU AND KASHMIR,...	476835	251899	224936	261724	163333	98391

5 rows × 54 columns

In [22]:

```python
def extract_state_name(row):
    if 'state_name:' in row:
        return row.split('state_name:', 1)[1].rstrip(',')
    return row

# Apply the function to apply function on each row
df['State_name'] = df['State_name'].apply(extract_state_name)
```

In [23]:

```
df.head()
```

Out[23]:

	State & District	Population	Male	Female	Literate	Male_Literate	Female_Literate	
0	district_code:1, state_name:JAMMU AND KASHMIR,...	870354	474190	396164	439654	282823	156831	1(
1	district_code:2, state_name:JAMMU AND KASHMIR,...	753745	398041	355704	335649	207741	127908	:
2	district_code:3, state_name:JAMMU AND KASHMIR,...	133487	78971	54516	93770	62834	30936	4
3	district_code:4, state_name:JAMMU AND KASHMIR,...	140802	77785	63017	86236	56301	29935	
4	district_code:5, state_name:JAMMU AND KASHMIR,...	476835	251899	224936	261724	163333	98391	!

5 rows × 54 columns

In [24]:

```
def extract_state_name(row):
    if 'district_name:' in row:
        return row.split('district_name:', 1)[1].rstrip(',')
    return row
df['District_name'] = df['District_name'].apply(extract_state_name)
```

In [25]:

```
df.head()
```

Out[25]:

	State & District	Population	Male	Female	Literate	Male_Literate	Female_Literate
0	district_code:1, state_name:JAMMU AND KASHMIR,...	870354	474190	396164	439654	282823	156831
1	district_code:2, state_name:JAMMU AND KASHMIR,...	753745	398041	355704	335649	207741	127908
2	district_code:3, state_name:JAMMU AND KASHMIR,...	133487	78971	54516	93770	62834	30936
3	district_code:4, state_name:JAMMU AND KASHMIR,...	140802	77785	63017	86236	56301	29935
4	district_code:5, state_name:JAMMU AND KASHMIR,...	476835	251899	224936	261724	163333	98391

5 rows × 54 columns

In [26]:

```
df.drop('State & District',axis=1,inplace=True)
```

In [27]:

```
df.head()
```

Out[27]:

	Population	Male	Female	Literate	Male_Literate	Female_Literate	SC	Male_SC	Fema
0	870354	474190	396164	439654	282823	156831	1048	1046	
1	753745	398041	355704	335649	207741	127908	368	343	
2	133487	78971	54516	93770	62834	30936	488	444	
3	140802	77785	63017	86236	56301	29935	18	12	
4	476835	251899	224936	261724	163333	98391	556	406	

5 rows × 53 columns

In [71]:

```
#perfrom nominal one hot encoding for state and district to convert this data into numeri
#During model building here now not need to perform it
```

In [72]:

```
#df['State_name'].value_counts().sum()
```

Out[72]:

```
640
```

In [73]:

```
#df['State_name'].nunique()
```

Out[73]:

```
35
```

In [74]:

```
#df['State_name'].value_counts()
```

Out[74]:

```
UTTAR PRADESH                  71
MADHYA PRADESH                 50
BIHAR                          38
MAHARASHTRA                    35
RAJASTHAN                      33
TAMIL NADU                     32
KARNATAKA                      30
ORISSA                         30
ASSAM                          27
GUJARAT                        26
JHARKHAND                      24
ANDHRA PRADESH                 23
JAMMU AND KASHMIR              22
HARYANA                        21
PUNJAB                         20
WEST BENGAL                    19
CHHATTISGARH                   18
ARUNACHAL PRADESH              16
KERALA                         14
UTTARAKHAND                    13
HIMACHAL PRADESH               12
NAGALAND                       11
MANIPUR                         9
NCT OF DELHI                    9
MIZORAM                         8
MEGHALAYA                       7
TRIPURA                         4
SIKKIM                          4
PONDICHERRY                     4
ANDAMAN AND NICOBAR ISLANDS     3
GOA                             2
DAMAN AND DIU                   2
LAKSHADWEEP                     1
CHANDIGARH                      1
DADRA AND NAGAR HAVELI          1
Name: State_name, dtype: int64
```

In [75]:

```
#df1=pd.get_dummies(df,columns=['State_name','District_name'])
```

In [76]:

```
#df1.shape
```

Out[76]:

```
(640, 720)
```

In [77]:

```
#pd.get_dummies(df,columns=['State_name','District_name'],drop_first=True).head()
```

Out[77]:

	Population	Male	Female	Literate	Male_Literate	Female_Literate	SC	Male_SC	Fema
0	870354	474190	396164	439654	282823	156831	1048	1046	
1	753745	398041	355704	335649	207741	127908	368	343	
2	133487	78971	54516	93770	62834	30936	488	444	
3	140802	77785	63017	86236	56301	29935	18	12	
4	476835	251899	224936	261724	163333	98391	556	406	

5 rows × 718 columns

Task 2: Sub Assignment 2: Exploring Indian States Data [Level Medium]

In [30]:

```
df.head()
```

Out[30]:

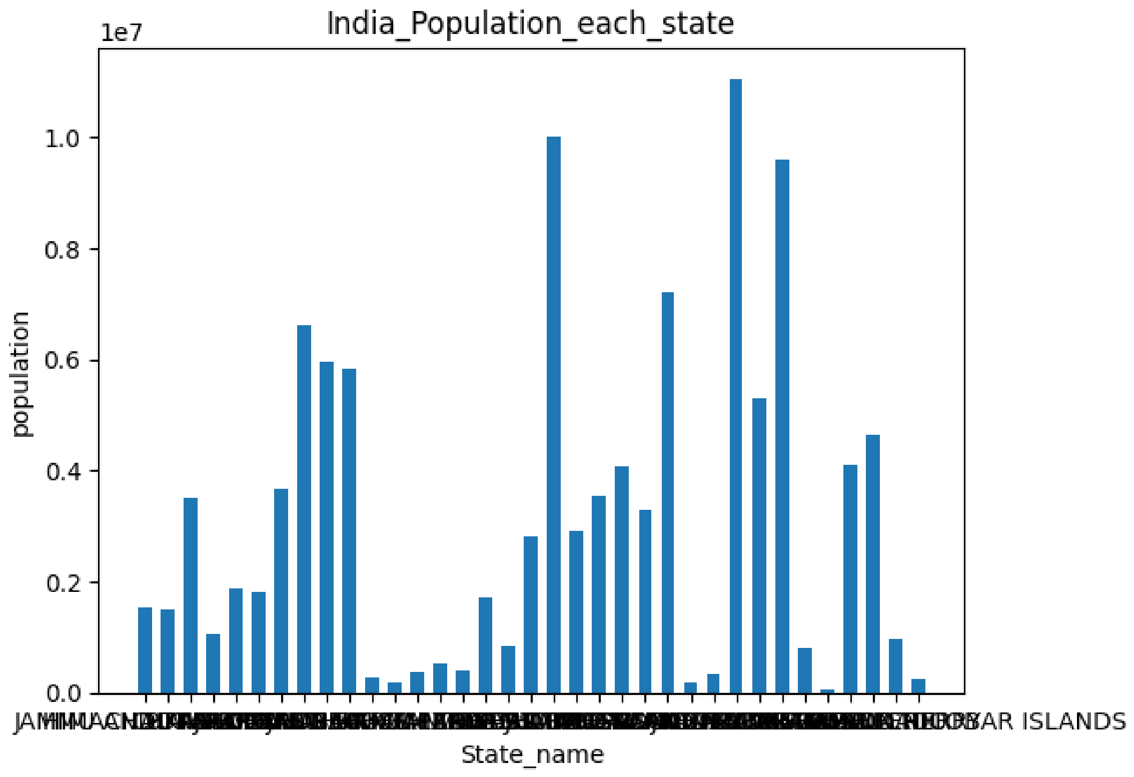
	Population	Male	Female	Literate	Male_Literate	Female_Literate	SC	Male_SC	Fema
0	870354	474190	396164	439654	282823	156831	1048	1046	
1	753745	398041	355704	335649	207741	127908	368	343	
2	133487	78971	54516	93770	62834	30936	488	444	
3	140802	77785	63017	86236	56301	29935	18	12	
4	476835	251899	224936	261724	163333	98391	556	406	

5 rows × 53 columns

1.What is the population of each state?

In [43]:

```
import matplotlib.pyplot as plt
x=df['State_name']
y=df['Population']
plt.xlabel('State_name',fontsize=10) #x axis label
plt.ylabel('population')
plt.title('India_Population_each_state')#title
plt.bar(x,y,width=0.6)
plt.show()

#this code for vertical bar graph and for horizontal bar graph we use(plt.barh())
```



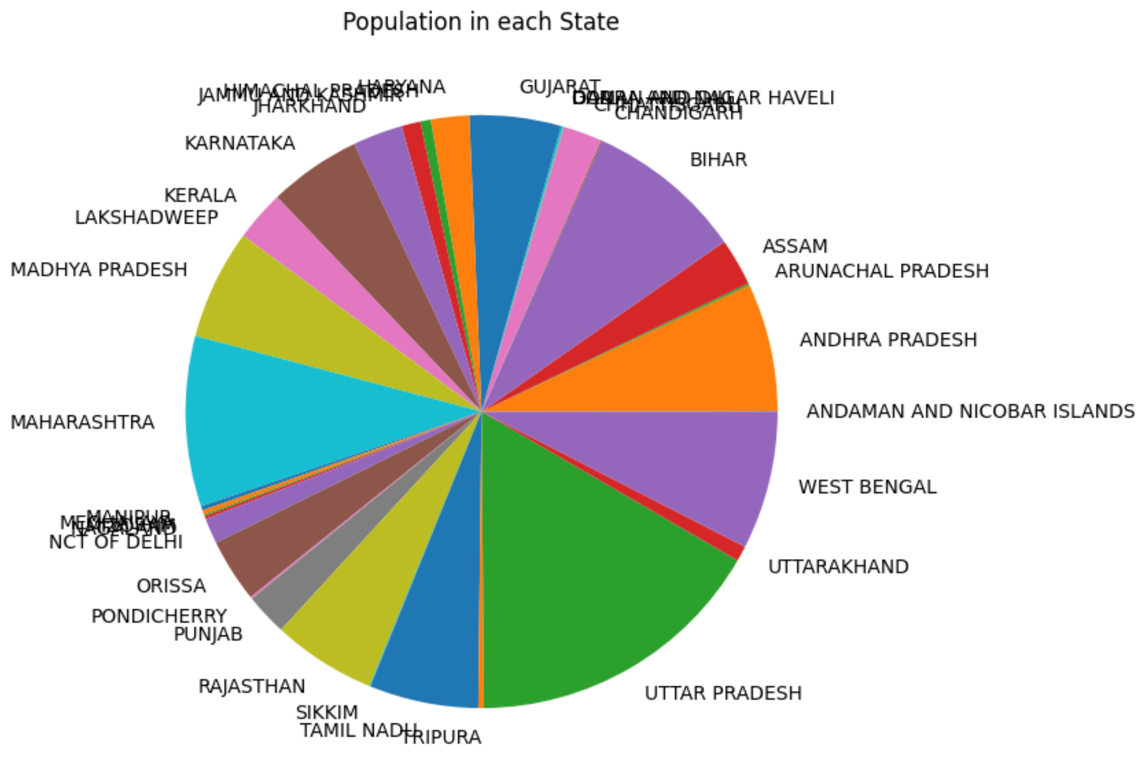
we can also use the pie chart for this

In [39]:

```
sp= df.groupby('State_name')['Population'].sum()
f= plt.figure(figsize =(10, 7))
plt.pie(sp, labels = sp.index.tolist())
plt.title('Population in each State')
```

Out[39]:

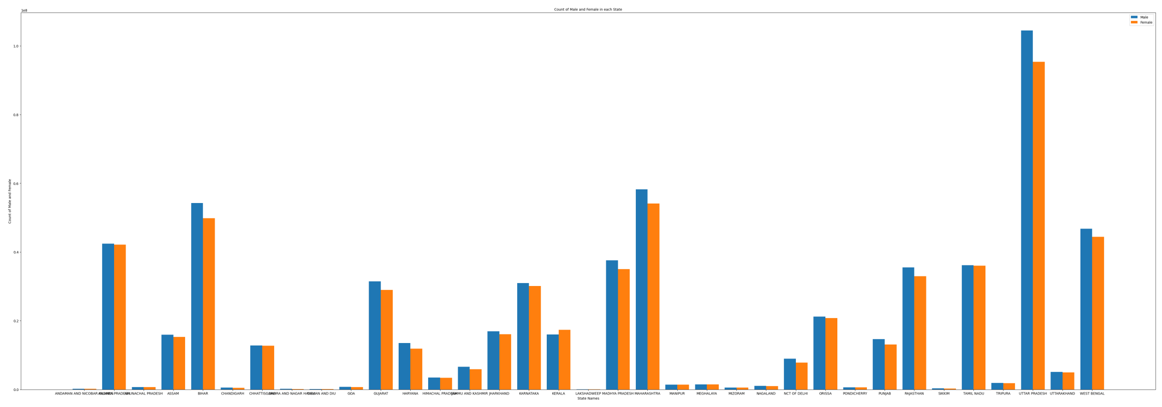
```
Text(0.5, 1.0, 'Population in each State')
```



2.How does the gender ratio vary across different states and districts?

In [52]:

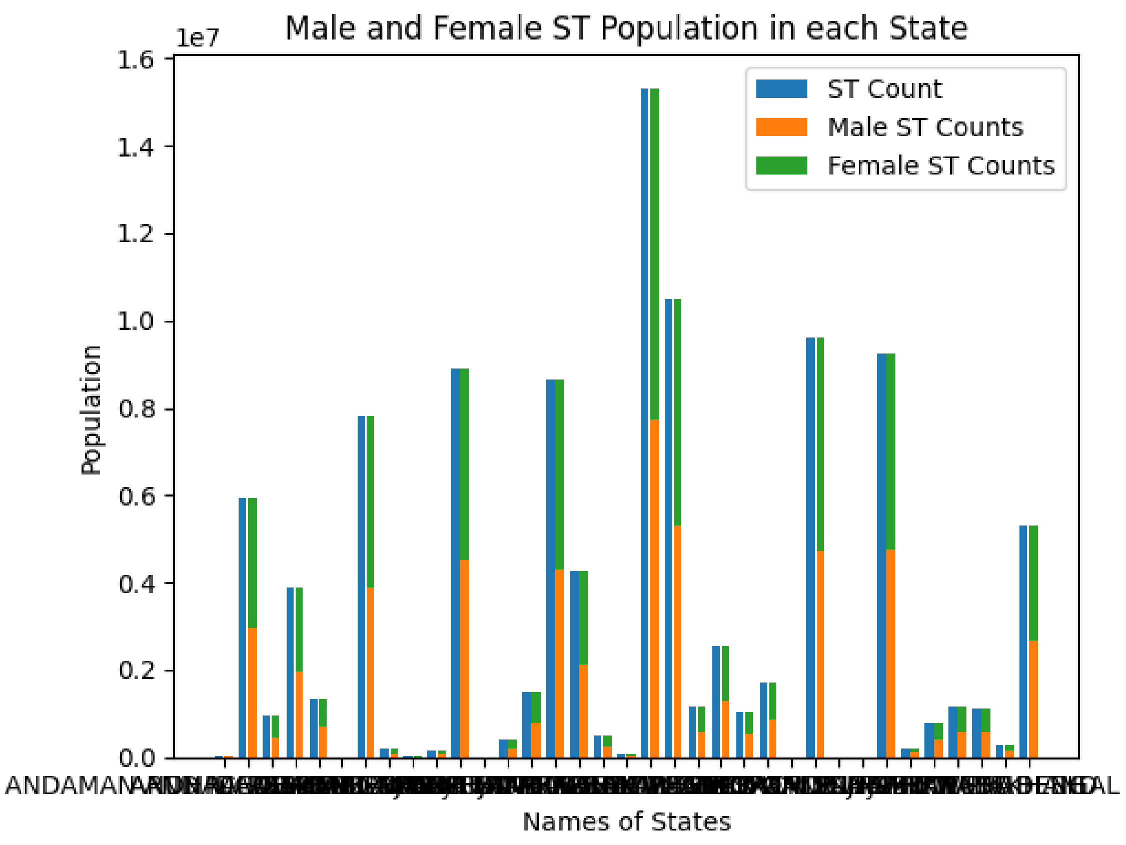
```
import numpy as np
state_male = df.groupby('State_name')['Male'].sum()
state_female = df.groupby('State_name')['Female'].sum()
merged_counts = pd.concat([state_male, state_female], axis=1)
# Compare number of Male and Female in each State
state_names = list(state_male.index.tolist())
X_axis = np.arange(len(state_names))
fig = plt.figure(figsize = (60, 20))
plt.bar(X_axis - 0.2, merged_counts['Male'], 0.4, label = 'Male')
plt.bar(X_axis + 0.2, merged_counts['Female'], 0.4, label = 'Female')
plt.xticks(X_axis, state_names)
plt.xlabel("State Names")
plt.ylabel("Count of Male and Female",fontsize=10)
plt.title("Count of Male and Female in each State",fontsize=10)
plt.legend()
plt.show()
```



3. How much ST Male and Female Population present in each State?

In [58]:

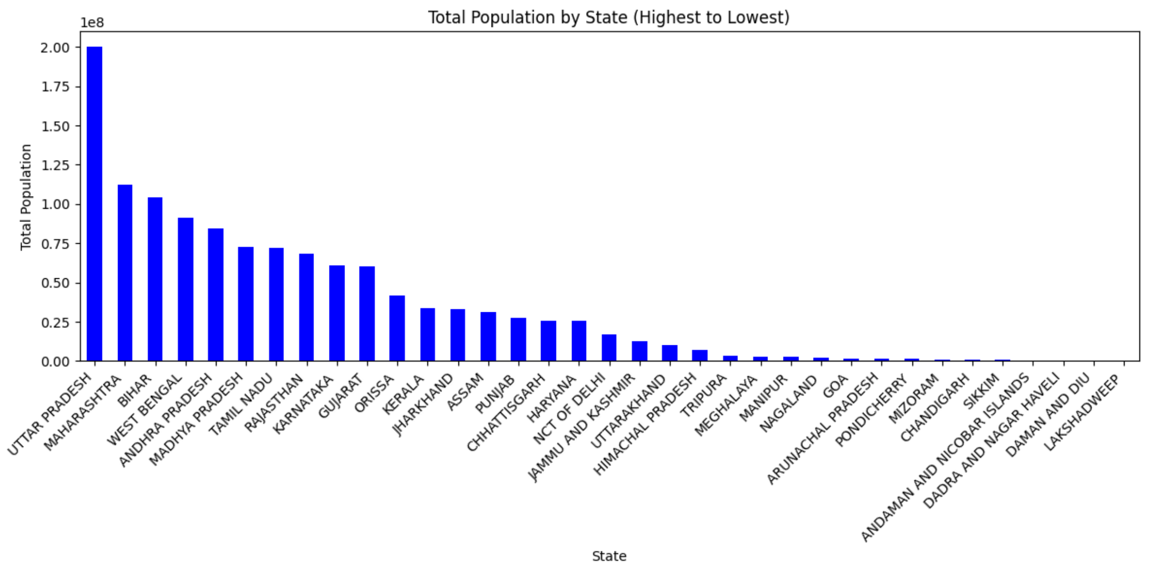
```
state_ST_counts = df.groupby('State_name')['ST'].sum()
state_Male_ST_counts = df.groupby('State_name')['Male_ST'].sum()
state_Female_ST_counts = df.groupby('State_name')['Female_ST'].sum()

# Create DataFrame from all these counts
lst = [state_ST_counts, state_Male_ST_counts, state_Female_ST_counts]
merged_ST_counts = pd.concat(lst, axis=1)

x = np.arange(len(merged_ST_counts))

categories = merged_ST_counts.index.tolist()

# Create a figure and axis
fig, ax = plt.subplots()

# Plot the simple bar
ax.bar(x-0.2, merged_ST_counts['ST'], label='ST Count', width=0.35)

# Plot the stacked bars
ax.bar(x+0.2, merged_ST_counts['Male_ST'], label='Male ST Counts', width=0.35)
ax.bar(x+0.2, merged_ST_counts['Female_ST'], label='Female ST Counts', width=0.35,
       bottom=merged_ST_counts['Male_ST'])

# Set labels and title
ax.set_xlabel('Names of States')
ax.set_ylabel('Population')
ax.set_title('Male and Female ST Population in each State')
ax.set_xticks(x)
ax.set_xticklabels(categories)
ax.legend()

# Show the plot
plt.tight_layout()
plt.show()
```



4.Which states have the highest and lowest total populations?

In [60]:

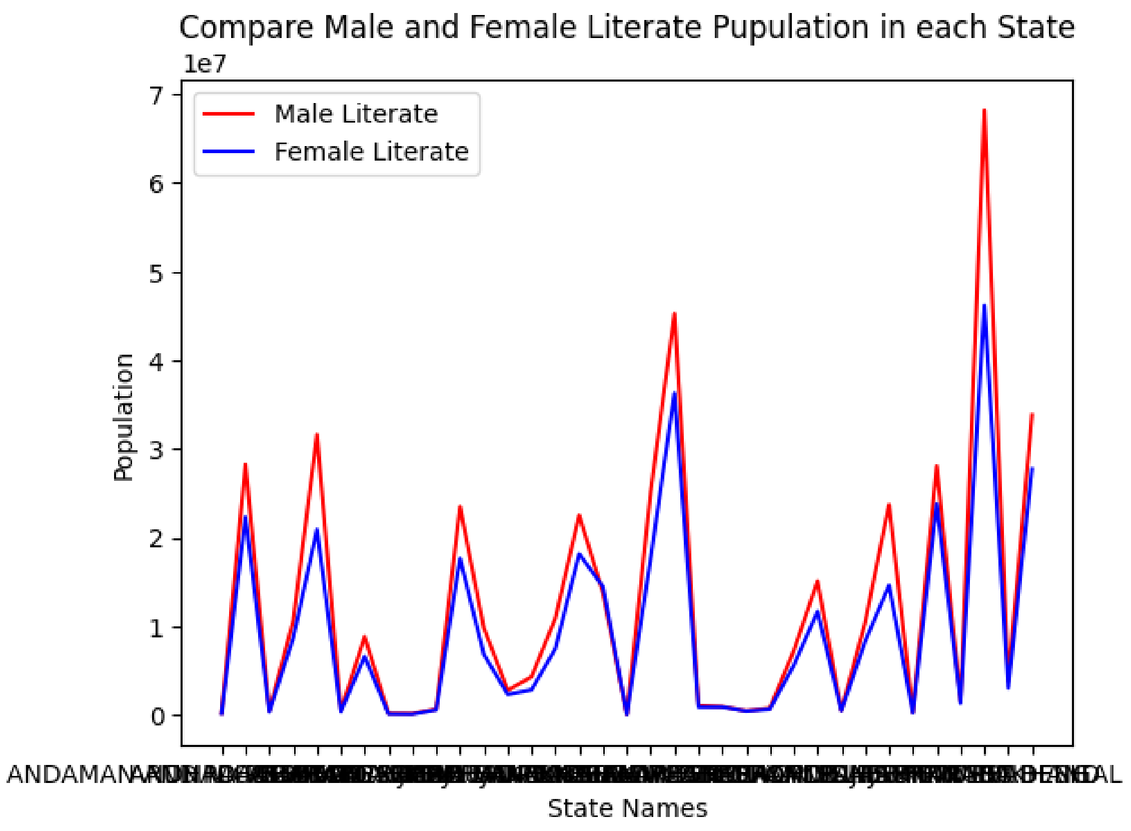
```
state_population = df.groupby('State_name')['Population'].sum()

# Sort the data in ascending order to find the states with the lowest population
state_population_sorted = state_population.sort_values(ascending=False)

# Create a bar graph
plt.figure(figsize=(12, 6))
state_population_sorted.plot(kind='bar', color='blue')
plt.xlabel('State')
plt.ylabel('Total Population')
plt.title('Total Population by State (Highest to Lowest)')
plt.xticks(rotation=45, ha="right")
plt.tight_layout()
plt.show()
```



5. What is the Population of Literate Male and Female in each State?

In [70]:

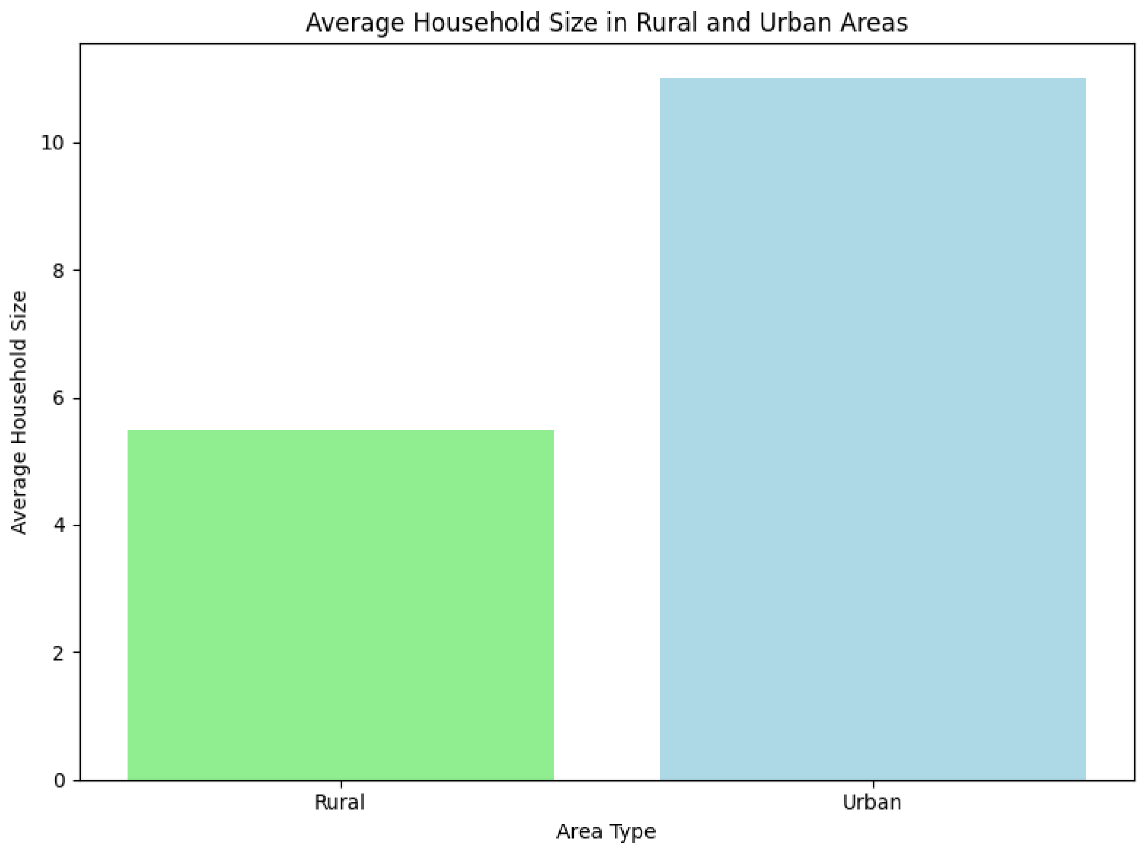
```
# Create the plot with multiple lines
state_Literate_counts = df.groupby('State_name')['Literate'].sum()
state_Male_Literate_counts = df.groupby('State_name')['Male_Literate'].sum()
state_Female_Literate_counts = df.groupby('State_name')['Female_Literate'].sum()
lst = [state_Literate_counts, state_Male_Literate_counts, state_Female_Literate_counts]
merged_Literate_counts = pd.concat(lst, axis=1)

state_names = merged_Literate_counts.index.tolist()
plt.plot(state_names, merged_Literate_counts['Male_Literate'], label='Male Literate', col
plt.plot(state_names, merged_Literate_counts['Female_Literate'], label='Female Literate',

# Customize the plot
plt.xlabel('State Names')
plt.ylabel('Population')
plt.title('Compare Male and Female Literate Pupulation in each State')
plt.legend()  # Show legend with labels

# Show the plot
plt.show()
```



6.What is the average household size in urban areas compared to rural areas?

In [77]:

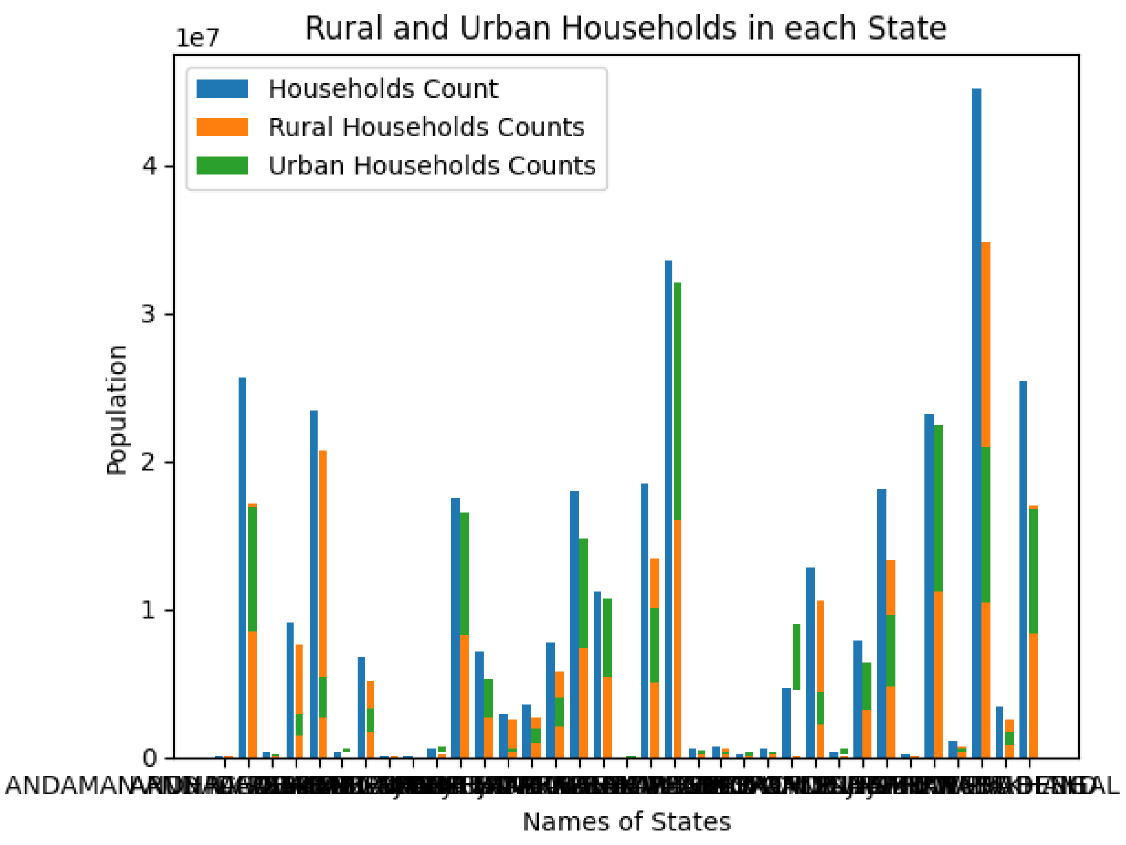
```
total_rural_households = df['Rural_Households'].sum()
total_urban_households = df['Urban_Households'].sum()
total_population = df['Population'].sum()

average_rural_household_size = total_population / total_rural_households
average_urban_household_size = total_population / total_urban_households

# Create a grouped bar chart to visualize the comparison
locations = ['Rural', 'Urban']
average_sizes = [average_rural_household_size, average_urban_household_size]

plt.figure(figsize=(8, 6))
plt.bar(locations, average_sizes, color=['lightgreen', 'lightblue'])
plt.xlabel('Area Type')
plt.ylabel('Average Household Size')
plt.title('Average Household Size in Rural and Urban Areas')
plt.tight_layout()
plt.show()
```



7. Households Distribution in each State?

In [80]:

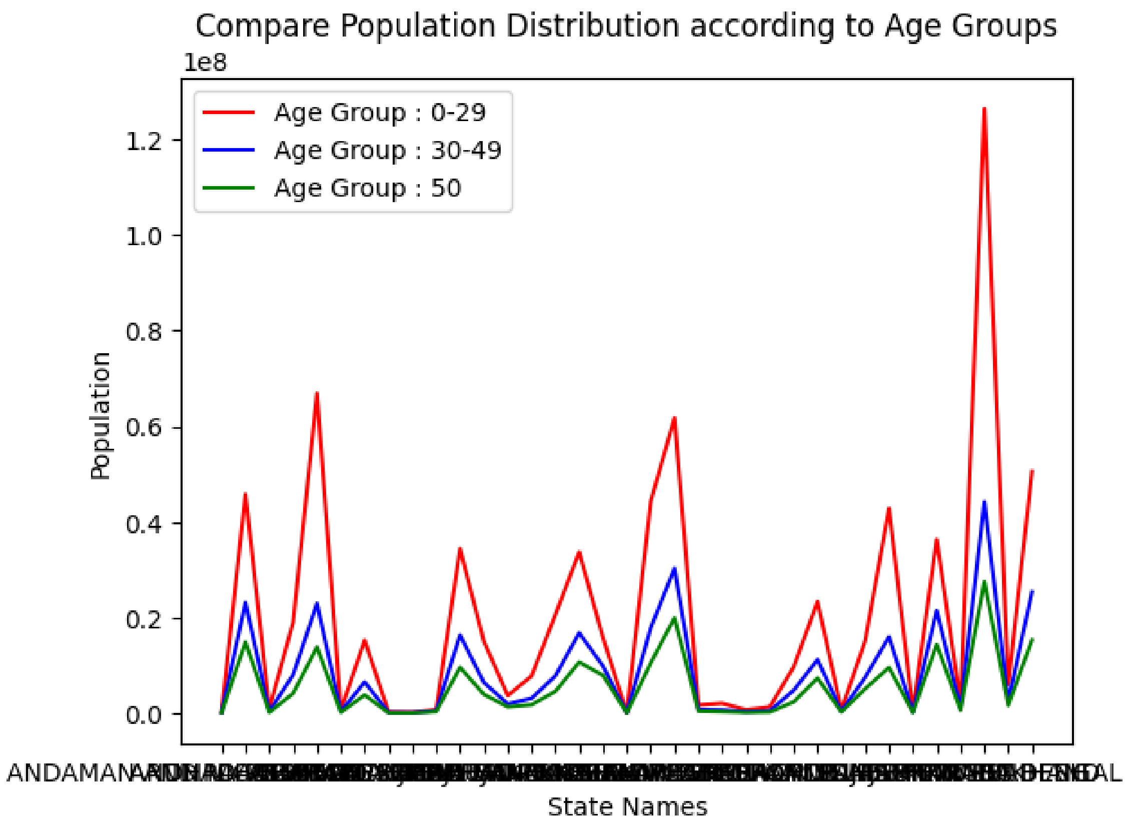
```
# Find sum of different types of Household Distributions in each state

state_Households_counts = df.groupby('State_name')['Households'].sum()
state_LPG_or_PNG_Households_counts = df.groupby('State_name')['LPG_or_PNG_Households'].su
state_Housholds_with_Electric_Lighting_counts = df.groupby('State_name')['Housholds_with_
state_Households_with_Internet_counts = df.groupby('State_name')['Households_with_Interne
state_Households_with_Computer_counts = df.groupby('State_name')['Households_with_Compute
state_Rural_Households_counts = df.groupby('State_name')['Rural_Households'].sum()
state_Urban_Households_counts = df.groupby('State_name')['Urban_Households'].sum()


lst = [state_LPG_or_PNG_Households_counts, state_Households_counts,
       state_Housholds_with_Electric_Lighting_counts, state_Households_with_Internet_cour
       state_Rural_Households_counts, state_Urban_Households_counts,
       state_Households_with_Computer_counts]
merged_Households_counts = pd.concat(lst, axis=1)
# Compare Rural and Urban Households in each State

# Create an array for the x-axis positions
x = np.arange(len(merged_Households_counts))

categories = merged_Households_counts.index.tolist()

# Create a figure and axis
fig, ax = plt.subplots()

# Plot the simple bar
ax.bar(x-0.2, merged_Households_counts['Households'], label='Households Count', width=0.3

# Plot the stacked bars
ax.bar(x+0.2, merged_Households_counts['Rural_Households'], label='Rural Households Count
       width=0.35)
ax.bar(x+0.2, merged_Households_counts['Urban_Households'], label='Urban Households Count
       width=0.35, bottom=merged_Households_counts['Urban_Households'])

# Set labels and title
ax.set_xlabel('Names of States')
ax.set_ylabel('Population')
ax.set_title('Rural and Urban Households in each State')
ax.set_xticks(x)
ax.set_xticklabels(categories)
ax.legend()

# Show the plot
plt.tight_layout()
plt.show()
```



8. Population Distribution of State according to Age Groups?

In [84]:

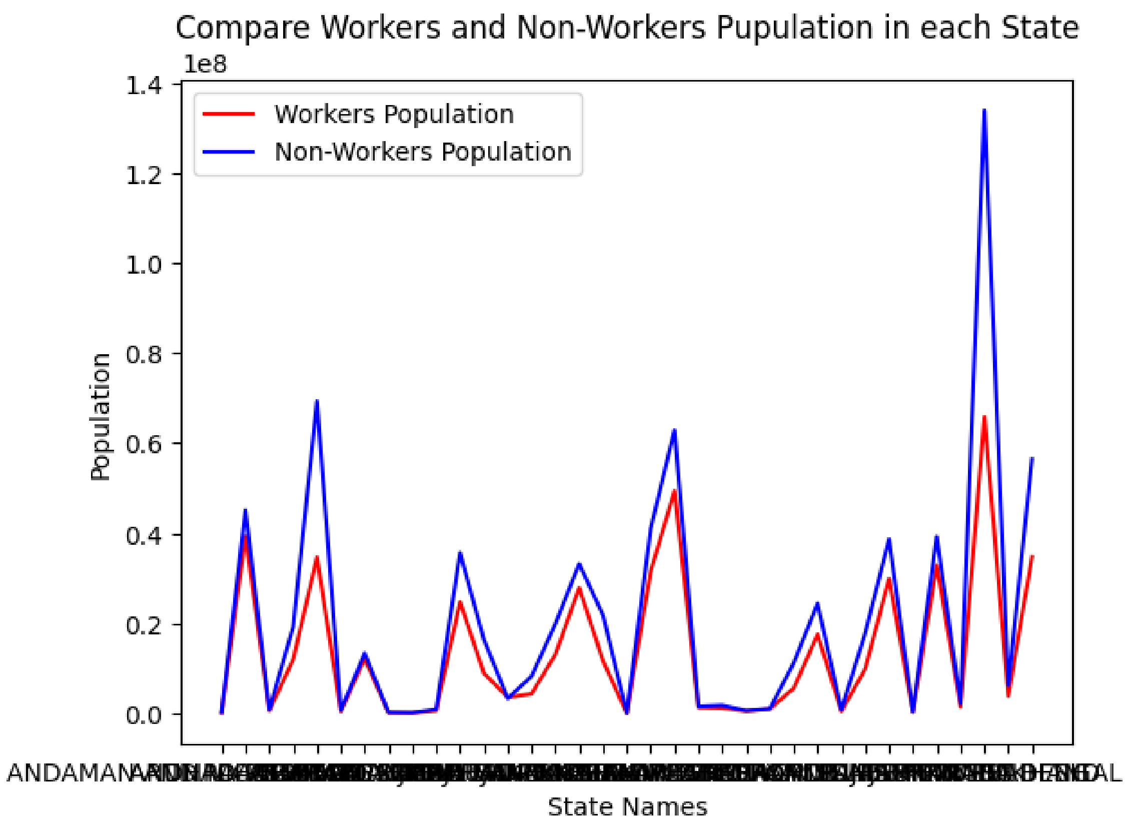
```
state_Age_Group_0_29_counts = df.groupby('State_name')['Age_Group_0_29'].sum()
state_Age_Group_30_49_counts = df.groupby('State_name')['Age_Group_30_49'].sum()
state_Age_Group_50_counts = df.groupby('State_name')['Age_Group_50'].sum()

# Create DataFrame from all these counts
merged_counts = pd.concat([state_Age_Group_0_29_counts, state_Age_Group_30_49_counts,
                           state_Age_Group_50_counts], axis=1)

# Create the plot with multiple lines

# fig = plt.figure(figsize = (50, 25))
names = merged_counts.index.tolist()

plt.plot(names, merged_counts['Age_Group_0_29'], label='Age Group : 0-29', color='r')
plt.plot(names, merged_counts['Age_Group_30_49'], label='Age Group : 30-49', color='b')
plt.plot(names, merged_counts['Age_Group_50'], label='Age Group : 50', color='g')

# Customize the plot
plt.xlabel('State Names')
plt.ylabel('Population')
plt.title('Compare Population Distribution according to Age Groups')
plt.legend()  # Show legend with labels

# Show the plot
plt.show()
```



9.How much Worker and Non-Worker Population present in each State ?

In [86]:

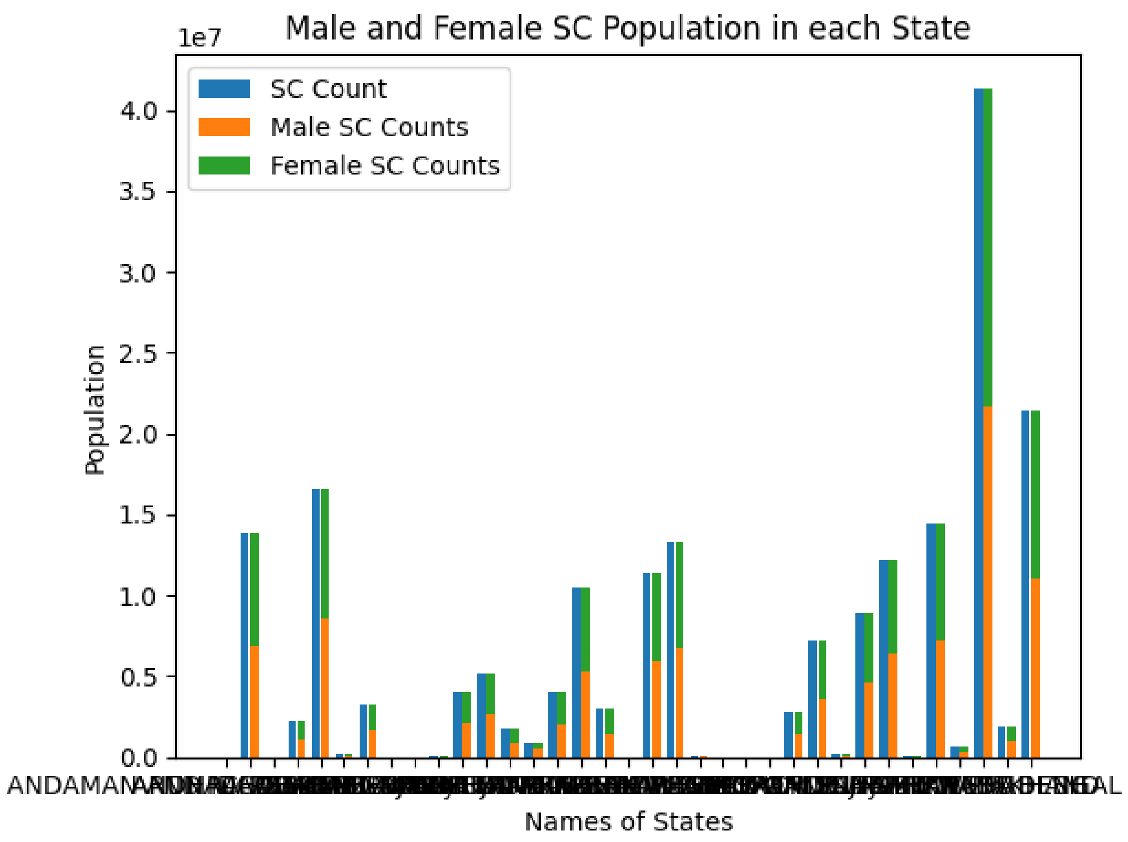
```
state_Workers_counts = df.groupby('State_name')['Workers'].sum()
state_Male_Workers_counts = df.groupby('State_name')['Male_Workers'].sum()
state_Female_Workers_counts = df.groupby('State_name')['Female_Workers'].sum()
state_Main_Workers_counts = df.groupby('State_name')['Main_Workers'].sum()
state_Marginal_Workers_counts = df.groupby('State_name')['Marginal_Workers'].sum()
state_Non_Workers_counts = df.groupby('State_name')['Non_Workers'].sum()
state_Cultivator_Workers_counts = df.groupby('State_name')['Cultivator_Workers'].sum()
state_Agricultural_Workers_counts = df.groupby('State_name')['Agricultural_Workers'].sum(
state_Household_Workers_counts = df.groupby('State_name')['Household_Workers'].sum()
state_Other_Workers_counts = df.groupby('State_name')['Other_Workers'].sum()

# Create DataFrame from all these counts
lst = [state_Workers_counts, state_Male_Workers_counts, state_Female_Workers_counts,
       state_Main_Workers_counts, state_Marginal_Workers_counts, state_Non_Workers_counts
       state_Cultivator_Workers_counts, state_Agricultural_Workers_counts,
       state_Household_Workers_counts, state_Other_Workers_counts]
merged_Workers_counts = pd.concat(lst, axis=1)

# Create the plot with multiple lines

# fig = plt.figure(figsize = (50, 25))

state_names = merged_Workers_counts.index.tolist()
plt.plot(state_names, merged_Workers_counts['Workers'], label='Workers Population', color
plt.plot(state_names, merged_Workers_counts['Non_Workers'], label='Non-Workers Populatior

# Customize the plot
plt.xlabel('State Names')
plt.ylabel('Population')
plt.title('Compare Workers and Non-Workers Pupulation in each State')
plt.legend()  # Show legend with labels

# Show the plot
plt.show()
```



10.How much SC Male and Female Population present in each State ?

In [90]:

```
# Find sum of SC, Male_SC and Female_SC population in each state
# using groupby() method

state_SC_counts = df.groupby('State_name')['SC'].sum()
state_Male_SC_counts = df.groupby('State_name')['Male_SC'].sum()
state_Female_SC_counts = df.groupby('State_name')['Female_SC'].sum()

# Create DataFrame from all these counts
lst = [state_SC_counts, state_Male_SC_counts, state_Female_SC_counts]
merged_SC_counts = pd.concat(lst, axis=1)

# Create an array for the x-axis positions
x = np.arange(len(merged_SC_counts))

categories = merged_SC_counts.index.tolist()

# Create a figure and axis
fig, ax = plt.subplots()

# Plot the simple bar
ax.bar(x-0.2, merged_SC_counts['SC'], label='SC Count', width=0.35)

# Plot the stacked bars
ax.bar(x+0.2, merged_SC_counts['Male_SC'], label='Male SC Counts', width=0.35)
ax.bar(x+0.2, merged_SC_counts['Female_SC'], label='Female SC Counts', width=0.35,
       bottom=merged_SC_counts['Male_SC'])

# Set labels and title
ax.set_xlabel('Names of States')
ax.set_ylabel('Population')
ax.set_title('Male and Female SC Population in each State')
ax.set_xticks(x)
ax.set_xticklabels(categories)
ax.legend()

# Show the plot
plt.tight_layout()
plt.show()
```


Male and Female SC Population in each State
1e7
SC Count
Male SC Counts
Female SC Counts
Population
Names of States
ANDAMAN
WEST BENGAL

```
In [ ]:
```