In [113]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [114]:
```
sales = pd.read_excel("supermarket.xls")
```

In [115]:
```
sales
```

Out[115]:

	Invoice ID	Branch	City	Customer_type	Gender	Product line	Unit price	Quantity	Tax 5%	
0	750-67-8428	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	5
1	226-31-3081	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	
2	631-41-3108	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	3
3	123-19-1176	A	Yangon	Member	Male	Health and beauty	58.22	8	23.2880	4
4	373-73-7910	A	Yangon	Normal	Male	Sports and travel	86.31	7	30.2085	6
...	...	...	...	...	...	...	...	...	...	
995	233-67-5758	C	Naypyitaw	Normal	Male	Health and beauty	40.35	1	2.0175	
996	303-96-2227	B	Mandalay	Normal	Female	Home and lifestyle	97.38	10	48.6900	10
997	727-02-1313	A	Yangon	Member	Male	Food and beverages	31.84	1	1.5920	
998	347-56-2442	A	Yangon	Normal	Male	Home and lifestyle	65.82	1	3.2910	
999	849-09-3807	A	Yangon	Member	Female	Fashion accessories	88.34	7	30.9190	6

1000 rows × 17 columns

In [116]:
```
sales.head()
```

Out[116]:

	Invoice ID	Branch	City	Customer_type	Gender	Product line	Unit price	Quantity	Tax 5%	T
0	750-67-8428	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9
1	226-31-3081	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2
2	631-41-3108	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5
3	123-19-1176	A	Yangon	Member	Male	Health and beauty	58.22	8	23.2880	489.0
4	373-73-7910	A	Yangon	Normal	Male	Sports and travel	86.31	7	30.2085	634.3

In [117]:
```
sales.tail()
```

Out[117]:

	Invoice ID	Branch	City	Customer_type	Gender	Product line	Unit price	Quantity	Tax 5%	
995	233-67-5758	C	Naypyitaw	Normal	Male	Health and beauty	40.35	1	2.0175	
996	303-96-2227	B	Mandalay	Normal	Female	Home and lifestyle	97.38	10	48.6900	10
997	727-02-1313	A	Yangon	Member	Male	Food and beverages	31.84	1	1.5920	
998	347-56-2442	A	Yangon	Normal	Male	Home and lifestyle	65.82	1	3.2910	
999	849-09-3807	A	Yangon	Member	Female	Fashion accessories	88.34	7	30.9190	6

In [118]:
```
sales.dtypes
```

Out[118]:
```
Invoice ID                      object
Branch                          object
City                            object
Customer_type                   object
Gender                          object
Product line                    object
Unit price                     float64
Quantity                         int64
Tax 5%                         float64
Total                          float64
Date                    datetime64[ns]
Time                            object
Payment                         object
cogs                           float64
gross margin percentage        float64
gross income                   float64
Rating                         float64
dtype: object
```

In [119]:
```
type(sales['Date'][0])
```

Out[119]:
```
pandas._libs.tslibs.timestamps.Timestamp
```

In [120]:
```
from pandas import to_datetime
```

In [121]:
```
sales['Date']=to_datetime(sales['Date'])
```

In [122]:
```
type(sales['Date'][0])
```

Out[122]:
```
pandas._libs.tslibs.timestamps.Timestamp
```

In [123]:
```
sales['Date'].dtype
```

Out[123]:
```
dtype('<M8[ns]')
```

In [124]:
```
type(sales['Time'][0])
```

Out[124]:
```
datetime.time
```

In [125]:
```
sales['Date']
```

Out[125]:
```
0     2021-01-05
1     2021-03-08
2     2021-03-03
3     2021-01-27
4     2021-02-08
         ...
995   2021-01-29
996   2021-03-02
997   2021-02-09
998   2021-02-22
999   2021-02-18
Name: Date, Length: 1000, dtype: datetime64[ns]
```

In [126]:
```
sales['Time']
```

Out[126]:
```
0      13:08:00
1      10:29:00
2      13:23:00
3      20:33:00
4      10:37:00
          ...
995    13:46:00
996    17:16:00
997    13:22:00
998    15:33:00
999    13:28:00
Name: Time, Length: 1000, dtype: object
```

In [127]:
```
def fetch_att(x):
    day=x.day
    month=x.month
    year=x.year
    return pd.Series([day,month,year])
```

In [128]:
```
sales[['day','month','year']]=sales['Date'].apply(fetch_att)
```

In [129]:
```
sales.head()
```

Out[129]:

	Invoice ID	Branch	City	Customer_type	Gender	Product line	Unit price	Quantity	Tax 5%	T
0	750-67-8428	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9
1	226-31-3081	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2
2	631-41-3108	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5
3	123-19-1176	A	Yangon	Member	Male	Health and beauty	58.22	8	23.2880	489.0
4	373-73-7910	A	Yangon	Normal	Male	Sports and travel	86.31	7	30.2085	634.3

In [130]:
```
sales['Date'].apply(lambda x:x.day)
```

Out[130]:
```
0       5
1       8
2       3
3      27
4       8
      ..
995    29
996     2
997     9
998    22
999    18
Name: Date, Length: 1000, dtype: int64
```

In [131]:
```
sales['Date'].dt.year
```

Out[131]:
```
0      2021
1      2021
2      2021
3      2021
4      2021
       ...
995    2021
996    2021
997    2021
998    2021
999    2021
Name: Date, Length: 1000, dtype: int64
```

In [132]:
```
sales.columns
```

Out[132]:
```
Index(['Invoice ID', 'Branch', 'City', 'Customer_type', 'Gender',
       'Product line', 'Unit price', 'Quantity', 'Tax 5%', 'Total', 'Date',
       'Time', 'Payment', 'cogs', 'gross margin percentage', 'gross income',
       'Rating', 'day', 'month', 'year'],
      dtype='object')
```

In [133]:
```
sales['hour']=sales['Time'].apply(lambda x:x.hour)
```

In [134]:
```
sales.head()
```

Out[134]:

	Invoice ID	Branch	City	Customer_type	Gender	Product line	Unit price	Quantity	Tax 5%	T
0	750-67-8428	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9
1	226-31-3081	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2
2	631-41-3108	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5
3	123-19-1176	A	Yangon	Member	Male	Health and beauty	58.22	8	23.2880	489.0
4	373-73-7910	A	Yangon	Normal	Male	Sports and travel	86.31	7	30.2085	634.3

5 rows × 21 columns

In [135]:
```
sales['Time']
```

Out[135]:
```
0      13:08:00
1      10:29:00
2      13:23:00
3      20:33:00
4      10:37:00
         ...
995    13:46:00
996    17:16:00
997    13:22:00
998    15:33:00
999    13:28:00
Name: Time, Length: 1000, dtype: object
```

In [136]:
```
sales.describe().T
```

Out[136]:

	count	mean	std	min	25%	50%	75%
Unit price	1000.0	55.672130	2.649463e+01	10.080000	32.875000	55.230000	77.93500(
Quantity	1000.0	5.510000	2.923431e+00	1.000000	3.000000	5.000000	8.000000
Tax 5%	1000.0	15.379369	1.170883e+01	0.508500	5.924875	12.088000	22.44525(
Total	1000.0	322.966749	2.458853e+02	10.678500	124.422375	253.848000	471.35025(
cogs	1000.0	307.587380	2.341765e+02	10.170000	118.497500	241.760000	448.90500(
gross margin percentage	1000.0	4.761905	6.131498e-14	4.761905	4.761905	4.761905	4.76190!
gross income	1000.0	15.379369	1.170883e+01	0.508500	5.924875	12.088000	22.44525(
Rating	1000.0	6.972700	1.718580e+00	4.000000	5.500000	7.000000	8.500000
day	1000.0	15.256000	8.693563e+00	1.000000	8.000000	15.000000	23.000000
month	1000.0	1.993000	8.352536e-01	1.000000	1.000000	2.000000	3.000000
year	1000.0	2021.000000	0.000000e+00	2021.000000	2021.000000	2021.000000	2021.000000
hour	1000.0	14.910000	3.186857e+00	10.000000	12.000000	15.000000	18.000000

In [137]:
```
sales.corr()
```

Out[137]:

	Unit price	Quantity	Tax 5%	Total	cogs	gross margin percentage	gross income	Rating
Unit price	1.000000	0.010778	0.633962	0.633962	0.633962	NaN	0.633962	-0.008778
Quantity	0.010778	1.000000	0.705510	0.705510	0.705510	NaN	0.705510	-0.015815
Tax 5%	0.633962	0.705510	1.000000	1.000000	1.000000	NaN	1.000000	-0.036442
Total	0.633962	0.705510	1.000000	1.000000	1.000000	NaN	1.000000	-0.036442
cogs	0.633962	0.705510	1.000000	1.000000	1.000000	NaN	1.000000	-0.036442
gross margin percentage	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
gross income	0.633962	0.705510	1.000000	1.000000	1.000000	NaN	1.000000	-0.036442
Rating	-0.008778	-0.015815	-0.036442	-0.036442	-0.036442	NaN	-0.036442	1.000000
day	0.057021	-0.043347	-0.002515	-0.002515	-0.002515	NaN	-0.002515	-0.007076
month	-0.027387	-0.014524	-0.022301	-0.022301	-0.022301	NaN	-0.022301	-0.042880
year	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
hour	0.008242	-0.007317	-0.002770	-0.002770	-0.002770	NaN	-0.002770	-0.030588

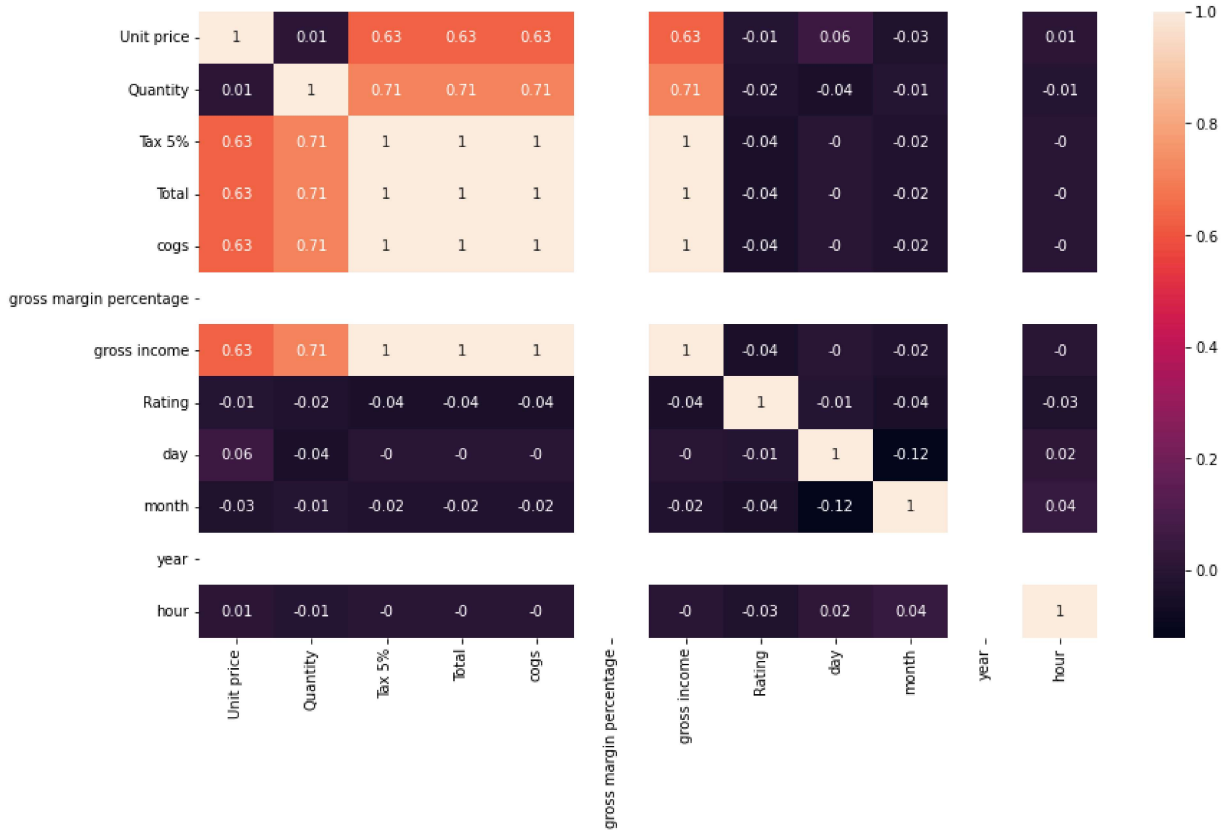
In [138]:
```
np.round(sales.corr(),2)
```

Out[138]:

	Unit price	Quantity	Tax 5%	Total	cogs	gross margin percentage	gross income	Rating	day	month	year	h
Unit price	1.00	0.01	0.63	0.63	0.63	NaN	0.63	-0.01	0.06	-0.03	NaN	(
Quantity	0.01	1.00	0.71	0.71	0.71	NaN	0.71	-0.02	-0.04	-0.01	NaN	-(
Tax 5%	0.63	0.71	1.00	1.00	1.00	NaN	1.00	-0.04	-0.00	-0.02	NaN	-(
Total	0.63	0.71	1.00	1.00	1.00	NaN	1.00	-0.04	-0.00	-0.02	NaN	-(
cogs	0.63	0.71	1.00	1.00	1.00	NaN	1.00	-0.04	-0.00	-0.02	NaN	-(
gross margin percentage	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	I
gross income	0.63	0.71	1.00	1.00	1.00	NaN	1.00	-0.04	-0.00	-0.02	NaN	-(
Rating	-0.01	-0.02	-0.04	-0.04	-0.04	NaN	-0.04	1.00	-0.01	-0.04	NaN	-(
day	0.06	-0.04	-0.00	-0.00	-0.00	NaN	-0.00	-0.01	1.00	-0.12	NaN	(
month	-0.03	-0.01	-0.02	-0.02	-0.02	NaN	-0.02	-0.04	-0.12	1.00	NaN	(
year	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	I
hour	0.01	-0.01	-0.00	-0.00	-0.00	NaN	-0.00	-0.03	0.02	0.04	NaN	

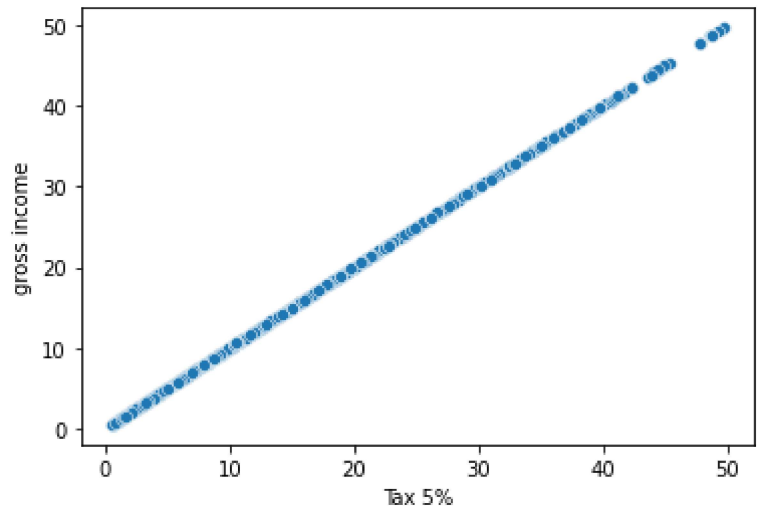
In [139]:
```
plt.figure(figsize=(14,8))
sns.heatmap(np.round(sales.corr(),2),annot=True)
```

Out[139]: <AxesSubplot:>



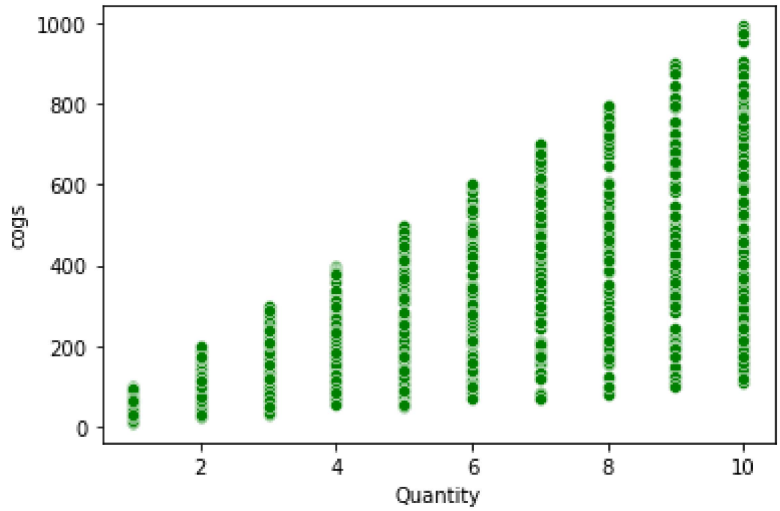
In [140]:
```
sns.scatterplot(x='Tax 5%',y='gross income',data=sales)
```

Out[140]: <AxesSubplot:xlabel='Tax 5%', ylabel='gross income'>



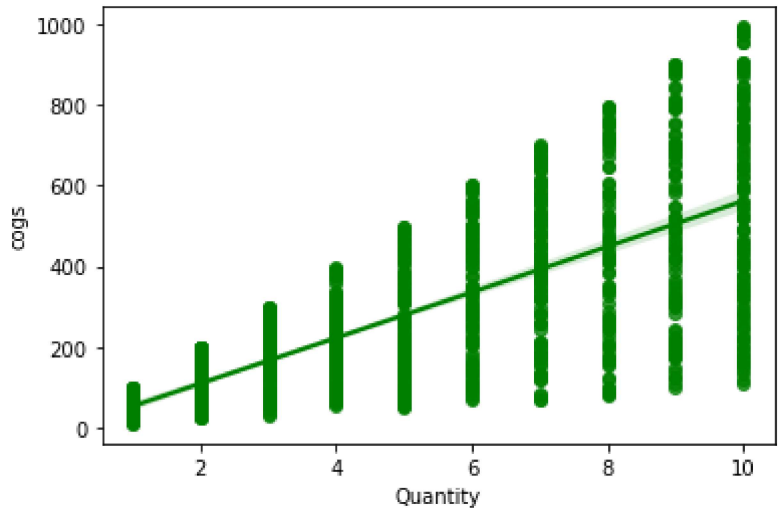
In [141]:
```
sns.scatterplot(x='Quantity',y='cogs',data=sales,color='green')
```

Out[141]: <AxesSubplot:xlabel='Quantity', ylabel='cogs'>



In [142]:
```
sns.regplot(x='Quantity',y='cogs',data=sales,color='green')
```

Out[142]: <AxesSubplot:xlabel='Quantity', ylabel='cogs'>



In [143]:
```
sales['City'].unique()
```

Out[143]: array(['Yangon', 'Naypyitaw', 'Mandalay'], dtype=object)

In [144]:
```
sales.groupby(['City'])['gross income'].median()
```

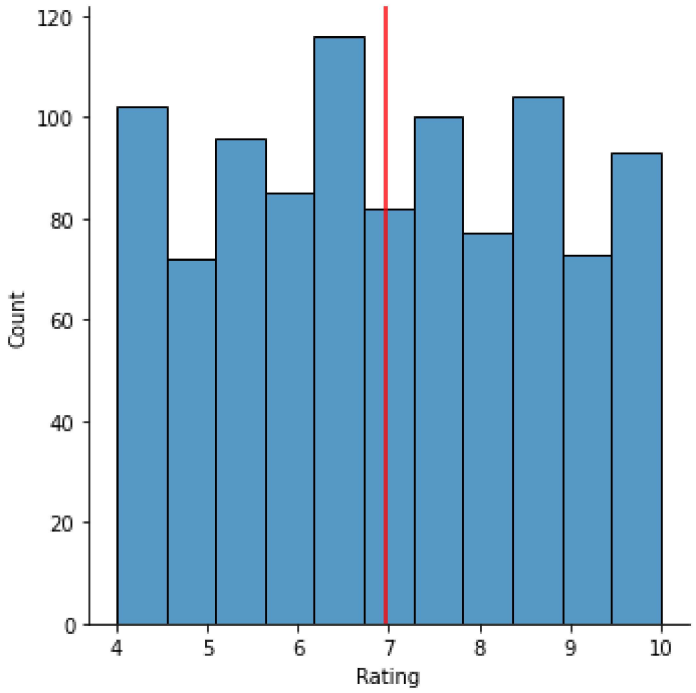
Out[144]:
```
City
Mandalay     12.04200
Naypyitaw    12.92475
Yangon       11.46800
Name: gross income, dtype: float64
```

In [145]:
```
sales['Rating'].mean()
```

Out[145]:
```
6.972700000000003
```

In [146]:
```
sns.displot(sales['Rating'],kde=False)
plt.axvline(x=np.mean(sales['Rating']),c='red',label='Avg Rating')
```

Out[146]:
```
<matplotlib.lines.Line2D at 0x201db71a520>
```



In [147]:
```
def return_countplot(column,hue_name=None):
    return sns.countplot(x=column,data=sales,hue=hue_name)
```

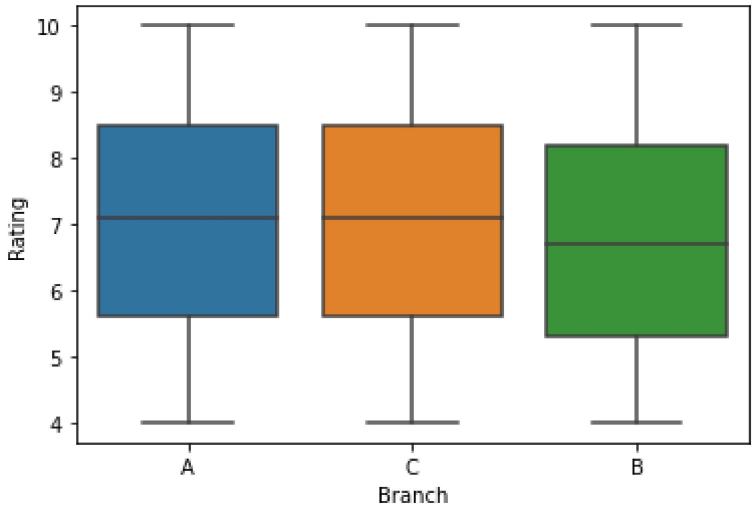
In [148]:
```
def return_boxplot(x_column,y_column):
    return sns.boxplot(x=x_column,y=y_column,data=sales)
```

In [149]:
```
def return_lineplot(x_column,y_column):
    return sns.lineplot(x=x_column,y=y_column,data=sales)
```

In [150]:
```
def return_rel_plot(x_col, y_col, col_name=None, row_name=None, rel_type=None, hu
    return sns.relplot(x=x_col, y=y_col, col=col_name, row=row_name, kind=rel_typ
```

In [151]:
```
return_boxplot('Branch','Rating')
```

Out[151]:
```
<AxesSubplot:xlabel='Branch', ylabel='Rating'>
```



In [155]:
```
sales.columns
```

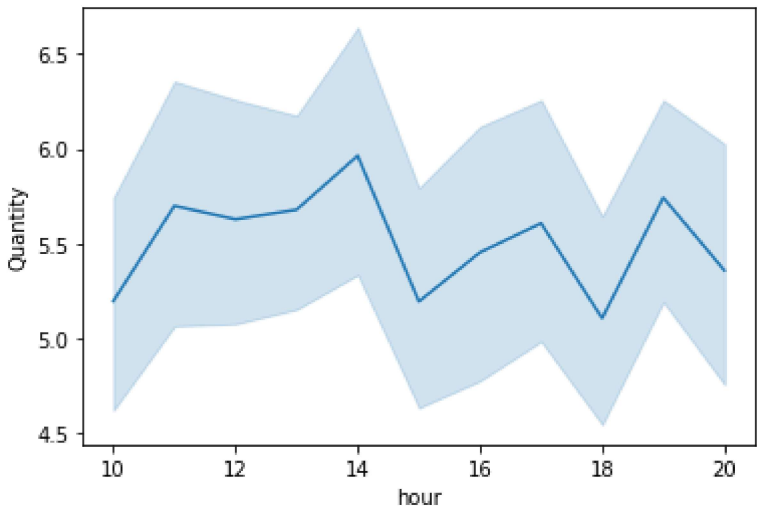
Out[155]:
```
Index(['Invoice ID', 'Branch', 'City', 'Customer_type', 'Gender',
       'Product line', 'Unit price', 'Quantity', 'Tax 5%', 'Total', 'Date',
       'Time', 'Payment', 'cogs', 'gross margin percentage', 'gross income',
       'Rating', 'day', 'month', 'year', 'hour'],
      dtype='object')
```

In [156]:
```
sales.dtypes
```

Out[156]:
```
Invoice ID                      object
Branch                          object
City                            object
Customer_type                   object
Gender                          object
Product line                    object
Unit price                     float64
Quantity                         int64
Tax 5%                         float64
Total                          float64
Date                    datetime64[ns]
Time                            object
Payment                         object
cogs                           float64
gross margin percentage        float64
gross income                   float64
Rating                         float64
day                              int64
month                            int64
year                             int64
hour                             int64
dtype: object
```

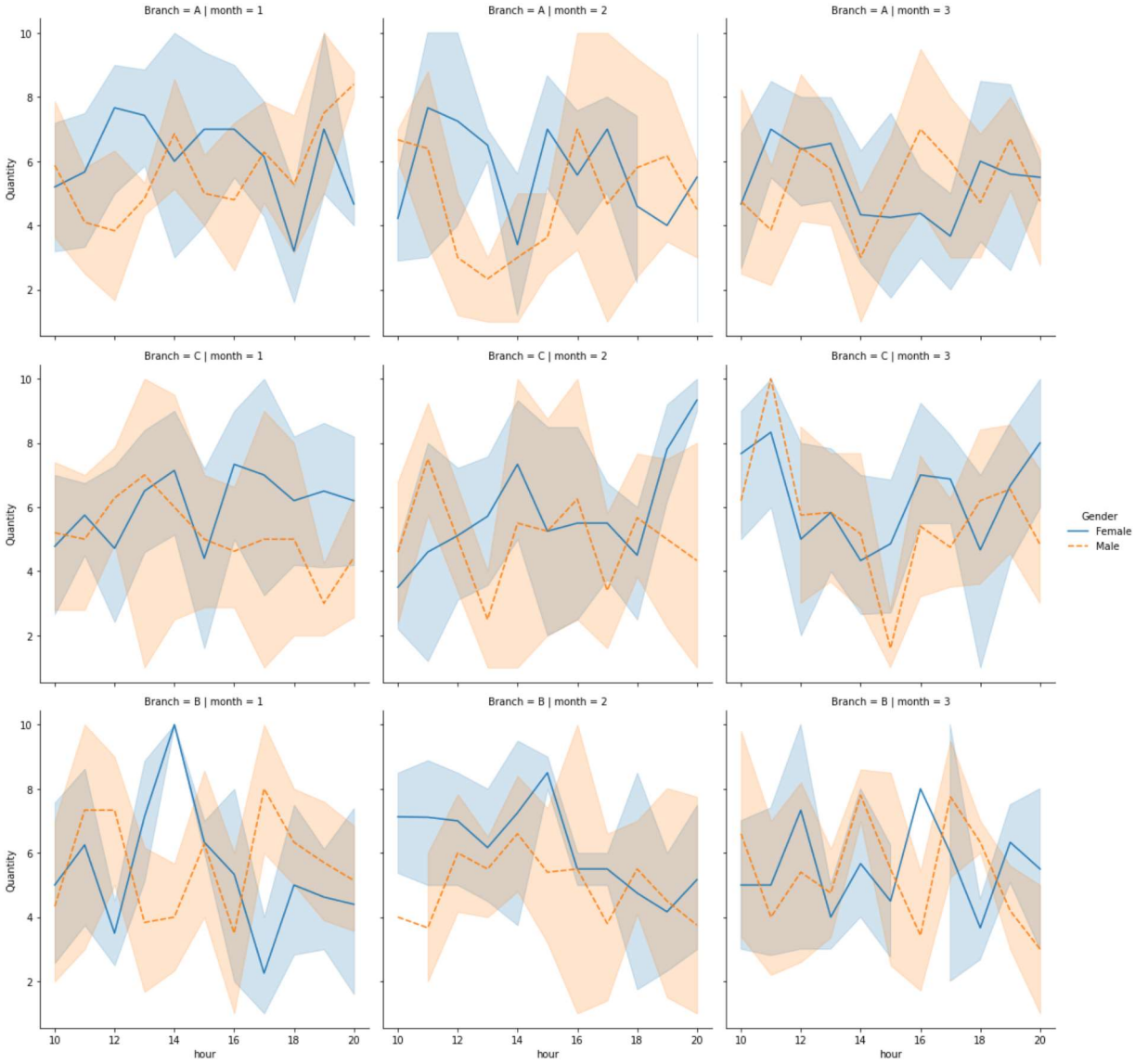
In [157]:
```
return_lineplot('hour','Quantity')
```

Out[157]:
```
<AxesSubplot:xlabel='hour', ylabel='Quantity'>
```



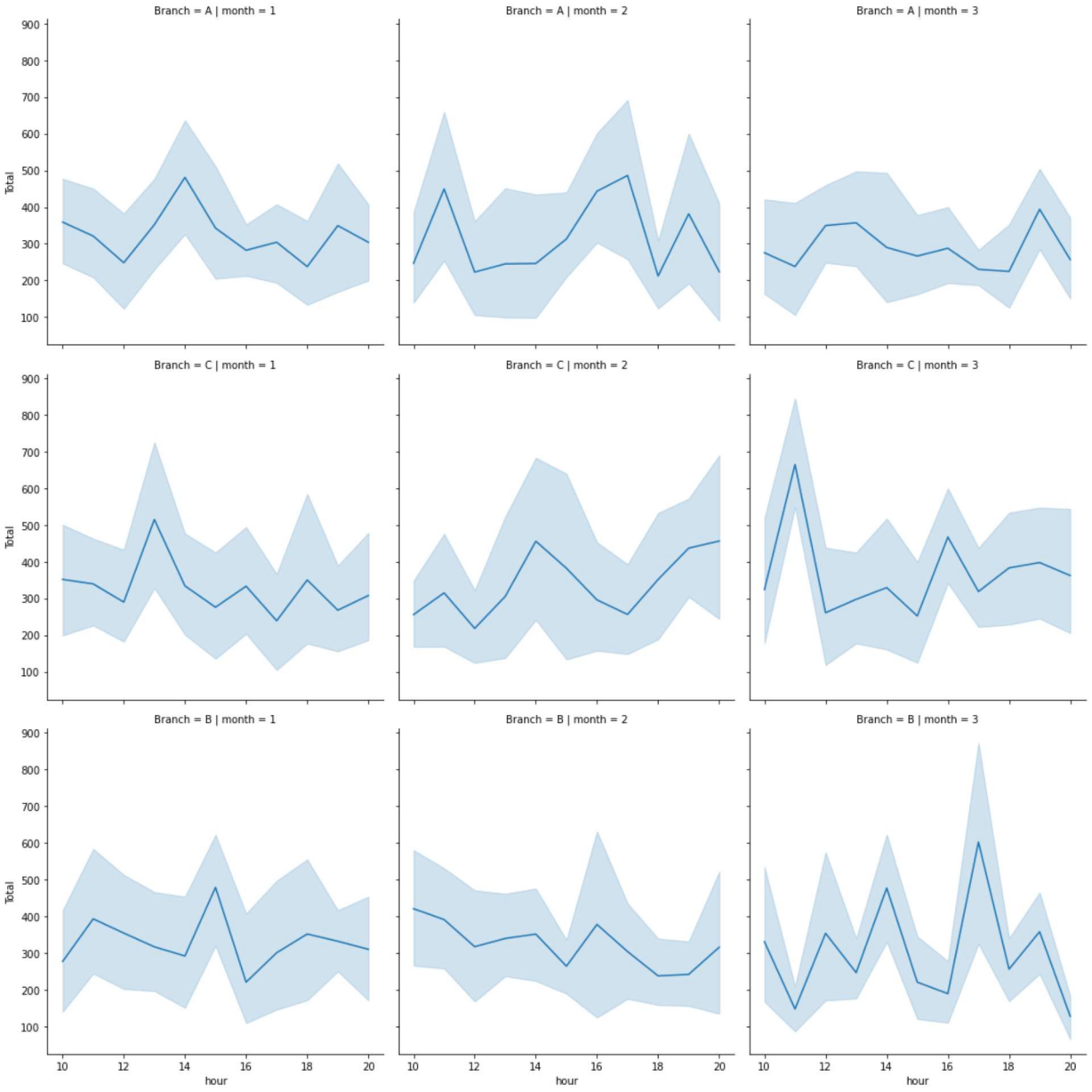
In [158]:
```
return_rel_plot(x_col='hour',y_col='Quantity',col_name='month',row_name='Branch',
```

Out[158]: <seaborn.axisgrid.FacetGrid at 0x201db868310>



In [159]:
```
return_rel_plot(x_col='hour',y_col='Total',col_name='month',row_name='Branch',re
```

Out[159]:
```
<seaborn.axisgrid.FacetGrid at 0x201dcf72130>
```

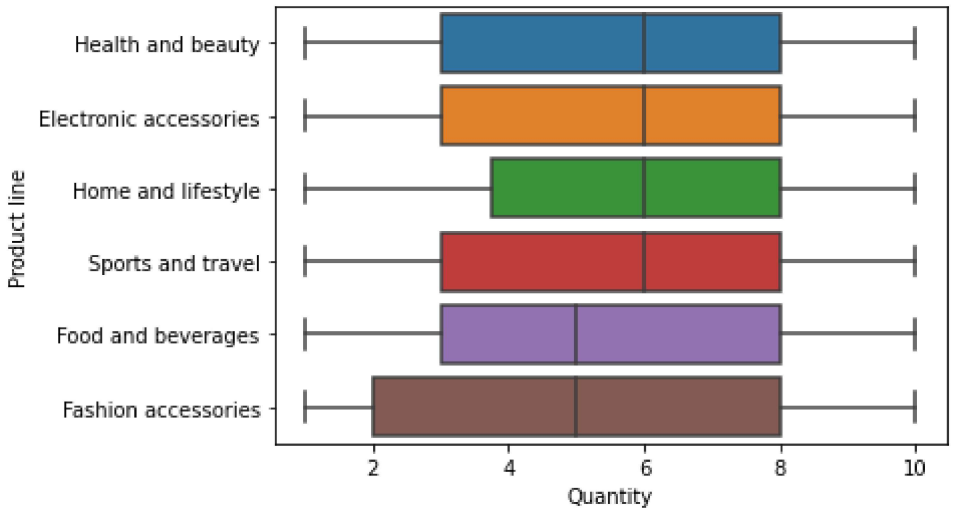


In [162]:
```
return_rel_plot(x_col='hour',y_col='Total',col_name='Product line',row_name='Bran
```

Out[162]: <seaborn.axisgrid.FacetGrid at 0x201dead4460>

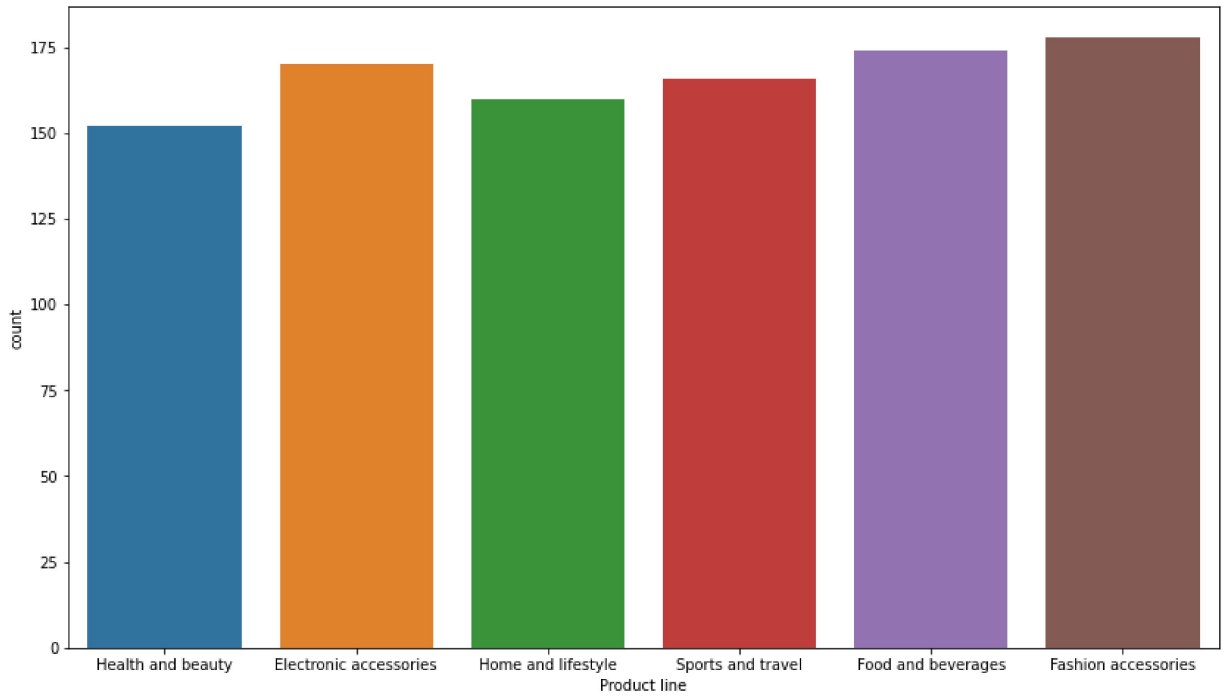
In [163]:
```
return_boxplot('Quantity','Product line')
```

Out[163]: <AxesSubplot:xlabel='Quantity', ylabel='Product line'>



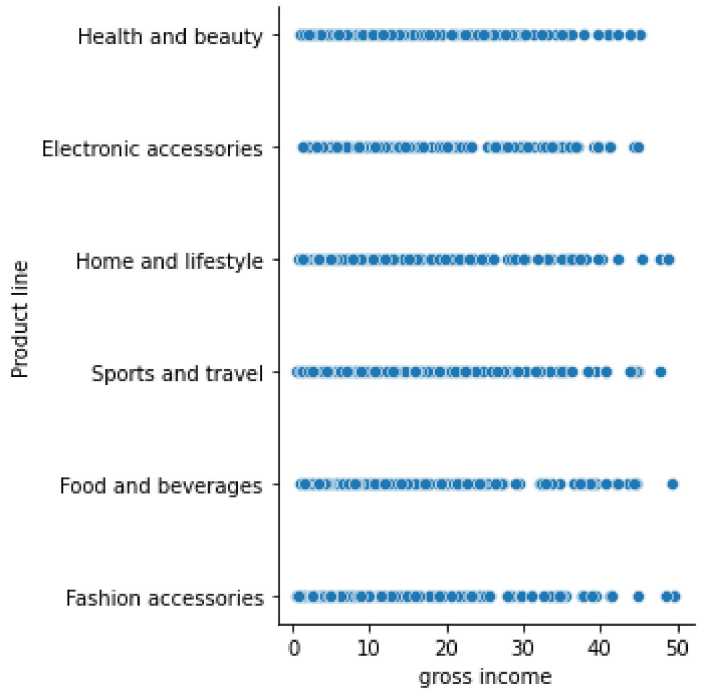
In [166]:
```
plt.figure(figsize=(14,8))
return_countplot('Product line')
```

Out[166]: <AxesSubplot:xlabel='Product line', ylabel='count'>



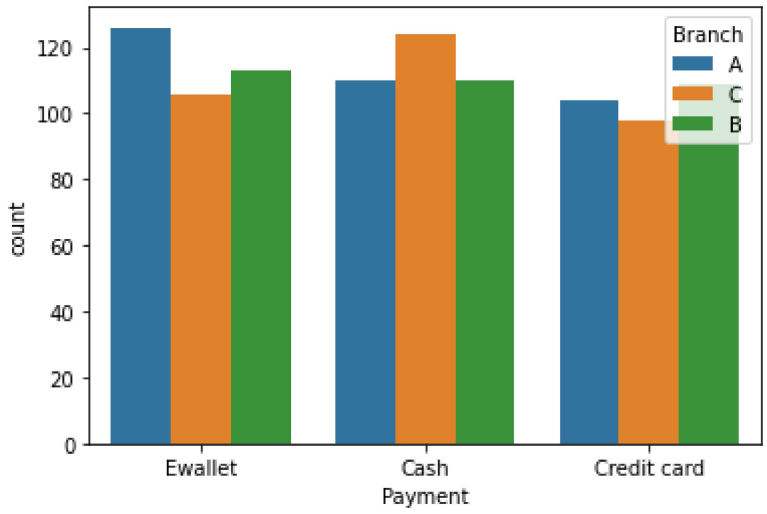
In [167]:
```
return_rel_plot('gross income','Product line',rel_type='scatter')
```

Out[167]: <seaborn.axisgrid.FacetGrid at 0x201e202dfd0>



In [168]:
```
return_countplot('Payment',hue_name='Branch')
```

Out[168]: <AxesSubplot:xlabel='Payment', ylabel='count'>



In [172]:
```
sales.groupby('Customer_type')['Total'].sum()
```

Out[172]:
```
Customer_type
Member    164223.444
Normal    158743.305
Name: Total, dtype: float64
```

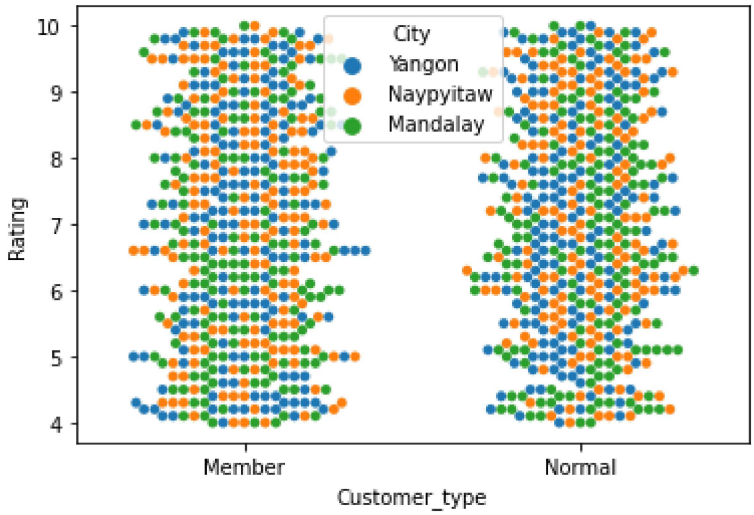
In [174]:
```
sales.groupby('Customer_type').agg({'Total':'sum'})
```

Out[174]:

Customer_type	Total
Member	164223.444
Normal	158743.305

In [176]:
```
sns.swarmplot(x='Customer_type',y='Rating',data=sales,hue='City')
```

Out[176]:
```
<AxesSubplot:xlabel='Customer_type', ylabel='Rating'>
```



In [ ]:
```

```