

Lab 10: Create table sales and products using key constraints (primary key and foreign key) insert suitable data and perform DML operation (SELECT CLAUSE, where CLAUSE, aggregation functions).

Source code:

```
CREATE TABLE products (
```

```
    product_id INT PRIMARY KEY,
```

```
    product_name VARCHAR(50) NOT NULL,
```

```
    price DECIMAL(10,2) NOT NULL
```

```
);
```

```
CREATE TABLE sales (
```

```
    sale_id INT PRIMARY KEY,
```

```
    product_id INT,
```

```
    quantity INT NOT NULL,
```

```
    sale_date DATE,
```

```
    FOREIGN KEY (product_id) REFERENCES products(product_id)
```

```
);
```

```
INSERT INTO products (product_id, product_name, price) VALUES
```

```
(1, 'Laptop', 85000.00),
```

```
(2, 'Smartphone', 35000.00),
```

```
(3, 'Headphones', 2500.00),
```

```
(4, 'Monitor', 15000.00),
```

```
(5, 'Keyboard', 1200.00);
```

```
INSERT INTO sales (sale_id, product_id, quantity, sale_date) VALUES
```

```
(1, 1, 2, '2025-08-01'),
```

```
(2, 2, 5, '2025-08-02'),
```

```
(3, 3, 10, '2025-08-03'),
```

```
(4, 1, 1, '2025-08-04'),
```

```
(5, 4, 3, '2025-08-05'),
```

```
(6, 5, 8, '2025-08-06');
```

SELECT * FROM products;

product_id	product_name	price
1	Laptop	85000.00
2	Smartphone	35000.00
3	Headphones	2500.00
4	Monitor	15000.00
5	Keyboard	1200.00

SELECT s.sale_id, p.product_name, s.quantity, p.price, (s.quantity * p.price) AS total_amount,
s.sale_date

FROM sales s

JOIN products p ON s.product_id = p.product_id;

sale_id	product_name	quantity	price	total_amount	sale_date
1	Laptop	2	85000.00	170000.00	2025-08-01
4	Laptop	1	85000.00	85000.00	2025-08-04
2	Smartphone	5	35000.00	175000.00	2025-08-02
3	Headphones	10	2500.00	25000.00	2025-08-03
5	Monitor	3	15000.00	45000.00	2025-08-05
6	Keyboard	8	1200.00	9600.00	2025-08-06

SELECT * FROM products

WHERE price > 20000;

product_id	product_name	price
1	Laptop	85000.00
2	Smartphone	35000.00

SELECT SUM(quantity) AS total_quantity_sold FROM sales;

total_quantity_sold
29

SELECT AVG(price) AS avg_price FROM products;

avg_price
27740.000000

SELECT MAX(quantity) AS max_quantity_sold FROM sales;

max_quantity_sold
10

SELECT MIN(quantity) AS min_quantity_sold FROM sales;

min_quantity_sold
1

SELECT COUNT(*) AS total_sales_transactions FROM sales;

total_sales_transactions
6

SELECT SUM(s.quantity * p.price) AS total_revenue

FROM sales s

JOIN products p ON s.product_id = p.product_id;

total_revenue
509600.00