In [80]:
```
import pandas as pd
import numpy as np
```

Shopify Data Science Intern Challenge

By : Siddharth Mittal

Question-1

Let's begin by reading the data first. I downloaded the data provided as a CSV file, renamed it as 'data.csv' and placed it in the working directory.

In [85]:
```
data = pd.read_csv('data.csv')
data.head(n = 10)
```

Out[85]:

	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
5	6	58	882	138	1	credit_card	2017-03-14 15:25:01
6	7	87	915	149	1	cash	2017-03-01 21:37:57
7	8	22	761	292	2	cash	2017-03-08 2:05:38
8	9	64	914	266	2	debit	2017-03-17 20:56:50
9	10	52	788	146	1	credit_card	2017-03-30 21:08:26

a)

Let's extract the `order_amount` feature as a `numpy` array so we can perform our analysis on it.

Let's also sort the array.

It was very clear after looking at the Excel spreadsheet that the `order_amount` feature contains quite a number of outliers in it.

In [86]:
```
orders = data['order_amount'].to_numpy()
orders = -np.sort(-orders)      #Sort in descending order
```

In [87]:
```
orders_above_10k = np.sum(np.logical_and(orders > 10**4, orders < 10**5))
```

In [88]:
```
orders_above_100k = np.sum(orders > 10**5)
```

In [89]:
```
print(orders_above_10k)
print(orders_above_100k)
```

```
44
19
```

There are 44 orders whose value is between $10,000 and \$100,000 and 19 orders whose value exceed $100,000.

It's these 63 orders which cause the AOV to be really high and not representative of the data since mean is very sensitive to outliers.

To calculate the average value of orders where we know there are outliers present, we can compute the $n\%$ trimmed mean, where we sort the data and discard $n\%$ of the highest values and $n\%$ of the lowest values and then calculate the mean.

For this question, let $n = 2$.
$2\% \text{ of } 5000 = 100$
This means to calculate the trimmed mean, we discard the first 100 and last 100 values in the sorted list of orders.

In [75]:
```
trimmed_mean = np.mean(orders[100:4900])
print('Trimmed mean = {:.2f}'.format(trimmed_mean))
```

```
Trimmed mean = 301.75
```

Hence, the new AOV comes out to be **$301.75** which is much more representative of the average value of orders rather than $3145.13.

However, we are also discarding 4% of the data to find this value.

b)

A much better metric which is not sensitive to outliers is the **median**.

c)

In [90]:
```
median = np.median(orders)
print('Median order value = ${}'.format(median))
```

```
Median order value = $284.0
```

The median order value is **$284**.

Question-2

a)

```
SELECT COUNT(OrderDate) AS NumberOfOrders
FROM Orders
WHERE ShipperID = (
    SELECT ShipperID
    FROM Shippers
    WHERE ShipperName = 'Speedy Express'
)
GROUP BY ShipperID;
```

Query result = **54**

b)

```
SELECT LastName
FROM Employees
WHERE EmployeeID = (
    SELECT EmployeeID
    FROM Orders
    GROUP BY EmployeeID
    ORDER BY COUNT(*) DESC
    LIMIT 1
);
```

Query result = **Peacock**

c)

```
SELECT ProductName
FROM Products
WHERE ProductID = (
    SELECT ProductID
    FROM OrderDetails
    WHERE OrderID IN (
        SELECT OrderID
        FROM Orders
        WHERE CustomerID IN (
            SELECT CustomerID
            FROM Customers
            WHERE Country = 'Germany'
        )
    )
    GROUP BY ProductID
    ORDER BY SUM(Quantity) DESC
    LIMIT 1
);
```

Query result = **Boston Crab Meat**