Products Table:

The Products table contains details about products, including their names, categories, and unit

prices. It provides reference data for linking product information to sales transactions.

Query:

-- Create Products table

Create schema Products;

use products;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

INSERT INTO Products (product\_id, product\_name, category, unit\_price) VALUES

(101, 'Laptop', 'Electronics', 500.00),

(102, 'smartphone', 'Electronics', 300.00),

(103, 'Headphones', 'Electronics', 30.00),

(104, 'Keyboard', 'Electronics', 20.00),

(105, 'Mouse', 'Electronics', 15.00);

-- 1. Retrieve all columns from the product table.

SELECT \* FROM Products;

-- 2. Retrieve the product\_name and unit\_price from the Products table.

SELECT product\_name, unit\_price FROM Products;

-- 3. Filter the Products table to show only products in the 'Electronics' category.

SELECT \* FROM Products WHERE unit\_price > 100;

-- 4. Retrieve the product\_id and product\_name from the Products table for products with aunit\_price greater than $100.

SELECT product\_id, product\_name FROM Products WHERE unit\_price > 100;

-- 5. Calculate the average unit\_price of products in the Products table.

SELECT AVG(unit\_price) AS average\_unit\_price FROM Products;

-- 6. Retrieve product\_name and unit\_price from the Products table with the Highest Unit Price

SELECT product\_name, unit\_price FROM Products WHERE unit\_price = (SELECT MAX(unit\_price) FROM Products);

-- 7. Retrieve the product\_name and unit\_price from the Products table, ordering the results byunit\_price in descending order.

SELECT product\_name, unit\_price FROM Products ORDER BY unit\_price DESC;

-- 8. Retrieve the product\_name and unit\_price from the Products table, filtering the unit\_price toshow only values between $20 and $600.

SELECT product\_name, unit\_price FROM Products WHERE unit\_price BETWEEN 20 AND 600;

-- 9. Retrieve the product\_name and category from the Products table, ordering the results bycategory in ascending order.

SELECT product\_name, category FROM Products ORDER BY category ASC;

Sales Table:

The Sales table records information about product sales, including the quantity sold, sale date,

and total price for each sale. It serves as a transactional data source for analyzing sales trends.

Query:

-- Create Sales table

Create schema Sale;

use Sale;

CREATE TABLE Sale (

sale\_id INT PRIMARY KEY,

product\_id INT,

quantity\_sold INT,

sale\_date DATE,

total\_price DECIMAL(10, 2)

);

-- Insert sample data into Sales table

INSERT INTO Sale (sale\_id, product\_id, quantity\_sold, sale\_date, total\_price) VALUES

(1, 101, 5, '2024-01-01', 2500.00),

(2, 102, 3, '2024-01-02', 900.00),

(3, 103, 2, '2024-01-02', 60.00),

(4, 104, 4, '2024-01-03', 80.00),

(5, 105, 6, '2024-01-03', 90.00);

-- 1. Retrieve all columns from the Sales table

SELECT \* FROM Sale;

-- 2. Retrieve the sale\_id and sale\_date from the Sales table

SELECT sale\_id, sale\_date FROM Sale;

-- 3. Filter the Sales table to show only sales with a total\_price greater than $100

SELECT \* FROM Sale WHERE total\_price > 100;

-- 4. Retrieve the sale\_id and total\_price from the Sales table for sales made on January 3, 2024

SELECT sale\_id, total\_price FROM Sale WHERE sale\_date = '2024-01-03';

-- 5. Calculate the total revenue generated from all sales in the Sales table

SELECT SUM(total\_price) AS total\_revenue FROM Sale;

-- 6. Calculate the total quantity\_sold from the Sales table

SELECT SUM(quantity\_sold) AS total\_quantity\_sold FROM Sale;

-- 7. Retrieve the sale\_id, product\_id, and total\_price from the Sales table for sales with a quantity\_sold greater than 4

SELECT sale\_id, product\_id, total\_price FROM Sale WHERE quantity\_sold > 4;

-- 8. Calculate the average total\_price of sales in the Sales table

SELECT AVG(total\_price) AS average\_total\_price FROM Sale;