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 cur\_Transactions IS

SELECT

c.CustomerID,

c.Name,

a.AccountID,

t.TransactionID,

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

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

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

ORDER BY

c.CustomerID, a.AccountID, t.TransactionDate;

v\_CustomerID Customers.CustomerID%TYPE;

v\_Name Customers.Name%TYPE;

v\_AccountID Accounts.AccountID%TYPE;

v\_TransactionID Transactions.TransactionID%TYPE;

v\_TransactionDate Transactions.TransactionDate%TYPE;

v\_Amount Transactions.Amount%TYPE;

v\_TransactionType Transactions.TransactionType%TYPE;

v\_CurrentCustomerID Customers.CustomerID%TYPE := NULL;

v\_CurrentAccountID Accounts.AccountID%TYPE := NULL;

BEGIN

OPEN cur\_Transactions;

LOOP

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

EXIT WHEN cur\_Transactions%NOTFOUND;

IF v\_CurrentCustomerID IS NULL OR v\_CurrentCustomerID != v\_CustomerID THEN

IF v\_CurrentCustomerID IS NOT NULL THEN

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

END IF;

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

DBMS\_OUTPUT.PUT\_LINE('Name : ' || v\_Name);

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

v\_CurrentCustomerID := v\_CustomerID;

v\_CurrentAccountID := NULL; -- Reset account ID for new customer

END IF;

IF v\_CurrentAccountID IS NULL OR v\_CurrentAccountID != v\_AccountID THEN

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

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

v\_CurrentAccountID := v\_AccountID;

END IF;

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

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

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

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

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

END LOOP;

CLOSE cur\_Transactions;

EXCEPTION

WHEN OTHERS THEN

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

IF cur\_Transactions%ISOPEN THEN

CLOSE cur\_Transactions;

END IF;

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

v\_AnnualFee NUMBER := 50;

CURSOR cur\_Accounts IS

SELECT AccountID, Balance

FROM Accounts

FOR UPDATE OF Balance;

v\_AccountID Accounts.AccountID%TYPE;

v\_Balance Accounts.Balance%TYPE;

BEGIN

OPEN cur\_Accounts;

LOOP

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

EXIT WHEN cur\_Accounts%NOTFOUND;

v\_Balance := v\_Balance - v\_AnnualFee;

UPDATE Accounts

SET Balance = v\_Balance,

LastModified = SYSDATE

WHERE CURRENT OF cur\_Accounts;

DBMS\_OUTPUT.PUT\_LINE('Account ID: ' || v\_AccountID || ' - New Balance: ' || v\_Balance);

END LOOP;

CLOSE cur\_Accounts;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

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

IF cur\_Accounts%ISOPEN THEN

CLOSE cur\_Accounts;

END IF;

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.

**‘GetNewInterestRate’ function:**

CREATE OR REPLACE FUNCTION GetNewInterestRate(v\_CurrentRate NUMBER) RETURN NUMBER IS

BEGIN

-- Example policy: increase interest rate by 0.5% if the current rate is below 5%, otherwise reduce by 0.25%

IF v\_CurrentRate < 5 THEN

RETURN v\_CurrentRate + 0.5;

ELSE

RETURN v\_CurrentRate - 0.25;

END IF;

END;

/  
  
**Cursor to update the interest rates:**

DECLARE

v\_LoanID Loans.LoanID%TYPE;

v\_InterestRate Loans.InterestRate%TYPE;

v\_NewInterestRate NUMBER;

CURSOR cur\_Loans IS

SELECT LoanID, InterestRate

FROM Loans

FOR UPDATE OF InterestRate;

BEGIN

OPEN cur\_Loans;

LOOP

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

EXIT WHEN cur\_Loans%NOTFOUND;

v\_NewInterestRate := GetNewInterestRate(v\_InterestRate);

-- Update the loan interest rate

UPDATE Loans

SET InterestRate = v\_NewInterestRate

WHERE CURRENT OF cur\_Loans;

DBMS\_OUTPUT.PUT\_LINE('Loan ID: ' || v\_LoanID || ' - New Interest Rate: ' || v\_NewInterestRate);

END LOOP;

CLOSE cur\_Loans;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

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

IF cur\_Loans%ISOPEN THEN

CLOSE cur\_Loans;

END IF;

ROLLBACK;

END;

/