Descriptive Stats Analysis with Income - Expense Data

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
df = pd.read_csv(r'C:\Users\Affan\OneDrive\Desktop\FSDS Course NIT\Prakash Sir S
df
```

Out[2]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annu
0	5000	8000	3	2000	
1	6000	7000	2	3000	
2	10000	4500	2	0	
3	10000	2000	1	0	
4	12500	12000	2	3000	
5	14000	8000	2	0	
6	15000	16000	3	35000	
7	18000	20000	5	8000	
8	19000	9000	2	0	
9	20000	9000	4	0	
10	20000	18000	4	8000	
11	22000	25000	6	12000	
12	23400	5000	3	0	
13	24000	10500	6	0	
14	24000	10000	4	0	
15	25000	12300	3	0	
16	25000	20000	3	3500	
17	25000	10000	6	0	
18	29000	6600	2	2000	
19	30000	13000	4	0	
20	30500	25000	5	5000	
21	32000	15000	4	0	
22	34000	19000	6	0	
23	34000	25000	3	4000	
24	35000	12000	3	0	
25	35000	25000	4	0	
26	39000	8000	4	0	
27	40000	10000	4	0	
28	42000	15000	4	0	
29	43000	12000	4	0	
30	45000	25000	6	0	
31	45000	40000	6	3500	
32	45000	10000	2	1000	

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annu
33	45000	22000	4	2500	
34	46000	25000	5	3500	
35	47000	15000	7	0	
36	50000	20000	4	0	
37	50500	20000	3	0	
38	55000	45000	6	12000	
39	60000	10000	3	0	
40	60000	50000	6	10000	
41	65000	20000	4	5000	
42	70000	9000	2	0	
43	80000	20000	4	0	
44	85000	25000	5	0	
45	90000	48000	7	0	
46	98000	25000	5	0	
47	100000	30000	6	0	
48	100000	50000	4	20000	
49	100000	40000	6	10000	

In [3]:
```
df.head()
```

Out[3]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annua
0	5000	8000	3	2000	
1	6000	7000	2	3000	
2	10000	4500	2	0	
3	10000	2000	1	0	
4	12500	12000	2	3000	

In [4]:
```
df.tail()
```

Out[4]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annu
45	90000	48000	7	0	
46	98000	25000	5	0	
47	100000	30000	6	0	
48	100000	50000	4	20000	
49	100000	40000	6	10000	

In [5]:
```
df.columns
```

Out[5]:
```
Index(['Mthly_HH_Income', 'Mthly_HH_Expense', 'No_of_Fly_Members',
       'Emi_or_Rent_Amt', 'Annual_HH_Income', 'Highest_Qualified_Member',
       'No_of_Earning_Members'],
      dtype='object')
```

In [9]:
```
df['No_of_Earning_Members'].unique()
```

Out[9]:
```
array([1, 2, 3, 4], dtype=int64)
```

In [10]:
```
df.describe()
```

Out[10]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Ar
count	50.000000	50.000000	50.000000	50.000000	
mean	41558.000000	18818.000000	4.060000	3060.000000	
std	26097.908979	12090.216824	1.517382	6241.434948	
min	5000.000000	2000.000000	1.000000	0.000000	
25%	23550.000000	10000.000000	3.000000	0.000000	
50%	35000.000000	15500.000000	4.000000	0.000000	
75%	50375.000000	25000.000000	5.000000	3500.000000	
max	100000.000000	50000.000000	7.000000	35000.000000	

In [11]:
```
df.shape
```

Out[11]:
```
(50, 7)
```

In [12]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 7 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Mthly_HH_Income           50 non-null     int64 
 1   Mthly_HH_Expense          50 non-null     int64 
 2   No_of_Fly_Members         50 non-null     int64 
 3   Emi_or_Rent_Amt           50 non-null     int64 
 4   Annual_HH_Income          50 non-null     int64 
 5   Highest_Qualified_Member  50 non-null     object
 6   No_of_Earning_Members     50 non-null     int64 
dtypes: int64(6), object(1)
memory usage: 2.9+ KB
```

In [13]:
```
df.isnull()
```

Out[13]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annu
0	False	False	False	False	
1	False	False	False	False	
2	False	False	False	False	
3	False	False	False	False	
4	False	False	False	False	
5	False	False	False	False	
6	False	False	False	False	
7	False	False	False	False	
8	False	False	False	False	
9	False	False	False	False	
10	False	False	False	False	
11	False	False	False	False	
12	False	False	False	False	
13	False	False	False	False	
14	False	False	False	False	
15	False	False	False	False	
16	False	False	False	False	
17	False	False	False	False	
18	False	False	False	False	
19	False	False	False	False	
20	False	False	False	False	
21	False	False	False	False	
22	False	False	False	False	
23	False	False	False	False	
24	False	False	False	False	
25	False	False	False	False	
26	False	False	False	False	
27	False	False	False	False	
28	False	False	False	False	
29	False	False	False	False	
30	False	False	False	False	
31	False	False	False	False	
32	False	False	False	False	

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annu
33	False	False	False	False	
34	False	False	False	False	
35	False	False	False	False	
36	False	False	False	False	
37	False	False	False	False	
38	False	False	False	False	
39	False	False	False	False	
40	False	False	False	False	
41	False	False	False	False	
42	False	False	False	False	
43	False	False	False	False	
44	False	False	False	False	
45	False	False	False	False	
46	False	False	False	False	
47	False	False	False	False	
48	False	False	False	False	
49	False	False	False	False	

In [14]:
```
df.isnull().count()
```

Out[14]:
```
Mthly_HH_Income            50
Mthly_HH_Expense           50
No_of_Fly_Members          50
Emi_or_Rent_Amt            50
Annual_HH_Income           50
Highest_Qualified_Member   50
No_of_Earning_Members      50
dtype: int64
```

In [15]:
```
df.isna().any()
```

Out[15]:
```
Mthly_HH_Income            False
Mthly_HH_Expense           False
No_of_Fly_Members          False
Emi_or_Rent_Amt            False
Annual_HH_Income           False
Highest_Qualified_Member   False
No_of_Earning_Members      False
dtype: bool
```

what is the mean expense of a household?

```
In [17]: df['Mthly_HH_Expense'].mean()
```

```
Out[17]: 18818.0
```

```
In [18]: df['Mthly_HH_Expense'].median()
```

```
Out[18]: 15500.0
```

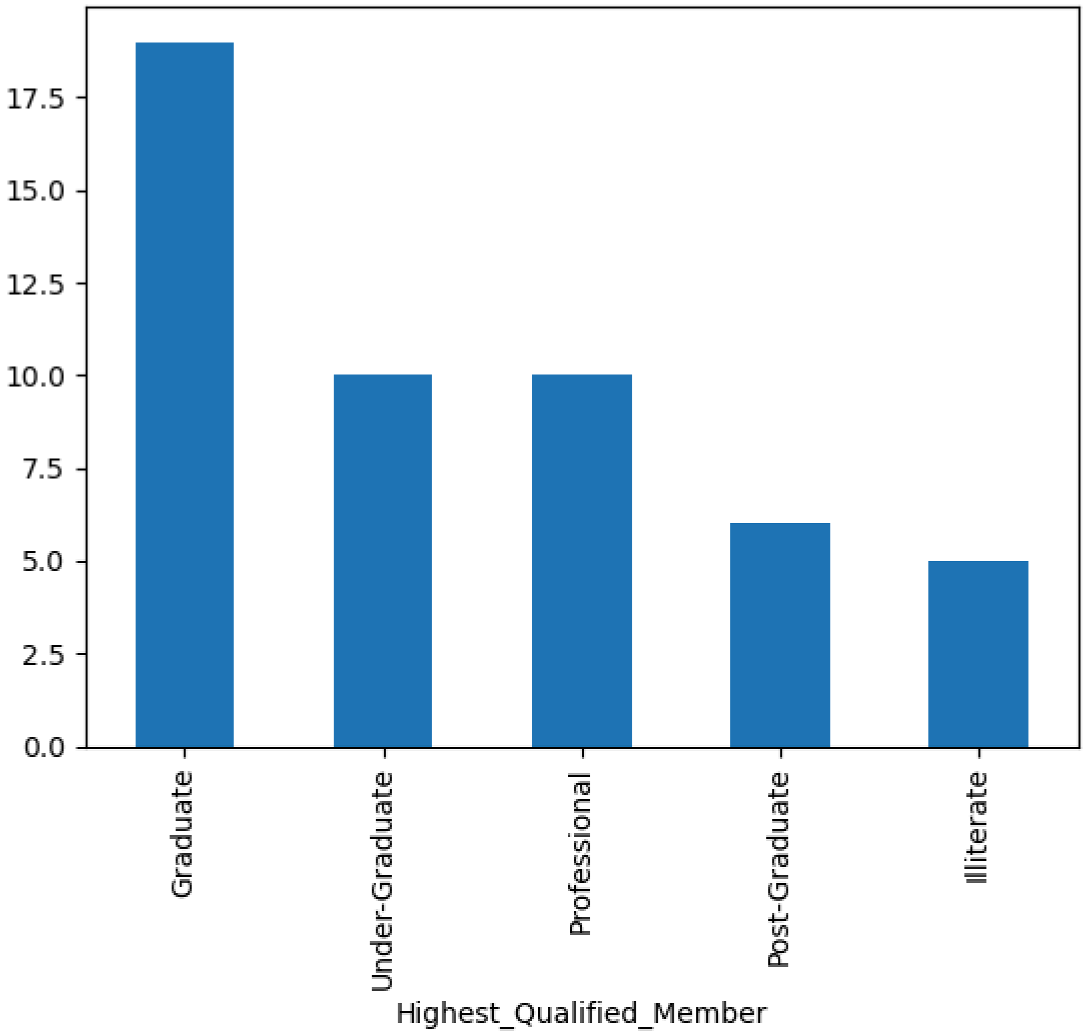
```
In [21]: mth_exp_tmp = pd.crosstab(index=df["Mthly_HH_Expense"], columns="count")
         mth_exp_tmp.reset_index(inplace=True)
         mth_exp_tmp[mth_exp_tmp['count'] == df.Mthly_HH_Expense.value_counts().max()]
```

Out[21]:

col_0	Mthly_HH_Expense	count
18	25000	8

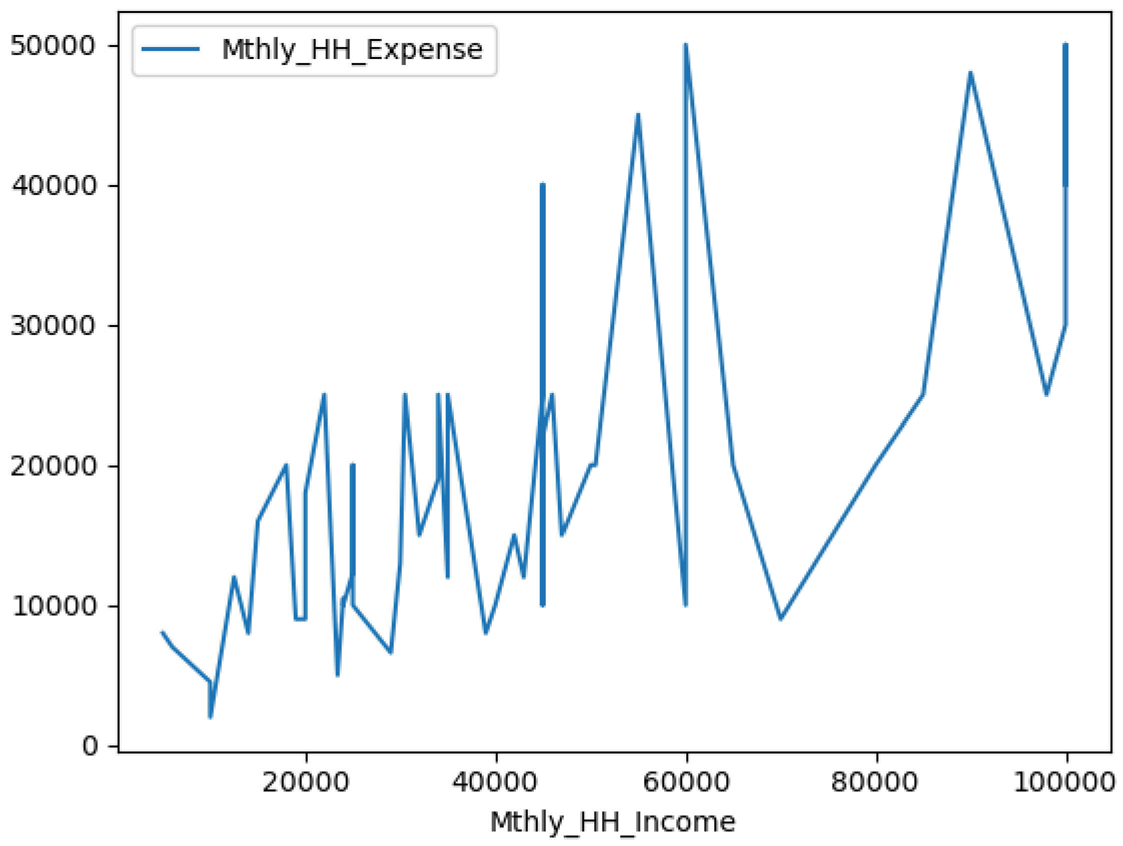
```
In [25]: df['Highest_Qualified_Member'].value_counts().plot(kind='bar')
```

```
Out[25]: <Axes: xlabel='Highest_Qualified_Member'>
```



```
In [27]: df.plot(x='Mthly_HH_Income',y='Mthly_HH_Expense')
         IQR = df['Mthly_HH_Expense'].quantile(0.75)-df['Mthly_HH_Expense'].quantile(0.25
         IQR
```

```
Out[27]: 15000.0
```



```
In [31]: df.iloc[:,0:5].std().to_frame().T
```

Out[31]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annua
0	26097.908979	12090.216824	1.517382	6241.434948	3.

```
In [35]: df.iloc[:,0:4].var().to_frame().T
```

Out[35]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt
0	6.811009e+08	1.461733e+08	2.302449	3.895551e+07

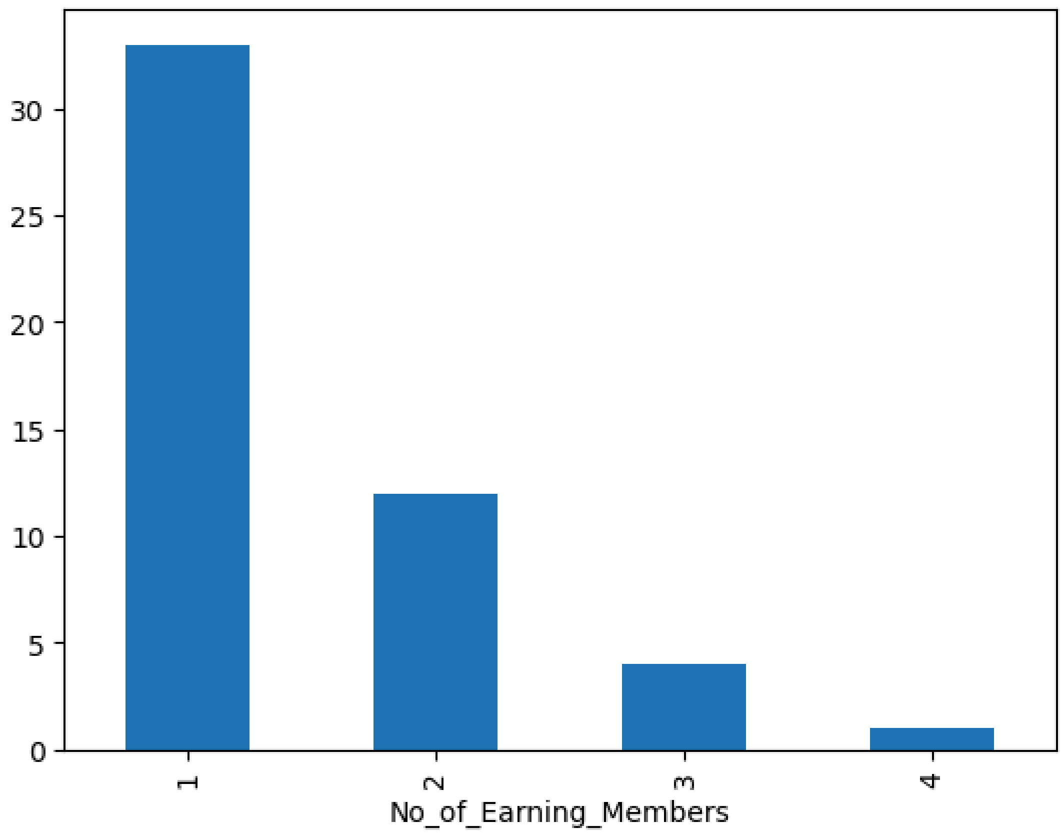
```
In [37]: df['Highest_Qualified_Member'].value_counts().to_frame().T
```

Out[37]:

Highest_Qualified_Member	Graduate	Under-Graduate	Professional	Post-Graduate	Illiterate
count	19	10	10	6	5

```
In [42]: df['No_of_Earning_Members'].value_counts().plot(kind='bar')
```

```
Out[42]: <Axes: xlabel='No_of_Earning_Members'>
```



13.Suppose you have option to invest in Stock A or Stock B. The stocks • have different expected returns and standard deviations. The expected return of Stock A is 15% and Stock B is 10%. Standard Deviation of the returns of these stocks is 10% and 5% respectively.

Which is better investment?

In [43]:
```
#Here we need to calculate the coeff of variation

Coeff_of_var_StockA=10/15
print(Coeff_of_var_StockA)
Coeff_of_var_StockB=5/10
print(Coeff_of_var_StockB)
```

```
0.6666666666666666
0.5
```

In [ ]:
```
```