**EXERCISE 3**-**STORED PROCEDURES**

**SCENARIO 1:** **ProcessMonthlyInterest**

**CREATING PROCEDURE:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest

IS

BEGIN

    UPDATE Accounts

    SET Balance = Balance + (Balance \* 0.01)

    WHERE AccountType = 'Savings';

      COMMIT;

      DBMS\_OUTPUT.PUT\_LINE('Monthly interest processed successfully.');

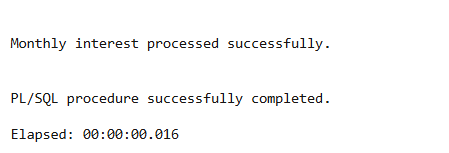
END;

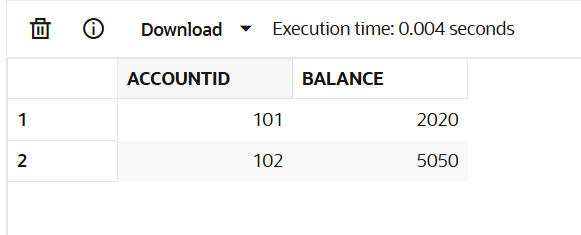
/

EXEC ProcessMonthlyInterest;

SELECT AccountID, Balance FROM Accounts;

**OUTPUT:**

****

****

**SCENARIO 2:** **UpdateEmployeeBonus**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

    p\_department VARCHAR2,

    p\_bonus\_percent NUMBER

)

IS

BEGIN

    UPDATE Employees

    SET Salary = Salary + (Salary \* p\_bonus\_percent / 100)

    WHERE Department = p\_department;

      COMMIT;

    DBMS\_OUTPUT.PUT\_LINE('Bonus applied successfully to ' || p\_department || ' department.');

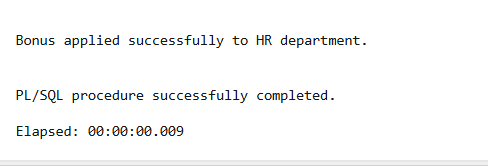
END;

/

SET SERVEROUTPUT ON;

EXEC UpdateEmployeeBonus('HR', 5);

**OUTPUT:**

****

**SCENARIO 3:** **TransferFunds**

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_from\_acc NUMBER,

p\_to\_acc NUMBER,

p\_amount NUMBER

)

IS

v\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = p\_from\_acc;

IF v\_balance < p\_amount THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Insufficient funds.');

RETURN;

END IF;

UPDATE Accounts

SET Balance = Balance - p\_amount

WHERE AccountID = p\_from\_acc;

UPDATE Accounts

SET Balance = Balance + p\_amount

WHERE AccountID = p\_to\_acc;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Transfer completed successfully.');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

ROLLBACK;

WHEN OTHERS THEN

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

ROLLBACK;

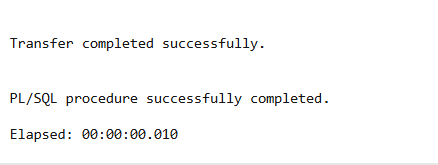
END;

/

SET SERVEROUTPUT ON;

EXEC TransferFunds(101, 102, 1000);

**OUTPUT:**

****

