**Exercise 3: Stored Procedures**

**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.**

**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.**

**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.**

**Code**

CREATE TABLE accounts (

  account\_id     NUMBER PRIMARY KEY,

  customer\_id    NUMBER,

  account\_type   VARCHAR2(20),

  balance        NUMBER(10, 2)

);

-- Create employees table

CREATE TABLE employees (

  employee\_id       NUMBER PRIMARY KEY,

  employee\_name     VARCHAR2(50),

  department\_id     NUMBER,

  salary            NUMBER(10, 2),

  performance\_rating NUMBER(5, 2)

);

-- Create transactions table (for fund transfers)

CREATE TABLE transactions (

  transaction\_id   NUMBER GENERATED ALWAYS AS IDENTITY,

  from\_account     NUMBER,

  to\_account       NUMBER,

  amount           NUMBER(10,2),

  transaction\_date DATE DEFAULT SYSDATE

);

-- Insert into accounts

INSERT INTO accounts VALUES (101, 1, 'savings', 5000.00);

INSERT INTO accounts VALUES (102, 2, 'checking', 3000.00);

INSERT INTO accounts VALUES (103, 3, 'savings', 8000.00);

INSERT INTO accounts VALUES (104, 4, 'savings', 12000.00);

-- Insert into employees

INSERT INTO employees VALUES (1, 'John Smith', 10, 50000.00, 4.5);

INSERT INTO employees VALUES (2, 'Alice Brown', 10, 60000.00, 4.0);

INSERT INTO employees VALUES (3, 'Bob Johnson', 20, 55000.00, 3.8);

COMMIT;

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

  FOR acc IN (

    SELECT account\_id, balance

    FROM accounts

    WHERE account\_type = 'savings'

  ) LOOP

    UPDATE accounts

    SET balance = balance + (acc.balance \* 0.01)

    WHERE account\_id = acc.account\_id;

    DBMS\_OUTPUT.PUT\_LINE('✅ Interest added to Account ID: ' || acc.account\_id ||

                         ' | New Balance: ' || TO\_CHAR(acc.balance \* 1.01, '99999.99'));

  END LOOP;

END;

/

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

  p\_dept\_id IN NUMBER,

  p\_bonus\_percent IN NUMBER

) IS

BEGIN

  FOR emp IN (

    SELECT employee\_id, salary

    FROM employees

    WHERE department\_id = p\_dept\_id

  ) LOOP

    UPDATE employees

    SET salary = salary + (emp.salary \* (p\_bonus\_percent / 100))

    WHERE employee\_id = emp.employee\_id;

    DBMS\_OUTPUT.PUT\_LINE('🎉 Bonus applied to Employee ID: ' || emp.employee\_id ||

                         ' | New Salary: ' || TO\_CHAR(emp.salary \* (1 + p\_bonus\_percent / 100), '99999.99'));

  END LOOP;

END;

/

CREATE OR REPLACE PROCEDURE TransferFunds(

  p\_from\_account IN NUMBER,

  p\_to\_account IN NUMBER,

  p\_amount IN NUMBER

) IS

  v\_balance NUMBER;

BEGIN

  SELECT balance INTO v\_balance FROM accounts WHERE account\_id = p\_from\_account FOR UPDATE;

  IF v\_balance < p\_amount THEN

    DBMS\_OUTPUT.PUT\_LINE('❌ Transfer failed: Insufficient balance.');

  ELSE

    UPDATE accounts SET balance = balance - p\_amount WHERE account\_id = p\_from\_account;

    UPDATE accounts SET balance = balance + p\_amount WHERE account\_id = p\_to\_account;

    INSERT INTO transactions(from\_account, to\_account, amount)

    VALUES (p\_from\_account, p\_to\_account, p\_amount);

    DBMS\_OUTPUT.PUT\_LINE('💸 Transfer successful: ₹' || p\_amount ||

                         ' from Account ' || p\_from\_account ||

                         ' to Account ' || p\_to\_account);

  END IF;

EXCEPTION

  WHEN NO\_DATA\_FOUND THEN

    DBMS\_OUTPUT.PUT\_LINE('❌ Account not found.');

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('❌ Error: ' || SQLERRM);

END;

/

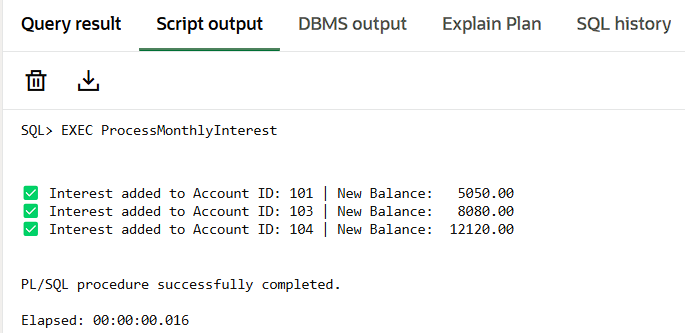
EXEC ProcessMonthlyInterest;

EXEC UpdateEmployeeBonus(10, 10);

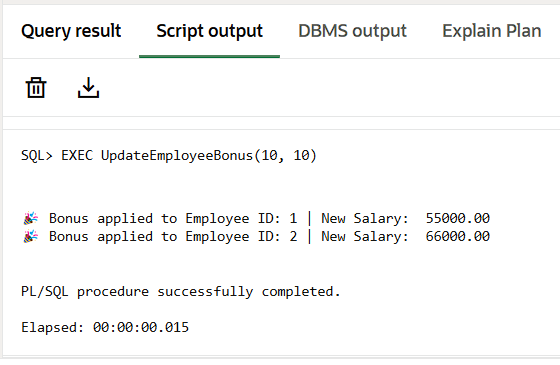
EXEC TransferFunds(101, 102, 1000);

**Output:**

**Scenario 1:**

****

**Scenario 2:**

****

**Scenario 3:**

