**Requirement:** Retrieves customer names, order IDs, product names, and quantities for orders placed on or after January 1, 2023, sorted by order ID.

To write the SQL query with three joins. We can create the following types of diagrams to aid in understanding the database structure and relationships:

1. **Entity-Relationship Diagram (ERD):**
   * Purpose: Visualizes the tables (entities) and their relationships via primary and foreign keys.
   * Details: For the query involving Customers, Orders, OrderDetails, and Products, the ERD would show:
     + Customers (CustomerID as primary key)
     + Orders (OrderID as primary key, CustomerID as foreign key)
     + OrderDetails (OrderID and ProductID as composite primary key or foreign keys)
     + Products (ProductID as primary key)
     + Relationships (e.g., one-to-many between Customers and Orders, Orders and OrderDetails, etc.).
   * Use: Helps confirm the join conditions (e.g., c.CustomerID = o.CustomerID).
2. **Database Schema Diagram**:
   * Purpose: Shows the structure of each table, including columns, data types, and constraints.
   * Details: Displays columns like CustomerID, FirstName, LastName (in Customers), OrderID, OrderDate (in Orders), etc., with primary/foreign key relationships.
   * Use: Ensures you select the correct columns (e.g., c.FirstName, p.ProductName) and understand data types for conditions like o.OrderDate >= '2023-01-01'.
3. **Join Diagram (or Query Flow Diagram):**
   * Purpose: Illustrates the join logic and sequence of table connections.
   * Details: Depicts Customers → Orders (INNER JOIN), Orders → OrderDetails (LEFT JOIN), and OrderDetails → Products (RIGHT JOIN), showing how data flows through the joins.
   * Use: Clarifies the type of joins (INNER, LEFT, RIGHT) and their impact on the result set.
4. **Data Flow Diagram (DFD):**
   * Purpose: Maps the flow of data from input (tables) to output (query results).
   * Details: Shows how data from Customers, Orders, OrderDetails, and Products is filtered (WHERE), joined, and sorted (ORDER BY).
   * Use: Helps visualize the query’s logic and identify potential performance bottlenecks.

SELECT

c.CustomerID,

c.FirstName,

c.LastName,

o.OrderID,

p.ProductName,

od.Quantity

FROM

Customers c

INNER JOIN Orders o ON c.CustomerID = o.CustomerID

LEFT JOIN OrderDetails od ON o.OrderID = od.OrderID

RIGHT JOIN Products p ON od.ProductID = p.ProductID

WHERE

o.OrderDate >= '2023-01-01'

ORDER BY

o.OrderID;