**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.

**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.

**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.

**Scenario 1: GenerateMonthlyStatements — Print Monthly Transactions**

**CODE:-**

DECLARE

  CURSOR trans\_cursor IS

    SELECT T.TransactionID, T.AccountID, T.TransactionDate, T.Amount, T.TransactionType, C.Name

    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);

  v\_trans trans\_cursor%ROWTYPE;

BEGIN

  OPEN trans\_cursor;

  LOOP

    FETCH trans\_cursor INTO v\_trans;

    EXIT WHEN trans\_cursor%NOTFOUND;

    DBMS\_OUTPUT.PUT\_LINE('Customer: ' || v\_trans.Name ||

                         ' | Transaction ID: ' || v\_trans.TransactionID ||

                         ' | Type: ' || v\_trans.TransactionType ||

                         ' | Amount: ' || v\_trans.Amount ||

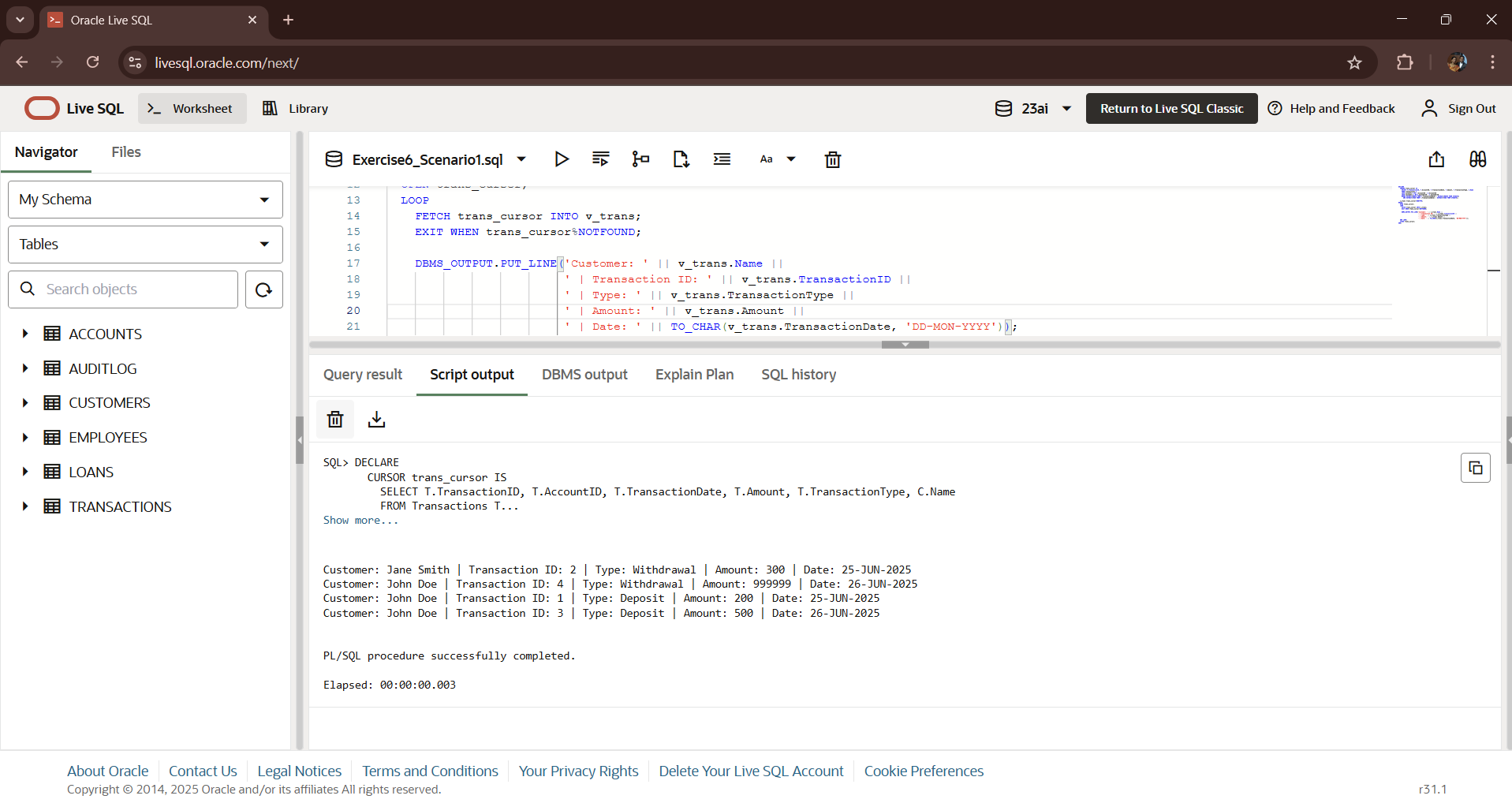
                         ' | Date: ' || TO\_CHAR(v\_trans.TransactionDate, 'DD-MON-YYYY'));

  END LOOP;

  CLOSE trans\_cursor;

END;

**OUTPUT:-**

****

**Scenario 2: ApplyAnnualFee — Deduct Fee from All Accounts**

**CODE:-**

DECLARE

  CURSOR acc\_cursor IS

    SELECT AccountID, Balance

    FROM Accounts;

  v\_acc acc\_cursor%ROWTYPE;

  v\_fee CONSTANT NUMBER := 100; -- Annual maintenance fee

BEGIN

  OPEN acc\_cursor;

  LOOP

    FETCH acc\_cursor INTO v\_acc;

    EXIT WHEN acc\_cursor%NOTFOUND;

    UPDATE Accounts

    SET Balance = Balance - v\_fee,

        LastModified = SYSDATE

    WHERE AccountID = v\_acc.AccountID;

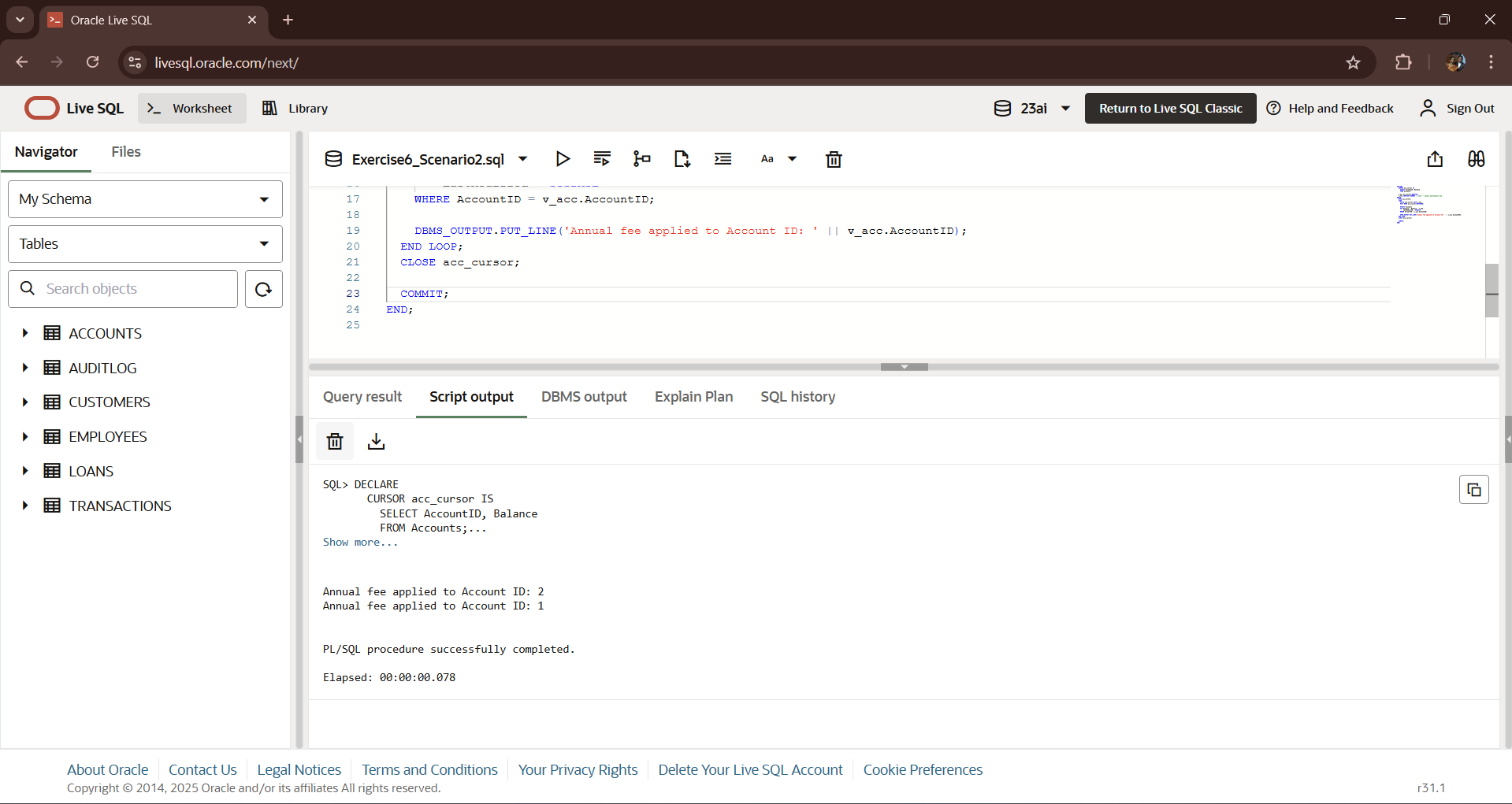
    DBMS\_OUTPUT.PUT\_LINE('Annual fee applied to Account ID: ' || v\_acc.AccountID);

  END LOOP;

  CLOSE acc\_cursor;

  COMMIT;

END;

**OUTPUT:-**

**Scenario 3: UpdateLoanInterestRates — Adjust Interest per Policy**

Assume a new policy like:

* If loan amount > 10000, reduce interest by 0.5%.
* Else increase by 0.25%.

**CODE:-**

DECLARE

  CURSOR loan\_cursor IS

    SELECT LoanID, LoanAmount, InterestRate

    FROM Loans;

  v\_loan loan\_cursor%ROWTYPE;

BEGIN

  OPEN loan\_cursor;

  LOOP

    FETCH loan\_cursor INTO v\_loan;

    EXIT WHEN loan\_cursor%NOTFOUND;

    IF v\_loan.LoanAmount > 10000 THEN

      UPDATE Loans

      SET InterestRate = InterestRate - 0.5

      WHERE LoanID = v\_loan.LoanID;

      DBMS\_OUTPUT.PUT\_LINE('Reduced interest for Loan ID: ' || v\_loan.LoanID);

    ELSE

      UPDATE Loans

      SET InterestRate = InterestRate + 0.25

      WHERE LoanID = v\_loan.LoanID;

      DBMS\_OUTPUT.PUT\_LINE('Increased interest for Loan ID: ' || v\_loan.LoanID);

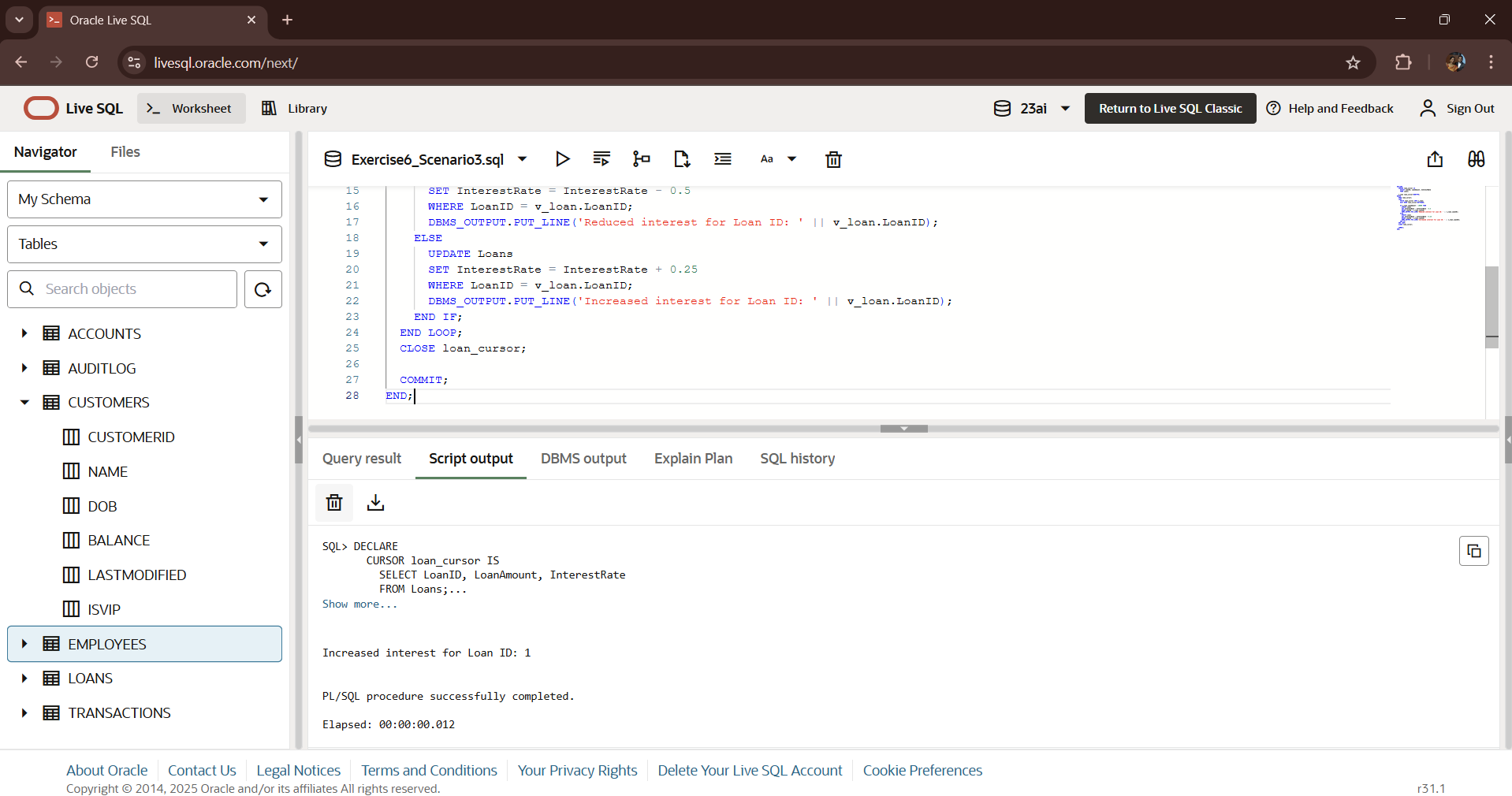
    END IF;

  END LOOP;

  CLOSE loan\_cursor;

  COMMIT;

END;

**OUTPUT:-**