--- 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);

**Output:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 |

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);

|  |  |  |  |
| --- | --- | --- | --- |
| **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 |

**Output:**

1. Retrieve all columns from the Sales table.

**Query:**

SELECT \* FROM Sales;

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

Query:

SELECT product\_name, unit\_price FROM

Products;

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

Query:

SELECT sale\_id, sale\_date FROM Sales;

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

Query:

SELECT \* FROM Sales WHERE total\_price >

100;

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

Query:

SELECT \* FROM Products WHERE category =

'Electronics';

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

Query:

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

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

**Query:**

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

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

**Query:**

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

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

Query:

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

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

Query:

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

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

Query:

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

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

Query:

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

1. Retrieve the total\_price of all sales, rounding the values to two decimal places.

Query:

SELECT ROUND(SUM(total\_price), 2) AS total\_sales   
FROM Sales;

1. Calculate the average total\_price of sales table.

Query:

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

1. Retrieve the sale\_id and sald\_date from the Sales table, formatting the sale\_date as ‘YYYY-MM-DD’ .

Query:

SELECT sale\_id, DATE\_FORMAT(sale\_date, '%Y-%m-%d') AS formatted\_date   
 FROM Sales;

1. Calculate the total revenue generated from sales of products in the ‘Electronics’ category.

Query:

SELECT SUM(Sales.total\_price) AS total\_revenue   
 FROM Sales   
 JOIN Products ON Sales.product\_id = Products.product\_id   
 WHERE Products.category = 'Electronics';

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

Query:

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

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

Query:

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

1. Calculate the total quantity\_slod of products in the ‘Electronics’ category.

Query:

SELECT SUM(quantity\_sold) AS total\_quantity\_sold   
 FROM Sales   
 JOIN Products ON Sales.product\_id = Products.product\_id   
 WHERE Products.category = 'Electronics';

1. Retrieve the product\_name and total\_price from the sales table, calaulating the total\_price as quantity\_sold multiplied by unit\_price.

Query:

SELECT product\_name, quantity\_sold \* unit\_price AS total\_price   
 FROM Sales   
 JOIN Products ON Sales.product\_id = Products.product\_id;