# Summer 2022 Data Science Intern Challenge

Please complete the following questions, and provide your thought process/work. You can attach your work in a text file, link, etc. on the application page. Please ensure answers are easily visible for reviewers!

**Question 1:** Given some sample data, write a program to answer the following: [click here to access the required data set](https://docs.google.com/spreadsheets/d/16i38oonuX1y1g7C_UAmiK9GkY7cS-64DfiDMNiR41LM/edit#gid=0)

On Shopify, we have exactly 100 sneaker shops, and each of these shops sells only one model of shoe. We want to do some analysis of the average order value (AOV). When we look at orders data over a 30 day window, we naively calculate an AOV of $3145.13. Given that we know these shops are selling sneakers, a relatively affordable item, something seems wrong with our analysis.

1. Think about what could be going wrong with our calculation. Think about a better way to evaluate this data.
   * 1. To understand the average price of an item, it is important that we divide by the number of items per order. In this case, there are some orders placed by user\_id 607 where 2000 items were ordered at once. This is either a typo, or an exception.
     2. One possibility was that shop with shop\_id 42 was clubbing their data into sets of 2000 orders and hence only had very large orders but this was not found to be true upon closer examination.
2. What metric would you report for this dataset?
   * 1. A better way would be to calculate the average of amount\_per\_item where amount\_per\_item for any given order is given by order amount / number of items.
3. What is its value?
   * 1. The mean of amount per item (amt\_per\_item) equals 387.7428

**Question 2:** For this question you’ll need to use SQL. [Follow this link](https://www.w3schools.com/SQL/TRYSQL.ASP?FILENAME=TRYSQL_SELECT_ALL) to access the data set required for the challenge. Please use queries to answer the following questions. Paste your queries along with your final numerical answers below.

1. How many orders were shipped by Speedy Express in total?  
   **54** orders were placed by Speedy Express in total

SELECT COUNT(DISTINCT OrderID)

FROM Orders

WHERE ShipperID = (

SELECT ShipperID FROM Shippers

WHERE ShipperName = 'Speedy Express');

1. What is the last name of the employee with the most orders?  
   Last name of the employee with the most orders is: **Peacock**

SELECT LastName from Employees

WHERE EmployeeID =

(SELECT EmployeeID

FROM (SELECT COUNT(OrderID) AS TotalOrders, EmployeeID

FROM Orders

GROUP BY EmployeeID

ORDER BY TotalOrders DESC

LIMIT 1))

1. What product was ordered the most by customers in Germany?

**Boston Crab Meat** was ordered the most by customers in Germany.  
I’ve avoided joins for a couple of operations which has resulted in longer code. This was because I wanted to reduce the number of rows on which the join operation would be performed such that the code will be more optimal even if this dataset were larger.

SELECT ProductName FROM Products

WHERE ProductID = (

-- Extract just the product id for the product with highest qty sold in Germany

SELECT ProductID

FROM (

-- List of ProductID and total Qty of that product ordered in Germany

-- List limited to one row with the highest Qty ordered

SELECT ProductID, SUM(Quantity) as QtyOrdered

FROM (

SELECT OrderDetails.OrderID, OrderDetails.ProductID, OrderDetails.Quantity

FROM OrderDetails

-- Filter out values from OrderDetails only for orders placed by Customers

-- in Germany

INNER JOIN

(

SELECT Orders.OrderID

FROM Orders

-- Filter out all orders from customers in Germany, avoid join here to

-- potentially reduce computational complexity as future JOIN operation

-- will only run on Orders that were placed within Germany

-- instead of Joining entire tables

WHERE Orders.CustomerID IN (

-- Create list of all customer ids in Germany

SELECT Customers.CustomerID

FROM Customers

WHERE Country = 'Germany'

)

) AS GermanOrders

ON OrderDetails.OrderID = GermanOrders.OrderID

)

GROUP BY ProductID

ORDER BY QtyOrdered DESC

LIMIT 1

)

)