**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 transaction\_cursor IS

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

FROM Transactions t

JOIN Accounts a ON t.AccountID = a.AccountID

JOIN Customers c ON a.CustomerID = c.CustomerID

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

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

ORDER BY c.CustomerID, t.TransactionDate;

v\_transaction\_id Transactions.TransactionID%TYPE;

v\_account\_id Transactions.AccountID%TYPE;

v\_transaction\_date Transactions.TransactionDate%TYPE;

v\_amount Transactions.Amount%TYPE;

v\_transaction\_type Transactions.TransactionType%TYPE;

v\_customer\_id Accounts.CustomerID%TYPE;

v\_customer\_name Customers.Name%TYPE;

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

BEGIN

OPEN transaction\_cursor;

LOOP

FETCH transaction\_cursor INTO v\_transaction\_id, v\_account\_id, v\_transaction\_date, v\_amount, v\_transaction\_type, v\_customer\_id, v\_customer\_name;

EXIT WHEN transaction\_cursor%NOTFOUND;

IF v\_current\_customer\_id IS NULL OR v\_current\_customer\_id != v\_customer\_id THEN

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

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

v\_current\_customer\_id := v\_customer\_id;

END IF;

DBMS\_OUTPUT.PUT\_LINE('Transaction ID: ' || v\_transaction\_id);

DBMS\_OUTPUT.PUT\_LINE('Account ID: ' || v\_account\_id);

DBMS\_OUTPUT.PUT\_LINE('Date: ' || TO\_CHAR(v\_transaction\_date, 'YYYY-MM-DD'));

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

DBMS\_OUTPUT.PUT\_LINE('Type: ' || v\_transaction\_type);

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

END LOOP;

CLOSE transaction\_cursor;

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 account\_cursor IS

SELECT AccountID, Balance

FROM Accounts;

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;

UPDATE Accounts

SET Balance = Balance - c\_annual\_fee

WHERE AccountID = v\_account\_id;

DBMS\_OUTPUT.PUT\_LINE('Annual Fee ' || c\_annual\_fee || ' Account ID: ' || v\_account\_id);

END LOOP;

COMMIT;

CLOSE account\_cursor;

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 loan\_cursor IS

SELECT LoanID, InterestRate

FROM Loans;

v\_loan\_id Loans.LoanID%TYPE;

v\_current\_interest\_rate Loans.InterestRate%TYPE;

BEGIN

OPEN loan\_cursor;

LOOP

FETCH loan\_cursor INTO v\_loan\_id, v\_current\_interest\_rate;

EXIT WHEN loan\_cursor%NOTFOUND;

UPDATE Loans

SET InterestRate = c\_new\_interest\_rate

WHERE LoanID = v\_loan\_id;

DBMS\_OUTPUT.PUT\_LINE('Updated interest' || c\_new\_interest\_rate || 'Loan ID: ' || v\_loan\_id);

END LOOP;

COMMIT;

CLOSE loan\_cursor;

END;