In [1]:
```
import pandas as pd
```

In [4]:
```
customers = pd.read_csv("customer_table.csv", delimiter=',')
items = pd.read_csv("items_sold_table.csv",  delimiter=',')
```

In [6]:
```
print(customers)
```

```
   name_id     name  gender   income
0        1      Bob    Male   40,000
1        2      Jim    Male   50,000
2        3     Rick    Male   80,000
3        4    Katie  Female  120,000
4        5   Ashley  Female   60,000
```

In [8]:
```
print(items)
```

```
    item_id     title  price  buyer_id
0         1     table     90         1
1         2  speakers    360         1
2         3        tv    400         2
3         1     table     90         3
4         2  speakers    360         3
5         3        tv    400         3
6         4     couch   1000         4
7         1     table     90         4
8         2  speakers    360         4
9         3        tv    400         4
10        5       car  25000         4
11        1     table     90         4
12        1     table     90         5
```

In [10]:
```
#trying a basic merge
combined1 = pd.merge(customers, items, left_on='name_id', right_on='buyer_id', how='inner')
print(combined1)
```

```
    name_id    name  gender   income  item_id     title  price  buyer_id
0         1     Bob    Male   40,000        1     table     90         1
1         1     Bob    Male   40,000        2  speakers    360         1
2         2     Jim    Male   50,000        3        tv    400         2
3         3    Rick    Male   80,000        1     table     90         3
4         3    Rick    Male   80,000        2  speakers    360         3
5         3    Rick    Male   80,000        3        tv    400         3
6         4   Katie  Female  120,000        4     couch   1000         4
7         4   Katie  Female  120,000        1     table     90         4
8         4   Katie  Female  120,000        2  speakers    360         4
9         4   Katie  Female  120,000        3        tv    400         4
10        4   Katie  Female  120,000        5       car  25000         4
11        4   Katie  Female  120,000        1     table     90         4
12        5  Ashley  Female   60,000        1     table     90         5
```

In [170]:
```
#lets see if we can figure out how much each person spend on items
print(
combined1.groupby(['name'])['price'].agg(['sum', 'mean']).round(1)
)
#I think the aggregation is a more versatile than just dot function
```

```
          sum    mean
name
Ashley     90    90.0
Bob       450   225.0
Jim       400   400.0
Katie   26940  4490.0
Rick      850   283.3
```

In [171]:
```
#and let's just sort some stuff
print(
combined1.groupby(['name'])['price'].agg(['sum', 'mean']).sort_values(['sum', 'mean'], ascending=
)
```

```
          sum    mean
name
Katie   26940  4490.0
Rick      850   283.3
Bob       450   225.0
Jim       400   400.0
Ashley     90    90.0
```

In [172]:
```
#now let's take the above aggregation and filter it just for men, the first version
print(
combined1[combined1['gender'] == 'Male'].groupby(['name'])['price'].agg(['sum', 'mean']).round(1)
)
print(
combined1[combined1['gender'] == 'Female'].groupby(['name'])['price'].agg(['sum', 'mean']).round(
)
```

```
      sum   mean
name
Bob   450  225.0
Jim   400  400.0
Rick  850  283.3
          sum  mean
name
Ashley     90    90
Katie   26940  4490
```

In [55]:
```
#let's see if we can get this inot say one summary stat about it
print(
combined1[combined1['gender'] == 'Male'].groupby(['name'])['price'].agg(['sum']).mean().round(2)
)
#This will be the average total for men.
print(
combined1[combined1['gender'] == 'Female'].groupby(['name'])['price'].agg(['sum']).mean().round(2
)
#This will be the average total for women.
#also notice the slick rounding
```

```
sum    566.67
dtype: float64
sum    13515.0
dtype: float64
```

In [177]:
```
#now let's briefly think about pivot tables with pandas
pivot_table = pd.pivot_table(combined1, values = 'price', index= 'name', columns = 'gender').roun
print(pivot_table)
```

```
gender  Female   Male
name
Ashley    90.0    NaN
Bob        NaN  225.0
Jim        NaN  400.0
Katie   4490.0    NaN
Rick       NaN  283.0
```

In [ ]:
```
#so a question that could be asked
#what is the average total spend by men
```

In [64]:
```
#let's review out data again
print(customers.head(3))
print(items.head(3))
```

```
   name_id  name gender  income
0        1   Bob   Male  40,000
1        2   Jim   Male  50,000
2        3  Rick   Male  80,000
   item_id     title  price  buyer_id
0        1     table     90         1
1        2  speakers    360         1
2        3        tv    400         2
```

In [65]:
```
#first let's JOIN the tables together
combined2 = pd.merge(customers, items, left_on='name_id', right_on='buyer_id')
print(combined2)
```

```
    name_id    name  gender   income  item_id     title  price  buyer_id
0         1     Bob    Male   40,000        1     table     90         1
1         1     Bob    Male   40,000        2  speakers    360         1
2         2     Jim    Male   50,000        3        tv    400         2
3         3    Rick    Male   80,000        1     table     90         3
4         3    Rick    Male   80,000        2  speakers    360         3
5         3    Rick    Male   80,000        3        tv    400         3
6         4   Katie  Female  120,000        4     couch   1000         4
7         4   Katie  Female  120,000        1     table     90         4
8         4   Katie  Female  120,000        2  speakers    360         4
9         4   Katie  Female  120,000        3        tv    400         4
10        4   Katie  Female  120,000        5       car  25000         4
11        4   Katie  Female  120,000        1     table     90         4
12        5  Ashley  Female   60,000        1     table     90         5
```

In [127]:
```
male_total_spend_mean = combined2[combined2['gender'] == 'Male'].groupby('name').sum()['price'].m
print(male_total_spend_mean)
#here's what i think goes on here:
#when you do the sum, you only include the integer columns, so you need to do the gender sort ear
```

```
566.7
```

In [128]:
```
#using the same logic, let's check the women:
female_total_spend_mean = combined2[combined2['gender'] == 'Female'].groupby('name').sum()['price
print(female_total_spend_mean)
```

```
13515.0
```

In [115]:
```
#now i am trying to basically replicate a having clause in the pandas stuff
```

In [169]:
```
#bizare, but this is effectivley how to simulate a HAVING clause in pandas
#indexed_df[indexed_df['petal length (cm)'] > 1.4]
#https://stackoverflow.com/questions/48304854/pandas-filter-method-with-lambda-function?rq=1
test = combined2.groupby('name').sum()
print(test.sort_values('price'))

print(test[test['price'] > 400].sort_values('price'))
#i've add the sort_values stuff to order it
#and the notes from the peson who got it:
#"""
#You can use the condition indexed_df['petal length (cm)'] > 1.4
#(here we use indexed_df, not  x) as a way to filter the dataframe, so:
#indexed_df[indexed_df['petal length (cm)'] > 1.4]
#How does this work?
#If you perform indexed_df['petal length (cm)'] you obtain the "column" of the dataframe:
#some sort of sequence where for every index, we get the value of that column.
#By performing a column > 1.4, we obtain some sort of column of booleans: True if the condition i
#for a certain row, and  False otherwise.
#We then can use such boolean column as an element for the dataframe indexed_df[boolean_column]
#to obtain only the rows where the corresponding row of the boolean_column is True.
#"""
```

```
        name_id  item_id  price  buyer_id
name
Ashley        5        1     90         5
Jim           2        3    400         2
Bob           2        3    450         2
Rick          9        6    850         9
Katie        24       16  26940        24
       name_id  item_id  price  buyer_id
name
Bob          2        3    450         2
Rick         9        6    850         9
Katie       24       16  26940        24
```

In [168]:
```
test = combined2.groupby('name').sum()
print(test.sort_values('price'))

print(test[(test['price'] > 400) == False].sort_values('price'))
#this way helps you get the other reveser
```

```
        name_id  item_id  price  buyer_id
name
Ashley        5        1     90         5
Jim           2        3    400         2
Bob           2        3    450         2
Rick          9        6    850         9
Katie        24       16  26940        24
        name_id  item_id  price  buyer_id
name
Ashley        5        1     90         5
Jim           2        3    400         2
```

In [ ]: