**Exercise 6: Cursors**

**Scenario 1:** Generate monthly statements for all customers.

**Question:** Write a PL/SQL block using an explicit cursor **GenerateMonthlyStatements** that retrieves all transactions for the current month and prints a statement for each customer.

**CODE**

SET SERVEROUTPUT ON;

DECLARE

    CURSOR c\_transactions IS

        SELECT DISTINCT c.CustomerID, c.Name, t.TransactionDate, t.Amount, t.TransactionType

        FROM Customers c

        JOIN Accounts a ON c.CustomerID = a.CustomerID

        JOIN Transactions t ON a.AccountID = t.AccountID

        WHERE EXTRACT(MONTH FROM t.TransactionDate) = EXTRACT(MONTH FROM SYSDATE)

          AND EXTRACT(YEAR FROM t.TransactionDate) = EXTRACT(YEAR FROM SYSDATE);

    v\_customerID Customers.CustomerID%TYPE;

    v\_name Customers.Name%TYPE;

    v\_transactionDate Transactions.TransactionDate%TYPE;

    v\_amount Transactions.Amount%TYPE;

    v\_transactionType Transactions.TransactionType%TYPE;

BEGIN

    OPEN c\_transactions;

    LOOP

        FETCH c\_transactions INTO v\_customerID, v\_name, v\_transactionDate, v\_amount, v\_transactionType;

        EXIT WHEN c\_transactions%NOTFOUND;

        -- Print the statement (for demonstration purposes, using DBMS\_OUTPUT)

        DBMS\_OUTPUT.PUT\_LINE('CustomerID: ' || v\_customerID);

        DBMS\_OUTPUT.PUT\_LINE('Name: ' || v\_name);

        DBMS\_OUTPUT.PUT\_LINE('Transaction Date: ' || v\_transactionDate);

        DBMS\_OUTPUT.PUT\_LINE('Amount: ' || v\_amount);

        DBMS\_OUTPUT.PUT\_LINE('Transaction Type: ' || v\_transactionType);

        DBMS\_OUTPUT.PUT\_LINE('--------------------------');

    END LOOP;

    CLOSE c\_transactions;

END;

/

**Scenario 2:** Apply annual fee to all accounts.

**Question:** Write a PL/SQL block using an explicit cursor **ApplyAnnualFee** that deducts an annual maintenance fee from the balance of all accounts.

**CODE**

DECLARE

    CURSOR c\_accounts IS

        SELECT AccountID, Balance

        FROM Accounts;

    v\_accountID Accounts.AccountID%TYPE;

    v\_balance Accounts.Balance%TYPE;

    v\_annual\_fee NUMBER := 50; -- Example annual fee

BEGIN

    OPEN c\_accounts;

    LOOP

        FETCH c\_accounts INTO v\_accountID, v\_balance;

        EXIT WHEN c\_accounts%NOTFOUND;

        -- Deduct the annual fee

        UPDATE Accounts

        SET Balance = v\_balance - v\_annual\_fee

        WHERE AccountID = v\_accountID;

        -- Optionally, log the update

        DBMS\_OUTPUT.PUT\_LINE('Applied fee to AccountID: ' || v\_accountID);

    END LOOP;

    CLOSE c\_accounts;

END;

/

**Scenario 3:** Update the interest rate for all loans based on a new policy.

**Question:** Write a PL/SQL block using an explicit cursor **UpdateLoanInterestRates** that fetches all loans and updates their interest rates based on the new policy.

**CODE**

DECLARE

    CURSOR c\_loans IS

        SELECT LoanID, InterestRate

        FROM Loans;

    v\_loanID Loans.LoanID%TYPE;

    v\_currentInterestRate Loans.InterestRate%TYPE;

    v\_newInterestRate NUMBER;

    -- Example function to calculate new interest rate

    FUNCTION calculate\_new\_interest\_rate(p\_current\_rate NUMBER) RETURN NUMBER IS

    BEGIN

        RETURN p\_current\_rate + 0.01; -- Example: increase interest rate by 1%

    END;

BEGIN

    OPEN c\_loans;

    LOOP

        FETCH c\_loans INTO v\_loanID, v\_currentInterestRate;

        EXIT WHEN c\_loans%NOTFOUND;

        v\_newInterestRate := calculate\_new\_interest\_rate(v\_currentInterestRate);

        -- Update the interest rate

        UPDATE Loans

        SET InterestRate = v\_newInterestRate

        WHERE LoanID = v\_loanID;

        -- Optionally, log the update

        DBMS\_OUTPUT.PUT\_LINE('Updated LoanID: ' || v\_loanID || ' to new interest rate: ' || v\_newInterestRate);

    END LOOP;

    CLOSE c\_loans;

END;

/