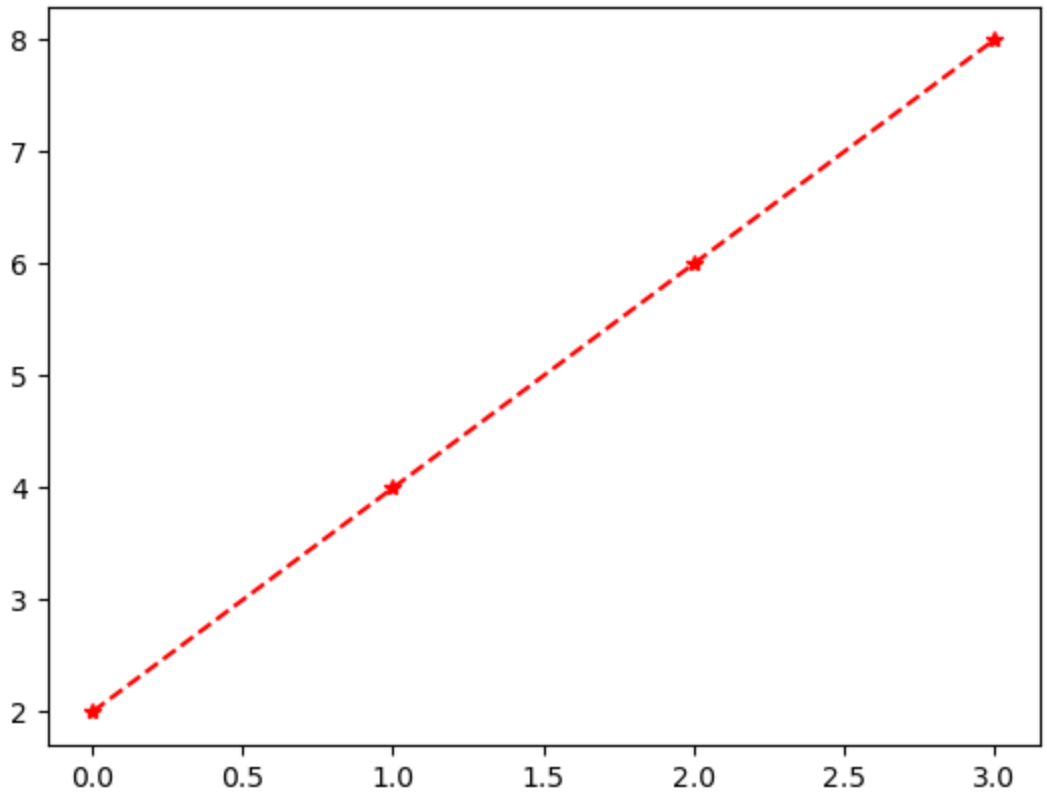
In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

import seaborn as sns
```

I) Line graph:

- plt.plot([x], y, [fmt_string])
- x = (0....n-1)

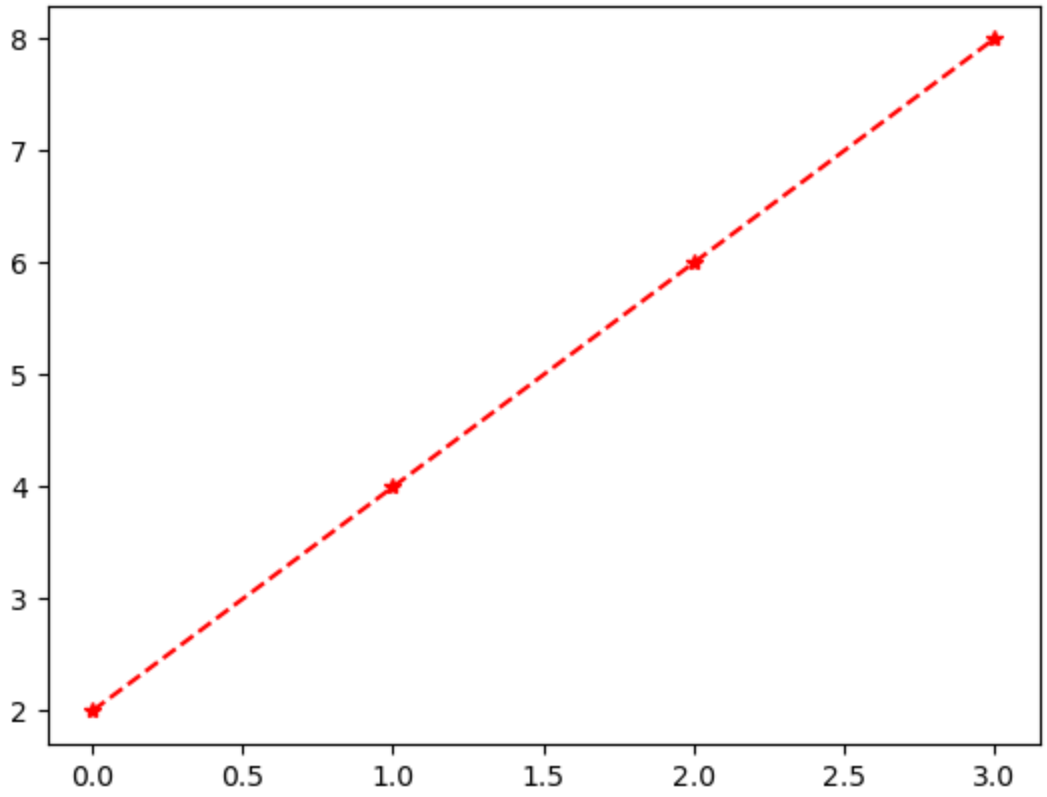
In [3]:
```
plt.plot([2,4,6,8],'r*--' )
```

Out[3]:
```
[<matplotlib.lines.Line2D at 0x20f8159ea40>]
```



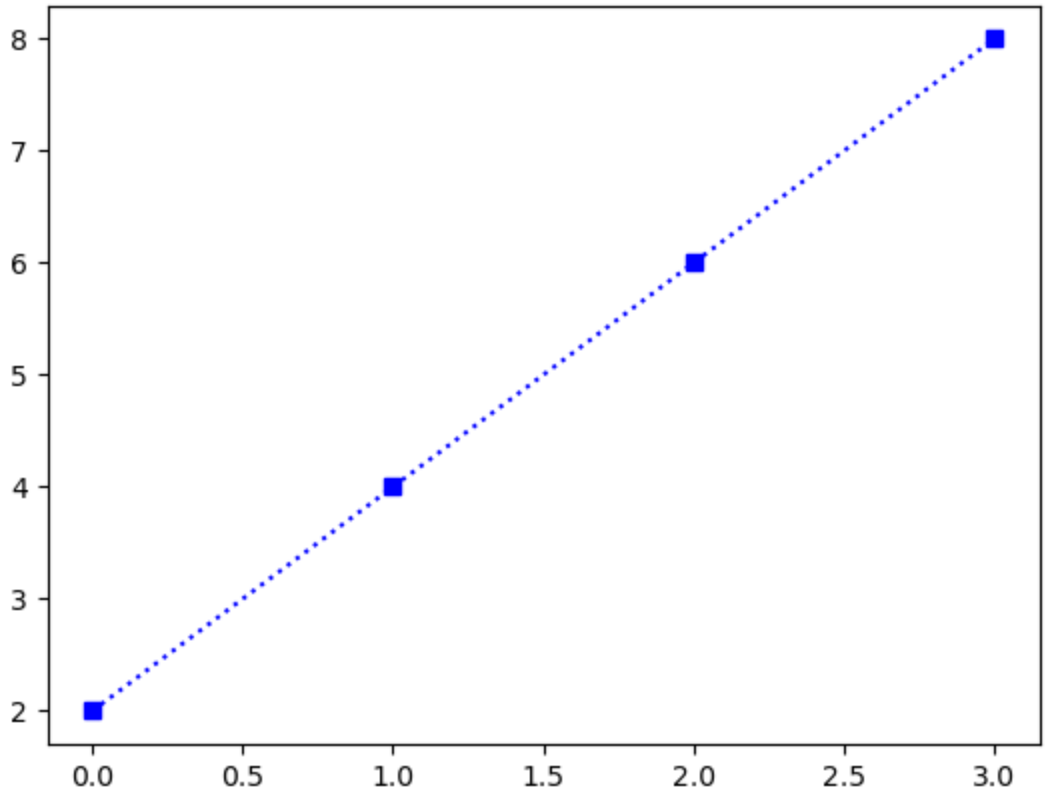
In [4]:
```
plt.plot([0,1,2,3],[2,4,6,8],'r*--' )
```

Out[4]:
```
[<matplotlib.lines.Line2D at 0x20f81e20c70>]
```



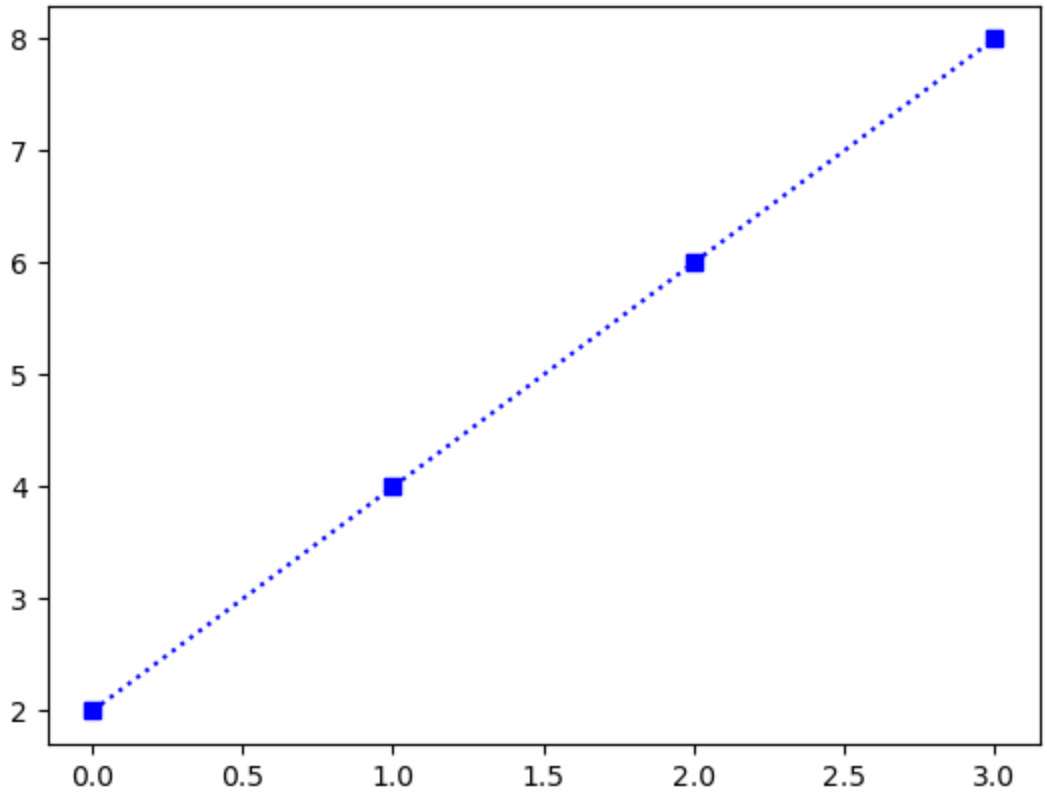
In [5]:
```
plt.plot([0,1,2,3],[2,4,6,8],'bs:' )
```

Out[5]:
```
[<matplotlib.lines.Line2D at 0x20f81e833a0>]
```



In [6]:
```
plt.plot([2,4,6,8],'bs:' )
```

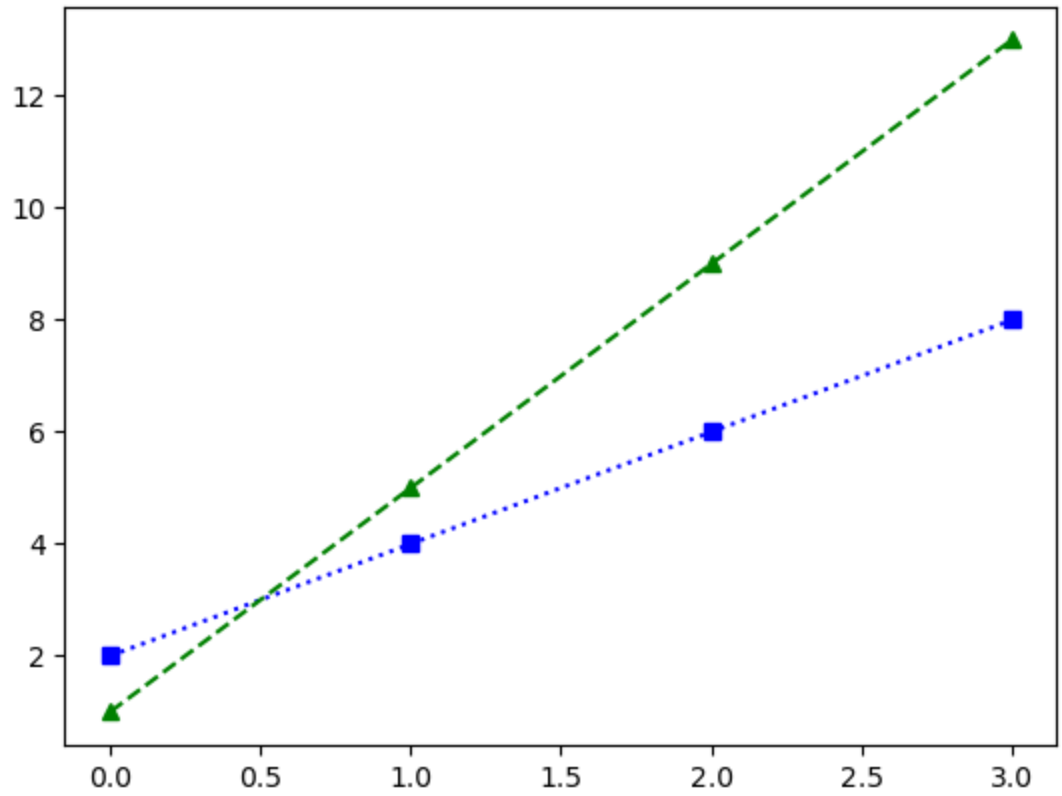
Out[6]:
```
[<matplotlib.lines.Line2D at 0x20f81f0d900>]
```



1.2) Multiple data-pairs:

In [7]:
```
plt.plot([2,4,6,8],'bs:', [1,5,9,13], 'g^--')
```

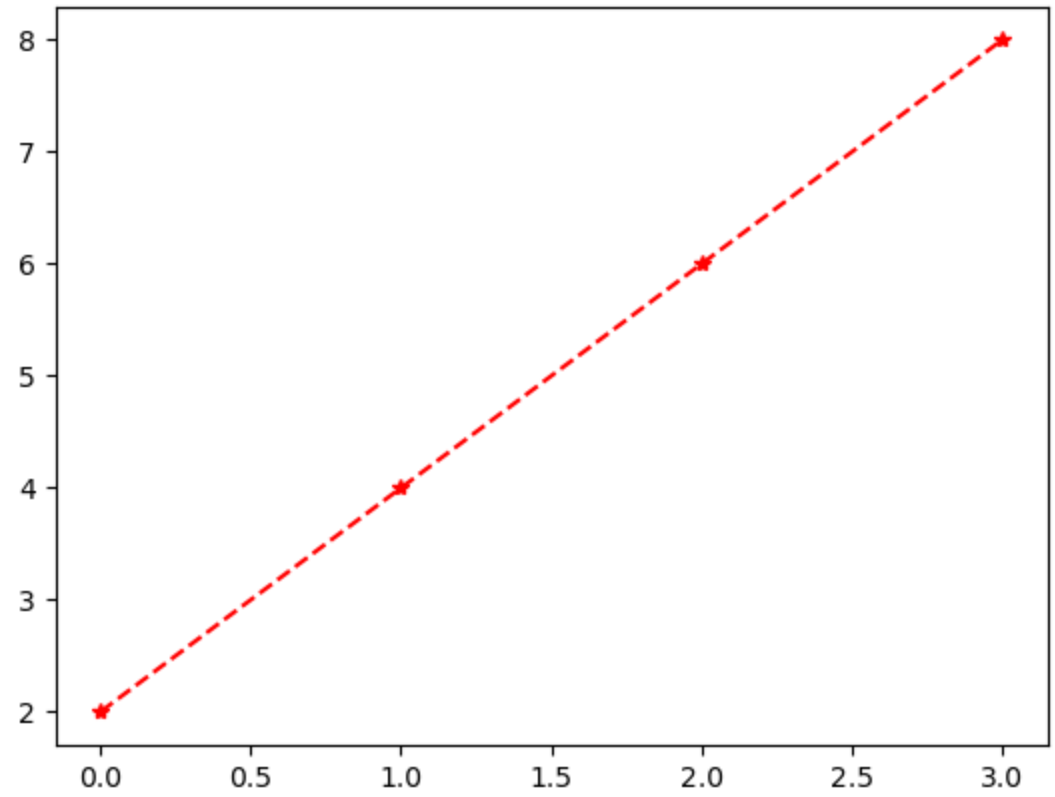
Out[7]:
```
[<matplotlib.lines.Line2D at 0x20f81f945b0>,
 <matplotlib.lines.Line2D at 0x20f81f94640>]
```



1.3) Use `markersize` and `linewidth` to modify the appearance of the Line.

In [8]:
```
plt.plot([2,4,6,8],'r*--' )
```

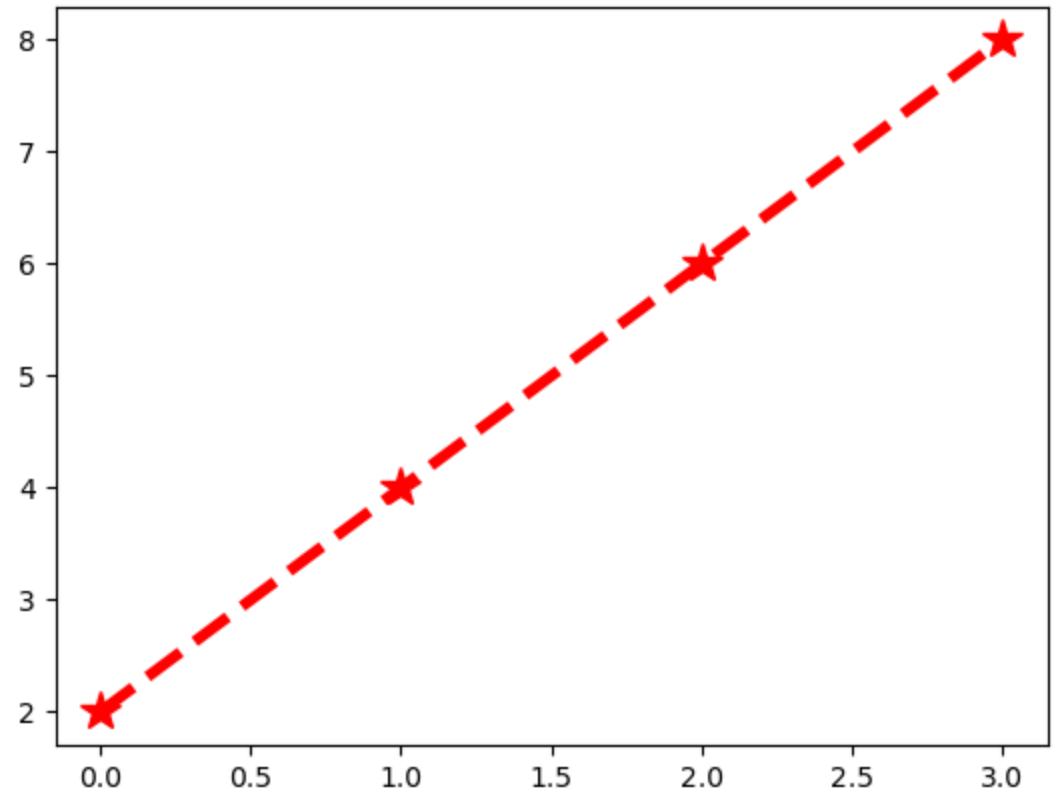
Out[8]: [<matplotlib.lines.Line2D at 0x20f81ff9c00>]



In [ ]:
```
# changing the Dimensions of the Line and Marker:
```

In [9]:
```
plt.plot([2,4,6,8],'r*--' , linewidth = 4, markersize = 15 )
```

Out[9]: [<matplotlib.lines.Line2D at 0x20f8204ba00>]



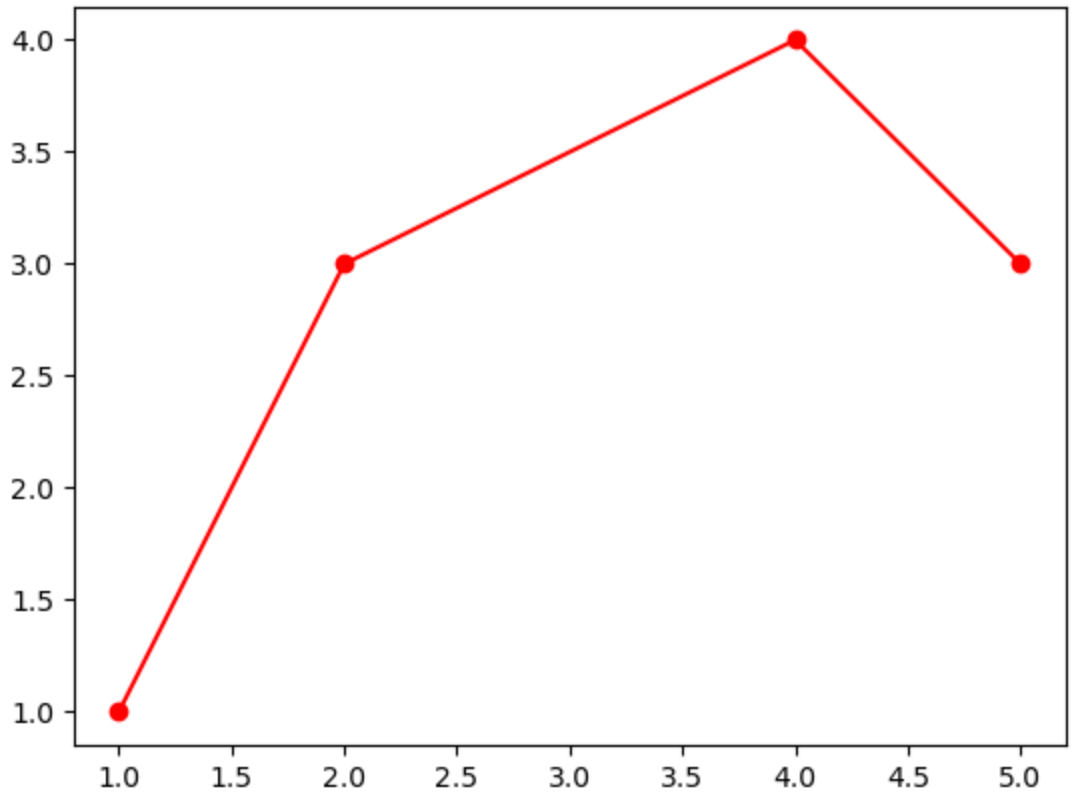
1.4)

- Instead of manually typing the list of x-co-ords and y-co-ords, you can specify a Pandas DF as the source.
- Syntax: `plt.plot('xcol','ycol', data= df_name)`

1.5)

- To create a new Figure object: `plt.figure()`
- To plot data-points in the form of lines/markers/colours: `plt.plot()`
- To display the Figure: `plt.show()`
- To save the Figure: `plt.savefig('filename.png')`

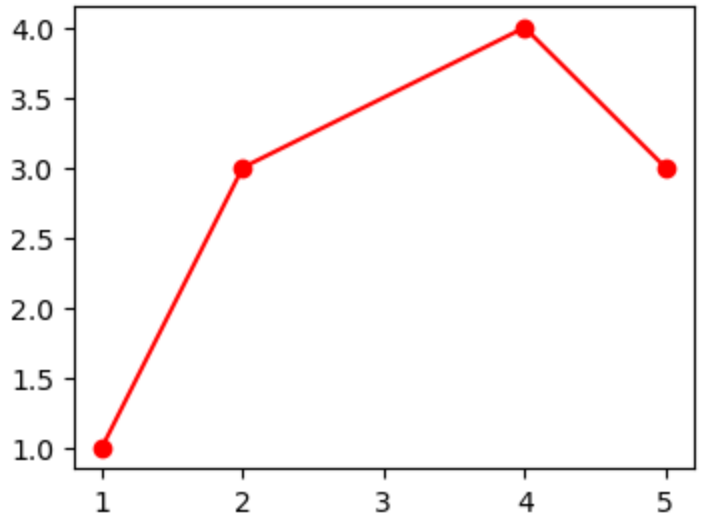
In [11]:
```
plt.figure()
plt.plot([1,2,4,5], [1,3,4,3], 'ro-')
plt.show()
plt.savefig('first_lineplot.png')
```




```
<Figure size 640x480 with 0 Axes>
```

In [14]:
```
# changing the Dimensions of the Figure:
```

In [15]:
```
plt.figure(figsize = (4,3))
plt.plot([1,2,4,5], [1,3,4,3], 'ro-')
plt.show()
```



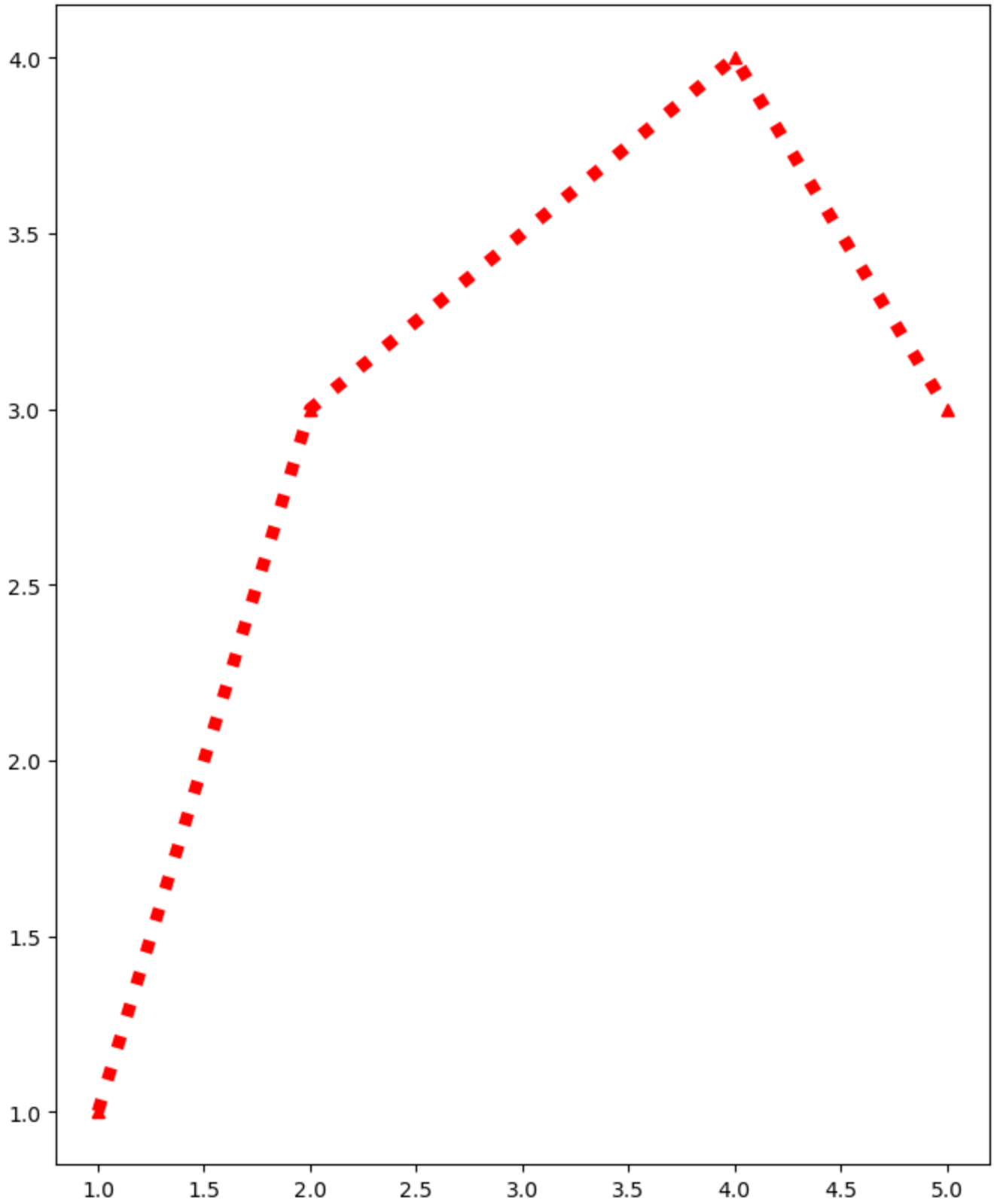
In [ ]:

Exercise: Plot a Line-graph:

- (1,1)
- (2,3)
- (4,4)
- (5,3)

Modify the line to have circular Markers to denote these datapoints.
The line should be dotted in style, and be red in colour.
The thickness of the line should be 6.
The figure should be 10 units in height and width of 8 units. Save the Figure as `example1.png`.

In [20]:
```
plt.figure(figsize = (8,10))
plt.plot([1,2,4,5],[1,3,4,3],'r^:', linewidth = 6)
plt.show()
plt.savefig('example1.png')
```

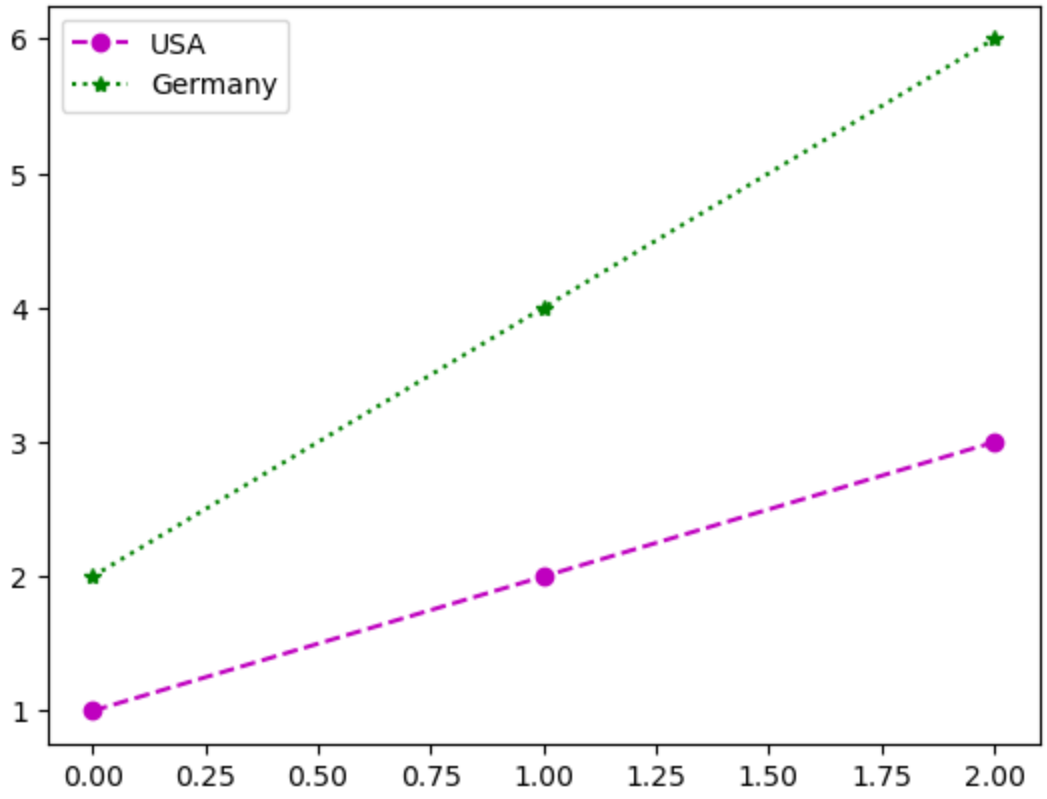



```
<Figure size 640x480 with 0 Axes>
```

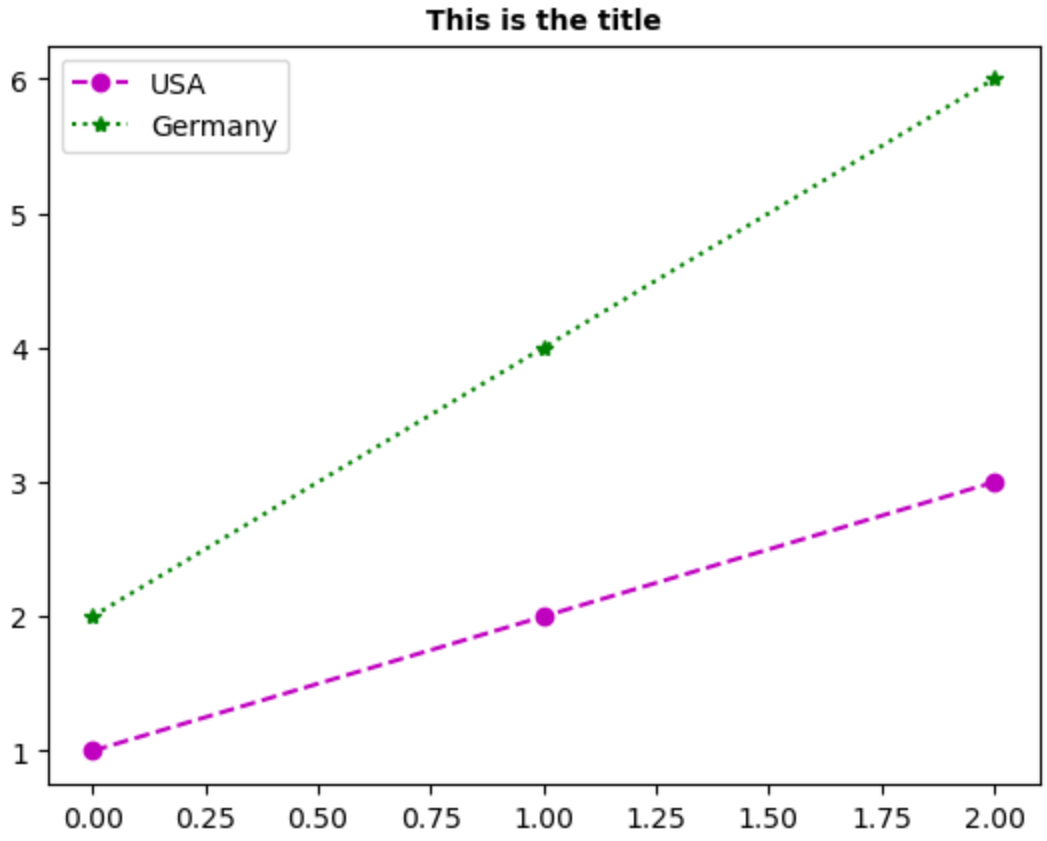
1.6)Adding Legends and Text:

In [23]:

```
plt.figure()
plt.plot([1,2,3], 'mo--',label = "USA" )
plt.plot([2,4,6], 'g*:', label = "Germany") #Adding the Legend
plt.legend()
plt.show()
```

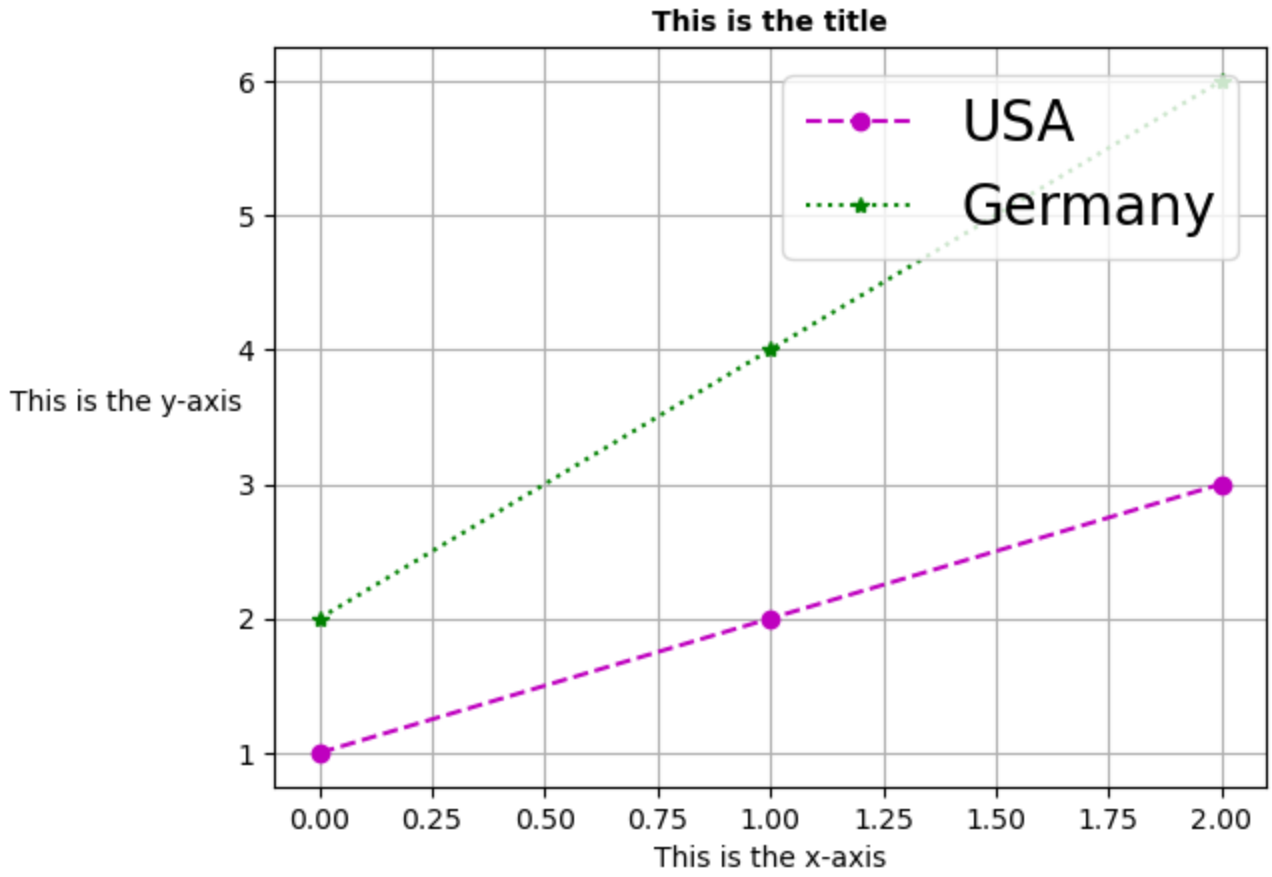


In [25]:
```
plt.figure()
plt.title("This is the title", fontweight = "bold", fontsize = 10) #Adding the Title
plt.plot([1,2,3], 'mo--',label = "USA" )
plt.plot([2,4,6], 'g*:', label = "Germany")
plt.legend()
plt.show()
```



In [35]:
```
plt.figure()
plt.title("This is the title", fontweight = "bold", fontsize = 10) #Adding the Title
plt.plot([1,2,3], 'mo--',label = "USA" )
```

```
plt.plot([2,4,6], 'g*:', label = "Germany")
plt.grid() #Adding grid()

plt.xlabel('This is the x-axis')
plt.ylabel('This is the y-axis', rotation = 0, labelpad = 40)

plt.legend(loc = 'upper right', fontsize = "20") #changing the location and Fontsize of
plt.show()
```



1.7) Plot stockprices' data:

In [39]:
```
stock_df = pd.read_csv("U:\\Users\\Reena.Shaw\\Downloads\\stockprices.csv")
```

In [40]:
```
stock_df.shape
```

Out[40]:
```
(6295, 7)
```

In [42]:
```
stock_df.head()
```

Out[42]:

	date	open	high	low	close	volume	Name
0	2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL
1	2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL
2	2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL
3	2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL
4	2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL

In [43]:
```
stock_df['Name'].unique()
```

Out[43]:
```
array(['AAPL', 'AMZN', 'FB', 'GOOGL', 'MSFT'], dtype=object)
```

In [44]:
```
google_df = stock_df[stock_df['Name']=='GOOGL']
google_df.shape
```

Out[44]:
```
(1259, 7)
```

In [45]:
```
apple_df = stock_df[stock_df['Name']=='AAPL']
apple_df.shape
```

Out[45]:
```
(1259, 7)
```

In [46]:
```
amazon_df = stock_df[stock_df['Name']=='AMZN']
amazon_df.shape
```

Out[46]:
```
(1259, 7)
```

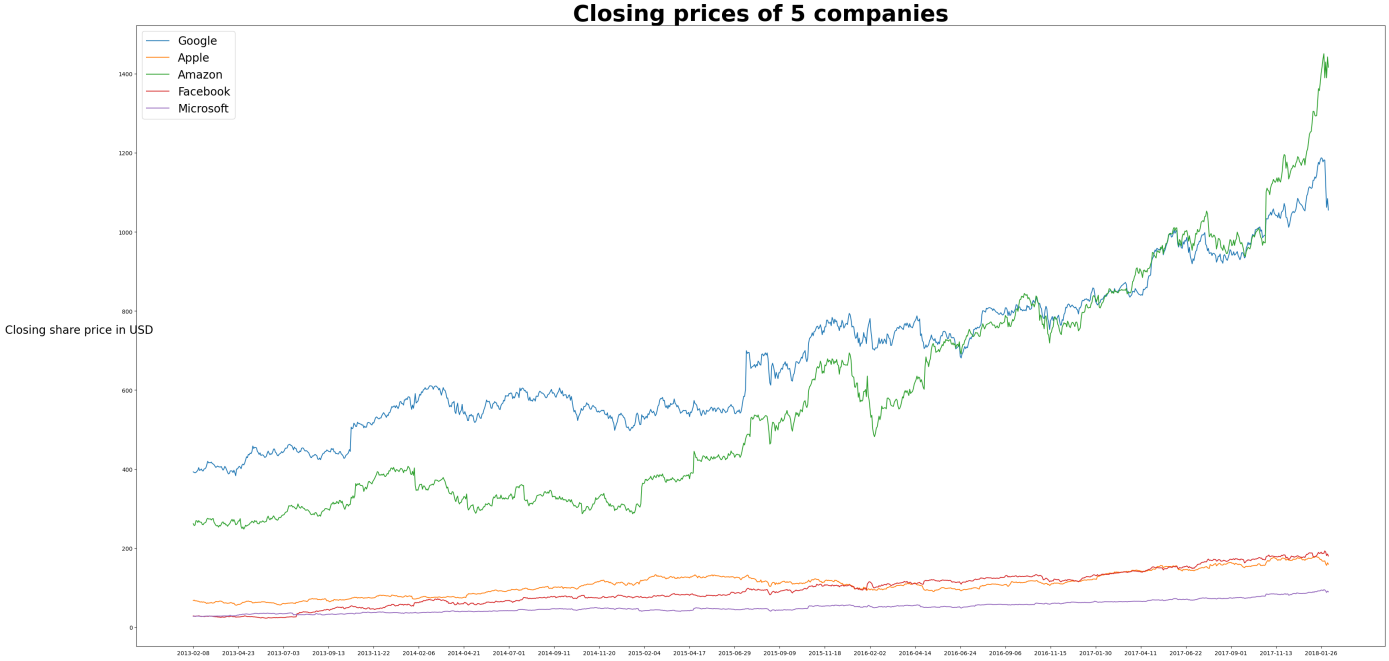
In [47]:
```
fb_df = stock_df[stock_df['Name']=='FB']
fb_df.shape
```

Out[47]:
```
(1259, 7)
```

In [48]:
```
msft_df = stock_df[stock_df['Name']=='MSFT']
msft_df.shape
```

Out[48]:
```
(1259, 7)
```

In [63]:
```
plt.figure(figsize = (40,20))

plt.plot('date','close', data = google_df, label = "Google")  #Google price
plt.plot('date','close', data = apple_df, label = "Apple")
plt.plot('date','close', data = amazon_df, label = "Amazon")
plt.plot('date','close', data = fb_df, label = "Facebook")
plt.plot('date','close', data = msft_df, label = "Microsoft")
plt.legend(fontsize = "20")

plt.ylabel('Closing share price in USD', rotation = 0, labelpad = 70, fontsize = "20")

plt.title('Closing prices of 5 companies', fontweight = "bold", fontsize = "40")

plt.xticks(np.arange(0,1260,50))


plt.show()
```



In [ ]:

In [ ]:

II) Bar graph:

2.1) Simple bar graph:

In [3]:
```
df = pd.read_csv('U:\\Users\\Reena.Shaw\\Downloads\CustomersTransactions2021.csv')
```

In [4]:
```
df.shape
```

Out[4]:
```
(34, 9)
```

In [5]:
```
df.head()
```

Out[5]:

	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking

In [7]:
```
df.head(15)
```

Out[7]:

	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking
5	1002	7	Cash	115.69	Bob Marley	4	ON	1	personal_banking
6	1002	8	Debit	21.37	Bob Marley	4	ON	1	personal_banking
7	1004	9	Cash	227.58	Taylor Swift	2	AB	2	hnw
8	1004	10	Credit	5.92	Taylor Swift	2	AB	2	hnw
9	1006	11	Debit	225.89	Stephen Smith	5	ON	4	hnw
10	1014	12	Cash	67.96	Delilah Avery	2	AB	3	personal_banking
11	1014	20	Debit	449.39	Delilah Avery	2	AB	3	personal_banking
12	1014	28	Cash	183.08	Delilah Avery	2	AB	3	personal_banking
13	1254	13	Debit	235.79	Johnson Cory	1	AB	4	hnw
14	1254	21	Cash	15.26	Johnson Cory	1	AB	4	hnw

2.3) 'Transaction_ID' v/s 'txn_total':

In [8]:
```
df.sort_values('id', inplace = True)
```

In [9]:
```
df.head(15)
```

Out[9]:

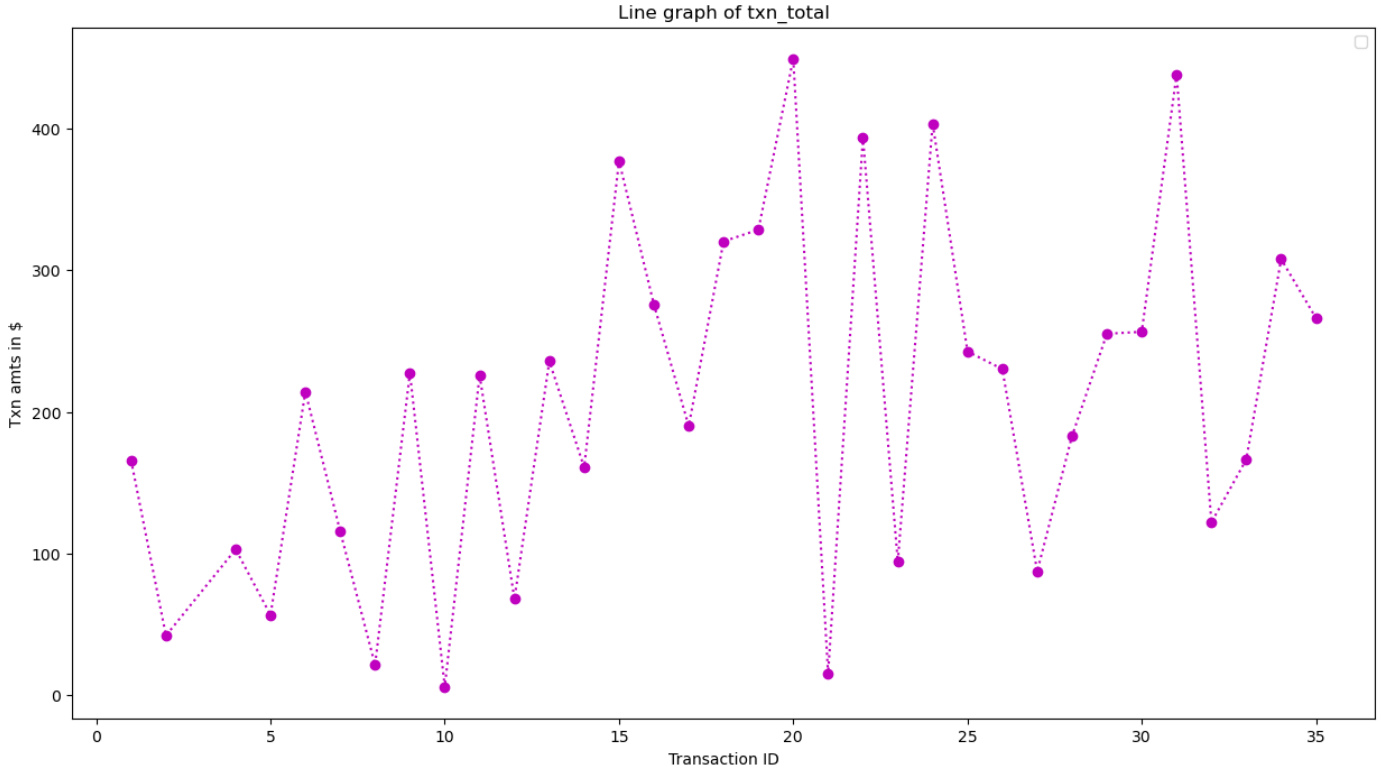
	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking
5	1002	7	Cash	115.69	Bob Marley	4	ON	1	personal_banking
6	1002	8	Debit	21.37	Bob Marley	4	ON	1	personal_banking
7	1004	9	Cash	227.58	Taylor Swift	2	AB	2	hnw
8	1004	10	Credit	5.92	Taylor Swift	2	AB	2	hnw
9	1006	11	Debit	225.89	Stephen Smith	5	ON	4	hnw
10	1014	12	Cash	67.96	Delilah Avery	2	AB	3	personal_banking
13	1254	13	Debit	235.79	Johnson Cory	1	AB	4	hnw
16	1285	14	Cash	160.89	Maria Alva	5	ON	2	wealth
19	1354	15	Credit	377.26	Jessica Fast	4	BC	5	personal_banking
22	1005	16	Debit	276.01	Mariah Anita Smith	2	BC	3	personal_banking

In [10]:
```
# Line graph:
```

In [15]:
```
plt.figure(figsize = (15,8))
plt.plot(df['id'], df['txn_total'], 'mo:')
plt.xlabel('Transaction ID')
plt.ylabel('Txn amts in $')
plt.title('Line graph of txn_total')
plt.legend()
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when legend() is called with no argument.
```

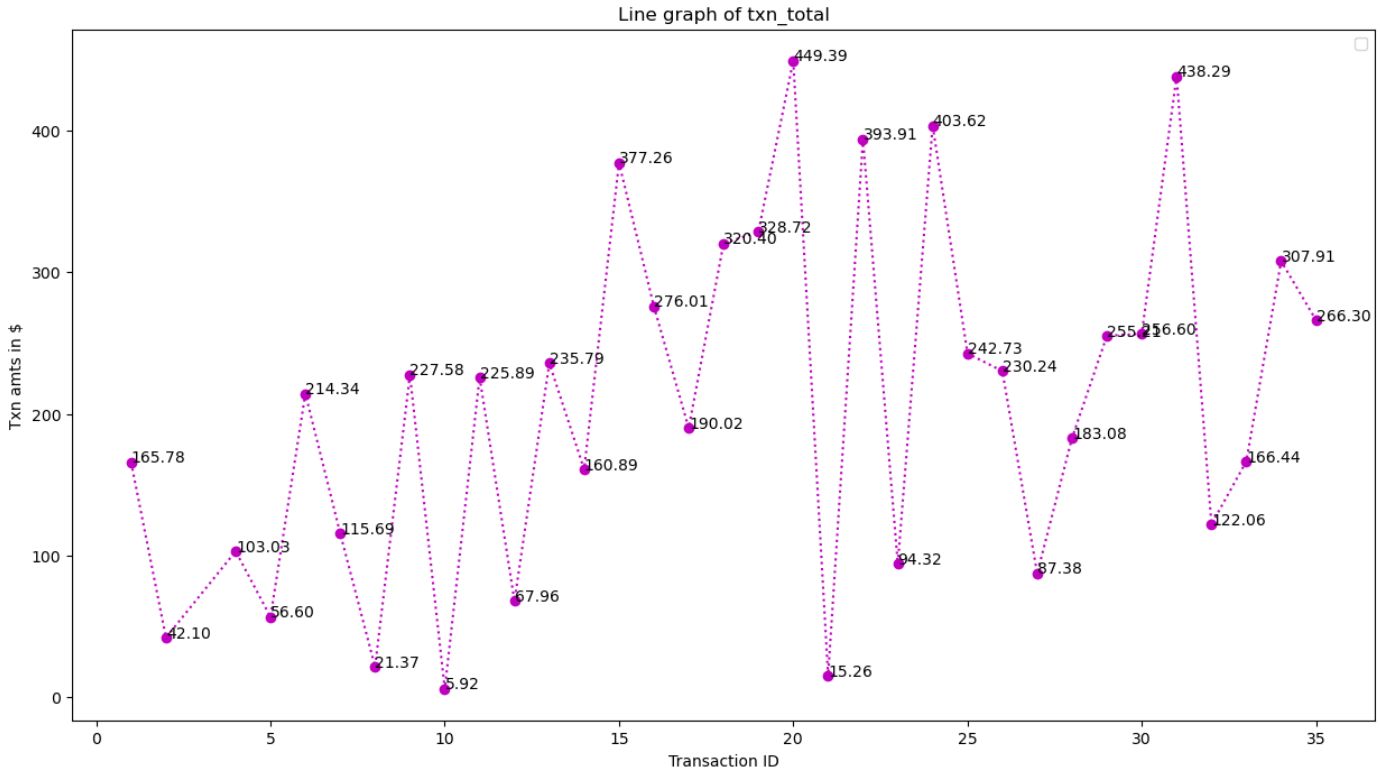
Out[15]:
```
<matplotlib.legend.Legend at 0x1fb167ea1d0>
```



In [16]:
```
# Annotate every marker:
```

In [20]:
```
plt.figure(figsize = (15,8))
plt.plot(df['id'], df['txn_total'], 'mo:')
plt.xlabel('Transaction ID')
plt.ylabel('Txn amts in $')
plt.title('Line graph of txn_total')
plt.legend()

for x,y in zip(df['id'],df['txn_total']):
    label = "{:.2f}".format(y)
    plt.annotate(label, (x,y))
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when legend() is called with no argument.
```



In [18]:
```
df[['id','txn_total']].head()
```

Out[18]:

	id	txn_total
0	1	165.78
1	2	42.10
2	4	103.03
3	5	56.60
4	6	214.34

III) Histogram:

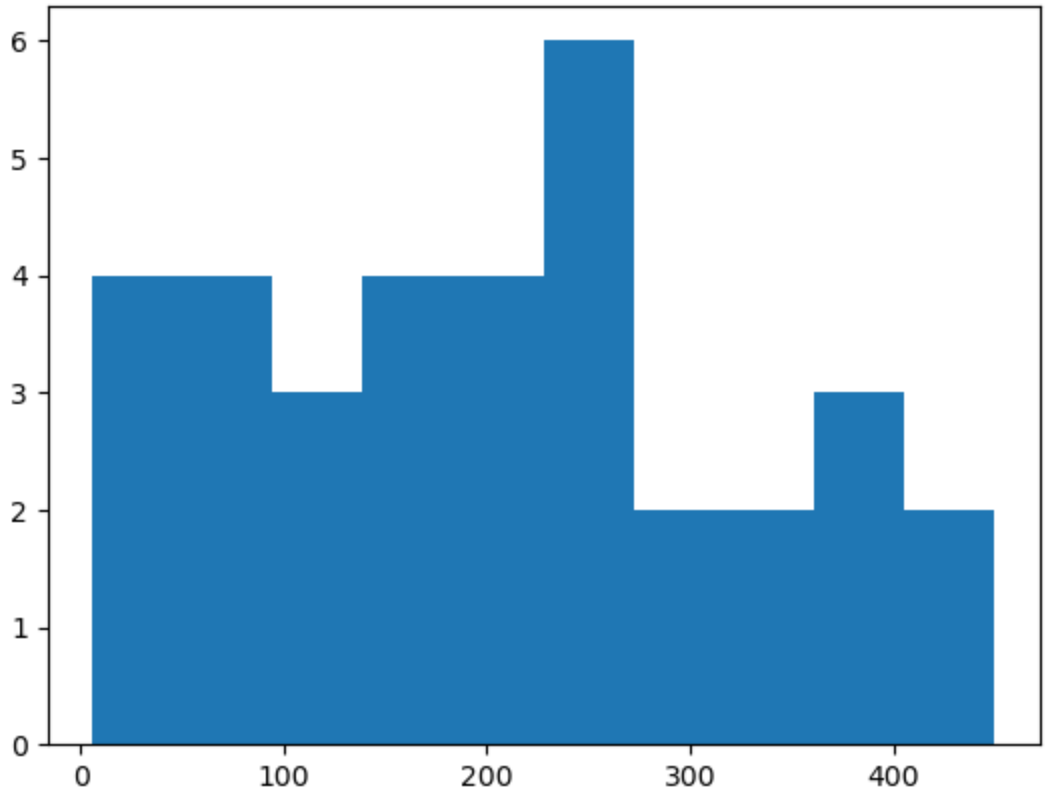
In [21]:
```
df.head()
```

Out[21]:

	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking

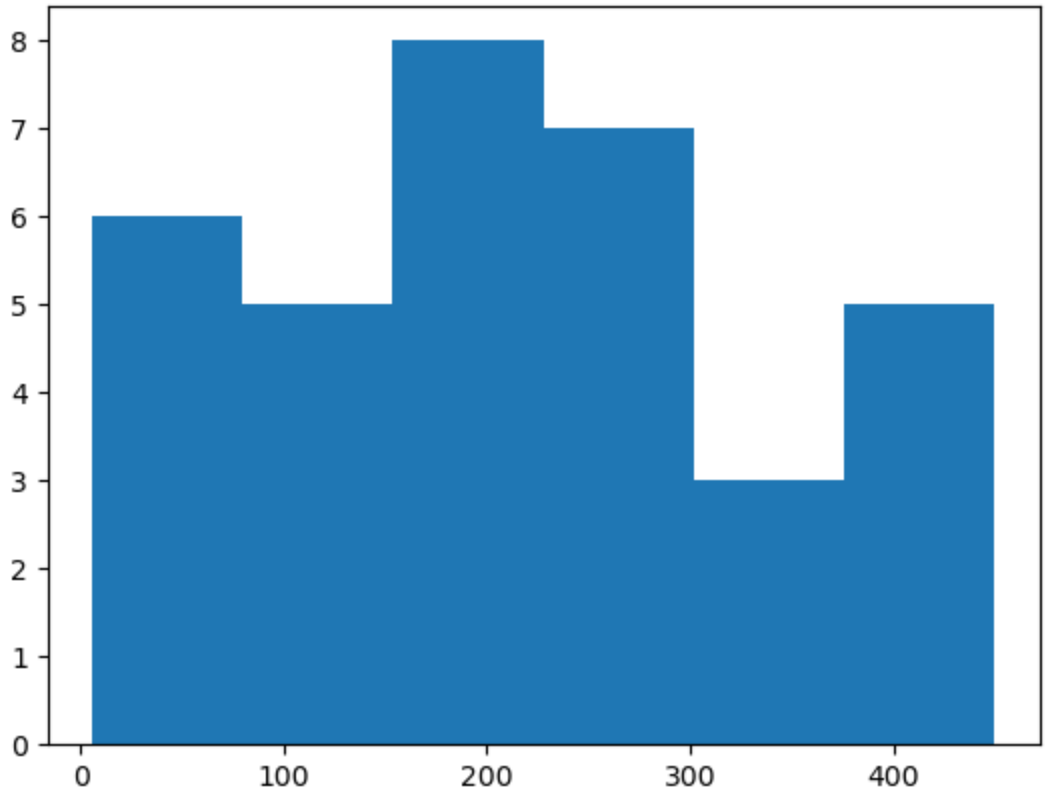
In [22]:
```
plt.hist(df['txn_total'])
```

Out[22]:
```
(array([4., 4., 3., 4., 4., 6., 2., 2., 3., 2.]),
 array([  5.92 ,  50.267,  94.614, 138.961, 183.308, 227.655, 272.002,
        316.349, 360.696, 405.043, 449.39 ]),
 <BarContainer object of 10 artists>)
```



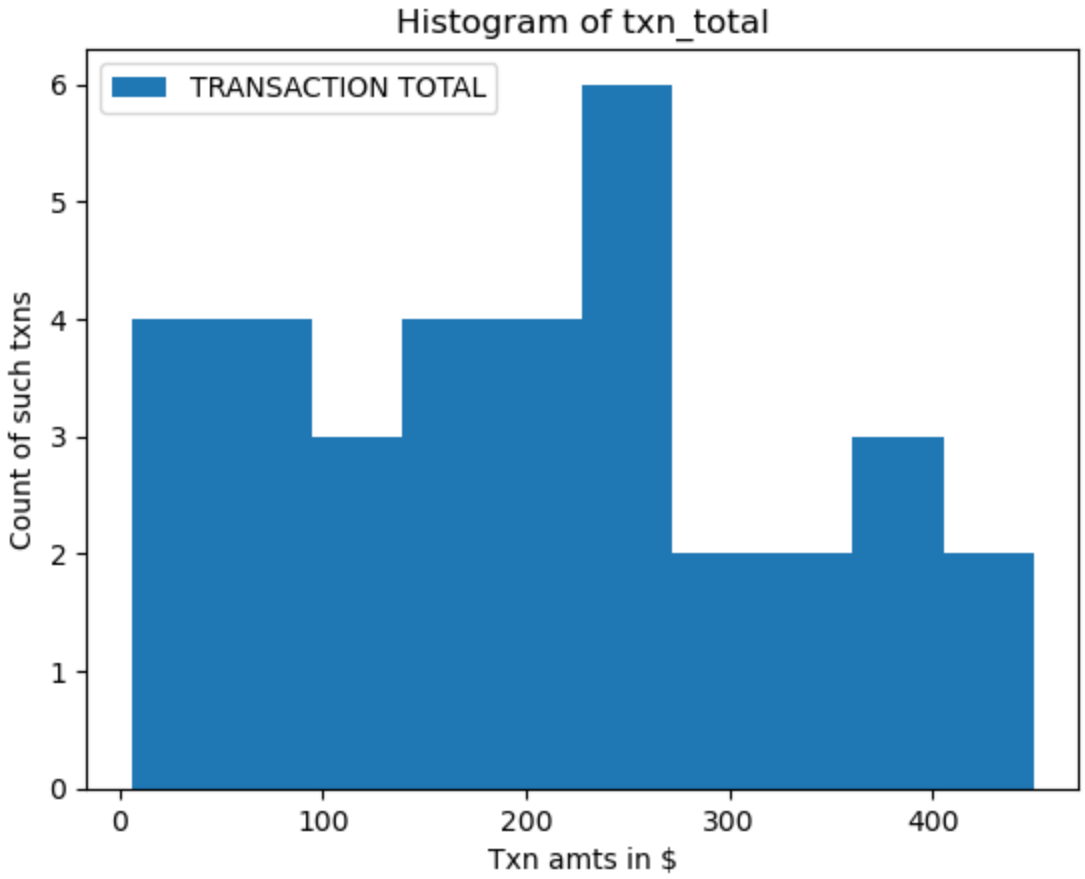
In [23]:
```
plt.hist(df['txn_total'], bins = 6)
```

Out[23]:
```
(array([6., 5., 8., 7., 3., 5.]),
 array([  5.92      ,  79.83166667, 153.74333333, 227.655     ,
        301.56666667, 375.47833333, 449.39      ]),
 <BarContainer object of 6 artists>)
```



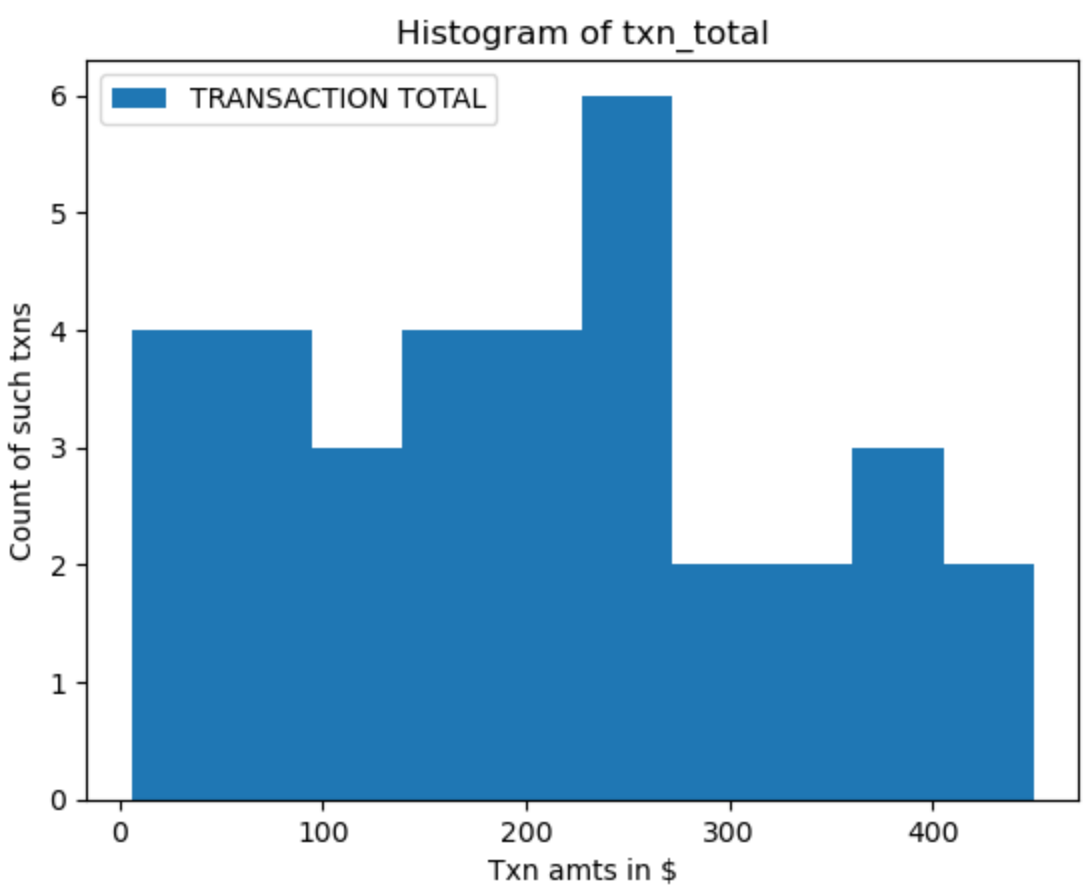
In [24]:
```
plt.hist(df['txn_total'], label = "TRANSACTION TOTAL")
plt.xlabel('Txn amts in $')
plt.ylabel('Count of such txns')
plt.title('Histogram of txn_total')
plt.legend()
```

Out[24]: <matplotlib.legend.Legend at 0x1fb176b9f00>



In [25]:
```
n, bins, patches = plt.hist(df['txn_total'], label = "TRANSACTION TOTAL")
plt.xlabel('Txn amts in $')
plt.ylabel('Count of such txns')
plt.title('Histogram of txn_total')
plt.legend()
```

Out[25]: <matplotlib.legend.Legend at 0x1fb17733880>



In [26]:
```
n
```

Out[26]:
```
array([4., 4., 3., 4., 4., 6., 2., 2., 3., 2.])
```

In [27]:
```
bins
```

Out[27]:
```
array([  5.92 ,  50.267,  94.614, 138.961, 183.308, 227.655, 272.002,
       316.349, 360.696, 405.043, 449.39 ])
```

In [28]:
```
patches
```

Out[28]:
```
<BarContainer object of 10 artists>
```

IV) Boxplots:

In [30]:
```
df.describe()
```

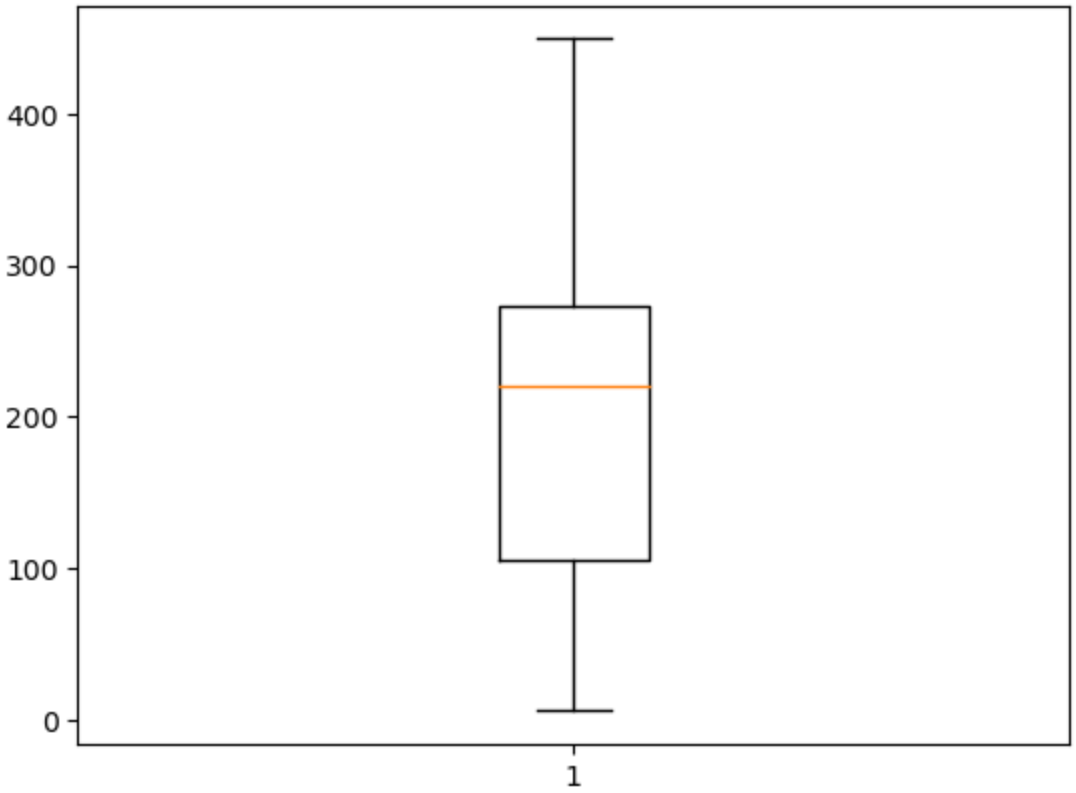
Out[30]:

	customer_id	id	txn_total	Products	Relationship
count	34.000000	34.000000	34.000000	34.000000	34.000000
mean	1175.411765	18.441176	207.296765	3.088235	3.441176
std	211.165900	10.057987	124.283187	1.311202	2.488655
min	1001.000000	1.000000	5.920000	1.000000	1.000000
25%	1004.250000	10.250000	106.195000	2.000000	2.000000
50%	1014.000000	18.500000	220.115000	3.000000	3.000000
75%	1336.750000	26.750000	273.582500	4.000000	4.750000
max	1585.000000	35.000000	449.390000	5.000000	10.000000

In [ ]:

In [29]:
```
plt.boxplot(df['txn_total'])
```

Out[29]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1fb17806e00>,
  <matplotlib.lines.Line2D at 0x1fb17838460>],
 'caps': [<matplotlib.lines.Line2D at 0x1fb17838700>,
  <matplotlib.lines.Line2D at 0x1fb178389a0>],
 'boxes': [<matplotlib.lines.Line2D at 0x1fb17838040>],
 'medians': [<matplotlib.lines.Line2D at 0x1fb17838c40>],
 'fliers': [<matplotlib.lines.Line2D at 0x1fb17838ee0>],
 'means': []}
```



In [31]:
```
# min = Q1 - 1.5*IQR
# max = Q3 + 1.5*IQR
```

In [ ]:

In [ ]:

V) Subplots:

In [32]:
```
df.head()
```

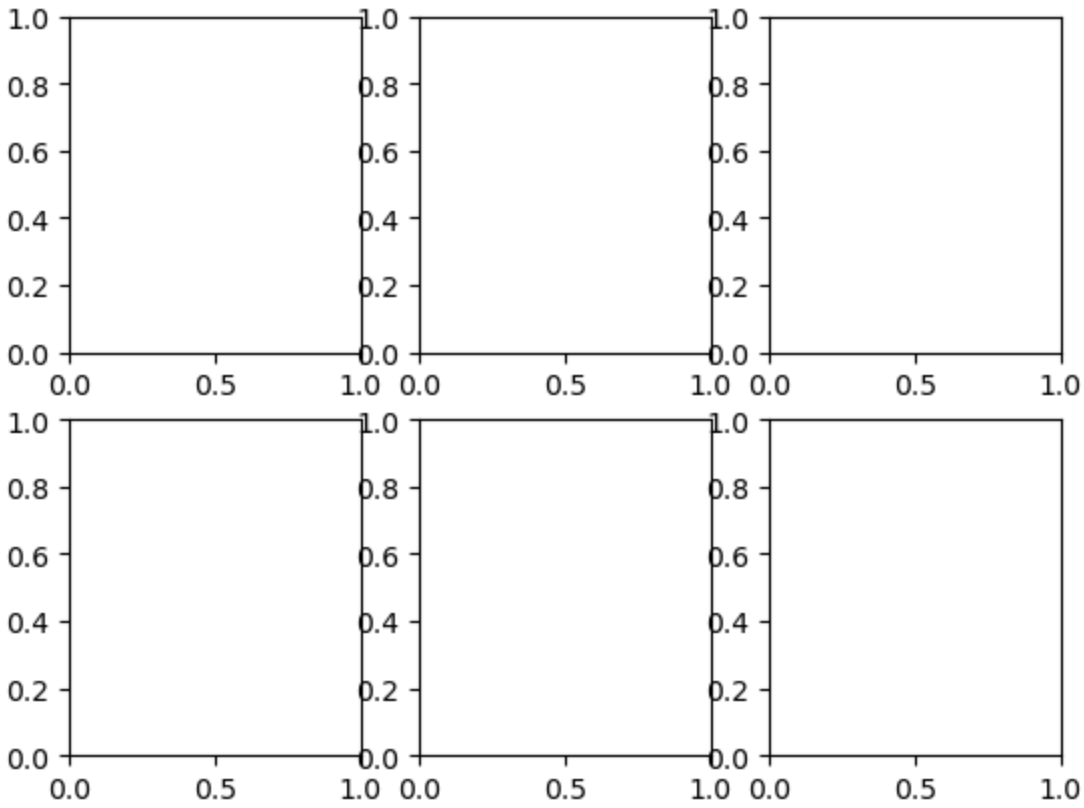
Out[32]:

	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking

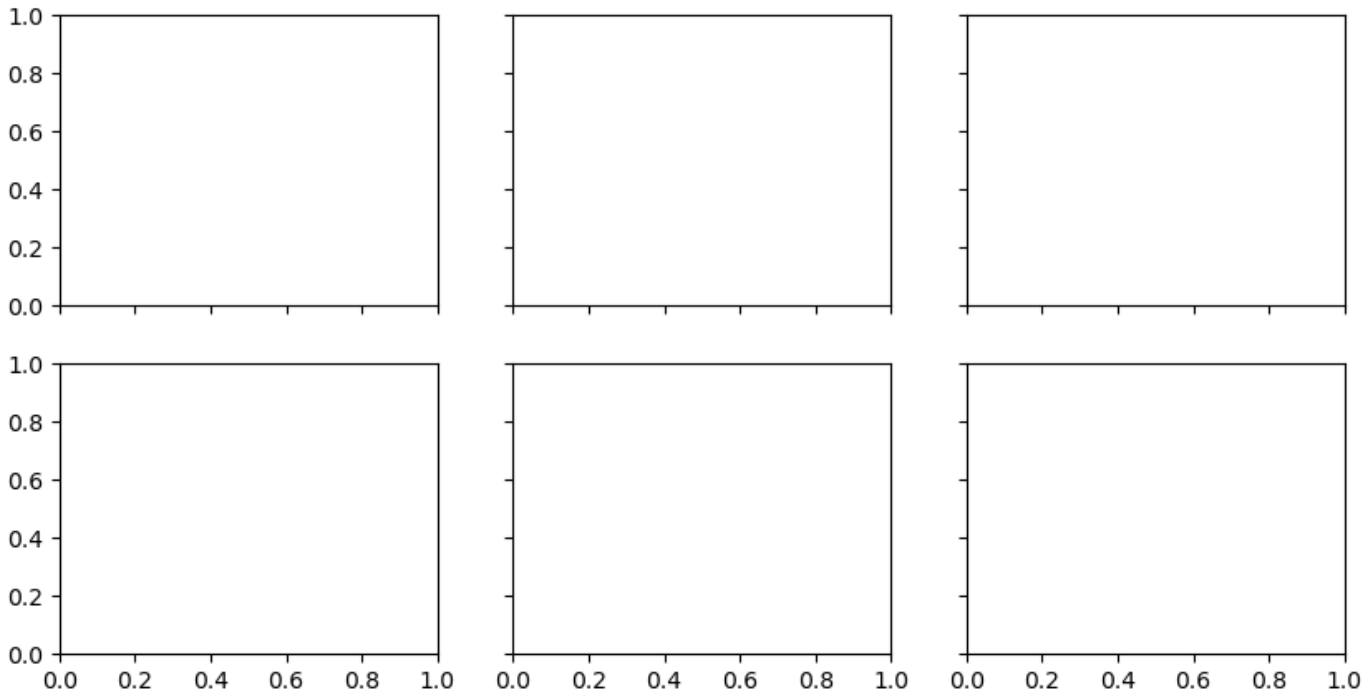
In [33]:
```
df['Customers'].unique()
```

Out[33]:
```
array(['Johnny Awesome', 'Bob Marley', 'Taylor Swift', 'Stephen Smith',
       'Delilah Avery', 'Johnson Cory', 'Maria Alva', 'Jessica Fast',
       'Mariah Anita Smith', 'Robert Optimus', 'Steven Prime',
       'Stephanie Element'], dtype=object)
```

In [34]:
```
fig, ((ax1, ax2, ax3), (ax4, ax5, ax6)) = plt.subplots(2,3)
```

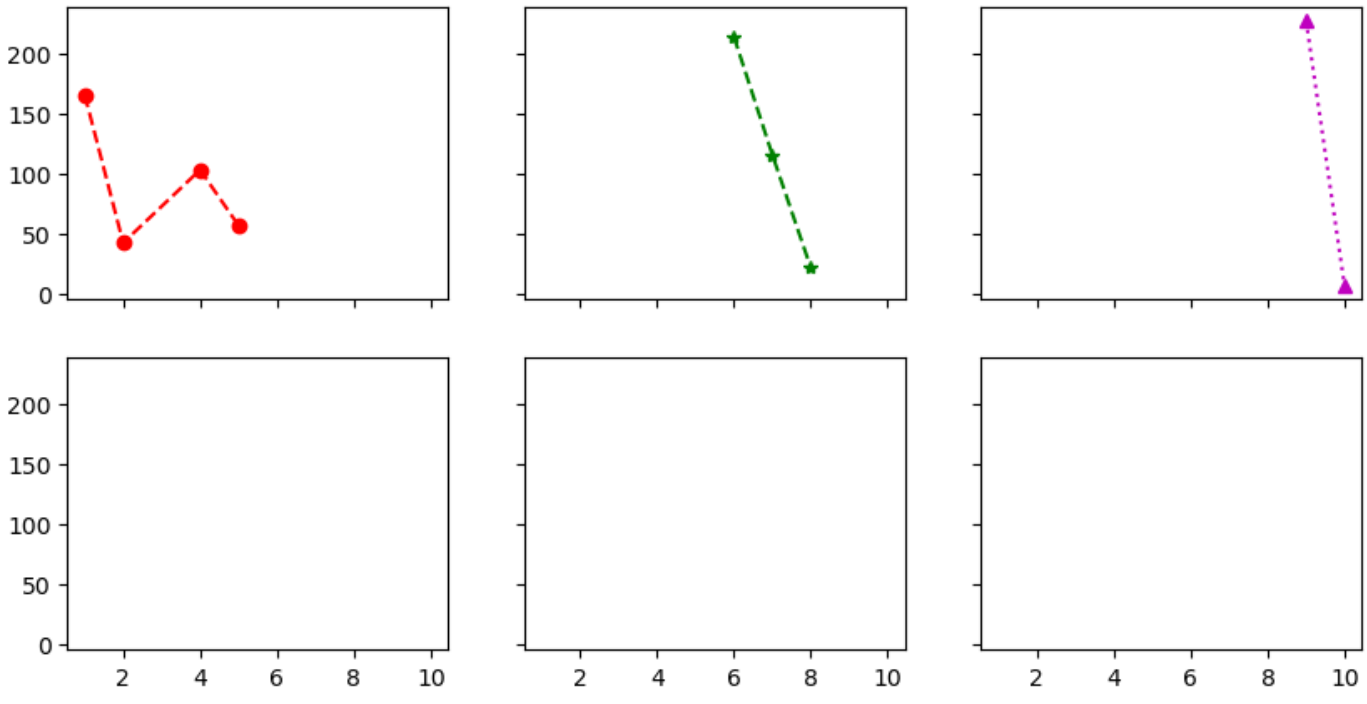


In [36]:
```
fig, ((ax1, ax2, ax3), (ax4, ax5, ax6)) = plt.subplots(2,3, sharex = True,
                                                       sharey= True,
                                                      figsize = (10,5))
```



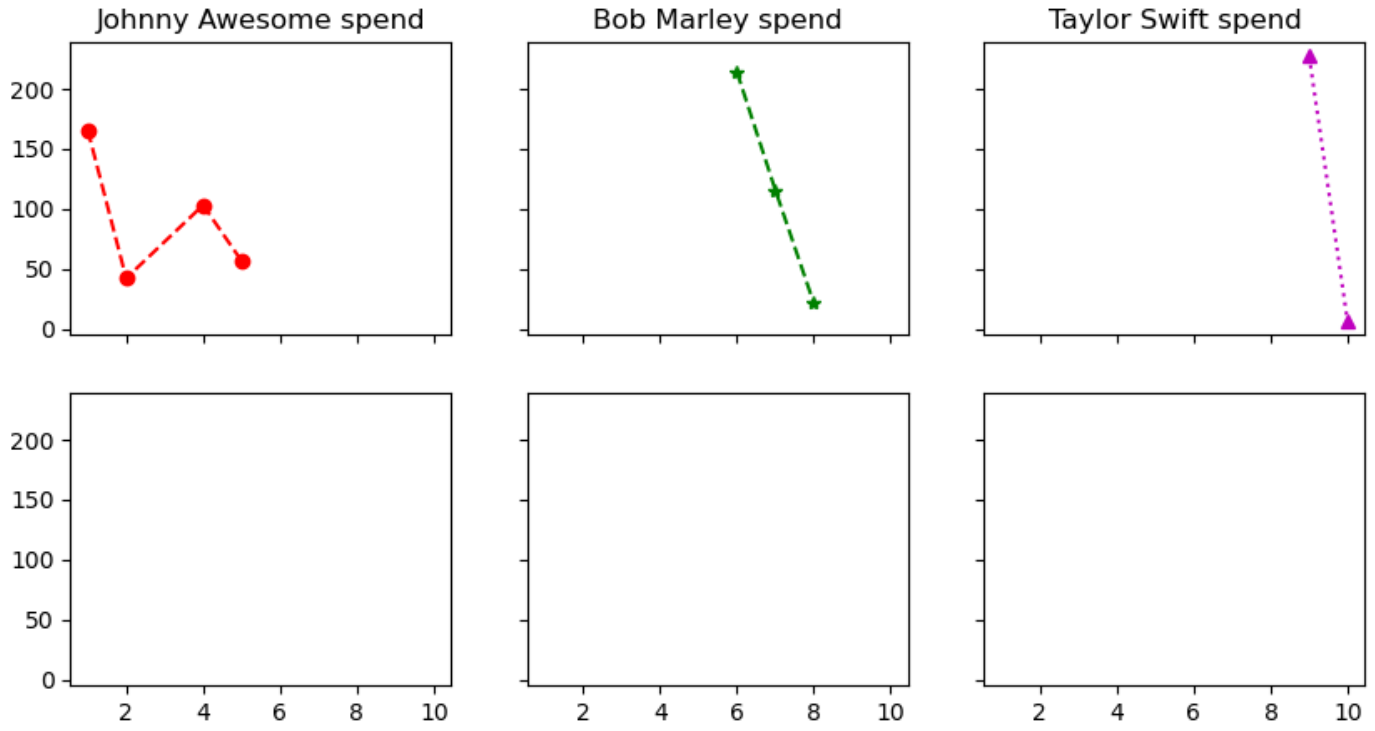
In [39]:
```
fig, ((ax1, ax2, ax3), (ax4, ax5, ax6)) = plt.subplots(2,3, sharex = True,
                                                       sharey= True,
                                                      figsize = (10,5))

ax1.plot(df[df['Customers']=='Johnny Awesome']['id'],
         df[df['Customers']=='Johnny Awesome']['txn_total'],
         'ro--')

ax2.plot(df[df['Customers']=='Bob Marley']['id'],
         df[df['Customers']=='Bob Marley']['txn_total'],
         'g*--')

ax3.plot(df[df['Customers']=='Taylor Swift']['id'],
         df[df['Customers']=='Taylor Swift']['txn_total'],
         'm^:')
```

Out[39]:
```
[<matplotlib.lines.Line2D at 0x1fb199806d0>]
```



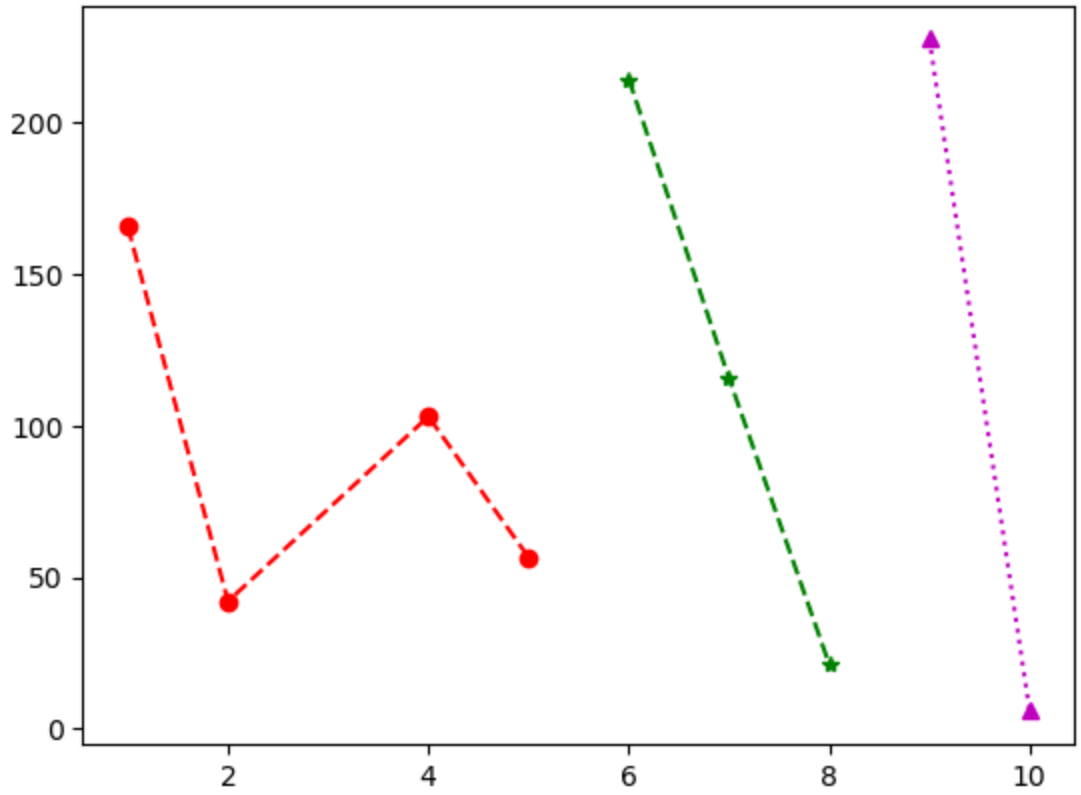
In [40]:
```
fig, ((ax1, ax2, ax3), (ax4, ax5, ax6)) = plt.subplots(2,3, sharex = True,
                                                       sharey= True,
                                                      figsize = (10,5))

ax1.plot(df[df['Customers']=='Johnny Awesome']['id'],
         df[df['Customers']=='Johnny Awesome']['txn_total'],
         'ro--')

ax2.plot(df[df['Customers']=='Bob Marley']['id'],
         df[df['Customers']=='Bob Marley']['txn_total'],
         'g*--')

ax3.plot(df[df['Customers']=='Taylor Swift']['id'],
         df[df['Customers']=='Taylor Swift']['txn_total'],
         'm^:')

ax1.set_title('Johnny Awesome spend')
ax2.set_title('Bob Marley spend')
ax3.set_title('Taylor Swift spend')
```

Out[40]:
```
Text(0.5, 1.0, 'Taylor Swift spend')
```



In [41]:
```
plt.plot(df[df['Customers']=='Johnny Awesome']['id'],
         df[df['Customers']=='Johnny Awesome']['txn_total'],
         'ro--')

plt.plot(df[df['Customers']=='Bob Marley']['id'],
         df[df['Customers']=='Bob Marley']['txn_total'],
         'g*--')

plt.plot(df[df['Customers']=='Taylor Swift']['id'],
         df[df['Customers']=='Taylor Swift']['txn_total'],
         'm^:')
```

Out[41]:
```
[<matplotlib.lines.Line2D at 0x1fb17754fd0>]
```



VI) Bar graphs:

In [42]:
```
df.head()
```

Out[42]:

	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking

In [43]:
```
df['Category'].unique()
```

Out[43]:
```
array(['wealth', 'personal_banking', 'hnw'], dtype=object)
```

In [44]:
```
df.groupby('Category').agg({'txn_total':'sum'}) #o/p as a DF
```

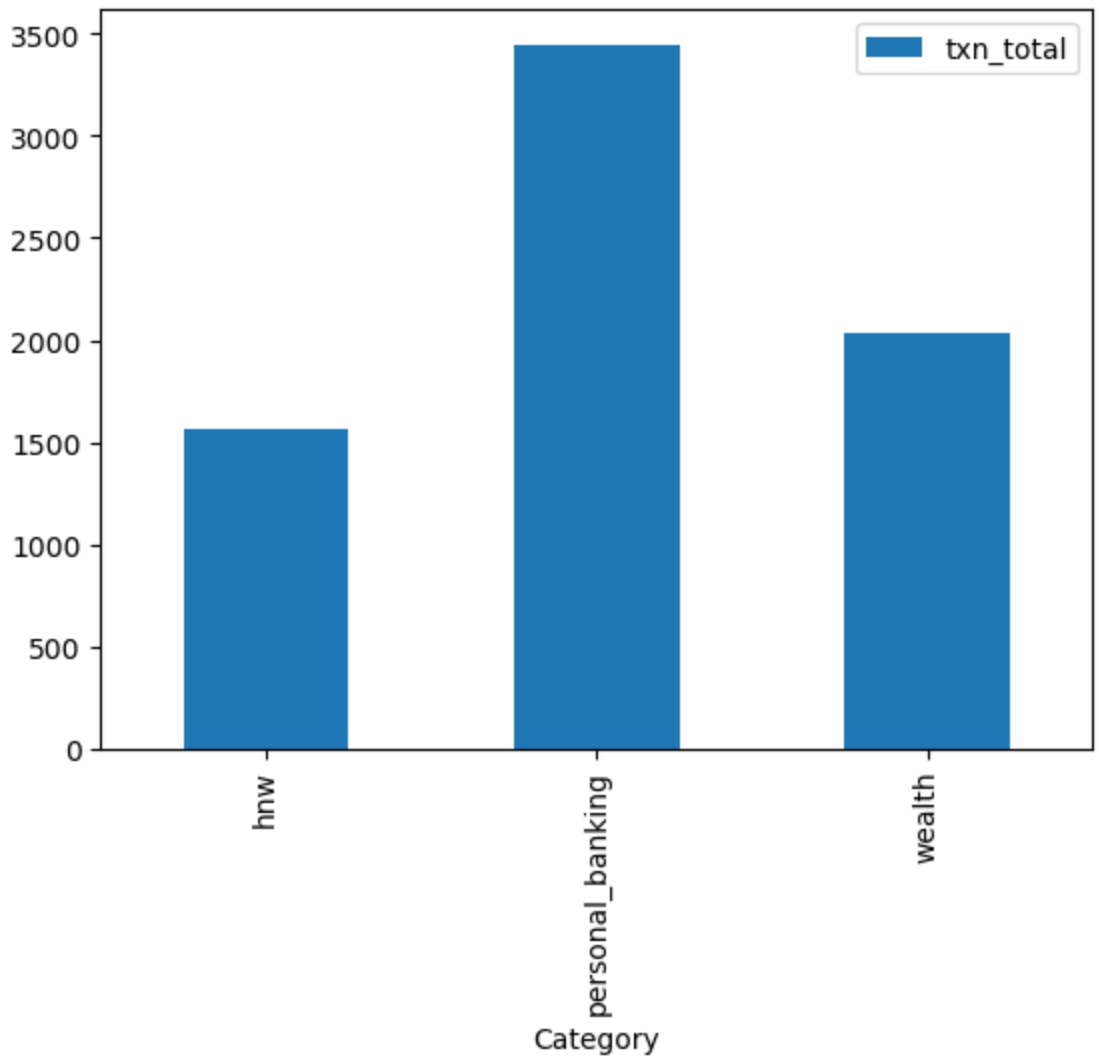
Out[44]:

Category	txn_total
hnw	1564.84
personal_banking	3445.79
wealth	2037.46

In [49]:
```
# 6.1) Simple Bar graph:
```

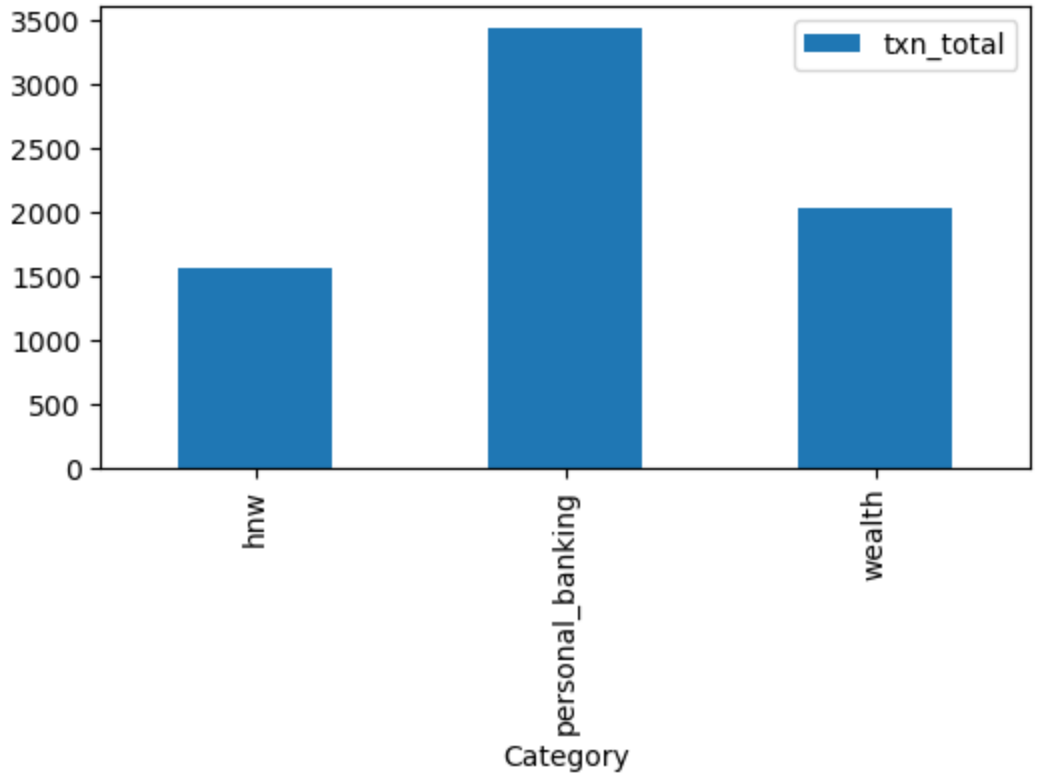
In [45]:
```
df.groupby('Category').agg({'txn_total':'sum'}).plot(kind = 'bar')
```

Out[45]:
```
<Axes: xlabel='Category'>
```



In [47]:
```
df.groupby('Category').agg({'txn_total':'sum'}).plot(kind = 'bar', figsize =(6,3))
```

Out[47]:
```
<Axes: xlabel='Category'>
```



In [48]:
```
df.groupby('Category')['txn_total'].sum() #o/p as a Series
```

Out[48]:
```
Category
hnw                  1564.84
personal_banking     3445.79
wealth               2037.46
Name: txn_total, dtype: float64
```

In [50]:
```
# 6.2) Stacked bar graph:
```

In [51]:
```
df.head()
```

Out[51]:

	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking

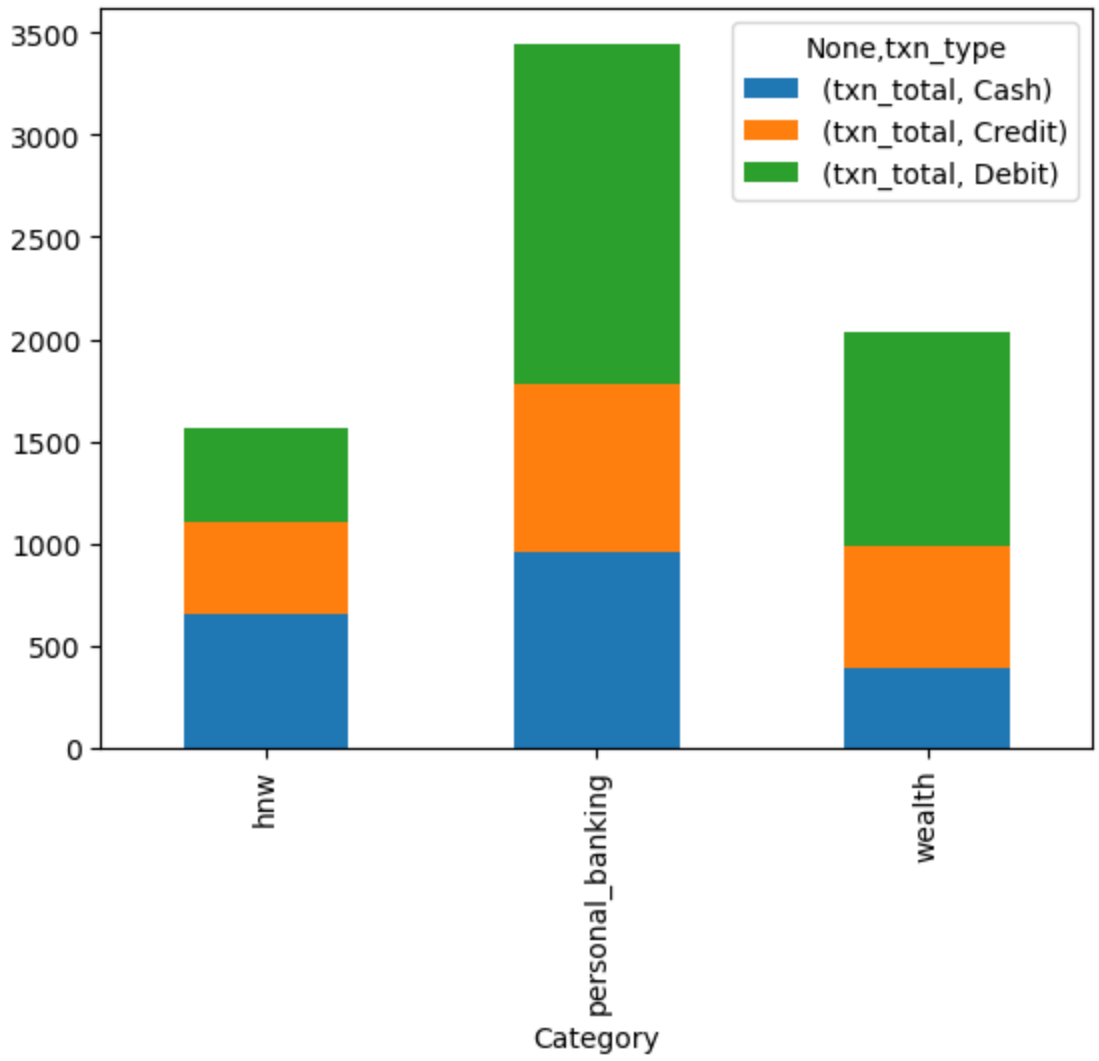
In [52]:
```
df.groupby(['Category','txn_type']).agg({'txn_total':'sum'})
```

Out[52]:

		txn_total
Category	txn_type	
hnw	Cash	652.01
	Credit	451.15
	Debit	461.68
personal_banking	Cash	961.75
	Credit	815.55
	Debit	1668.49
wealth	Cash	391.13
	Credit	595.64
	Debit	1050.69

In [54]:
```
df.groupby(['Category','txn_type']).agg({'txn_total':'sum'}).\
unstack('txn_type').plot(kind='bar', stacked = True)
```

Out[54]:
```
<Axes: xlabel='Category'>
```

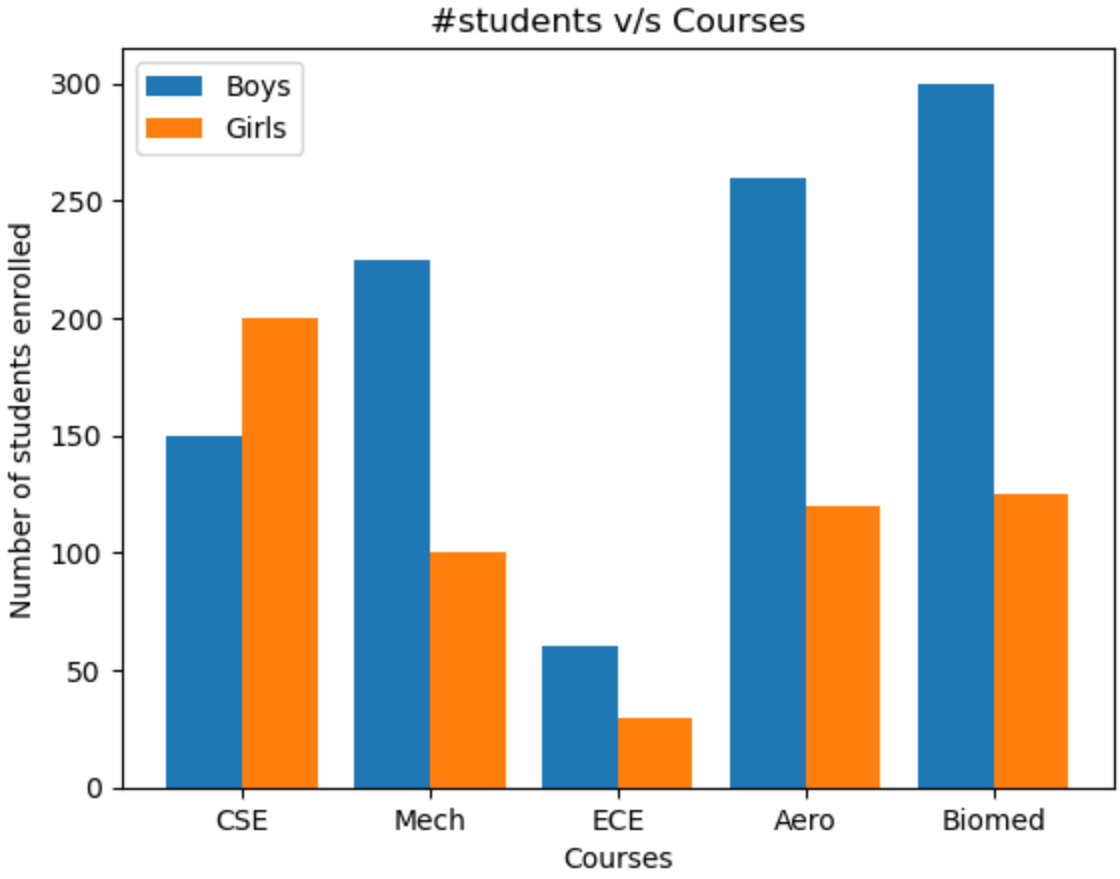


6.3) Grouped bar graph/ Multiple-category bar graph:

In [ ]:
```
# Syntax:
# plt.bar(xaxis_categories, bar_heights, width_per_bar)
```

In [63]:
```
w = 0.4 #width_per_bar

course_list = ["CSE", "Mech", "ECE", "Aero","Biomed"]
bar1 = np.arange(len(course_list)) # bar1 centred at 0,1,2,3,4
bar2 = [(i+w) for i in bar1]

boyscount_barheights = [150, 225, 60, 260, 300]
girlscount_barheights = [200, 100, 30, 120, 125]



plt.bar(bar1, boyscount_barheights, w, label = "Boys")
plt.bar(bar2, girlscount_barheights, w, label = "Girls")

plt.xticks(bar1 + w/2, course_list)

plt.xlabel("Courses")
plt.ylabel("Number of students enrolled")
plt.title("#students v/s Courses")
plt.legend()
```

Out[63]:
```
<matplotlib.legend.Legend at 0x1fb19bdfe20>
```



In [56]:
```
course_list = ["CSE", "Mech", "ECE", "Aero","Biomed"]
len(course_list)
```

Out[56]:
```
5
```

In [57]:
```
np.arange(len(course_list))
```

Out[57]:
```
array([0, 1, 2, 3, 4])
```

VII) Scatterplot:

In [64]:
```
import seaborn as sns
```

In [65]:
```
df.head()
```

Out[65]:

	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking

In [66]:
```
len(df)
```

Out[66]:
```
34
```

In [68]:
```
df['acct_balance'] = np.random.randint(0,1000, size = len(df))
```

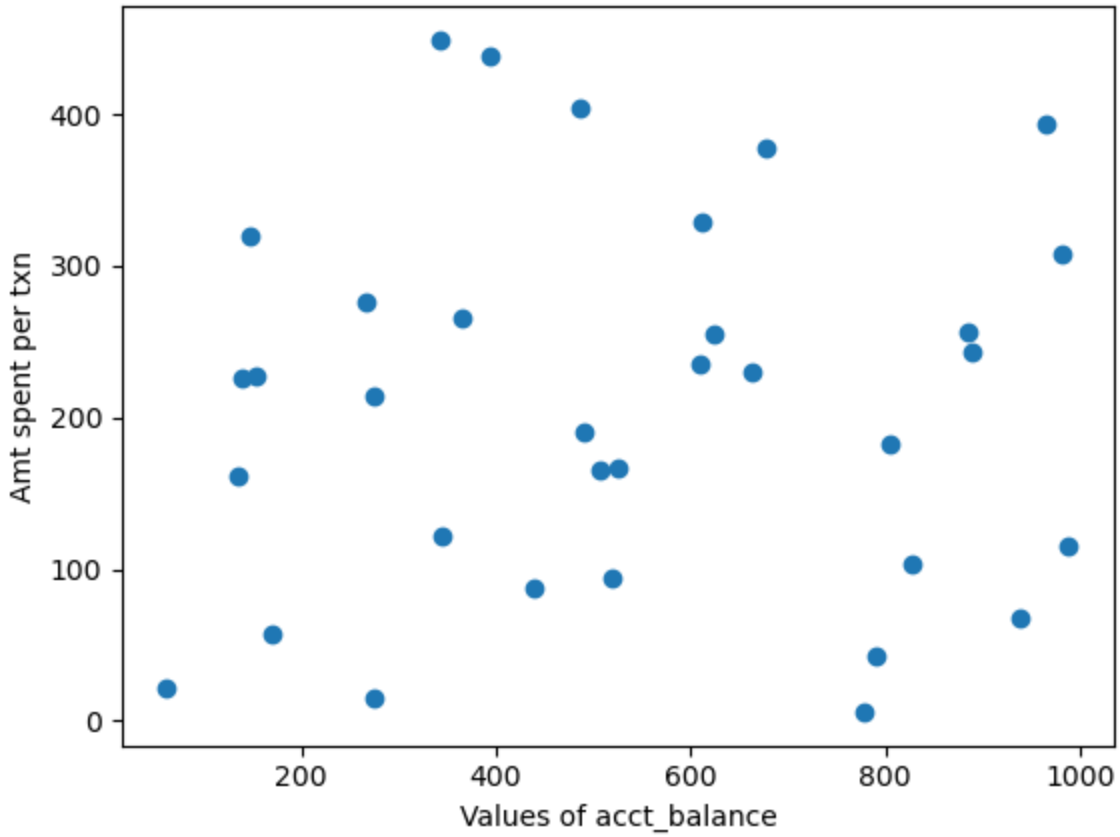
In [69]:
```
df.head()
```

Out[69]:

	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category	ac
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth	
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth	
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth	
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth	
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking	

In [71]:
```
plt.scatter(df['acct_balance'],df['txn_total'] )
plt.xlabel('Values of acct_balance')
plt.ylabel('Amt spent per txn')
```

Out[71]:
```
Text(0, 0.5, 'Amt spent per txn')
```

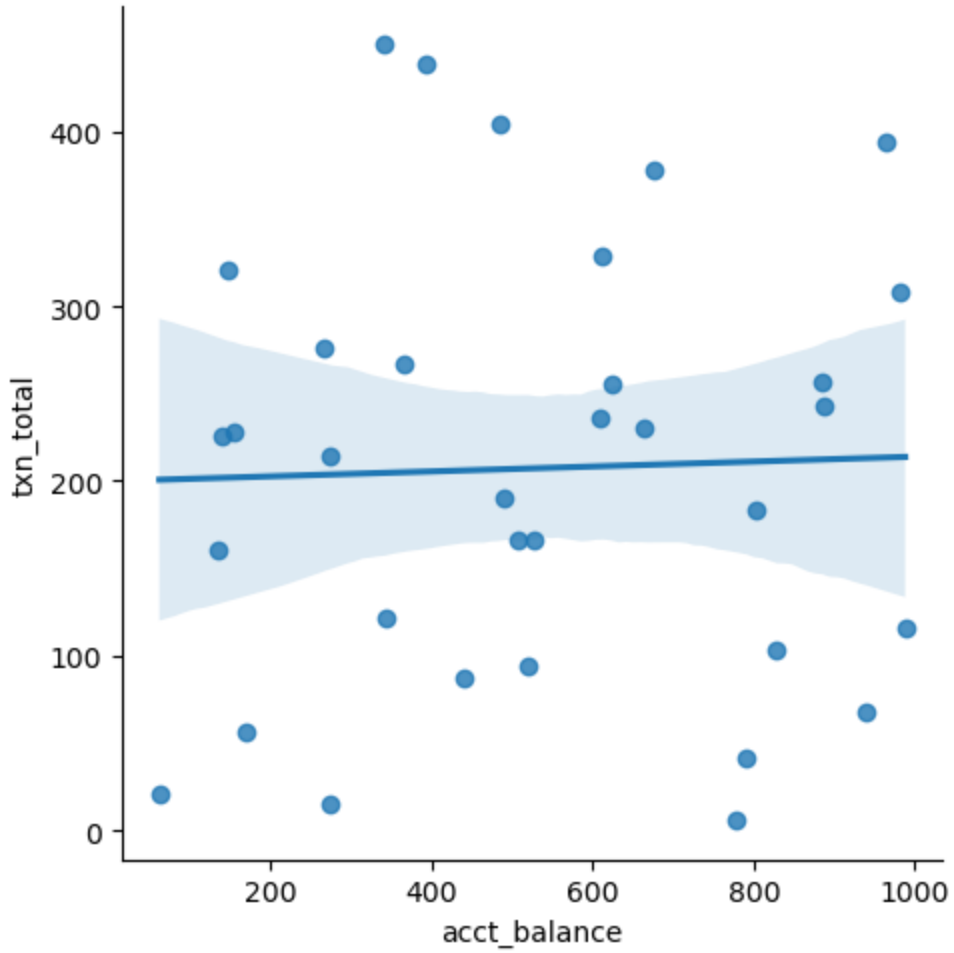


VIII) Seaborn:

In [74]:
```
# 8.1) scatterplot:
```

In [72]:
```
sns.lmplot(x = 'acct_balance', y = 'txn_total', data = df)
```

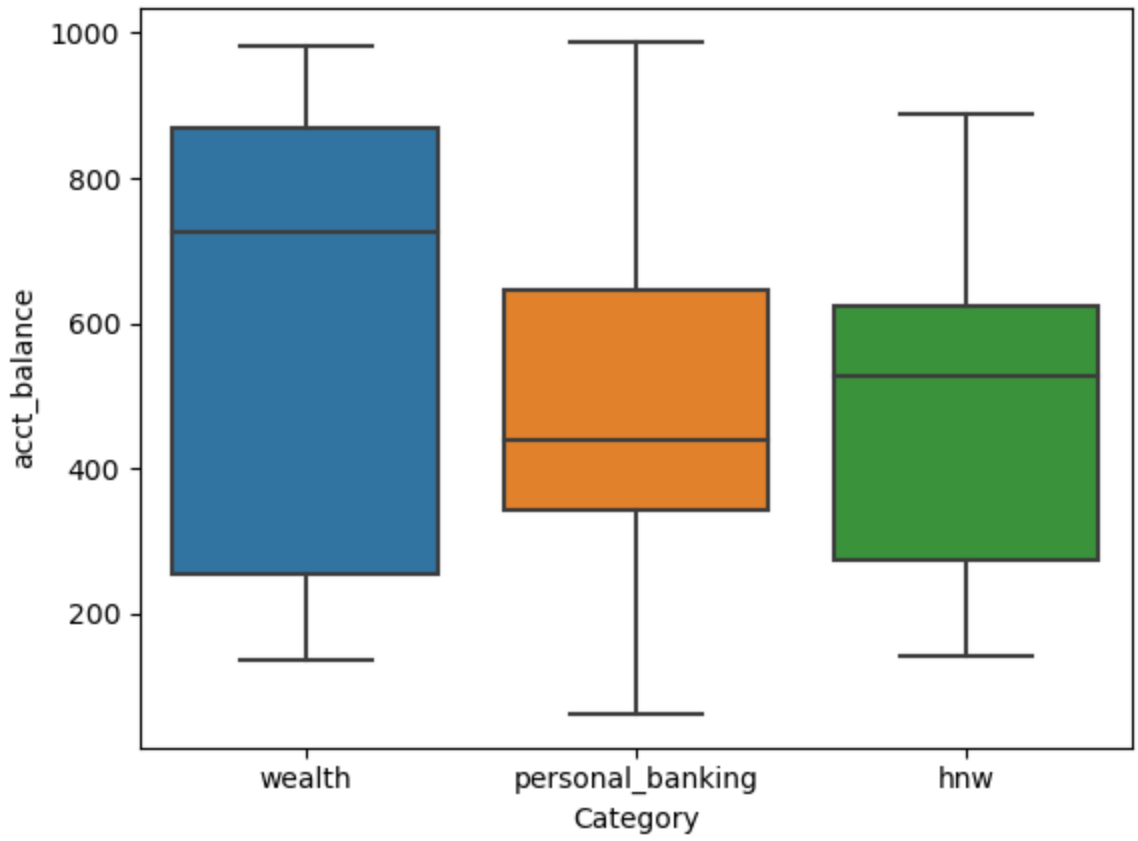
Out[72]:
```
<seaborn.axisgrid.FacetGrid at 0x1fb1bf8ed10>
```



In [75]:
```
# 8.2) boxplot:
```

In [76]:
```
sns.boxplot(x = 'Category', y = 'acct_balance', data = df)
```

Out[76]:
```
<Axes: xlabel='Category', ylabel='acct_balance'>
```

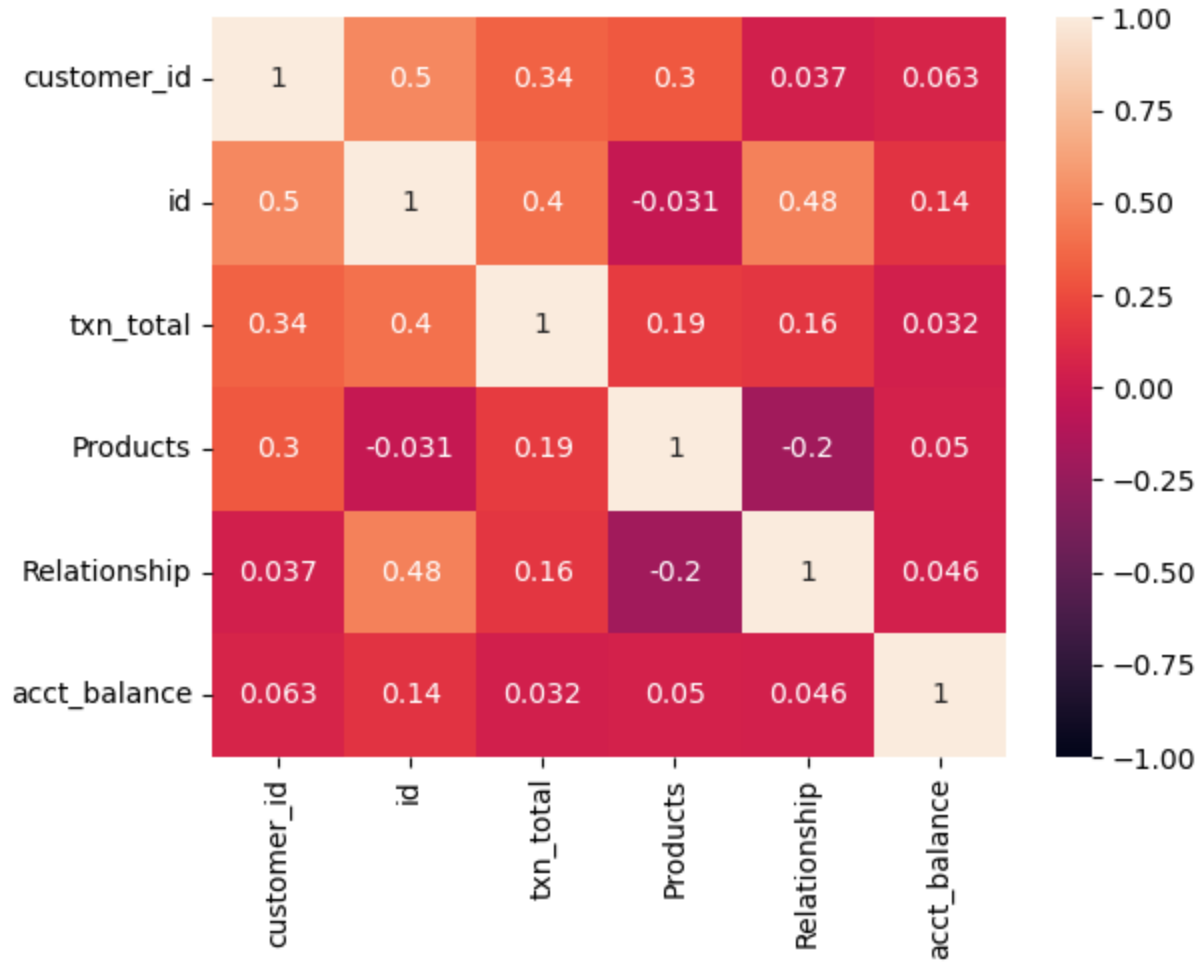


In [77]:
```
# 8.3) Heatmap:
```

In [78]:
```
sns.heatmap(df.corr(), annot = True, vmin = -1, vmax = 1)
```

```
U:\Users\Reena.Shaw\AppData\Local\Temp\ipykernel_1840\1230598035.py:1: FutureWarning: Th
e default value of numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value of numeric_only to
silence this warning.
  sns.heatmap(df.corr(), annot = True, vmin = -1, vmax = 1)
```

Out[78]:
```
<Axes: >
```



In [79]:
```
# 8.4) Bubble chart:
```

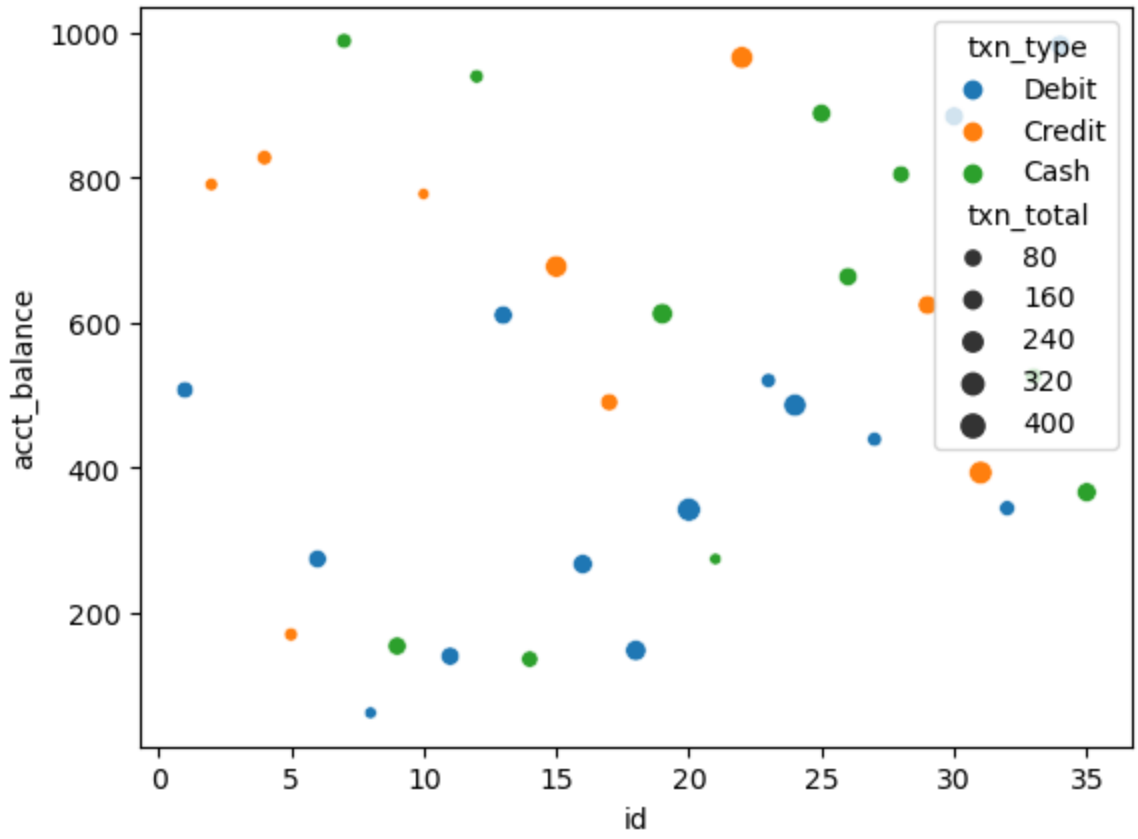
In [81]:
```
df.head()
```

Out[81]:

	customer_id	id	txn_type	txn_total	Customers	Products	Province	Relationship	Category	ac
0	1001	1	Debit	165.78	Johnny Awesome	3	ON	1	wealth	
1	1001	2	Credit	42.10	Johnny Awesome	3	ON	1	wealth	
2	1001	4	Credit	103.03	Johnny Awesome	3	ON	1	wealth	
3	1001	5	Credit	56.60	Johnny Awesome	3	ON	1	wealth	
4	1002	6	Debit	214.34	Bob Marley	4	ON	1	personal_banking	

In [82]:
```
sns.scatterplot(x = 'id', y = 'acct_balance', data = df, hue = 'txn_type',
                size = 'txn_total')
```

Out[82]:
```
<Axes: xlabel='id', ylabel='acct_balance'>
```



In [ ]: