In [1]:

```
## Import Packages

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

%matplotlib inline

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

In [2]:

```
# Load the Dataset

income=pd.read_csv(r"D:\Full Stack Data Science\24 Aug\24 Aug\Descriptive stats\Inc_
income
```

Out[2]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	Highest
0	5000	8000	3	2000	64200	
1	6000	7000	2	3000	79920	
2	10000	4500	2	0	112800	
3	10000	2000	1	0	97200	
4	12500	12000	2	3000	147000	
5	14000	8000	2	0	196560	
6	15000	16000	3	35000	167400	
7	18000	20000	5	8000	216000	
8	19000	9000	2	0	218880	
9	20000	9000	4	0	220800	
10	20000	18000	4	8000	278400	
11	22000	25000	6	12000	279840	
12	23400	5000	3	0	292032	
13	24000	10500	6	0	316800	
14	24000	10000	4	0	244800	
15	25000	12300	3	0	246000	
16	25000	20000	3	3500	261000	
17	25000	10000	6	0	258000	
18	29000	6600	2	2000	348000	
19	30000	13000	4	0	385200	
20	30500	25000	5	5000	351360	
21	32000	15000	4	0	445440	
22	34000	19000	6	0	330480	
23	34000	25000	3	4000	469200	
24	35000	12000	3	0	466200	
25	35000	25000	4	0	449400	
26	39000	8000	4	0	556920	
27	40000	10000	4	0	412800	
28	42000	15000	4	0	488880	
29	43000	12000	4	0	619200	
30	45000	25000	6	0	523800	
31	45000	40000	6	3500	507600	
32	45000	10000	2	1000	437400	
33	45000	22000	4	2500	610200	
34	46000	25000	5	3500	596160	
35	47000	15000	7	0	456840	
36	50000	20000	4	0	570000	
37	50500	20000	3	0	581760	
38	55000	45000	6	12000	600600	

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	Highest
39	60000	10000	3	0	590400	
40	60000	50000	6	10000	590400	
41	65000	20000	4	5000	647400	
42	70000	9000	2	0	756000	
43	80000	20000	4	0	1075200	
44	85000	25000	5	0	1142400	
45	90000	48000	7	0	885600	
46	98000	25000	5	0	1152480	
47	100000	30000	6	0	1404000	
48	100000	50000	4	20000	1032000	
49	100000	40000	6	10000	1320000	

In [3]:
```
# To check the dimensions of dataset
income.shape
```

Out[3]:
```
(50, 7)
```

There is 50 rows and 7 columns in our dataset.

In [5]:
```
# To check the size of dataset , size=rows*columns
income.size
```

Out[5]:
```
350
```

In [6]:
```
# This gives the information about dataset
income.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 [7]:
```
# To find Descriptive Statistics
income.describe()
```

Out[7]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	No_
count	50.000000	50.000000	50.000000	50.000000	5.000000e+01	
mean	41558.000000	18818.000000	4.060000	3060.000000	4.900190e+05	
std	26097.908979	12090.216824	1.517382	6241.434948	3.201358e+05	
min	5000.000000	2000.000000	1.000000	0.000000	6.420000e+04	
25%	23550.000000	10000.000000	3.000000	0.000000	2.587500e+05	
50%	35000.000000	15500.000000	4.000000	0.000000	4.474200e+05	
75%	50375.000000	25000.000000	5.000000	3500.000000	5.947200e+05	
max	100000.000000	50000.000000	7.000000	35000.000000	1.404000e+06	

Interpretation

- 1. Count of every column is 50.
- 2. Mean of monthly houshold income is 41558 Rs.
- 3. Mean of monthly household expenses is 18818 Rs.

In [8]:
```
# To check the null values in dataset
income.isnull().sum()
```

Out[8]:
```
Mthly_HH_Income             0
Mthly_HH_Expense            0
No_of_Fly_Members           0
Emi_or_Rent_Amt             0
Annual_HH_Income            0
Highest_Qualified_Member    0
No_of_Earning_Members       0
dtype: int64
```

In [10]:
```
income.isnull().any().any()
```

Out[10]:
```
False
```

There is no null values in our dataset.

In [11]:
```
# Mean of expenses of household
income.Mthly_HH_Expense.mean()
```

Out[11]:
```
18818.0
```

In [12]:
```
# Median of household expense
income.Mthly_HH_Expense.median()
```

Out[12]:
```
15500.0
```

In [14]:
```
# Monthly Expense for most of the household
income.Mthly_HH_Expense.mode()
```

Out[14]:
```
Mthly_HH_Expense
25000    1
Name: count, dtype: int64
```

In [24]:
```
# Crosstab is used for Compute a simple cross tabulation of two (or more) factors.

common_expense=pd.crosstab(index=income.Mthly_HH_Expense,columns='count')
common_expense.reset_index(inplace=True)
common_expense[common_expense['count']==income.Mthly_HH_Expense.value_counts().max()
```

Out[24]:

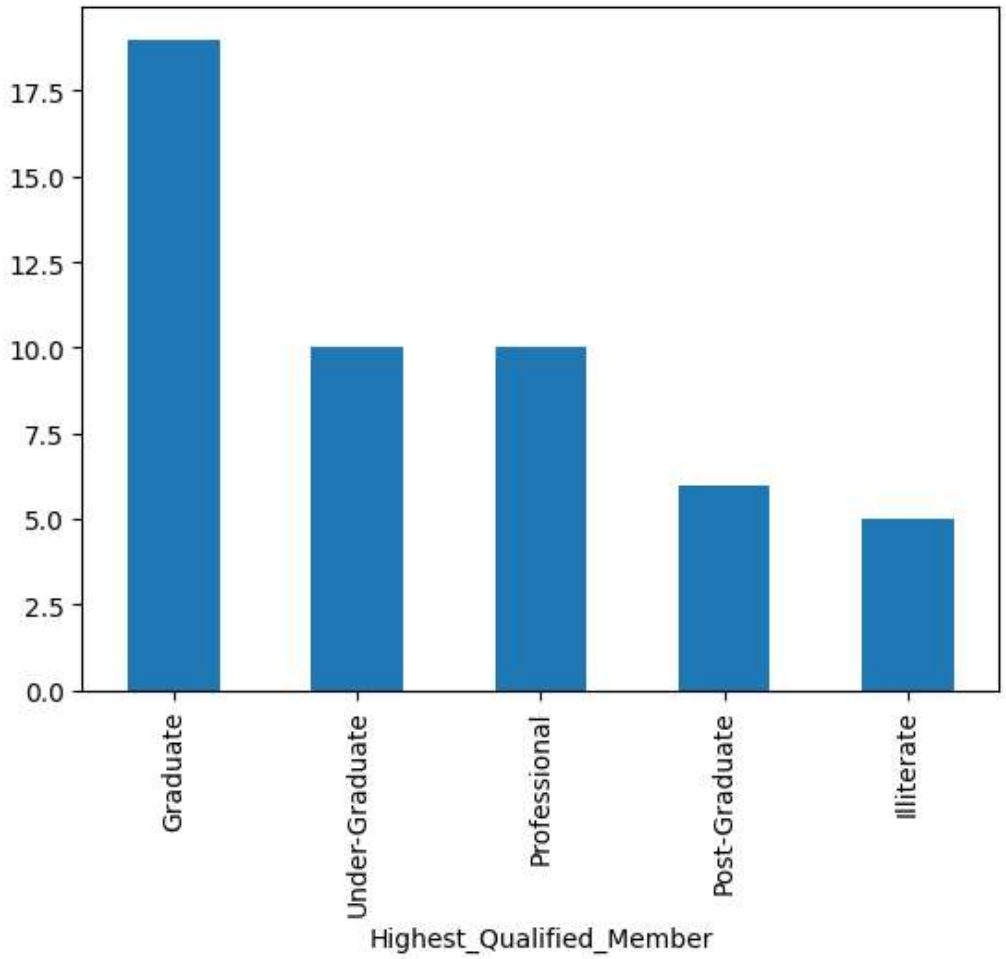
col_0	Mthly_HH_Expense	count
18	25000	8

In [25]:
```
expense_count=income.Mthly_HH_Expense.value_counts()
max_count=expense_count.max()
most_common_expense=expense_count[expense_count==max_count]
most_common_expense
```

Out[25]:
```
Mthly_HH_Expense
25000    8
Name: count, dtype: int64
```

In [38]:
```
# Plot the histogram to count the highest qualified member
income.Highest_Qualified_Member.value_counts().plot(kind='bar')
```

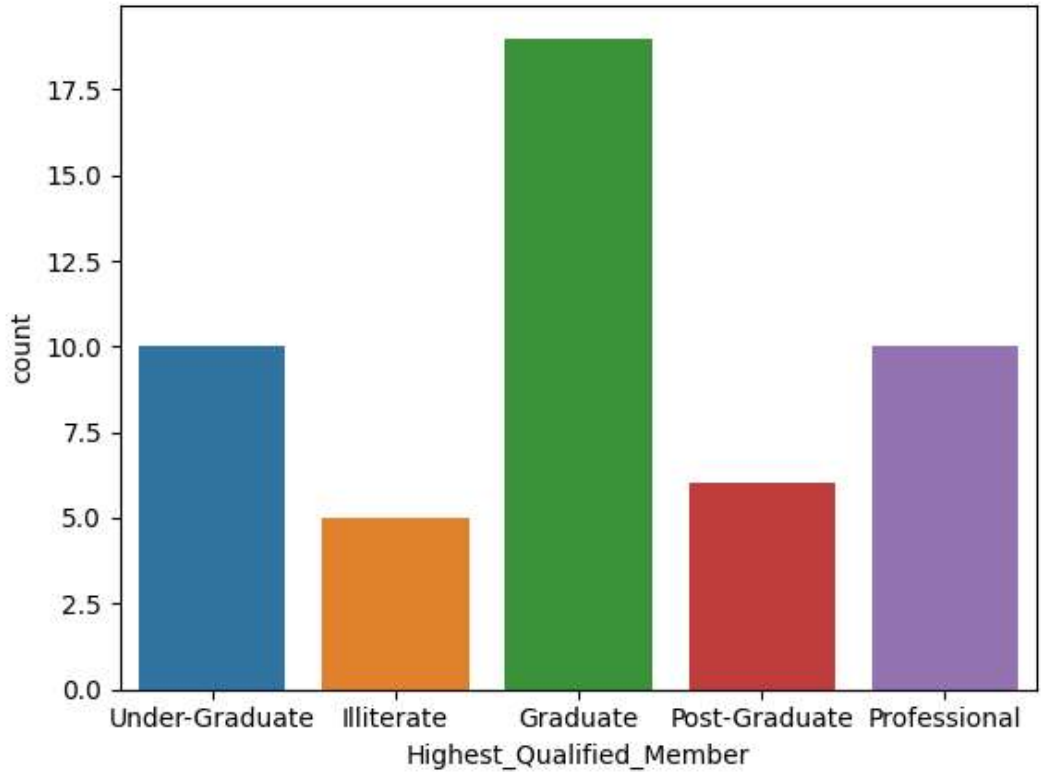
Out[38]: <Axes: xlabel='Highest_Qualified_Member'>



OR

In [44]:
```
1 sns.countplot(data=income,x='Highest_Qualified_Member')
```

Out[44]: <Axes: xlabel='Highest_Qualified_Member', ylabel='count'>

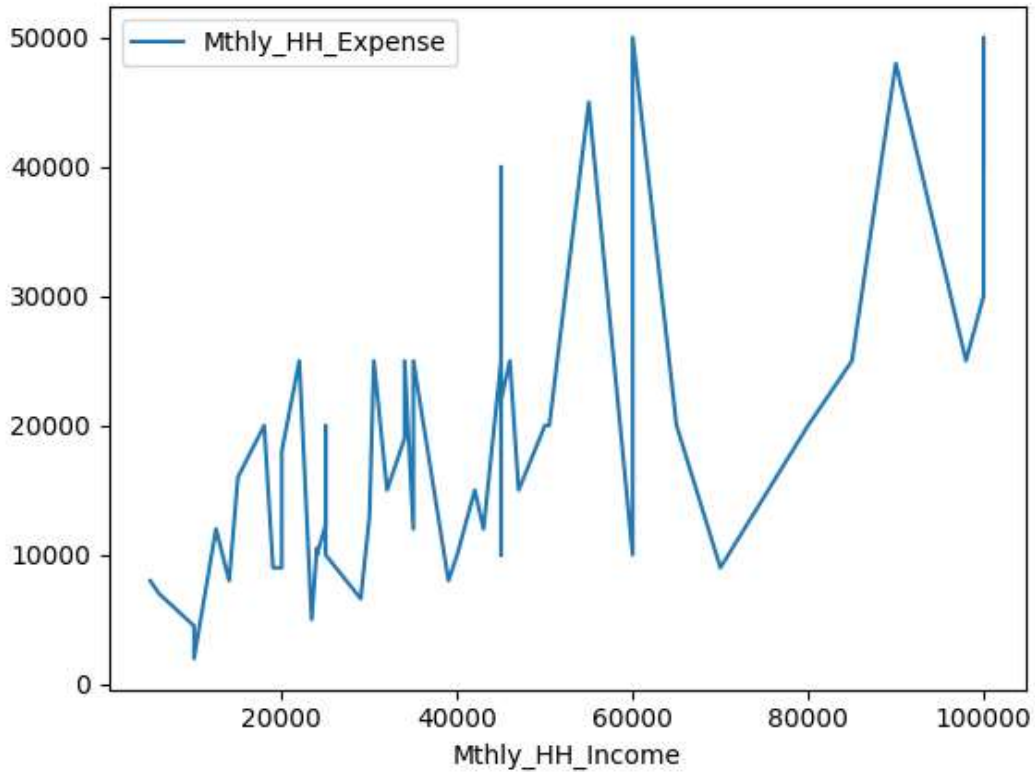


In [46]:
```
1 # Calculate IQR(Difference between 75% and 25% Quartile)
2
3 IQR=income.Mthly_HH_Expense.quantile(0.75)-income.Mthly_HH_Expense.quantile(0.25)
4 IQR
```

Out[46]: 15000.0

In [47]:
```
# Plot of Monthly Household Income and Expense
income.plot(x='Mthly_HH_Income',y='Mthly_HH_Expense')
```

Out[47]: <Axes: xlabel='Mthly_HH_Income'>



In [70]:
```
# Calculate Standard Deviation for First 4 columns
pd.DataFrame(income.iloc[:,0:4].std().to_frame()).T
```

Out[70]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt
0	26097.908979	12090.216824	1.517382	6241.434948

In [78]:
```
std=income.describe()
std[2:3]
```

Out[78]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	No_of_
std	26097.908979	12090.216824	1.517382	6241.434948	320135.792123	

In [74]:
```
# Variance of first 3 columns
pd.DataFrame(income.iloc[:,0:3].var().to_frame()).T
```

Out[74]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members
0	6.811009e+08	1.461733e+08	2.302449

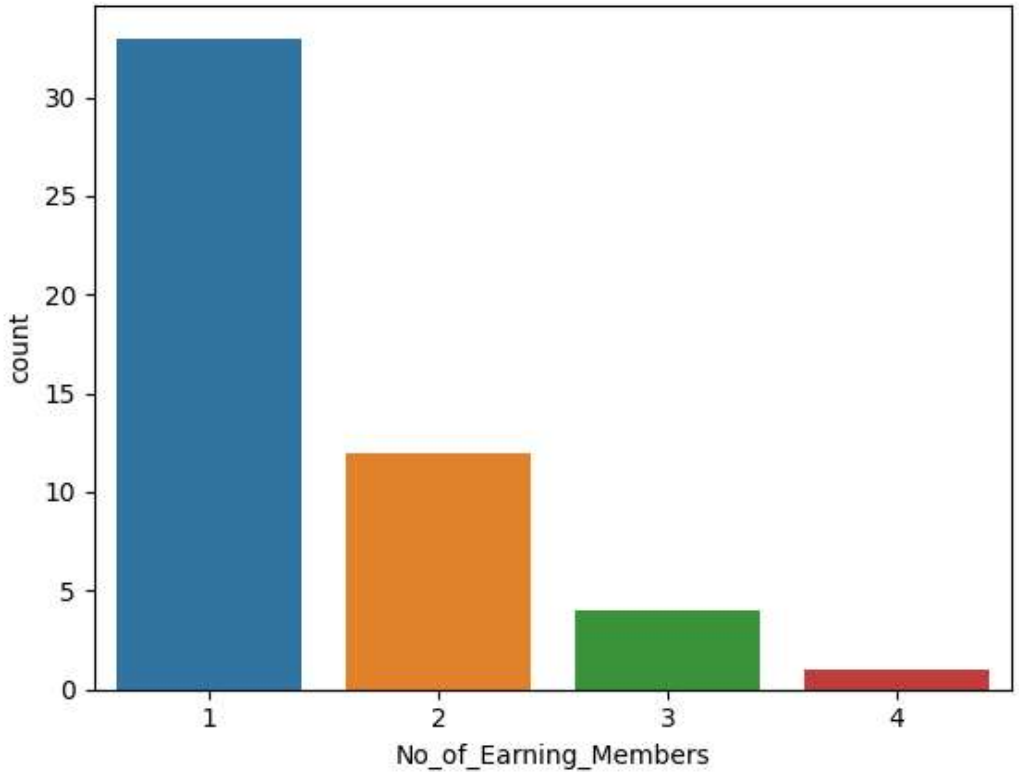
In [83]:
```
# Calculate the count of highest qualified member

income.Highest_Qualified_Member.value_counts()
```

Out[83]:
```
Highest_Qualified_Member
Graduate          19
Under-Graduate    10
Professional      10
Post-Graduate      6
Illiterate         5
Name: count, dtype: int64
```

In [85]:
```
# Plot the Histogram to count the No_of_Earning_Member
sns.countplot(data=income,x='No_of_Earning_Members')
```

Out[85]:
```
<Axes: xlabel='No_of_Earning_Members', ylabel='count'>
```



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?

Here we need to calculate the coefficient of variation

Stock A

- Expected Returns: 15%
- Expected Standard Deviation:10%

Stock B

- Expected Returns: 10%
- Expected Standard Deviation:5%

In [91]:
```
# Coefficient of Variation
# CV_A=(S.D/Mean)*100

CV_A=(10/15)*100
print(CV_A)

CV_B=(5/10)*100
print(CV_B)
```

```
66.66666666666666
50.0
```

- Stock A has a coefficient of variation of 66.67%, indicating that its risk (measured by standard deviation) is relatively high compared to its expected return.
- Stock B has a coefficient of variation of 50%, indicating that its risk is relatively lower compared to its expected return.

In this case, Stock B has a lower coefficient of variation, suggesting that it offers better risk-adjusted returns compared to Stock A

In [ ]:
```
1
```