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),

(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.**

mysql> select \* from products;

**+------------+--------------+-------------+------------+**

**| product\_id | product\_name | category | unit\_price |**

**+------------+--------------+-------------+------------+**

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

mysql> select product\_name,unit\_price from products;

**+--------------+------------+**

**| product\_name | unit\_price |**

**+--------------+------------+**

**| Laptop | 500.00 |**

**| Smartphone | 300.00 |**

**| Headphones | 30.00 |**

**| Keyboard | 20.00 |**

**| Mouse | 15.00 |**

**+--------------+------------+**

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

mysql> select \* from products where category = 'electronics';

**+------------+--------------+-------------+------------+**

**| product\_id | product\_name | category | unit\_price |**

**+------------+--------------+-------------+------------+**

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

mysql> select product\_name,product\_id from products where unit\_price>100;

**+--------------+------------+**

**| product\_name | product\_id |**

**+--------------+------------+**

**| Laptop | 101 |**

**| Smartphone | 102 |**

**+--------------+------------+**

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

mysql> select avg(unit\_price) from products;

**+-----------------+**

**| avg(unit\_price) |**

**+-----------------+**

**| 173.000000 |**

**+-----------------+**

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

mysql> select product\_name,unit\_price from products where unit\_price=(select max(unit\_price) from products);

+--------------+------------+

| product\_name | unit\_price |

+--------------+------------+

| Laptop | 500.00 |

+--------------+------------

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

mysql> select product\_name,unit\_price from products order by unit\_price desc;

+--------------+------------+

| product\_name | unit\_price |

+--------------+------------+

| Laptop | 500.00 |

| Smartphone | 300.00 |

| Headphones | 30.00 |

| Keyboard | 20.00 |

| Mouse | 15.00 |

+--------------+------------+

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

mysql> select product\_name,unit\_price from products

-> where unit\_price >=20 and unit\_price<=600;

+--------------+------------+

| product\_name | unit\_price |

+--------------+------------+

| Laptop | 500.00 |

| Smartphone | 300.00 |

| Headphones | 30.00 |

| Keyboard | 20.00 |

+--------------+------------+

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

mysql> select product\_name,category from products order by product\_name asc;

+--------------+-------------+

| product\_name | category |

+--------------+-------------+

| Headphones | Electronics |

| Keyboard | Electronics |

| Laptop | Electronics |

| Mouse | Electronics |

| Smartphone | Electronics |

+--------------+-------------+

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),

(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.**

mysql> select \* from sales;

**+---------+------------+---------------+------------+-------------+**

**| sale\_id | product\_id | quantity\_sold | sale\_date | total\_price |**

**+---------+------------+---------------+------------+-------------+**

**| 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 the sale\_id and sale\_date from the Sales table.**

mysql> select sale\_id,sale\_date from sales;

**+---------+------------+**

**| sale\_id | sale\_date |**

**+---------+------------+**

**| 1 | 2024-01-01 |**

**| 2 | 2024-01-02 |**

**| 3 | 2024-01-02 |**

**| 4 | 2024-01-03 |**

**| 5 | 2024-01-03 |**

**+---------+------------+**

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

mysql> select \* from sales where Total\_price > 100;

**+---------+------------+---------------+------------+-------------+**

**| sale\_id | product\_id | quantity\_sold | sale\_date | total\_price |**

**+---------+------------+---------------+------------+-------------+**

**| 1 | 101 | 5 | 2024-01-01 | 2500.00 |**

**| 2 | 102 | 3 | 2024-01-02 | 900.00 |**

**+---------+------------+---------------+------------+-------------+**

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

mysql> select sale\_id,total\_price from sales where sale\_date =' 2024-01-03';

**+---------+-------------+**

**| sale\_id | total\_price |**

**+---------+-------------+**

**| 4 | 80.00 |**

**| 5 | 90.00 |**

**+---------+-------------+**

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

mysql> select sum(total\_price) from sales;

**+------------------+**

**| sum(total\_price) |**

**+------------------+**

**| 3630.00 |**

**+------------------+**

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

mysql> select sum(quantity\_sold) from sales;

**+--------------------+**

**| sum(quantity\_sold) |**

**+--------------------+**

**| 20 |**

**+--------------------+**

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

mysql> select sale\_id,product\_id,total\_price from sales where quantity\_sold >4;

**+---------+------------+-------------+**

**| sale\_id | product\_id | total\_price |**

**+---------+------------+-------------+**

**| 1 | 101 | 2500.00 |**

**| 5 | 105 | 90.00 |**

**+---------+------------+-------------+**

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

mysql> select avg(total\_price) from sales;

**+------------------+**

**| avg(total\_price) |**

**+------------------+**

**| 726.000000 |**

**+------------------+**