**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:**

SET SERVEROUTPUT ON;

DECLARE

CURSOR trans\_cursor IS

SELECT c.Name, t.AccountID, t.TransactionDate, t.Amount, t.TransactionType

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\_name Customers.Name%TYPE;

v\_acc\_id Accounts.AccountID%TYPE;

v\_date Transactions.TransactionDate%TYPE;

v\_amt Transactions.Amount%TYPE;

v\_type Transactions.TransactionType%TYPE;

BEGIN

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

OPEN trans\_cursor;

LOOP

FETCH trans\_cursor INTO v\_name, v\_acc\_id, v\_date, v\_amt, v\_type;

EXIT WHEN trans\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Customer: ' || v\_name || ' | Account: ' || v\_acc\_id ||

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

' | Type: ' || v\_type || ' | Amount: ' || v\_amt);

END LOOP;

CLOSE trans\_cursor;

END;

/

**Output:**

**A computer screen shot of white text

AI-generated content may be incorrect.A black screen with white text

AI-generated content may be incorrect.**

**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:**

SET SERVEROUTPUT ON;

DECLARE

CURSOR acc\_cursor IS

SELECT AccountID, Balance FROM Accounts;

v\_acc\_id Accounts.AccountID%TYPE;

v\_balance Accounts.Balance%TYPE;

v\_fee CONSTANT NUMBER := 100; -- Example fixed fee

BEGIN

OPEN acc\_cursor;

LOOP

FETCH acc\_cursor INTO v\_acc\_id, v\_balance;

EXIT WHEN acc\_cursor%NOTFOUND;

IF v\_balance >= v\_fee THEN

UPDATE Accounts

SET Balance = Balance - v\_fee,

LastModified = SYSDATE

WHERE AccountID = v\_acc\_id;

DBMS\_OUTPUT.PUT\_LINE('Fee applied to Account: ' || v\_acc\_id || ' | Fee: ' || v\_fee);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Skipped Account: ' || v\_acc\_id || ' | Insufficient Balance');

END IF;

END LOOP;

CLOSE acc\_cursor;

COMMIT;

END;

/

**Output:**

**A computer screen shot of white text

AI-generated content may be incorrect.**

**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:**

SET SERVEROUTPUT ON;

DECLARE

CURSOR loan\_cursor IS

SELECT LoanID, LoanAmount, InterestRate FROM Loans

FOR UPDATE;

v\_loan\_id Loans.LoanID%TYPE;

v\_loan\_amt Loans.LoanAmount%TYPE;

v\_interest Loans.InterestRate%TYPE;

BEGIN

OPEN loan\_cursor;

LOOP

FETCH loan\_cursor INTO v\_loan\_id, v\_loan\_amt, v\_interest;

EXIT WHEN loan\_cursor%NOTFOUND;

IF v\_loan\_amt > 10000 THEN

UPDATE Loans

SET InterestRate = InterestRate + 0.5

WHERE LoanID = v\_loan\_id;

DBMS\_OUTPUT.PUT\_LINE('Increased rate for Loan ' || v\_loan\_id);

ELSE

UPDATE Loans

SET InterestRate = InterestRate - 0.25

WHERE LoanID = v\_loan\_id;

DBMS\_OUTPUT.PUT\_LINE('Decreased rate for Loan ' || v\_loan\_id);

END IF;

END LOOP;

CLOSE loan\_cursor;

COMMIT;

END;

/

**Output:**

**A computer screen with white text

AI-generated content may be incorrect.**