# Fall 2021 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.

An average order value is the average total of every order placed over a defined period. An average order value of $3145.13 has been calculated using the only order\_amount. But a better way to evaluate this data is to also consider total\_items along with order\_amount.

1. What metric would you report for this dataset?

First, need to report total order\_amount by using the SUM function and then report the total of total\_items by using the SUM function again. Afterward, I divided the total order\_amount with the total of total\_items.

1. What is its value?

Total order\_amount

=SUM(D2:D5001)

It returned 15725640.

Total of total\_items

=SUM(E2:E5001)

It returned 43936.

Total order\_amount/ Total of total\_items

=I2/I3

It returned 357.92

**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.

Diagram

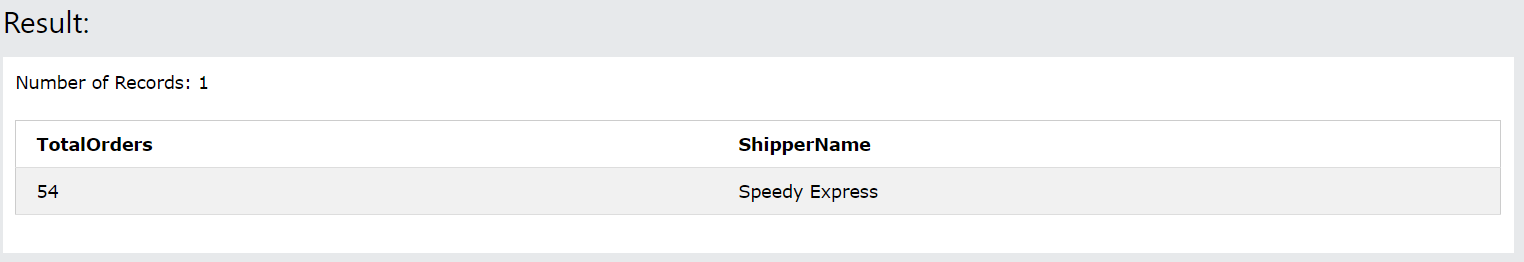
Description automatically generated

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

**Thought Process:**

The first thing that came up to my mind after I read the question was to check all the tables in the given database. Then I decided to draw an entity-relationship diagram to get the visual representation of the database. The question that was raised in my mind was that what are the required fields I would need to write down in my query. It is mentioned in the question that I need orders (OrderId) that are present in two tables, Orders and OrderDetails. In the Shippers table, ShipperName is present from where I can extract SpeedyExpress details. Now comes the question of how many tables I need to get all the required fields. By looking at the diagram, Orders and Shippers have a relationship, so I decided to eliminate the OrderDetails table. To find the total number of orders, I used COUNT aggregation on the OrderID field from the Orders table. I joined two tables using the common ShipperID field and mentioned Speedy Express in the where condition.

**Query:** SELECT COUNT(OrderID) as TotalOrders, S.ShipperName FROM Orders O, Shippers S where O.ShipperID = S.ShipperID AND ShipperName = "Speedy Express";

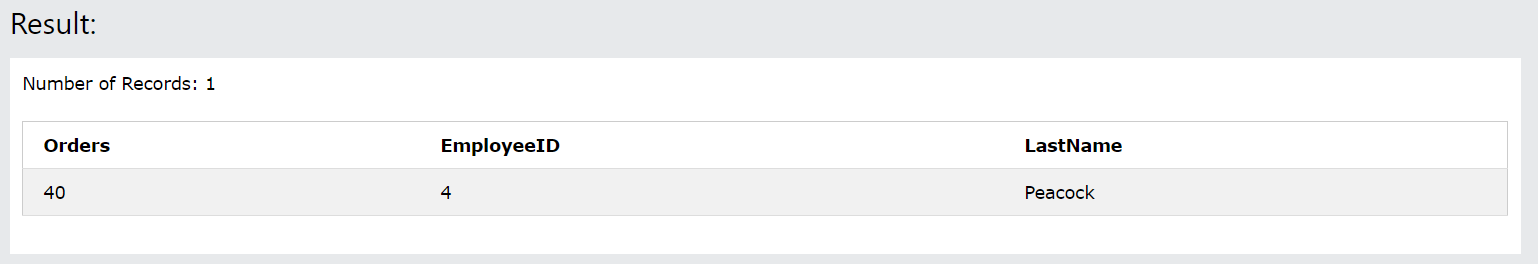


1. What is the last name of the employee with the most orders?

**Thought Process:**

After reading the question, the first thing that popped up in my mind was that the last name of an employee can be extracted from the Employees table. To find the maximum orders placed by an employee, I counted the number of orders using OrderID and then found the maximum count using the MAX function. It is not possible to perform both the aggregation functions in one single query, so I used a subquery to count the number of orders and selected the EmployeeID column from the Orders table. I checked the entity-relationship diagram for the relationship between the Employees table and Orders table. I figured out that the EmployeeID column in the Orders table refers to the EmployeeID in the Employees table. Therefore, I joined both the tables using the EmployeeID field. After I calculated the number of orders, in the outer query, I performed the MAX aggregation function on the result returned from the inner query to find the maximum orders placed.

**Query:**  SELECT MAX(O.Order\_Count) as Orders, E.EmployeeID, E.LastName FROM Employees E, (SELECT COUNT(OrderID) as Order\_Count, EmployeeID FROM Orders GROUP BY EmployeeID) O WHERE O.EmployeeID = E.EmployeeID;



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

**Thought Process:**

For this query, I needed to combine in total 4 tables: Customers, Orders, OrderDetails, and Products. I decided to divide it into three different subqueries. From the diagram, it was evident that there is a relationship between Customers and Orders, Orders and OrderDetails, and lastly, Products and OrderDetails. To extract the customers living in Germany and the orders placed by them, I combined two tables using CustomerID: Customers and Orders. Using the results obtained, again combined the result with the OrderDetails table to get Country and OrderID of OrderDetails table. Now, I needed to calculate the number of products ordered which I was able to achieve using the result obtained by extracting data from Orders, Customers, and OrderDetails tables and then used the COUNT aggregation function on the OrderID field. Lastly, I found the product ordered the most by customers in Germany by combining the three subqueries into the main query using the MAX aggregation function and performing join of the subqueries’ result and Products table on the ProductID field.

(SELECT OC.Country, OD.OrderID FROM (SELECT C.CustomerID, C.Country, O.OrderID FROM Customers C, Orders O WHERE O.CustomerID = C.CustomerID AND C.Country = "Germany") OC, OrderDetails OD WHERE OD.OrderID = OC.OrderID);

SELECT MAX(O.Product\_Count) as ProductCount, P.ProductID, P.ProductName FROM Products P, (SELECT COUNT(OrderID) as Product\_Count, ProductID FROM OrderDetails GROUP BY ProductID) O WHERE O.ProductID = P.ProductID;

**Query:**

SELECT MAX(OP.Product\_Count) as ProductCount, P.ProductID, P.ProductName

FROM Products P, (SELECT COUNT(OD.OrderID) as Product\_Count, OD.ProductID

FROM (SELECT OrdCust.Country, OrdDet.OrderID, OrdDet.ProductID

FROM (SELECT C.CustomerID, C.Country, O.OrderID

FROM Customers C, Orders O

WHERE O.CustomerID = C.CustomerID AND C.Country = "Germany") OrdCust, OrderDetails OrdDet

WHERE OrdDet.OrderID = OrdCust.OrderID) OD GROUP BY OD.ProductID) OP  WHERE OP.ProductID = P.ProductID;

