John Como Shopify Data Science Challenge Question 1

All work is shown first and final solutions are at the bottom.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
# Read in csv file
shopify = pd.read_csv('Datasets/2019 Winter Data Science Intern Challenge D
```

In [3]:
```
shopify.shape
```

Out[3]:
```
(5000, 7)
```

In [6]:
```
shopify.head()
```

Out[6]:

	order_id	shop_id	user_id	order_amount	total_items	payment_method	created_at
0	1	53	746	224	2	cash	2017-03-13 12:36:56
1	2	92	925	90	1	cash	2017-03-03 17:38:52
2	3	44	861	144	1	cash	2017-03-14 4:23:56
3	4	18	935	156	1	credit_card	2017-03-26 12:43:37
4	5	18	883	156	1	credit_card	2017-03-01 4:35:11

In [5]:
```
shopify.describe()
```

Out[5]:

	order_id	shop_id	user_id	order_amount	total_items
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.00000
mean	2500.500000	50.078800	849.092400	3145.128000	8.78720
std	1443.520003	29.006118	87.798982	41282.539349	116.32032
min	1.000000	1.000000	607.000000	90.000000	1.00000
25%	1250.750000	24.000000	775.000000	163.000000	1.00000
50%	2500.500000	50.000000	849.000000	284.000000	2.00000
75%	3750.250000	75.000000	925.000000	390.000000	3.00000
max	5000.000000	100.000000	999.000000	704000.000000	2000.00000

Above we can see how the average amount of $3145.13 was reached. This method of averaging takes the feature or column total and divides by the number of rows. In this dataset, it is important to take the sum of the total orders and divide them by the total_items in each purchase.

```
In [8]: # check for nulls
        shopify.isnull().sum()
```

```
Out[8]: order_id          0
        shop_id           0
        user_id           0
        order_amount      0
        total_items       0
        payment_method    0
        created_at        0
        dtype: int64
```

Excellent!!

```
In [9]: shopify.dtypes
```

```
Out[9]: order_id           int64
        shop_id            int64
        user_id            int64
        order_amount       int64
        total_items        int64
        payment_method    object
        created_at        object
        dtype: object
```

Since our two important features order_amount and total_items are integer features we can sum and divide

```
In [15]: TOTAL_ITEMS = shopify['total_items'].sum()
         print(TOTAL_ITEMS)
```

```
43936
```

```
In [16]: TOTAL_ORDER_AMOUNT = shopify['order_amount'].sum()
         print(TOTAL_ORDER_AMOUNT)
```

```
15725640
```

```
In [14]: TOTAL_ORDER_AMOUNT / TOTAL_ITEMS
```

```
Out[14]: 357.92152221412965
```

```
In [19]: # This calculation is to once again verify the $3145.13 from prior
         TOTAL_ORDER_AMOUNT / 5000
```

```
Out[19]: 3145.128
```

In [25]:
```
sales_by_store = shopify.groupby('shop_id')['order_amount'].sum()
print(sales_by_store)
```

```
shop_id
1      13588
2       9588
3      14652
4      13184
5      13064
       ...
96     16830
97     15552
98     14231
99     18330
100     8547
Name: order_amount, Length: 100, dtype: int64
```

In [26]:
```
sales_by_store.sort_values()
```

Out[26]:
```
shop_id
92         6840
32         7979
56         8073
100        8547
2          9588
         ...
6         22627
81        22656
89        23128
78      2263800
42     11990176
Name: order_amount, Length: 100, dtype: int64
```

In [ ]:
```

```

Final Solutions

a) The average taken was the total order amount divided by the number of rows, incorrectly assuming each row correlated to one item being purchased. A better way to evaluate this data would be taking all the order amounts and dividing them by the total number of orders as done above.

b) A metric I would report which would interesting is sales by region, or in our case by store location. Performing a simple .groupby('shop_id')['order_amount'].sum() would tell us the performance of a desired column in the dataset as done above. Seeing which stores outperform others could save the business valuable dollars as some locations may not be worth staying open.

c) From my calculation above, it is clear that there is a large range in store preformance, with shop_id 92 generating 6840 in sales while shop_id 42 is generating 11990176 in sales. There are many things not given that make it more difficult to give possible reasons for this such as location, employees, etc., but it is helpful to know some stores are producing much more than others.

```
In [ ]:
```