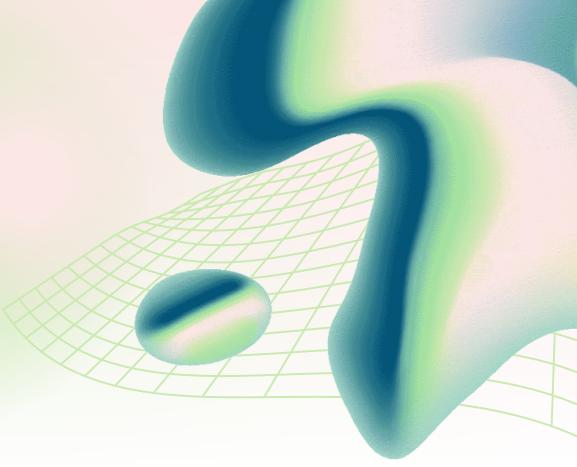


SQL AND BUSINESS ANALYTICS: WEEK 2 OVERVIEW

Exploring the Role of SQL in Enhancing Business Insights



Week 2 Documentation: SQL Implementation & Business Analytics

Project Title

Sales & Customer Analytics Using SQL – Week 2

Objective

In Week 2, we converted the cleaned dataset into a star-schema SQL database and performed key analytics to generate basic business insights, including a Single Customer View.

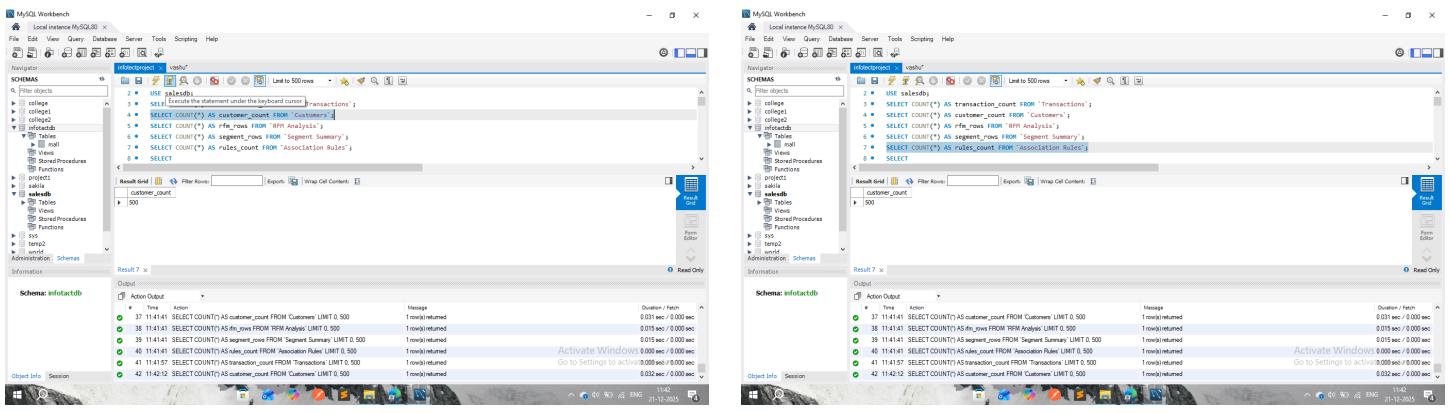
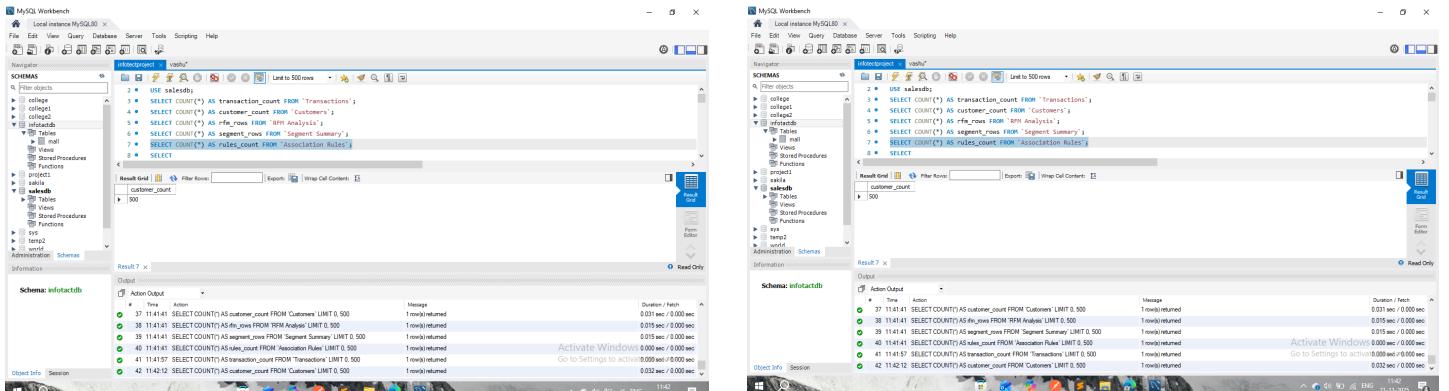
1. Database & Table Setup

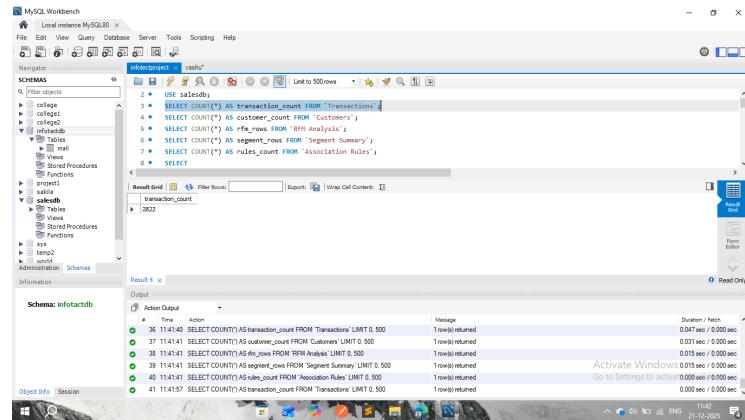
1.1 Tables Imported

We used the MySQL Data Import Wizard to load the following cleaned CSV/Excel files:

| Table Name | Source File | Description |
|-------------------|-----------------------|---------------------------------|
| customers | Customers.csv | Customer master data |
| transactions | Transactions.csv | Sales transactions (fact table) |
| rfm_analysis | RFM Analysis.csv | RFM scoring data |
| segment_summary | Segment Summary.csv | Segment performance aggregates |
| association_rules | Association Rules.csv | Product association metrics |

SCHREENSHOOTS





2. Data Cleaning & Preliminary Checks

2.1 Studied the Tables

We reviewed table structures using:

- DESCRIBE transactions;
- DESCRIBE customers;
- DESCRIBE rfm_analysis;
- DESCRIBE segment_summary;
- DESCRIBE association_rules;

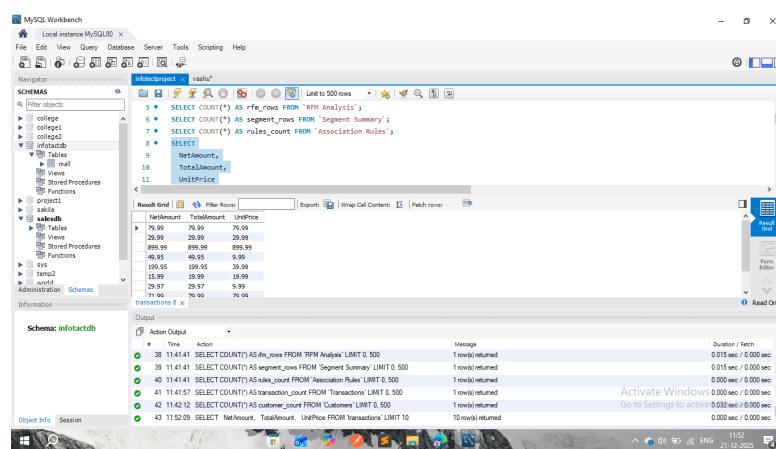
2.2 Cleaned Numeric Columns

Sales amount fields originally contained currency symbols (\$). We removed \$ from:

- NetAmount
- TotalAmount
- UnitPrice

to ensure they are numeric and aggregatable.

Screenshot:



2.3 Check for NULL Keys

We verified key columns do not contain NULL values:

- SELECT COUNT(*) FROM transactions WHERE CustomerID IS NULL;
- SELECT COUNT(*) FROM transactions WHERE ProductID IS NULL;
- SELECT COUNT(*) FROM customers WHERE CustomerID IS NULL;

✓ No NULL primary key values found.

3. Table Enhancements: Data Types & Keys

3.1 Convert Key Columns to Fixed Types

To enable foreign keys, we changed TEXT to VARCHAR for keys:

```
ALTER TABLE transactions MODIFY COLUMN CustomerID VARCHAR(50);
ALTER TABLE transactions MODIFY COLUMN ProductID VARCHAR(50);
ALTER TABLE customers MODIFY COLUMN CustomerID VARCHAR(50);
```

3.2 Add Primary/Unique Keys

```
ALTER TABLE customers ADD PRIMARY KEY (CustomerID);
```

Screenshot: Customers primary key added

3.3 Add Foreign Keys

```
ALTER TABLE transactions
ADD CONSTRAINT fk_transactions_customers
FOREIGN KEY (CustomerID)
REFERENCES customers(CustomerID);
```

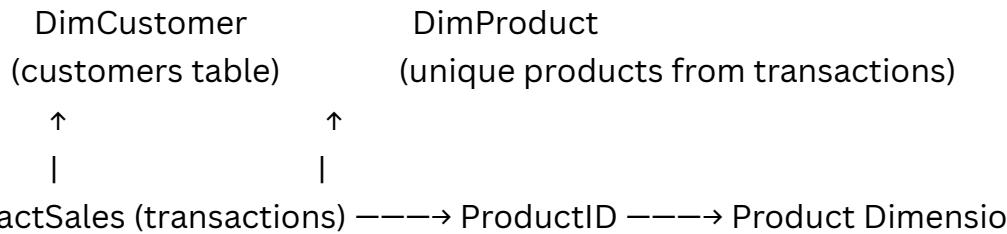
```
ALTER TABLE transactions
ADD CONSTRAINT fk_transactions_products
FOREIGN KEY (ProductID)
REFERENCES products(ProductID);
```

Screenshot: Foreign key constraints

```
-- #FOREGIN KEY Transactions → Products
39 • ALTER TABLE transactions
40   ADD CONSTRAINT fk_transactions_products
41     FOREIGN KEY (ProductID) REFERENCES products(ProductID);
42
43
```

```
26      #FOREGIN KEY Transactions → Customers
27 • ALTER TABLE transactions
28      MODIFY COLUMN CustomerID VARCHAR(50);
29 • ALTER TABLE transactions
30      MODIFY COLUMN ProductID VARCHAR(50);
31 • ALTER TABLE customers
32      MODIFY COLUMN CustomerID VARCHAR(50);
33 • ALTER TABLE customers
34      ADD PRIMARY KEY (CustomerID);
35 • ALTER TABLE transactions
36      ADD CONSTRAINT fk_transactions_customers
37      FOREIGN KEY (CustomerID) REFERENCES customers(CustomerID);
--
```

4. Star Schema Diagram (Text Representation)



- ✓ Fact table: transactions
- ✓ Dimension tables: customers, products (derived)

5. Basic Business Analytics Queries (Week 2)

5.1 Total Sales Revenue

```
SELECT SUM(NetAmount) AS total_sales FROM transactions;
```

Result

```

MySQL Workbench - Local instance MySQL80 - infotactproject - vashu*
File Edit View Query Database Server Tools Scripting Help
Navigator: infotactproject - vashu*
Schemas: infotactdb
Tables: total_sales
Views: 
Stored Procedures: 
Functions: 
Project1: 
Sakila: 
Salesdb: 
Tables: 
Views: 
Stored Procedures: 
Functions: 
Sys: 
Temp2: 
World: 
Administration: 
Information: Schemas
Object Info Session
Result 9 x
Schema: infotactdb
Output:
Action Output
# Time Action
41 11:41:57 SELECT COUNT(*) AS transaction_count FROM `Transactions` LIMIT 0, 500 1 row(s) returned 0.000 sec / 0.000 sec
42 11:42:12 SELECT COUNT(*) AS customer_count FROM `Customers` LIMIT 0, 500 1 row(s) returned 0.032 sec / 0.000 sec
43 11:52:09 SELECT NetAmount, TotalAmount, UnitPrice FROM `transactions` LIMIT 10 10 row(s) returned 0.000 sec / 0.000 sec
44 11:54:57 ALTER TABLE `transactions` ADD CONSTRAINT fk_transactions_customers FOREIGN KEY... Error Code: 1826. Duplicate foreign key constraint name `fk_transactions_customers` to activate Windows. 0.031 sec
45 11:55:16 ALTER TABLE `transactions` ADD CONSTRAINT fk_transactions_products FOREIGN KEY... Error Code: 1826. Duplicate foreign key constraint name `fk_transactions_products` to activate Windows. 0.000 sec
46 11:58:54 SELECT SUM(NetAmount) AS total_sales FROM `transactions` LIMIT 0, 500 1 row(s) returned 0.015 sec / 0.000 sec
Result Grid | Filter Rows: | Export: | Wrap Cell Content: 
total_sales
▶ 405008.78999999556
Result Grid
Form Editor
Read Only

```

5.2 Total Unique Customers

SELECT COUNT(DISTINCT CustomerID) AS total_customers FROM transactions;

Result

```

MySQL Workbench - Local instance MySQL80 - infotactproject - vashu*
File Edit View Query Database Server Tools Scripting Help
Navigator: infotactproject - vashu*
Schemas: infotactdb
Tables: total_customers
Views: 
Stored Procedures: 
Functions: 
Project1: 
Sakila: 
Salesdb: 
Tables: 
Views: 
Stored Procedures: 
Functions: 
Sys: 
Temp2: 
World: 
Administration: 
Information: Schemas
Object Info Session
Result 10 x
Schema: infotactdb
Output:
Action Output
# Time Action
42 11:42:12 SELECT COUNT(*) AS customer_count FROM `Customers` LIMIT 0, 500 1 row(s) returned 0.032 sec / 0.000 sec
43 11:52:09 SELECT NetAmount, TotalAmount, UnitPrice FROM `transactions` LIMIT 10 10 row(s) returned 0.000 sec / 0.000 sec
44 11:54:57 ALTER TABLE `transactions` ADD CONSTRAINT fk_transactions_customers FOREIGN KEY... Error Code: 1826. Duplicate foreign key constraint name `fk_transactions_customers` to activate Windows. 0.031 sec
45 11:55:16 ALTER TABLE `transactions` ADD CONSTRAINT fk_transactions_products FOREIGN KEY... Error Code: 1826. Duplicate foreign key constraint name `fk_transactions_products` to activate Windows. 0.000 sec
46 11:58:54 SELECT SUM(NetAmount) AS total_sales FROM `transactions` LIMIT 0, 500 1 row(s) returned 0.015 sec / 0.000 sec
47 12:00:49 SELECT COUNT(DISTINCT CustomerID) AS total_customers FROM `transactions` LIMIT 0, 500 1 row(s) returned 0.047 sec / 0.000 sec
Result Grid | Filter Rows: | Export: | Wrap Cell Content: 
total_customers
▶ 500
Result Grid
Form Editor
Read Only

```

5.3 Monthly Sales Trend (Report)

SELECT

```
DATE_FORMAT(InvoiceDate, '%Y-%m') AS month,
SUM(NetAmount) AS monthly_sales
FROM transactions
GROUP BY month
ORDER BY month;
```

Result

The screenshot shows the MySQL Workbench interface. In the top-left pane, the Navigator displays the database schema with tables like college, college1, college2, infotactdb, project1, sakila, salesdb, sys, temp2, and world. The infotactdb schema is selected. In the main central pane, a query editor window titled 'infotactproject' contains the SQL code for the monthly sales trend report. Below the code, the 'Result Grid' shows a single row with 'month' as 'NULL' and 'monthly_sales' as '405008.7899999556'. In the bottom-left pane, the 'Object Info' tab is active, showing the history of actions taken on the database. The status bar at the bottom right indicates the date and time as '21-12-2025 12:01'.

5.4 Best-Selling Product (by Quantity)

SELECT

```
ProductName,
SUM(Quantity) AS total_quantity
FROM transactions
GROUP BY ProductName
ORDER BY total_quantity DESC
LIMIT 10;
```

Result

The screenshot shows the MySQL Workbench interface with a query editor window titled 'infotectproject' and a result grid window titled 'Result 12'. The query executed is:

```

64 ORDER BY total_quantity DESC
65 LIMIT 10;
66 #Best-Revenue Product
67 • SELECT
68     ProductName,

```

The result grid displays the following data:

| ProductName | total_quantity |
|-----------------|----------------|
| Book - Python | 555 |
| Microwave | 517 |
| Pen Set | 502 |
| Toaster | 476 |
| Blender | 468 |
| Jeans | 467 |
| Book - Business | 464 |
| USB Cable | 446 |
| T-Shirt | 445 |
| Mouse | 443 |

The 'Information' pane shows the schema for 'infotactdb'.

6. Single Customer View (Customer 360)

Query:

```

SELECT
    c.CustomerID,
    c.CustomerName,
    COUNT(t.InvoiceNo) AS total_orders,
    SUM(t.NetAmount) AS total_spent,
    MAX(t.InvoiceDate) AS last_purchase_date
FROM transactions t
JOIN customers c
    ON t.CustomerID = c.CustomerID
GROUP BY c.CustomerID, c.CustomerName
ORDER BY total_spent DESC;

```

Result

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the database is set to 'Local instance MySQL80'. The main area displays a query window titled 'infotactproject' with the session name 'vashu'. The query is:

```

82     FROM transactions t
83   JOIN customers c
84     ON t.CustomerID = c.CustomerID
85   GROUP BY c.CustomerID, c.CustomerName
86   ORDER BY total_spent DESC;

```

The result grid shows 14 rows of customer data:

| CustomerID | CustomerName | total_orders | total_spent | last_purchase_date |
|------------|--------------------|--------------|--------------------|--------------------|
| CUST00210 | Customer CUST00210 | 9 | 3064.73 | 30-04-2023 |
| CUST00149 | Customer CUST00149 | 12 | 2966.07 | 28-03-2023 |
| CUST00301 | Customer CUST00301 | 12 | 2784.39 | 30-01-2024 |
| CUST00412 | Customer CUST00412 | 8 | 2721.71 | 30-01-2025 |
| CUST00228 | Customer CUST00228 | 10 | 2562.72 | 29-10-2024 |
| CUST00197 | Customer CUST00197 | 14 | 2441.99 | 31-03-2025 |
| CUST00066 | Customer CUST00066 | 11 | 2429.69 | 25-11-2025 |
| CUST00080 | Customer CUST00080 | 6 | 2407.3100000000004 | 22-05-2025 |
| CUST00091 | Customer CUST00091 | 7 | 2293.79 | 30-11-2023 |
| CUST00345 | Customer CUST00345 | 6 | 2262.33 | 22-03-2023 |

The bottom pane shows the 'Output' history with the following log entries:

- # 46 11:58:54 SELECT SUM(NetAmount) AS total_sales FROM transactions LIMIT 0, 500 1 row(s) returned 0.015 sec / 0.000 sec
- # 47 12:00:49 SELECT COUNT(DISTINCT CustomerID) AS total_customers FROM transactions LIMIT 0, 5... 1 row(s) returned 0.047 sec / 0.000 sec
- # 48 12:01:37 SELECT DATE_FORMAT(InvoiceDate, "%Y-%m") AS month, SUM(NetAmount) AS monthly... 1 row(s) returned 0.031 sec / 0.000 sec
- # 49 12:02:21 SELECT ProductName, SUM(Quantity) AS total_quantity FROM transactions GROUP BY... 10 row(s) returned 0.032 sec / 0.000 sec
- # 50 12:03:25 SELECT ProductName, SUM(NetAmount) AS total_revenue FROM transactions GROUP ... 10 row(s) returned 0.031 sec / 0.000 sec
- # 51 12:04:14 SELECT c.CustomerID, c.CustomerName, COUNT(t.InvoiceNo) AS total_orders, SUM(t...

7. Additional Analytics (Optional)

RFM Segment Summary

```

SELECT Customer_Segment,
       COUNT(*) AS customers_in_segment,
       AVG(Monetary) AS avg_spend
  FROM rfm_analysis
 GROUP BY Customer_Segment;

```

Result

Top Association Rules

```

SELECT *
  FROM association_rules
 ORDER BY Lift DESC
 LIMIT 10;

```

Result

8. Validation & Things Checked

| Check | Status |
|-------------------------------------|------------|
| Totals in SQL = Totals in CSV/Excel | ✓ Verified |
| No null key values | ✓ Verified |
| Data types corrected (numeric/DATE) | ✓ Yes |
| Foreign keys established | ✓ Yes |
| Queries producing expected results | ✓ Yes |

9. SQL Scripts Used (DDL + Analytics)

(List all scripts you ran; optionally attach a .sql file in your submission.)

- CREATE TABLE ... for all tables
- ALTER TABLE ... to add foreign keys
- Aggregation and analytics SELECT queries

(Include the full script text here or as an appendix.)

Week 2 Final Deliverables Checklist

- ✓ SQL table creation scripts
- ✓ Data loaded into tables successfully
- ✓ Basic analytics SQL queries
- ✓ Single Customer View SQL
- ✓ Screenshots of query results
- ✓ Validation tests and documentation

Summary Statement (for Reviewers)

“In Week 2, we converted cleaned datasets into a star-schema SQL database and performed basic sales and customer analytics, including a Single Customer View for customer-level insights.”

Creating a SQL Command Documentation

This document provides an overview of the SQL commands used to manage and analyze the salesdb database. The commands are grouped into categories for easier reference and

understanding.

Database Setup

Create and Use Database

```
CREATE DATABASE salesdb;  
USE salesdb;
```

Data Counting Queries

Count Rows in Tables

```
SELECT COUNT(*) AS transaction_count FROM `Transactions`;  
SELECT COUNT(*) AS customer_count FROM `Customers`;  
SELECT COUNT(*) AS rfm_rows FROM `RFM Analysis`;  
SELECT COUNT(*) AS segment_rows FROM `Segment Summary`;  
SELECT COUNT(*) AS rules_count FROM `Association Rules`;
```

Data Retrieval

Sample Data from Transactions

```
SELECT  
    NetAmount,  
    TotalAmount,  
    UnitPrice  
FROM `transactions`  
LIMIT 10;
```

Describe Table Structures

```
DESCRIBE `transactions`;  
DESCRIBE `customers`;  
DESCRIBE `rfm_analysis`;  
DESCRIBE `segment_summary`;  
DESCRIBE `association_rules`;
```

Data Integrity Checks

Check for NULL Keys

```
SELECT COUNT(*) FROM transactions WHERE CustomerID IS NULL;  
SELECT COUNT(*) FROM transactions WHERE ProductID IS NULL;  
SELECT COUNT(*) FROM customers WHERE CustomerID IS NULL;
```

Foreign Key Constraints

Define Foreign Keys

```
# Adjust column types for Foreign Key Constraints
ALTER TABLE transactions
MODIFY COLUMN CustomerID VARCHAR(50);
ALTER TABLE transactions
MODIFY COLUMN ProductID VARCHAR(50);
ALTER TABLE customers
MODIFY COLUMN CustomerID VARCHAR(50);

# Add Primary Key and Foreign Keys
ALTER TABLE customers
ADD PRIMARY KEY (CustomerID);

ALTER TABLE transactions
ADD CONSTRAINT fk_transactions_customers
FOREIGN KEY (CustomerID) REFERENCES customers(CustomerID);

ALTER TABLE transactions
ADD CONSTRAINT fk_transactions_products
FOREIGN KEY (ProductID) REFERENCES products(ProductID);
```

Analytical Queries

Key Analytics

Total Sales

```
SELECT SUM(NetAmount) AS total_sales
FROM transactions;
```

Total Unique Customers

```
SELECT COUNT(DISTINCT CustomerID) AS total_customers
FROM transactions;
```

Monthly Sales Trend

```
SELECT
DATE_FORMAT(InvoiceDate, '%Y-%m') AS month,
SUM(NetAmount) AS monthly_sales
FROM transactions
```

```
GROUP BY month  
ORDER BY month;
```

Best-Selling Product (by Quantity)

```
SELECT  
    ProductName,  
    SUM(Quantity) AS total_quantity  
FROM transactions  
GROUP BY ProductName  
ORDER BY total_quantity DESC  
LIMIT 10;
```

Best-Revenue Product

```
SELECT  
    ProductName,  
    SUM(NetAmount) AS total_revenue  
FROM transactions  
GROUP BY ProductName  
ORDER BY total_revenue DESC  
LIMIT 10;
```

Single Customer View (Customer 360)

```
SELECT  
    c.CustomerID,  
    c.CustomerName,  
    COUNT(t.InvoiceNo) AS total_orders,  
    SUM(t.NetAmount) AS total_spent,  
    MAX(t.InvoiceDate) AS last_purchase_date  
FROM transactions t  
JOIN customers c  
    ON t.CustomerID = c.CustomerID  
GROUP BY c.CustomerID, c.CustomerName  
ORDER BY total_spent DESC;
```

This document serves as a comprehensive guide to the SQL commands for managing and analyzing data within the salesdb database. These commands facilitate the creation of structured insights and ensure data integrity.