**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.

**CODE:** DECLARE

-- Cursor to retrieve transactions for the current month

CURSOR c\_transactions IS

SELECT t.ACCOUNT\_ID, a.CUSTOMER\_NAME, t.TRANSACTION\_DATE, t.AMOUNT

FROM TRANSACTIONS t

JOIN ACCOUNTS a ON t.ACCOUNT\_ID = a.ACCOUNT\_ID

WHERE EXTRACT(MONTH FROM t.TRANSACTION\_DATE) = EXTRACT(MONTH FROM SYSDATE)

AND EXTRACT(YEAR FROM t.TRANSACTION\_DATE) = EXTRACT(YEAR FROM SYSDATE)

ORDER BY a.CUSTOMER\_NAME, t.TRANSACTION\_DATE;

-- Record to hold cursor data

r\_transaction c\_transactions%ROWTYPE;

BEGIN

-- Open the cursor and fetch each record

OPEN c\_transactions;

LOOP

FETCH c\_transactions INTO r\_transaction;

EXIT WHEN c\_transactions%NOTFOUND;

-- Print statement for each transaction

DBMS\_OUTPUT.PUT\_LINE('Customer: ' || r\_transaction.CUSTOMER\_NAME);

DBMS\_OUTPUT.PUT\_LINE('Transaction Date: ' || TO\_CHAR(r\_transaction.TRANSACTION\_DATE, 'DD-MON-YYYY'));

DBMS\_OUTPUT.PUT\_LINE('Amount: ' || TO\_CHAR(r\_transaction.AMOUNT, 'FM999,999.00'));

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

END LOOP;

-- Close the cursor

CLOSE c\_transactions;

EXCEPTION

WHEN OTHERS THEN

-- Handle any unexpected errors

DBMS\_OUTPUT.PUT\_LINE('Error generating statements: ' || 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.

**CODE: DECLARE**

v\_annual\_fee CONSTANT NUMBER := 50.00;

CURSOR c\_accounts IS

SELECT ACCOUNT\_ID, BALANCE

FROM ACCOUNTS

FOR UPDATE;

r\_account c\_accounts%ROWTYPE;

BEGIN

OPEN c\_accounts;

LOOP

FETCH c\_accounts INTO r\_account;

EXIT WHEN c\_accounts%NOTFOUND;

IF r\_account.BALANCE >= v\_annual\_fee THEN

UPDATE ACCOUNTS

SET BALANCE = r\_account.BALANCE - v\_annual\_fee

WHERE ACCOUNT\_ID = r\_account.ACCOUNT\_ID;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Account ' || r\_account.ACCOUNT\_ID || ' has insufficient balance for fee deduction.');

END IF;

END LOOP;

COMMIT;

CLOSE c\_accounts;

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

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

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.

**CODE:**

DECLARE

v\_new\_interest\_rate CONSTANT NUMBER := 5.00;

CURSOR c\_loans IS

SELECT LOAN\_ID, INTEREST\_RATE

FROM LOANS

FOR UPDATE;

r\_loan c\_loans%ROWTYPE;

BEGIN

OPEN c\_loans;

LOOP

FETCH c\_loans INTO r\_loan;

EXIT WHEN c\_loans%NOTFOUND;

UPDATE LOANS

SET INTEREST\_RATE = v\_new\_interest\_rate

WHERE LOAN\_ID = r\_loan.LOAN\_ID;

END LOOP;

COMMIT;

CLOSE c\_loans;

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

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

END;