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.

ANS: 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 |
+------------+--------------+-------------+------------+
```

2. Retrieve the product_name and unit_price from the Products table.

Ans: 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 |

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

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

ANS: 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 |

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

4. Retrieve the product_id and product_name from the Products table for products with a unit_price greater than $100.

ANS: mysql> select product_id,product_name from products where unit_price >100;

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

| product_id | product_name |

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

| 101 | Laptop |

| 102 | Smartphone |

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

5. **Calculate the average unit_price of products in the Products table.**

ANS: mysql> select avg(unit_price) from Products;

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

| avg(unit_price) |

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

| 173.000000 |

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

6. **Retrieve product_name and unit_price from the Products table with the Highest Unit Price.**

ANS: mysql> select product_name,unit_price from Products order by unit_price desc limit 1;

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

| product_name | unit_price |

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

| Laptop | 500.00 |

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

7. **Retrieve the product_name and unit_price from the Products table, ordering the results by unit_price in descending order.**

ANS: 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 |

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

8. **Retrieve the product_name and unit_price from the Products table, filtering the unit_price to show only values between $20 and $600.**

ANS: 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 |

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

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

ANS: select product_name,category from Products order by category asc;

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

| product_name | category |

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

| Laptop | Electronics |

| Smartphone | Electronics |

| Headphones | Electronics |

| Keyboard | Electronics |

| Mouse | 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)

);

-- Insert sample data into Sales table

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.

Ans: 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 |
+---------+------------+--------------+------------+-------------+
```

2. Retrieve the sale_id and sale_date from the Sales table.

ANS: 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 |
+---------+------------+
```

3. Filter the Sales table to show only sales with a total_price greater than $100.

ANS: 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 |
+---------+------------+---------------+------------+-------------+
```

4. Retrieve the sale_id and total_price from the Sales table for sales made on January 3, 2024.

ANS: mysql> select sale_id,total_price from Sales where sale_date="2024-01-03";

```
+---------+-------------+
| sale_id | total_price |
+---------+-------------+
|       4 |       80.00 |
|       5 |       90.00 |
+---------+-------------+
```

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

ANS: mysql> select sum(total_price) from Sales;

```
+------------------+
| sum(total_price) |
+------------------+
|         3630.00 |
+------------------+
```

6. Calculate the total quantity_sold from the Sales table.

ANS: mysql> select sum(quantity_sold) from Sales;

```
+--------------------+
| sum(quantity_sold) |
+--------------------+
|                 20 |
```

7. Retrieve the sale_id, product_id, and total_price from the Sales table for sales with a quantity_sold greater than 4.

ANS: 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 |
```

8. Calculate the average total_price of sales in the Sales table.

ANS: mysql> select avg(total_price) from Sales;

```
+------------------+
| avg(total_price) |
+------------------+
|      726.000000 |
+------------------+
```