**Week2: PL/SQL PRORAMMING**

**Exercise 3: Stored Procedures**

**-- SCANARIO 1**

**a)CODE INPUT=>**

**CREATE DATABASE IF NOT EXISTS bank\_db;**

**USE bank\_db;**

**CREATE TABLE IF NOT EXISTS SavingsAccount (**

**account\_id INT PRIMARY KEY,**

**account\_holder\_name VARCHAR(100),**

**balance DECIMAL(15,2) CHECK (balance >= 0)**

**);**

**CREATE TABLE IF NOT EXISTS Employee (**

**employee\_id INT PRIMARY KEY,**

**employee\_name VARCHAR(100),**

**department VARCHAR(100),**

**salary DECIMAL(15,2) CHECK (salary >= 0)**

**);**

**INSERT INTO SavingsAccount VALUES**

**(101, 'Amit Sharma', 5000.00),**

**(102, 'Riya Verma', 7000.00),**

**(103, 'Kunal Mehta', 3000.00);**

**INSERT INTO Employee VALUES**

**(1, 'Neha Singh', 'Sales', 40000.00),**

**(2, 'Rahul Jain', 'IT', 55000.00),**

**(3, 'Anjali Roy', 'Sales', 45000.00);**

**DELIMITER //**

**CREATE PROCEDURE ProcessMonthlyInterest()**

**BEGIN**

**UPDATE SavingsAccount**

**SET balance = balance + (balance \* 0.01);**

**SELECT 'Monthly interest of 1% has been applied to all accounts.' AS message;**

**END //**

**DELIMITER ;**

**CALL ProcessMonthlyInterest();**

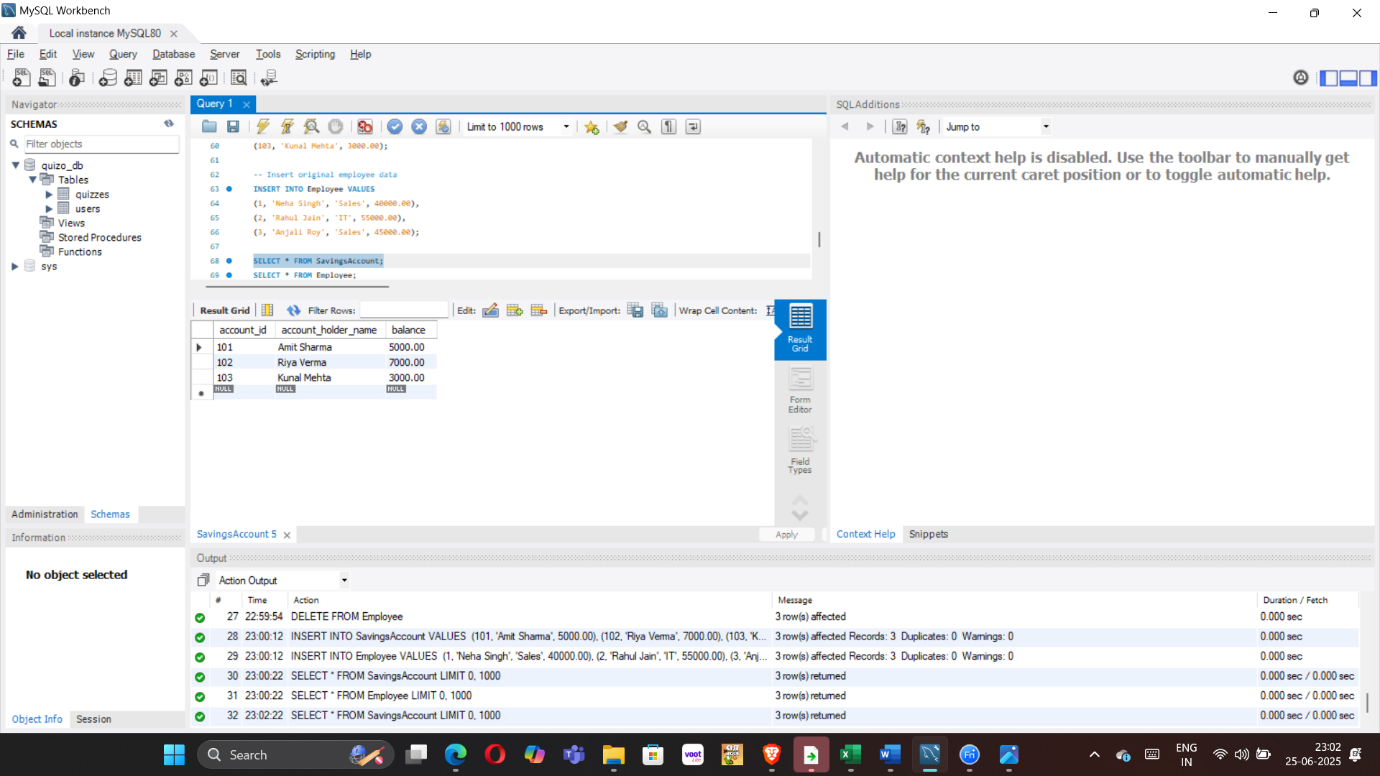
**SET SQL\_SAFE\_UPDATES = 0;**

**CALL ProcessMonthlyInterest();**

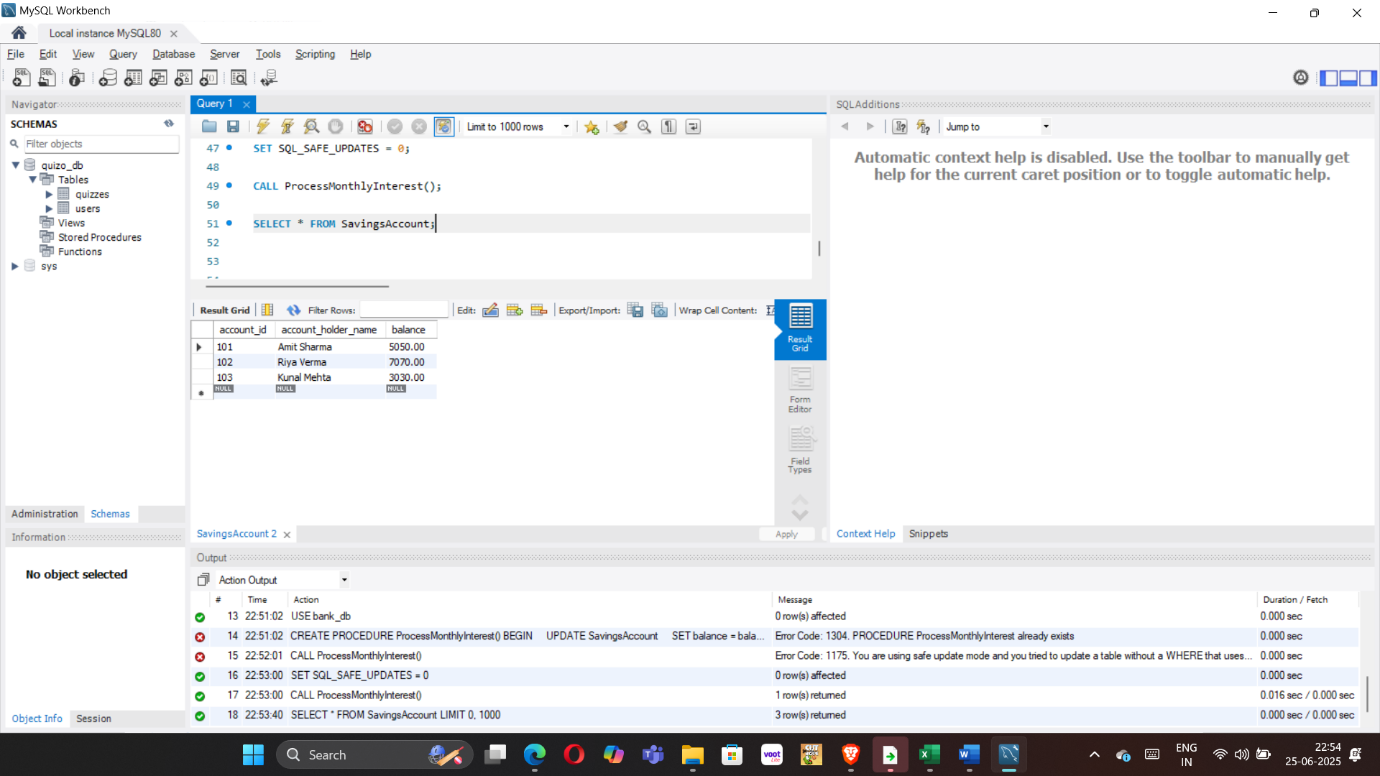
**SELECT \* FROM SavingsAccount;**

**b)OUTPUT**

**=>SAVINGS TABLE BEFORE RUNNING THE QUERY**

****

**=>SAVINGS TABLE AFTER RUNNING THE QUERY**

****

**-- SCANARIO 2**

**a)CODE INPUT=>**

**DELIMITER //**

**CREATE PROCEDURE UpdateEmployeeBonus(**

**IN dept\_name VARCHAR(100),**

**IN bonus\_percent DECIMAL(5,2)**

**)**

**BEGIN**

**UPDATE Employee**

**SET salary = salary + (salary \* bonus\_percent / 100)**

**WHERE department = dept\_name;**

**SELECT CONCAT('Bonus of ', bonus\_percent, '% applied to department: ', dept\_name) AS message;**

**END //**

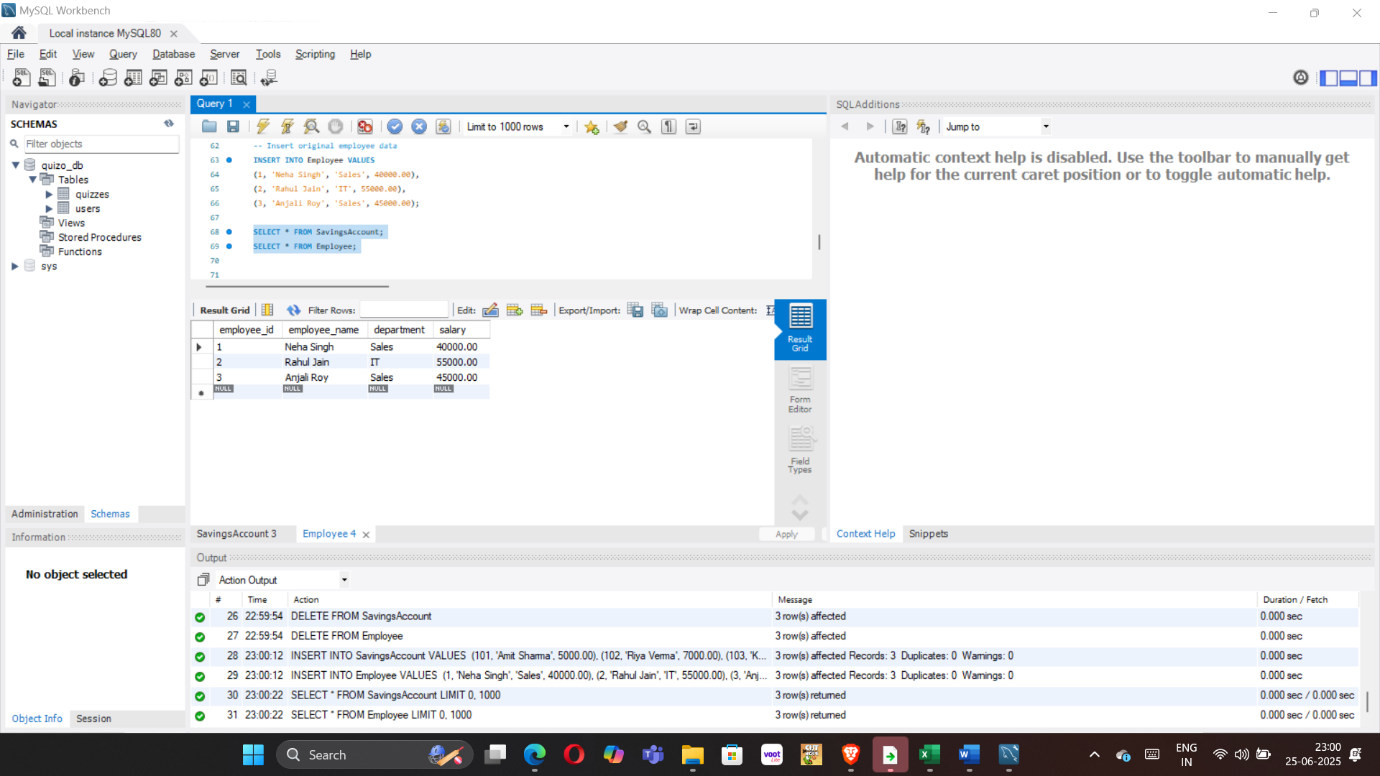
**DELIMITER ;**

**CALL UpdateEmployeeBonus('Sales', 10.0);**

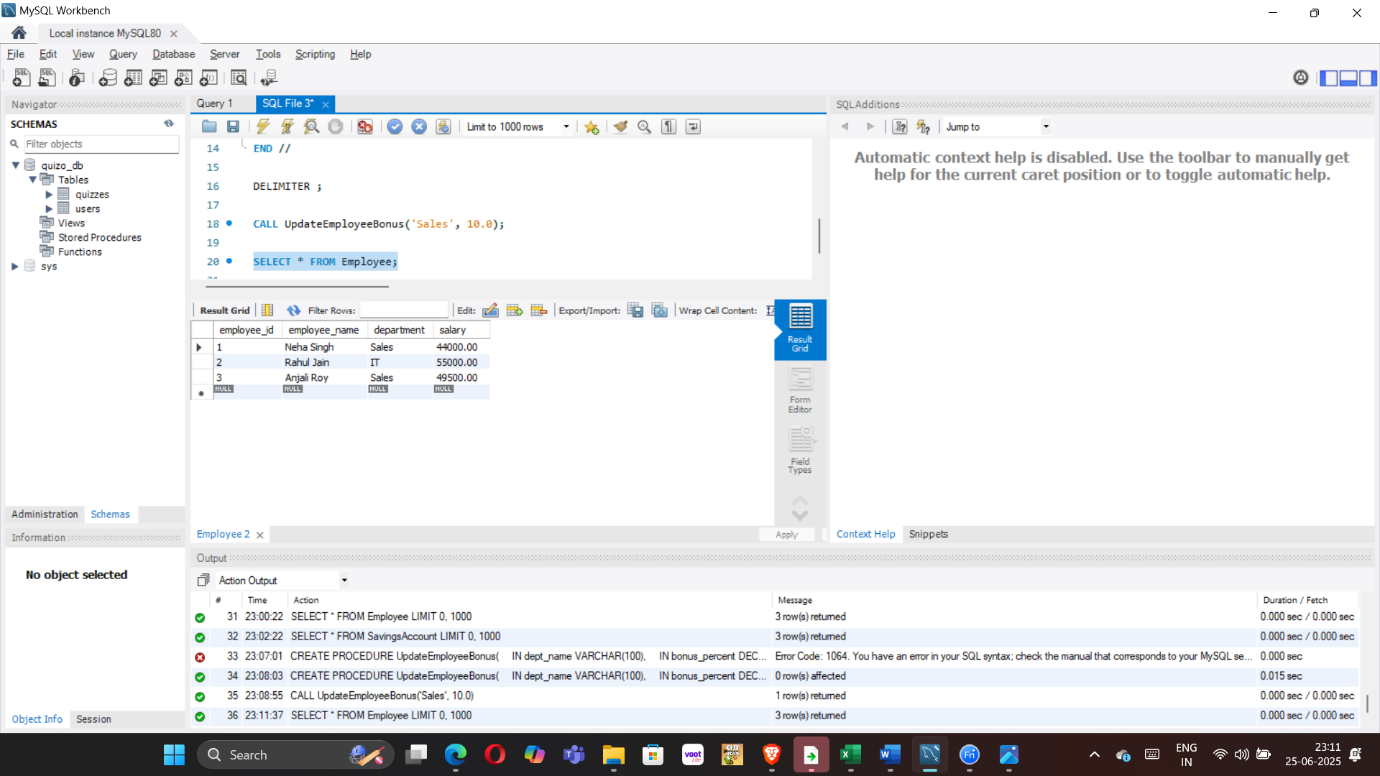
**SELECT \* FROM Employee;**

**b)OUTPUT**

**=>EMPLOYEE TABLE BEFORE RUNNING THE QUERY**

****

**=>EMPLOYEE TABLE AFTER RUNNING THE QUERY**

****

**-- SCANARIO 3**

**a)CODE INPUT=>**

**DELIMITER //**

**CREATE PROCEDURE TransferFunds(**

**IN from\_account\_id INT,**

**IN to\_account\_id INT,**

**IN transfer\_amount DECIMAL(15,2)**

**)**

**BEGIN**

**DECLARE sender\_balance DECIMAL(15,2);**

**-- Get the balance of the source account**

**SELECT balance INTO sender\_balance**

**FROM SavingsAccount**

**WHERE account\_id = from\_account\_id;**

**-- Check if the source account has enough balance**

**IF sender\_balance < transfer\_amount THEN**

**SIGNAL SQLSTATE '45000'**

**SET MESSAGE\_TEXT = 'Insufficient funds in the source account.';**

**ELSE**

**-- Deduct amount from sender**

**UPDATE SavingsAccount**

**SET balance = balance - transfer\_amount**

**WHERE account\_id = from\_account\_id;**

**-- Add amount to receiver**

**UPDATE SavingsAccount**

**SET balance = balance + transfer\_amount**

**WHERE account\_id = to\_account\_id;**

**-- Confirmation message**

**SELECT CONCAT('₹', transfer\_amount, ' transferred from Account ', from\_account\_id,**

**' to Account ', to\_account\_id) AS message;**

**END IF;**

**END //**

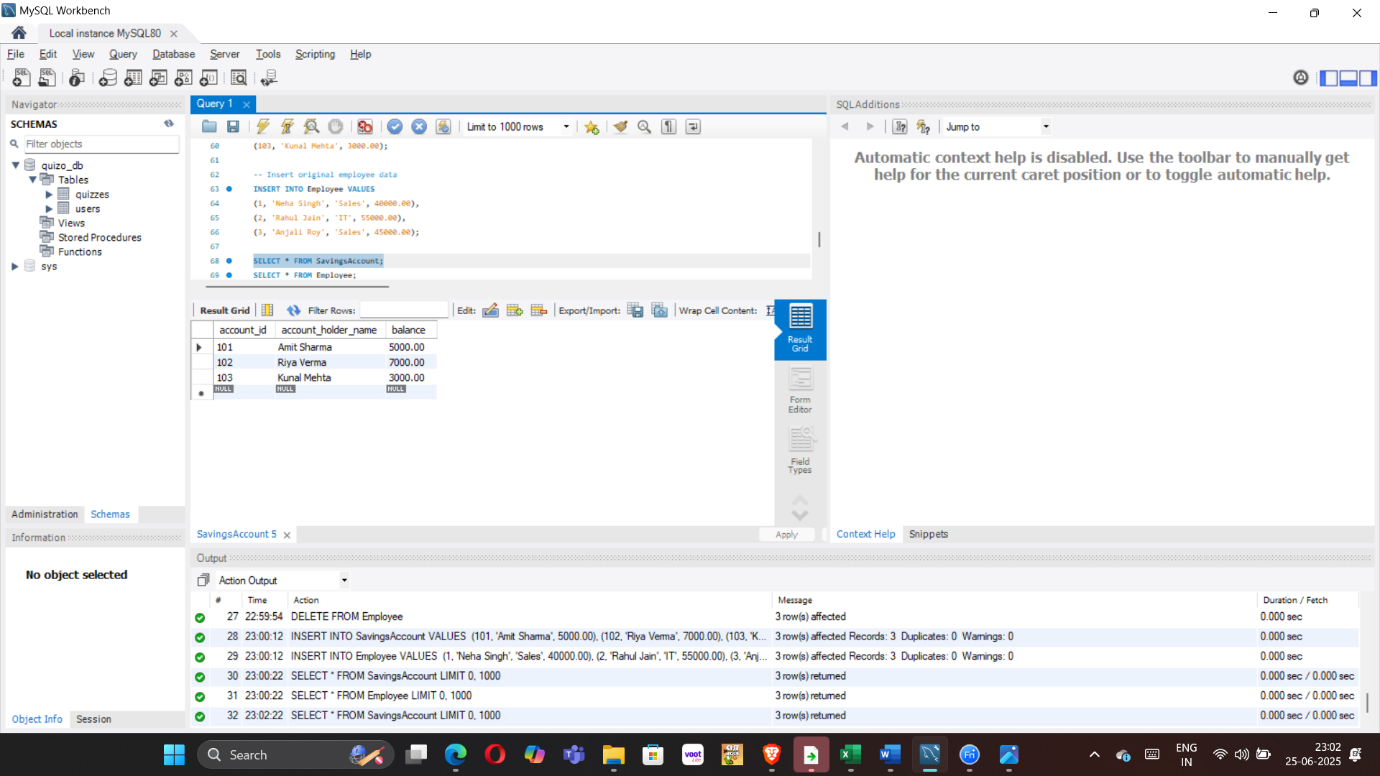
**DELIMITER ;**

**CALL TransferFunds(101, 102, 2000.00);**

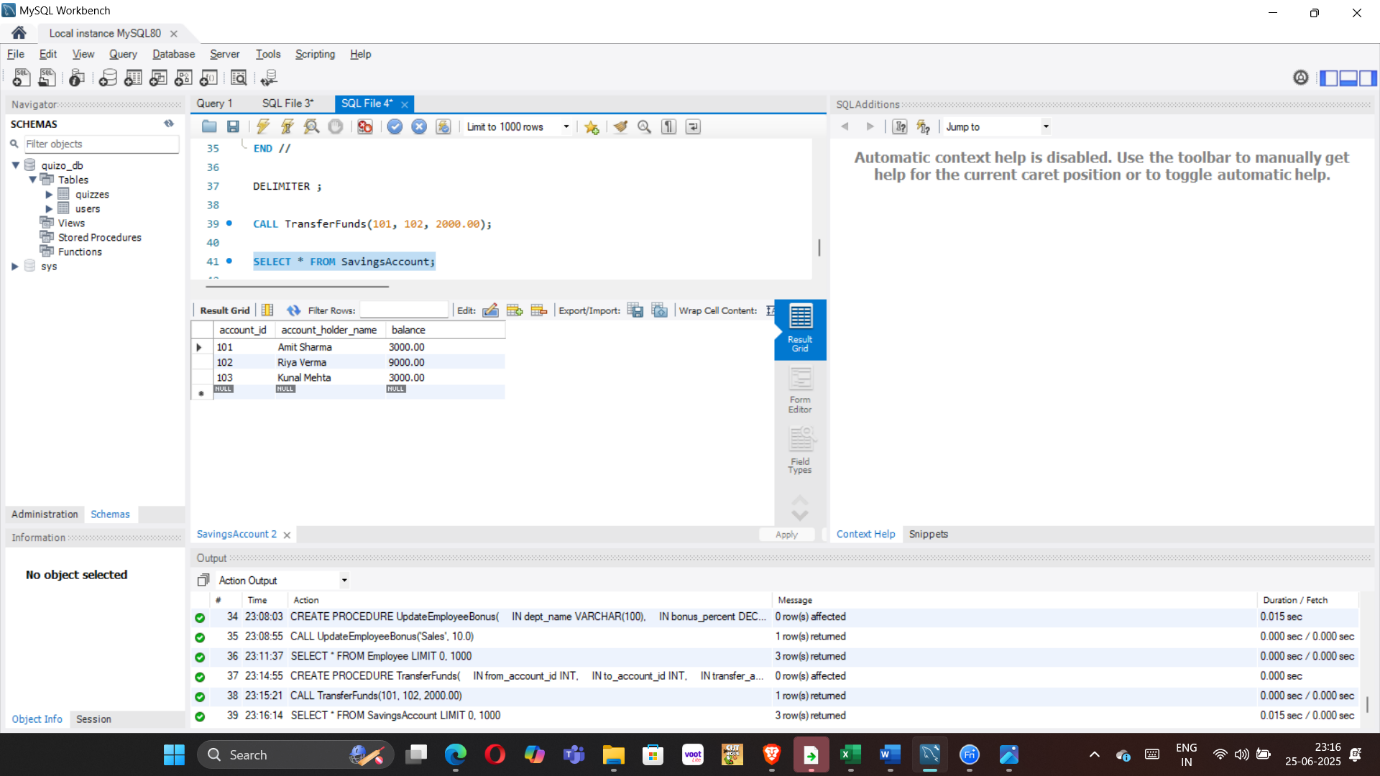
**SELECT \* FROM SavingsAccount;**

**b)OUTPUT**

**=>SAVINGS ACCOUNT TABLE BEFORE RUNNING THE QUERY**

****

**=>SAVINGS ACCOUNT TABLE AFTER RUNNING THE QUERY**

****

**Exercise 1: Control Structures**

**-- SCANARIO 1**

**a)CODE INPUT=>**

**CREATE DATABASE bank;**

**USE bank;**

**CREATE TABLE customers (**

**id INT PRIMARY KEY,**

**name VARCHAR(100),**

**age INT,**

**balance DECIMAL(10, 2),**

**interestRate DECIMAL(4,2),**

**IsVIP BOOLEAN DEFAULT FALSE**

**);**

**CREATE TABLE loans (**

**loan\_id INT PRIMARY KEY,**

**customer\_id INT,**

**due\_date DATE,**

**FOREIGN KEY (customer\_id) REFERENCES customers(id)**

**);**

**INSERT INTO customers (id, name, age, balance, interestRate, IsVIP) VALUES**

**(1, 'Anshul', 11, 2000.00, 1.5, FALSE),**

**(2, 'Paro', 12, 7700.00, 9.0, FALSE),**

**(3, 'Uday', 13, 11000.00, 6.0, FALSE),**

**(4, 'keena', 14,5000.00, 5.5, FALSE);**

**INSERT INTO loans (loan\_id, customer\_id, due\_date) VALUES**

**(101, 1, CURDATE() + INTERVAL 10 DAY), -- due in 10 days**

**(102, 2, CURDATE() + INTERVAL 29 DAY), -- due in 29 days**

**(103, 3, CURDATE() + INTERVAL 40 DAY),**

**(104, 3, CURDATE() + INTERVAL 50 DAY); -- due after 30 days**

**SELECT \* FROM customers;**

**DELIMITER //**

**CREATE PROCEDURE ApplySeniorDiscount()**

**BEGIN**

**DECLARE done INT DEFAULT FALSE;**

**DECLARE cust\_id INT;**

**DECLARE cust\_age INT;**

**DECLARE cur CURSOR FOR**

**SELECT id, age FROM customers;**

**DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;**

**OPEN cur;**

**read\_loop: LOOP**

**FETCH cur INTO cust\_id, cust\_age;**

**IF done THEN**

**LEAVE read\_loop;**

**END IF;**

**IF cust\_age > 60 THEN**

**UPDATE customers**

**SET interestRate = interestRate - 1.00**

**WHERE id = cust\_id;**

**END IF;**

**END LOOP;**

**CLOSE cur;**

**END //**

**DELIMITER ;**

**-- To run:**

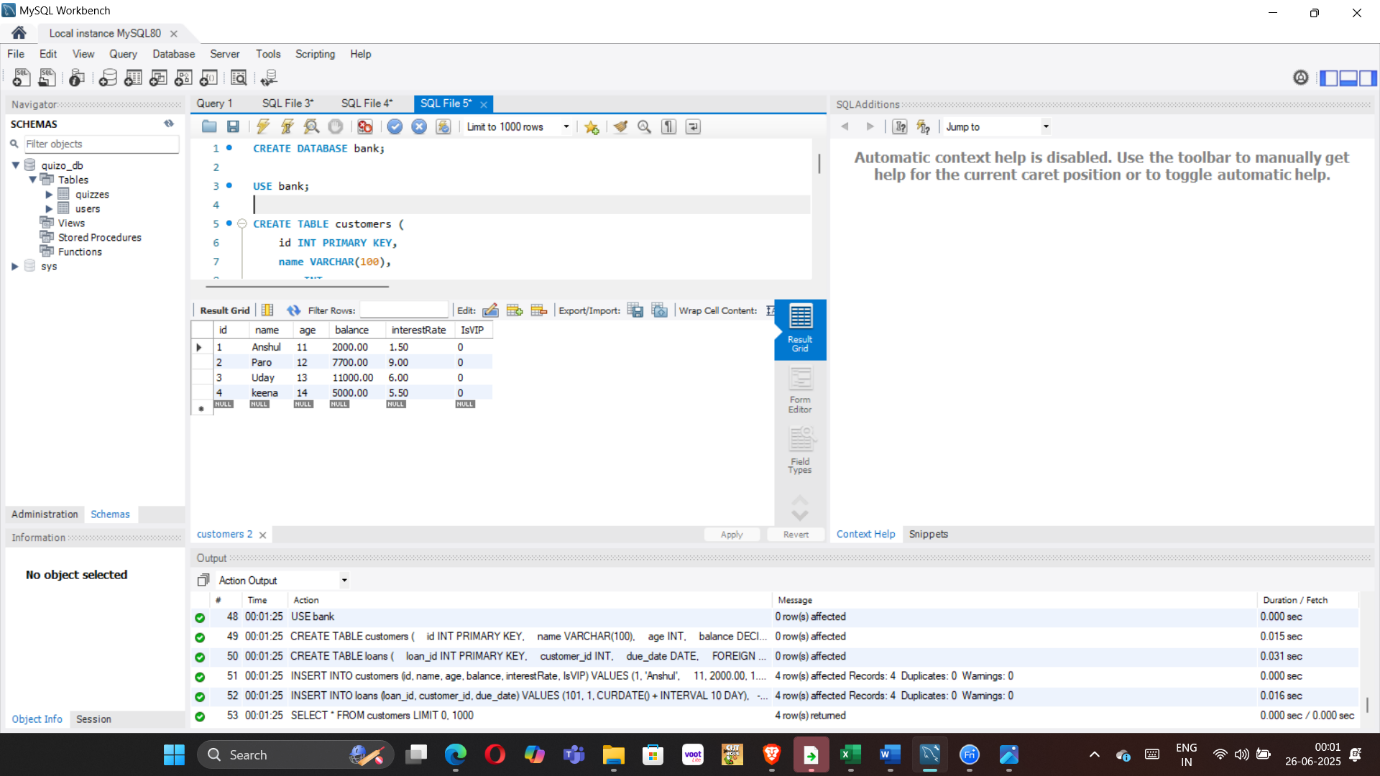
**CALL ApplySeniorDiscount();**

**SELECT name, interestRate FROM customers WHERE age > 60;**

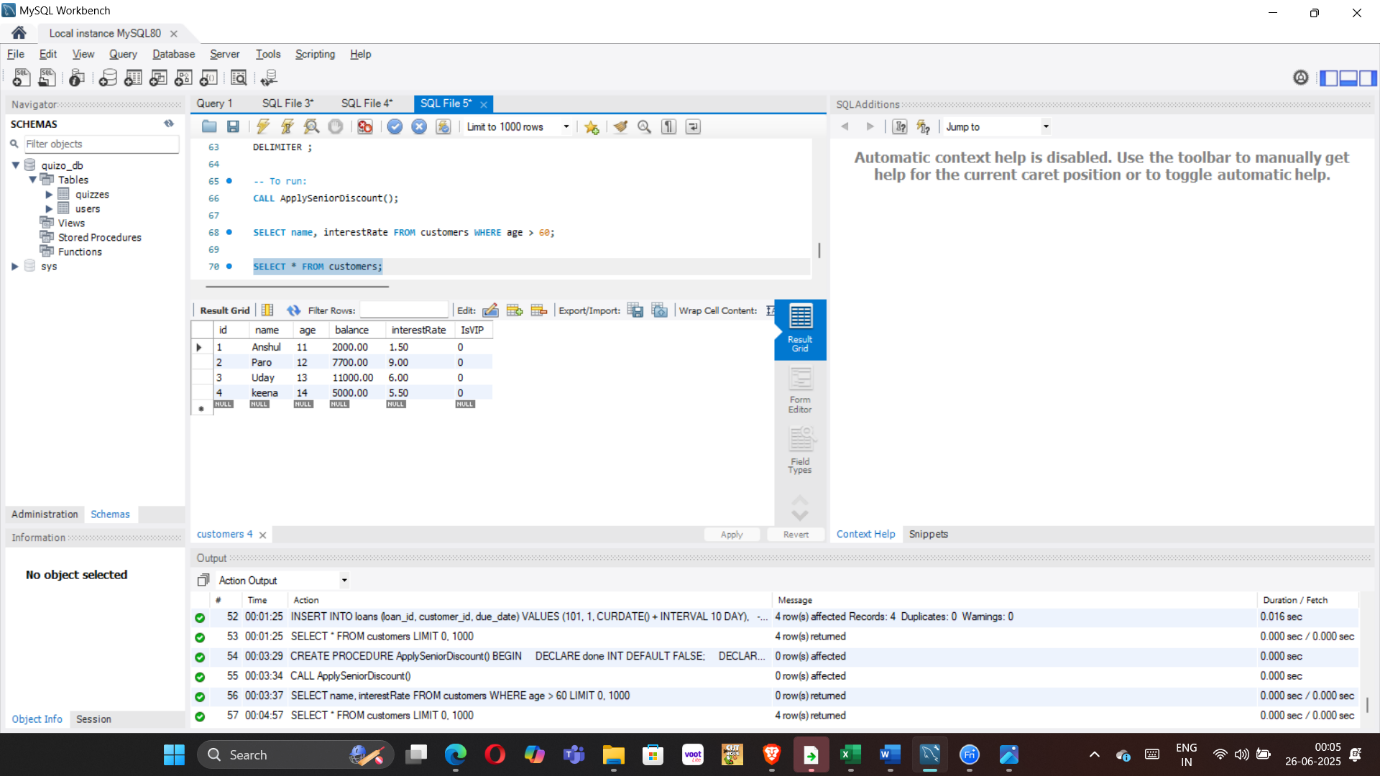
**SELECT \* FROM customers;**

**b)OUTPUT**

**=>CUSTOMERS TABLE BEFORE RUNNING THE QUERY**

****

**=>CUSTOMERS TABLE BEFORE RUNNING THE QUERY**

****

**-- SCANARIO 2**

**a)CODE INPUT**

**DELIMITER //**

**CREATE PROCEDURE PromoteVIPCustomers()**

**BEGIN**

**UPDATE customers**

**SET IsVIP = TRUE**

**WHERE balance > 10000;**

**END //**

**DELIMITER ;**

**CALL PromoteVIPCustomers();**

**SELECT \* FROM customers;**

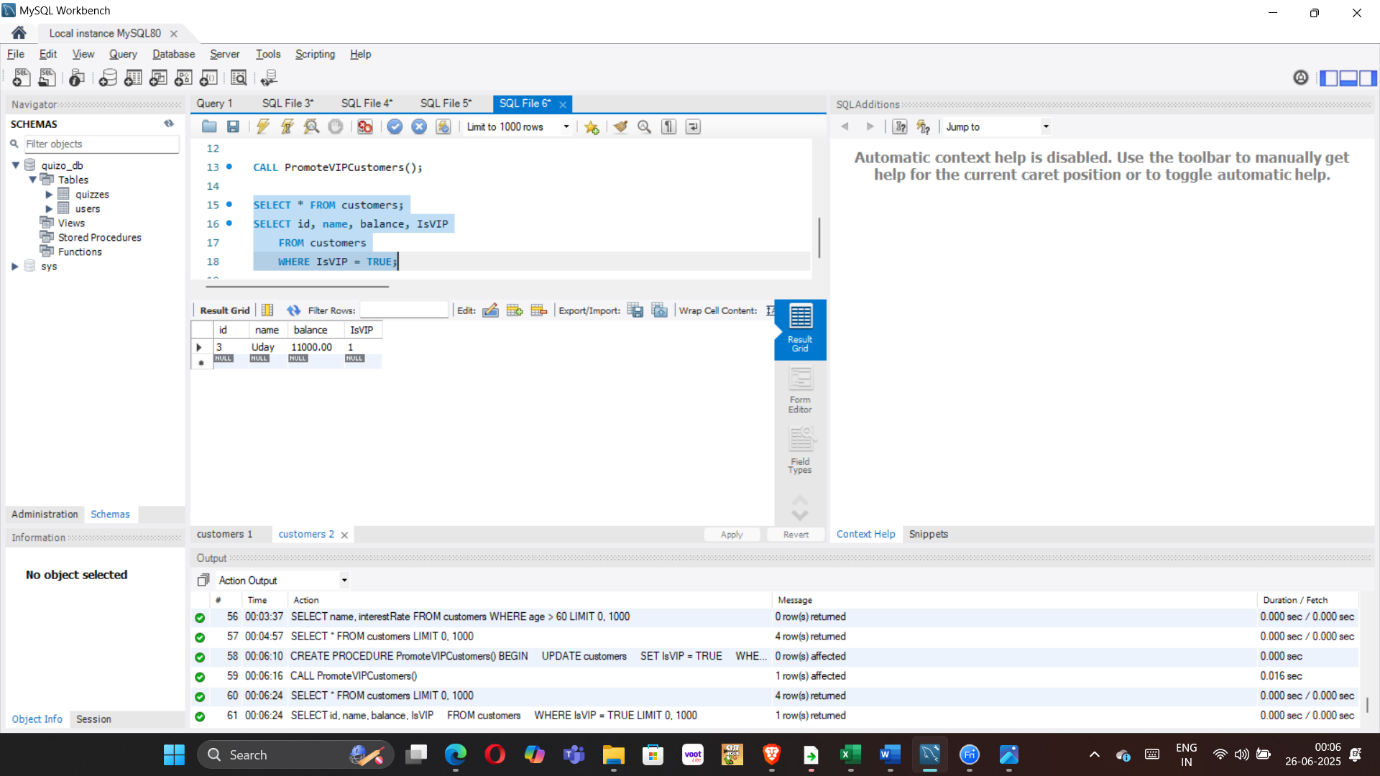
**SELECT id, name, balance, IsVIP**

**FROM customers**

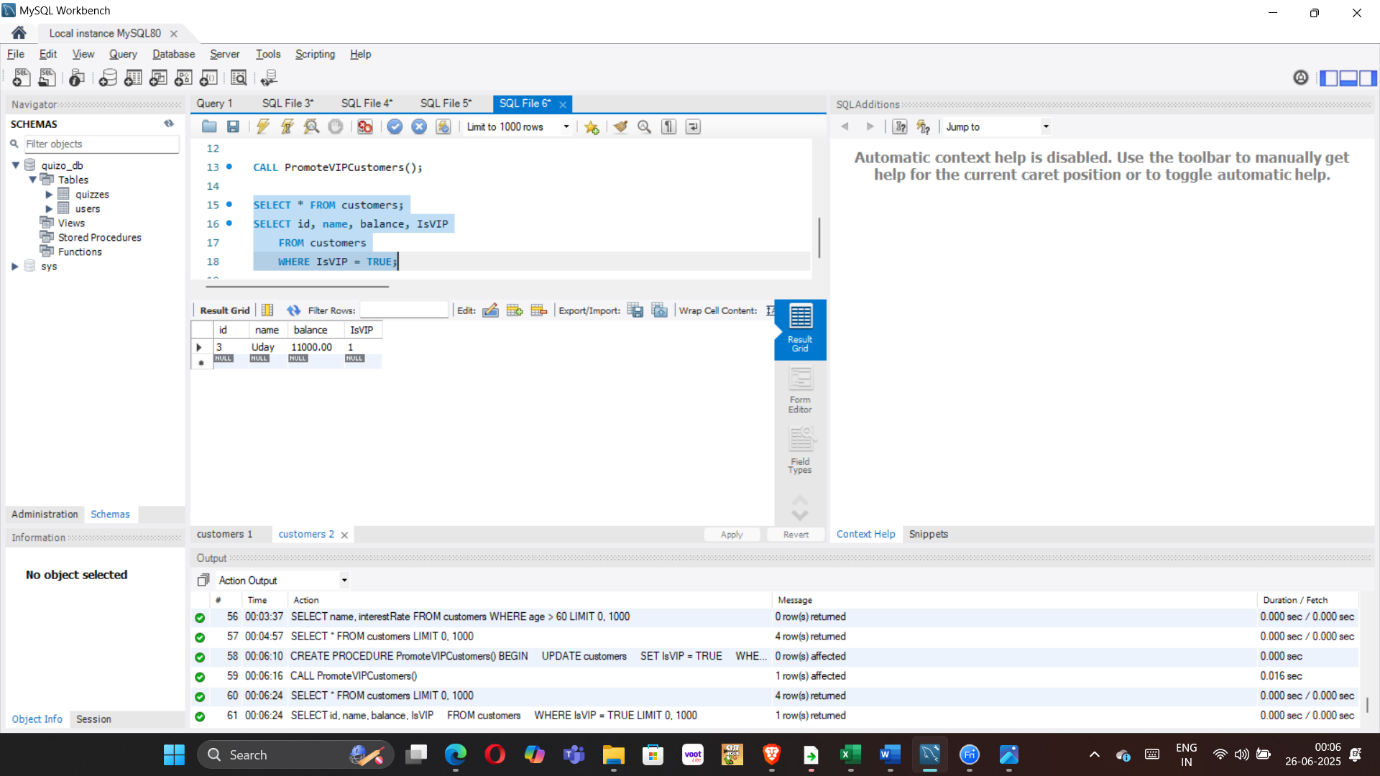
**WHERE IsVIP = TRUE;**

**b)OUTPUT**

**=>CUSTOMERS TABLE AFTER RUNNING THE QUERY**



**=>CUSTOMERS TABLE AFTER RUNNING THE QUERY**



**-- SCANARIO 3**

**a)CODE INPUT**

**DELIMITER //**

**CREATE PROCEDURE SendLoanReminders()**

**BEGIN**

**DECLARE done INT DEFAULT FALSE;**

**DECLARE cust\_id INT;**

**DECLARE due\_dt DATE;**

**DECLARE cur CURSOR FOR**

**SELECT customer\_id, due\_date FROM loans**

**WHERE due\_date BETWEEN CURDATE() AND DATE\_ADD(CURDATE(), INTERVAL 30 DAY);**

**DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;**

**OPEN cur;**

**reminder\_loop: LOOP**

**FETCH cur INTO cust\_id, due\_dt;**

**IF done THEN**

**LEAVE reminder\_loop;**

**END IF;**

**-- For now, we print the message (in real apps you'd send SMS/email)**

**SELECT CONCAT('Reminder: Loan for customer ID ', cust\_id, ' is due on ', due\_dt) AS Reminder;**

**END LOOP;**

**CLOSE cur;**

**END //**

**DELIMITER ;**

**CALL SendLoanReminders();**

**Select\*from loans;**

**b)OUTPUT**

