

**DEPARTMENT OF MECHATRONICS**

**SQL**

**ASSIGNMENT -2**

|  |  |
| --- | --- |
| **NAME OF THE STUDENT** | **Bharaneedharan K** |
| **REGISTER NUMBER** | **22MTL003** |
| **YEAR/SEMESTER** | **IV/VII** |
| **ASSIGNMENT NUMBER** | **2** |

**SQL ASSIGNMENT -2**

**Questions:**

**I. 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.

*1. Retrieve all columns from the product table*

**Query:**

create database production;

use production;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

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);

SELECT \* FROM Products;

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

*2. Retrieve all columns from the product table*

**Query:**

create database production;

use production;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

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);

SELECT product\_name, unit\_price

FROM Products;

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

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

**Query:**

create database production;

use production;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

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),

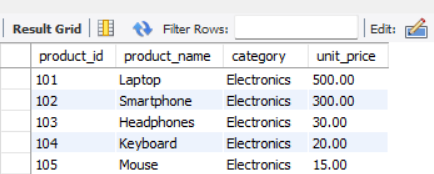
(105, 'Mouse', 'Electronics', 15.00);

SELECT \*

FROM Products

WHERE category = 'Electronics';

**Output:**



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

**Query:**

create database production;

use production;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

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),

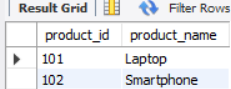
(105, 'Mouse', 'Electronics', 15.00);

SELECT product\_id, product\_name

FROM Products

WHERE unit\_price > 100;

**Output:**



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

**Query:**

create database production;

use production;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

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),

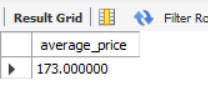
(104, 'Keyboard', 'Electronics', 20.00),

(105, 'Mouse', 'Electronics', 15.00);

SELECT AVG(unit\_price) AS average\_price

FROM Products;

**Output:**



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

**Query:**

create database production;

use production;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

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);

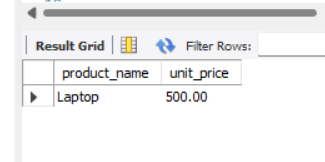
SELECT product\_name, unit\_price

FROM Products

ORDER BY unit\_price DESC

LIMIT 1;

**Output:**



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

**Query:**

create database production;

use production;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

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),

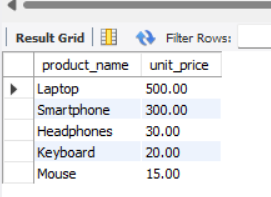
(105, 'Mouse', 'Electronics', 15.00);

SELECT product\_name, unit\_price

FROM Products

ORDER BY unit\_price DESC;

**Output:**



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

**Query:**

create database production;

use production;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

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),

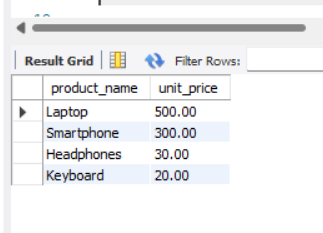
(105, 'Mouse', 'Electronics', 15.00);

SELECT product\_name, unit\_price

FROM Products

WHERE unit\_price BETWEEN 20 AND 600;

**Output:**



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

**Query:**

create database production;

use production;

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100),

category VARCHAR(50),

unit\_price DECIMAL(10, 2)

);

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),

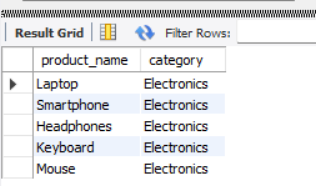
(105, 'Mouse', 'Electronics', 15.00);

SELECT product\_name, category

FROM Products

ORDER BY category ASC;

**Output:**



**QUESTION 2:**

II. 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.

1. Retrieve all columns from the Sales table.

**Query:**

create database salesm;

use salesm;

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 (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),

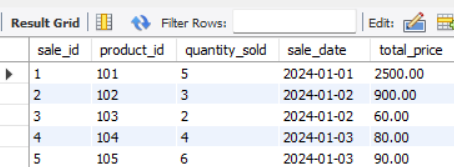
(3, 103, 2, '2024-01-02', 60.00),

(4, 104, 4, '2024-01-03', 80.00),

(5, 105, 6, '2024-01-03', 90.00);

SELECT \* FROM Sales;

**Output:**



2.Retrieve the sale\_id and sale\_date from the Sales table.

**Query:**

create database salesm;

use salesm;

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 (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),

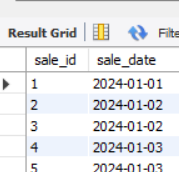
(4, 104, 4, '2024-01-03', 80.00),

(5, 105, 6, '2024-01-03', 90.00);

SELECT sale\_id, sale\_date

FROM Sales;

**Output:**



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

**Query:**

create database salesm;

use salesm;

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 (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),

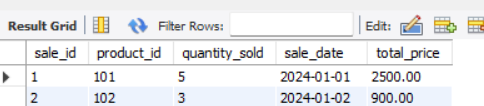
(5, 105, 6, '2024-01-03', 90.00);

SELECT \*

FROM Sales

WHERE total\_price > 100;

**Output:**



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

**Query:**

create database salesm;

use salesm;

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 (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),

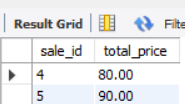
(5, 105, 6, '2024-01-03', 90.00);

SELECT sale\_id, total\_price

FROM Sales

WHERE sale\_date = '2024-01-03';

**Output:**



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

**Query:**

create database salesm;

use salesm;

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 (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),

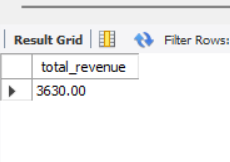
(4, 104, 4, '2024-01-03', 80.00),

(5, 105, 6, '2024-01-03', 90.00);

SELECT SUM(total\_price) AS total\_revenue

FROM Sales;

**Output:**



6.Calculate the total quantity\_sold from the Sales table.

**Query:**

create database salesm;

use salesm;

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 (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),

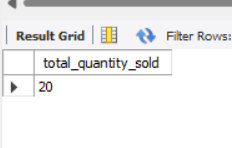
(4, 104, 4, '2024-01-03', 80.00),

(5, 105, 6, '2024-01-03', 90.00);

SELECT SUM(quantity\_sold) AS total\_quantity\_sold

FROM Sales;

**Output:**



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

**Query:**

create database salesm;

use salesm;

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 (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),

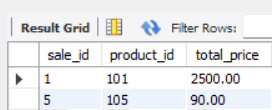
(5, 105, 6, '2024-01-03', 90.00);

SELECT sale\_id, product\_id, total\_price

FROM Sales

WHERE quantity\_sold > 4;

**Output:**



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

**Query:**

create database salesm;

use salesm;

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 (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),

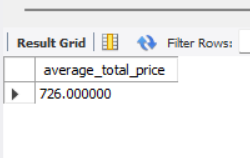
(4, 104, 4, '2024-01-03', 80.00),

(5, 105, 6, '2024-01-03', 90.00);

SELECT AVG(total\_price) AS average\_total\_price

FROM Sales;

**Output:**



THANK YOU