Introduction: The dataset given is the brief description about the California housing data.
The dataset is given to predict the median house prices of California.
Below are the details given in the dataset. Let us understand the type of the data given in each of the given features.

Longitude: The values given in this column are Quantitative(Continuous) in nature. Continuous data is always measured and cannot be counted. They are generally collected from precise measurements.

Latitude: The values given in this column are Quantitative(Continuous) in nature. Continuous data is always measured and cannot be counted. They are generally collected from precise measurements.

Housing_median_age: Falls under the quantitative data and it is continuous in nature. It is a housing median age of the house.

Total_rooms: This is of quantitative datatypes and is discrete in nature as the number of rooms can be counted.

Total_bedrooms: This is of quantitative datatype and is discrete in nature as the number of bedrooms can be counted.

Population: This is of quantitative datatype and is discrete in nature as the population can be counted.

Households: This is of quantitative datatype and is discrete in nature.

Median_income: The values given in this column are Quatitative (Continuous) in nature.

Median_house_value: This comes under continuous datatype.

Ocean_proximity: Statistically it is of categorical data i.e., it is of nominal data type. It shows the location of the house with respect to ocean.

```
IMPORTING THE EXCEL FILE TO GOOGLE COLAB

from google.colab import files
uploaded=files.upload()
```

Reading the excel file

```
import numpy as np #importing numpy module
import pandas as pd #importing pandas library
import seaborn as sns #importing seaborn library
import matplotlib.pyplot as plt #importing matplotlib library

df= pd.read_excel("housing.xlsx") #reading the 'housing' excel file
df                                # displaying the excel file
```

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

20640 rows × 10 columns

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.

```
df['median_income'].mean()
```

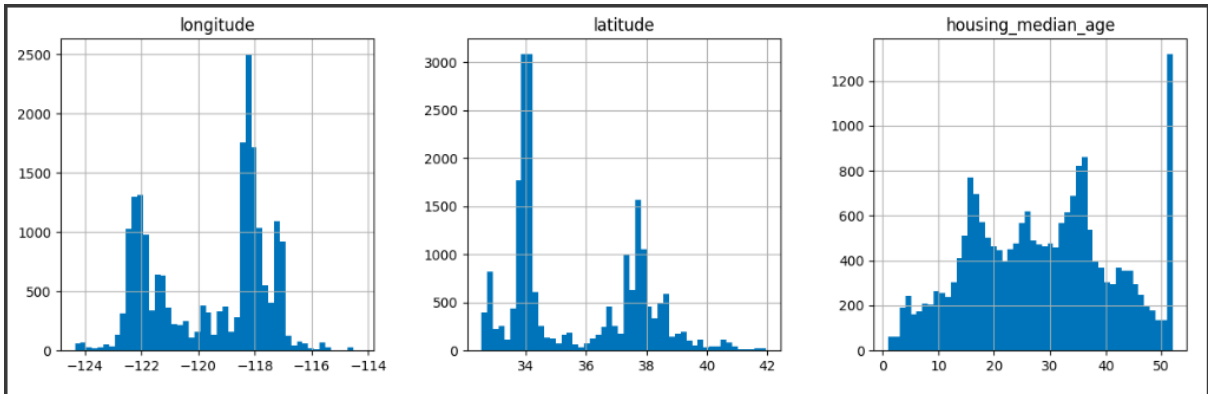
mean() is used in the above code to fetch the average of 'median_income' from the given dataset.
The average median_income of the given data set is Output: 3.8706710029070246

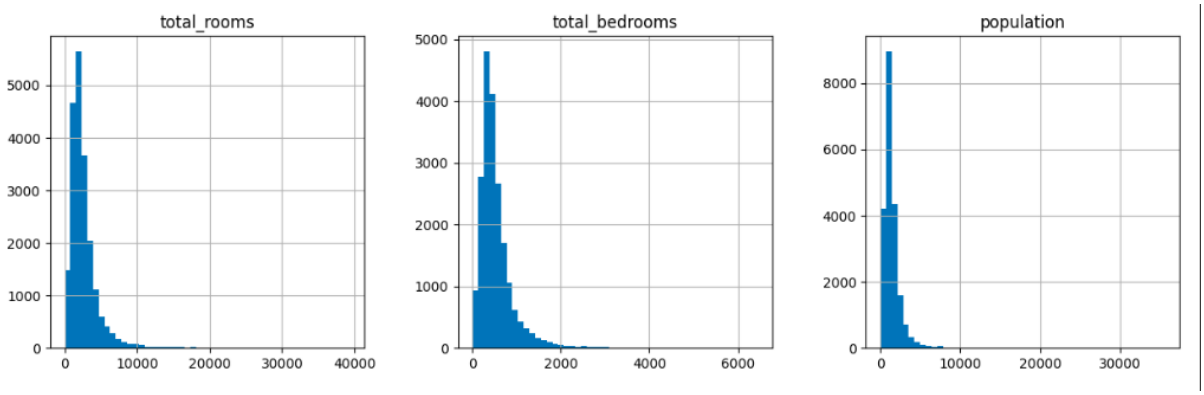
To check the distribution of data

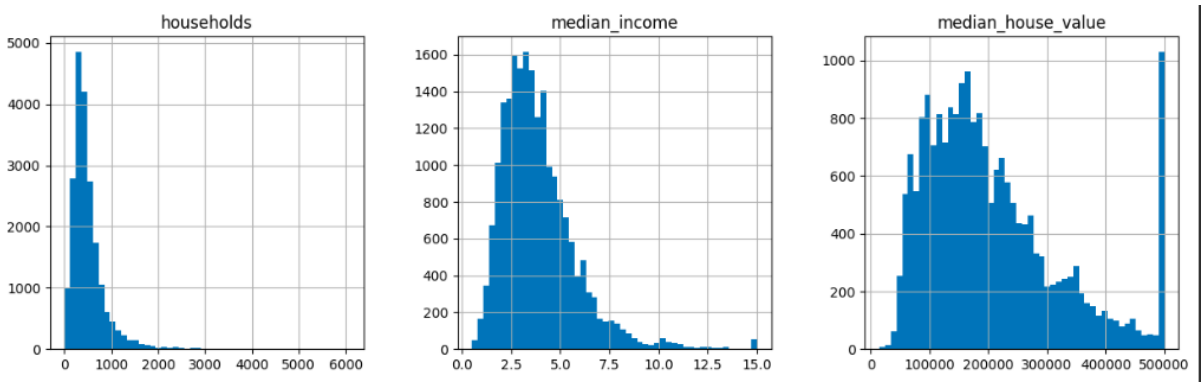
Histogram is used to see the distribution of the numerical data

```
df.hist(bins=50,figsize=(20,20))
plt.show()
```

bins - is used to set up the number of bins to be shown while plotting the graph.
figsize=() is used to set the size of the graph for clear visuals.







Observations:

- The outliers are present for housing_median_age and for median_house_value.
- Below one's are RIGHT SKEWED

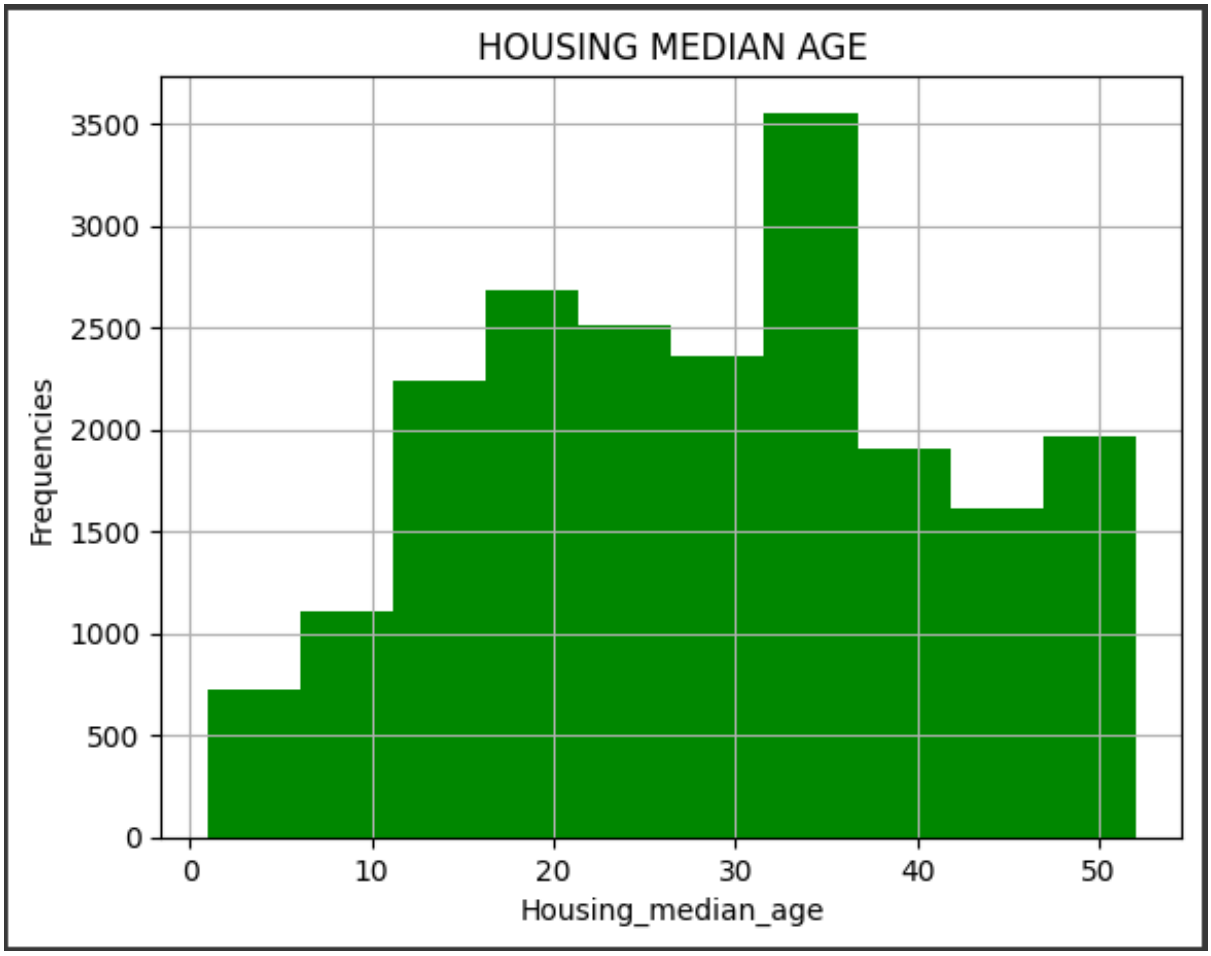
1. total_rooms,
2. total_bedrooms,
3. population,
4. households,
5. median_income

- While latitude and logitude are of asymmetric ,i.e., highly skewed.

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

```
plt.hist(df["housing_median_age"],color='g')  #Histogram is used to see the distribution of a numerical value. 'g' denoting the color of the graph using color method
plt.title("HOUSING MEDIAN AGE") #Placing the title of the graph
plt.xlabel("Housing_median_age") # x-axis=housing_median_age
plt.ylabel("Frequencies") #y-axis=Frequencies
```

```
plt.grid(True) #Setting grid() to TRUE to get the grid on the graph
plt.show() # to display the graph based on above queries
```



From the above hist plot we can come to the analysis that it is distributed symmetrically.

we can know the skewness of the above plot by using :Skewed =3*(mean-median)/standard deviation

```
df['housing_median_age'].mean()
```

Output: `28.639486434108527`

```
df['housing_median_age'].median()
```

Output: `29.0`

```
df['housing_median_age'].std()
```

Output: `12.58555761211165`

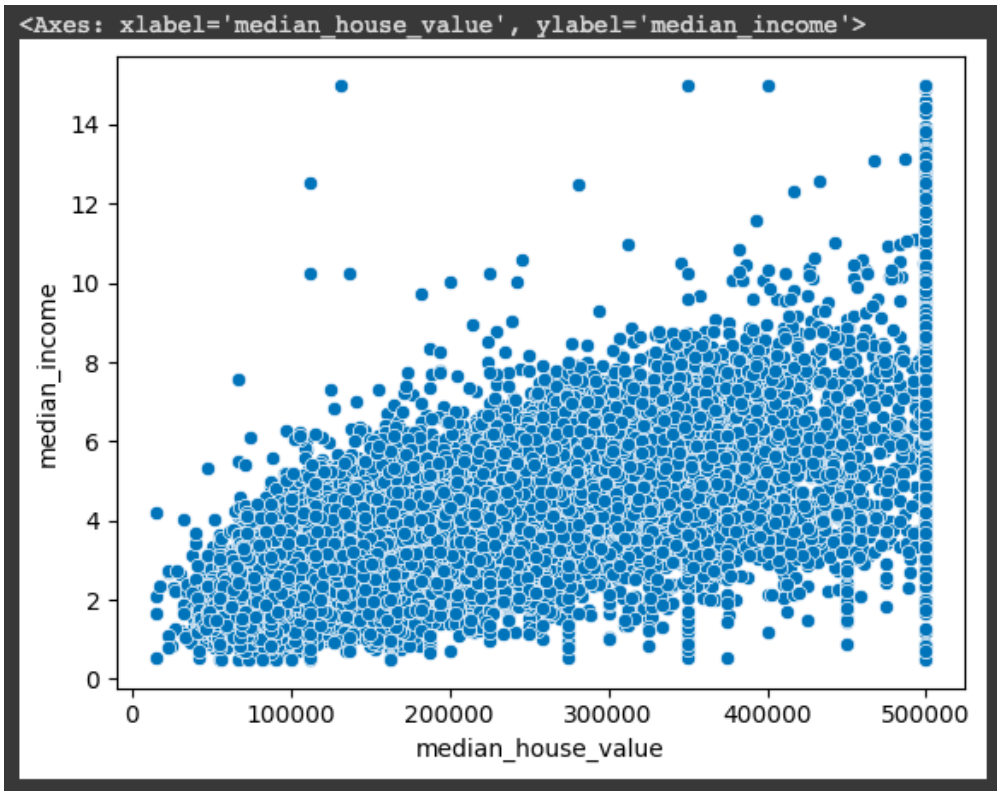
```
Skewed=3*(28.63-29.0)/12.58
Skewed
```

Output: `-0.0882352941176473`

The Skewness of the above plot is -0.08 which is -0.5 to -0.1.From this it is to be concluded that it is of perfectly symmetrical.

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

```
#scatter plot gives the relationship between median_house_value and median_income.
sns.scatterplot(x="median_house_value",y="median_income",data=df)
```



From the above visualisation it is to be analysed that with an increase in the median_house_value there is also an increase in the median income. While, an outlier is present in median_house_value as shown in the figure. Therefore, median_house_value is directly proportional to median income.

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

```
df[df.isnull().any(axis=1)]
```

In the above code, missing values are identified by using isnull() method.

This missing values are identified in the column 'total_bedrooms'.

```
new_data=df.dropna(subset=["total_bedrooms"])
new_data
```

In the above code, the missing values are dropped from the column named ‘total_bedrooms’ by using dropna() method.

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

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

```
df["total_bedrooms"]=df["total_bedrooms"].fillna(df['total_bedrooms'].mean())
df
```

A new dataset had been created where the missing values in the 'total_bedrooms' which are denoted by NaN are replaced with the mean value of the 'total_bedrooms' with the help of .fillna().

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

20640 rows x 10 columns

Row no.290, 341, 538 and other rows with the missing values are replaced with the mean value of 537.87 of 'total_bedrooms'.

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

```
df.head() #The head() returns the first 5 rows for the object based on position.
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41	880	129.0	322	126	8.3252	452600	NEAR BAY
1	-122.22	37.86	21	7099	1106.0	2401	1138	8.3014	358500	NEAR BAY
2	-122.24	37.85	52	1467	190.0	496	177	7.2574	352100	NEAR BAY
3	-122.25	37.85	52	1274	235.0	558	219	5.6431	341300	NEAR BAY
4	-122.25	37.85	52	1627	280.0	565	259	3.8462	342200	NEAR BAY

MEDIAN():

When observed measurements are ranked from smallest to highest. It is not Sensitive to Outliers. Commonly used by Discrete and Continuous data.

A median for a datarframe can be known by using: df['column_name'].median().

A point to be noted that median() can be calculated for Discrete and Continuous data.

Median() function givens the median value that is 50th percentile of the set of observations.

Longitude Median

```
df['longitude'].median()
```

Output:

```
-118.49
```

Latitude Median

```
df['latitude'].median()
```

Output:

```
34.26
```

Housing_Median_Age Median

```
df['housing_median_age'].median()
```

Output:

```
29.0
```

Total Rooms Median

```
df['total_rooms'].median()
```

Output:

```
2127.0
```

Total bedrooms Median

```
df['total_bedrooms'].median()
```

Output:

```
438.0
```

The median()value for 'total_bedrooms' is 438.0

Population Median

```
df['population'].median()
```

Output:

```
1166.0
```

Households Median

```
df['households'].median()
```

Output:

```
409.0
```

Median Income

```
df['median_income'].median()
```

Output:

```
3.5347999999999997
```

Median_house_value Median

```
df['median_house_value'].median()
```

Output:

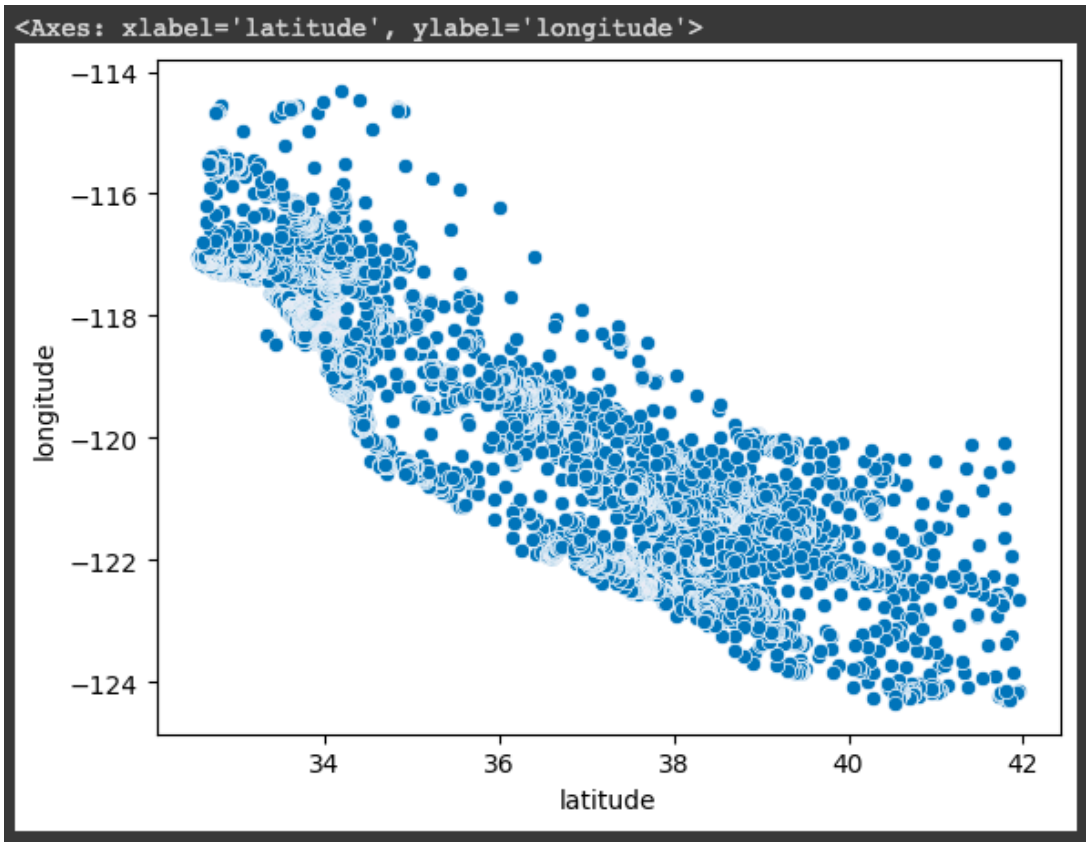
```
179700.0
```

7. Plot latitude versus longitude and explain your observations.

Scatter plot can be used to find the relationship between latitude and longitude.

```
sns.scatterplot(x='latitude',y='longitude',data=df)
```

Output:



From the above plot, it is to be noted that with an decrease in longitude, latitude is increased.

Hence,

- latitude and longitude are not dependent on each other.
- longitude is inversely proportional to latitude.
- From the above plot it is to be noted that latitude vs longitude has negative, i.e., both are moving in an opposite direction.

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

```
new_data=df.loc[df["ocean_proximity"]=="NEAR OCEAN"]
new_data
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
1850	-124.17	41.80	16	2739	480.0	1259	436	3.7557	109400	NEAR OCEAN
1851	-124.30	41.80	19	2672	552.0	1298	478	1.9797	85800	NEAR OCEAN
1852	-124.23	41.75	11	3159	616.0	1343	479	2.4805	73200	NEAR OCEAN
1853	-124.21	41.77	17	3461	722.0	1947	647	2.5795	68400	NEAR OCEAN
1854	-124.19	41.78	15	3140	714.0	1645	640	1.6654	74600	NEAR OCEAN
...	...	...	...	...	...	...	...	...	...	...
20380	-118.83	34.14	16	1316	194.0	450	173	10.1597	500001	NEAR OCEAN
20381	-118.83	34.14	16	1956	312.0	671	319	6.4001	321800	NEAR OCEAN
20423	-119.00	34.08	17	1822	438.0	578	291	5.4346	428600	NEAR OCEAN
20424	-118.75	34.18	4	16704	2704.0	6187	2207	6.6122	357600	NEAR OCEAN
20425	-118.75	34.17	18	6217	858.0	2703	834	6.8075	325900	NEAR OCEAN

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

```
new_data['median_income'].mean()
```

Output:

```
4.0057848006019565
```

```
new_data['median_income'].median()
```

Output:

```
3.64705
```

The mean and median value of the 'median_income' in the created new dataset are: 4.005784 and 3.64705 respectively.

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.

```
import numpy as np
conditions = [
              (df['total_bedrooms'] <=10),
              (df['total_bedrooms'] >=11)&(df['total_bedrooms']
<=1000),
              (df['total_bedrooms'] > 1000)
             ]
values = ['small','medium','large']

df['total_bedroom_size']=np.select(conditions,values)

df
```

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

New column named total_bedroom_size had been added. Where the total_bedroom_size had been compared to the total_bedrooms while mentioning about the sizes of the total_bedroom_size, where:

- If the total_bedroom_size <=10 it is indicated as "small"
- If the total_bedroom_size >=11 or <=1000 it is indicated as "medium"
- If the total_bedroom_size >1000 it is indicated as "large"