Start coding or generate with AI.

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


# California Dataset
from google.colab import files
uploaded = files.upload()

housing = pd.read_excel("housing+(1).xlsx")
housing
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving housing+(1).xlsx to housing+(1).xlsx

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value
0	-122.23	37.88	41	880	129.0	322	126	8.3252	452600
1	-122.22	37.86	21	7099	1106.0	2401	1138	8.3014	358500
2	-122.24	37.85	52	1467	190.0	496	177	7.2574	352100
3	-122.25	37.85	52	1274	235.0	558	219	5.6431	341300
4	-122.25	37.85	52	1627	280.0	565	259	3.8462	342200
...	...	...	...	...	...	...	...	...	...
20635	-121.09	39.48	25	1665	374.0	845	330	1.5603	78100
20636	-121.21	39.49	18	697	150.0	356	114	2.5568	77100
20637	-121.22	39.43	17	2254	485.0	1007	433	1.7000	92300
20638	-121.32	39.43	18	1860	409.0	741	349	1.8672	84700
20639	-121.24	39.37	16	2785	616.0	1387	530	2.3886	89400

```
housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  int64
 3   total_rooms         20640 non-null  int64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  int64
 6   households          20640 non-null  int64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  int64
 9   ocean_proximity     20640 non-null  object
dtypes: float64(4), int64(5), object(1)
memory usage: 1.6+ MB
```

Start coding or generate with AI.

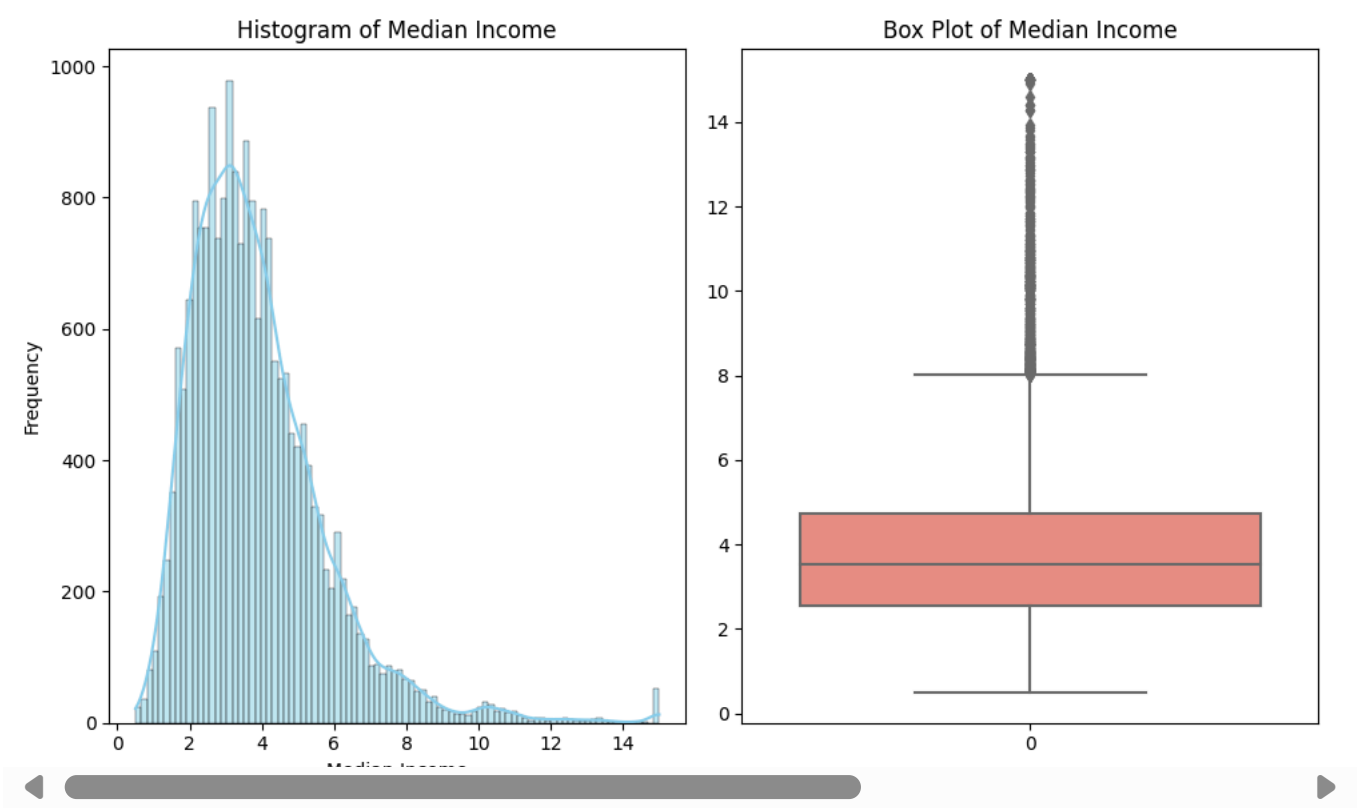
1.**What is the average median income of the data set and check the distribution of data using appropriate plots. Please explain the distribution of the plot.**

```
avg_median_income = np.mean(housing.median_income)
print(avg_median_income.round(2))
```

```
3.87
```

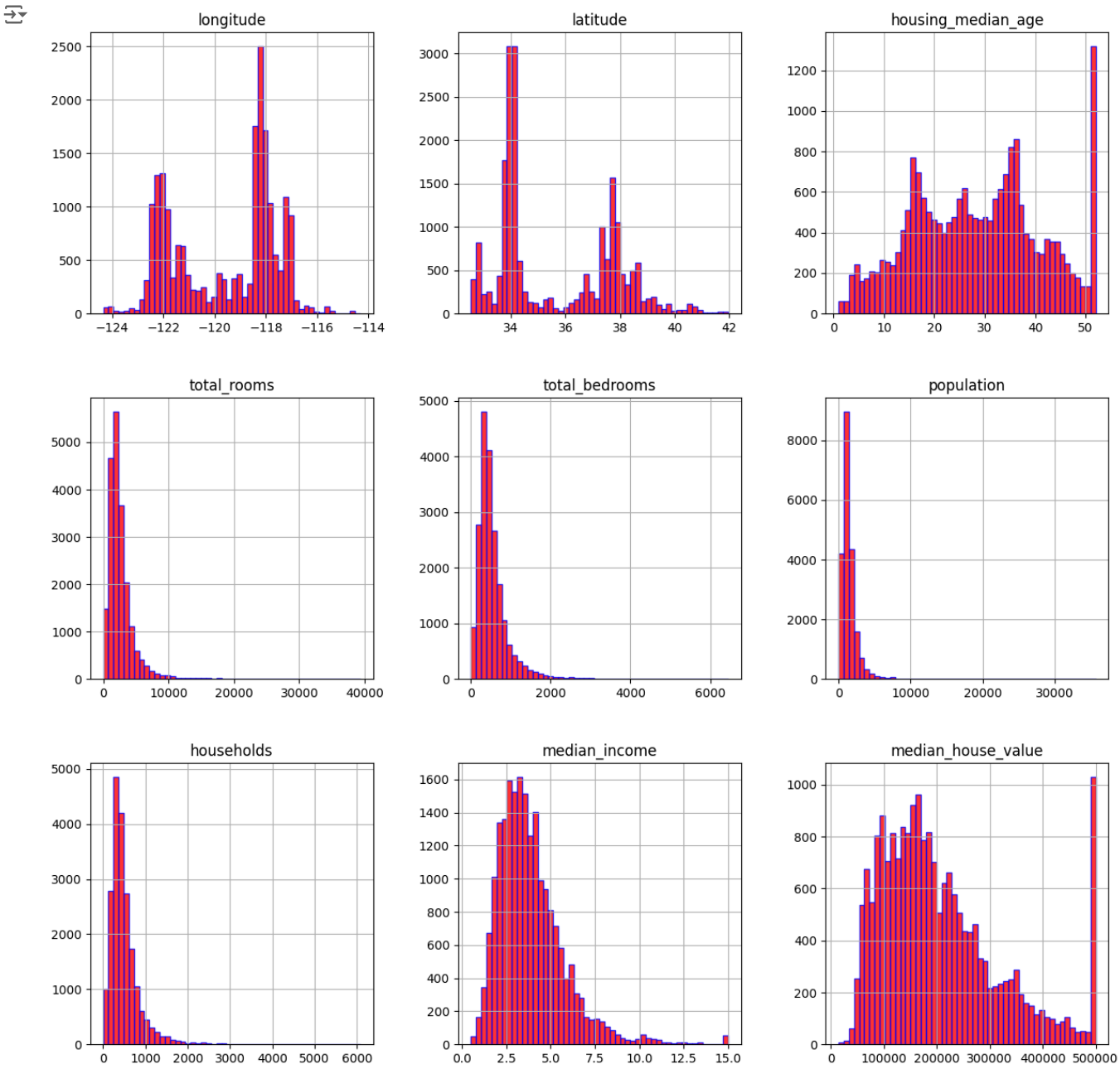
```
#Create distribution plots
plt.figure(figsize=(10, 6))

# Plot a histogram to visualize the distribution
plt.subplot(1,2,1)
sns.histplot(housing.median_income, kde=True, color='skyblue')
plt.title('Histogram of Median Income')
plt.xlabel('Median Income')
```

```
plt.ylabel('Frequency')

# Plot a box plot to visualize central tendency and spread
plt.subplot(1, 2, 2)
sns.boxplot(data=housing.median_income, color='salmon')
plt.title('Box Plot of Median Income')

plt.tight_layout()
```



Start coding or generate with AI.

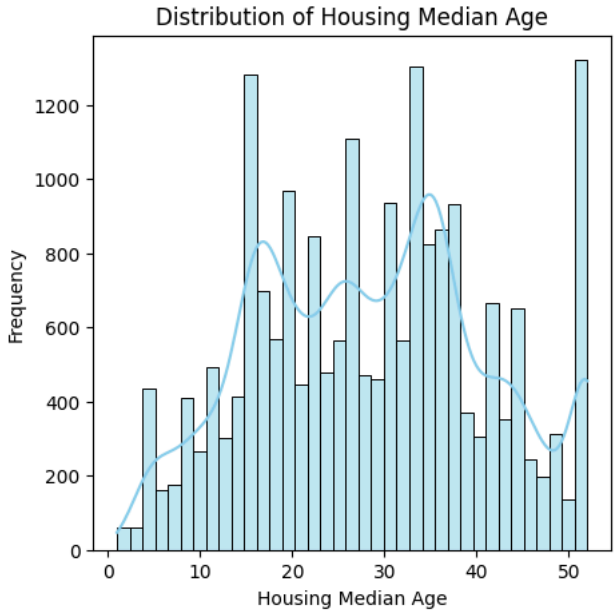
```
housing.hist(bins=50, figsize=(15,15),facecolor='r', edgecolor='blue',alpha=0.8)
plt.show()
```

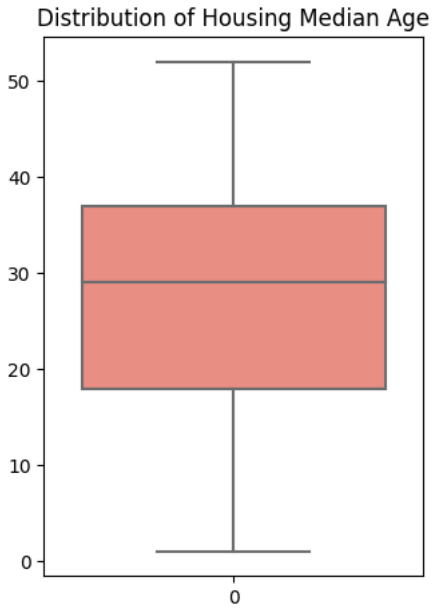


2.**Draw an appropriate plot to see the distribution of housing_median_age and explain your observations.**

```
# Create a histogram to visualize the distribution

plt.figure(figsize=(5, 5))
sns.histplot(housing.housing_median_age, kde=True, color='skyblue')
plt.title('Distribution of Housing Median Age')
plt.xlabel('Housing Median Age')
plt.ylabel('Frequency')
plt.show()

# Create a box plot for distribution
plt.subplot(1,2,2)
sns.boxplot(data=housing.housing_median_age, color="salmon")
plt.title('Distribution of Housing Median Age')
```

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


# Calculate basic statistics
mean_age = np.mean(housing.housing_median_age)
median_age = np.median(housing.housing_median_age)
std_dev_age = np.std(housing.housing_median_age)

print(f"Mean Housing Median Age: {mean_age:.2f}")
print(f"Median Housing Median Age: {median_age:.2f}")
print(f"Standard Deviation: {std_dev_age:.2f}")
```

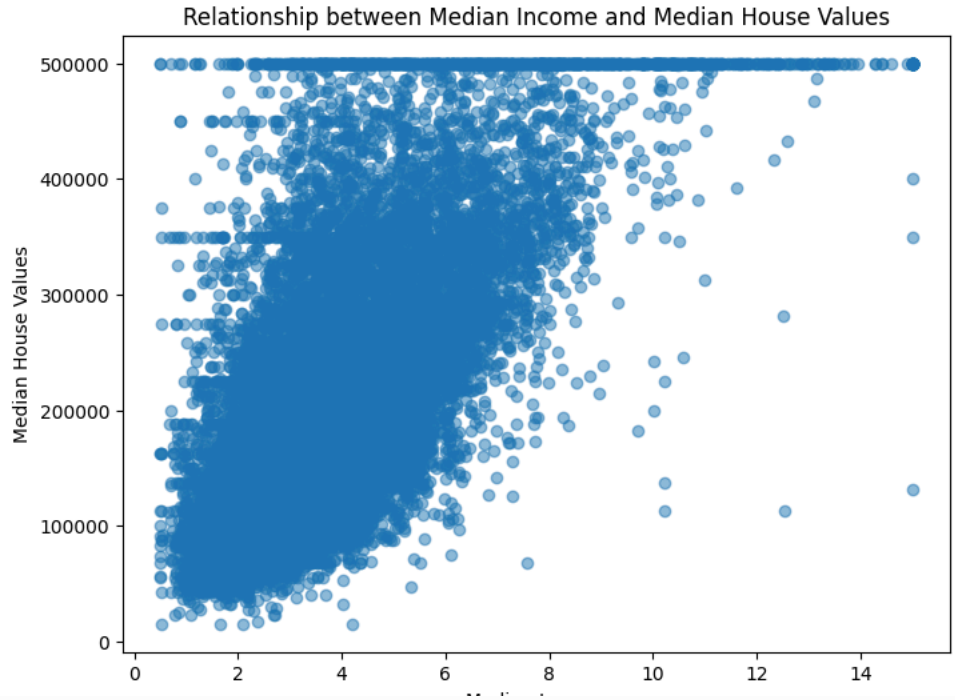





```
Mean Housing Median Age: 28.64
Median Housing Median Age: 29.00
```

3.**Show with the help of visualization, how median_income and median_house_values are related?**

```
# Create a scatter plot to visualize the relationship
x= housing['median_income']
y= housing['median_house_value']
plt.figure(figsize=(8, 6))
plt.scatter(x, y, alpha=0.5)
plt.title('Relationship between Median Income and Median House Values')
plt.xlabel('Median Income')
plt.ylabel('Median House Values')
plt.show()
```




```
correlation = np.corrcoef(housing.median_income, housing.median_house_value)[0, 1]
print(f"Correlation Coefficient: {correlation:.2f}")


#It is a Positive correlation though not strong
```

```
Correlation Coefficient: 0.69
```

Start coding or generate with AI.

4.Create a data set by deleting the corresponding examples from the data set for which total_bedrooms are not available.

```
# Create a new DataFrame with examples where 'total_bedrooms' are available
filtered_data = housing.dropna(subset=['total_bedrooms'])

# If you want to reset the index in the new DataFrame
filtered_data.reset_index(drop=True, inplace=True)

# Now, 'filtered_data' contains only the examples with available 'total_bedrooms'
# You can use it for further analysis

# To verify the resulting DataFrame
print(filtered_data.head())
```

```
   longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
0    -122.23     37.88                  41        880           129.0
1    -122.22     37.86                  21       7099          1106.0
2    -122.24     37.85                  52       1467           190.0
3    -122.25     37.85                  52       1274           235.0
4    -122.25     37.85                  52       1627           280.0

   population  households  median_income  median_house_value ocean_proximity
0         322         126         8.3252              452600        NEAR BAY
1        2401        1138         8.3014              358500        NEAR BAY
2         496         177         7.2574              352100        NEAR BAY
3         558         219         5.6431              341300        NEAR BAY
4         565         259         3.8462              342200        NEAR BAY
```

5.Create a data set by filling the missing data with the mean value of the total_bedrooms in the original data set.

```
# Calculate the mean of 'total_bedrooms' excluding missing values
mean_total_bedrooms = housing['total_bedrooms'].mean()

# Create a new DataFrame by filling missing 'total_bedrooms' with the mean
filled_data = housing.fillna({'total_bedrooms': mean_total_bedrooms})

# Now, 'filled_data' contains the original data with missing 'total_bedrooms' filled by the mean value
```

```
# To verify the resulting DataFrame
print(filled_data.head())
```

```
   longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
0    -122.23     37.88                  41          880           129.0
1    -122.22     37.86                  21         7099          1106.0
2    -122.24     37.85                  52         1467           190.0
3    -122.25     37.85                  52         1274           235.0
4    -122.25     37.85                  52         1627           280.0

   population  households  median_income  median_house_value ocean_proximity
0         322         126         8.3252              452600        NEAR BAY
1        2401        1138         8.3014              358500        NEAR BAY
2         496         177         7.2574              352100        NEAR BAY
3         558         219         5.6431              341300        NEAR BAY
4         565         259         3.8462              342200        NEAR BAY
```

Start coding or generate with AI.

6.Write a programming construct (create a user defined function) to calculate the median value of the data set wherever required.

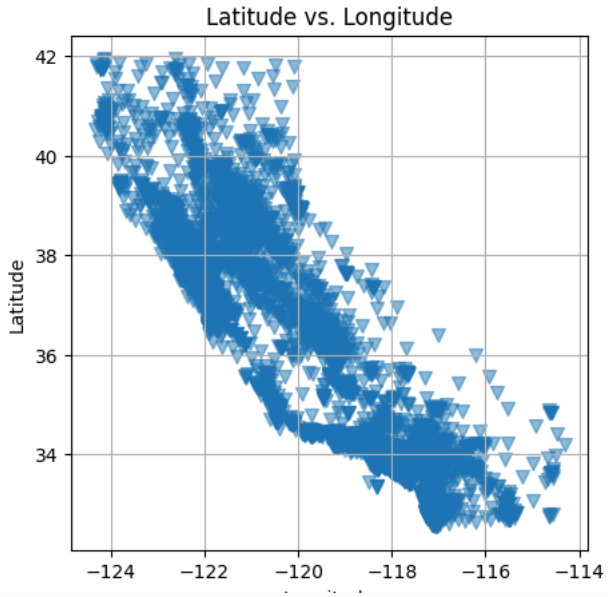
```
def calculate_median(data):
    # Sort the data in ascending order
    sorted_data = sorted(data)

    # Find the length of the data set
    n = len(sorted_data)

    # Check if the data set has an odd or even number of elements
    if n % 2 == 1:
        # For an odd number of elements, return the middle value
        median = sorted_data[n // 2]
    else:
        # For an even number of elements, return the average of the two middle values
        middle1 = sorted_data[(n // 2) - 1]
        middle2 = sorted_data[n // 2]
        median = (middle1 + middle2) / 2.0

    return median

# for Example taking total bedrooms column to calculate median
data = housing['total_bedrooms']
median_value = calculate_median(data)
print(f"The median value of the data is: {median_value}")
```

```
The median value of the data is: 429.0
```

Start coding or generate with AI.

7.Plot latitude versus longitude and explain your observations.

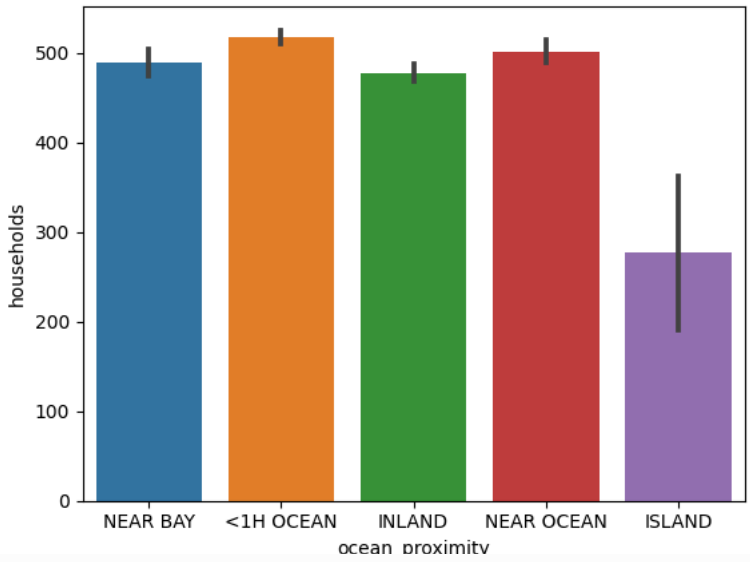
```
latitude = housing['latitude']
longitude = housing['longitude']

# Create a scatter plot of latitude versus longitude
plt.figure(figsize=(5, 5))
plt.scatter(longitude, latitude, alpha=0.5, s=50, marker='v')  # 's' parameter adjusts point size

plt.title('Latitude vs. Longitude')
plt.xlabel('Longitude')
plt.ylabel('Latitude')

plt.grid(True)
plt.show()
```



8.Create a data set for which the ocean_proximity is 'Near ocean'.

```
# Filter the dataset to include only rows where ocean_proximity is 'Near ocean'
near_ocean_data = housing[housing['ocean_proximity'] == 'NEAR BAY']

# To verify the resulting DataFrame
print(near_ocean_data.head())

#Graph

sns.barplot(x=housing['ocean_proximity'], y=housing['households'], data=housing)
```

```
   longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
0    -122.23     37.88                  41          880           129.0
1    -122.22     37.86                  21         7099          1106.0
2    -122.24     37.85                  52         1467           190.0
3    -122.25     37.85                  52         1274           235.0
4    -122.25     37.85                  52         1627           280.0

   population  households  median_income  median_house_value ocean_proximity
0         322         126         8.3252              452600        NEAR BAY
1        2401        1138         8.3014              358500        NEAR BAY
2         496         177         7.2574              352100        NEAR BAY
3         558         219         5.6431              341300        NEAR BAY
4         565         259         3.8462              342200        NEAR BAY
<Axes: xlabel='ocean_proximity', ylabel='households'>
```



9.**Find the mean and median of the median income for the data set created in question 8.**

```
# Calculate the mean and median of 'median_income'
mean_median_income = near_ocean_data['median_income'].mean()
```

```
median_median_income = near_ocean_data['median_income'].median()

print(f"Mean Median Income for 'Near Ocean': {mean_median_income:.2f}")
print(f"Median Median Income for 'Near Ocean': {median_median_income:.2f}")
```

```
Mean Median Income for 'Near Ocean': 4.17
Median Median Income for 'Near Ocean': 3.82
```

Start coding or generate with AI.

10.Please create a new column named total_bedroom_size. If the total bedrooms is 10 or less, it should be quoted as small. If the total bedrooms is 11 or more but less than 1000, it should be medium, otherwise it should be considered large.

```
# Define the conditions and labels for the new column
conditions = [
    housing['total_bedrooms'] <= 10,
    (housing['total_bedrooms'] > 10) & (housing['total_bedrooms'] < 1000),
    housing['total_bedrooms'] >= 1000
]

labels = ['Small', 'Medium', 'Large']

# Create the 'total_bedroom_size' column
housing['total_bedroom_size'] = pd.cut(housing['total_bedrooms'],
                                       bins=[0, 10, 1000, float('inf')],
                                       labels=labels, include_lowest=True)

# Now, 'total_bedroom_size' column has values 'Small', 'Medium', or 'Large' based on the conditions

# To verify the resulting DataFrame
print(housing.head())

# Distribution of total_bedrooms by size
sns.catplot(x='total_bedroom_size', y='total_bedrooms', data=housing)
```

```
   longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
0    -122.23     37.88                  41          880           129.0
1    -122.22     37.86                  21         7099          1106.0
2    -122.24     37.85                  52         1467           190.0
3    -122.25     37.85                  52         1274           235.0
4    -122.25     37.85                  52         1627           280.0

   population  households  median_income  median_house_value ocean_proximity  \
0         322         126         8.3252              452600        NEAR BAY
1        2401        1138         8.3014              358500        NEAR BAY
2         496         177         7.2574              352100        NEAR BAY
3         558         219         5.6431              341300        NEAR BAY
4         565         259         3.8462              342200        NEAR BAY

  total_bedroom_size
0             Medium
1              Large
2             Medium
3             Medium
4             Medium
<seaborn.axisgrid.FacetGrid at 0x7a2d1eacee30>
```