**Explanation of queries**

**1. Table Creation:**

In First step i have created the tables: Customers, Suppliers, Products, Orders, and OrderDetails. each with specific columns (like CustomerID, ProductName, etc.) ready to hold data. The PRIMARY KEY constraint ensures that each record in a table has a unique identifier. The FOREIGN KEY constraints establish relationships between tables (e.g., an OrderID in OrderDetails must match an OrderID in Orders).

**2. Data Insertion:**

Next, i have inserted the sample data into the tables.

**3. Retrieving Orders for a Specific Customer:**

SQL

SELECT c.FirstName, c.LastName, p.ProductName, o.OrderDate, od.Quantity

FROM Orders o

JOIN OrderDetails od ON o.OrderID = od.OrderID

JOIN Products p ON od.ProductID = p.ProductID

JOIN Customers c ON o.CustomerID = c.CustomerID

WHERE c.CustomerID = 1;

* **Joins:** The database starts by joining the tables. It looks for matching OrderID values in Orders and OrderDetails, then matching ProductID values in OrderDetails and Products, and finally, matching CustomerID values in Orders and Customers. This creates a combined dataset with all the related information.
* **Filtering:** The WHERE clause filters this combined data, keeping only the rows where CustomerID is 1.
* **Selection:** Finally, the database selects the specified columns (FirstName, LastName, ProductName, OrderDate, Quantity) from the filtered data and displays the result.

**4. Finding the Most Purchased Product:**

SQL

SELECT p.ProductName, SUM(od.Quantity) AS TotalQuantity

FROM OrderDetails od

JOIN Products p ON od.ProductID = p.ProductID

GROUP BY p.ProductName

ORDER BY TotalQuantity DESC

LIMIT 1;

* **Join:** The database joins OrderDetails and Products.
* **Grouping:** It groups the joined data by ProductName.
* **Aggregation:** The SUM(od.Quantity) calculates the total quantity sold for each product.
* **Ordering:** The ORDER BY clause sorts the results by TotalQuantity in descending order.
* **Limiting:** LIMIT 1 selects only the first row, which represents the product with the highest total quantity sold.

**5. Updating Stock Quantity:**

SQL

UPDATE Products

SET StockQuantity = StockQuantity - od.Quantity

FROM OrderDetails od

WHERE od.ProductID = Products.ProductID

AND od.OrderID = 101;

* **Matching:** The database identifies the product in the Products table where ProductID matches the ProductID in OrderDetails and OrderID is 101.
* **Update:** It then updates the StockQuantity of that product by subtracting the Quantity from OrderDetails.

**6. Deleting a Customer's Record:**

SQL

DELETE FROM OrderDetails

WHERE OrderID IN (SELECT OrderID FROM Orders WHERE CustomerID = 1);

DELETE FROM Orders

WHERE CustomerID = 1;

DELETE FROM Customers

WHERE CustomerID = 1;

* **Order of Operations (Crucial):** The database *must* execute these DELETE statements in the specified order. First, it deletes related records from OrderDetails because it has a foreign key referencing Orders. Then it deletes from Orders because it has a foreign key referencing Customers. Finally, it can delete the customer from the Customers table. This prevents foreign key constraint violations.