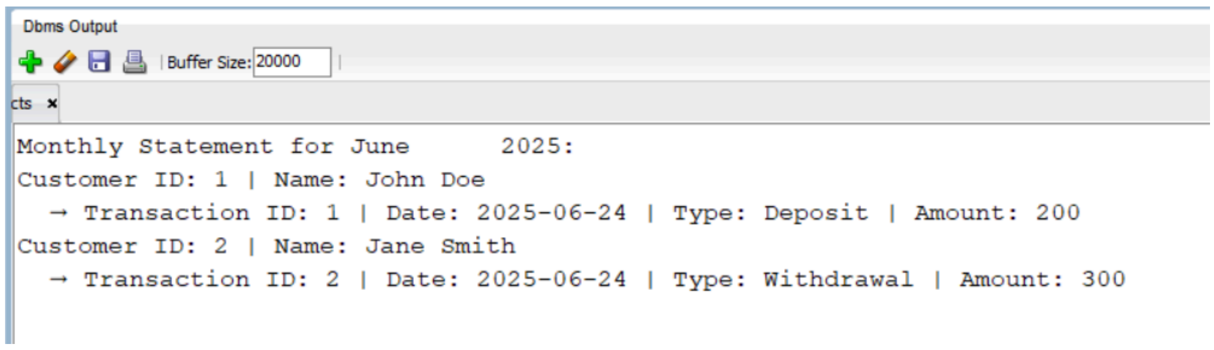
Exercise 6: Cursors

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.

Solution:

```
DECLARE
   CURSOR txn_cursor IS
      SELECT c.CustomerID, c.Name, 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 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_cust_id     Customers.CustomerID%TYPE;
   v_name        Customers.Name%TYPE;
   v_txn_id      Transactions.TransactionID%TYPE;
   v_txn_date    Transactions.TransactionDate%TYPE;
   v_amount      Transactions.Amount%TYPE;
   v_type        Transactions.TransactionType%TYPE;
BEGIN
   DBMS_OUTPUT.PUT_LINE('Monthly Statement for ' || TO_CHAR(SYSDATE, 'Month
YYYY') || ':');

   OPEN txn_cursor;
   LOOP
      FETCH txn_cursor INTO v_cust_id, v_name, v_txn_id, v_txn_date, v_amount, v_type;
      EXIT WHEN txn_cursor%NOTFOUND;

      DBMS_OUTPUT.PUT_LINE('Customer ID: ' || v_cust_id || ' | Name: ' || v_name);
      DBMS_OUTPUT.PUT_LINE('  → Transaction ID: ' || v_txn_id ||
                     ' | Date: ' || TO_CHAR(v_txn_date, 'YYYY-MM-DD') ||
                     ' | Type: ' || v_type ||
                     ' | Amount: ' || v_amount);
   END LOOP;
   CLOSE txn_cursor;
```

```
END;
/
```



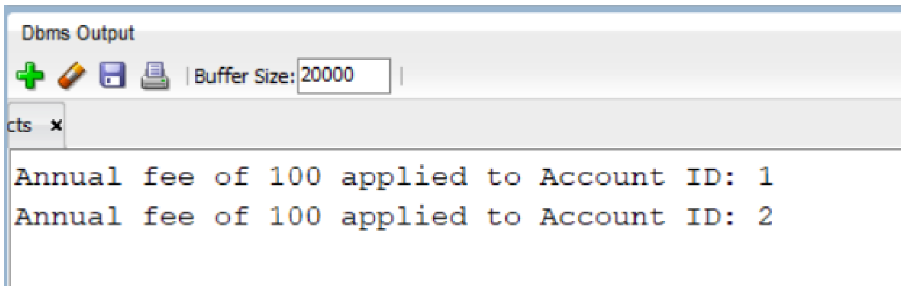
```
Monthly Statement for June      2025:
Customer ID: 1 | Name: John Doe
  → Transaction ID: 1 | Date: 2025-06-24 | Type: Deposit | Amount: 200
Customer ID: 2 | Name: Jane Smith
  → Transaction ID: 2 | Date: 2025-06-24 | Type: Withdrawal | Amount: 300
```

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.

Solution:

```
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;  -- Annual maintenance 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('Annual fee of ' || v_fee || ' applied to Account ID: ' ||
v_acc_id);
        ELSE
            DBMS_OUTPUT.PUT_LINE('Skipping Account ID: ' || v_acc_id || ' due to
insufficient balance.');
        END IF;
    END LOOP;
    CLOSE acc_cursor;

    COMMIT;
END;
/
```

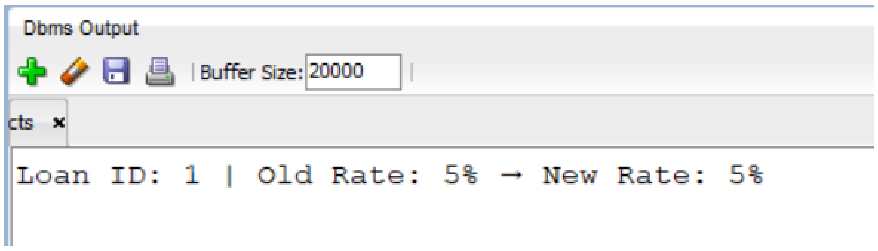


```
Annual fee of 100 applied to Account ID: 1
Annual fee of 100 applied to Account ID: 2
```

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.

Solution:

```
SET SERVEROUTPUT ON;

DECLARE
    CURSOR loan_cursor IS
        SELECT LoanID, LoanAmount, InterestRate
        FROM Loans;

    v_loan_id    Loans.LoanID%TYPE;
    v_amount     Loans.LoanAmount%TYPE;
    v_old_rate   Loans.InterestRate%TYPE;
    v_new_rate   NUMBER;
BEGIN
    OPEN loan_cursor;
    LOOP
        FETCH loan_cursor INTO v_loan_id, v_amount, v_old_rate;
        EXIT WHEN loan_cursor%NOTFOUND;

        -- Determine new rate based on amount
        IF v_amount < 5000 THEN
          v_new_rate := 6;
        ELSIF v_amount <= 10000 THEN
          v_new_rate := 5;
        ELSE
          v_new_rate := 4.5;
        END IF;

        UPDATE Loans
        SET InterestRate = v_new_rate
        WHERE LoanID = v_loan_id;

        DBMS_OUTPUT.PUT_LINE('Loan ID: ' || v_loan_id ||
                             ' | Old Rate: ' || v_old_rate ||
                             '% → New Rate: ' || v_new_rate || '%');
    END LOOP;
    CLOSE loan_cursor;

    COMMIT;
END;
/
```



```
Loan ID: 1 | Old Rate: 5% → New Rate: 5%
```