Question 2: For this question you'll need to use SQL. Follow this link 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.

Initial Analysis

When dealing with a new dataset, performing some initial analysis can be helpful to get an idea of what we're dealing with. This can be done by selecting the first row from each table to avoid loading unnecessary rows for very lengthy tables, which from my training in PostgreSQL and usage of SQLite in a Canada Learning Code workshop I mentored in, can be performed using:

```
SELECT * FROM table_name
LIMIT 1;
```

However, I was met with errors when doing this on the provided editor! The editor used a different version of SQL than I was used to, and it turns out that SELECT TOP 1 FROM table_name was to be used instead. I approached the following problems with this knowledge that I might be dealing with an unfamiliar version of SQL, and kept in mind that certain queries could be different than what I was used to seeing.

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

```
SELECT COUNT(*) FROM Orders;
```

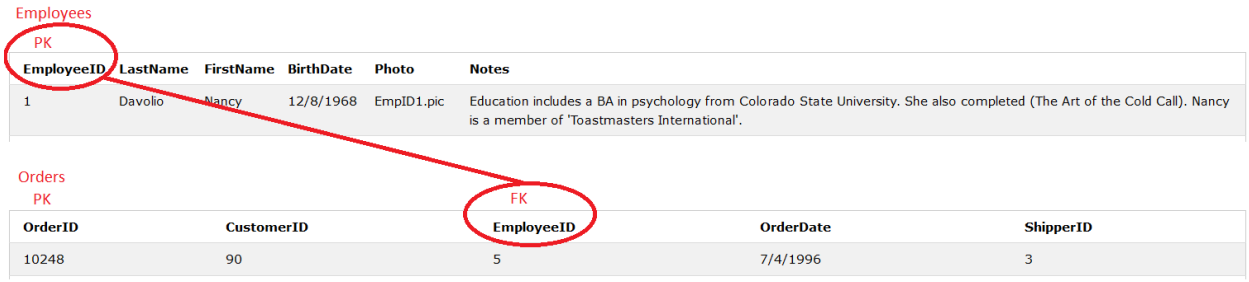
Number of Records: 1

Expr1000
196

Answer: 196 orders were shipped.

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

Since there is no table with both employee name and order number information, some sort of join must be done. From the initial analysis, an ideal candidate for the join would be the Primary Key of the Employees table, EmployeeID, to the corresponding Foreign Key column in the Orders table (see diagram below). Since we're looking for cases where the EmployeeID is the same in both tables, an INNER JOIN would be performed. Then, a GROUP BY on the Employees' LastName can be done to see how many orders each Employee performed, with the assumption that each employee has a unique last name. We can then ORDER BY the number of orders in descending order. As we only need one result, we can select TOP 1 to limit only one result to show.



The primary-foreign key relations between the Employees and Orders tables used to perform joins.

```
SELECT TOP 1 Employees.LastName, COUNT(*) AS numOrders FROM Orders
INNER JOIN Employees
ON Orders.EmployeeID = Employees.EmployeeID
GROUP BY Employees.LastName
ORDER BY COUNT(*) DESC;
```

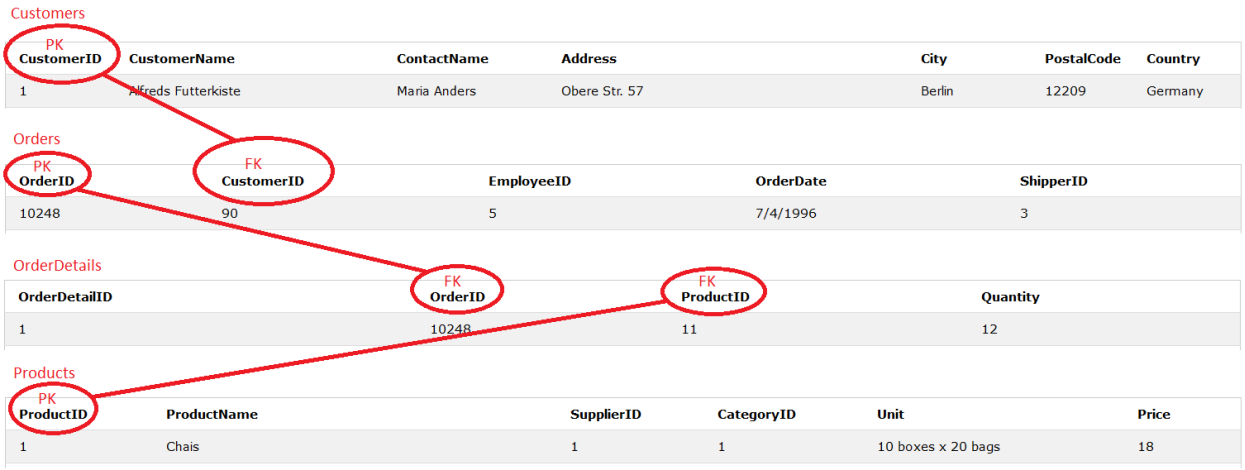
Number of Records: 1

LastName	numOrders
Peacock	40

Answer: The last name of the employee with the most orders is "Peacock".

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

This question is similar to b), in which the desired columns are not all in the same table. However, there is an added layer of difficulty in that they exist in more than two tables, necessitating multiple INNER JOINs! A WHERE statement was used to filter for customers in Germany, and the resultant rows were ordered in descending order by the aggregate SUM function called on the Quantity column from the OrderDetails table.. Upon performing this task, I noticed that unlike PostgreSQL, brackets were necessary to perform the consecutive INNER JOINs.



The primary-foreign key relations between the Customers, Orders, OrderDetails, and Products tables used to perform joins.

```
SELECT TOP 1 Products.ProductName, SUM(OrderDetails.Quantity) AS total_quantity
FROM (((Customers
INNER JOIN Orders
ON Customers.CustomerID = Orders.CustomerID)
INNER JOIN OrderDetails
ON Orders.OrderID = OrderDetails.OrderID)
INNER JOIN Products
ON OrderDetails.ProductID = Products.ProductID)
WHERE Customers.Country = "Germany"
GROUP BY Products.ProductName
ORDER BY SUM(OrderDetails.Quantity) DESC;
```

Number of Records: 1

ProductName	total_quantity
Boston Crab Meat	160

Answer: “Boston Crab Meat” is the most ordered product by customers in Germany.