BLACK FRIDAY SALES ANALYSIS

In [2]:

```
import pandas as pd
import matplotlib
import seaborn as sns
```

In [3]:

```
df = pd.read_csv('BlackFriday.csv')
```

1. Dataset Walkthrough :

In [4]:

```
df.info() # Provides basic info of dataset
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 537577 entries, 0 to 537576
Data columns (total 12 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   User_ID                     537577 non-null  int64  
 1   Product_ID                  537577 non-null  object 
 2   Gender                      537577 non-null  object 
 3   Age                         537577 non-null  object 
 4   Occupation                  537577 non-null  int64  
 5   City_Category               537577 non-null  object 
 6   Stay_In_Current_City_Years  537577 non-null  object 
 7   Marital_Status              537577 non-null  int64  
 8   Product_Category_1          537577 non-null  int64  
 9   Product_Category_2          370591 non-null  float64
 10  Product_Category_3          164278 non-null  float64
 11  Purchase                    537577 non-null  int64  
dtypes: float64(2), int64(5), object(5)
memory usage: 49.2+ MB
```

We cannot delete all rows with null values as it will remove around 75% of the data. so, we are dropping the columns with the null values.

In [5]:

```
del df['Product_Category_2']
del df['Product_Category_3']
```

In [6]:

```
df.isnull().sum()
```

Out[6]:

```
User_ID                     0
Product_ID                  0
Gender                      0
Age                         0
Occupation                  0
City_Category               0
Stay_In_Current_City_Years  0
Marital_Status              0
Product_Category_1          0
Purchase                    0
dtype: int64
```

2. Analysing Columns :

2.1 How many number of unique values in each columns are there?

In [7]:

```
df['User_ID'].nunique()  # N unique returns number of unique.
```

Out[7]:

```
5891
```

In [8]:

```
df['Product_ID'].nunique()
```

Out[8]:

```
3623
```

In [9]:

```
df['Gender'].nunique()
```

Out[9]:

```
2
```

In [10]:

```
df['Age'].unique()
```

Out[10]:

```
array(['0-17', '55+', '26-35', '46-50', '51-55', '36-45', '18-25'],
      dtype=object)
```

In [11]:

```
df['Occupation'].unique()
```

Out[11]:

```
array([10, 16, 15,  7, 20,  9,  1, 12, 17,  0,  3,  4, 11,  8, 19,  2, 18,
        5, 14, 13,  6])
```

In [12]:

```
df['City_Category'].unique()
```

Out[12]:

```
array(['A', 'C', 'B'], dtype=object)
```

In [13]:

```
df['Stay_In_Current_City_Years'].unique()
```

Out[13]:

```
array(['2', '4+', '3', '1', '0'], dtype=object)
```

In [14]:

```
df['Marital_Status'].unique()
```

Out[14]:

```
array([0, 1])
```

In [15]:

```
df['Product_Category_1'].unique()
```

Out[15]:

```
array([ 3,  1, 12,  8,  5,  4,  2,  6, 14, 11, 13, 15,  7, 16, 18, 10, 17,
        9])
```

In [16]:

```
for column in df.columns[:-1]:
    print(df[column].nunique(),'\t: ',column )
```

```
5891    :  User_ID
3623    :  Product_ID
2       :  Gender
7       :  Age
21      :  Occupation
3       :  City_Category
5       :  Stay_In_Current_City_Years
2       :  Marital_Status
18      :  Product_Category_1
```

2.2 What are the average money spent by each user?

In [17]:

```
df['Purchase'].sum()/df['User_ID'].nunique()
```

Out[17]:

```
851751.5494822611
```

3. Analyzing Gender :

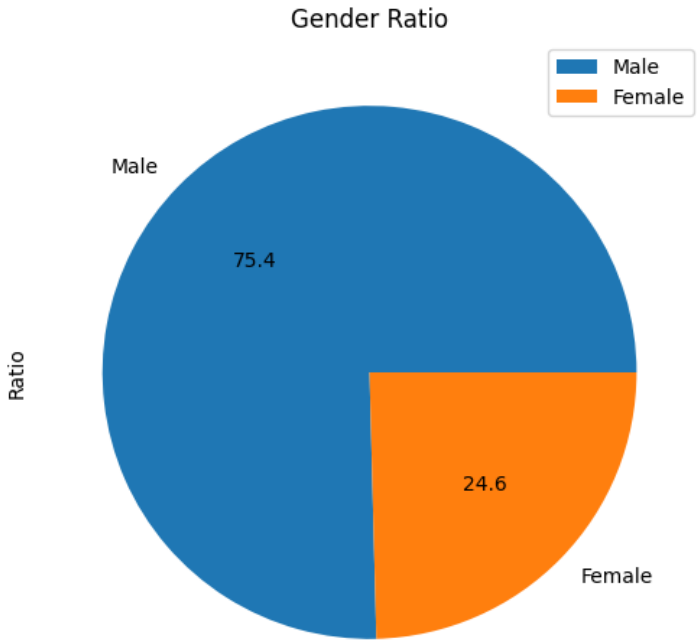
3.1 Naive approach to plot

In [18]:

```
data = pd.DataFrame({'Ratio' : [len(df[df['Gender'] == 'M']),len(df[df['Gender'] == 'F'])]},
                    index = ['Male', 'Female'])
data.plot.pie(y='Ratio',
              figsize = (6,6),
              autopct = "%.1f",
              title = 'Gender Ratio') # Autopct shows percentage in plot
```

Out[18]:

```
<Axes: title={'center': 'Gender Ratio'}, ylabel='Ratio'>
```



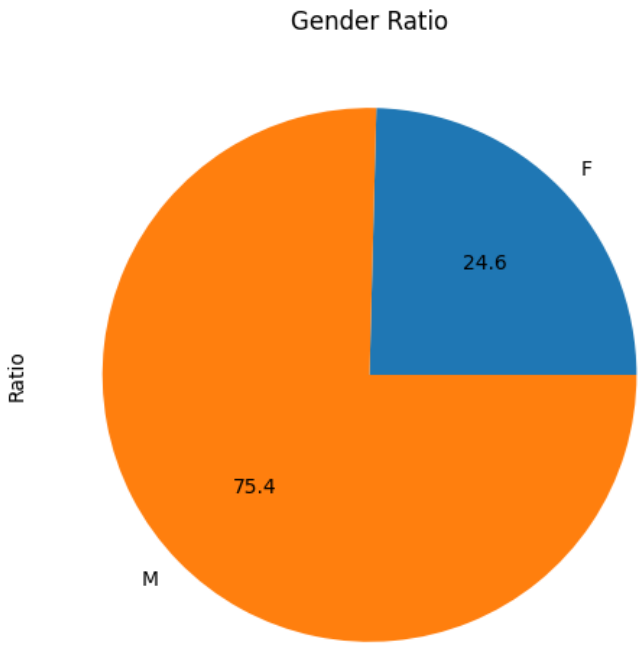
3.2 Second Approach using group by Function :

In [56]:

```
df.groupby('Gender').size().plot( kind = 'pie', # Size gives size of each category
                                  autopct = '%.1f',
                                  ylabel = 'Ratio',
                                  title = 'Gender Ratio',
                                  figsize = (6,6))
```

Out[56]:

```
<Axes: title={'center': 'Gender Ratio'}, ylabel='Ratio'>
```

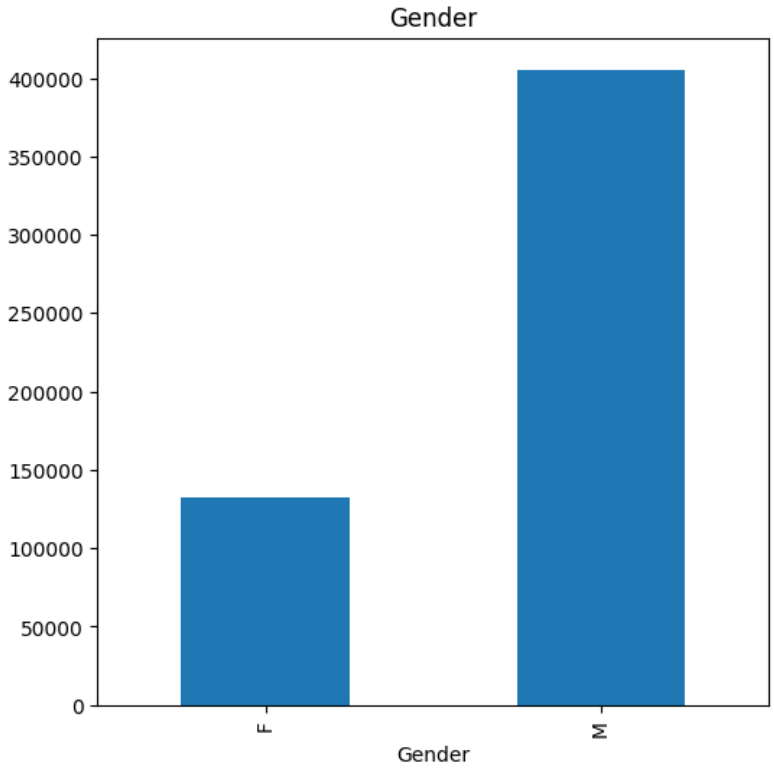


In [57]:

```
df.groupby('Gender').size().plot( kind = 'bar',
                                  title = 'Gender',
                                  figsize = (6,6)) # Size gives size of each category
```

Out[57]:

```
<Axes: title={'center': 'Gender'}, xlabel='Gender'>
```



3.3 How muc money they have spent?

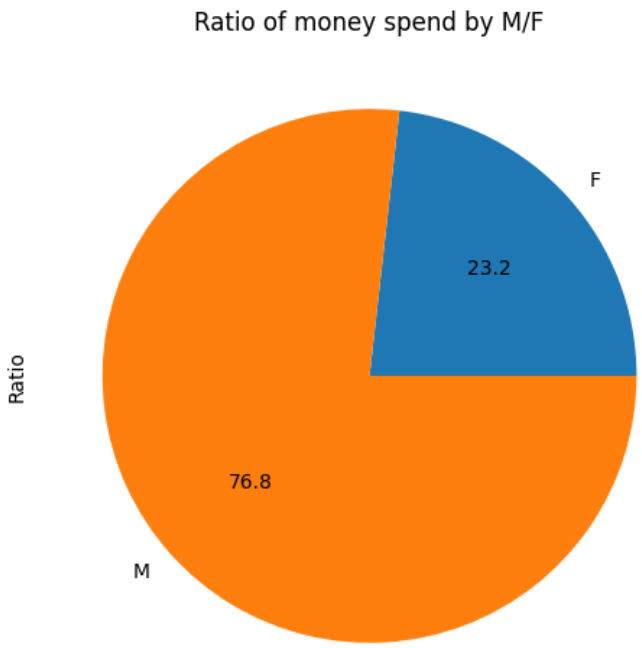
In [58]:

```
df.groupby('Gender').sum()['Purchase'].plot( kind = 'pie',
                                    autopct = '%.1f',
                                    ylabel = 'Ratio',
                                    title = 'Ratio of money spend by M/F',
                                    figsize = (6,6))
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/787149718.py:1: FutureWarning: The defa
ult value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('Gender').sum()['Purchase'].plot( kind = 'pie',
```

Out[58]:

```
<Axes: title={'center': 'Ratio of money spend by M/F'}, ylabel='Ratio'>
```



3.4 Who is spending more per purchase?

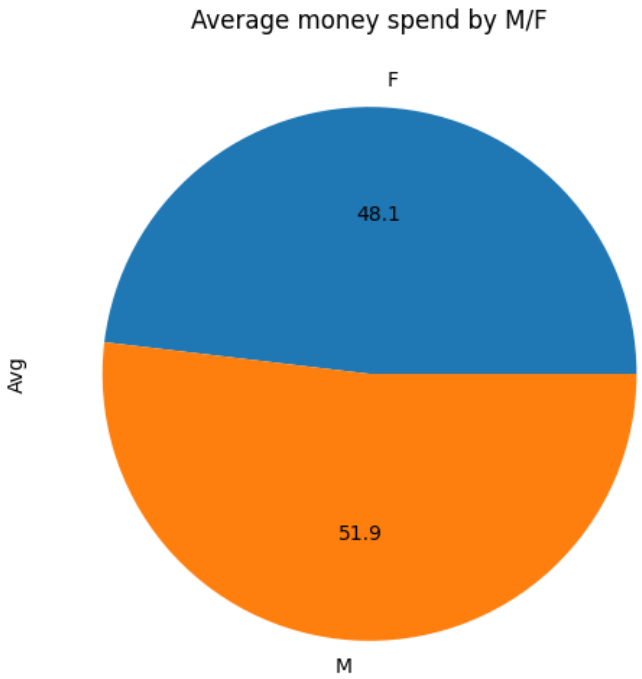
In [59]:

```
df.groupby('Gender').mean()['Purchase'].plot( kind = 'pie',
                                 autopct = '%.1f',
                                 ylabel = 'Avg',
                                 title = 'Average money spend by M/F',
                                 figsize = (6,6))
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/2188739655.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only wil
l default to False. Either specify numeric_only or select only columns which should be valid for the fun
ction.
  df.groupby('Gender').mean()['Purchase'].plot( kind = 'pie',
```

Out[59]:

```
<Axes: title={'center': 'Average money spend by M/F'}, ylabel='Avg'>
```



4. Analyzing age and marital status :

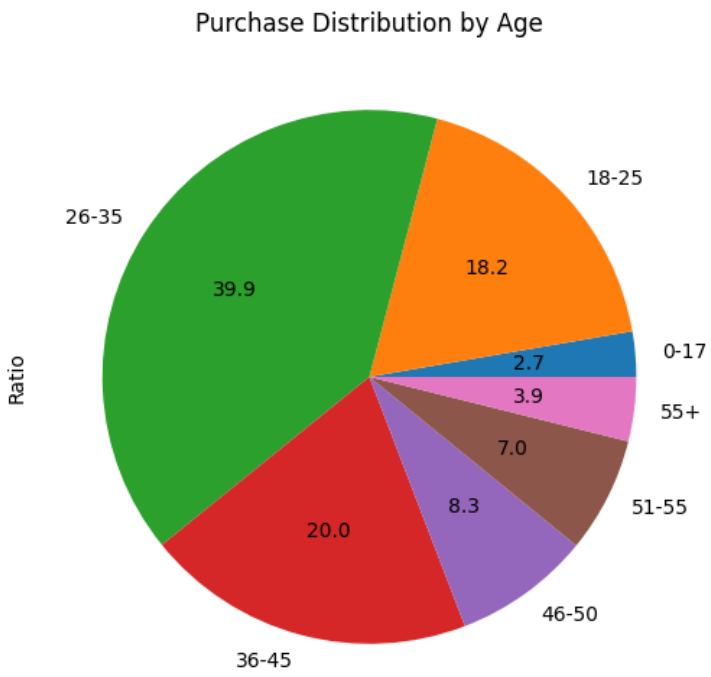
4.1 Which age group is making the most purchases?

In [60]:

```
df.groupby('Age').size().plot( kind = 'pie',
                               autopct = '%.1f',
                               ylabel = 'Ratio',
                               title = 'Purchase Distribution by Age',
                               figsize = (6,6))
```

Out[60]:

```
<Axes: title={'center': 'Purchase Distribution by Age'}, ylabel='Ratio'>
```

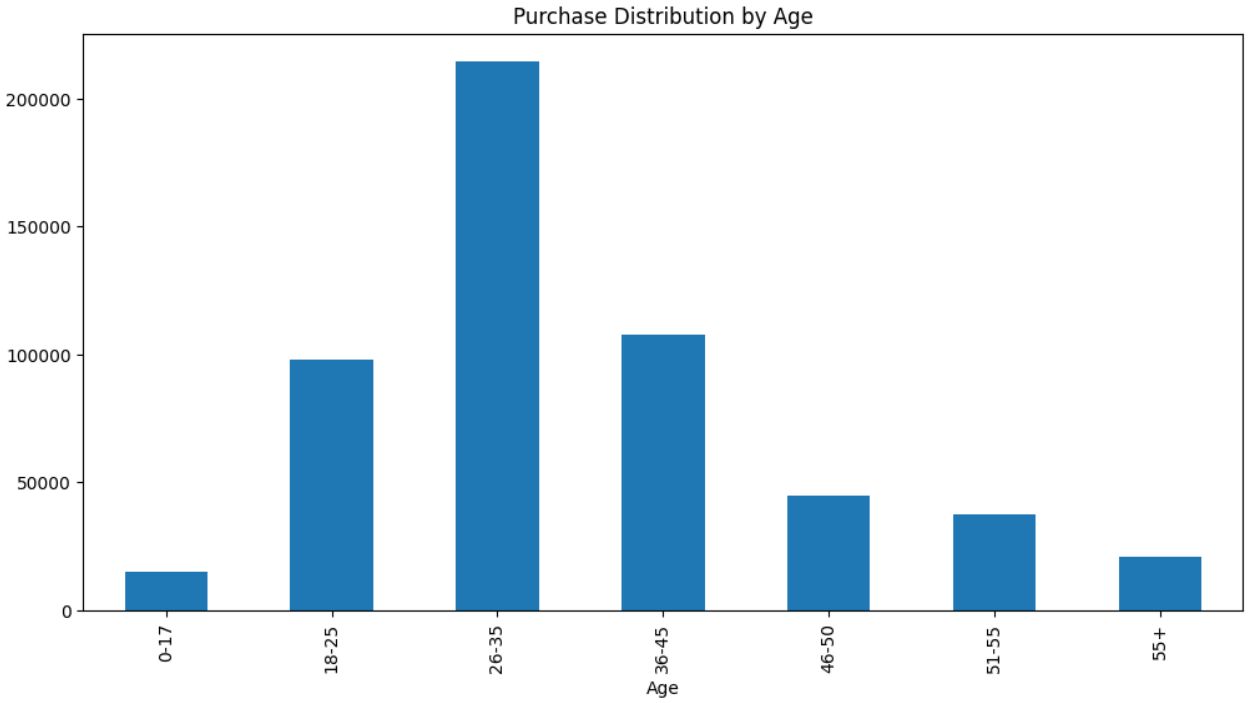


In [61]:

```
df.groupby('Age').size().plot( kind = 'bar',
                               title = 'Purchase Distribution by Age',
                               figsize = (12,6))
```

Out[61]:

```
<Axes: title={'center': 'Purchase Distribution by Age'}, xlabel='Age'>
```



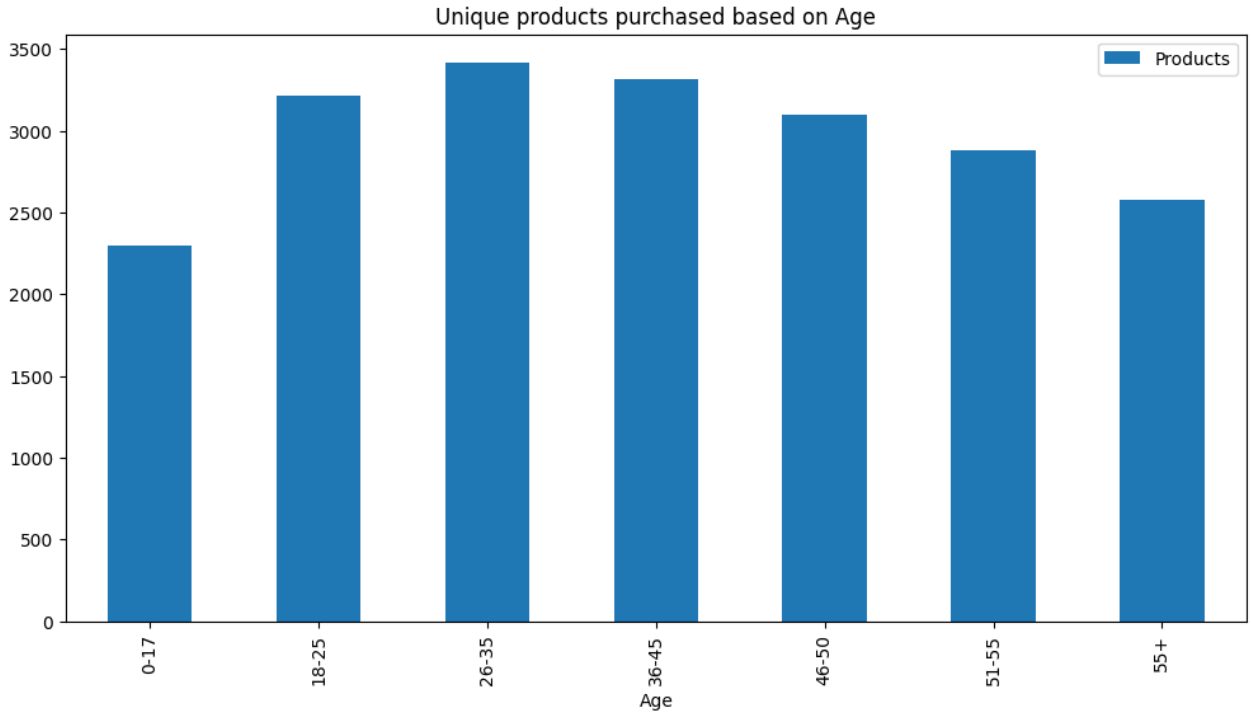
4.2 How many unique products people are purchasing?

In [72]:

```
lst = []

for i in df['Age'].unique():
    lst.append([i, df[df['Age']==i]['Product_ID'].nunique()])

df_pr = pd.DataFrame(lst, columns = ['Age', 'Products'])

df_pr.sort_values(by = 'Age', ascending=True).plot.bar(x = 'Age',
                                                       figsize = (12,6),
                                                       title = 'Unique products purchased based on Age')
```

Out[72]:

```
<Axes: title={'center': 'Unique products purchased based on Age'}, xlabel='Age'>
```



4.3 Which Age group is spending the most?

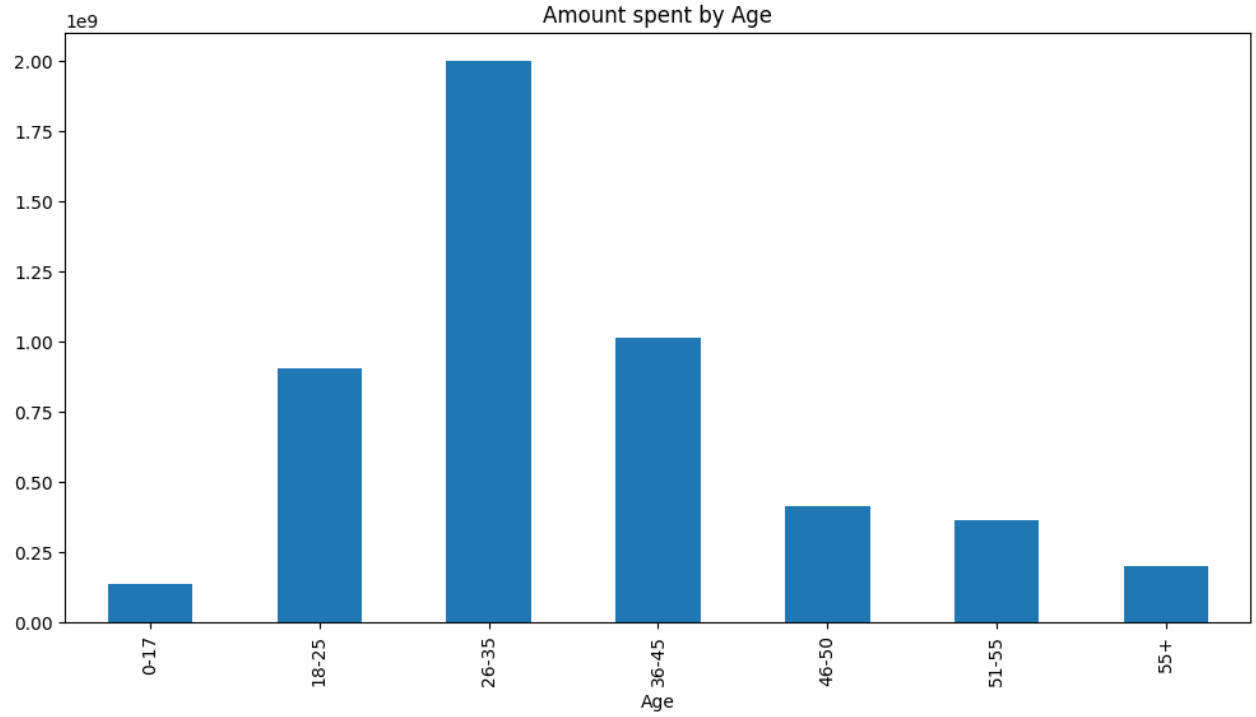
In [75]:

```
df.groupby('Age').sum()['Purchase'].plot( kind = 'bar',
                                title = 'Amount spent by Age',
                                 figsize = (12,6))
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/3554910632.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('Age').sum()['Purchase'].plot( kind = 'bar',
```

Out[75]:

```
<Axes: title={'center': 'Amount spent by Age'}, xlabel='Age'>
```



4.4 What is the average money spent by each age group per product?

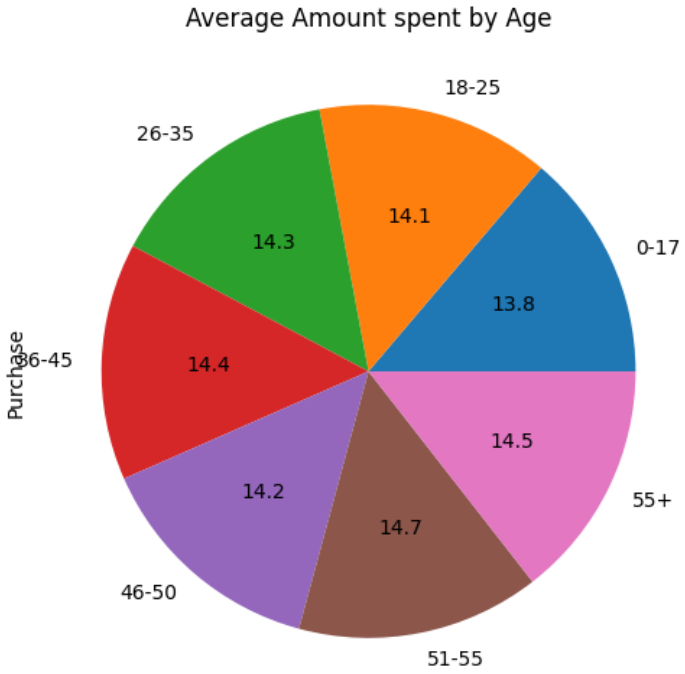
In [77]:

```
df.groupby('Age').mean()['Purchase'].plot( kind = 'pie',
                                title = 'Average Amount spent by Age',
                                 figsize = (12,6),
                                 autopct = '%.1f')
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/1867559634.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only wil
l default to False. Either specify numeric_only or select only columns which should be valid for the fun
ction.
  df.groupby('Age').mean()['Purchase'].plot( kind = 'pie',
```

Out[77]:

```
<Axes: title={'center': 'Average Amount spent by Age'}, ylabel='Purchase'>
```



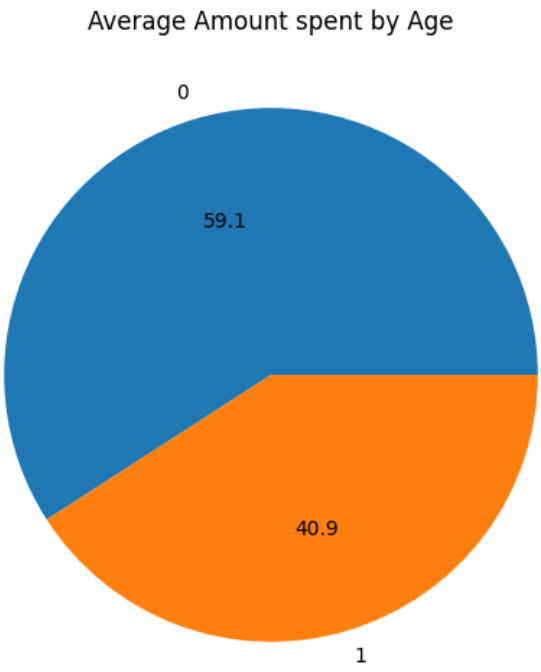
4.5 What is the ratio of married/unmarried peoples in dataset?

In [84]:

```
df.groupby('Marital_Status').size().plot( kind = 'pie',
                                 title = 'Married/Unmarried People',
                                  figsize = (12,6),
                                  autopct = '%.1f')
```

Out[84]:

```
<Axes: title={'center': 'Average Amount spent by Age'}>
```



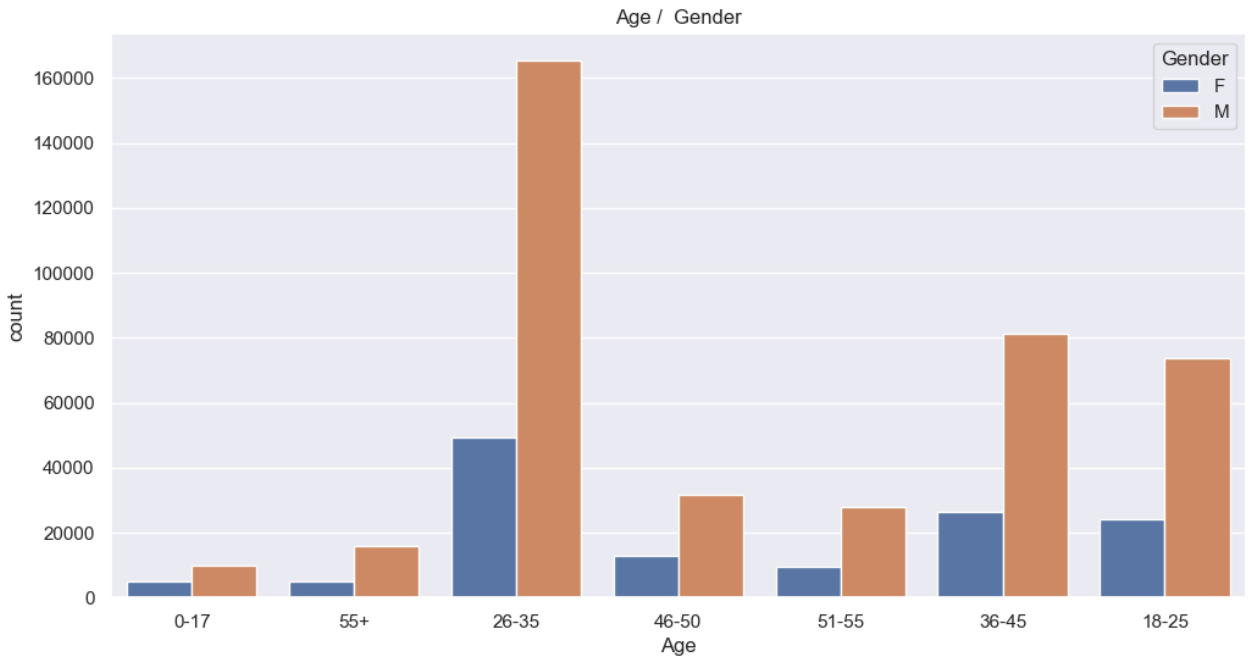
5. Analyzing Multiple columns together :

In [103]:

```
sns.set(rc = {'figure.figsize' : (12,6)})
sns.countplot(x = 'Age', hue = 'Gender', data = df).set_title('Age /  Gender')
```

Out[103]:

```
Text(0.5, 1.0, 'Age /  Gender')
```

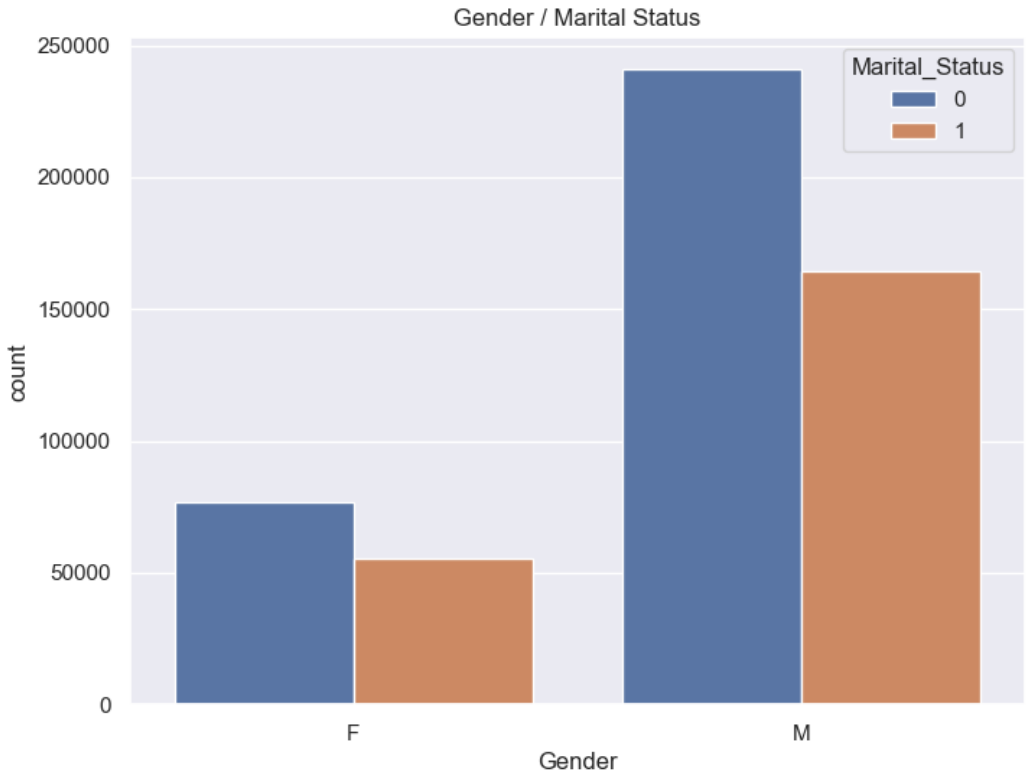


In [102]:

```
sns.set(rc = {'figure.figsize' : (8,6)})
sns.countplot(x = 'Gender', hue = 'Marital_Status', data = df).set_title('Gender / Marital Status')
```

Out[102]:

```
Text(0.5, 1.0, 'Gender / Marital Status')
```

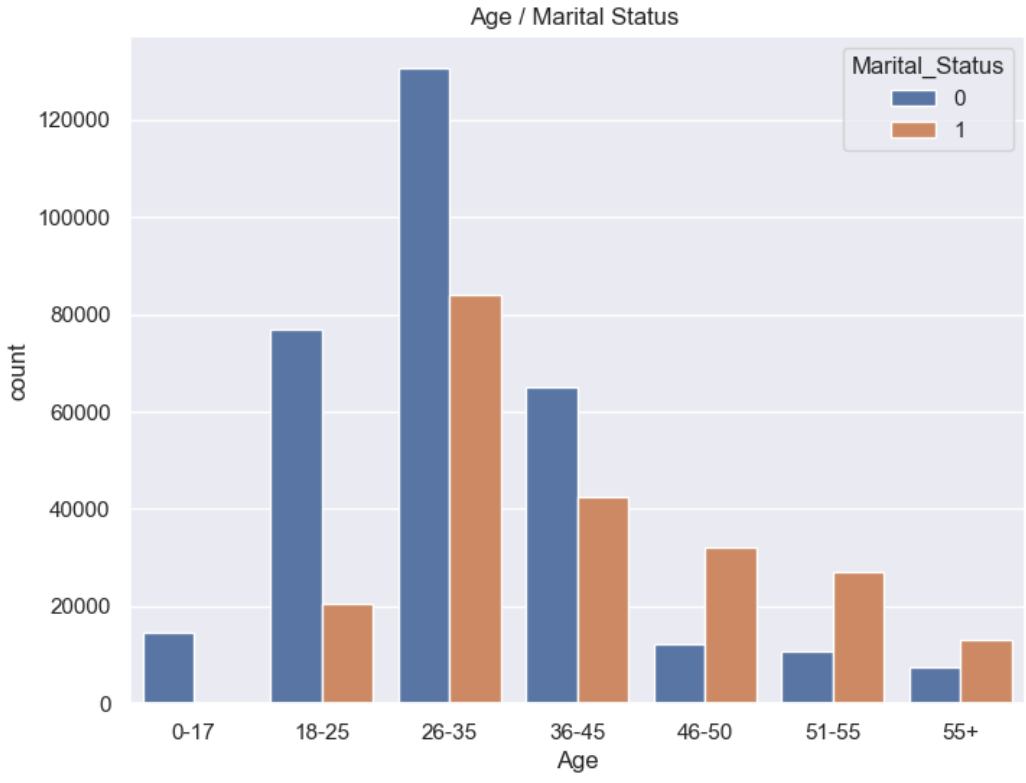


In [105]:

```
sns.set(rc = {'figure.figsize' : (8,6)})
sns.countplot(x = 'Age', hue = 'Marital_Status', data = df.sort_values(by='Age', ascending=True)).set_title('Age / Ma
```

Out[105]:

```
Text(0.5, 1.0, 'Age / Marital Status ')
```



In [110]:

```
sns.countplot(x = df['City_Category']).set_title('City Category')
```

Out[110]:

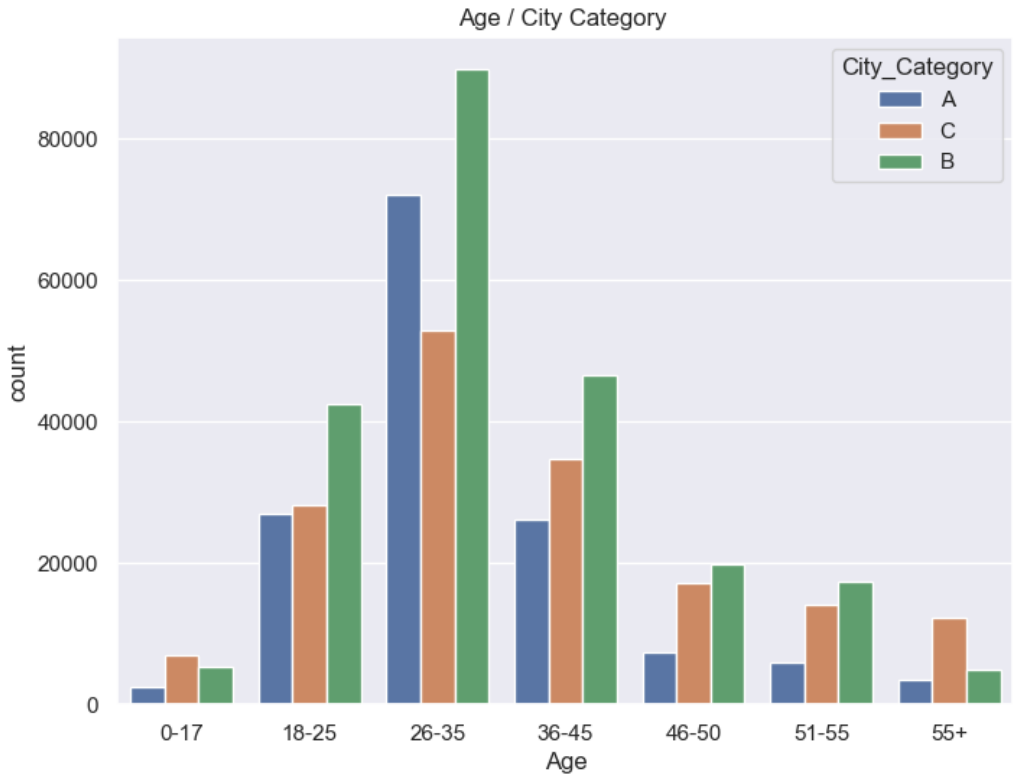
```
Text(0.5, 1.0, 'City Category')
```

In [109]:

```
sns.set(rc = {'figure.figsize' : (8,6)})
sns.countplot(x = 'Age',
              hue = 'City_Category',
              data = df.sort_values(by='Age', ascending=True)).set_title('Age / City Category ')
```

Out[109]:

```
Text(0.5, 1.0, 'Age / City Category ')
```

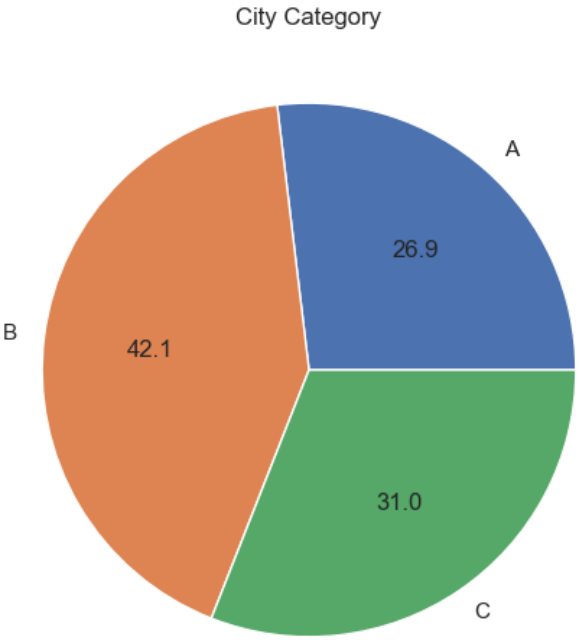


In [108]:

```
df.groupby('City_Category').size().plot(kind = 'pie', autopct = "%.1f", title = 'City Category')
```

Out[108]:

```
<Axes: title={'center': 'City Category'}>
```



In [112]:

```
sns.set(rc = {'figure.figsize' : (8,6)})
sns.countplot(x = 'City_Category',
              hue = 'Marital_Status',
              data = df).set_title('City Category / Marital_Status ')
```

Out[112]:

```
Text(0.5, 1.0, 'City Category / Marital_Status ')
```

In [113]:

```
sns.set(rc = {'figure.figsize' : (8,6)})
sns.countplot(x = 'City_Category',
              hue = 'Gender',
              data = df).set_title('City Category / Gender ')
```

Out[113]:

```
Text(0.5, 1.0, 'City Category / Gender ')
```

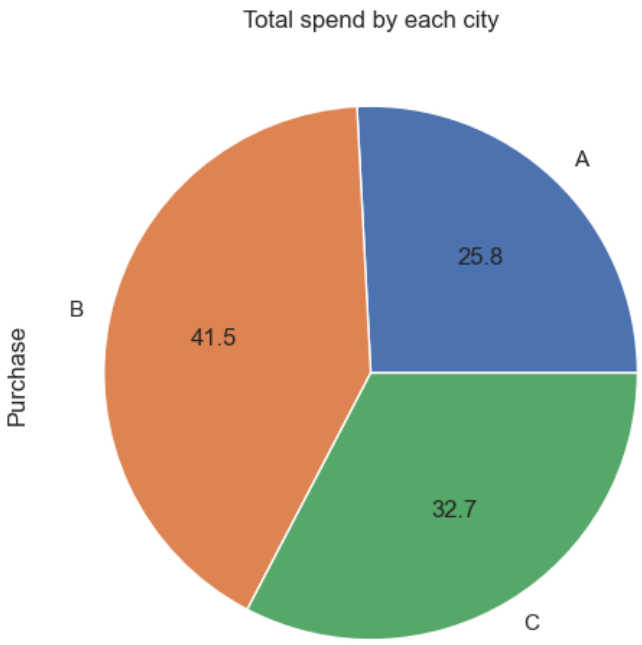
In [120]:

```
df.groupby('City_Category').sum()['Purchase'].plot.pie(autopct = '%.1f', title = 'Total spend by each city')
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/1344245533.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('City_Category').sum()['Purchase'].plot.pie(autopct = '%.1f', title = 'Total spend by each
city')
```

Out[120]:

```
<Axes: title={'center': 'Total spend by each city'}, ylabel='Purchase'>
```



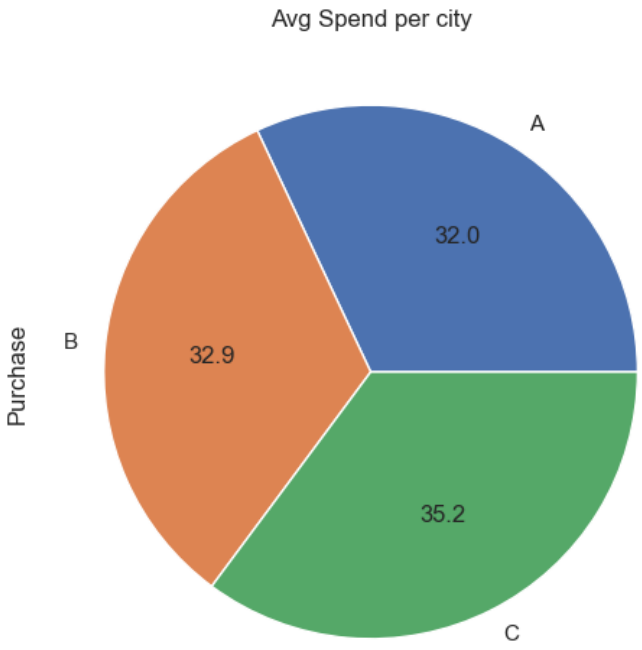
In [119]:

```
df.groupby('City_Category').mean()['Purchase'].plot.pie(autopct = '%.1f', title= 'Avg Spend per city')
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/791627196.py:1: FutureWarning: The defa
ult value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('City_Category').mean()['Purchase'].plot.pie(autopct = '%.1f', title= 'Avg Spend per city')
```

Out[119]:

```
<Axes: title={'center': 'Avg Spend per city'}, ylabel='Purchase'>
```



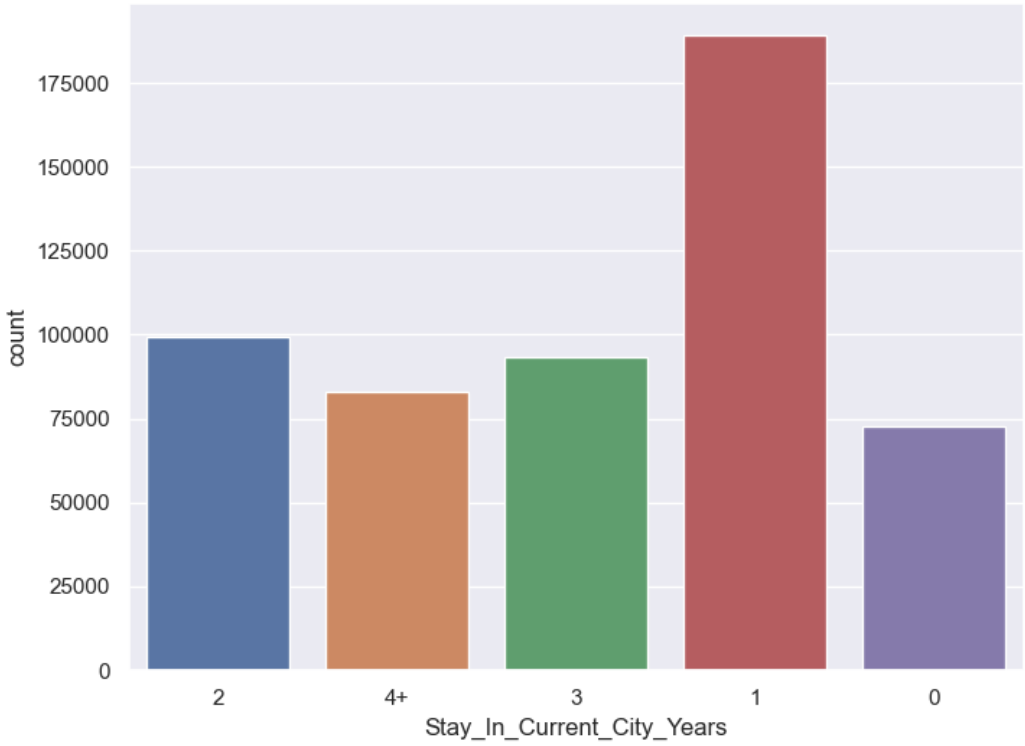
6. Occupation & Product Analysis :

In [121]:

```
sns.countplot(x = df['Stay_In_Current_City_Years'])
```

Out[121]:

```
<Axes: xlabel='Stay_In_Current_City_Years', ylabel='count'>
```

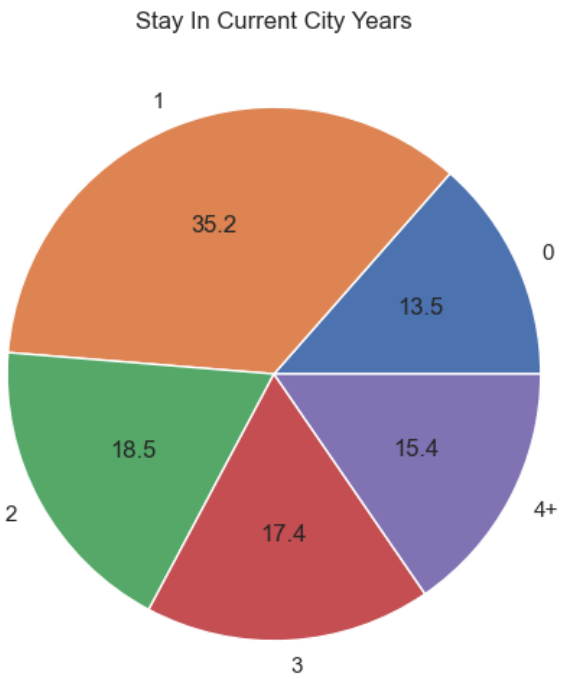


In [132]:

```
df.groupby('Stay_In_Current_City_Years').size().plot.pie(autopct = "%.1f", title = "Stay In Current City Years")
```

Out[132]:

```
<Axes: title={'center': 'Stay In Current City Years'}>
```

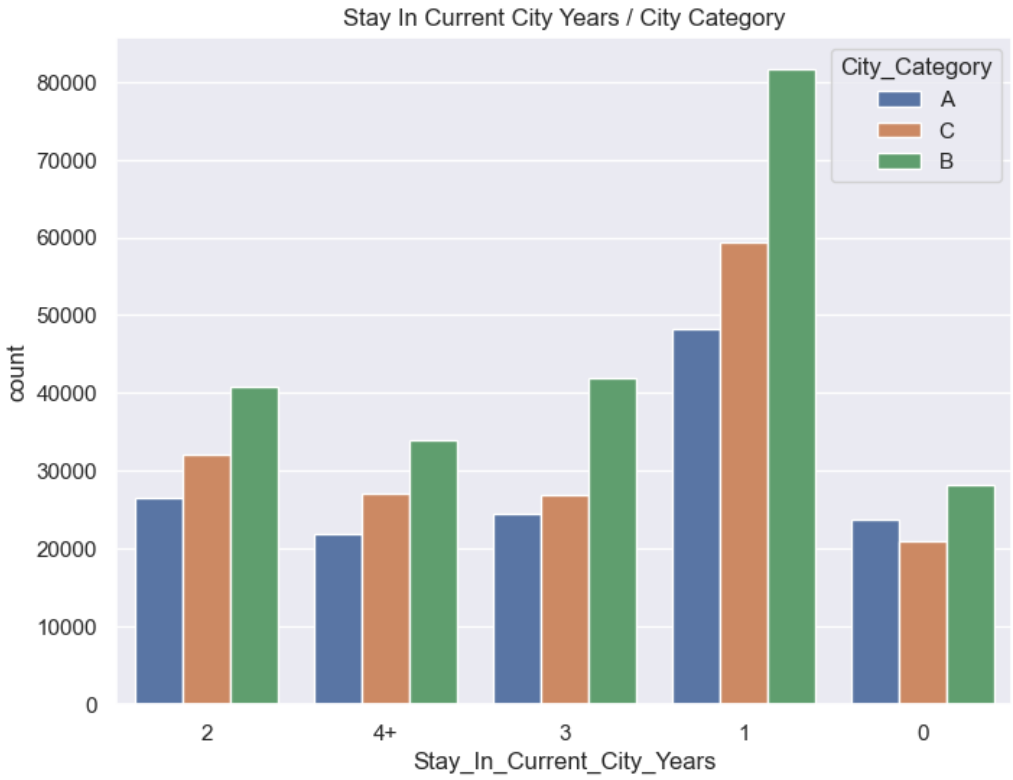


In [130]:

```
sns.countplot(x = 'Stay_In_Current_City_Years',
              hue = 'City_Category', data = df).set_title ("Stay In Current City Years / City Category")
```

Out[130]:

```
Text(0.5, 1.0, 'Stay In Current City Years / City Category')
```



In [128]:

```
sns.countplot(x = 'Stay_In_Current_City_Years',
              hue = 'Gender', data = df).set_title ("Stay In Current City Years / Gender")
```

Out[128]:

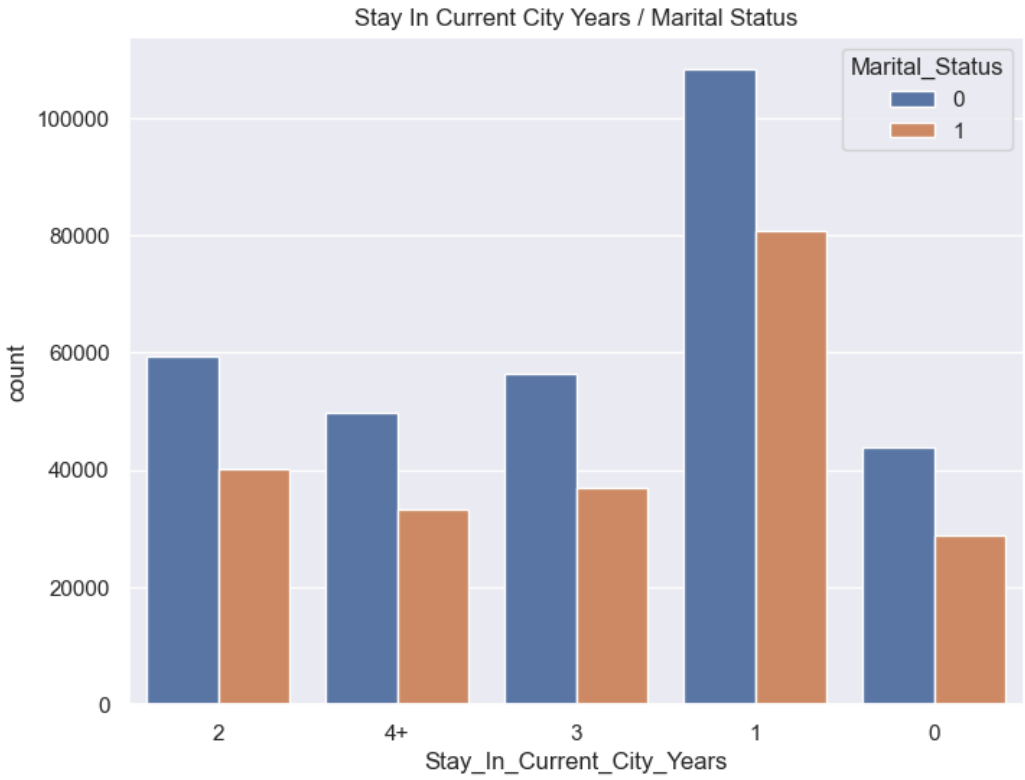
```
Text(0.5, 1.0, 'Stay In Current City Years / Gender')
```

In [129]:

```
sns.countplot(x = 'Stay_In_Current_City_Years',
              hue = 'Marital_Status', data = df).set_title ("Stay In Current City Years / Marital Status")
```

Out[129]:

```
Text(0.5, 1.0, 'Stay In Current City Years / Marital Status')
```



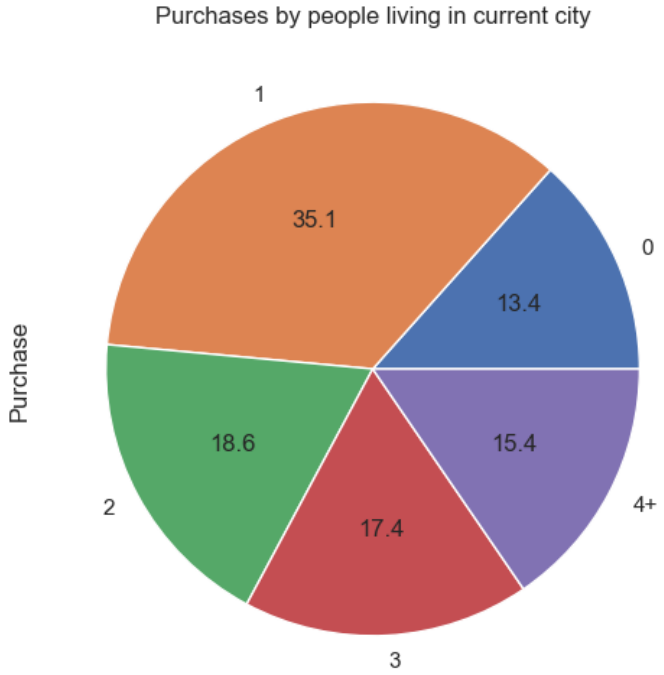
In [134]:

```
df.groupby('Stay_In_Current_City_Years').sum()['Purchase'].plot.pie(autopct = "%.1f",
                                                                    title = "Purchases by people living in current ci
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/1095909450.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('Stay_In_Current_City_Years').sum()['Purchase'].plot.pie(autopct = "%.1f",
```

Out[134]:

```
<Axes: title={'center': 'Purchases by people living in current city'}, ylabel='Purchase'>
```



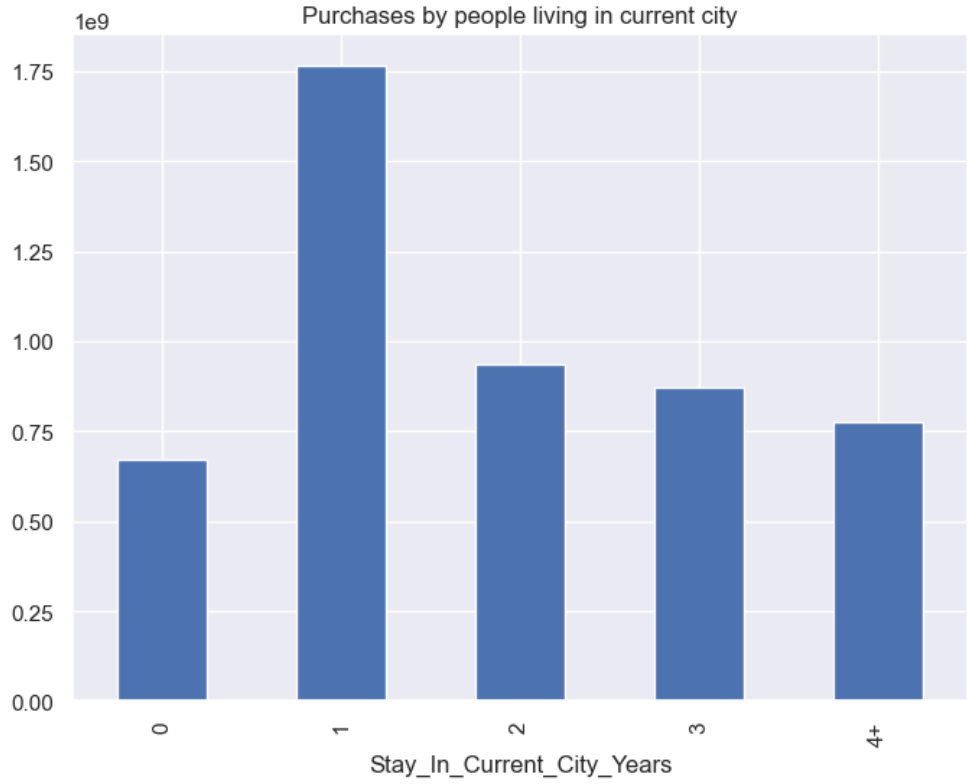
In [137]:

```
df.groupby('Stay_In_Current_City_Years').sum()['Purchase'].plot(kind = 'bar',
                                                                title = "Purchases by people living in current ci
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/2494234089.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('Stay_In_Current_City_Years').sum()['Purchase'].plot(kind = 'bar',
```

Out[137]:

```
<Axes: title={'center': 'Purchases by people living in current city'}, xlabel='Stay_In_Current_City_Year
s'>
```



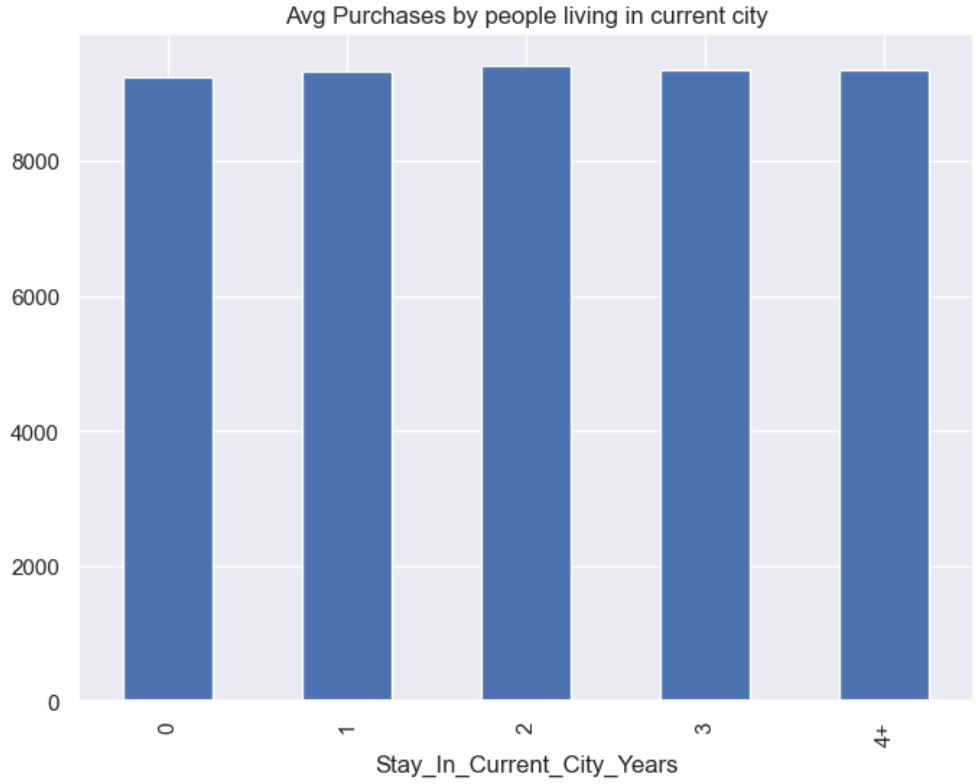
In [138]:

```
df.groupby('Stay_In_Current_City_Years').mean()['Purchase'].plot(kind = 'bar',
                                                                 title = "Avg Purchases by people living in curren
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/2878320080.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only wil
l default to False. Either specify numeric_only or select only columns which should be valid for the fun
ction.
  df.groupby('Stay_In_Current_City_Years').mean()['Purchase'].plot(kind = 'bar',
```

Out[138]:

```
<Axes: title={'center': 'Avg Purchases by people living in current city'}, xlabel='Stay_In_Current_City_
Years'>
```



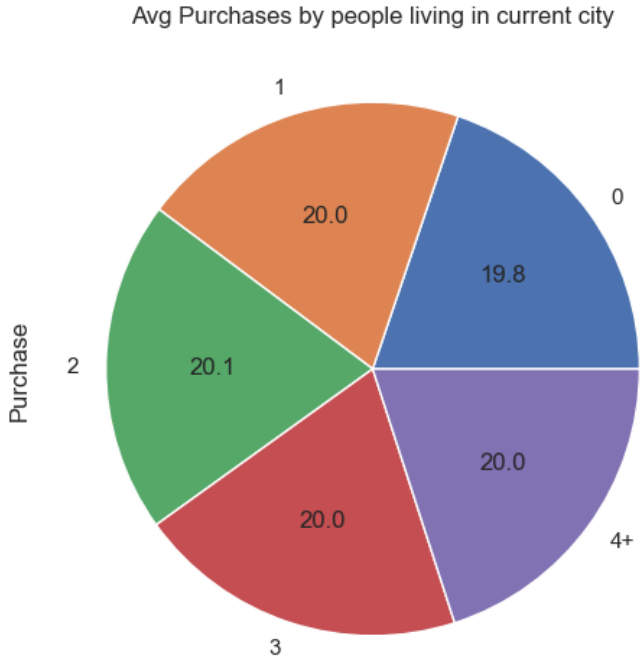
In [139]:

```
.groupby('Stay_In_Current_City_Years').mean()['Purchase'].plot.pie(autopct = "%.1f",
                                                                   title = "Avg Purchases by people living in current
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/939171007.py:1: FutureWarning: The defa
ult value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('Stay_In_Current_City_Years').mean()['Purchase'].plot.pie(autopct = "%.1f",
```

Out[139]:

```
<Axes: title={'center': 'Avg Purchases by people living in current city'}, ylabel='Purchase'>
```

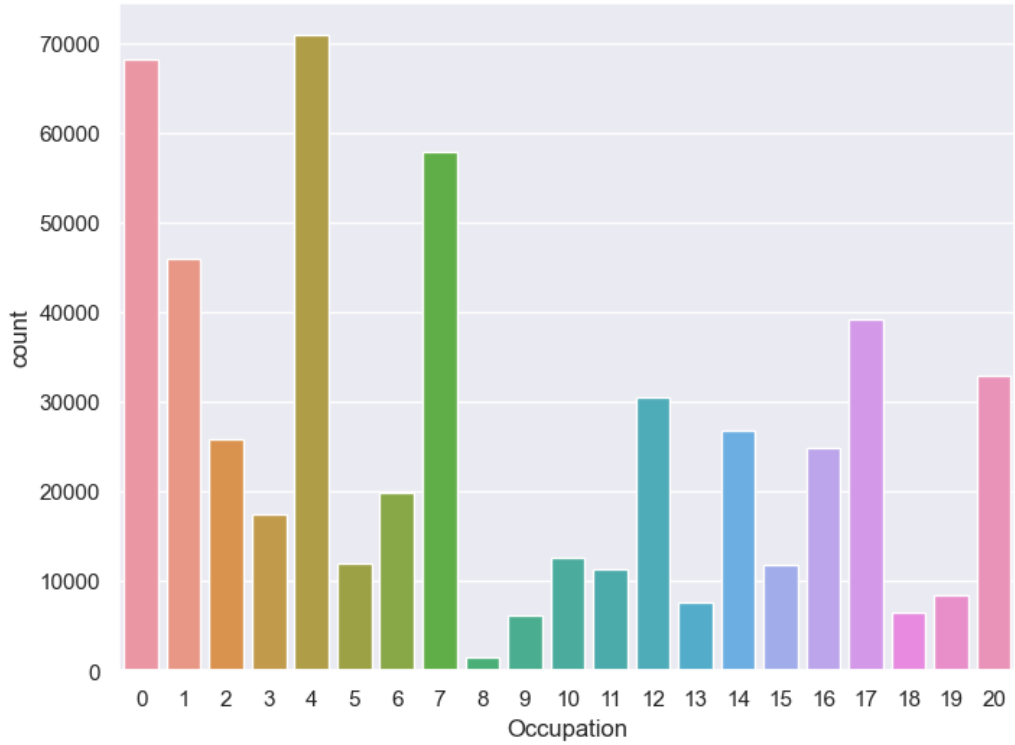


In [140]:

```
sns.countplot(x = df['Occupation'])
```

Out[140]:

```
<Axes: xlabel='Occupation', ylabel='count'>
```

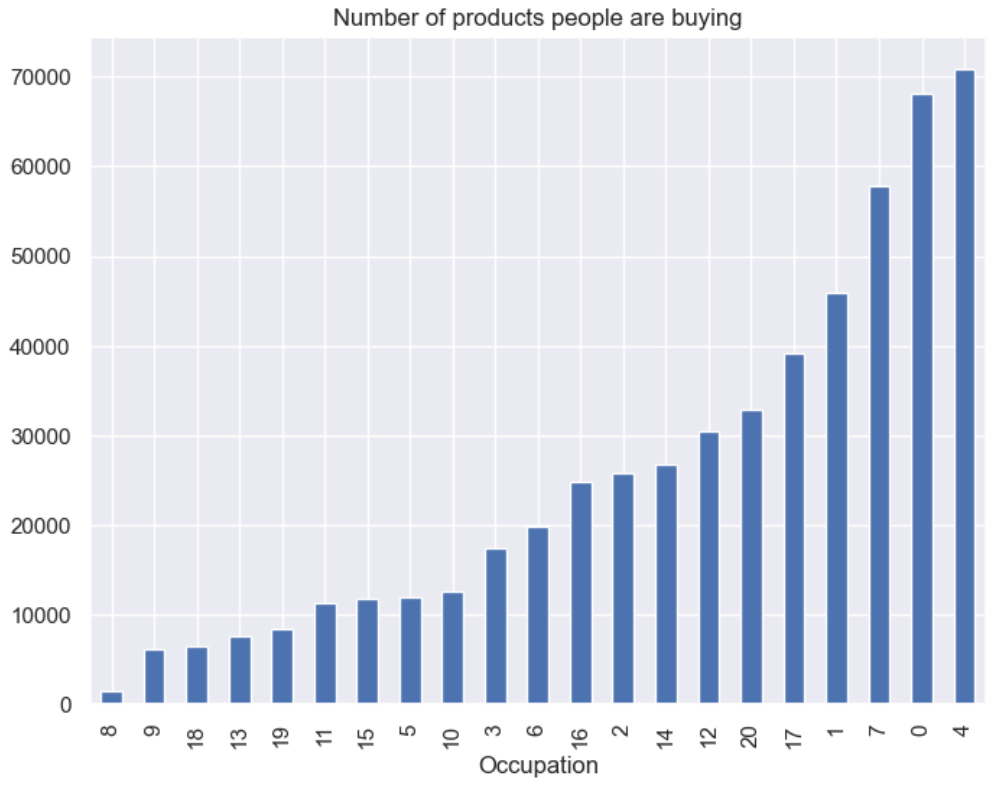


In [149]:

```
df.groupby('Occupation').size().sort_values().plot(kind = 'bar',
                                                   title= 'Number of products people are buying')
```

Out[149]:

```
<Axes: title={'center': 'Number of products people are buying'}, xlabel='Occupation'>
```



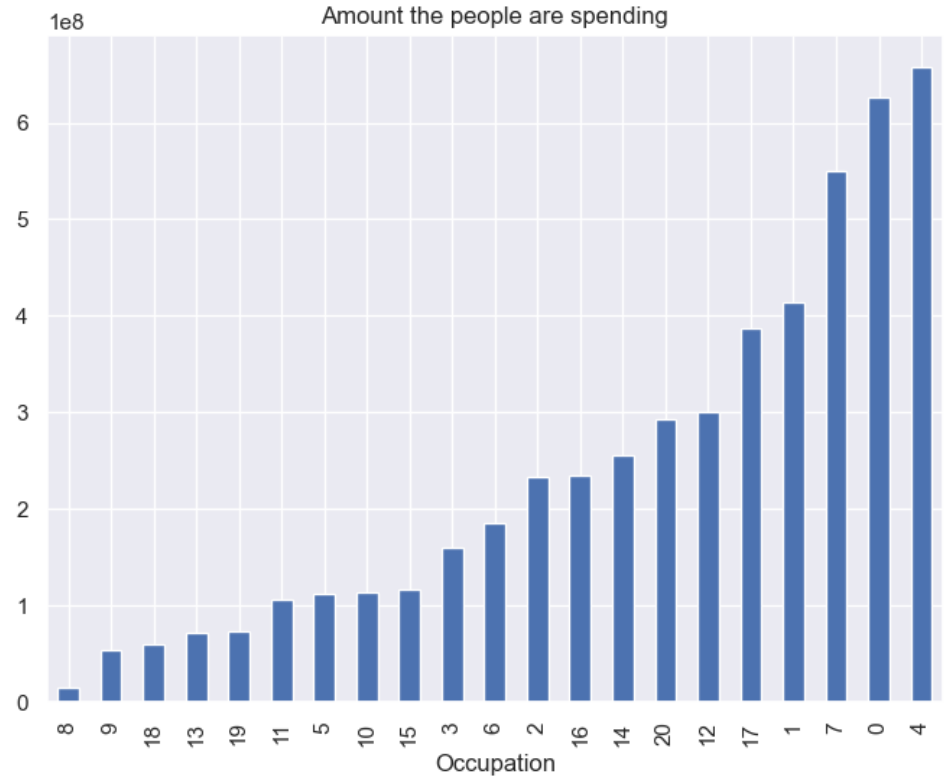
In [147]:

```
df.groupby('Occupation').sum()['Purchase'].sort_values().plot(kind = 'bar',
                                                              title = 'Amount the people are spending')
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/1585062034.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('Occupation').sum()['Purchase'].sort_values().plot(kind = 'bar', title = 'Amount the people
are spending')
```

Out[147]:

```
<Axes: title={'center': 'Amount the people are spending'}, xlabel='Occupation'>
```



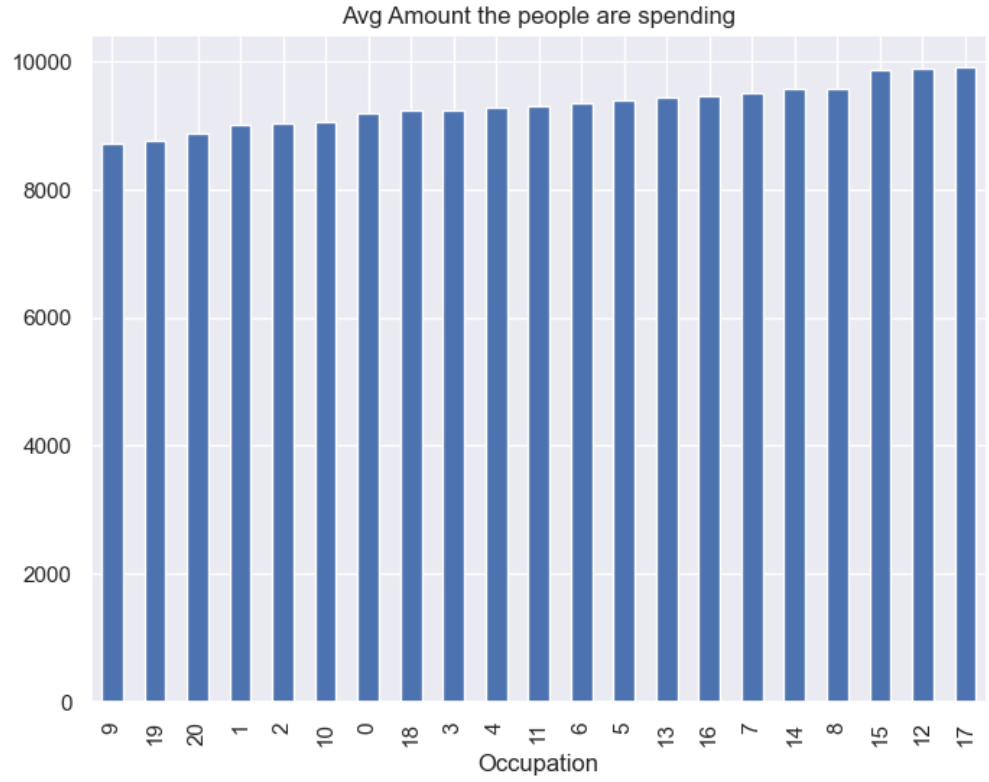
In [151]:

```
df.groupby('Occupation').mean()['Purchase'].sort_values().plot(kind = 'bar',
                                                               title = 'Avg Amount the people are spending')
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/2022356456.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only wil
l default to False. Either specify numeric_only or select only columns which should be valid for the fun
ction.
  df.groupby('Occupation').mean()['Purchase'].sort_values().plot(kind = 'bar', title = 'Avg Amount the p
eople are spending')
```

Out[151]:

```
<Axes: title={'center': 'Avg Amount the people are spending'}, xlabel='Occupation'>
```

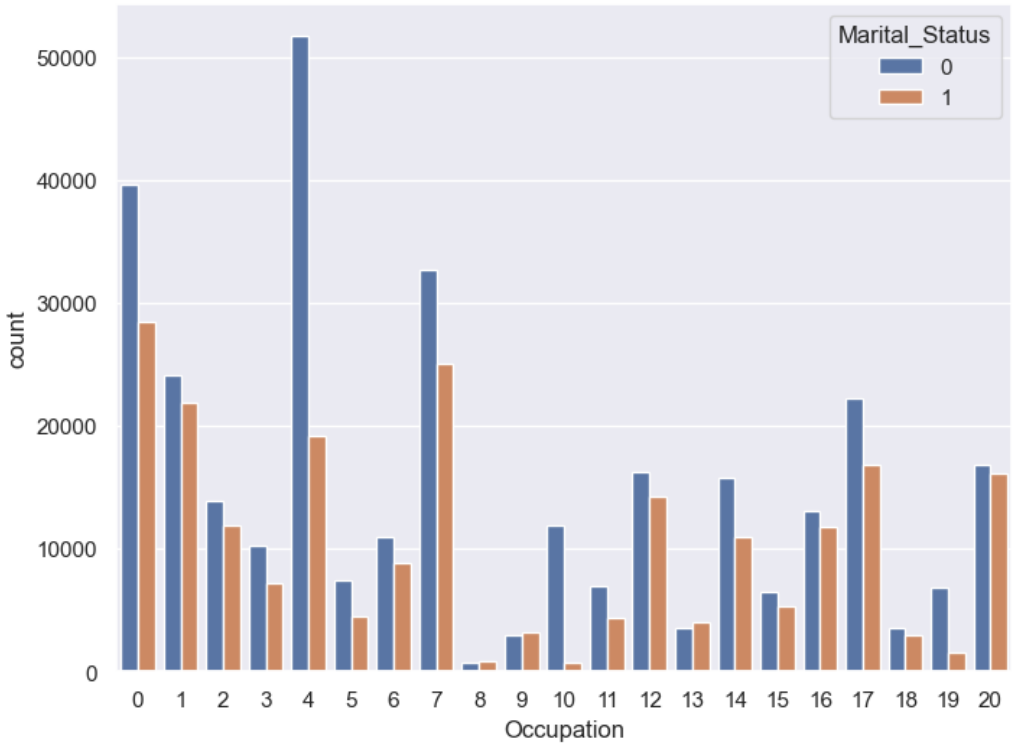


In [152]:

```
sns.countplot(x = 'Occupation', hue = 'Marital_Status', data = df)
```

Out[152]:

```
<Axes: xlabel='Occupation', ylabel='count'>
```

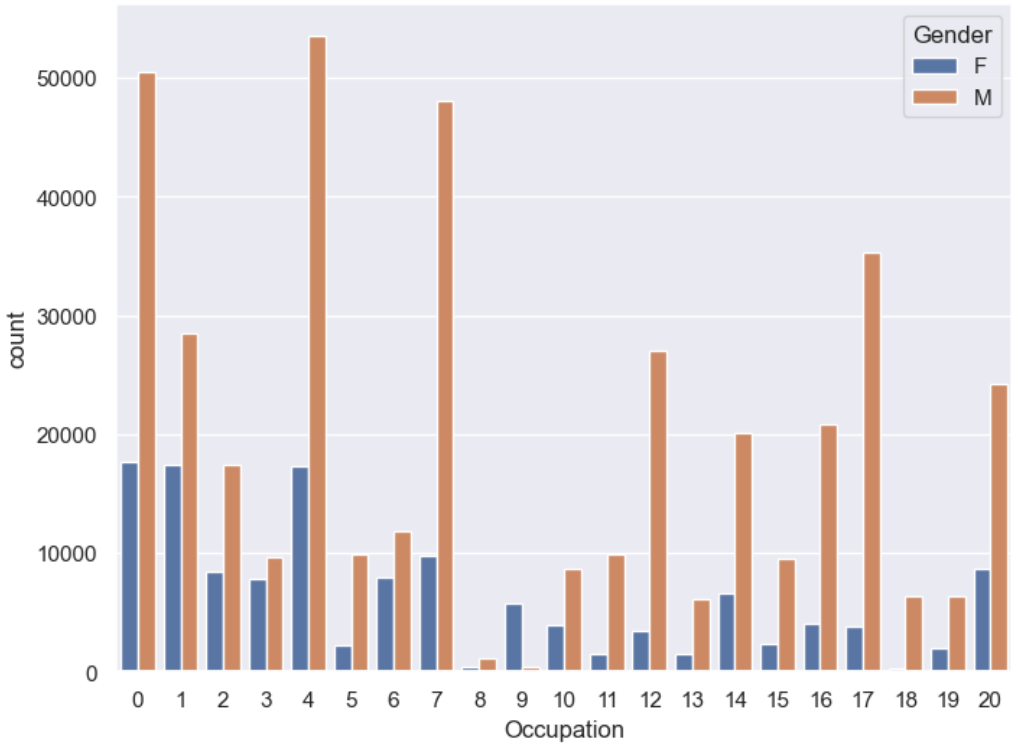


In [153]:

```
sns.countplot(x = 'Occupation', hue = 'Gender', data = df)
```

Out[153]:

```
<Axes: xlabel='Occupation', ylabel='count'>
```

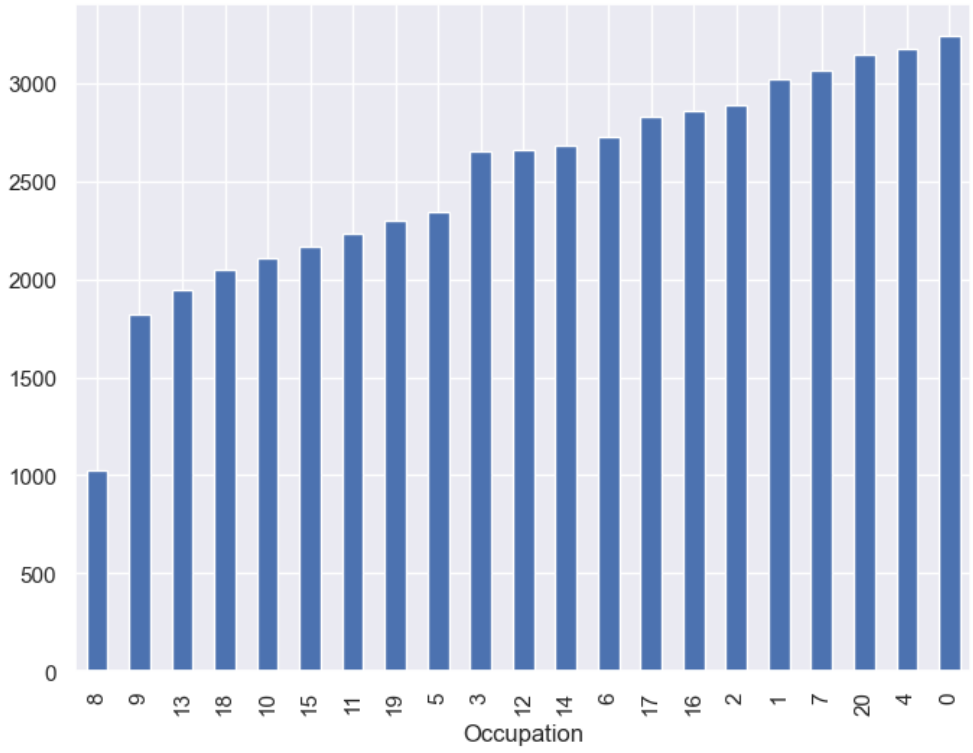


In [155]:

```
df.groupby('Occupation').nunique()['Product_ID'].sort_values().plot(kind = 'bar')
```

Out[155]:

```
<Axes: xlabel='Occupation'>
```

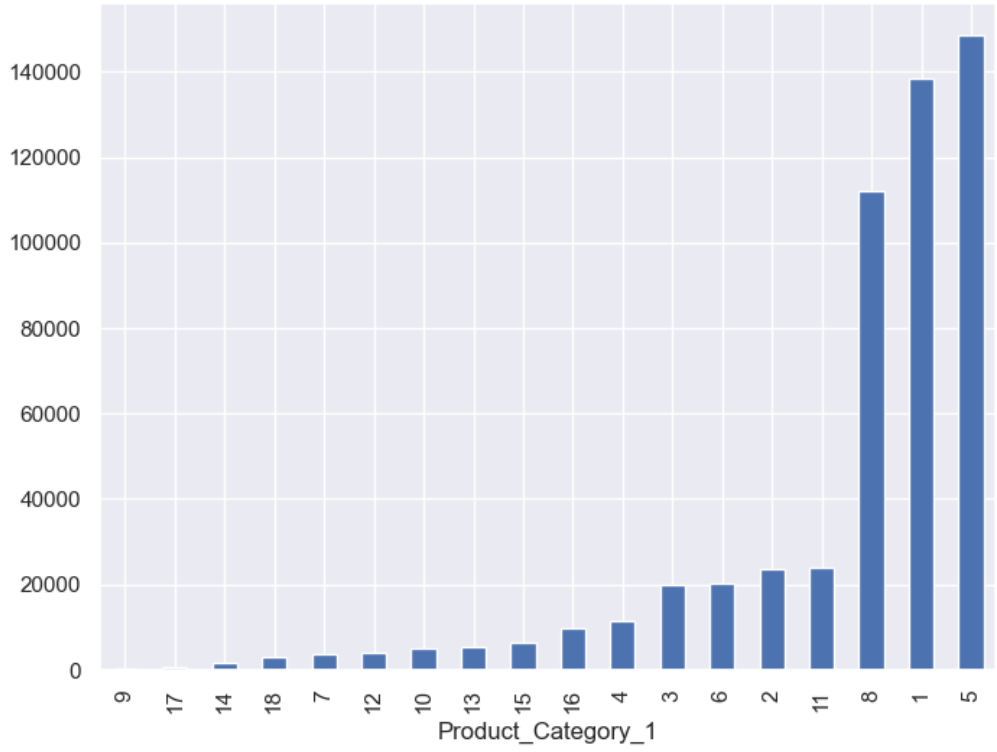


In [157]:

```
df.groupby('Product_Category_1').size().sort_values().plot(kind = 'bar')
```

Out[157]:

```
<Axes: xlabel='Product_Category_1'>
```



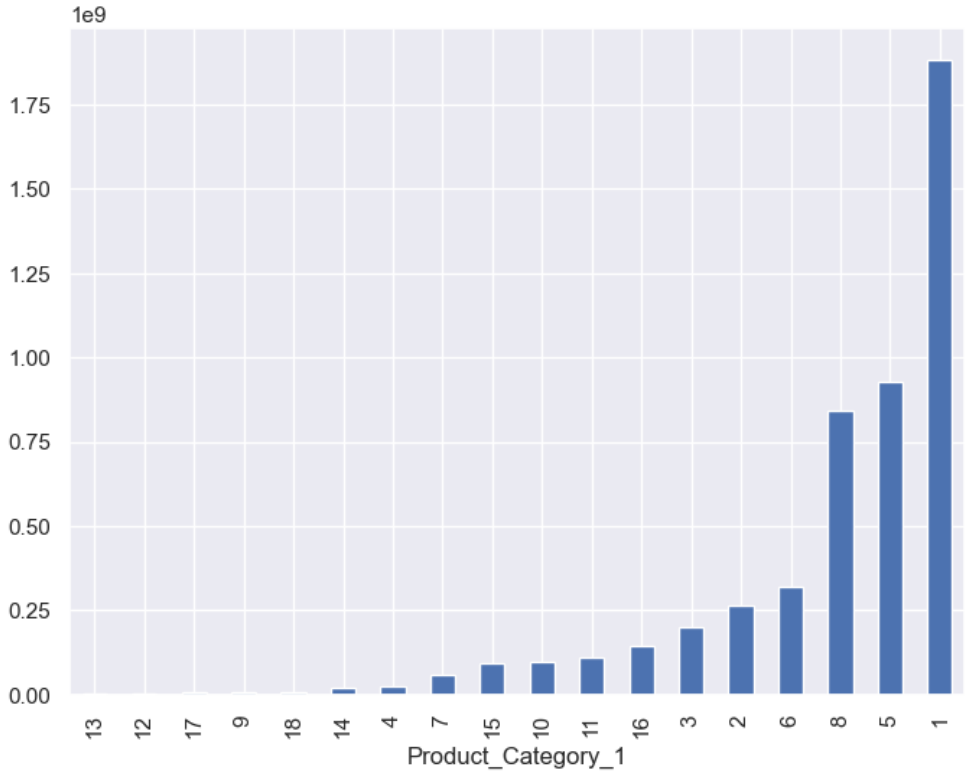
In [158]:

```
df.groupby('Product_Category_1').sum()['Purchase'].sort_values().plot(kind = 'bar')
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/2703638129.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('Product_Category_1').sum()['Purchase'].sort_values().plot(kind = 'bar')
```

Out[158]:

```
<Axes: xlabel='Product_Category_1'>
```



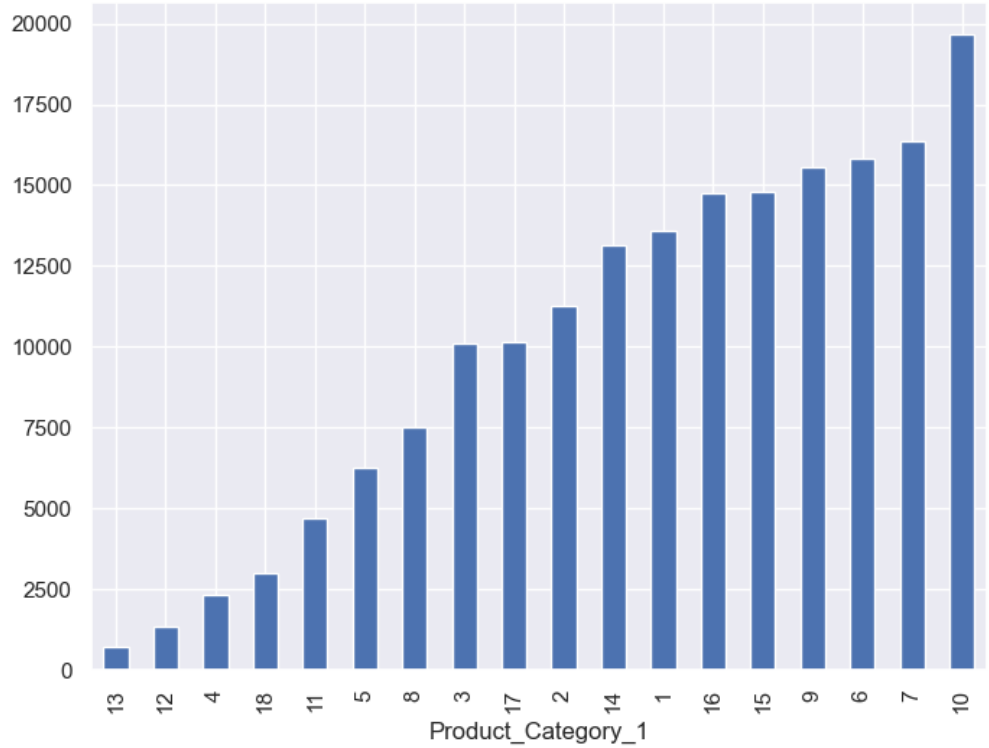
In [159]:

```
df.groupby('Product_Category_1').mean()['Purchase'].sort_values().plot(kind = 'bar')
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/3011761601.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only wil
l default to False. Either specify numeric_only or select only columns which should be valid for the fun
ction.
  df.groupby('Product_Category_1').mean()['Purchase'].sort_values().plot(kind = 'bar')
```

Out[159]:

```
<Axes: xlabel='Product_Category_1'>
```



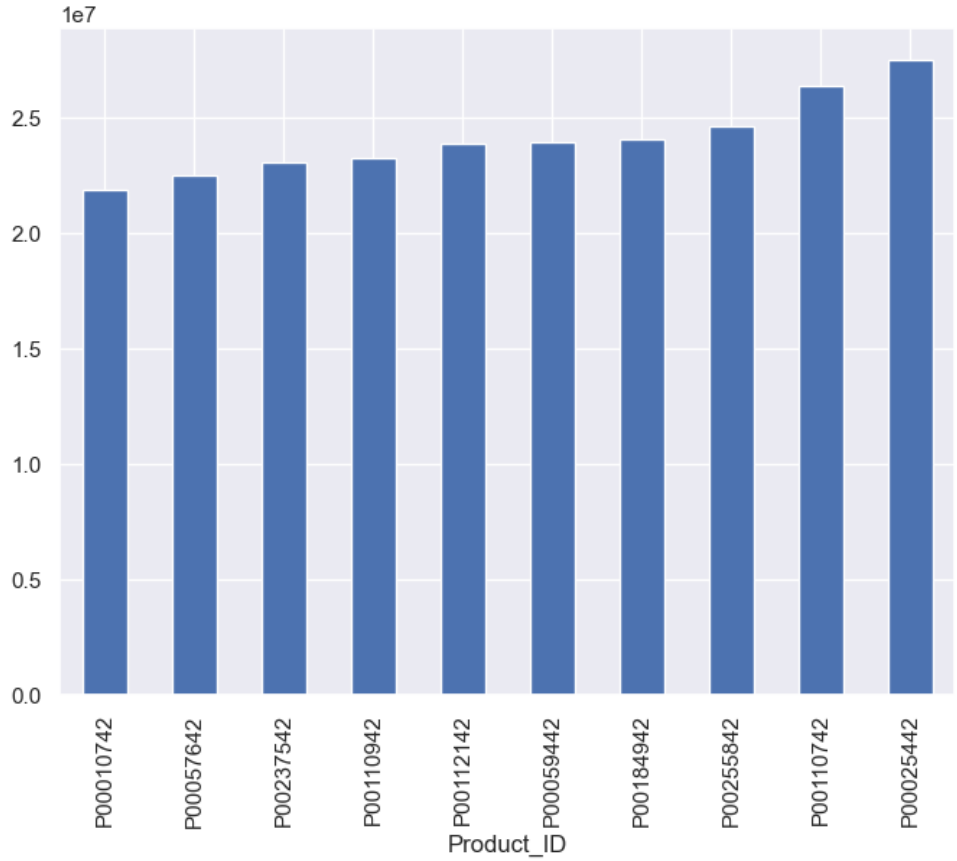
In [160]:

```
df.groupby('Product_ID').sum()['Purchase'].nlargest(10).sort_values().plot(kind = 'bar')
# Nlargest return largest n results
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/2828224466.py:1: FutureWarning: The def
ault value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('Product_ID').sum()['Purchase'].nlargest(10).sort_values().plot(kind = 'bar') # Nlargest re
turn largest n results
```

Out[160]:

```
<Axes: xlabel='Product_ID'>
```

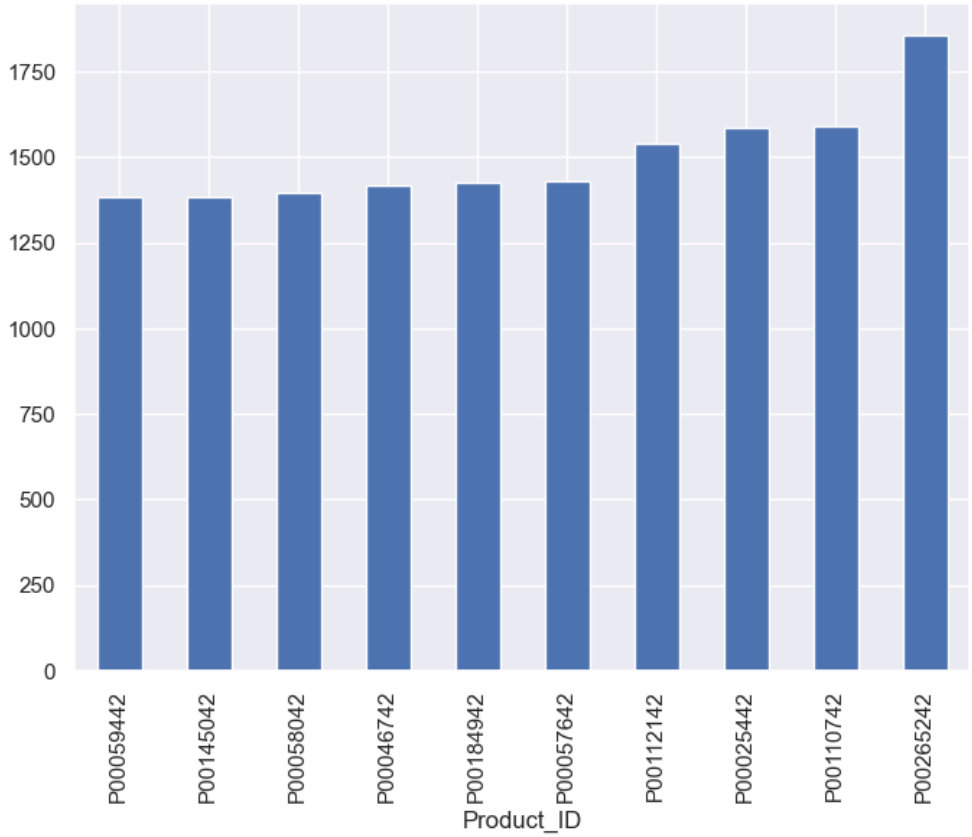


In [161]:

```
df.groupby('Product_ID').size().nlargest(10).sort_values().plot(kind = 'bar')
```

Out[161]:

```
<Axes: xlabel='Product_ID'>
```



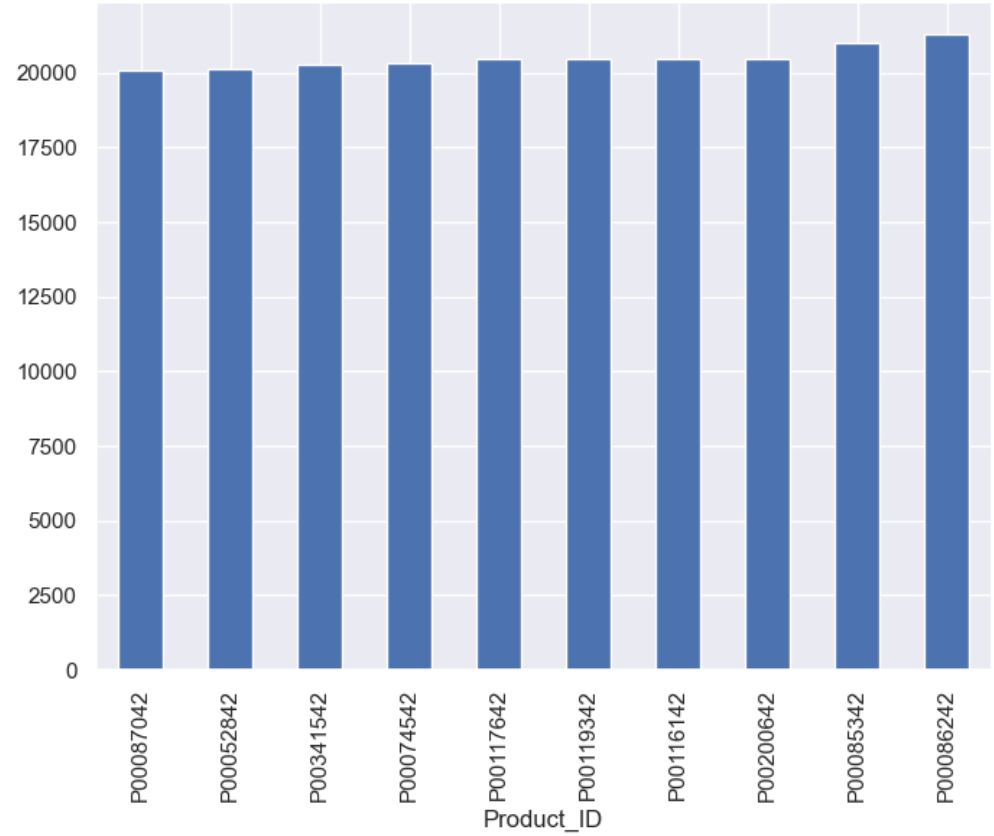
In [162]:

```
df.groupby('Product_ID').mean()['Purchase'].nlargest(10).sort_values().plot(kind = 'bar')
```

```
/var/folders/6w/mv1nv9w5131g49jj6l9ppgbh0000gp/T/ipykernel_55992/485863399.py:1: FutureWarning: The defa
ult value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only will
default to False. Either specify numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('Product_ID').mean()['Purchase'].nlargest(10).sort_values().plot(kind = 'bar')
```

Out[162]:

```
<Axes: xlabel='Product_ID'>
```

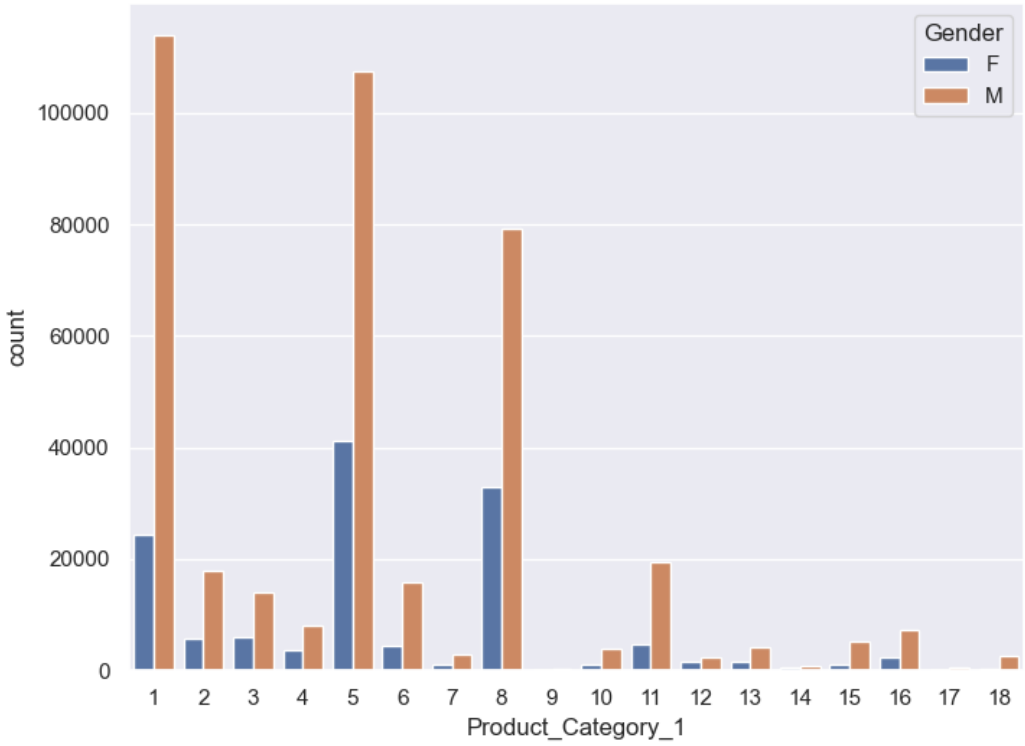


In [163]:

```
sns.countplot(x = 'Product_Category_1', hue = 'Gender', data = df)
```

Out[163]:

```
<Axes: xlabel='Product_Category_1', ylabel='count'>
```

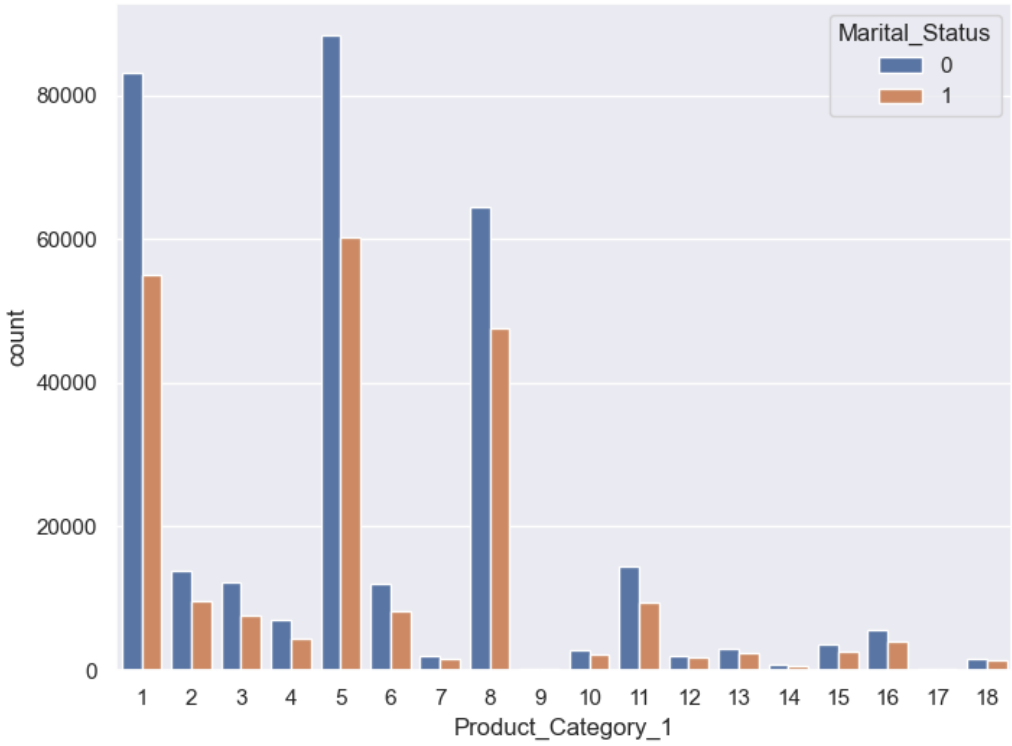


In [164]:

```
sns.countplot(x = 'Product_Category_1', hue = 'Marital_Status', data = df)
```

Out[164]:

```
<Axes: xlabel='Product_Category_1', ylabel='count'>
```



7. Combining Gender and Marital Status

In [19]:

```
l = []

for i in range(len(df)):
    l.append(df['Gender'][i] + "_" + str(df['Marital_Status'][i]))

df['Marital_Gender'] = l

df.head()
```

Out[19]:

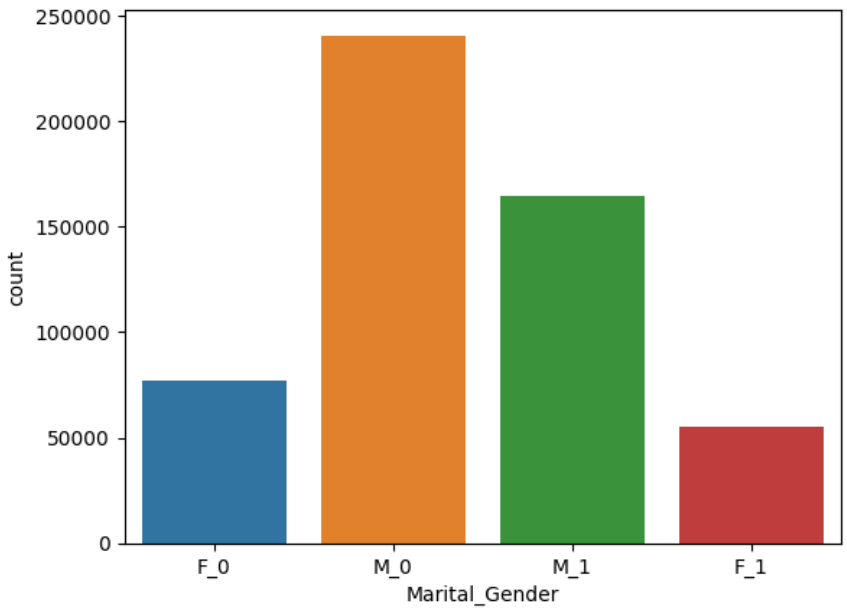
	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category_1	Purchase	M
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370	
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200	
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422	
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057	
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969	

In [21]:

```
sns.countplot(x = df['Marital_Gender'])
```

Out[21]:

```
<Axes: xlabel='Marital_Gender', ylabel='count'>
```

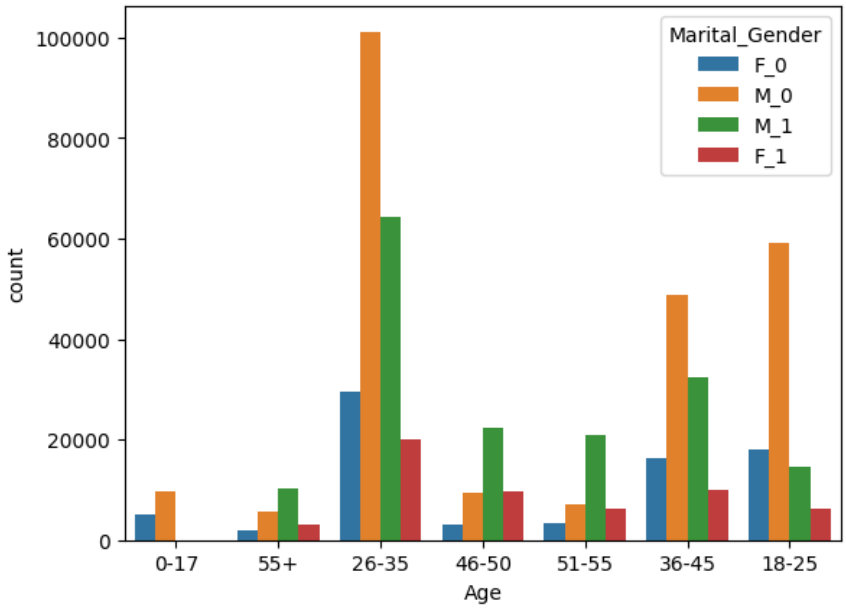


In [22]:

```
sns.countplot(x = df['Age'], hue = df['Marital_Gender'], data = df )
```

Out[22]:

```
<Axes: xlabel='Age', ylabel='count'>
```

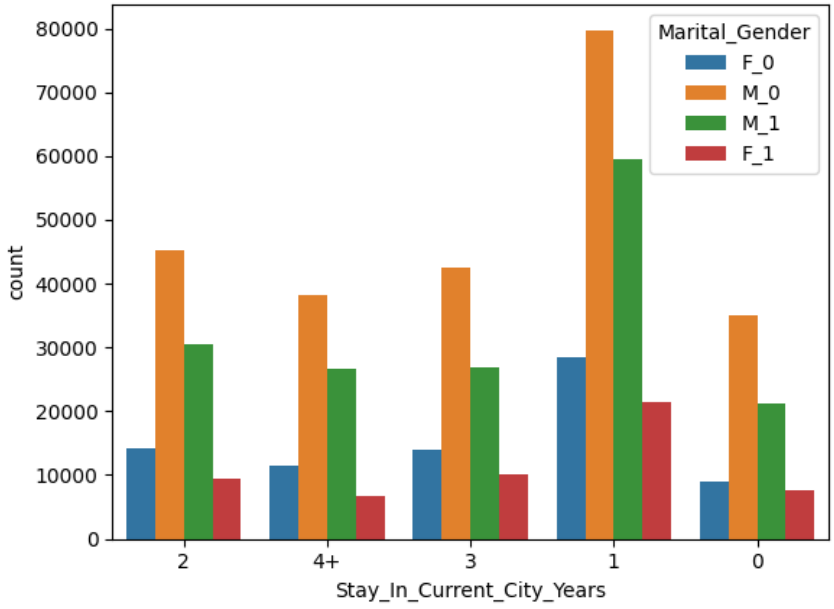


In [23]:

```
sns.countplot(x = df['Stay_In_Current_City_Years'], hue = df['Marital_Gender'], data = df )
```

Out[23]:

```
<Axes: xlabel='Stay_In_Current_City_Years', ylabel='count'>
```

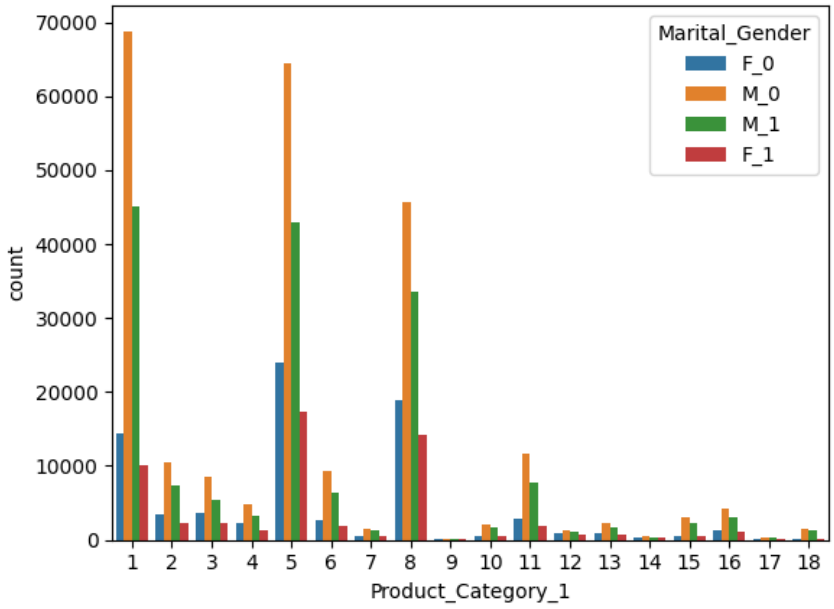


In [24]:

```
sns.countplot(x = df['Product_Category_1'], hue = df['Marital_Gender'], data = df )
```

Out[24]:

```
<Axes: xlabel='Product_Category_1', ylabel='count'>
```

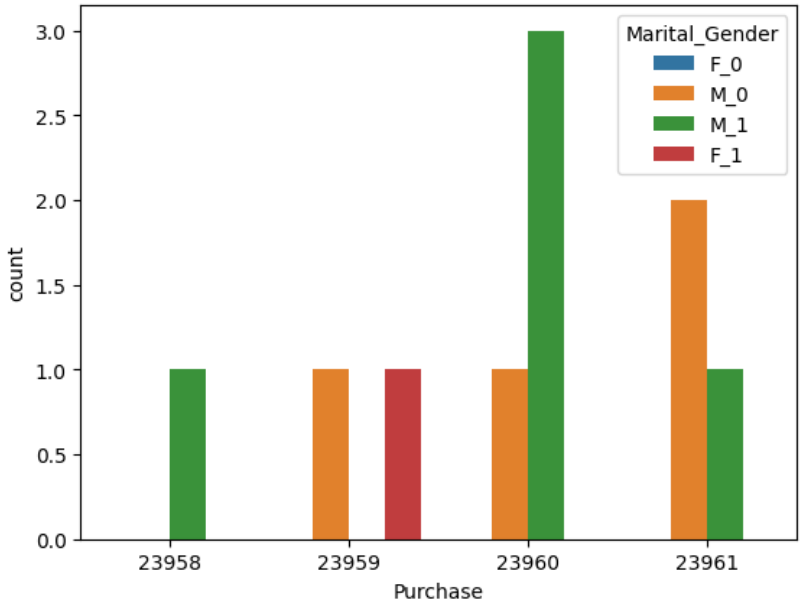


In [26]:

```
sns.countplot(x = df['Purchase'].nlargest(10), hue = df['Marital_Gender'], data = df )
```

Out[26]:

```
<Axes: xlabel='Purchase', ylabel='count'>
```

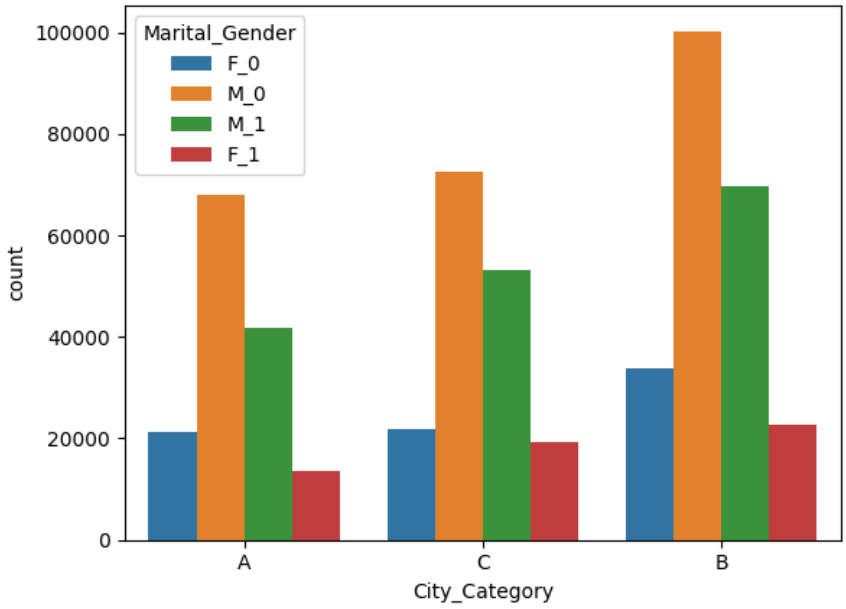


In [27]:

```
sns.countplot(x = df['City_Category'], hue = df['Marital_Gender'], data = df )
```

Out[27]:

```
<Axes: xlabel='City_Category', ylabel='count'>
```



In [ ]: