**EXERCISE 6: CURSORS**

**Scenario 1: GenerateMonthlyStatements - Retrieve Transactions for Current Month**

DECLARE

CURSOR cur\_Transactions IS

SELECT 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)

ORDER BY c.CustomerID, t.TransactionDate;

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 cur\_Transactions;

FETCH cur\_Transactions INTO v\_CustomerID, v\_Name, v\_TransactionDate, v\_Amount, v\_TransactionType;

WHILE cur\_Transactions%FOUND LOOP

DBMS\_OUTPUT.PUT\_LINE('Customer ID: ' || v\_CustomerID || ', Name: ' || v\_Name);

DBMS\_OUTPUT.PUT\_LINE('Transaction Date: ' || TO\_CHAR(v\_TransactionDate, 'YYYY-MM-DD') || ', Amount: ' || v\_Amount || ', Type: ' || v\_TransactionType);

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

FETCH cur\_Transactions INTO v\_CustomerID, v\_Name, v\_TransactionDate, v\_Amount, v\_TransactionType;

END LOOP;

CLOSE cur\_Transactions;

END;

/

**Scenario 2: ApplyAnnualFee - Deduct Annual Maintenance Fee**

DECLARE

CURSOR cur\_Accounts IS

SELECT AccountID, Balance

FROM Accounts;

v\_AccountID Accounts.AccountID%TYPE;

v\_Balance Accounts.Balance%TYPE;

c\_AnnualFee CONSTANT NUMBER := 50; -- Define the annual fee amount

BEGIN

OPEN cur\_Accounts;

FETCH cur\_Accounts INTO v\_AccountID, v\_Balance;

WHILE cur\_Accounts%FOUND LOOP

-- Deduct the annual fee from the account balance

IF v\_Balance >= c\_AnnualFee THEN

UPDATE Accounts

SET Balance = Balance - c\_AnnualFee

WHERE AccountID = v\_AccountID;

DBMS\_OUTPUT.PUT\_LINE('Annual fee applied to Account ID: ' || v\_AccountID || ', New Balance: ' || (v\_Balance - c\_AnnualFee));

ELSE

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance for Account ID: ' || v\_AccountID || ' to deduct annual fee.');

END IF;

FETCH cur\_Accounts INTO v\_AccountID, v\_Balance;

END LOOP;

CLOSE cur\_Accounts;

COMMIT; -- Commit the transaction to apply changes

END;

/

**Scenario 3: UpdateLoanInterestRates - Update Interest Rates Based on New Policy**

DECLARE

CURSOR cur\_Loans IS

SELECT LoanID, InterestRate

FROM Loans;

v\_LoanID Loans.LoanID%TYPE;

v\_InterestRate Loans.InterestRate%TYPE;

c\_HighRateReduction CONSTANT NUMBER := 0.5; -- Reduction for high rates

c\_LowRateReduction CONSTANT NUMBER := 0.25; -- Reduction for low rates

BEGIN

OPEN cur\_Loans;

FETCH cur\_Loans INTO v\_LoanID, v\_InterestRate;

WHILE cur\_Loans%FOUND LOOP

-- Update the interest rate based on the new policy

IF v\_InterestRate > 5 THEN

UPDATE Loans

SET InterestRate = InterestRate - c\_HighRateReduction

WHERE LoanID = v\_LoanID;

DBMS\_OUTPUT.PUT\_LINE('Reduced interest rate for Loan ID: ' || v\_LoanID || ', New Rate: ' || (v\_InterestRate - c\_HighRateReduction));

ELSE

UPDATE Loans

SET InterestRate = InterestRate - c\_LowRateReduction

WHERE LoanID = v\_LoanID;

DBMS\_OUTPUT.PUT\_LINE('Reduced interest rate for Loan ID: ' || v\_LoanID || ', New Rate: ' || (v\_InterestRate - c\_LowRateReduction));

END IF;

FETCH cur\_Loans INTO v\_LoanID, v\_InterestRate;

END LOOP;

CLOSE cur\_Loans;

COMMIT; -- Commit the transaction to apply changes

END;

/