**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:**

**Scenario 1:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

FOR acc IN (

SELECT account\_id, balance

FROM accounts

WHERE UPPER(account\_type) = 'SAVINGS'

) LOOP

UPDATE accounts

SET balance = balance + (balance \* 0.01)

WHERE account\_id = acc.account\_id;

DBMS\_OUTPUT.PUT\_LINE('Interest applied to account: ' || acc.account\_id);

END LOOP;

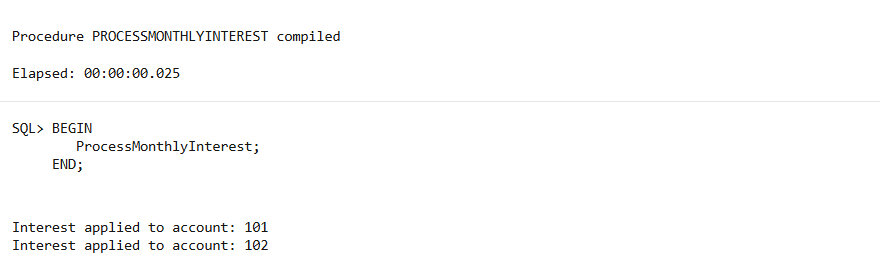
END;

BEGIN

ProcessMonthlyInterest;

END;

**Output:**



**Scenario 2:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

p\_dept\_id IN NUMBER,

p\_bonus\_pct IN NUMBER

) IS

BEGIN

FOR emp IN (

SELECT emp\_id, salary

FROM employees

WHERE department\_id = p\_dept\_id

) LOOP

UPDATE employees

SET salary = salary + (salary \* p\_bonus\_pct / 100)

WHERE emp\_id = emp.emp\_id;

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

END LOOP;

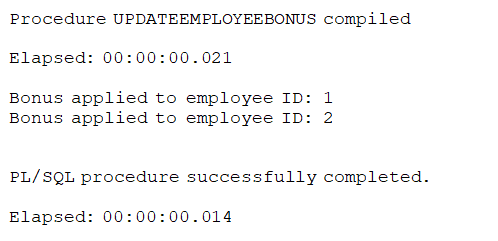
END;

BEGIN

UpdateEmployeeBonus(10, 10); -- 10% bonus for Dept 10

END;

**Output:**



**Scenario 3:**

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_from\_acc IN NUMBER,

p\_to\_acc IN NUMBER,

p\_amount IN NUMBER

) IS

v\_balance NUMBER;

BEGIN

-- Check source account balance

SELECT balance INTO v\_balance

FROM accounts

WHERE account\_id = p\_from\_acc;

IF v\_balance < p\_amount THEN

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

ELSE

UPDATE accounts

SET balance = balance - p\_amount

WHERE account\_id = p\_from\_acc;

UPDATE accounts

SET balance = balance + p\_amount

WHERE account\_id = p\_to\_acc;

DBMS\_OUTPUT.PUT\_LINE('Transferred ₹' || p\_amount ||

' from ' || p\_from\_acc ||

' to ' || p\_to\_acc);

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

WHEN OTHERS THEN

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

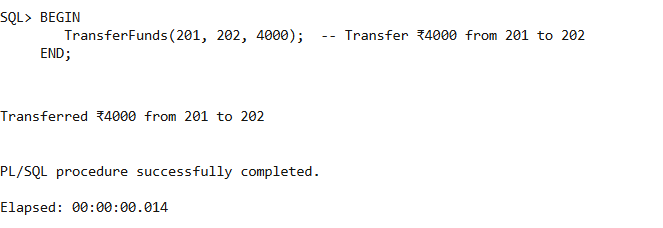
END;

BEGIN

TransferFunds(201, 202, 4000); -- Transfer ₹4000 from 201 to 202

END;

**Output:**



**Table creation:**

CREATE TABLE accounts (

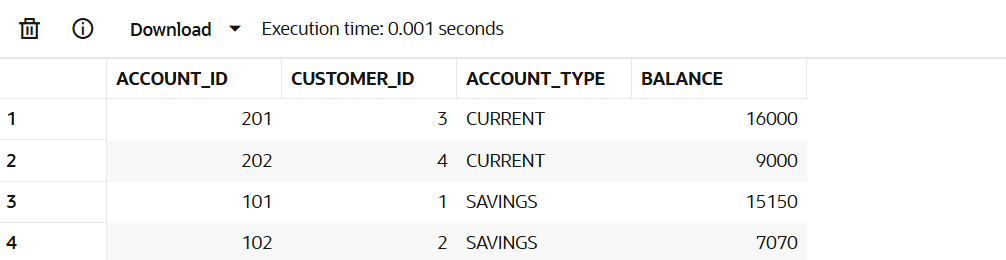
account\_id NUMBER PRIMARY KEY,

customer\_id NUMBER,

account\_type VARCHAR2(20), -- e.g., 'SAVINGS', 'CURRENT'

balance NUMBER(12,2)

);



CREATE TABLE employees (

emp\_id NUMBER PRIMARY KEY,

name VARCHAR2(100),

department\_id NUMBER,

salary NUMBER(10,2)

);

