**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.

Ans:

SET SERVEROUTPUT ON;

DECLARE

  CURSOR transaction\_cursor IS

    SELECT

      t.TransactionID,

      t.AccountID,

      t.Amount,

      t.TransactionType,

      t.TransactionDate,

      a.CustomerID,

      c.Name AS CustomerName

    FROM

      Transactions t

    JOIN

      Accounts a ON t.AccountID = a.AccountID

    JOIN

      Customers c ON a.CustomerID = c.CustomerID

    WHERE

      t.TransactionDate >= TRUNC(SYSDATE, 'MM') AND

      t.TransactionDate < ADD\_MONTHS(TRUNC(SYSDATE, 'MM'), 1);

  TYPE transaction\_record\_type IS RECORD (

    TransactionID Transactions.TransactionID%TYPE,

    AccountID Transactions.AccountID%TYPE,

    Amount Transactions.Amount%TYPE,

    TransactionType Transactions.TransactionType%TYPE,

    TransactionDate Transactions.TransactionDate%TYPE,

    CustomerID Accounts.CustomerID%TYPE,

    CustomerName Customers.Name%TYPE

  );

  transaction\_record transaction\_record\_type;

  v\_current\_customer\_id Accounts.CustomerID%TYPE := NULL;

  v\_current\_customer\_name Customers.Name%TYPE;

BEGIN

  OPEN transaction\_cursor;

  LOOP

    FETCH transaction\_cursor INTO transaction\_record;

    EXIT WHEN transaction\_cursor%NOTFOUND;

    IF v\_current\_customer\_id IS NULL OR v\_current\_customer\_id != transaction\_record.CustomerID THEN

      IF v\_current\_customer\_id IS NOT NULL THEN

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

      END IF;

      v\_current\_customer\_id := transaction\_record.CustomerID;

      v\_current\_customer\_name := transaction\_record.CustomerName;

      DBMS\_OUTPUT.PUT\_LINE('Monthly Statement for Customer: ' || v\_current\_customer\_name);

      DBMS\_OUTPUT.PUT\_LINE('Customer ID: ' || v\_current\_customer\_id);

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

      DBMS\_OUTPUT.PUT\_LINE('Transaction ID | Account ID | Amount | Type | Date');

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

    END IF;

    DBMS\_OUTPUT.PUT\_LINE(transaction\_record.TransactionID || ' | ' ||

                         transaction\_record.AccountID || ' | ' ||

                         transaction\_record.Amount || ' | ' ||

                         transaction\_record.TransactionType || ' | ' ||

                         TO\_CHAR(transaction\_record.TransactionDate, 'DD-MON-YYYY'));

  END LOOP;

  CLOSE transaction\_cursor;

  IF v\_current\_customer\_id IS NOT NULL THEN

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

  END IF;

EXCEPTION

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

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.

**Ans:**

DECLARE

  CURSOR account\_cursor IS

    SELECT AccountID, Balance

    FROM Accounts

    FOR UPDATE OF Balance;

  v\_annual\_fee NUMBER := 50;

  v\_account\_id Accounts.AccountID%TYPE;

  v\_balance Accounts.Balance%TYPE;

BEGIN

  OPEN account\_cursor;

  LOOP

    FETCH account\_cursor INTO v\_account\_id, v\_balance;

    EXIT WHEN account\_cursor%NOTFOUND;

    IF v\_balance >= v\_annual\_fee THEN

      UPDATE Accounts

      SET Balance = Balance - v\_annual\_fee

      WHERE CURRENT OF account\_cursor;

      DBMS\_OUTPUT.PUT\_LINE('Account ID: ' || v\_account\_id || ' - Annual fee applied. New balance: ' || (v\_balance - v\_annual\_fee));

    ELSE

      DBMS\_OUTPUT.PUT\_LINE('Account ID: ' || v\_account\_id || ' - Insufficient funds to apply annual fee.');

    END IF;

  END LOOP;

  CLOSE account\_cursor;

EXCEPTION

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

    ROLLBACK;

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.

**Ans:**

DECLARE

  CURSOR loan\_cursor IS

    SELECT LoanID, InterestRate

    FROM Loans

    FOR UPDATE OF InterestRate;

  TYPE loan\_record\_type IS RECORD (

    LoanID Loans.LoanID%TYPE,

    InterestRate Loans.InterestRate%TYPE

  );

  loan\_record loan\_record\_type;

  v\_interest\_rate\_adjustment NUMBER := 0.5;

BEGIN

  OPEN loan\_cursor;

  LOOP

    FETCH loan\_cursor INTO loan\_record;

    EXIT WHEN loan\_cursor%NOTFOUND;

    loan\_record.InterestRate := loan\_record.InterestRate + v\_interest\_rate\_adjustment;

    UPDATE Loans

    SET InterestRate = loan\_record.InterestRate

    WHERE CURRENT OF loan\_cursor;

    DBMS\_OUTPUT.PUT\_LINE('Updated LoanID ' || loan\_record.LoanID || ' to new InterestRate ' || loan\_record.InterestRate);

  END LOOP;

  CLOSE loan\_cursor;

  COMMIT;

EXCEPTION

  WHEN OTHERS THEN

    ROLLBACK;

    DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END;