**WEEK 2 – PL/SQL PROGRAMMING**

**Exercise – 1: Control Structures**

1. Creating Customers and Loans table

CREATE TABLE CUSTOMERS (

CustID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

Age NUMBER,

Balance NUMBER(10,2),

IsVIP VARCHAR2(5)

);

CREATE TABLE LOANS (

LoanID NUMBER PRIMARY KEY,

CustID NUMBER,

InterestRate NUMBER(5,2),

DueDate DATE,

FOREIGN KEY (CustID) REFERENCES CUSTOMERS(CustID)

);

1. Inserting data(here I have added the sample data..actual data differs)

INSERT INTO CUSTOMERS VALUES (1, 'Alice', 65, 12000, 'FALSE');

INSERT INTO CUSTOMERS VALUES (2, 'Bob', 45, 8000, 'FALSE');

INSERT INTO CUSTOMERS VALUES (3, 'Charlie', 70, 9500, 'FALSE');

INSERT INTO CUSTOMERS VALUES (4, 'Diana', 30, 15000, 'FALSE');

INSERT INTO LOANS VALUES (101, 1, 10.5, SYSDATE + 10);

INSERT INTO LOANS VALUES (102, 2, 11.0, SYSDATE + 40);

INSERT INTO LOANS VALUES (103, 3, 9.5, SYSDATE + 20);

INSERT INTO LOANS VALUES (104, 4, 8.0, SYSDATE + 5);

COMMIT;

**Scenario 1:** The bank wants to apply a discount to loan interest rates for customers above 60 years old.

**Question:** Write a PL/SQL block that loops through all customers, checks their age, and if they are above 60, apply a 1% discount to their current loan interest rates.

BEGIN

  FOR rec IN (

    SELECT l.LoanID, l.InterestRate

    FROM LOANS l

    JOIN CUSTOMERS c ON l.CustID = c.CustID

    WHERE c.Age > 60

  )

  LOOP

    UPDATE LOANS

    SET InterestRate = rec.InterestRate - 1

    WHERE LoanID = rec.LoanID;

  END LOOP;

  COMMIT;

END;

/

**Scenario 2:** A customer can be promoted to VIP status based on their balance.

**Question:** Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over $10,000.

BEGIN

  FOR rec IN (

    SELECT CustID FROM CUSTOMERS WHERE Balance > 10000

  )

  LOOP

    UPDATE CUSTOMERS

    SET IsVIP = 'TRUE'

    WHERE CustID = rec.CustID;

  END LOOP;

  COMMIT;

END;

/

**Scenario 3:** The bank wants to send reminders to customers whose loans are due within the next 30 days.

**Question:** Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.

SET SERVEROUTPUT ON;

BEGIN

  FOR rec IN (

    SELECT c.Name, l.DueDate

    FROM LOANS l

    JOIN CUSTOMERS c ON l.CustID = c.CustID

    WHERE l.DueDate BETWEEN SYSDATE AND SYSDATE + 30

  )

  LOOP

    DBMS\_OUTPUT.PUT\_LINE('Reminder: Dear ' || rec.Name ||

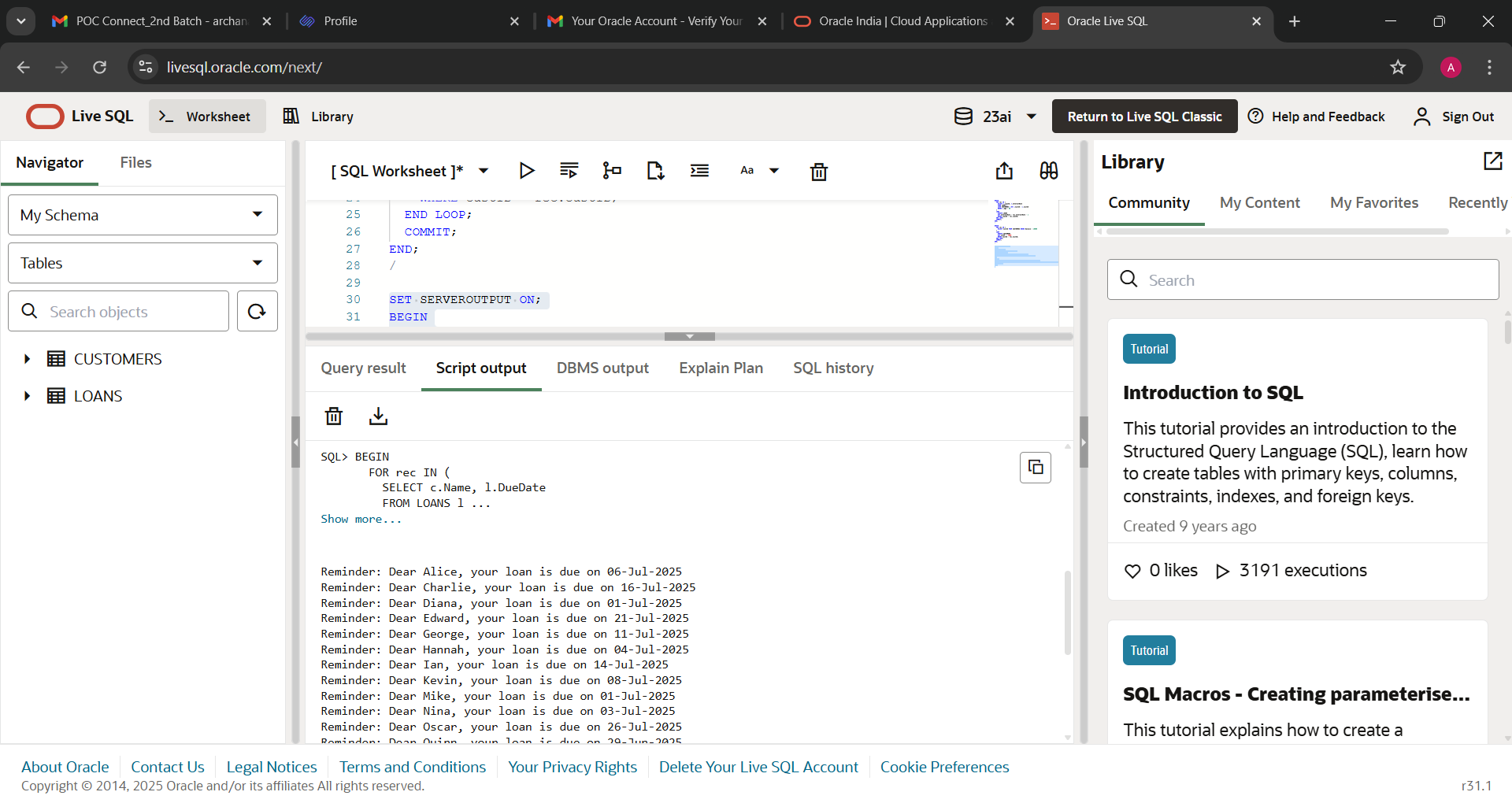
                         ', your loan is due on ' || TO\_CHAR(rec.DueDate, 'DD-Mon-YYYY'));

  END LOOP;

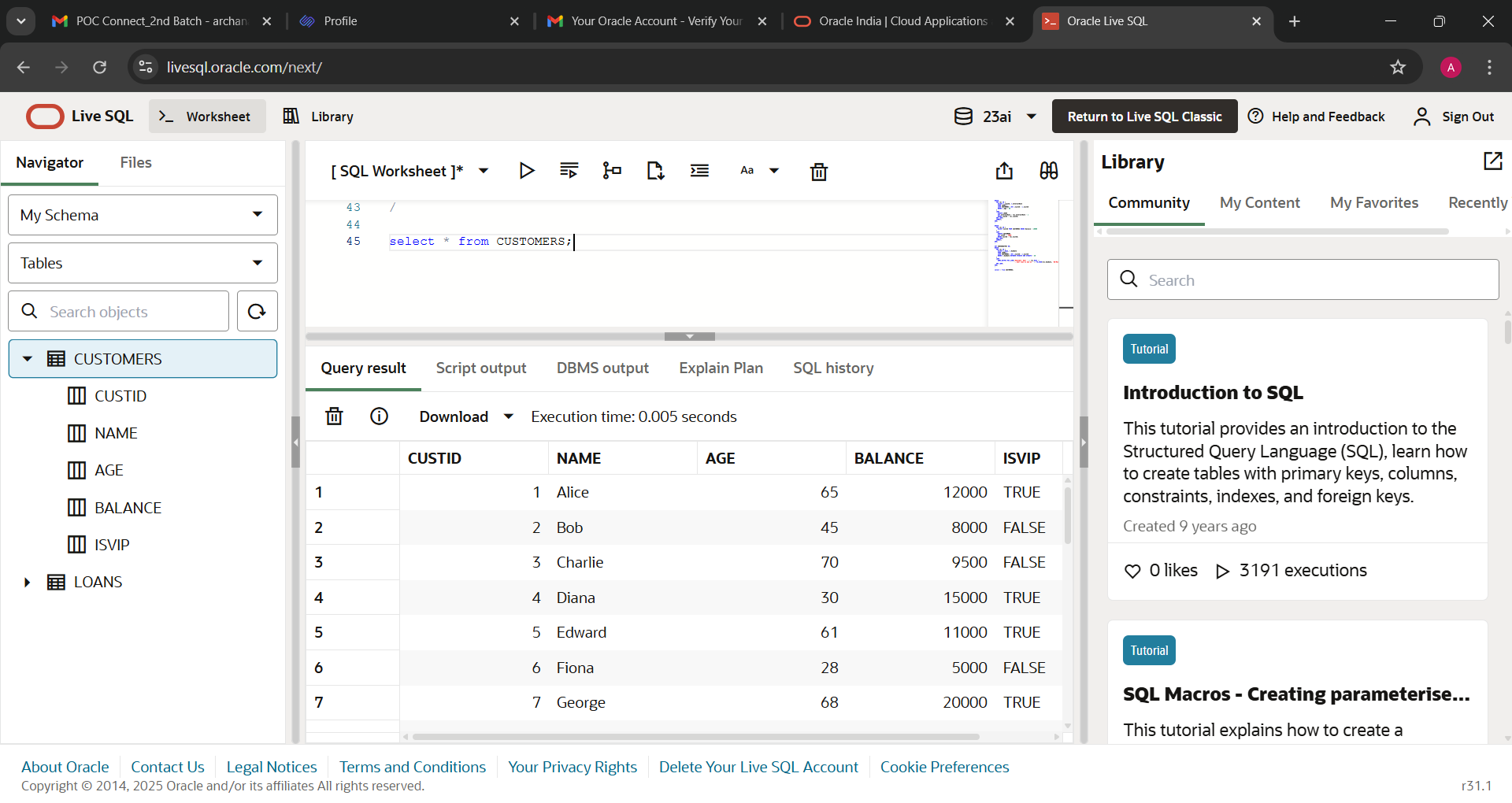
END;

/

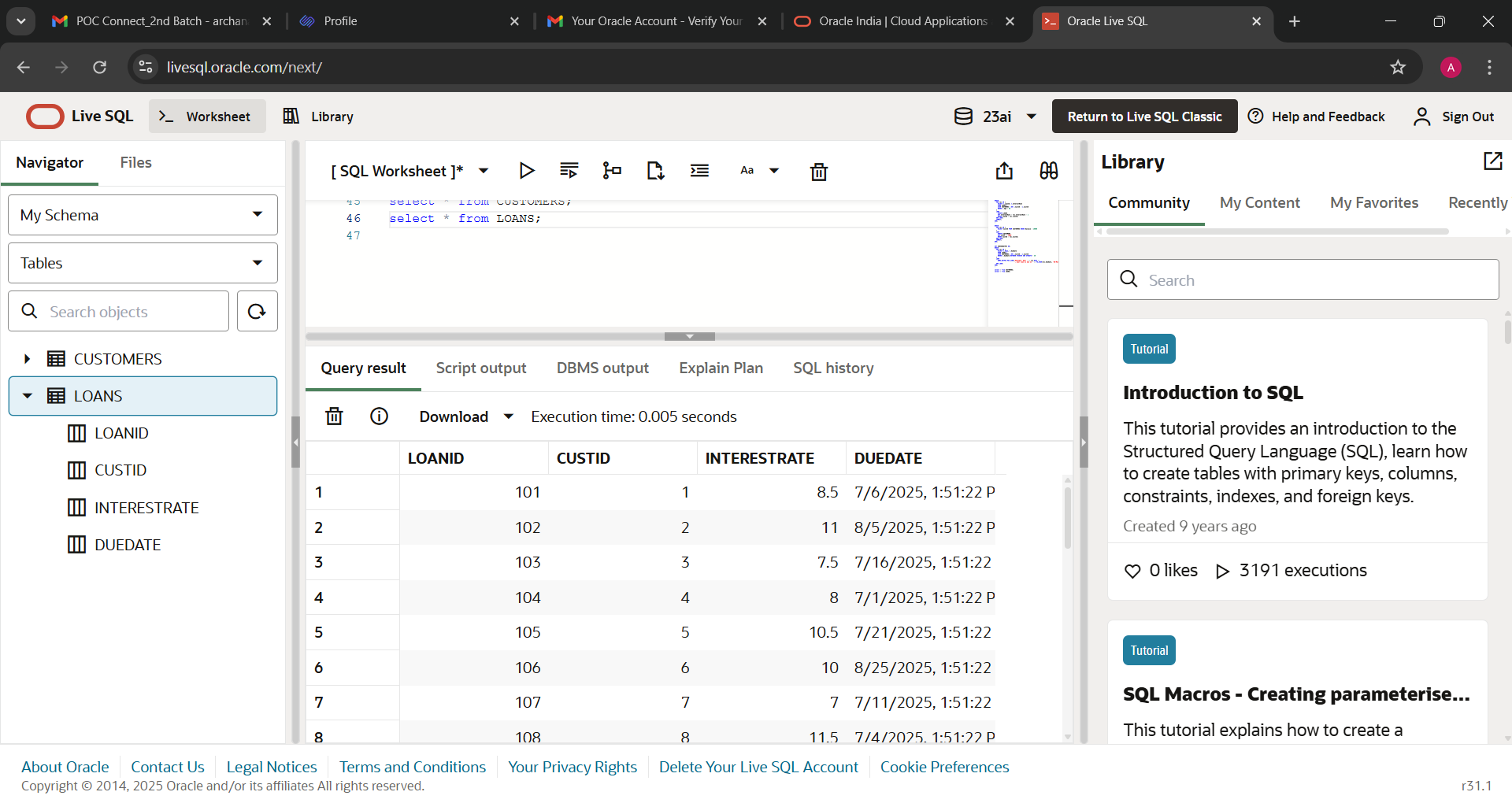
**Output:**



CUSTOMERS TABLE:



LOANS TABLE:



**EXERCISE 3 – STORED PROCEDURES**

1.Creating tables

CREATE TABLE SAVINGS\_ACCOUNTS (

  AccountID   NUMBER PRIMARY KEY,

  CustID      NUMBER,

  Balance     NUMBER(12,2)

);

CREATE TABLE EMPLOYEES (

  EmpID       NUMBER PRIMARY KEY,

  Name        VARCHAR2(100),

  Department  VARCHAR2(50),

  Salary      NUMBER(10,2)

);

CREATE TABLE ACCOUNTS (

  AccountID   NUMBER PRIMARY KEY,

  CustID      NUMBER,

  Balance     NUMBER(12,2)

);

2.Insert data into each table

INSERT INTO SAVINGS\_ACCOUNTS VALUES (101, 1, 10000);

INSERT INTO SAVINGS\_ACCOUNTS VALUES (102, 2, 15000);

INSERT INTO SAVINGS\_ACCOUNTS VALUES (103, 3, 8000);

INSERT INTO SAVINGS\_ACCOUNTS VALUES (104, 4, 20000);

INSERT INTO SAVINGS\_ACCOUNTS VALUES (105, 5, 9500);

INSERT INTO EMPLOYEES VALUES (1, 'Alice', 'IT', 50000);

INSERT INTO EMPLOYEES VALUES (2, 'Bob', 'Finance', 60000);

INSERT INTO EMPLOYEES VALUES (3, 'Charlie', 'IT', 55000);

INSERT INTO EMPLOYEES VALUES (4, 'Diana', 'HR', 48000);

INSERT INTO EMPLOYEES VALUES (5, 'Ethan', 'Finance', 62000);

INSERT INTO ACCOUNTS VALUES (201, 1, 20000);

INSERT INTO ACCOUNTS VALUES (202, 2, 15000);

INSERT INTO ACCOUNTS VALUES (203, 3, 12000);

INSERT INTO ACCOUNTS VALUES (204, 4, 18000);

INSERT INTO ACCOUNTS VALUES (205, 5, 9000);

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

**Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

  UPDATE SAVINGS\_ACCOUNTS

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

  COMMIT;

END;

/

**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

**Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

  p\_department IN VARCHAR2,

  p\_bonus\_pct  IN NUMBER   -- e.g., 0.10 for 10%

) IS

BEGIN

  UPDATE EMPLOYEES

  SET Salary = Salary + (Salary \* p\_bonus\_pct)

  WHERE Department = p\_department;

  COMMIT;

END;

/

**Scenario 3:** Customers should be able to transfer funds between their accounts.

**Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

CREATE OR REPLACE PROCEDURE TransferFunds(

  p\_from\_account IN NUMBER,

  p\_to\_account   IN NUMBER,

  p\_amount       IN NUMBER

) IS

  v\_balance NUMBER;

BEGIN

  -- Get current balance of source account

  SELECT Balance INTO v\_balance

  FROM ACCOUNTS

  WHERE AccountID = p\_from\_account;

  -- Check sufficient balance

  IF v\_balance < p\_amount THEN

    RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance in source account');

  ELSE

    -- Debit source account

    UPDATE ACCOUNTS

    SET Balance = Balance - p\_amount

    WHERE AccountID = p\_from\_account;

    -- Credit destination account

    UPDATE ACCOUNTS

    SET Balance = Balance + p\_amount

    WHERE AccountID = p\_to\_account;

    COMMIT;

  END IF;

END;

/

QUERIES TO EXECUTE:

-- Run monthly interest

EXEC ProcessMonthlyInterest;

-- Give 10% bonus to IT department

EXEC UpdateEmployeeBonus('IT', 0.10);

-- Transfer Rs. 500 from account 101 to account 202

EXEC TransferFunds(101, 202, 500);

OUTPUT:

