```
import pandas as pd
df= pd.read_csv("https://raw.githubusercontent.com/syrusdhark/programming/main/housing.csv")
df
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	populati
0	-122.23	37.88	41.0	880.0	129.0	32
1	-122.22	37.86	21.0	7099.0	1106.0	240
2	-122.24	37.85	52.0	1467.0	190.0	49
3	-122.25	37.85	52.0	1274.0	235.0	55
4	-122.25	37.85	52.0	1627.0	280.0	56
...	...	...	...	...	...	
20635	-121.09	39.48	25.0	1665.0	374.0	84
20636	-121.21	39.49	18.0	697.0	150.0	35
20637	-121.22	39.43	17.0	2254.0	485.0	100
20638	-121.32	39.43	18.0	1860.0	409.0	74
20639	-121.24	39.37	16.0	2785.0	616.0	138

20640 rows × 10 columns

```
df.dropna(inplace=True)


df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 20433 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   longitude           20433 non-null  float64
 1   latitude            20433 non-null  float64
 2   housing_median_age  20433 non-null  float64
 3   total_rooms         20433 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20433 non-null  float64
 6   households          20433 non-null  float64
 7   median_income       20433 non-null  float64
 8   median_house_value  20433 non-null  float64
 9   ocean_proximity     20433 non-null  object 
dtypes: float64(9), object(1)
memory usage: 1.7+ MB
```

```
from sklearn.model_selection import train_test_split

x= df.drop(["median_house_value"],axis=1)

y= df['median_house_value']


x_train,x_test,y_train,y_test = train_test_split(x,y, test_size=0.2)


train_data = x_train.join(y_train)


train_data
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	populati
9667	-119.54	38.51	14.0	1250.0	272.0	72

```
import seaborn as sns
```

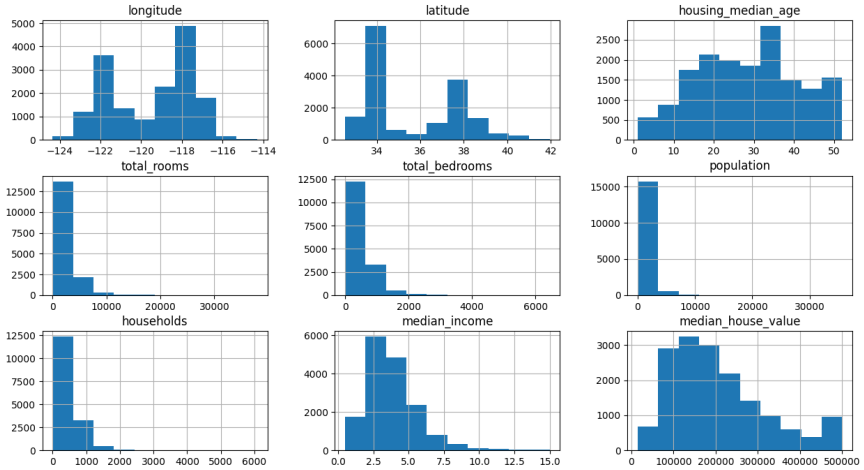
16640	-120.66	35.29	23.0	1932.0	487.0	138

```
import matplotlib.pyplot as plt
import numpy as np
```

...	...	...	...	...	...	

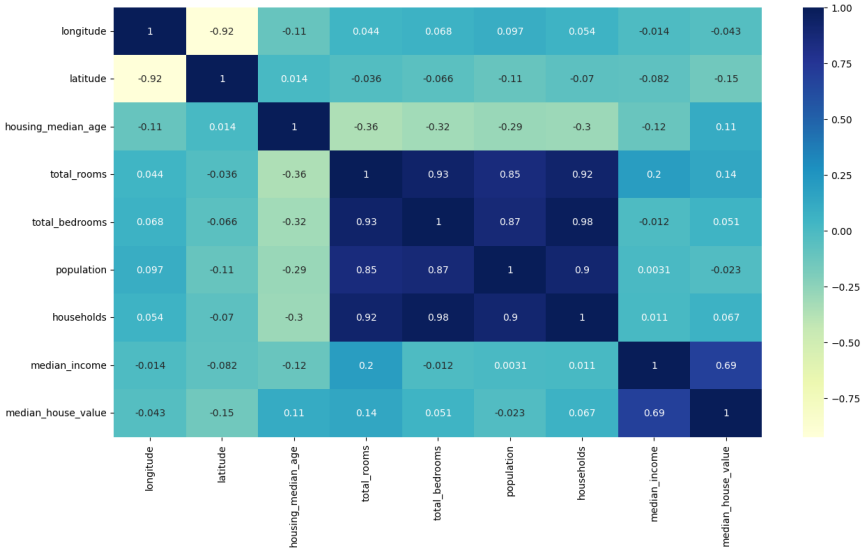
```
train_data.hist(figsize=(15,8))
```

```
array([[<Axes: title={'center': 'longitude'}>,
        <Axes: title={'center': 'latitude'}>,
        <Axes: title={'center': 'housing_median_age'}>],
       [<Axes: title={'center': 'total_rooms'}>,
        <Axes: title={'center': 'total_bedrooms'}>,
        <Axes: title={'center': 'population'}>],
       [<Axes: title={'center': 'households'}>,
        <Axes: title={'center': 'median_income'}>,
        <Axes: title={'center': 'median_house_value'}>]], dtype=object)
```




```
plt.figure(figsize=(15,8))
sns.heatmap(train_data.corr(), annot =True, cmap="YlGnBu")
```

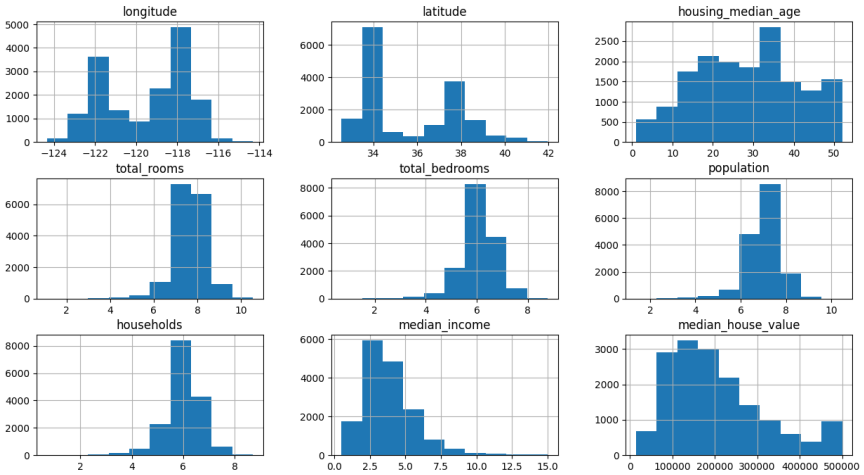
```
<ipython-input-11-e12f1c5680f9>:2: FutureWarning: The default value of numeric_only i
  sns.heatmap(train_data.corr(), annot =True, cmap="YlGnBu")
<Axes: >
```




```
train_data['total_rooms'] = np.log(train_data['total_rooms'] + 1)
train_data['total_bedrooms'] = np.log(train_data['total_bedrooms'] + 1)
train_data['population'] = np.log(train_data['population'] + 1)
train_data['households'] = np.log(train_data['households'] + 1)
```

```
train_data.hist(figsize=(15,8))
```

```
array([[<Axes: title={'center': 'longitude'}>,
        <Axes: title={'center': 'latitude'}>,
        <Axes: title={'center': 'housing_median_age'}>],
       [<Axes: title={'center': 'total_rooms'}>,
        <Axes: title={'center': 'total_bedrooms'}>,
        <Axes: title={'center': 'population'}>],
       [<Axes: title={'center': 'households'}>,
        <Axes: title={'center': 'median_income'}>,
        <Axes: title={'center': 'median_house_value'}>]], dtype=object)
```




```
train_data= train_data.join(pd.get_dummies(train_data.ocean_proximity)).drop(['ocean_proximity'] , axis=1)
```

```
train_data['bedroom_ratio'] = train_data['total_bedrooms'] / train_data['total_rooms']
train_data['household_rooms'] = train_data['total_rooms'] / train_data['households']
```

```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

x_train,y_train = train_data.drop(['median_house_value'] , axis=1 ) , train_data['median_house_value']

x_train_s = scaler.fit_transform(x_train)

reg = LinearRegression()

reg.fit(x_train_s,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
test_data = x_test.join(y_test)

test_data['total_rooms'] = np.log(test_data['total_rooms'] + 1)
test_data['total_bedrooms'] = np.log(test_data['total_bedrooms'] + 1)
test_data['population'] = np.log(test_data['population'] + 1)
test_data['households'] = np.log(test_data['households'] + 1)

test_data= test_data.join(pd.get_dummies(test_data.ocean_proximity)).drop(['ocean_proximity'] , axis=1)

test_data['bedroom_ratio'] = test_data['total_bedrooms'] / test_data['total_rooms']
test_data['household_rooms'] = test_data['total_rooms'] / test_data['households']
```

```
x_test,y_test = test_data.drop(['median_house_value'] , axis=1 ) , test_data['median_house_value']
```

```
from sklearn.ensemble import RandomForestRegressor

forest = RandomForestRegressor()

forest.fit(x_train_s,y_train)
```

```
▾ RandomForestRegressor
RandomForestRegressor()
```

```
forest.score(x_train_s,y_train)
```

```
0.9737426557120061
```