-- 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);

-- 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)  
);  
Note product\_id foreign key

-- 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.**

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

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

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

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

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

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

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

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

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

**11. Count Sales Per Day from the Sales table**

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

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

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

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

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

**17. Retrieve the sale\_id and sale\_date from the Sales table, formatting the sale\_date as 'YYYY-MM-DD'.**

**18. Calculate the total revenue generated from sales of products in the 'Electronics' category.**

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

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

**21. Calculate the total quantity\_sold of products in the 'Electronics' category.**

**22. Retrieve the product\_name and total\_price from the Sales table, calculating the total\_price as quantity\_sold multiplied by unit\_price.**

**23. Find the Products Not Sold from Products table.**

**24.  Calculate the total revenue generated from sales for each product category.**

**25.  Identify products with total sales exceeding 30.**