Question 1:

In [1]:

```
import pandas as pd


shopify_data = pd.read_excel("C:/Users/Mouna/Desktop/Jobs/Shopify/2019 Winter Data Science Intern Challenge Data Set.xlsx")
shopify_data.head()
```

Out[1]:

	order_id	shop_id	user_id	order_amount	total_items	payment_method	created_at
0	1	53	746	224	2	cash	2017-03-13 12:36:56.190
1	2	92	925	90	1	cash	2017-03-03 17:38:51.999
2	3	44	861	144	1	cash	2017-03-14 04:23:55.595
3	4	18	935	156	1	credit_card	2017-03-26 12:43:36.649
4	5	18	883	156	1	credit_card	2017-03-01 04:35:10.773

In [2]:

```
shopify_data.describe()
```

Out[2]:

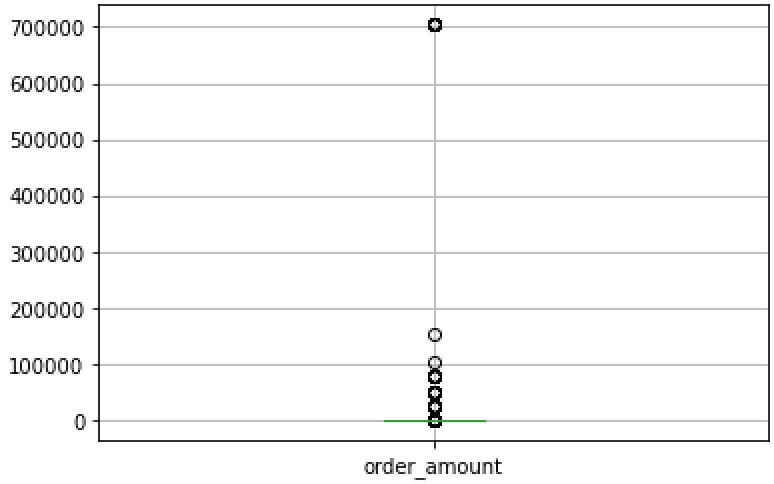
	order_id	shop_id	user_id	order_amount	total_items
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.00000
mean	2500.500000	50.078800	849.092400	3145.128000	8.78720
std	1443.520003	29.006118	87.798982	41282.539349	116.32032
min	1.000000	1.000000	607.000000	90.000000	1.00000
25%	1250.750000	24.000000	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

In [3]:

```
shopify_data.boxplot(column='order_amount')
```

Out[3]:

```
<AxesSubplot:>
```



Analysis of order_amount

Following observations are derived from the data -

1) min and 25% values of total_items is 1, but their corresponding order_amount is different, i.e min order_amount is 90 and 25% order_amount is 163. Since all the shops are selling the same shoe model, this means each shop sells the same shoe for a different price

2) There are observations with total_items as 2000 whose order_amount is 704000. This is seen from the max value above. Due to this outlier, the mean of the order_amount is shifting to a higher value of 3145.128. This gives a wrong notion of the montly average being of this value. The standard deviation 41282.5 signifies the values deviating from the mean by a large amount. In such scenarios with the presence of outliers, we can use median to determine an average value. Another approach can be to discard the values beyond +/-1.5 IQR(Inter quartile range) and then take the mean value.

a) Think about what could be going wrong with our calculation. Think about a better way to evaluate this data.

The total number of observations in this dataset is 5000, hence the mean is being calculated as sum of order_amount divided by 5000 which is giving a value of 3145.128. However, to obtain the average order we should ideally calculate as total order amount divided by total no. of orders.

In [4]:
```
shopify_data['order_amount'].mean() #This is the mean of the order_amount from the dataset which is wrong. We need to divide by tot
```

Out[4]:
```
3145.128
```

b) What metric would you report for this dataset?

In [5]:
```
total_order_amount = shopify_data['order_amount'].sum()
```

```
total_number_of_items = shopify_data['total_items'].sum()
print(total_order_amount)
print(total_number_of_items)
```

```
15725640
43936
```

Here, we can see that the total no of items is 43936. This would be the denominator for finding the average order value.

c) What is its value?

```
In [6]: print(total_order_amount/total_number_of_items)
```

```
357.92152221412965
```

Therefore, the average order value is $357.9 for an observed month.

Question 2:

a) How many orders were shipped by Speedy Express in total?

SELECT COUNT(*) FROM Orders JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID WHERE Shippers.ShipperName='Speedy Express';

Answer - 54

b) What is the last name of the employee with the most orders?

SELECT Employees.LastName FROM Orders JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID GROUP BY LastName ORDER BY COUNT(*) DESC LIMIT 1;

Answer - Peacock

c) What product was ordered the most by customers in Germany?

CREATE VIEW GermanyCustomerOrders AS SELECT Orders.OrderID, Customers.Country, OrderDetails.Quantity, Products.ProductName FROM Orders, OrderDetails JOIN Customers ON Orders.CustomerID=Customers.CustomerID JOIN Products ON OrderDetails.ProductID=Products.ProductID WHERE Country='Germany';

CREATE VIEW ProductOrders AS SELECT ProductName, Quantity, COUNT(*) as 'Orders' FROM GermanyCustomerOrders GROUP BY ProductName;

SELECT ProductName FROM ProductOrders ORDER BY (Quantity * Orders) desc LIMIT 1;

Answer - Camembert Pierrot