**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.

**PL/SQL Block :-**

DECLARE

CURSOR cur\_monthly\_transactions IS

SELECT c.CustomerID, c.Name AS CustomerName, 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\_customer\_id Customers.CustomerID%TYPE;

v\_name Customers.Name%TYPE;

v\_date Transactions.TransactionDate%TYPE;

v\_amount Transactions.Amount%TYPE;

v\_type Transactions.TransactionType%TYPE;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('--- Monthly Customer Transaction Statements ---');

OPEN cur\_monthly\_transactions;

LOOP

FETCH cur\_monthly\_transactions INTO v\_customer\_id, v\_name, v\_date, v\_amount, v\_type;

EXIT WHEN cur\_monthly\_transactions%NOTFOUND;

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

', Name: ' || v\_name ||

', Date: ' || TO\_CHAR(v\_date, 'DD-MON-YYYY') ||

', Amount: ' || v\_amount ||

', Type: ' || v\_type);

END LOOP;

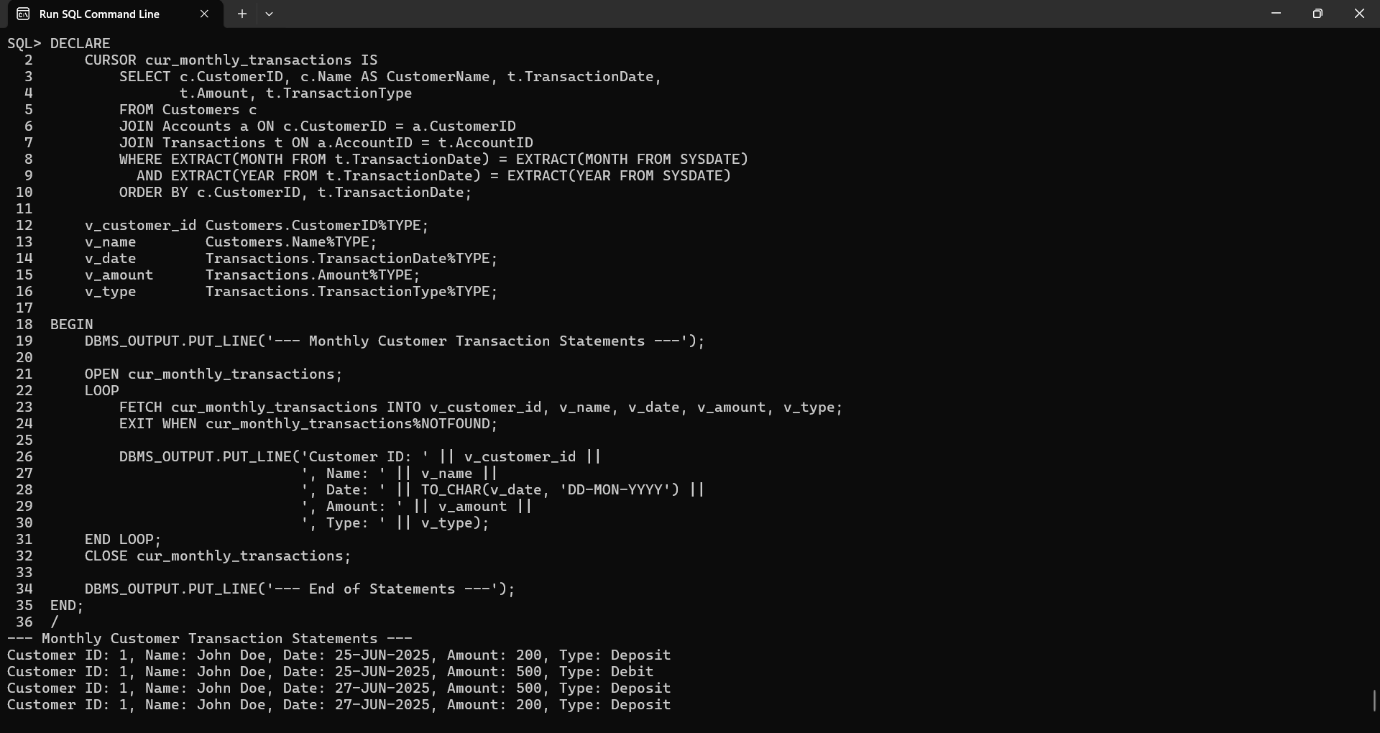
CLOSE cur\_monthly\_transactions;

DBMS\_OUTPUT.PUT\_LINE('--- End of Statements ---');

**END;**

**/**

**Output :-**



**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.

**PL/SQL Block :-**

DECLARE

-- Annual fee amount

v\_annual\_fee CONSTANT NUMBER := 200;

-- Cursor to fetch all accounts

CURSOR cur\_accounts IS

SELECT AccountID, Balance

FROM Accounts

FOR UPDATE;

-- Variables to hold account data

v\_account\_id Accounts.AccountID%TYPE;

v\_balance Accounts.Balance%TYPE;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('--- Applying Annual Fee to All Accounts ---');

OPEN cur\_accounts;

LOOP

FETCH cur\_accounts INTO v\_account\_id, v\_balance;

EXIT WHEN cur\_accounts%NOTFOUND;

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

UPDATE Accounts

SET Balance = Balance - v\_annual\_fee

WHERE AccountID = v\_account\_id;

DBMS\_OUTPUT.PUT\_LINE('Fee applied to Account ID: ' || v\_account\_id ||

', New Balance: ' || (v\_balance - v\_annual\_fee));

ELSE

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

' due to insufficient balance.');

END IF;

END LOOP;

CLOSE cur\_accounts;

DBMS\_OUTPUT.PUT\_LINE('--- Annual Fee Application Completed ---');

END;

/

**Output :-**



**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.

**PL/SQL Block :-**

DECLARE

-- Cursor to fetch all loans

CURSOR cur\_loans IS

SELECT LoanID, InterestRate

FROM Loans

FOR UPDATE;

-- Variables for cursor

v\_loan\_id Loans.LoanID%TYPE;

v\_interest Loans.InterestRate%TYPE;

v\_new\_rate NUMBER;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('--- Updating Loan Interest Rates ---');

OPEN cur\_loans;

LOOP

FETCH cur\_loans INTO v\_loan\_id, v\_interest;

EXIT WHEN cur\_loans%NOTFOUND;

-- Apply policy

IF v\_interest < 6 THEN

v\_new\_rate := v\_interest + 1;

ELSIF v\_interest BETWEEN 6 AND 10 THEN

v\_new\_rate := v\_interest + 0.5;

ELSE

v\_new\_rate := v\_interest; -- No change

END IF;

-- Update if there is a change

IF v\_new\_rate != v\_interest THEN

UPDATE Loans

SET InterestRate = v\_new\_rate

WHERE LoanID = v\_loan\_id;

DBMS\_OUTPUT.PUT\_LINE('Loan ID: ' || v\_loan\_id ||

' | Old Rate: ' || v\_interest ||

' | New Rate: ' || v\_new\_rate);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Loan ID: ' || v\_loan\_id ||

' | Interest Rate unchanged (' || v\_interest || ')');

END IF;

END LOOP;

CLOSE cur\_loans;

DBMS\_OUTPUT.PUT\_LINE('--- Interest Rate Update Completed ---');

END;

/

**Output :-**

