**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.

DECLARE

-- Cursor to retrieve all transactions for the current month

CURSOR GenerateMonthlyStatements IS

SELECT t.TransactionID, t.AccountID, t.TransactionDate, t.Amount, t.TransactionType, a.CustomerID

FROM Transactions t

JOIN Accounts a ON t.AccountID = a.AccountID

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

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

-- Record type for cursor

r\_transaction GenerateMonthlyStatements%ROWTYPE;

-- Variables to hold customer data

v\_customer\_name Customers.Name%TYPE;

v\_balance NUMBER;

BEGIN

-- Open the cursor

OPEN GenerateMonthlyStatements;

-- Loop through the cursor

LOOP

FETCH GenerateMonthlyStatements INTO r\_transaction;

EXIT WHEN GenerateMonthlyStatements%NOTFOUND;

-- Retrieve customer name and balance

SELECT Name, Balance INTO v\_customer\_name, v\_balance

FROM Customers

WHERE CustomerID = r\_transaction.CustomerID;

-- Print the monthly statement

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

DBMS\_OUTPUT.PUT\_LINE('Account ID: ' || r\_transaction.AccountID);

DBMS\_OUTPUT.PUT\_LINE('Transaction Date: ' || r\_transaction.TransactionDate);

DBMS\_OUTPUT.PUT\_LINE('Amount: ' || r\_transaction.Amount);

DBMS\_OUTPUT.PUT\_LINE('Transaction Type: ' || r\_transaction.TransactionType);

DBMS\_OUTPUT.PUT\_LINE('Balance: ' || v\_balance);

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

END LOOP;

-- Close the cursor

CLOSE GenerateMonthlyStatements;

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.

DECLARE

-- Cursor to retrieve all accounts

CURSOR ApplyAnnualFee IS

SELECT AccountID, Balance

FROM Accounts;

-- Record type for cursor

r\_account ApplyAnnualFee%ROWTYPE;

-- Annual fee amount

annual\_fee NUMBER := 50; -- Example fee amount, adjust as needed

BEGIN

-- Open the cursor

OPEN ApplyAnnualFee;

-- Loop through the cursor

LOOP

FETCH ApplyAnnualFee INTO r\_account;

EXIT WHEN ApplyAnnualFee%NOTFOUND;

-- Deduct the annual fee from the account balance

UPDATE Accounts

SET Balance = Balance - annual\_fee

WHERE AccountID = r\_account.AccountID;

END LOOP;

-- Close the cursor

CLOSE ApplyAnnualFee;

-- Commit the changes

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Annual fee applied to all 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.

DECLARE

-- Cursor to retrieve all loans

CURSOR UpdateLoanInterestRates IS

SELECT LoanID, InterestRate

FROM Loans;

-- Record type for cursor

r\_loan UpdateLoanInterestRates%ROWTYPE;

-- New interest rate to apply

new\_interest\_rate NUMBER := 6; -- Example new interest rate, adjust as needed

BEGIN

-- Open the cursor

OPEN UpdateLoanInterestRates;

-- Loop through the cursor

LOOP

FETCH UpdateLoanInterestRates INTO r\_loan;

EXIT WHEN UpdateLoanInterestRates%NOTFOUND;

-- Update the interest rate for the loan

UPDATE Loans

SET InterestRate = new\_interest\_rate

WHERE LoanID = r\_loan.LoanID;

END LOOP;

-- Close the cursor

CLOSE UpdateLoanInterestRates;

-- Commit the changes

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Interest rates updated for all loans.');

END;

/