**DATABASE MANAGEMENT SYSTEM:**

**PROGRAM NO.1**

**Install and set up MySQL. Create a database and a table to store employee details. Perform basic operations like INSERT & DELETE.**

##CREATE DATABASE EmployeeDB;

USE EmployeeDB;

drop table Employee;

CREATE TABLE Employee (

EmployeeID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

Salary DECIMAL(10, 2),

JoiningDate DATE,

ActiveStatus BOOLEAN

);

INSERT INTO Employee (Name, Salary, JoiningDate, ActiveStatus)

VALUES

('Bhakti Raskar', 55000.00, '2023-06-15', TRUE),

(‘Shravani Unde', 72000.50, '2022-09-25', TRUE),

('Mark Smith', 48000.75, '2021-12-10', FALSE),

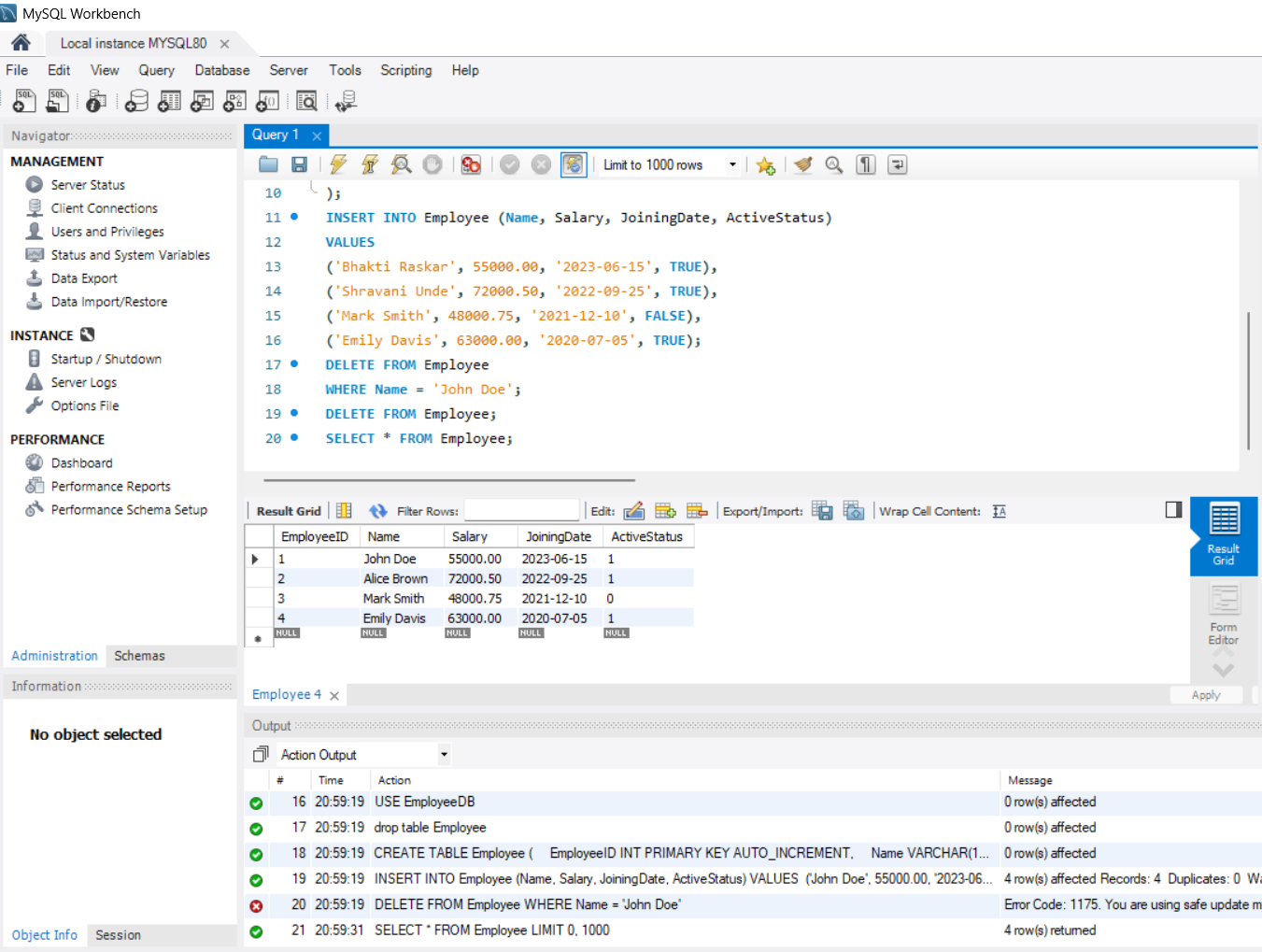
('Emily Davis', 63000.00, '2020-07-05', TRUE);

DELETE FROM Employee

WHERE Name = 'John Doe';

DELETE FROM Employee;

SELECT \* FROM Employee;



**PROGRAM NO.02**

**Create a table for storing student information. Insert sample data and perform basic operations: INSERT, UPDATE, DELETE, and SELECT.**

CREATE TABLE Students (

StudentID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Age INT,

Major VARCHAR(50)

);

INSERT INTO Students (StudentID, FirstName, LastName, Age, Major)

VALUES

(1, 'Aarav', 'Patel', 20, 'Computer Science'),

(2, 'Ishita', 'Sharma', 22, 'Mathematics'),

(3, 'Rohan', 'Mehta', 19, 'Physics'),

(4, 'Sanya', 'Kapoor', 21, 'Chemistry');

UPDATE Students

SET Age = 21

WHERE StudentID = 3;

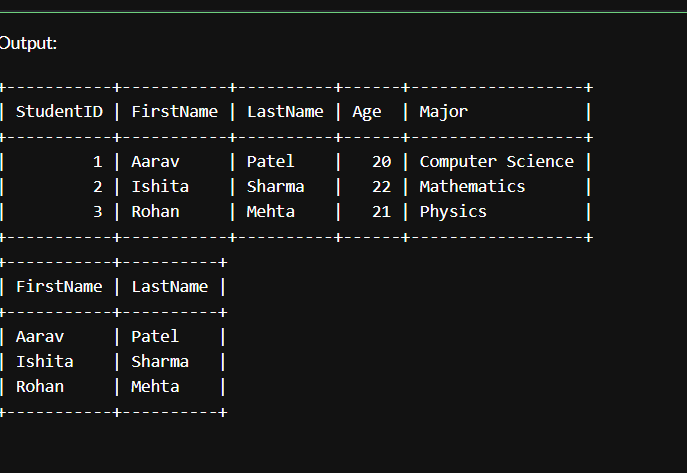
DELETE FROM Students

WHERE StudentID = 4;

SELECT \* FROM Students;

SELECT FirstName, LastName

FROM Students;



**PROGRAM NO.03**

**Create a table with columns for EmployeeID, Name, Salary, JoiningDate, and**

**ActiveStatus using different data types. Insert sample data and perform queries to**

**manipulate and retrieve data.**

*CREATE TABLE employee (*

*EmployeeID INT PRIMARY KEY AUTO\_INCREMENT,*

*Name VARCHAR(100) NOT NULL,*

*Salary DECIMAL(10, 2),*

*JoiningDate DATE,*

*ActiveStatus BOOLEAN*

*);*

*INSERT INTO employee (Name, Salary, JoiningDate, ActiveStatus) VALUES*

*('Bhakti Raskar', 75000.00, '2022-08-01', TRUE),*

*('Smith Marky', 68000.50, '2021-06-15', TRUE),*

*('John Doe', 54000.75, '2020-03-20', FALSE),*

*('Ashish Gaikwad', 82000.00, '2019-11-10', TRUE);*

*UPDATE employee SET Salary = 79000.00 WHERE Name = 'Bhakti Raskar';*

*DELETE FROM employee WHERE Name = 'John Doe';*

*SELECT \* FROM employee;*

*SELECT \* FROM employee WHERE ActiveStatus = TRUE;*

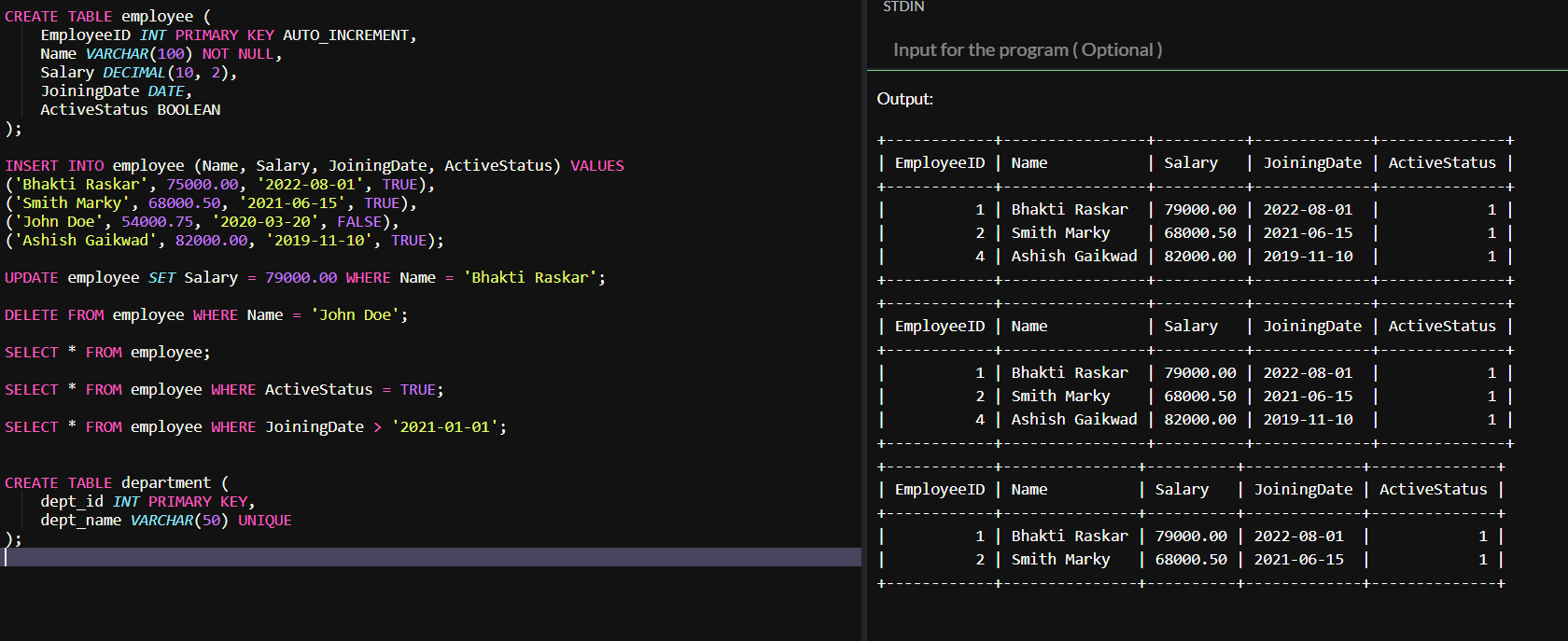
*SELECT \* FROM employee WHERE JoiningDate > '2021-01-01';*

*CREATE TABLE department (*

*dept\_id INT PRIMARY KEY,*

*dept\_name VARCHAR(50) UNIQUE*

*);*

**

**PROGRAM NO.04**

**Creating Employee Table with Constraints**

**Aim: Create a table to store employee information with constraints like Primary Key, Foreign Key, and Unique.**

CREATE TABLE department (

dept\_id INT PRIMARY KEY,

dept\_name VARCHAR(50) UNIQUE

);

CREATE TABLE employee (

emp\_id INT PRIMARY KEY,

emp\_name VARCHAR(50),

email VARCHAR(100) UNIQUE,

dept\_id INT,

FOREIGN KEY (dept\_id) REFERENCES department(dept\_id)

);

INSERT INTO department VALUES (1, 'HR');

INSERT INTO department VALUES (2, 'IT');

INSERT INTO employee VALUES (101, 'Bhakti', 'bhakti@example.com', 1);

INSERT INTO employee VALUES (102, 'Ashish', 'ashish@example.com', 2);

SELECT \* FROM department;

SELECT \* FROM employee;

-- **Invalid Data Tests (Uncomment one by one to test)**

**-- Duplicate Primary Key (emp\_id = 101 already exists)**

**-- INSERT INTO employee VALUES (101, 'Sahil', 'sahil@example.com', 1);**

**-- Duplicate Unique email (email = 'bhakti@example.com' already used)**

**-- INSERT INTO employee VALUES (103, 'Aman', 'bhakti@example.com', 1);**

**-- Invalid Foreign Key (dept\_id = 5 does not exist)**

**-- INSERT INTO employee VALUES (104, 'Meera', 'meera@example.com', 5);**



**PROGRAM NO.05**

Testing Employee Constraints

**Aim: To test constraints like PRIMARY KEY, UNIQUE, and CHECK by inserting invalid data into the Employee table.**

Code:

CREATE TABLE customer (

cust\_id INT PRIMARY KEY,

cust\_name VARCHAR(50) NOT NULL,

age INT CHECK (age >= 18),

email VARCHAR(100) UNIQUE NOT NULL,

city VARCHAR(50) DEFAULT 'Pune'

);

INSERT INTO customer VALUES (1, 'Bhakti', 20, 'bhakti@example.com', 'Delhi');

INSERT INTO customer (cust\_id, cust\_name, age, email)

VALUES (2, 'Ashu', 25, 'ashu@example.com');

SELECT \* FROM customer;

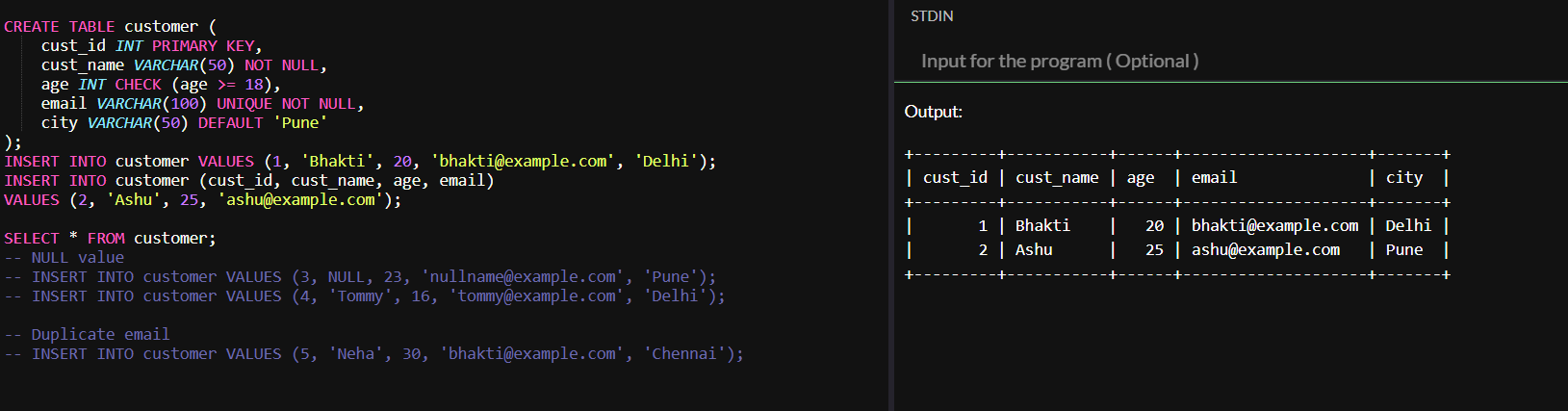
-- NULL value

-- INSERT INTO customer VALUES (3, NULL, 23, 'nullname@example.com', 'Pune');

-- INSERT INTO customer VALUES (4, 'Tommy', 16, 'tommy@example.com', 'Delhi');

-- Duplicate email

-- INSERT INTO customer VALUES (5, 'Neha', 30, 'bhakti@example.com', 'Chennai');



**PROGRAM NO.06**

**Use DDL commands to create tables and DML commands to insert, update & delete**

**data. Write SELECT queries to retrieve and verify data changes..**

*CREATE TABLE customers (*

*cust\_id INT PRIMARY KEY,*

*cust\_name VARCHAR(50) NOT NULL,*

*city VARCHAR(50)*

*);*

*CREATE TABLE orders (*

*order\_id INT PRIMARY KEY,*

*cust\_id INT,*

*product VARCHAR(50),*

*amount DECIMAL(10, 2),*

*FOREIGN KEY (cust\_id) REFERENCES customers(cust\_id) ON DELETE CASCADE*

*);*

*INSERT INTO customers VALUES (1, 'Kunal', 'Nashik');*

*INSERT INTO customers VALUES (2, 'Riya', 'Pune');*

*INSERT INTO customers VALUES (3, 'Aman', 'Mumbai');*

*INSERT INTO orders VALUES (101, 1, 'Laptop', 55000.00);*

*INSERT INTO orders VALUES (102, 2, 'Phone', 20000.00);*

*INSERT INTO orders VALUES (103, 1, 'Keyboard', 1500.00);*

*SELECT \* FROM customers;*

*SELECT \* FROM orders;*

*UPDATE customers SET city = 'Aurangabad' WHERE cust\_id = 3;*

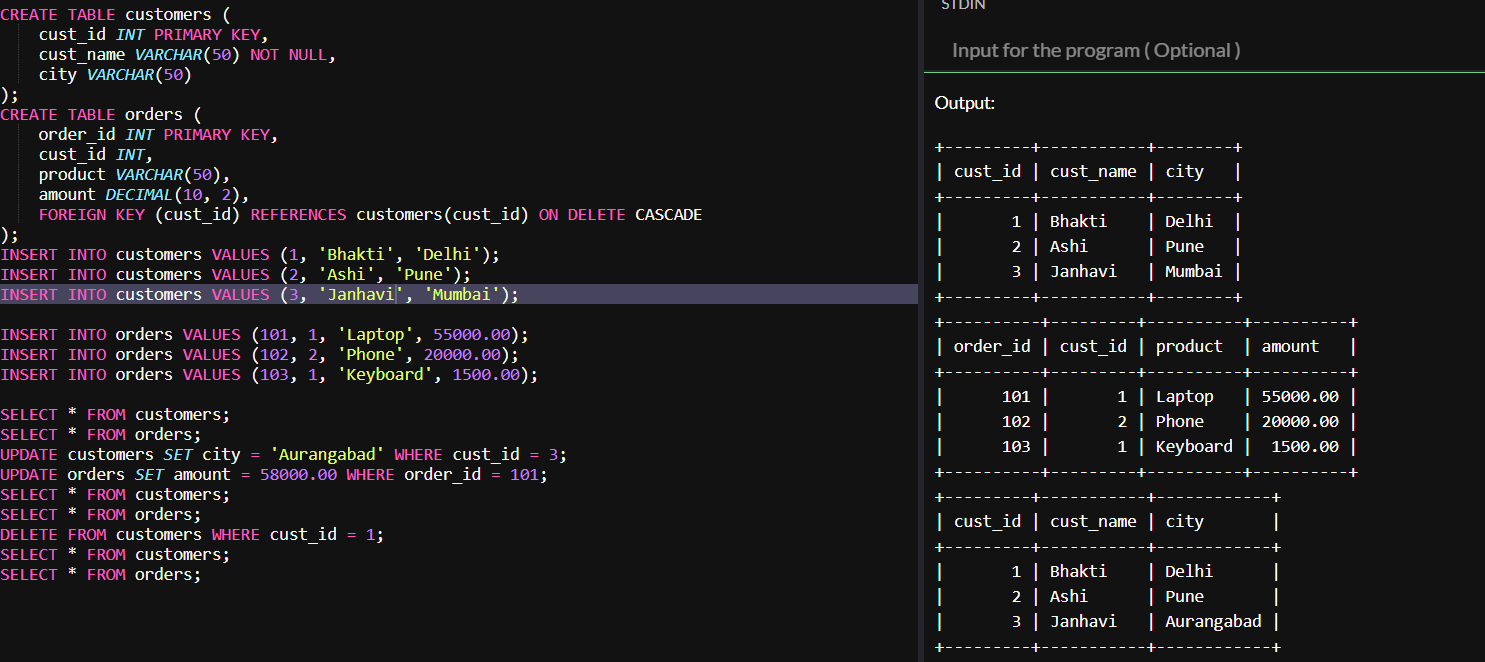
*UPDATE orders SET amount = 58000.00 WHERE order\_id = 101;*

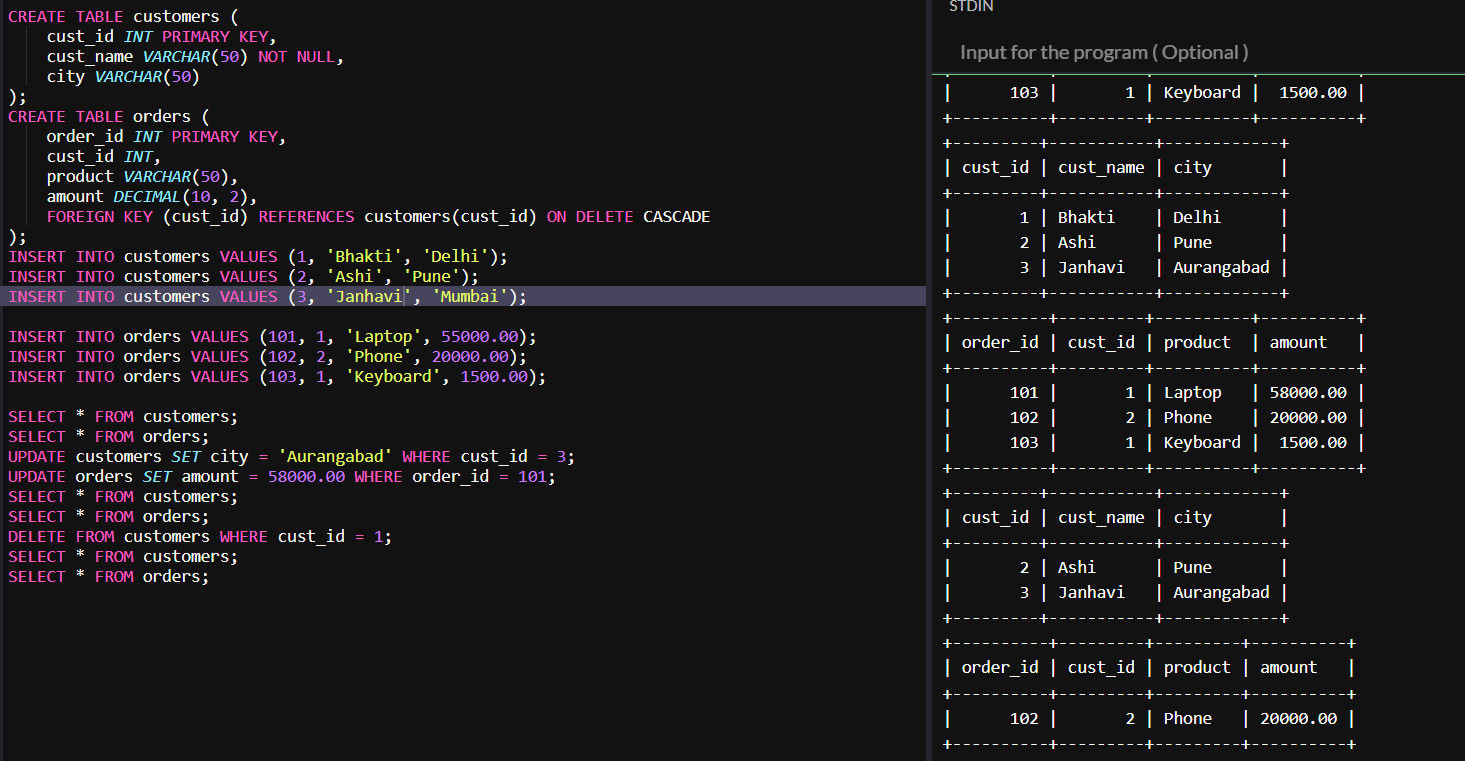
*SELECT \* FROM customers;*

*SELECT \* FROM orders;*

*DELETE FROM customers WHERE cust\_id = 1;*

*SELECT \* FROM customers;*

*SELECT \* FROM orders;* ******

******

**PROGRAM NO.07**

**7. Create a Sales table and use aggregate functions like COUNT, SUM, AVG, MIN, and**

**MAX to summarize sales data and calculate statistics.**

*CREATE TABLE Sales (*

*SaleID INT PRIMARY KEY,*

*ProductName VARCHAR(50),*

*Quantity INT,*

*Price DECIMAL(10, 2),*

*SaleDate DATE*

*);*

*INSERT INTO Sales VALUES (1, 'Laptop', 2, 60000.00, '2025-04-01');*

*INSERT INTO Sales VALUES (2, 'Mouse', 5, 500.00, '2025-04-02');*

*INSERT INTO Sales VALUES (3, 'Keyboard', 3, 1500.00, '2025-04-02');*

*INSERT INTO Sales VALUES (4, 'Monitor', 1, 12000.00, '2025-04-03');*

*INSERT INTO Sales VALUES (5, 'Laptop', 1, 60000.00, '2025-04-03');*

*INSERT INTO Sales VALUES (6, 'Mouse', 10, 500.00, '2025-04-04');*

*SELECT COUNT(\*) AS TotalSales FROM Sales;*

*SELECT SUM(Quantity \* Price) AS TotalRevenue FROM Sales;*

*SELECT AVG(Price) AS AveragePrice FROM Sales;*

*SELECT MIN(Price) AS MinimumPrice FROM Sales;*

*SELECT MAX(Price) AS MaximumPrice FROM Sales;*

*SELECT ProductName, SUM(Quantity) AS TotalUnitsSold*

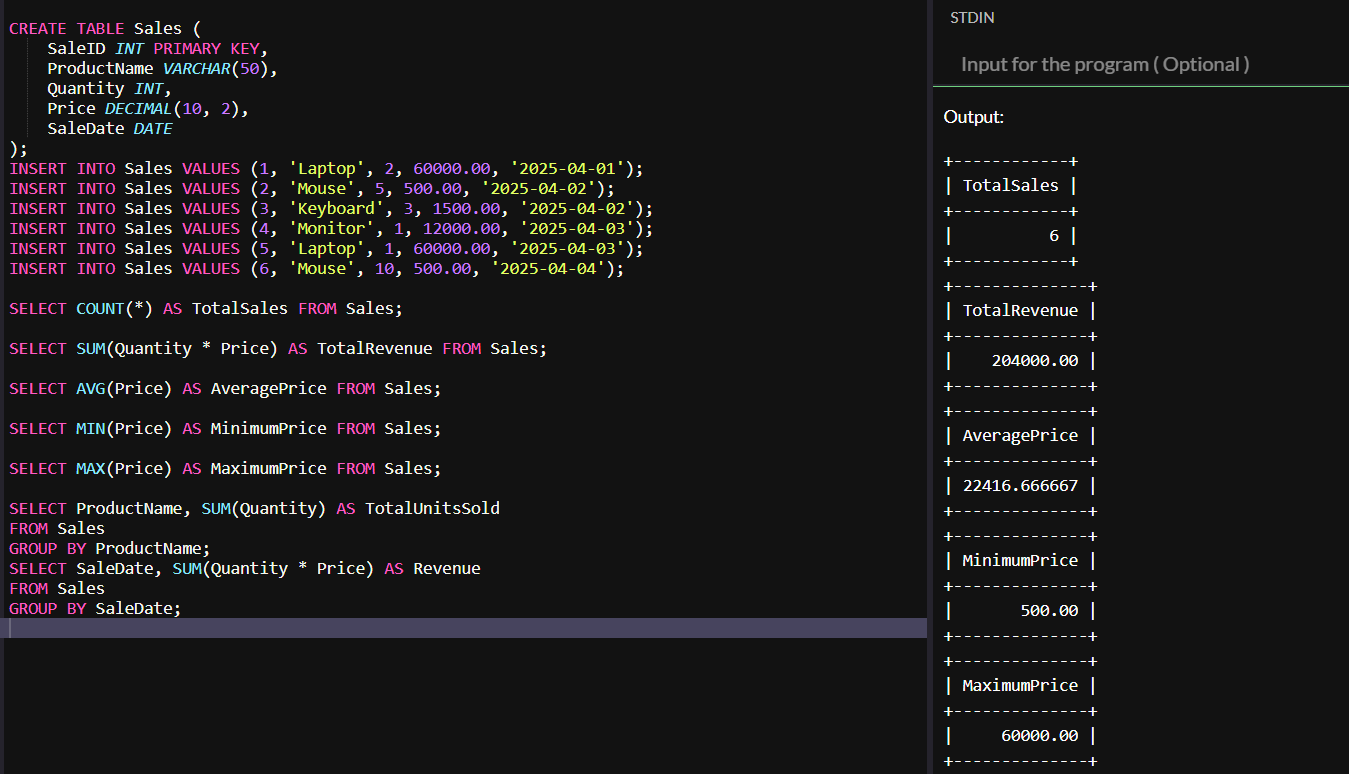
*FROM Sales*

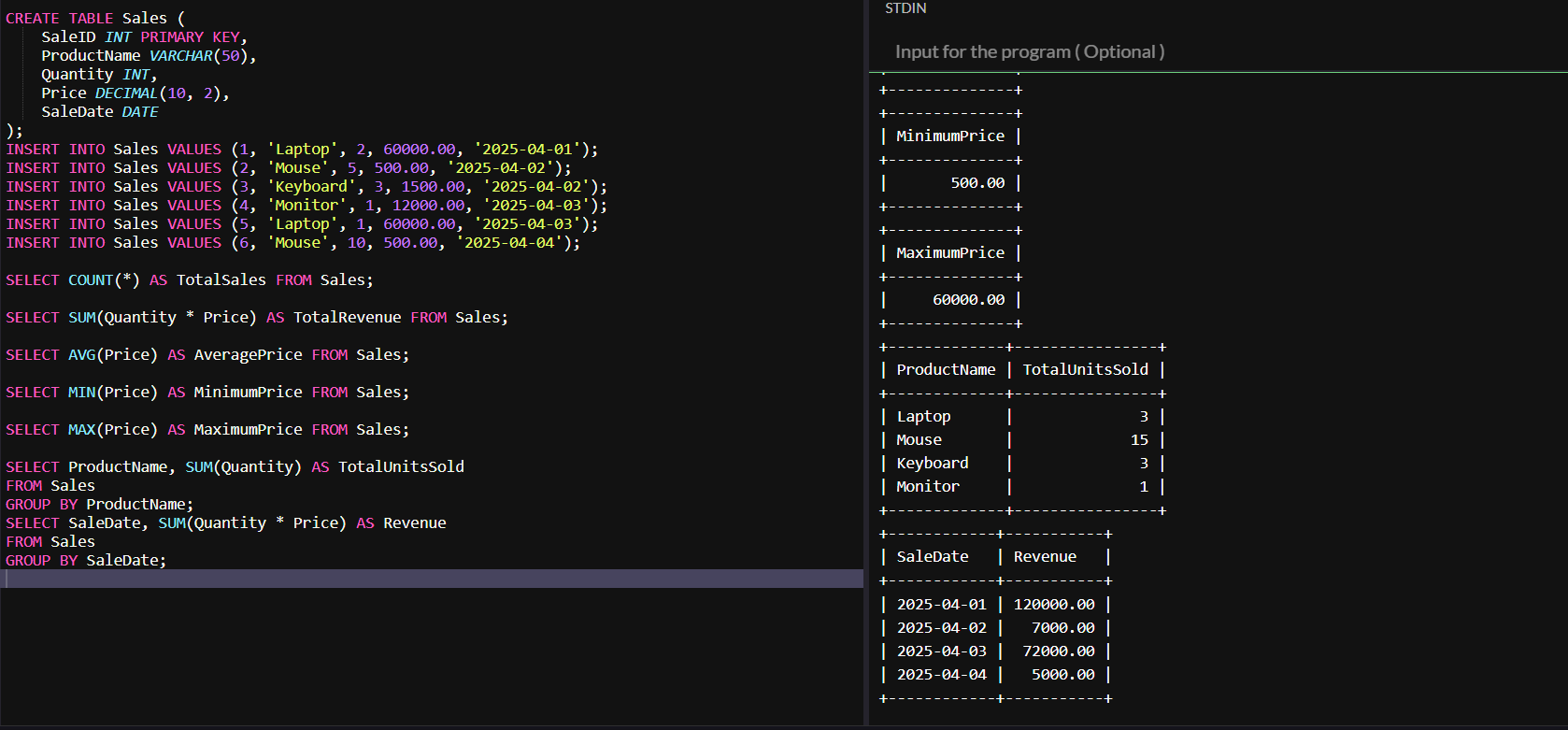
*GROUP BY ProductName;*

*SELECT SaleDate, SUM(Quantity \* Price) AS Revenue*

*FROM Sales*

*GROUP BY SaleDate;*

****

****

**PROGRAM NO.08**

**Given Customers and Orders tables, write SQL queries to perform INNERJOIN LEFTJOIN, and RIGHTJOIN to retrieve combined data for customer orders..**

*CREATE TABLE Customers (*

*CustomerID INT PRIMARY KEY,*

*Name VARCHAR(50),*

*City VARCHAR(50)*

*);*

*CREATE TABLE Orders (*

*OrderID INT PRIMARY KEY,*

*CustomerID INT,*

*Product VARCHAR(50),*

*OrderDate DATE,*

*FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)*

*);*

*INSERT INTO Customers (CustomerID, Name, City) VALUES*

*(1, 'Bhakti', 'Delhi'),*

*(2, 'Ashu', 'Pune'),*

*(3, 'Priya', 'Nashik'),*

*(4, 'Aditya', 'Nagpur');*

*INSERT INTO Orders (OrderID, CustomerID, Product, OrderDate) VALUES*

*(101, 1, 'Laptop', '2025-02-01'),*

*(102, 2, 'Mobile', '2025-02-03'),*

*(103, 1, 'Tablet', '2025-02-04'),*

*(104, 3, 'Headphones', '2025-02-05');*

*SELECT Customers.CustomerID, Name, City, Product, OrderDate*

*FROM Customers*

*INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;*

*SELECT Customers.CustomerID, Name, City, Product, OrderDate*

*FROM Customers*

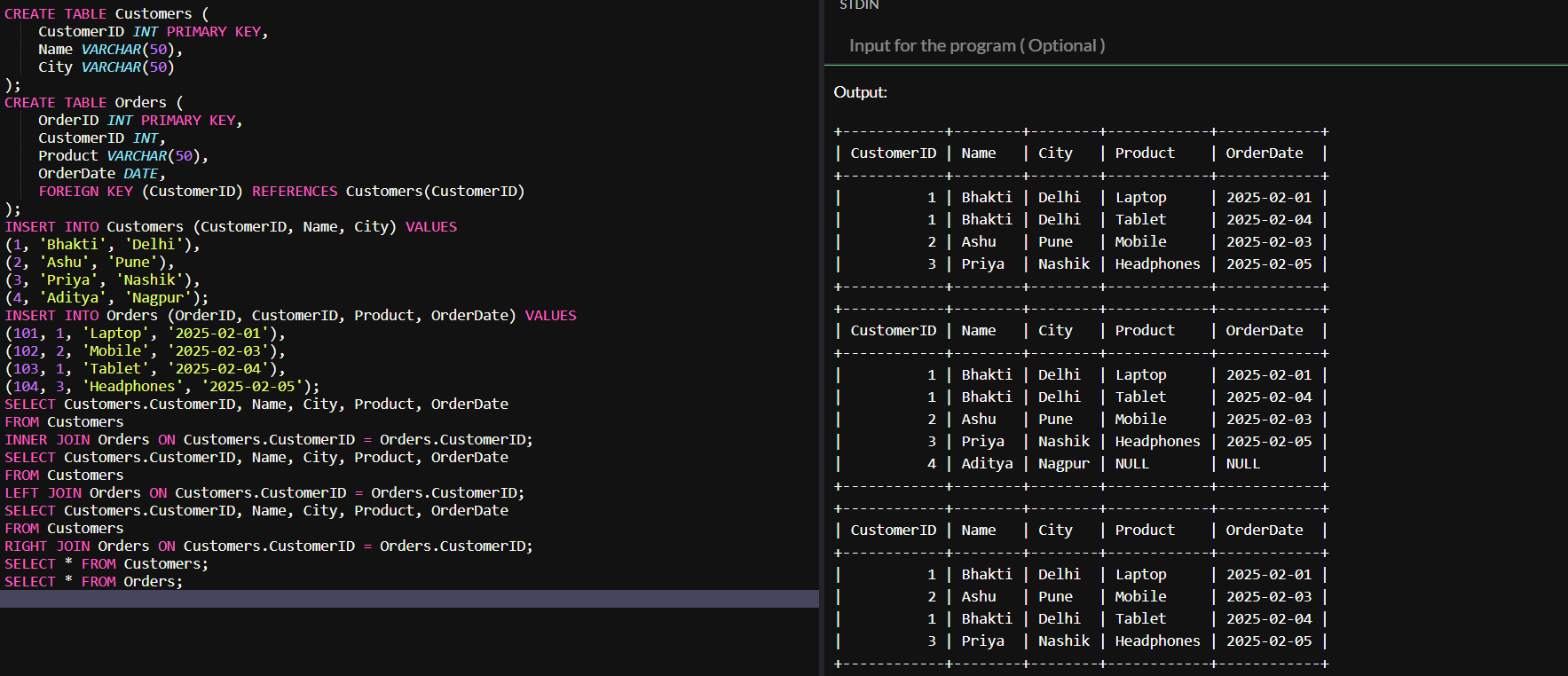
*LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;*

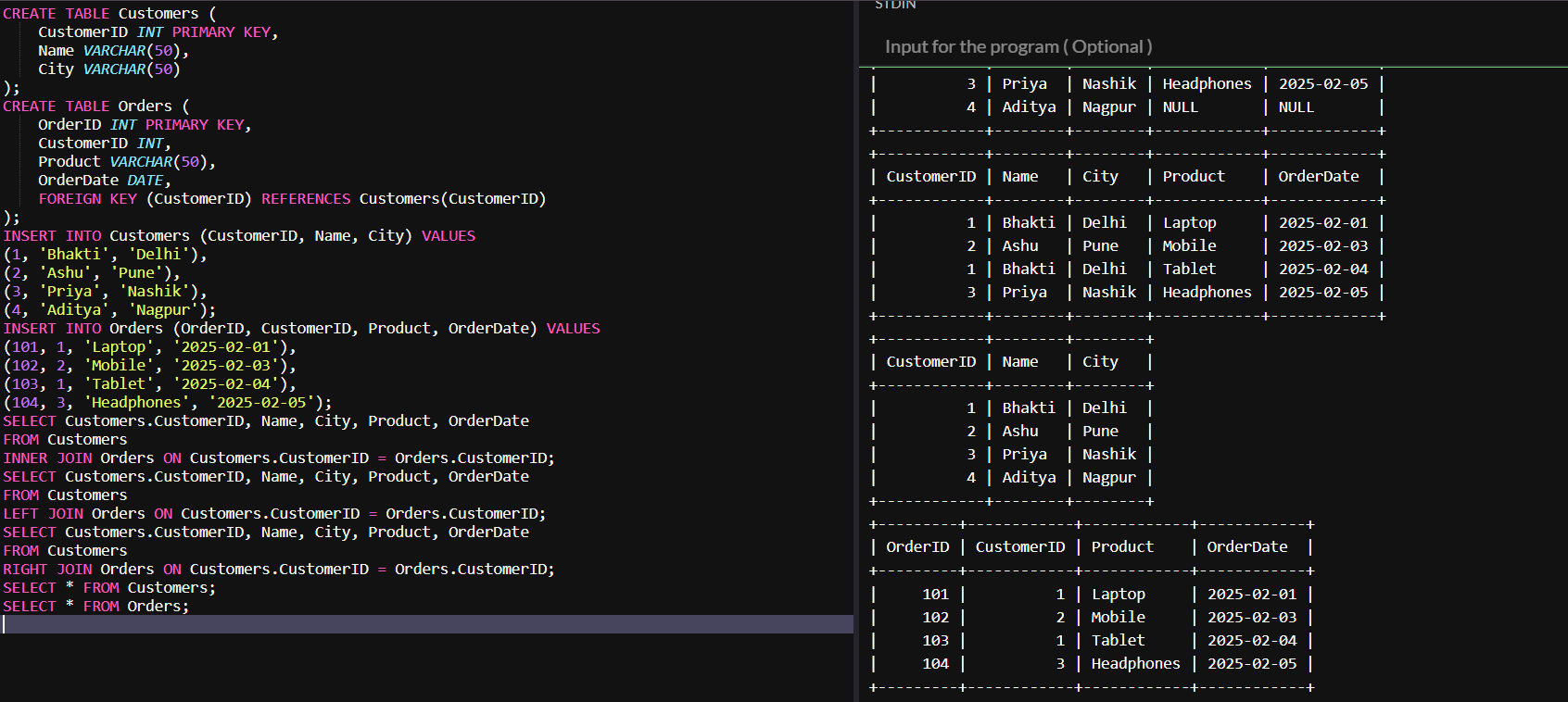
*SELECT Customers.CustomerID, Name, City, Product, OrderDate*

*FROM Customers*

*RIGHT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;*

*SELECT \* FROM Customers;*

*SELECT \* FROM Orders;*

**