**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.

**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.

**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.

**Scenario 1:**

DECLARE

CURSOR cust\_cursor IS

SELECT DISTINCT c.CustomerID, c.Name

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);

CURSOR trans\_cursor(p\_customer\_id NUMBER) IS

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

FROM Accounts a

JOIN Transactions t ON a.AccountID = t.AccountID

WHERE a.CustomerID = p\_customer\_id

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

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

ORDER BY t.TransactionDate;

v\_total\_deposits NUMBER := 0;

v\_total\_withdrawals NUMBER := 0;

BEGIN

FOR cust\_rec IN cust\_cursor LOOP

DBMS\_OUTPUT.PUT\_LINE('Monthly Statement for ' || cust\_rec.Name || ' (Customer ID: ' || cust\_rec.CustomerID || ')');

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

v\_total\_deposits := 0;

v\_total\_withdrawals := 0;

FOR trans\_rec IN trans\_cursor(cust\_rec.CustomerID) LOOP

DBMS\_OUTPUT.PUT\_LINE(

'Account: ' || trans\_rec.AccountID ||

', Date: ' || TO\_CHAR(trans\_rec.TransactionDate, 'YYYY-MM-DD') ||

', Amount: ' || trans\_rec.Amount ||

', Type: ' || trans\_rec.TransactionType

);

IF trans\_rec.TransactionType = 'Deposit' THEN

v\_total\_deposits := v\_total\_deposits + trans\_rec.Amount;

ELSIF trans\_rec.TransactionType = 'Withdrawal' THEN

v\_total\_withdrawals := v\_total\_withdrawals + trans\_rec.Amount;

END IF;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Total Deposits: ' || v\_total\_deposits);

DBMS\_OUTPUT.PUT\_LINE('Total Withdrawals: ' || v\_total\_withdrawals);

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

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

END LOOP;

EXCEPTION

WHEN OTHERS THEN

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

END;

/

**Scenario 2:**

DECLARE

CURSOR account\_cursor IS

SELECT AccountID, Balance

FROM Accounts

FOR UPDATE OF Balance;

v\_annual\_fee NUMBER := 50; -- Set the annual fee amount

BEGIN

FOR acc\_rec IN account\_cursor LOOP

IF acc\_rec.Balance >= v\_annual\_fee THEN

UPDATE Accounts

SET Balance = Balance - v\_annual\_fee,

LastModified = SYSDATE

WHERE CURRENT OF account\_cursor;

DBMS\_OUTPUT.PUT\_LINE('Annual fee of ' || v\_annual\_fee || ' applied to Account ID: ' || acc\_rec.AccountID);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance to apply annual fee for Account ID: ' || acc\_rec.AccountID);

END IF;

END LOOP;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

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

END;

/

**Scenario 3:**

DECLARE

CURSOR loan\_cursor IS

SELECT LoanID, LoanAmount, InterestRate

FROM Loans

FOR UPDATE OF InterestRate;

v\_new\_rate NUMBER;

BEGIN

FOR loan\_rec IN loan\_cursor LOOP

-- Define the new policy for interest rates

IF loan\_rec.LoanAmount <= 5000 THEN

v\_new\_rate := 5.5;

ELSIF loan\_rec.LoanAmount <= 10000 THEN

v\_new\_rate := 6.0;

ELSE

v\_new\_rate := 6.5;

END IF;

-- Update the interest rate

UPDATE Loans

SET InterestRate = v\_new\_rate

WHERE CURRENT OF loan\_cursor;

DBMS\_OUTPUT.PUT\_LINE('Updated interest rate for Loan ID: ' || loan\_rec.LoanID ||

' from ' || loan\_rec.InterestRate || '% to ' || v\_new\_rate || '%');

END LOOP;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error updating loan interest rates: ' || SQLERRM);

END;

/