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 VALUES (101, 'laptop','Electronics',500.00);

INSERT INTO Products VALUES (102, 'Smartphone', 'Electronics', 100.00);

INSERT INTO Products VALUES (103, 'Headphones', 'Electronics', 80.000);

INSERT INTO Products VALUES (104, 'Keyboard', 'Electronics', 20.00);

INSERT INTO Products VALUES (105, 'Mouse', 'Electronics', 15.00);

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

**SELECT \* FROM products.products;**

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

**SELECT product\_name , unit\_price FROM products;**

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

**SELECT \* FROM Products WHERE category = 'Electronics';**

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

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

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

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

1. **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);

1. **Retrieve the product\_name and unit\_price from the Products table, ordering the results by unit\_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 to show only values between $20 and $600.**

SELECT product\_name , unit\_price FROM Products WHERE unit\_price between 20.00 and 600.00;

**9.Retrieve the product\_name and category from the Products table, ordering the results by category 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 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 sample data into Sales table

INSERT INTO Sales VALUES (1, 101, 5, '2024-01-01', 2500.00);

INSERT INTO Sales VALUES (2, 102, 3, '2024-01-02', 900.00);

INSERT INTO Sales VALUES (3, 103, 2, '2024-01-02', 60.00);

INSERT INTO Sales VALUES (4, 104, 4, '2024-01-03', 80.00);

INSERT INTO Sales VALUES (5, 105, 6, '2024-01-03', 90.00);

1. **Retrieve all columns from the Sales table.**

**SELECT \* FROM Sales**

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

**SELECT sale\_id , sale\_date FROM Sales;**

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

SELECT sale\_id ,total\_price FROM Sales WHERE total\_price>100 ;

1. **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 WHERE sale\_date= '2024-01-03';

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

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

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

SELECT SUM(quantity\_sold) AS total FROM Sales;

1. **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, quantity\_sold,total\_price FROM Sales WHERE quantity\_sold >4;

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

**SELECT AVG(total\_price) as avg\_price FROM Sales ;**