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

```
hdata = pd.read_csv('housdata.csv')
```

```
hdata
```

| | CONTROL | AGE1 | BEDRMS | PER | REGION | METRO3 | LMED | FMR | L30 | L50 | ... | FMTINCRELFMRCAT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | '734778500142' | 56 | 4 | 2 | '4' | '2' | 70998 | 1925 | 18392 | 30628 | ... | '3 GT FMR' |
| **1** | '730204310145' | 54 | 6 | 4 | '2' | '1' | 62425 | 1199 | 18729 | 31213 | ... | '3 GT FMR' |
| **2** | '295273830133' | 39 | 4 | 4 | '1' | '2' | 91523 | 1819 | 27459 | 45761 | ... | '3 GT FMR' |
| **3** | '301895590141' | 49 | 5 | 3 | '2' | '2' | 70700 | 1216 | 19100 | 31800 | ... | '3 GT FMR' |
| **4** | '100007130148' | 30 | 2 | 2 | '3' | '1' | 61059 | 685 | 14662 | 24438 | ... | '2 50.1 - 100% FMR' |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **3946** | '187812570142' | 25 | 3 | 4 | '3' | '1' | 67900 | 1236 | 20350 | 33950 | ... | '3 GT FMR' |
| **3947** | '280828360140' | 67 | 3 | 6 | '2' | '1' | 83900 | 1143 | 29150 | 48650 | ... | '3 GT FMR' |
| **3948** | '187827900149' | 78 | 4 | 2 | '3' | '1' | 61239 | 1047 | 14818 | 24711 | ... | '3 GT FMR' |
| **3949** | '187828980149' | -9 | 4 | -6 | '2' | '2' | 83900 | 1284 | 17600 | 29350 | ... | '.' |
| **3950** | '200958770141' | 44 | 3 | 4 | '2' | '2' | 63054 | 889 | 18943 | 31562 | ... | NaN |

3951 rows × 100 columns

```
# Remove rows with negative values in 'VALUE' column
hdata = hdata[hdata['VALUE'] >= 0]
# Remove rows with null values in 'FMTZADEQ' column
hdata = hdata.dropna(subset=['FMTZADEQ'])
```

```
hdata
```

| | CONTROL | AGE1 | BEDRMS | PER | REGION | METRO3 | LMED | FMR | L30 | L50 | ... | FMTINCRELFMRCAT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | '734778500142' | 56 | 4 | 2 | '4' | '2' | 70998 | 1925 | 18392 | 30628 | ... | '3 GT FMR' |
| **1** | '730204310145' | 54 | 6 | 4 | '2' | '1' | 62425 | 1199 | 18729 | 31213 | ... | '3 GT FMR' |
| **2** | '295273830133' | 39 | 4 | 4 | '1' | '2' | 91523 | 1819 | 27459 | 45761 | ... | '3 GT FMR' |
| **3** | '301895590141' | 49 | 5 | 3 | '2' | '2' | 70700 | 1216 | 19100 | 31800 | | '3 GT FMR' |

## Q1

How does the housing quality vary across different regions? (ZADEQ, REGION)

```
import re

# Define a function to clean and replace values in 'ZADEQ' column
# bcecoues there r 2 values that hv the same meaning "2" and '2'.
def clean_zadeq(value):
    if re.match(r"^\s*'2'?\s*$", value):
        return '2'
    else:
        return value

hdata['ZADEQ'] = hdata['ZADEQ'].apply(clean_zadeq)

hdata['ZADEQ'] = hdata['ZADEQ'].replace('-', np.nan)



## Get the unique values in the 'ZADEQ' column to make sure data is clean
unique_values = hdata['ZADEQ'].unique()

print("Unique values in 'ZADEQ' column:", unique_values)
```

```
Unique values in 'ZADEQ' column: ["'1'" '2' "'-6'" "'3'"]
```

```
# Group the data by 'REGION' and 'ZADEQ' and count the number of occurrences
counts = hdata.groupby(['REGION', 'ZADEQ']).size().reset_index(name='count')

Q1 = counts.pivot(index='REGION', columns='ZADEQ', values='count')

# Fill missing values with 0
Q1 = Q1.fillna(0)

# Rename the columns to proppr naming
Q1 = Q1.rename(columns={"'1'": "Highly Adequate", "2": "Middle Adequate", "'3'": "Low Adequate"})
Q1 = Q1.reindex(columns=["Highly Adequate", "Middle Adequate", "Low Adequate"])

print(Q1)
```

```
ZADEQ   Highly Adequate  Middle Adequate  Low Adequate
REGION
'1'                 589                9             4
'2'                 462                6             2
'3'                 606                4             2
'4'                 511                4             4
```

```
# Calculate the total count of units in each region
total_counts = Q1.sum(axis=1)

# Calculate the percentage of highly adequate units in each region
percentage_highly_adequate = (Q1["Highly Adequate"] / total_counts) * 100

# Display the resulting percentages
print("Percentage of Highly Adequate Units in Each Region:")
print(percentage_highly_adequate)
```

```
Percentage of Highly Adequate Units in Each Region:
REGION
'1'    97.840532
'2'    98.297872
'3'    99.019608
'4'    98.458574
dtype: float64
```

## Q2

which housing unit has the most number of rooms and has an under-average value? (CONTROL, ROOMS, VALUE)

```
# Calculate the average value

hdata['VALUE'] = hdata['VALUE'].replace('-', np.nan)
average_value = hdata['VALUE'].mean()

# Filter to under the average and positive value
filtered_data = hdata[(hdata['VALUE'] < average_value) & (hdata['VALUE'] >= 0)]

# Find the unit with the most rooms
housing_unit = filtered_data[filtered_data['ROOMS'] == filtered_data['ROOMS'].max()]

print(housing_unit[['CONTROL', 'ROOMS', 'VALUE']])

print(average_value)
```

```
             CONTROL  ROOMS   VALUE
455  '184502790140'     14  200000
504217.58601458604
```

## Q3

How does the housing quality vary across different year ranges? (FMTZADEQ, FMTBUILT)

```
unique_values = hdata['FMTBUILT'].unique()
print(unique_values)
```

```
# Remove rows with '-5' in the 'FMTBUILT' column
hdata = hdata[hdata['FMTBUILT'] != "'-5'"]

# Display the filtered data
print(hdata[['CONTROL', 'FMTBUILT']])
```

```
["'2000-2009'" "'1990-1999'" "'1960-1979'" "'1980-1989'" "'1940-1959'"
 "'-5'"]
             CONTROL     FMTBUILT
0     '734778500142'  '2000-2009'
1     '730204310145'  '1990-1999'
2     '295273830133'  '1960-1979'
3     '301895590141'  '1980-1989'
6     '466383110144'  '1960-1979'
...              ...          ...
3939  '730783800149'  '1990-1999'
3944  '288127700140'  '1960-1979'
3945  '299883930344'  '1960-1979'
3947  '280828360140'  '1960-1979'
3949  '187828980149'  '1960-1979'

[2060 rows x 2 columns]
```

```
# Remove rows with '-5' in the 'FMTZADEQ' column
hdata  = hdata [hdata ['FMTZADEQ'] != "'-5'"]

# Display the filtered data
print(hdata [ 'FMTZADEQ'])
```

```
unique_values = hdata['FMTZADEQ'].unique()
```

```
print(unique_values)
```

```
0                 '1 Adequate'
1                 '1 Adequate'
2                 '1 Adequate'
3                 '1 Adequate'
6                 '1 Adequate'
                 ...
3938              '1 Adequate'
3939              '1 Adequate'
3944              '1 Adequate'
3945              '1 Adequate'
3947    '2 Moderately Inadequ'
Name: FMTZADEQ, Length: 1950, dtype: object
["'1 Adequate'" "'2 Moderately Inadequ'" "'3 Severely Indadequa'"]
```

```
# Clean the column
hdata['FMTZADEQ'] = hdata['FMTZADEQ'].str.replace("'", "")

# Group the data by 'FMTBUILT' and 'FMTZADEQ'
grouped_counts = hdata.groupby(['FMTBUILT', 'FMTZADEQ']).size().reset_index(name='count')

Q3 = grouped_counts.pivot(index='FMTZADEQ', columns='FMTBUILT', values='count')

# Sort the columns
Q3 = Q3.reindex(sorted(Q3.columns), axis=1)

print(Q3)
```

```
FMTBUILT              '1940-1959'  '1960-1979'  '1980-1989'  '1990-1999'  \
FMTZADEQ
1 Adequate                  228.0        473.0        288.0        447.0
2 Moderately Inadequ          5.0          4.0          2.0          3.0
3 Severely Indadequa          1.0          4.0          1.0          NaN

FMTBUILT              '2000-2009'
FMTZADEQ
1 Adequate                  492.0
2 Moderately Inadequ          1.0
3 Severely Indadequa          1.0
```

## Q4

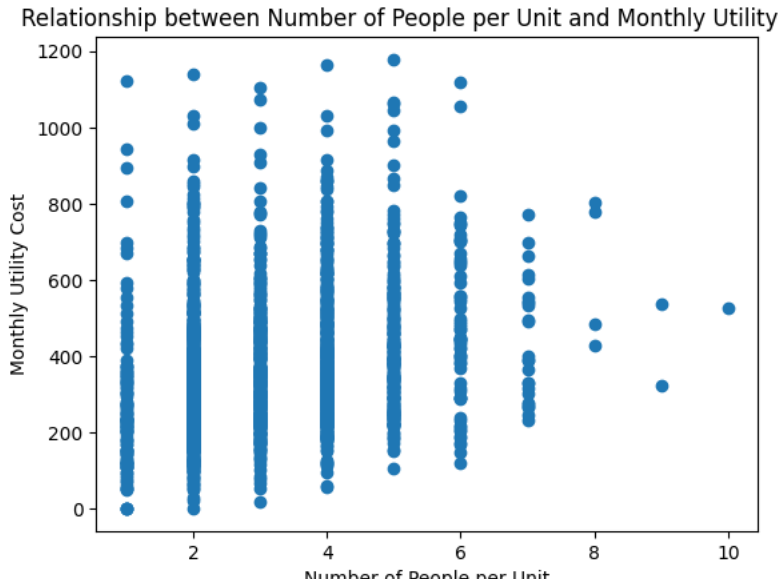
what is the relationship between the number of people per unit and the monthly utility cost? (UTILITY, PER)

```
import pandas as pd

# Calculate the correlation coefficient
correlation = hdata['PER'].corr(hdata['UTILITY'])

print("Correlation coefficient between 'PER' and 'UTILITY':", correlation)
```

```
Correlation coefficient between 'PER' and 'UTILITY': 0.2657760945756594
```

Data Visuallisation:

```
import matplotlib.pyplot as plt

# Create a scatter plot of 'PER' vs 'UTILITY'
plt.scatter(hdata['PER'], hdata['UTILITY'])

# Set plot labels and title
plt.xlabel('Number of People per Unit')
plt.ylabel('Monthly Utility Cost')
plt.title('Relationship between Number of People per Unit and Monthly Utility Cost')

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




## Q5

Are there any significant differences in the adequacy of the housing unit and the Location? (METRO3, FMTZADEQ) METRO3 indicates whether a housing unit is located in a central city, suburb, or outside a metropolitan area.

```
# Compute cross-tabulation of METRO3 and FMTZADEQ
cross_tab = pd.crosstab(hdata['METRO3'], hdata['FMTZADEQ'], dropna=False)
print(cross_tab)
```

```
FMTZADEQ  1 Adequate  2 Moderately Inadequ  3 Severely Indadequa
METRO3
'1'             357                     5                     4
'2'            1020                     7                     3
'3'             364                     3                     0
'4'              71                     0                     0
'5'             116                     0                     0
```

## Q6

How does the quality of housing vary based on the number of occupants in a housing unit? (PER, FMTZADEQ)

```
[ ] ↳ 1 cell hidden
```

## Q7

Are there any disparities in housing adequacy across income groups?(ZINC2,FMTZADEQ)

```
# clean 'ZINC2' column
hdata = hdata[hdata['ZINC2'] >= 0]
print(hdata[['ZINC2']])
```

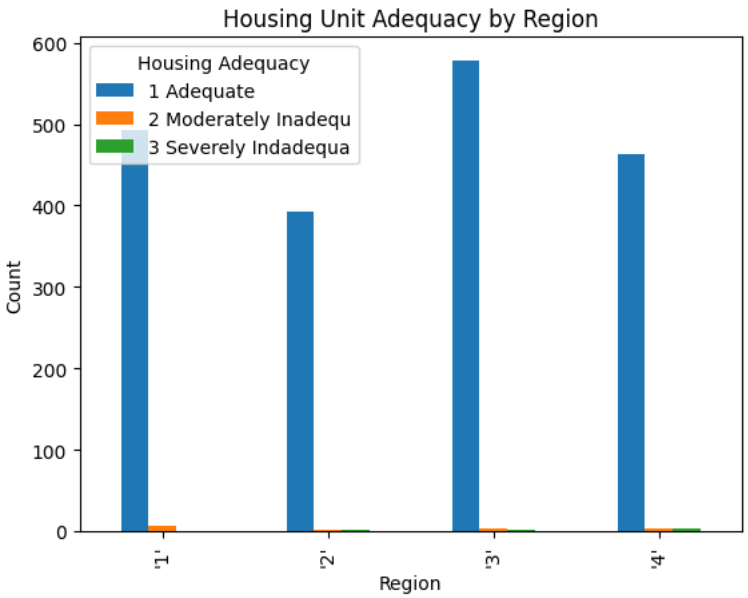
```
        ZINC2
0      785402
1      742441
2      725402
3      695402
6      691480
...       ...
3938   176000
3939   176000
3944   175800
3945   175600
3947   175538

[1949 rows x 1 columns]
```

```
# Create income classes
income_classes = ['Low Income', 'Medium Income', 'High Income']

# Group 'ZINC2' into income classes and create a new column 'IncomeGroup'
hdata['IncomeGroup'] = pd.cut(hdata['ZINC2'], bins=[0, 30000, 70000, float('inf')], labels=income_classes)

# Group the data by 'IncomeGroup' and 'FMTZADEQ'
Q7 = hdata.groupby(['IncomeGroup', 'FMTZADEQ']).size().unstack()

print(Q7)
```

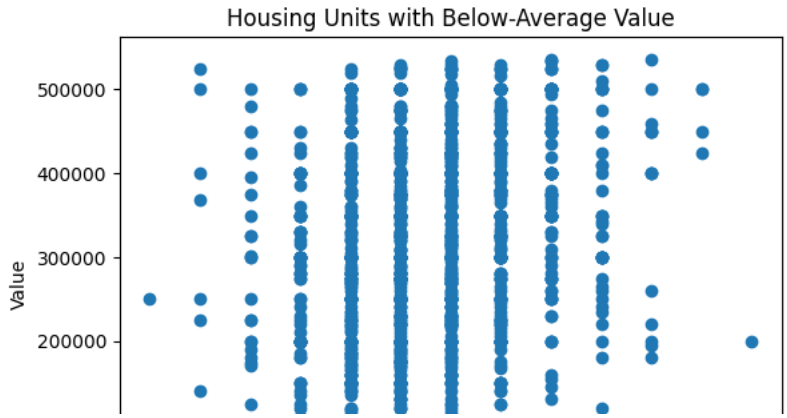
```
FMTZADEQ       1 Adequate  2 Moderately Inadequ  3 Severely Indadequa
IncomeGroup
Low Income              0                     0                     0
Medium Income           0                     0                     0
High Income          1927                    15                     7
```

## Data Visualisation

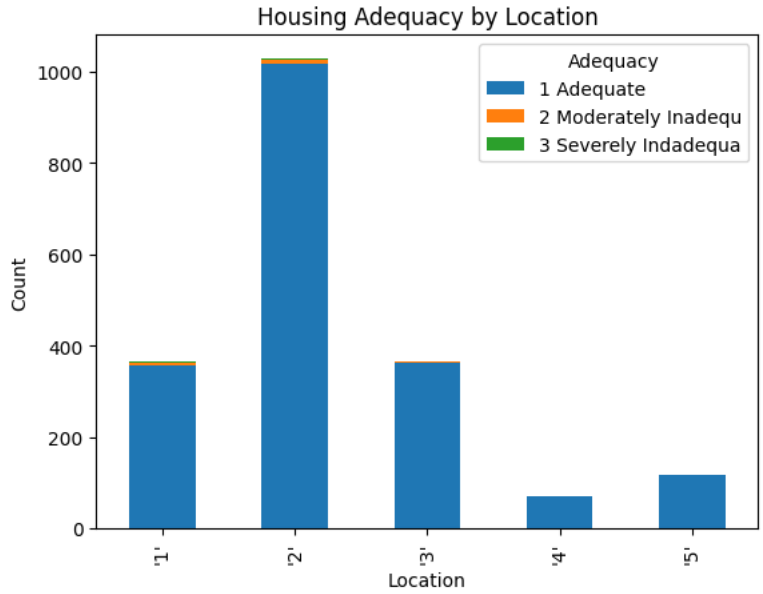
```
region_adequacy_counts = hdata.groupby(['REGION', 'FMTZADEQ']).size().unstack()
region_adequacy_counts.plot(kind='bar')
plt.xlabel('Region')
plt.ylabel('Count')
plt.title('Housing Unit Adequacy by Region')
plt.legend(title='Housing Adequacy')
plt.show()
```



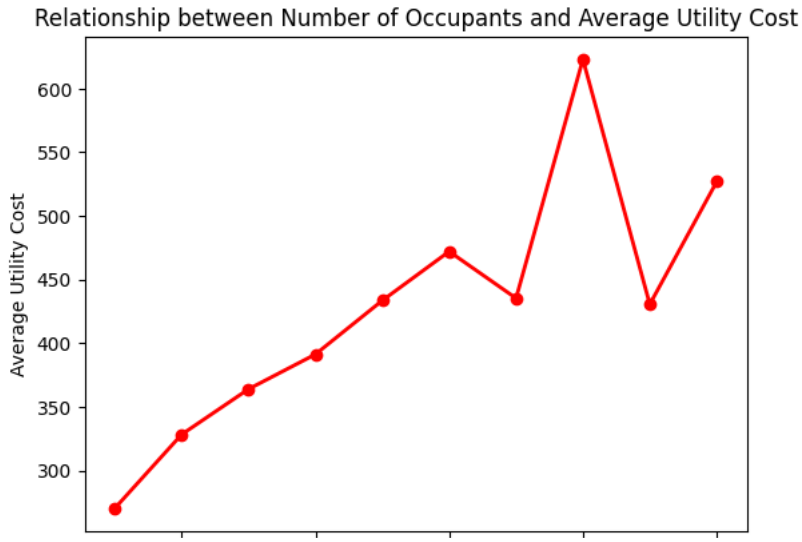

```
# Filter data for housing units with below-average value
below_avg_value = hdata[hdata['VALUE'] < hdata['VALUE'].mean()]

# Plotting
plt.scatter(below_avg_value['ROOMS'], below_avg_value['VALUE'])
plt.xlabel('Number of Rooms')
plt.ylabel('Value')
plt.title('Housing Units with Below-Average Value')
plt.show()
```



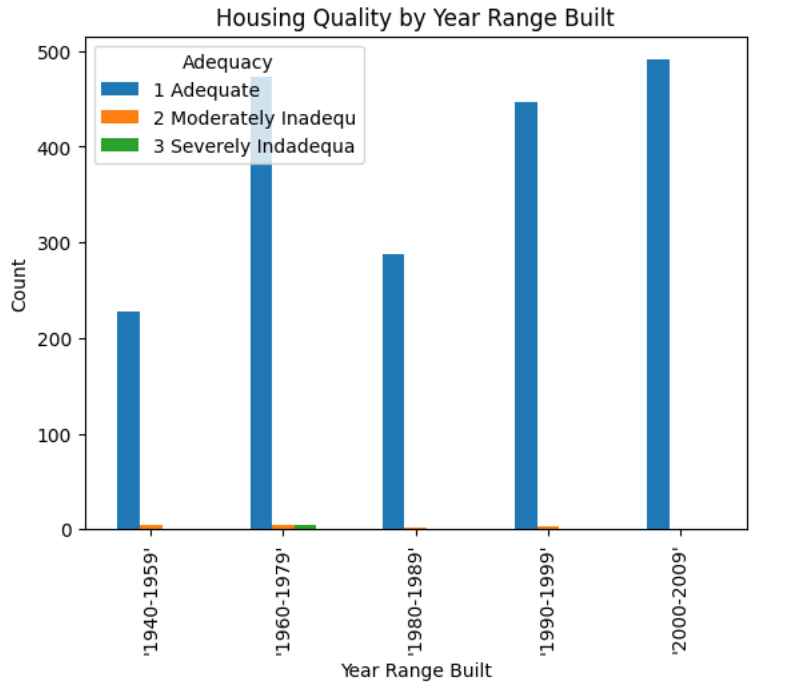

```
import matplotlib.pyplot as plt

# Group data by location and adequacy, and calculate counts
grouped_counts = hdata.groupby(['METRO3', 'FMTZADEQ']).size().unstack()

# Plotting
grouped_counts.plot(kind='bar', stacked=True)
plt.xlabel('Location')
plt.ylabel('Count')
plt.title('Housing Adequacy by Location')
plt.legend(title='Adequacy')
plt.show()
```




```
import matplotlib.pyplot as plt

# Group data by 'PER' and calculate average utility cost
utility_costs = hdata.groupby('PER')['UTILITY'].mean()

# Set up the line plot
fig, ax = plt.subplots()

ax.plot(utility_costs, color='red', marker='o', linestyle='-', linewidth=2)

ax.set_xlabel('Number of Occupants')
ax.set_ylabel('Average Utility Cost')
ax.set_title('Relationship between Number of Occupants and Average Utility Cost')

plt.show()
```




```
grouped_counts = hdata.groupby(['FMTBUILT', 'FMTZADEQ']).size().unstack()

grouped_counts.plot(kind='bar')
plt.xlabel('Year Range Built')
plt.ylabel('Count')
plt.title('Housing Quality by Year Range Built')
plt.legend(title='Adequacy')
plt.show()
```




## remove noise columns

```
columns_to_keep = ['CONTROL', 'PER', 'REGION', 'METRO3',
                   'VALUE', 'ZINC2', 'ROOMS', 'ZADEQ',
                   'UTILITY', 'FMTZADEQ']

# Drop all other columns
hdata = hdata[columns_to_keep]
hdata
```

| | CONTROL | PER | REGION | METRO3 | VALUE | ZINC2 | ROOMS | ZADEQ | UTILITY | F |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | '734778500142' | 2 | '4' | '2' | 2465647 | 785402 | 9 | '1' | 1009.166667 | A |
| **1** | '730204310145' | 4 | '2' | '1' | 740000 | 742441 | 10 | '1' | 381.000000 | A |
| **2** | '295273830133' | 4 | '1' | '2' | 450000 | 725402 | 9 | '1' | 710.166667 | A |
| **3** | '301895590141' | 3 | '2' | '2' | 665000 | 695402 | 9 | '1' | 471.416667 | A |
| **6** | '466383110144' | 5 | '4' | '2' | 1200000 | 691480 | 8 | '1' | 478.666667 | A |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **3938** | '389303160146' | 3 | '3' | '1' | 200000 | 176000 | 10 | '1' | 373.000000 | A |

## output the dataframe

```
# Output the DataFrame to a CSV file
hdata.to_csv('output.csv', index=False)
```

```
from google.colab import files

files.download('output.csv')
```

✓ 0s completed at 8:34 AM