

Data Analysis in MHA Population Dataset from Kaggle

```
In [53]: # At first, let's import some useful libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

Data pre-processing

```
In [54]: # Now let's import our data from our Computer.
My_Data = pd.read_csv(r"C:/Users/USER/Downloads/archive (29)/MHA Population")
```

```
In [55]: # Let's understand our data
My_Data.shape
```

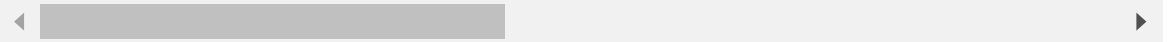
Out[55]: (132315, 29)

```
In [56]: # Show the first 5 rows
My_Data.head()
```

Out[56]:

	Census Year	District	Taluka	Town/Village	No. of households	Total population	Total male population	pop
0	2011	AHMADNAGAR	AKOLA	ABIT KHIND	201	732	359	
1	2011	AHMADNAGAR	AKOLA	AGAR	37	247	162	
2	2011	AHMADNAGAR	AKOLA	AGASTINAGAR	357	1536	799	
3	2011	AHMADNAGAR	AKOLA	AKOLA	3861	18278	9381	
4	2011	AHMADNAGAR	AKOLA	AMBAD	529	2590	1352	

5 rows × 29 columns



```
In [57]: # Columns of Dataset
My_Data.columns
```

```
Out[57]: Index(['Census Year', 'District', 'Taluka', 'Town/Village',
               'No. of households', 'Total population', 'Total male population',
               'Total female population', 'Total 0 to 6 year children',
               'Male 0 to 6 year children', 'Female 0 to 6 year children',
               'Total SC population', 'Male SC population', 'Female SC populatio
n',
               'Total ST population', 'Male ST population', 'Female ST populatio
n',
               'Total literates', 'Male literates', 'Female literates',
               'Total illiterates', 'Male illiterates', 'Female illiterates',
               'Total main workers', 'Male main workers', 'Female main workers',
               'Total non workers', 'Male non workers', 'Female non workers'],
              dtype='object')
```

```
In [58]: # Types of the columns
My_Data.dtypes
```

```
Out[58]: Census Year          int64
District          object
Taluka            object
Town/Village      object
No. of households  int64
Total population   int64
Total male population int64
Total female population int64
Total 0 to 6 year children int64
Male 0 to 6 year children int64
Female 0 to 6 year children int64
Total SC population int64
Male SC population int64
Female SC population int64
Total ST population int64
Male ST population int64
Female ST population int64
Total literates     int64
Male literates      int64
Female literates     int64
Total illiterates    float64
Male illiterates     float64
Female illiterates    float64
Total main workers   int64
Male main workers    int64
Female main workers  int64
Total non workers    int64
Male non workers     int64
Female non workers   int64
dtype: object
```

```
In [59]: # Let's see for missing values
missing_values = My_Data.isna().sum()

# Print the result
missing_values
```

```
Out[59]: Census Year      0
District      0
Taluka        16
Town/Village  1036
No. of households      0
Total population      0
Total male population  0
Total female population      0
Total 0 to 6 year children      0
Male 0 to 6 year children      0
Female 0 to 6 year children      0
Total SC population      0
Male SC population      0
Female SC population      0
Total ST population      0
Male ST population      0
Female ST population      0
Total literates      0
Male literates      0
Female literates      0
Total illiterates    49347
Male illiterates    49347
Female illiterates    49347
Total main workers      0
Male main workers      0
Female main workers      0
Total non workers      0
Male non workers      0
Female non workers      0
dtype: int64
```

As we see, there are 49347 missing values from three columns: Total literates, Male illiterates and Female illiterates. Let's create new columns that they will not have missing values using some other columns in our calculations.

```
In [60]: #Create a new column : Total_iliterates
My_Data['Total_iliterates'] = np.where(
    (My_Data['Total population'].isna())
    | (My_Data['Total literates'].isna()),
    np.nan,
    My_Data['Total population'] - My_Data['Total literates'])

# Create a new column : Male_iliterates
My_Data['Male_iliterates'] = np.where(
    (My_Data['Total male population'].isna())
    | (My_Data['Male literates'].isna()),
    np.nan,
    My_Data['Total male population'] - My_Data['Male literates'])

# Create a new column : Female_iliterates
My_Data['Female_iliterates'] = np.where(
    (My_Data['Total female population'].isna())
    | (My_Data['Female literates'].isna()),
    np.nan,
    My_Data['Total female population'] - My_Data['Female literates'])
```

```
In [61]: #Remove the old columns,Total iliterates, Male iliterates,Female iliterates
My_Data.drop(['Total iliterates', 'Male iliterates', 'Female iliterates'],
             axis=1, inplace=True)
```

We need to change the type of the elements of our new columns to integers so that they don't create any problems in our analysis afterwards.

```
In [62]: # Convert Total_iliterates, Male_iliterates, Female_iliterates to integers
My_Data['Total_iliterates'] = My_Data['Total_iliterates'].astype('int64')
My_Data['Male_iliterates'] = My_Data['Male_iliterates'].astype('int64')
My_Data['Female_iliterates'] = My_Data['Female_iliterates'].astype('int64')
```

```
In [63]: # Ok let's see our data frame's types again
My_Data.dtypes
```

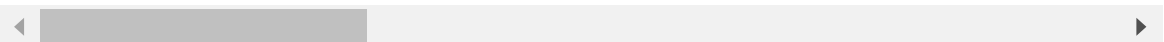
```
Out[63]: Census Year          int64
District          object
Taluka            object
Town/Village      object
No. of households  int64
Total population   int64
Total male population int64
Total female population int64
Total 0 to 6 year children int64
Male 0 to 6 year children int64
Female 0 to 6 year children int64
Total SC population int64
Male SC population int64
Female SC population int64
Total ST population int64
Male ST population int64
Female ST population int64
Total literates    int64
Male literates     int64
Female literates   int64
Total main workers int64
Male main workers  int64
Female main workers int64
Total non workers  int64
Male non workers   int64
Female non workers int64
Total_illiterates  int64
Male_illiterates   int64
Female_illiterates int64
dtype: object
```

```
In [64]: # Check some statistics about numeric data
My_Data.describe()
```

```
Out[64]:
```

	Census Year	No. of households	Total population	Total male population	Total female population	Total 0 to 6 year childrer
count	132315.000000	1.323150e+05	1.323150e+05	1.323150e+05	1.323150e+05	1.323150e+05
mean	2000.406492	8.222010e+02	4.144163e+03	2.156298e+03	1.987865e+03	6.286691e+02
std	8.264585	1.213648e+04	5.730412e+04	3.084739e+04	2.648095e+04	7.273950e+03
min	1991.000000	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	1991.000000	1.020000e+02	4.930000e+02	2.460000e+02	2.450000e+02	7.300000e+01
50%	2001.000000	1.930000e+02	9.460000e+02	4.780000e+02	4.680000e+02	1.440000e+02
75%	2011.000000	3.610000e+02	1.785000e+03	9.090000e+02	8.770000e+02	2.730000e+02
max	2011.000000	2.105604e+06	9.925891e+06	5.460145e+06	4.465746e+06	1.340673e+06

8 rows × 7 columns



```
In [65]: # Now want to see how many of Town/Villages have Total Population=0
zero_population_df = My_Data[My_Data['Total population'] == 0]

# Get the count of rows
num_zero_population_towns_villages = len(zero_population_df)

# Print the count
print("Number of Towns/Villages with Total Population = 0:", num_zero_popul
```

Number of Towns/Villages with Total Population = 0: 2616

```
In [66]: # We will delete these rows because our analysis will not focus there.
My_Data = My_Data[My_Data['Total population'] != 0]

# Reset the index after removing rows
My_Data.reset_index(drop=True, inplace=True)
```

We will create a new column named as Population_Group, which will separate the districts into villages, towns and cities according to their population in accordance with the division of the Indians

```
In [67]: # Define the bin breaks and labels
bin_breaks = [0, 15000, 100000, 100000000]
bin_labels = ["Village", "Town", "City"]

# Create a copy of the DataFrame
My_Data = My_Data.copy()

My_Data['Population_Group'] = pd.cut(My_Data['Total population'], bins=bin_
```

```
In [68]: My_Data = My_Data.copy()
My_Data['Part_time_workers'] = My_Data['Total population'] - (My_Data['Total
#Check the columns
My_Data.columns
```

```
Out[68]: Index(['Census Year', 'District', 'Taluka', 'Town/Village',
               'No. of households', 'Total population', 'Total male population',
               'Total female population', 'Total 0 to 6 year children',
               'Male 0 to 6 year children', 'Female 0 to 6 year children',
               'Total SC population', 'Male SC population', 'Female SC populatio
n',
               'Total ST population', 'Male ST population', 'Female ST populatio
n',
               'Total literates', 'Male literates', 'Female literates',
               'Total main workers', 'Male main workers', 'Female main workers',
               'Total non workers', 'Male non workers', 'Female non workers',
               'Total_illiterates', 'Male_illiterates', 'Female_illiterates',
               'Population_Group', 'Part_time_workers'],
              dtype='object')
```

```
In [69]: #Remove columns: Taluka and Town/Village because we will not use them anymore
My_Data.drop(['Taluka', 'Town/Village'], axis=1, inplace=True)
```

Now we have 29 columns and 129699 rows and we are ready to start our analysis process

In [70]: `My_Data.head()`

Out[70]:

	Census Year	District	No. of households	Total population	Total male population	Total female population	Total 0 to 6 year children	Male to yea childre
0	2011	AHMADNAGAR	201	732	359	373	73	3
1	2011	AHMADNAGAR	37	247	162	85	27	1
2	2011	AHMADNAGAR	357	1536	799	737	178	9
3	2011	AHMADNAGAR	3861	18278	9381	8897	2066	110
4	2011	AHMADNAGAR	529	2590	1352	1238	333	18

5 rows × 29 columns



In [71]: `# Remove any duplicate value`
`My_Data.drop_duplicates(inplace=True)`

Data analysis and visualization

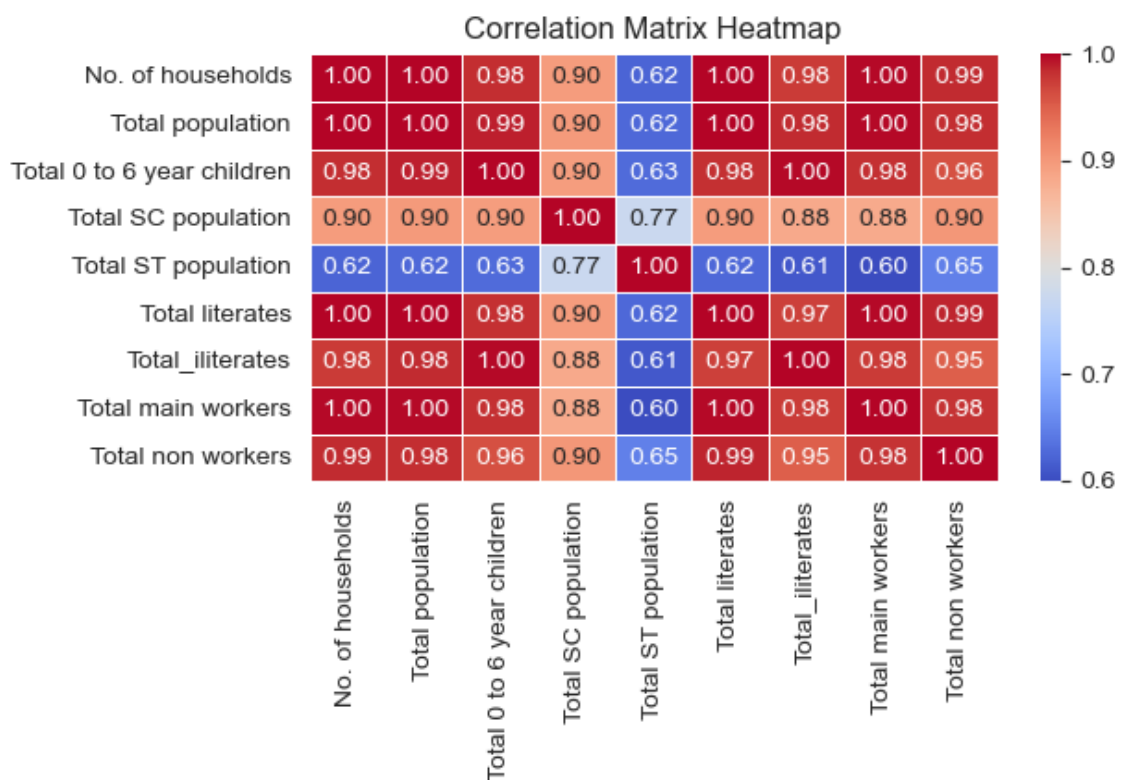
```

In [72]: # Starting with correlation matrix to understand how our data are related
# Select the above numerical columns
numerical_data = My_Data[[
    "No. of households",
    "Total population",
    "Total 0 to 6 year children",
    "Total SC population",
    "Total ST population",
    "Total literates",
    "Total_illiterates",
    "Total main workers",
    "Total non workers"
]]

# Calculate the correlation matrix
cor_matrix = numerical_data.corr()

# Plot the correlation matrix using a heatmap
plt.figure(figsize=(6, 3))
sns.heatmap(cor_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=
plt.title("Correlation Matrix Heatmap")
plt.show()

```




```
In [73]: # Group by district in descending total population
Pop_by_District=My_Data.groupby("District")["Total population"].sum()
Pop_by_District = Pop_by_District.sort_values(ascending=False)
Pop_by_District = Pop_by_District.reset_index()
Pop_by_District.columns = ["District", "Total Population"]

# Display the DataFrame
Pop_by_District.head(5)
```

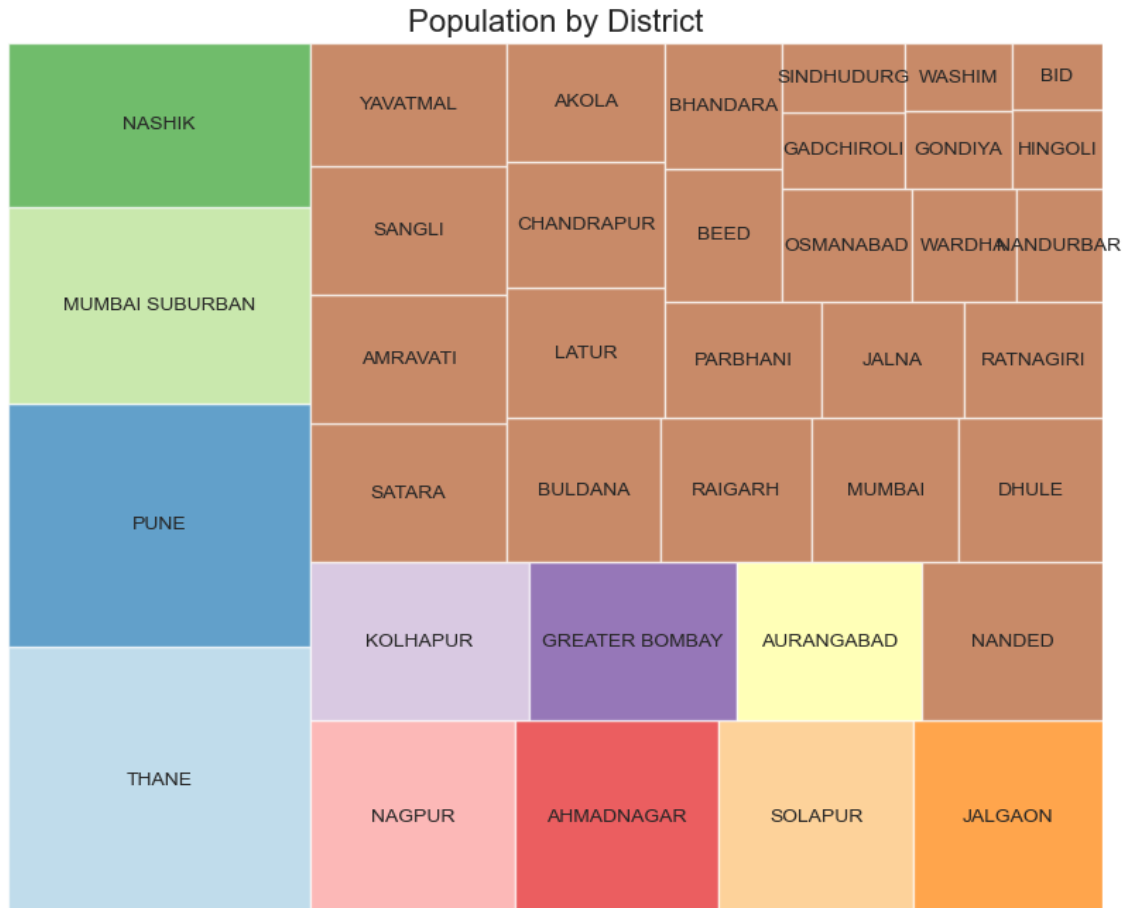
Out[73]:

	District	Total Population
0	THANE	24441123
1	PUNE	22194495
2	MUMBAI SUBURBAN	17997381
3	NASHIK	14952335
4	NAGPUR	12008346

```
In [74]: # import squarify
import squarify
districts = Pop_by_District['District']
population = Pop_by_District['Total Population']
# Treemap of all Districts in our Dataset
plt.figure(figsize=(10, 8))

squarify.plot(sizes=population, label=districts, alpha=0.7, color=plt.cm.Pa
plt.title('Population by District', fontsize=16)
plt.axis('off')

plt.show()
```



```
In [75]: # New dataframe of Census Year, Population_Group and Total population
grouping_data_by_CensusYear_PopGroup= My_Data.groupby(['Census Year', 'Popu
grouping_data_by_CensusYear_PopGroup
```

Out[75]:

	Census Year	Population_Group	Total population
0	1991	Village	48637372
1	1991	Town	7408775
2	1991	City	22890821
3	2001	Village	55028462
4	2001	Town	9901011
5	2001	City	31949146
6	2011	Village	61101342
7	2011	Town	12304055
8	2011	City	38968920

```
In [76]: # Adding a Percentage column of each Population group in each year
grouping_data_by_CensusYear_PopGroup['Percentage'] = (grouping_data_by_Cens
                                                         .transform(lambda x: (x
grouping_data_by_CensusYear_PopGroup['Percentage'] = grouping_data_by_Censu
grouping_data_by_CensusYear_PopGroup
```

Out[76]:

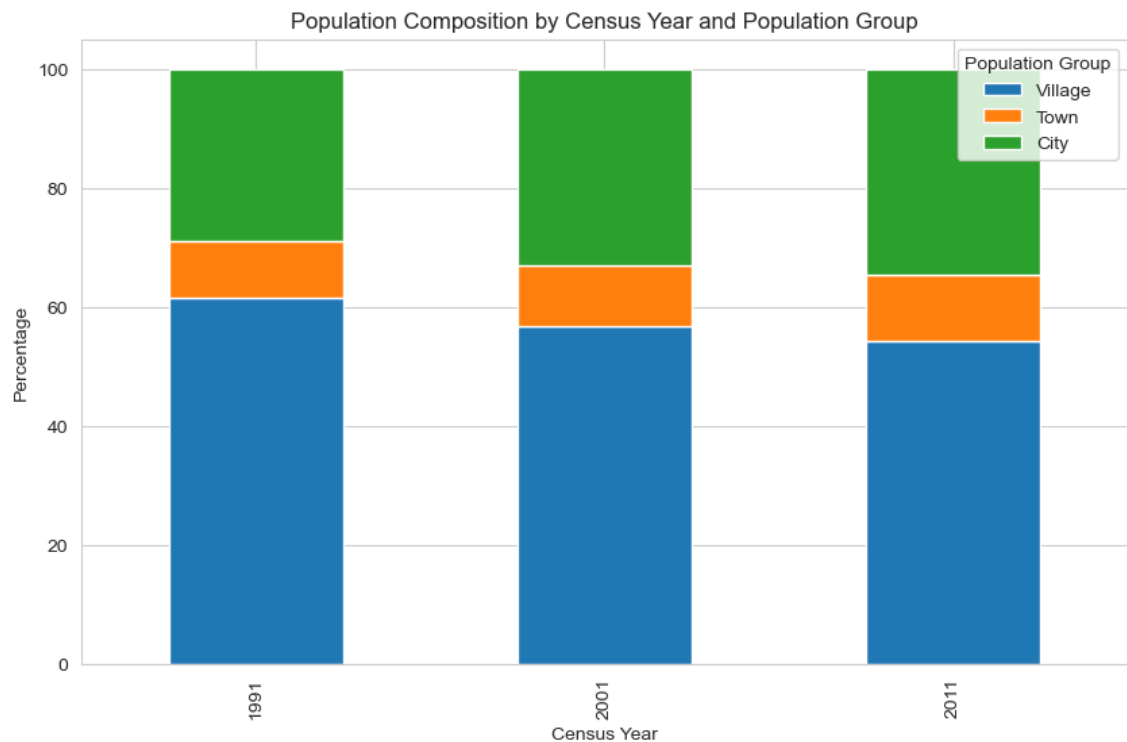
	Census Year	Population_Group	Total population	Percentage
0	1991	Village	48637372	61.62
1	1991	Town	7408775	9.39
2	1991	City	22890821	29.00
3	2001	Village	55028462	56.80
4	2001	Town	9901011	10.22
5	2001	City	31949146	32.98
6	2011	Village	61101342	54.37
7	2011	Town	12304055	10.95
8	2011	City	38968920	34.68

```
In [77]: # Stacked bar chart showing the composition of population by Census Year and an
grouped_data = grouping_data_by_CensusYear_PopGroup.groupby(['Census Year',

stacked = grouped_data.plot(kind='bar', stacked=True, figsize=(10, 6))

stacked.set_xlabel('Census Year')
stacked.set_ylabel('Percentage')
stacked.set_title('Population Composition by Census Year and Population Gro
stacked.legend(title='Population Group')

plt.show()
```



```
In [78]: # Creation of new Data Frame of number of Households in total population
percentage_data_Households_Total_Pop = My_Data.groupby(['Census Year', 'Pop
percentage_data_Households_Total_Pop = percentage_data_Households_Total_Pop
percentage_data_Households_Total_Pop.rename(columns={0: 'Percentage'}, inplace=True)
percentage_data_Households_Total_Pop['Percentage'] = percentage_data_Househ

# Heatmap showing the percentage of Number of Households in Total Population
plt.figure(figsize=(8, 4))
sns.set_style("whitegrid")
ax = sns.heatmap(data=percentage_data_Households_Total_Pop.pivot(index='Cen
ax.set_xlabel('Population Group')
ax.set_ylabel('Census Year')
ax.set_title('Percentage of No. of Households in Total Population by Census
plt.show()
```

Percentage of No. of Households in Total Population by Census Year and Population Group



```
In [79]: # Filtering the data to continue our analysis
filtered_data = My_Data[My_Data['Census Year'].isin([1991, 2001, 2011])]

# Calculation of the percentage of children aged 0-6 by Census Year
percentage_data1 = (filtered_data.groupby('Census Year')['Total 0 to 6 year
                    .sum() / filtered_data.groupby('Census Year')['Total pop
                    .sum() * 100]).reset_index()
```

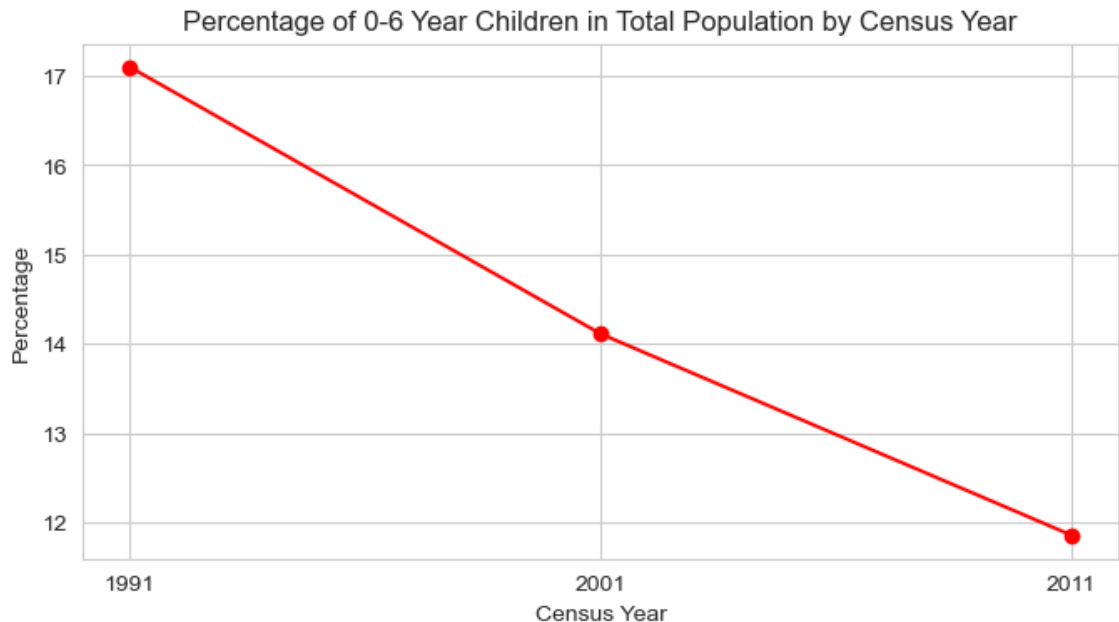
```
In [80]: # Making the data frame of percentage_data1 containing Census Year and Perc
percentage_data1.rename(columns={0: 'Percentage'}, inplace=True)
percentage_data1
```

Out[80]:

	Census Year	Percentage
0	1991	17.096115
1	2001	14.111603
2	2011	11.859042

In [81]: *# Line chart showing the percentage of 0-6 year children in Total Population*

```
plt.figure(figsize=(8, 4))
plt.plot(percentage_data1['Census Year'], percentage_data1['Percentage'], m
plt.xlabel('Census Year')
plt.ylabel('Percentage')
plt.title('Percentage of 0-6 Year Children in Total Population by Census Ye
plt.grid(True)
plt.xticks([1991, 2001, 2011])
plt.show()
```



In [82]: *# Calculation of the percentage of boys 0-6 years old per census year*

```
male_percentage_data = (My_Data.groupby('Census Year')['Male 0 to 6 year ch
                        .sum() / My_Data.groupby('Census Year')['Total 0 to
                        .sum() * 100).reset_index()
male_percentage_data.rename(columns={0: "Percentage"}, inplace=True)

# Calculation of the percentage of girls 0-6 years old per census year
female_percentage_data = (My_Data.groupby('Census Year')['Female 0 to 6 yea
                        .sum() / My_Data.groupby('Census Year')['Total 0
                        .sum() * 100).reset_index()
female_percentage_data.rename(columns={0: "Percentage"}, inplace=True)
```

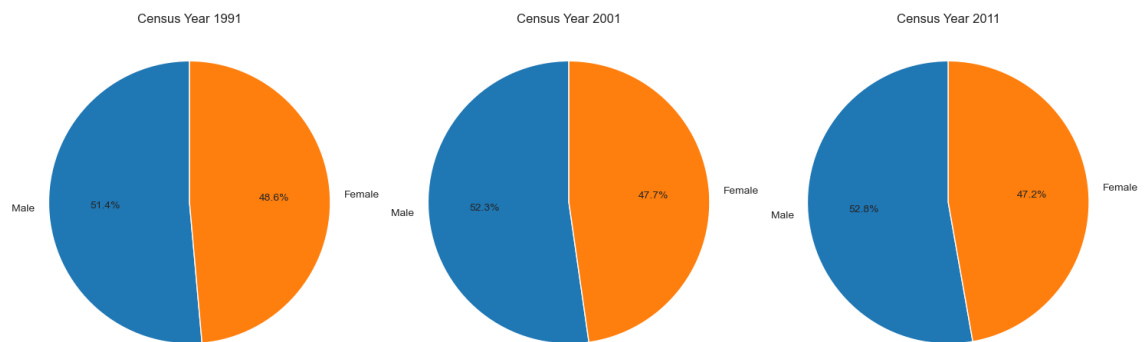
In [83]: *#Create the pie charts for each year*

```
fig, axs = plt.subplots(1, 3, figsize=(15, 5))

for i, year in enumerate([1991, 2001, 2011]):
    male_data = male_percentage_data[male_percentage_data['Census Year'] ==
    female_data = female_percentage_data[female_percentage_data['Census Yea

    ax = axs[i]
    ax.pie(
        [male_data['Percentage'].values[0], female_data['Percentage'].value
        labels=['Male', 'Female'],
        autopct='%1.1f%%',
        startangle=90,
    )
    ax.axis('equal')
    ax.set_title(f'Census Year {year}')

plt.tight_layout()
plt.show()
```



Now we are going to explore how literacy rate change by years

```

In [84]: # Create an empty List to store data for each year
data_by_year = []
#define the census years
census_years = [1991, 2001, 2011]
for year in census_years:
    data_year = My_Data[My_Data["Census Year"] == year]
    literacy_rate_year = data_year.groupby(["District"])[["Total literates"]]
    sorted_year = literacy_rate_year.sort_values(ascending=False)

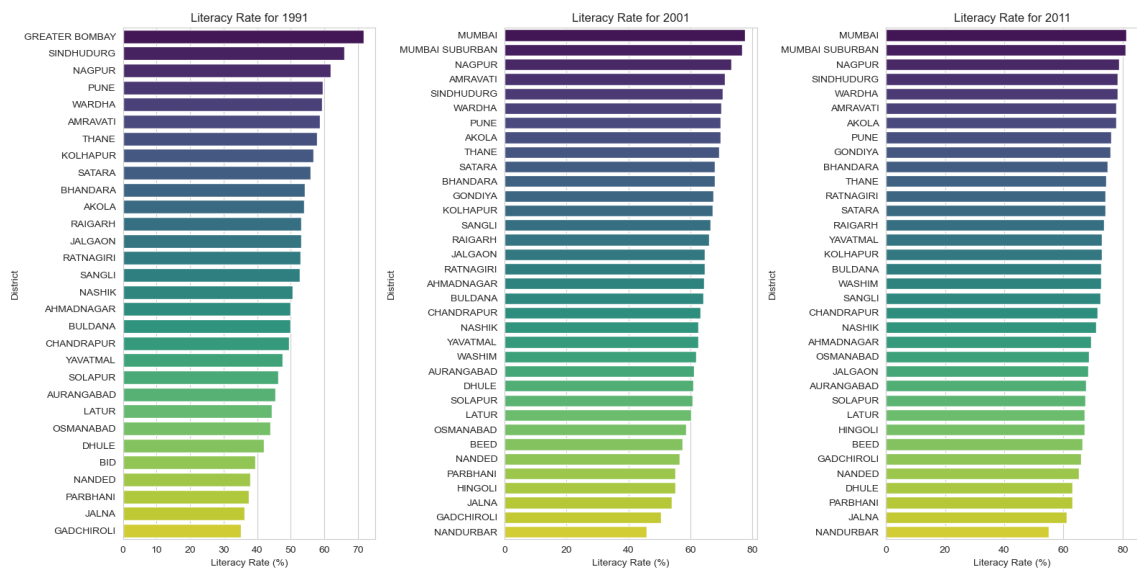
    data_by_year.append(sorted_year)

# Create horizontal bar charts for each year with sorted districts and color
plt.figure(figsize=(16, 8))
palette = "viridis"

for i, sorted_data in enumerate(data_by_year):
    plt.subplot(131 + i)
    sns.barplot(x=sorted_data, y=sorted_data.index, orient="h", palette=palette)
    plt.xlabel('Literacy Rate (%)')
    plt.ylabel('District')
    plt.title(f'Literacy Rate for {census_years[i]}')

plt.tight_layout()
plt.show()

```




```

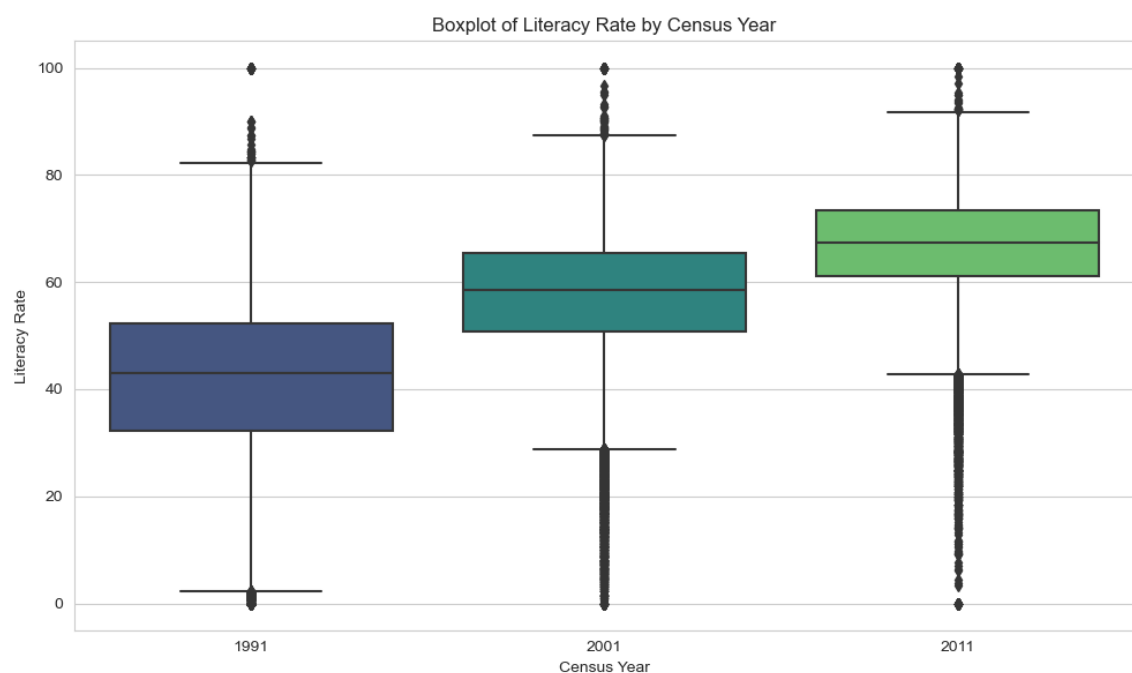
In [85]: # Calculation of Literacy Rate column
My_Data["Literacy Rate"] = (My_Data["Total literates"] / My_Data["Total pop

# Creation of a new Data frame with columns: District, Census Year, Literac
districts_census_years_df = My_Data[["District", "Census Year", "Literacy R

# Data frame display
districts_census_years_df

# Boxplot with axes :Literacy Rate and Census Year
plt.figure(figsize=(10, 6))
sns.boxplot(data=districts_census_years_df, x="Census Year", y="Literacy Ra
plt.xlabel("Census Year")
plt.ylabel("Literacy Rate")
plt.title("Boxplot of Literacy Rate by Census Year")
plt.tight_layout()
plt.show()

```



```

In [86]: # Group the data by "Population_Group" and "Census Year" and calculate the
grouped_data1 = My_Data.groupby(["Population_Group", "Census Year"])["Total

# Create a DataFrame with the grouped data
grouped_df = grouped_data1.reset_index()
grouped_df.columns = ["Population_Group", "Census Year", "Mean Literacy Rat

# Create a line chart
plt.figure(figsize=(10, 6))
sns.lineplot(data=grouped_df, x="Census Year", y="Mean Literacy Rate", hue=
plt.xlabel("Census Year")
plt.ylabel("Mean Literacy Rate (%)")
plt.title("Mean Literacy Rate by Population Group and Census Year")
plt.legend(title="Population Group")

# Add percentage labels to the line chart
for _, row in grouped_df.iterrows():
    plt.text(row["Census Year"], row["Mean Literacy Rate"], f"{row['Mean Li

# Set the x-axis ticks to only display 1991, 2001, and 2011
plt.xticks([1991, 2001, 2011])

plt.tight_layout()
plt.show()

```



```
In [87]: # Filter the data for the year 1991
data_1991 = My_Data[My_Data["Census Year"] == 1991]

# Calculate the percentage of male literates by district
male_lit_per_1991 = data_1991.groupby("District").apply(lambda x: (x["Male
male_lit_per_1991.columns = ["District", "Percentage_Male"]

# Calculate the percentage of female literates by district
female_lit_per_1991 = data_1991.groupby("District").apply(lambda x: (x["Fem
female_lit_per_1991.columns = ["District", "Percentage_Female"]

# Now we have two DataFrames: male_lit_per_1991 and female_lit_per_1991
# Merge the male and female DataFrames on the "District" column
merged_lit_per_1991 = male_lit_per_1991.merge(female_lit_per_1991, on="Dist

# Now we have a single DataFrame with both male and female percentages for
merged_lit_per_1991.head(3)
```

Out[87]:

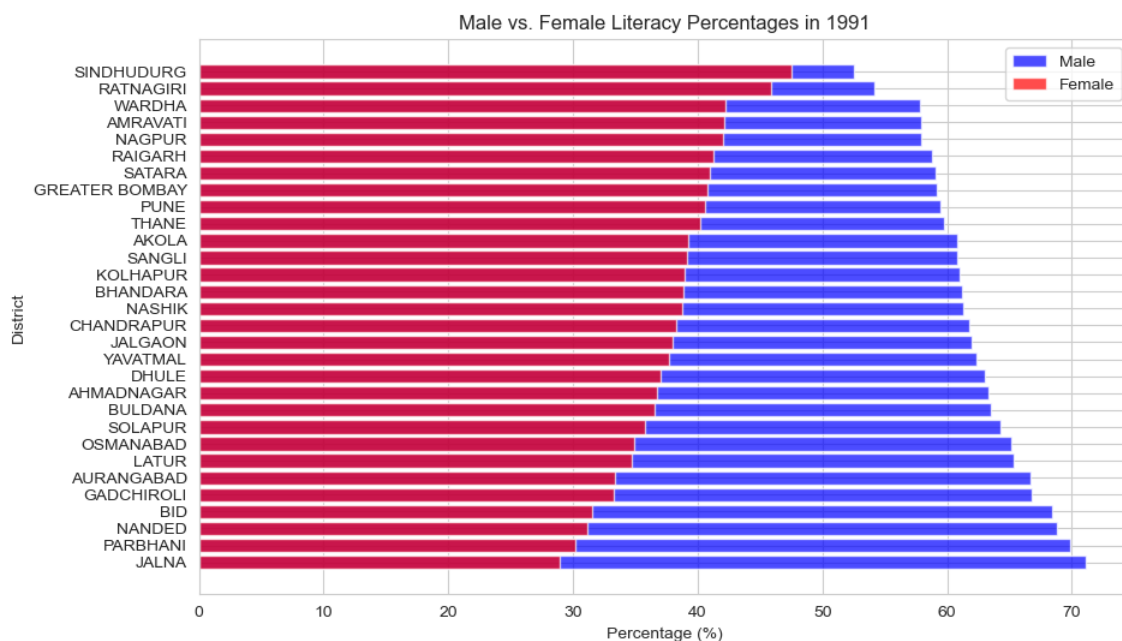
	District	Percentage_Male	Percentage_Female
0	AHMADNAGAR	63.306984	36.693016
1	AKOLA	60.792484	39.207516
2	AMRAVATI	57.879250	42.120750

```
In [88]: # Sort the DataFrame by Percentage_Male in ascending order
sorted_df = merged_lit_per_1991.sort_values(by="Percentage_Male")

# Create a horizontal bar plot
plt.figure(figsize=(10, 6))
plt.barh(sorted_df["District"], sorted_df["Percentage_Male"], label="Male",
plt.barh(sorted_df["District"], sorted_df["Percentage_Female"], label="Fema
plt.xlabel("Percentage (%)")
plt.ylabel("District")
plt.title("Male vs. Female Literacy Percentages in 1991")
plt.legend()

# Invert the y-axis to display the highest value at the top
plt.gca().invert_yaxis()

plt.show()
```



```
In [89]: # Filter the data for the year 1991
data_2001 = My_Data[My_Data["Census Year"] == 2001]

# Calculate the percentage of male literates by district
male_lit_per_2001 = data_2001.groupby("District").apply(lambda x: (x["Male
male_lit_per_2001.columns = ["District", "Percentage_Male"]

# Calculate the percentage of female literates by district
female_lit_per_2001 = data_2001.groupby("District").apply(lambda x: (x["Fem
female_lit_per_2001.columns = ["District", "Percentage_Female"]

# Now you have two DataFrames: male_lit_per_1991 and female_lit_per_1991
# Merge the male and female DataFrames on the "District" column
merged_lit_per_2001 = male_lit_per_2001.merge(female_lit_per_2001, on="Dist

# Now you have a single DataFrame with both male and female percentages for
merged_lit_per_2001.head(3)
```

Out[89]:

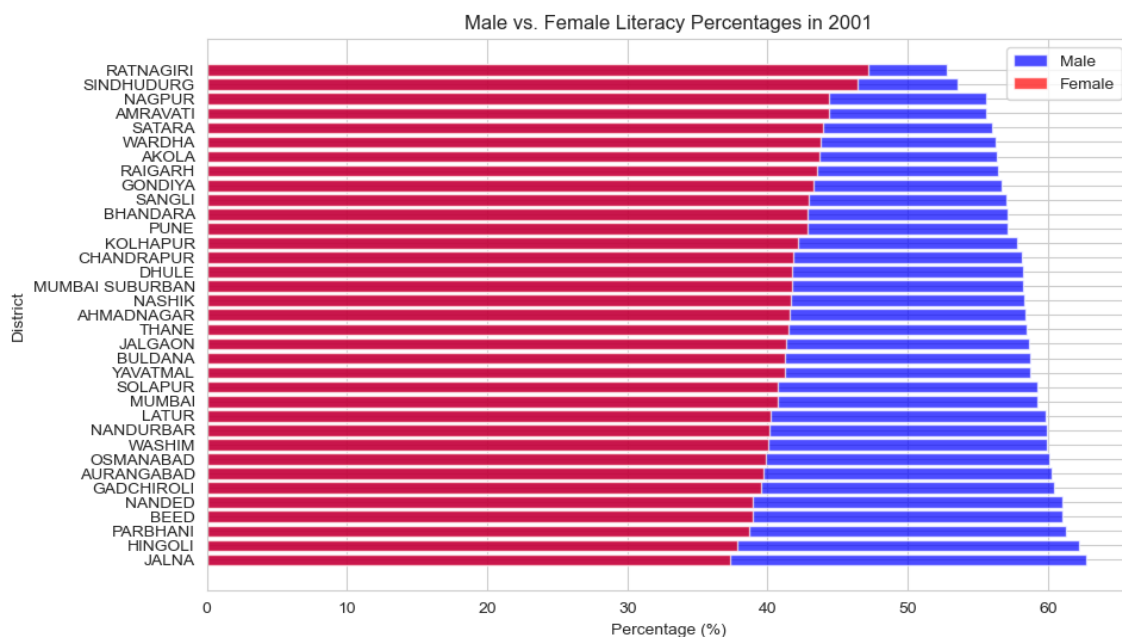
	District	Percentage_Male	Percentage_Female
0	AHMADNAGAR	58.378771	41.621229
1	AKOLA	56.329955	43.670045
2	AMRAVATI	55.616391	44.383609

```
In [90]: # Sort the DataFrame by Percentage_Male in ascending order
sorted_df = merged_lit_per_2001.sort_values(by="Percentage_Male")

# Create a horizontal bar plot
plt.figure(figsize=(10, 6))
plt.barh(sorted_df["District"], sorted_df["Percentage_Male"], label="Male",
plt.barh(sorted_df["District"], sorted_df["Percentage_Female"], label="Fema
plt.xlabel("Percentage (%)")
plt.ylabel("District")
plt.title("Male vs. Female Literacy Percentages in 2001")
plt.legend()

# Invert the y-axis to display the highest value at the top
plt.gca().invert_yaxis()

plt.show()
```



```
In [91]: # Filter the data for the year 1991
data_2011 = My_Data[My_Data["Census Year"] == 2011]

# Calculate the percentage of male literates by district
male_lit_per_2011 = data_2011.groupby("District").apply(lambda x: (x["Male
male_lit_per_2011.columns = ["District", "Percentage_Male"]

# Calculate the percentage of female literates by district
female_lit_per_2011 = data_2011.groupby("District").apply(lambda x: (x["Fem
female_lit_per_2011.columns = ["District", "Percentage_Female"]

# Now you have two DataFrames: male_lit_per_1991 and female_lit_per_1991
# Merge the male and female DataFrames on the "District" column
merged_lit_per_2011 = male_lit_per_2011.merge(female_lit_per_2011, on="Dist

# Now you have a single DataFrame with both male and female percentages for
merged_lit_per_2011.head(3)
```

Out[91]:

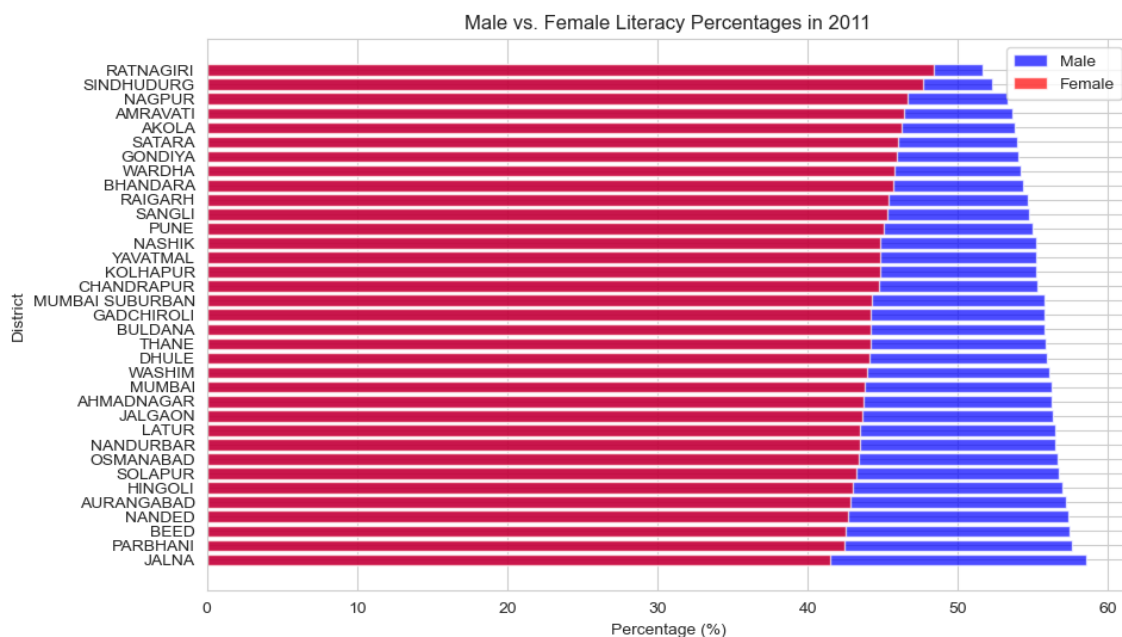
	District	Percentage_Male	Percentage_Female
0	AHMADNAGAR	56.263226	43.736774
1	AKOLA	53.779297	46.220703
2	AMRAVATI	53.604335	46.395665

```
In [92]: # Sort the DataFrame by Percentage_Male in ascending order
sorted_df = merged_lit_per_2011.sort_values(by="Percentage_Male")

# Create a horizontal bar plot
plt.figure(figsize=(10, 6))
plt.barh(sorted_df["District"], sorted_df["Percentage_Male"], label="Male",
plt.barh(sorted_df["District"], sorted_df["Percentage_Female"], label="Fema
plt.xlabel("Percentage (%)")
plt.ylabel("District")
plt.title("Male vs. Female Literacy Percentages in 2011")
plt.legend()

# Invert the y-axis to display the highest value at the top
plt.gca().invert_yaxis()

plt.show()
```




```

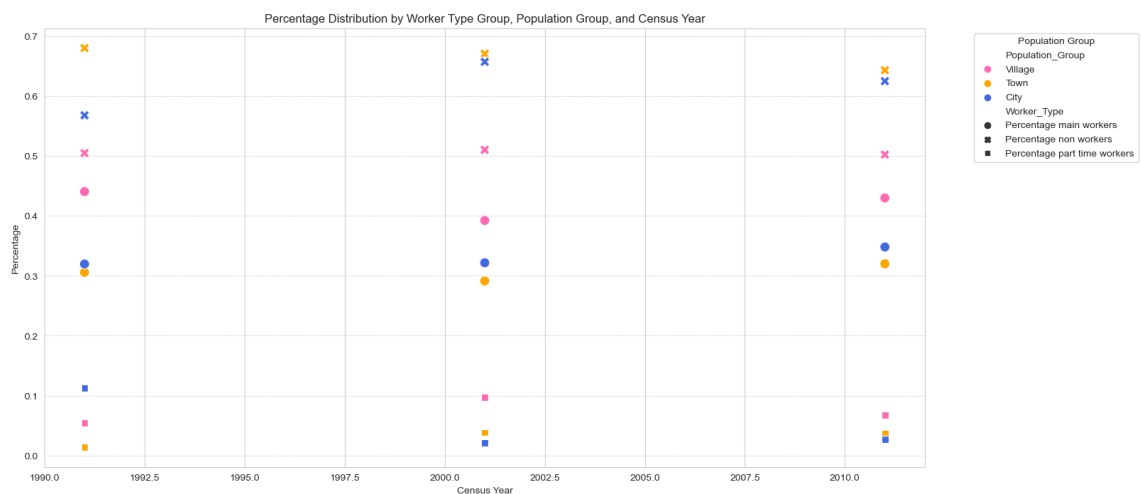
In [100]: # Group the data by Census Year and Population_Group and aggregate them in
grouped_data2 = My_Data.groupby(['Census Year', 'Population_Group']).agg({
    'Total main workers': 'sum',
    'Total non workers': 'sum',
    'Part_time_workers': 'sum',
    'Total population': 'sum'
}).reset_index()

# Calculate percentages
grouped_data2['Percentage main workers'] = grouped_data2['Total main worker
grouped_data2['Percentage non workers'] = grouped_data2['Total non workers'
grouped_data2['Percentage part time workers'] = grouped_data2['Part_time_wo

# Reshape the data for plotting
grouped_data2_long = pd.melt(grouped_data2, id_vars=['Census Year', 'Popula
    value_vars=['Percentage main workers', 'Percen
    var_name='Worker_Type', value_name='Percentage

# Create the plot using seaborn
plt.figure(figsize=(16, 8))
sns.scatterplot(data=grouped_data2_long, x='Census Year', y='Percentage', h
    palette={"City": "#4169E1", "Town": "#FFA500", "Village": "#
plt.title("Percentage Distribution by Worker Type Group, Population Group,
plt.xlabel("Census Year")
plt.ylabel("Percentage")
plt.legend(title="Population Group", bbox_to_anchor=(1.05, 1), loc='upper r
plt.grid(True, axis='y', linestyle='--', linewidth=0.5)
plt.show()

```



```

In [95]: # Calculate male non-workers' percentages by Census Year
male_non_workers = My_Data.groupby('Census Year').apply(
    lambda x: (x['Male non workers'].sum() / x['Total male population']).sum()
).reset_index(name='Male_Percentage')

# Calculate female non-workers' percentages by Census Year
female_non_workers = My_Data.groupby('Census Year').apply(
    lambda x: (x['Female non workers'].sum() / x['Total female population']).sum()
).reset_index(name='Female_Percentage')

# Combine male_non_workers and female_non_workers DataFrames
combined_data = pd.merge(male_non_workers, female_non_workers, on='Census Y

# Create the bar plot
plt.figure(figsize=(10, 6))
colors = ["#4169E1", "#FF69B4"]

bar_width = 0.8
index = combined_data['Census Year']

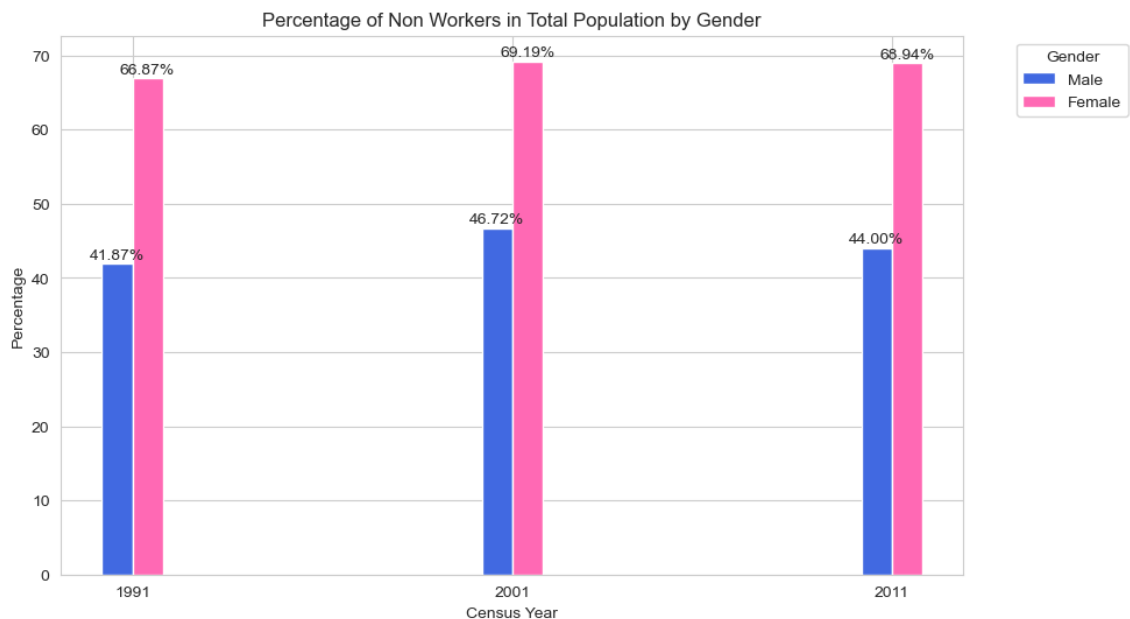
bars1 = plt.bar(index, combined_data['Male_Percentage'],
                bar_width, label='Male', color=colors[0])
bars2 = plt.bar(index + bar_width, combined_data['Female_Percentage'],
                bar_width, label='Female', color=colors[1])

plt.title("Percentage of Non Workers in Total Population by Gender")
plt.xlabel("Census Year")
plt.ylabel("Percentage")
plt.xticks(index + bar_width / 2, combined_data['Census Year'])
plt.legend(title="Gender", bbox_to_anchor=(1.05, 1), loc="upper left")

# Add percentage labels on top of each bar
for bar in bars1 + bars2:
    height = bar.get_height()
    plt.annotate(f'{height:.2f}%', xy=(bar.get_x() + bar.get_width() / 2, h
    xytext=(0, 3), textcoords='offset points', ha='center')

plt.show()

```



```
In [96]: My_Data['Sum_ST_SC_Population'] = My_Data['Total ST population'] + My_Data[
My_Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 123678 entries, 0 to 129698
Data columns (total 31 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Census Year                          123678 non-null int64
1   District                             123678 non-null object
2   No. of households                    123678 non-null int64
3   Total population                     123678 non-null int64
4   Total male population                123678 non-null int64
5   Total female population              123678 non-null int64
6   Total 0 to 6 year children           123678 non-null int64
7   Male 0 to 6 year children            123678 non-null int64
8   Female 0 to 6 year children          123678 non-null int64
9   Total SC population                  123678 non-null int64
10  Male SC population                   123678 non-null int64
11  Female SC population                 123678 non-null int64
12  Total ST population                  123678 non-null int64
13  Male ST population                   123678 non-null int64
14  Female ST population                 123678 non-null int64
15  Total literates                      123678 non-null int64
16  Male literates                      123678 non-null int64
17  Female literates                     123678 non-null int64
18  Total main workers                   123678 non-null int64
19  Male main workers                    123678 non-null int64
20  Female main workers                  123678 non-null int64
21  Total non workers                    123678 non-null int64
22  Male non workers                     123678 non-null int64
23  Female non workers                   123678 non-null int64
24  Total_illiterates                    123678 non-null int64
25  Male_illiterates                     123678 non-null int64
26  Female_illiterates                   123678 non-null int64
27  Population_Group                     123678 non-null category
28  Part_time_workers                    123678 non-null int64
29  Literacy Rate                        123678 non-null float64
30  Sum_ST_SC_Population                 123678 non-null int64
dtypes: category(1), float64(1), int64(28), object(1)
memory usage: 29.4+ MB
```

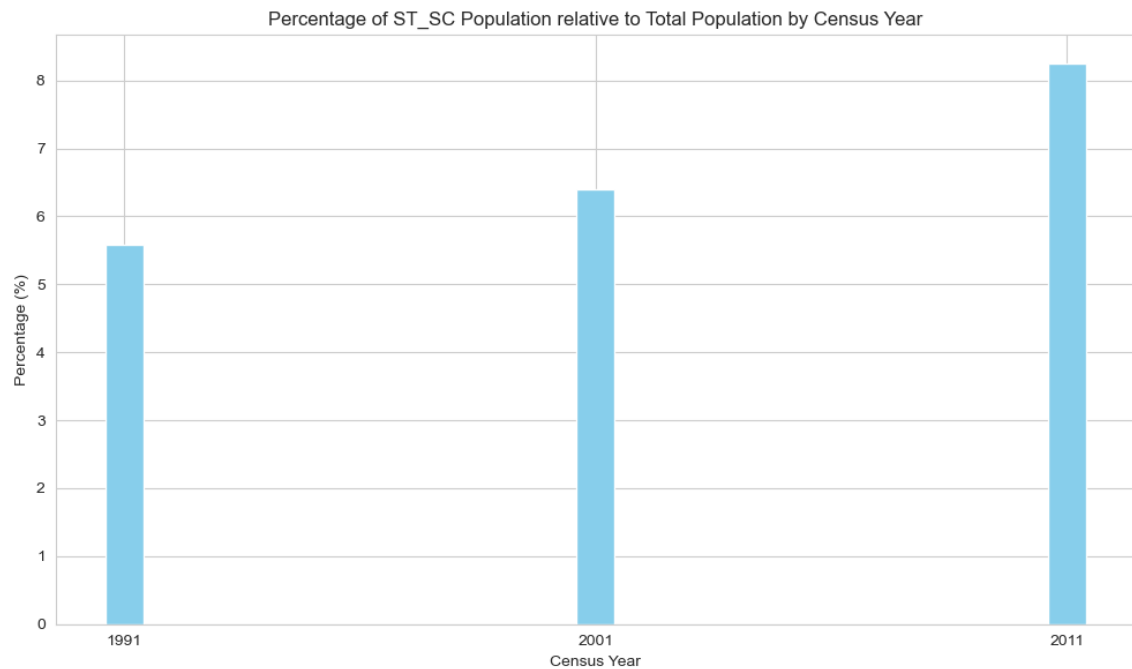
```
In [97]: # Percentage of ST and SC population in Total Population
Perc_Sum_ST_SC_Population = (My_Data.groupby(["Census Year"])[ 'Sum_ST_SC_Po
/ My_Data['Total population'].sum()) * 100

# Create a DataFrame with the result
ST_SC_in_totalpop_per_year= pd.DataFrame({'Census Year': Perc_Sum_ST_SC_Pop
'Perc_ST_SC': Perc_Sum_ST_SC_Popu
ST_SC_in_totalpop_per_year
```

Out[97]:

	Census Year	Perc_ST_SC
0	1991	5.578241
1	2001	6.405128
2	2011	8.253623

```
In [98]: #Bar plot of ST and SC pop in total population by year
plt.figure(figsize=(10, 6))
plt.bar(ST_SC_in_totalpop_per_year['Census Year'], ST_SC_in_totalpop_per_ye
plt.xlabel('Census Year')
plt.ylabel('Percentage (%)')
plt.title('Percentage of ST_SC Population relative to Total Population by C
plt.xticks(ST_SC_in_totalpop_per_year['Census Year'])
plt.tight_layout()
plt.show()
```



```
In [99]: # Creation of a new column: Percentage_ST_SC_in_TotalPop
My_Data["Percentage_ST_SC_in_TotalPop"] = (My_Data["Sum_ST_SC_Population"]

# Grouping by District and Population_Group
grouped_data3 = My_Data.groupby(["District", "Population_Group"])["Percenta

#
top_10_districts = grouped_data3.nlargest(10, "Percentage_ST_SC_in_TotalPop

# We want to see the top 10 villages with the highest percentage ratio of S
top_10_districts
```

Out[99]:

	District	Population_Group	Percentage_ST_SC_in_TotalPop
66	NANDURBAR	Village	86.505187
30	GADCHIROLI	Village	72.355157
27	DHULE	Village	60.232107
69	NASHIK	Village	56.097628
99	THANE	Village	54.010450
6	AMRAVATI	Village	48.459964
24	CHANDRAPUR	Village	44.453276
60	NAGPUR	Village	40.066494
108	YAVATMAL	Village	39.817312
33	GONDIYA	Village	37.021420

Conclusions

1. Introduction:

In our initial exploration of the population dataset, we discovered that the total Scheduled Tribes (ST) population showed relatively lower correlations compared to other columns, hinting at unique characteristics. We observed changing residential patterns over the decades. In 1991, villages were home to 61.62% of the population, but by 2011, this had fallen to 37%. Meanwhile, cities saw an increase from 29% in 1991 to 34.68% in 2011.

2. Children's Population:

A concerning trend emerged in the proportion of children aged 0-6 within the total population. In 1991, they made up 17.1%, but this percentage steadily declined to 11.9% in 2011, reflecting challenges faced by disadvantaged populations

3. Gender Disparity Among Children:

Our findings depicted a growing gender disparity among children. In 1991, female children accounted for 48.6%, but by 2011, this number had dropped to 47.2%, possibly influenced by cultural preferences.

4. Literacy Rates:

In the realm of literacy rates, Greater Bombay consistently stood out with almost double the percentage of literates compared to other districts. Thane and Mumbai Suburban traded positions between 2001 and 2011, highlighting changes in literacy dynamics.

5. Employment Patterns:

Over the years, there was a notable shift in employment patterns. In 1991, cities saw 11.3% part-time workers, 32% full-time workers, and 56.8% non-workers. By 2001, the numbers transformed to 2.1%, 32.2%, and 65.7%, respectively, indicating evolving work patterns. Villages, on the other hand, consistently had a higher percentage of main workers across all population groups.

6. Educational Dynamics in Villages: Migration's Impact on Literacy Rates

The most remarkable surge in literacy rates occurred within the Population Group of Villages. Starting at 45.61% in the initial period, the proportion of literates in villages significantly rose to 59.71% and further soared to 67.46%. Consequently, we can infer that those who moved were predominantly illiterate individuals seeking better opportunities in larger cities, while the literate population largely remained behind. This dynamic sheds light on the shifting educational landscape and the impact of migration on literacy rates.

7. Women's Employment Status:

Despite increasing literacy rates among women, the majority remained outside the workforce. In 1991, 66.87% of women were non-workers, and this percentage remained high in subsequent years. Conversely, men experienced fluctuations in their non-worker percentages, eventually settling at 44%.

8. Social Vulnerability:

The populations of Scheduled Tribes (ST) and Scheduled Castes (SC) increased from 5.57% to 8.25% between 1991 and 2011. Interestingly, the first ten areas inhabited by ST and SC populations are all villages.