**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.**

**CODE:**

CREATE TABLE accounts (

accountid NUMBER PRIMARY KEY,

balance NUMBER(10,2),

acctype VARCHAR2(20)

);

INSERT INTO accounts VALUES (101, 800, 'savings');

INSERT INTO accounts VALUES (102, 700, 'savings');

INSERT INTO accounts VALUES (103, 1200, 'current');

COMMIT;

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

UPDATE accounts

SET balance = balance + (balance \* 0.01)

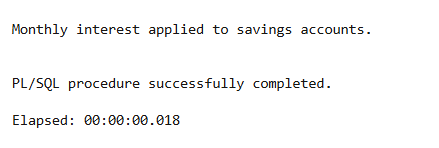
WHERE acctype = 'savings';

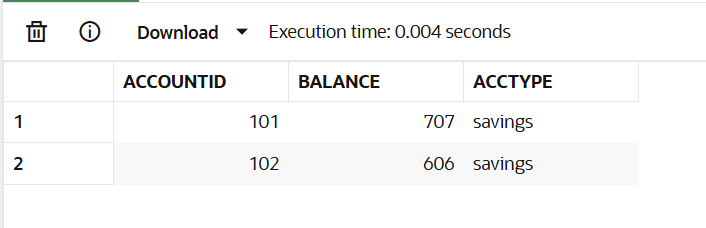
DBMS\_OUTPUT.PUT\_LINE('Monthly interest applied to savings accounts.');

END;

/

**OUTPUT:**

****



**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.**

**CODE:**

CREATE TABLE employees (

empid NUMBER PRIMARY KEY,

name VARCHAR2(100),

department VARCHAR2(50),

salary NUMBER(10,2)

);

INSERT INTO employees VALUES (1, 'Meena', 'HR', 70000);

INSERT INTO employees VALUES (2, 'Sri', 'IT', 80000);

INSERT INTO employees VALUES (3, 'Harini', 'IT', 75000);

COMMIT;

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

dept\_name IN VARCHAR2,

bonus\_pct IN NUMBER

) AS

BEGIN

UPDATE employees

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

WHERE department = dept\_name;

DBMS\_OUTPUT.PUT\_LINE('Bonus applied to department: ' || dept\_name);

END;

/

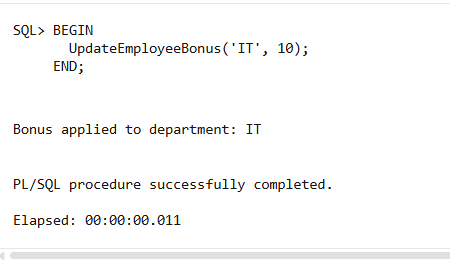
BEGIN

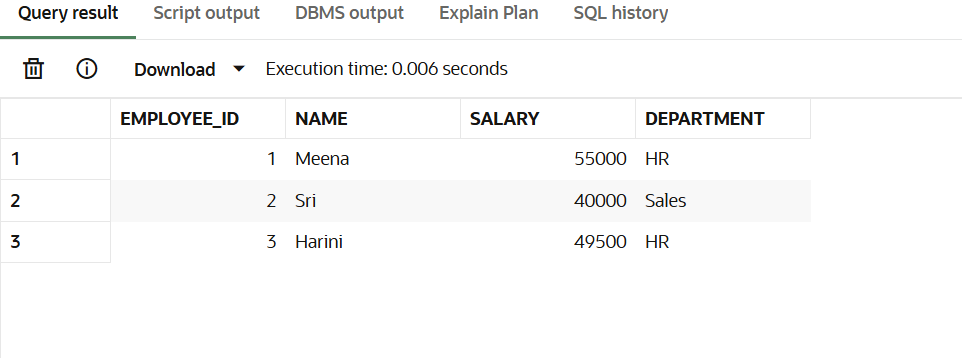
UpdateEmployeeBonus('IT', 10);

END;

/

**OUTPUT:**

****

****

**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 OR REPLACE PROCEDURE TransferFunds (

from\_acc IN NUMBER,

to\_acc IN NUMBER,

amount IN NUMBER

) AS

bal NUMBER;

BEGIN

SELECT balance INTO bal FROM accounts WHERE accountid = from\_acc;

IF bal < amount THEN

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance in source account.');

RETURN;

END IF;

UPDATE accounts SET balance = balance - amount WHERE accountid = from\_acc;

UPDATE accounts SET balance = balance + amount WHERE accountid = to\_acc;

DBMS\_OUTPUT.PUT\_LINE('Transfer successful.');

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

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

WHEN OTHERS THEN

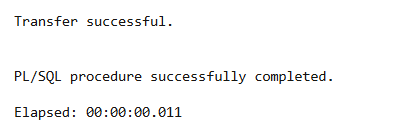
DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

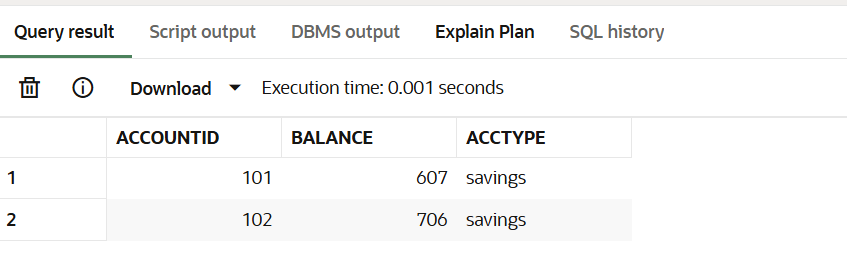
ROLLBACK;

END;

/

**OUTPUT:**

****

****