**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 TABLE Products (  
 product\_id INT PRIMARY KEY,  
 product\_name VARCHAR(100),  
 category VARCHAR(50),  
 unit\_price DECIMAL(10, 2)  
);  
-- Insert sample data into Products table  
  
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 product\_table;

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

select name,unit\_price from product\_table;

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

select name,category from product\_table

where category='elecctronics';

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

select pro\_id,name,unit\_price from product\_table

where unit\_price>100;

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

select avg(unit\_price) from product\_table;

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

select name,max(unit\_price) from product\_table

group by name limit 1;

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

select name,unit\_price from product\_table

order by unit\_price desc;

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

select name,unit\_price from product\_table

where unit\_price between 20 and 600;

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

select name,category from product\_table

order by category;

**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 TABLE Sales (  
 sale\_id INT PRIMARY KEY,  
 product\_id INT,  
 quantity\_sold INT,  
 sale\_date DATE,  
 total\_price DECIMAL(10, 2)  
 FOREIGN KEY (product\_id) REFERENCES Products(product\_id)  
);  
  
  
INSERT INTO Sales (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 sales\_table;

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

select sale\_id,sale\_date from sales\_table

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

select sale\_id,pro\_id,total\_price from sales\_table

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 sales\_table

where sale\_date='2024-01-03';

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

select sum(total\_price) as revenue\_generated from sales\_table;

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

select sum(quantity\_sold) as total\_quantity from sales\_table;

**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,pro\_id,total\_price from sales\_table

where quantity\_sold >4;

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

select sum(quantity\_sold) as total\_quantity from sales\_table;