PL/SQL

# Excercise1:

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

BEGIN

FOR cust IN (

SELECT c.CustomerID, l.LoanID, l.InterestRate,

TRUNC(MONTHS\_BETWEEN(SYSDATE, c.DOB)/12) AS Age

FROM Customers c

JOIN Loans l ON c.CustomerID = l.CustomerID

)

LOOP

IF cust.Age > 60 THEN

UPDATE Loans

SET InterestRate = InterestRate - 1

WHERE LoanID = cust.LoanID;

END IF;

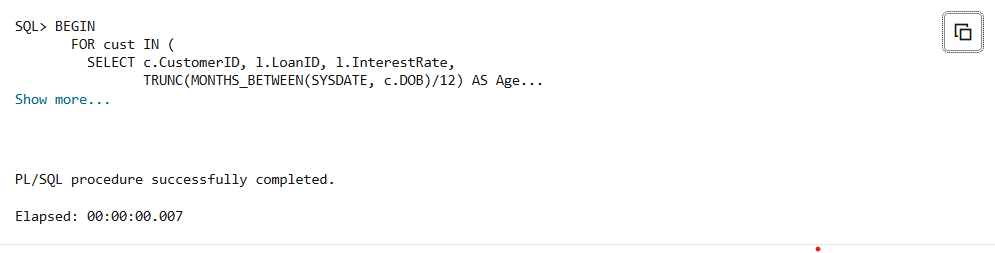
END LOOP;

COMMIT;

END;

/

**Output:**

****

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

BEGIN

FOR cust IN (

SELECT CustomerID, Balance

FROM Customers

WHERE Balance > 10000

)

LOOP

UPDATE Customers

SET IsVIP = 'TRUE'

WHERE CustomerID = cust.CustomerID;

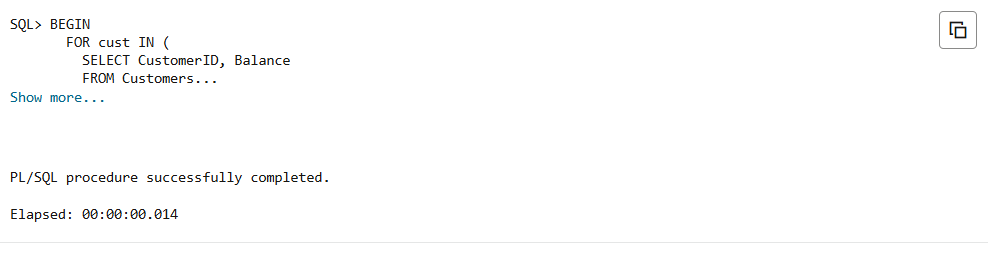
END LOOP;

COMMIT;

END;

/

**Output:**



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

BEGIN

FOR loan\_rec IN (

SELECT CustomerID, EndDate

FROM Loans

WHERE EndDate BETWEEN SYSDATE AND SYSDATE + 30

)

LOOP

DBMS\_OUTPUT.PUT\_LINE(

'Reminder: Loan due on ' || TO\_CHAR(loan\_rec.EndDate, 'DD-MON-YYYY') ||

' for Customer ID: ' || loan\_rec.CustomerID

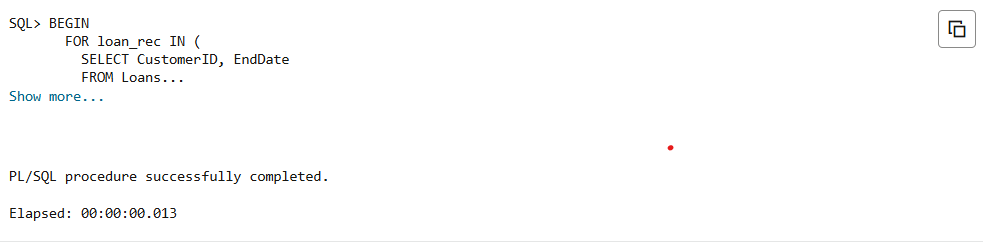
);

END LOOP;

END;

/

**Output:**



# Excercise3:

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

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

FOR acc IN (

SELECT AccountID, Balance

FROM Accounts

WHERE AccountType = 'Savings'

) LOOP

UPDATE Accounts

SET Balance = Balance + (acc.Balance \* 0.01),

LastModified = SYSDATE

WHERE AccountID = acc.AccountID;

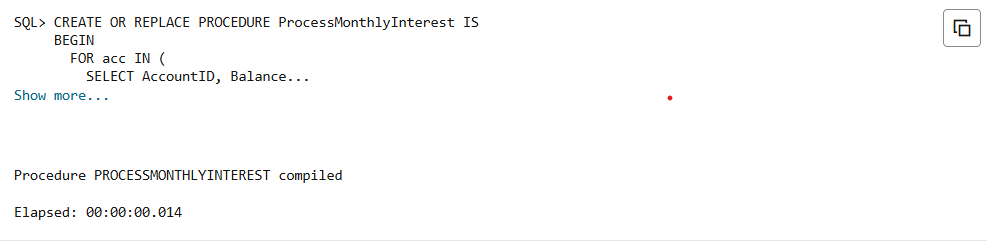
END LOOP;

COMMIT;

END;

/

**Output:**

****

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

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

p\_department IN Employees.Department%TYPE,

p\_bonus\_percent IN NUMBER

) IS

BEGIN

UPDATE Employees

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

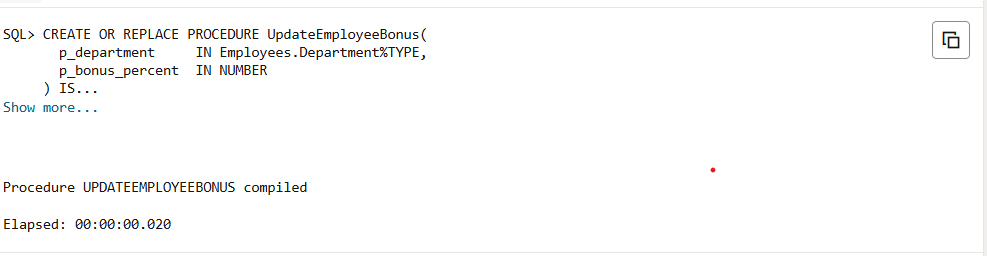
WHERE Department = p\_department;

COMMIT;

END;

/

**Output:**

****

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

CREATE OR REPLACE PROCEDURE TransferFunds(

  p\_from\_account   IN Accounts.AccountID%TYPE,

  p\_to\_account     IN Accounts.AccountID%TYPE,

  p\_amount         IN NUMBER

) IS

  v\_balance NUMBER;

BEGIN

  SELECT Balance INTO v\_balance

  FROM Accounts

  WHERE AccountID = p\_from\_account

  FOR UPDATE;

  IF v\_balance < p\_amount THEN

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

  END IF;

  UPDATE Accounts

  SET Balance = Balance - p\_amount,

      LastModified = SYSDATE

  WHERE AccountID = p\_from\_account;

  UPDATE Accounts

  SET Balance = Balance + p\_amount,

      LastModified = SYSDATE

  WHERE AccountID = p\_to\_account;

  COMMIT;

END;

/

**Output:**

