

Retail Electronics Sales Data Analysis

Introduction

This document provides a comprehensive guide to analysing sales and customer data for a retail electronics company using SQL. The project demonstrates how to:

- Create a structured database for retail operations
- Insert and manage customer and sales data
- Perform various types of SQL queries to extract business insights
- Analyse customer behaviour, sales trends, and product performance

Table of Contents:

- **Step1: Database Schema**
- **Step 2: Data Insertion**
- **Step 3: SQL Queries to Explore Insights**
- **Step 4: Business Analysis & Use Cases**
- **Step 5: Summary & Conclusion**

Let's start building!

Step 1: Setting Up the Database Schema

Create Database

```
CREATE DATABASE Project;
USE project;
```

Customers Table

```
CREATE TABLE Customers(
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(100),
    LastName VARCHAR(100),
    City VARCHAR(100),
    JoinDate DATE
);
```

Orders Table

```
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT FOREIGN KEY REFERENCES Customers(CustomerID),
    OrderDate DATE,
    Product VARCHAR(50),
    Quantity INT,
    Price INT
);
```

Step 2: Data Insertion

After setting up the schema, we now populate each table with initial sample data. This step ensures that we have enough variety to run real-time analytics and answer business queries later in the project.

Insert Customers:

```
INSERT INTO Customers
VALUES(1,'John','Doe','Mumbai','2024-01-05'),
      (2,'Alice','Smith','Delhi','2024-02-15'),
      (3,'Bob','Brown','Bangalore','2024-03-20'),
      (4,'Sara','White','Delhi','2024-01-25'),
      (5,'Mike','Black','Chennai','2024-02-10');
```

Insert Orders:

```
INSERT INTO Orders
VALUES (101, 1, '2024-04-10', 'Laptop', 1, 55000),
       (102, 2, '2024-04-12', 'Mouse', 2, 800),
       (103, 1, '2024-04-15', 'Keyboard', 1, 1500),
       (104, 3, '2024-04-20', 'Laptop', 1, 50000),
       (105, 4, '2024-04-22', 'Headphones', 1, 2000),
       (106, 2, '2024-04-25', 'Laptop', 1, 52000),
       (107, 5, '2024-04-28', 'Mouse', 1, 700),
       (108, 3, '2024-05-02', 'Keyboard', 1, 1600);
```

Step 3: SQL Queries to Explore Insights

Now that the database is set up with customer and order records, we'll begin answering real-world business questions using SQL queries. These queries will help:

- Monitor sales and order trends
- Understand customer purchase behaviour
- Track product performance
- Measure revenue by product, city, and customer segment

The next section includes SQL queries grouped by business objectives, each designed to reveal key operational insights:

- Customer & Order Analysis
- Sales Performance by Product and Region
- Revenue Trends & High-Value Orders
- Customer Retention and Loyalty Metrics
- Comparative Sales Insights using Aggregation and Ranking

Step 4: Business Analysis & Use Cases

1. Customers from Mumbai

```
SELECT * FROM Customers  
WHERE City = 'Mumbai';
```

Insight: Helps identify customers located in a specific region for location-based targeting and promotions.

2. Orders for Laptops

```
SELECT * FROM Orders  
WHERE Product = 'Laptop';
```

Insight: Tracks high-demand products to help in inventory forecasting and marketing.

3. Total Orders Placed

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

Insight: Gives an overview of order volume, useful for business performance tracking.

4. Orders with Price Between ₹50,000 and ₹80,000

```
SELECT * FROM Orders  
WHERE Price  
BETWEEN 50000 AND 80000;
```

Insight: Highlights mid-to-high value transactions that contribute significantly to revenue.

5. Products Priced Above ₹10,000

```
SELECT * FROM Orders  
WHERE Price > 10000;
```

Insight: Identifies premium products frequently purchased, useful for upselling opportunities.

6. Customer and Their Product Orders

```
SELECT c.firstname + " " + c.lastname AS FullName, o.product  
FROM Customers c JOIN Orders o  
ON c.CustomerID = o.CustomerID;
```

Insight: Connects customer identity with product interest for personalization strategies.

7. Customers Who Have Not Placed Orders

```
SELECT c.customerid, o.orderid  
FROM Customers c JOIN orders o  
ON c.CustomerID = o.CustomerID  
WHERE o.OrderID IS NULL;
```

```
SELECT * FROM Customers  
WHERE CustomerID NOT IN (SELECT DISTINCT customerid FROM orders);
```

Insight: Targets inactive customers for re-engagement campaigns.

8. Total Revenue from All Order

```
SELECT SUM(price * quantity) AS TotalRevenue  
FROM orders;
```

Insight: Measures overall sales performance, essential for financial reporting

9. Total Quantity of Mouse Sold

```
SELECT SUM(quantity) as TotalRevenue  
FROM Orders  
WHERE Product = 'Mouse';
```

Insight: Evaluates popularity and movement of specific low-cost accessories.

10. Total Sales Per Customer

```
SELECT c.firstname, sum(o.quantity * o.price) AS SALES  
FROM Customers c join orders o  
ON c.CustomerID = o.CustomerID  
GROUP BY c.FirstName;
```

Insight: Ranks customers by contribution to revenue, useful for loyalty programs.

11. Orders Per City

```
SELECT c.city, COUNT(o.orderid) AS Orders  
FROM Customers c JOIN Orders o  
ON c.CustomerID = o.CustomerID  
GROUP BY c.City;
```

Insight: Assesses geographical performance for regional expansion planning.

12. Customers Who Spent Over ₹50,000

```
SELECT c.* FROM Customers c  
WHERE c.CustomerID in (  
    SELECT CustomerID FROM orders  
    GROUP BY CustomerID  
    HAVING SUM(price) > 50000);
```

Insight: Identifies high-value customers who can be prioritized for premium service.

13. Order Value Label (High/Low)

```
SELECT orderid, price,  
CASE  
    WHEN price > 50000 THEN 'high value'  
    ELSE 'low value'  
END AS ValueLabel  
FROM orders;
```

Insight: Helps categorize transactions for targeted promotions and fraud analysis

14. Running Total of Revenue by Date

```
SELECT orderid, orderdate, price,  
    SUM(price) OVER (order by orderdate) AS runningrevenue  
FROM orders;
```

Insight: Visualizes cumulative revenue growth to understand trends over time.

15. Row Number of Each Order by Customer

```
SELECT orderID, customerID, OrderDate, Price,  
    ROW_NUMBER() OVER (PARTITION BY CustomerID ORDER BY  
    orderdate) AS RowNum  
FROM orders;
```

Insight: Useful for analysing repeat purchase behaviour and customer order history

16. Row Number by Customer and Price

```
SELECT orderID, customerID, OrderDate, Price,  
    ROW_NUMBER() OVER (PARTITION BY CustomerID ORDER BY price)  
AS RowNum  
FROM orders;
```

Insight: Tracks order preferences based on price sensitivity within customers.

17. Rank Orders by Price (RANK)

```
SELECT orderID, customerID, Price,  
    RANK() OVER (ORDER BY price DESC) AS PriceRank  
FROM orders;
```

Insight: Identifies the most and least expensive orders for comparative reporting.

18. Rank Orders by Quantity

```
SELECT orderID, customerID, Quantity,  
    RANK() OVER (ORDER BY quantity DESC) AS QuantityRank  
FROM orders;
```

Insight: Highlights bulk purchases and helps identify wholesale behaviors.

19. Dense Rank by Price

```
SELECT orderID, customerID, Price,  
    DENSE_RANK() OVER (ORDER BY price DESC) AS PriceRank  
FROM orders;
```

Insight: Provides consistent ranking for orders with same price, useful for fair comparison.

20. Customers with More Than One Order

```
SELECT customerid, COUNT(orderid) AS totalorders  
FROM orders  
GROUP BY CustomerID  
HAVING COUNT(orderid) > 1;
```

Insight: Identifies repeat buyers who contribute to customer retention and lifetime value.

Step 6: Conclusion

This project demonstrates how SQL can transform retail sales data into actionable business insights. From customer behaviour and product performance to revenue trends and geographic analysis, each query reveals patterns that can help optimize inventory, improve marketing strategies, and guide data-driven decision making. Well-structured SQL queries empower retail businesses to understand their operations at a granular level and make strategic improvements.

The techniques shown here can be extended to:

- Predictive analysis of future sales
- Customer churn prediction
- Inventory optimization
- Personalized marketing campaigns
- Pricing strategy optimization