**WEEK-2**

**PL/SQL programming**

**1) Control Structures**

**\* Scenario 1:** Apply 1% Discount for Customers Above 60

BEGIN

  FOR cust\_rec IN (

    SELECT l.LoanID, l.InterestRate, c.Name, c.DOB

    FROM Loans l

    JOIN Customers c ON l.CustomerID = c.CustomerID

  ) LOOP

    IF MONTHS\_BETWEEN(SYSDATE, cust\_rec.DOB) / 12 > 60 THEN

      UPDATE Loans

      SET InterestRate = InterestRate - 1

      WHERE LoanID = cust\_rec.LoanID;

      DBMS\_OUTPUT.PUT\_LINE('Discount applied for customer: ' || cust\_rec.Name);

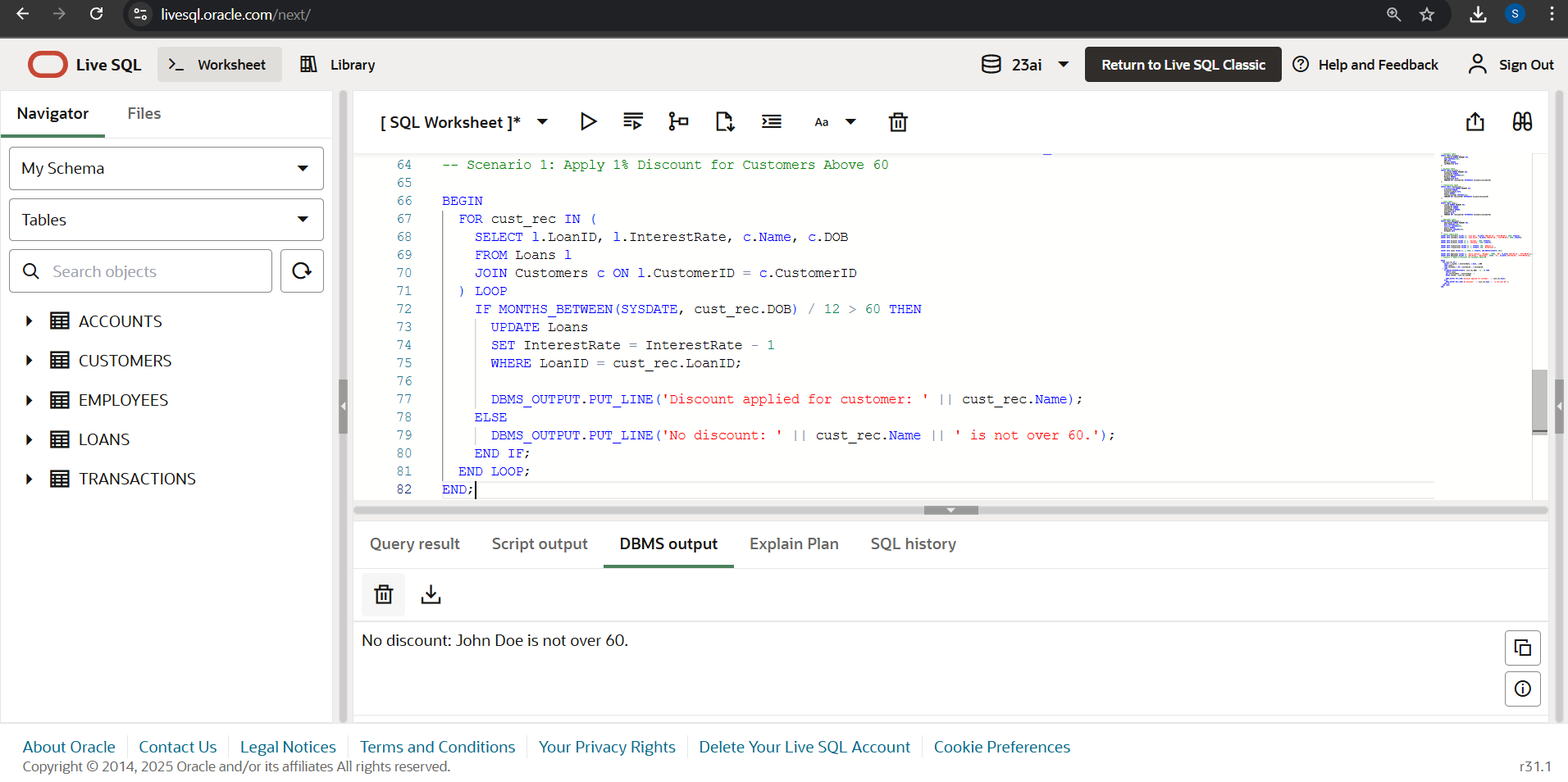
    ELSE

      DBMS\_OUTPUT.PUT\_LINE('No discount: ' || cust\_rec.Name || ' is not over 60.');

    END IF;

  END LOOP;

END;

**Output:** 

\* **Scenario 2:** Set IsVIP = TRUE for Customers with Balance > $10,000

ALTER TABLE Customers ADD IsVIP VARCHAR2(5);

BEGIN

FOR cust IN (SELECT CustomerID, Name, Balance FROM Customers) LOOP

IF cust.Balance > 10000 THEN

UPDATE Customers

SET IsVIP = 'TRUE'

WHERE CustomerID = cust.CustomerID;

DBMS\_OUTPUT.PUT\_LINE(cust.Name || ' is marked as VIP.');

ELSE

UPDATE Customers

SET IsVIP = 'FALSE'

WHERE CustomerID = cust.CustomerID;

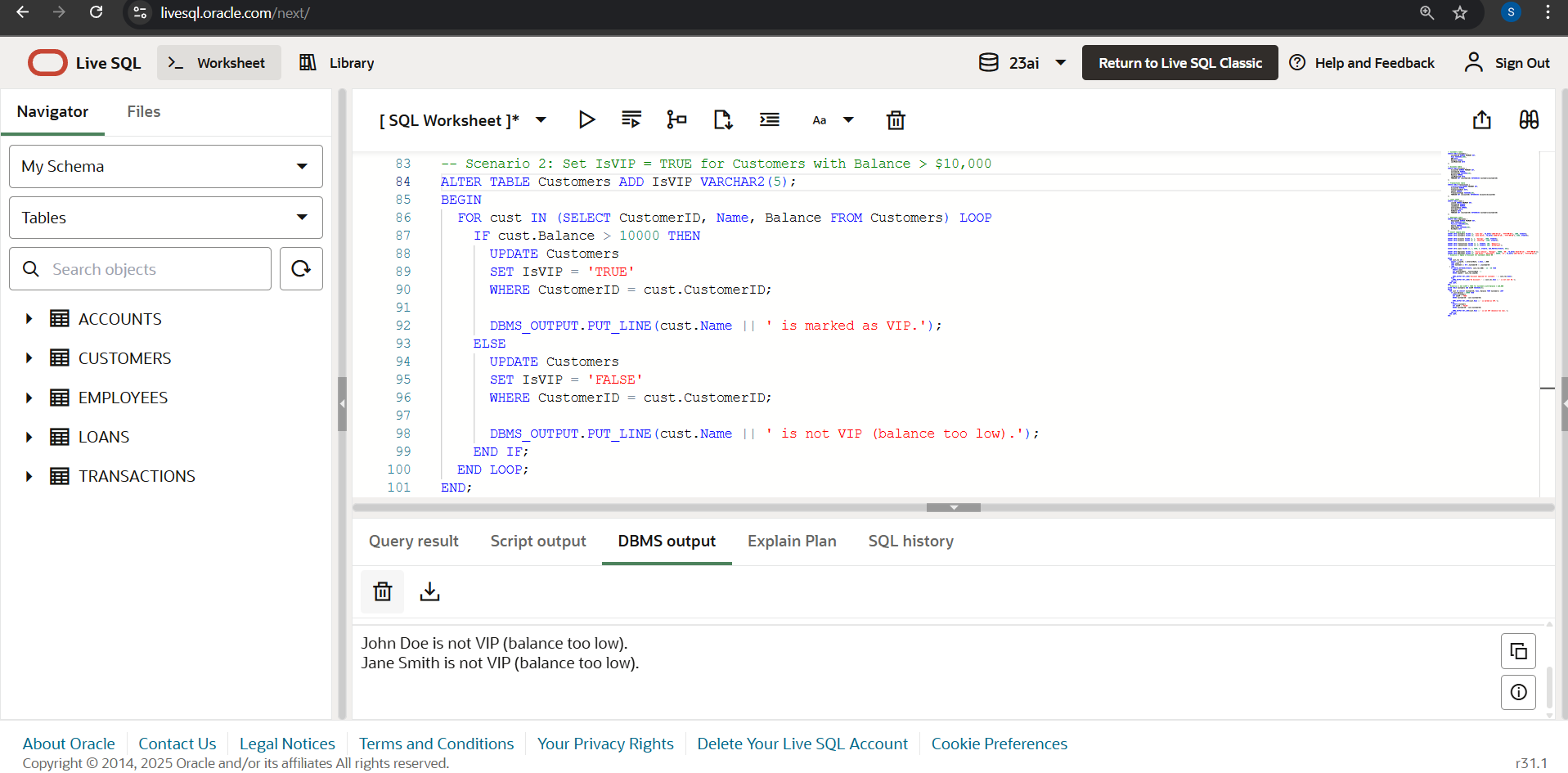
DBMS\_OUTPUT.PUT\_LINE(cust.Name || ' is not VIP (balance too low).');

END IF;

END LOOP;

END;

**Output:**



**\* Scenario 3:** Reminders for Loans Due in Next 30 Days

BEGIN

DECLARE

found BOOLEAN := FALSE;

BEGIN

FOR loan\_rec IN (

SELECT l.LoanID, l.EndDate, c.Name

FROM Loans l

JOIN Customers c ON l.CustomerID = c.CustomerID

WHERE l.EndDate BETWEEN SYSDATE AND SYSDATE + 30

) LOOP

found := TRUE;

DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ID ' || loan\_rec.LoanID || ' for ' || loan\_rec.Name || ' is due on ' || TO\_CHAR(loan\_rec.EndDate, 'YYYY-MM-DD'));

END LOOP;

IF NOT found THEN

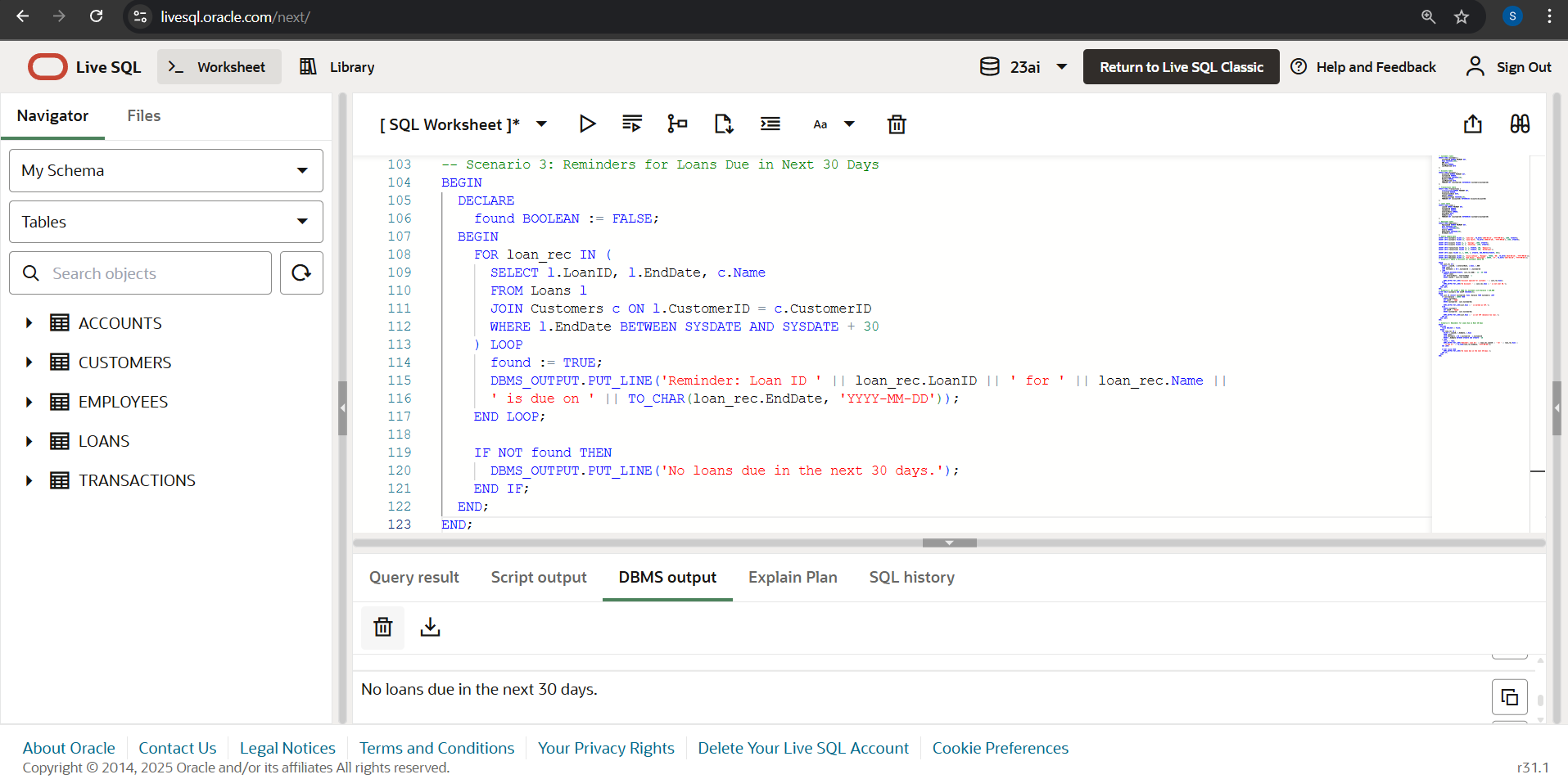
DBMS\_OUTPUT.PUT\_LINE('No loans due in the next 30 days.');

END IF;

END;

END;

**Output:**



**2)Stored Procedures**

**\* Scenario 1:** Apply 1% interest to all Savings accounts.

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

FOR acc IN (

SELECT AccountID, Balance

FROM Accounts

WHERE AccountType = 'Savings'

) LOOP

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01),

LastModified = SYSDATE

WHERE AccountID = acc.AccountID;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Monthly interest applied to all Savings accounts.');

END;

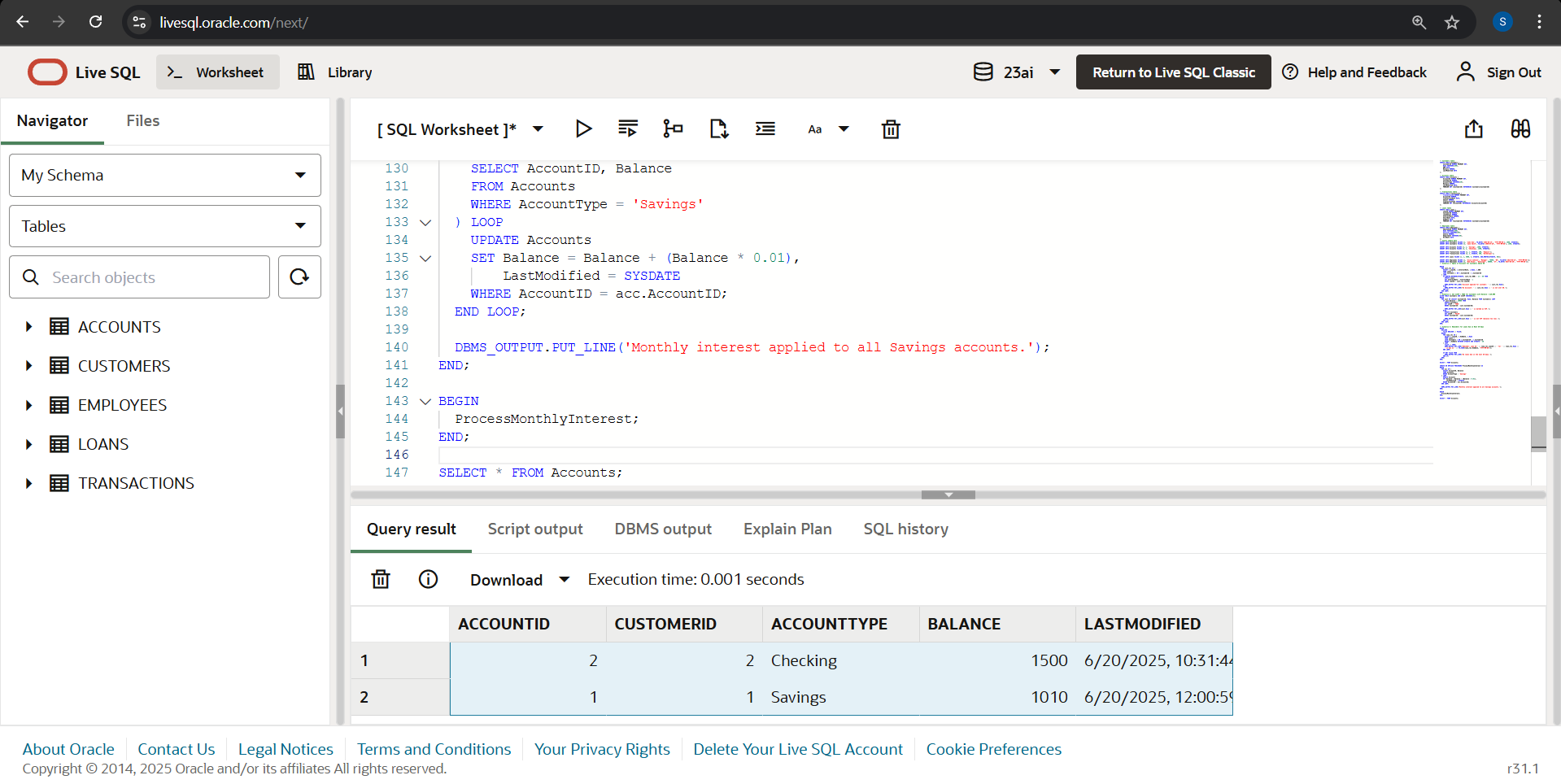
BEGIN

ProcessMonthlyInterest;

END;

SELECT \* FROM Accounts;

**Output:**



* Monthly interest applied to all Savings accounts.
* Account ID 1: Balance updated from 1000 to 1010

**\* Scenario 2:** Bonus for Employees in IT Department

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

dept\_name IN VARCHAR2,

bonus\_percent IN NUMBER

) AS

BEGIN

UPDATE Employees

SET Salary = Salary + (Salary \* bonus\_percent / 100)

WHERE Department = dept\_name;

DBMS\_OUTPUT.PUT\_LINE('Bonus applied to employees in department: ' || dept\_name);

END;

BEGIN

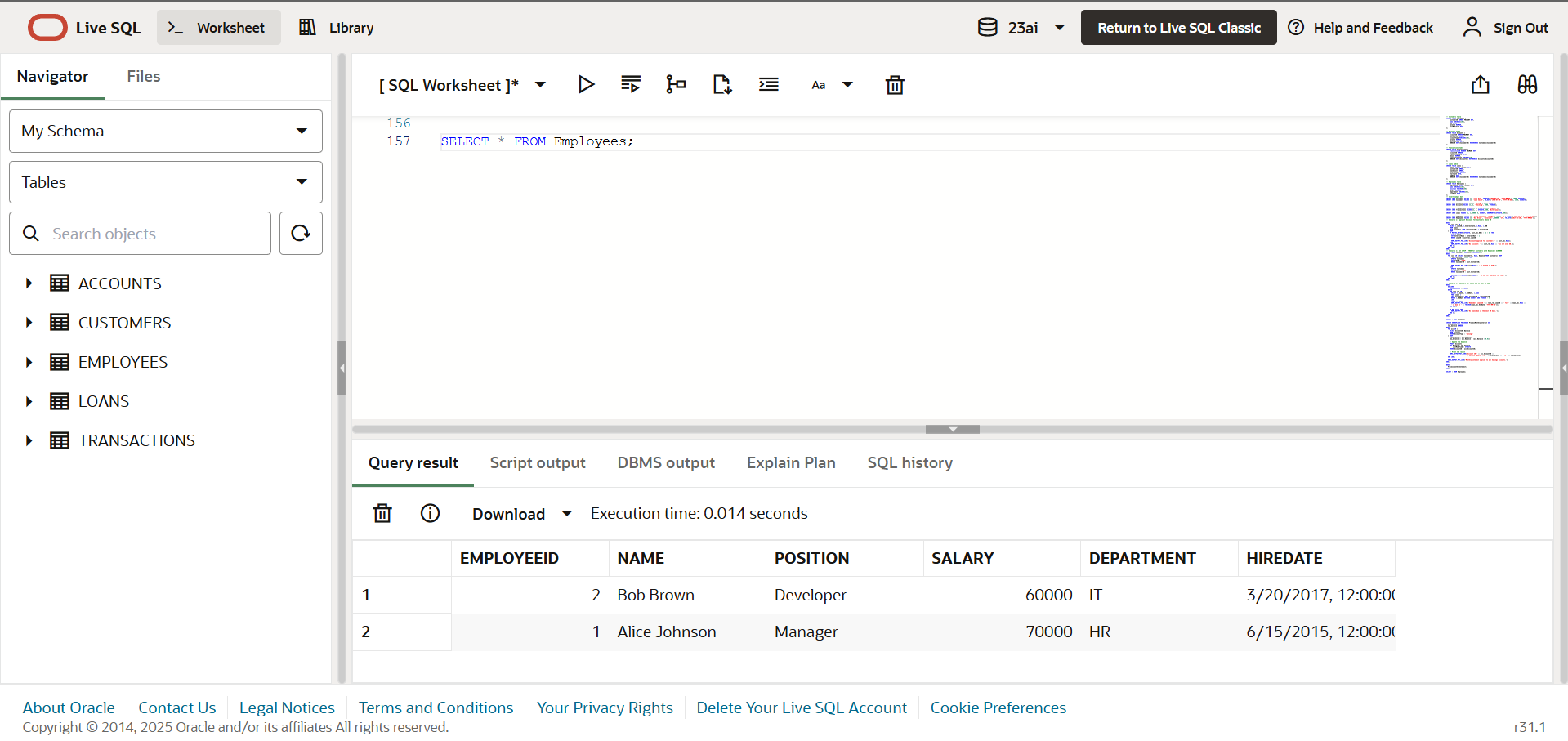
UpdateEmployeeBonus('IT', 10);

END;

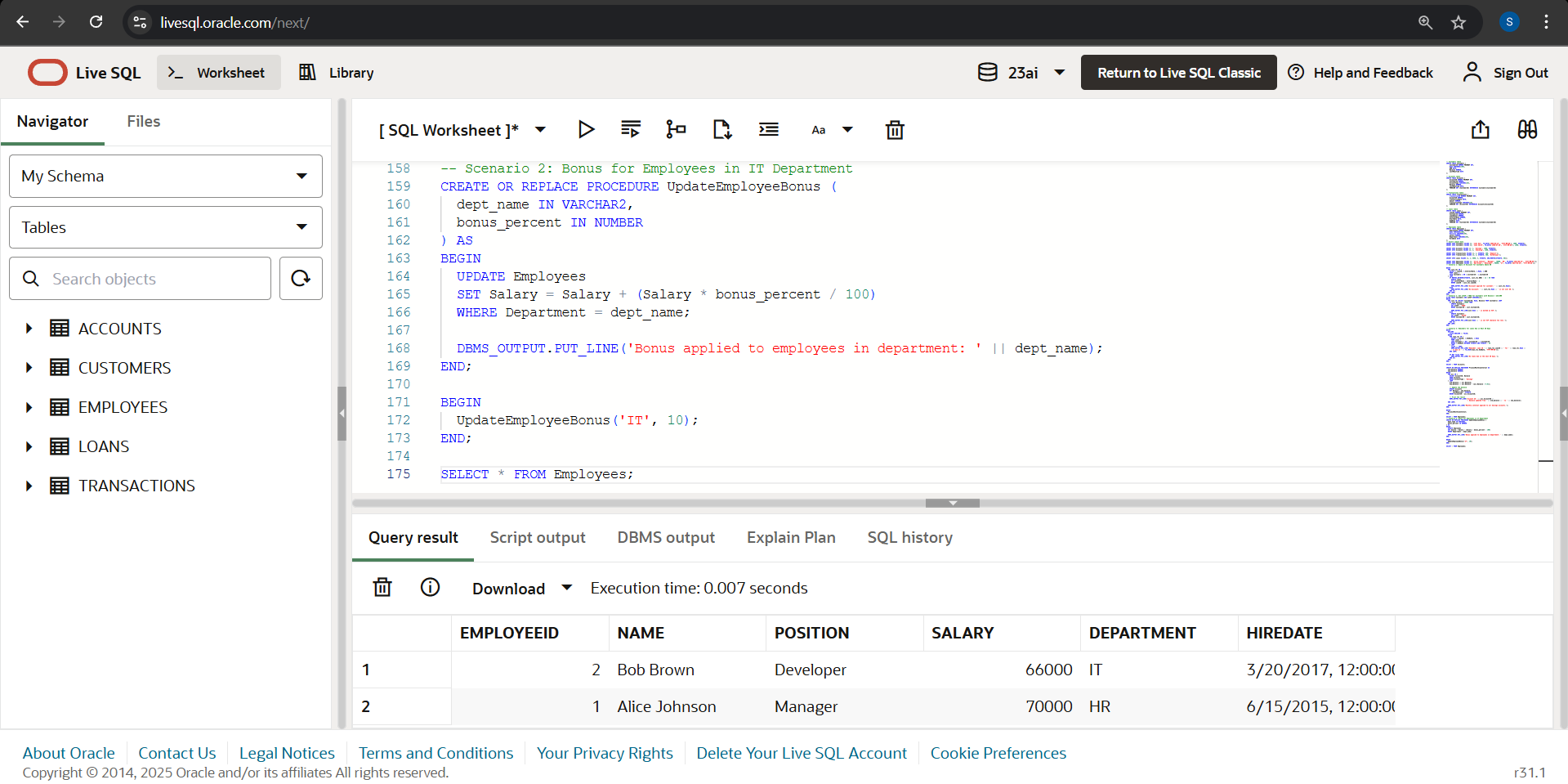
SELECT \* FROM Employees;

**Output:**

**Before:**



**After:**



* Bonus applied to employees in department: IT

**\* Scenario 3:** Transfer Funds Between Accounts

CREATE OR REPLACE PROCEDURE TransferFunds (

from\_account\_id IN NUMBER,

to\_account\_id IN NUMBER,

amount IN NUMBER

) AS

from\_balance NUMBER;

BEGIN

SELECT Balance INTO from\_balance

FROM Accounts

WHERE AccountID = from\_account\_id;

IF from\_balance < amount THEN

DBMS\_OUTPUT.PUT\_LINE('Transfer failed: Insufficient balance.');

ELSE

UPDATE Accounts

SET Balance = Balance - amount,

LastModified = SYSDATE

WHERE AccountID = from\_account\_id;

UPDATE Accounts

SET Balance = Balance + amount,

LastModified = SYSDATE

WHERE AccountID = to\_account\_id;

DBMS\_OUTPUT.PUT\_LINE('Transfer of ' || amount || ' from Account ' || from\_account\_id || ' to Account ' || to\_account\_id || ' completed.');

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('One or both account IDs not found.');

WHEN OTHERS THEN

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

END;

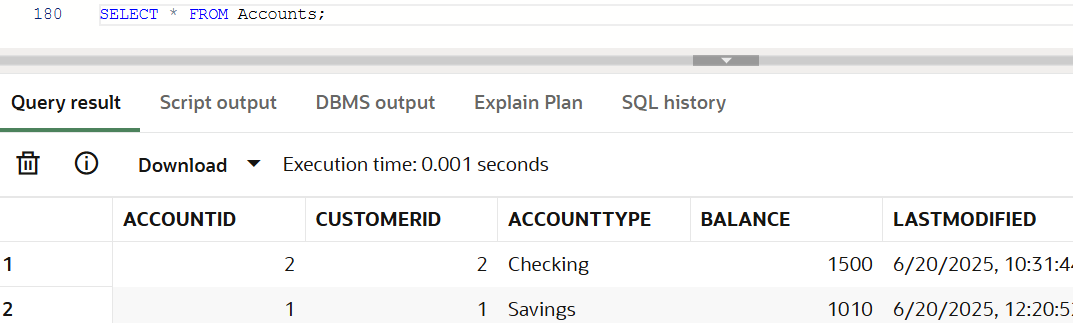
BEGIN

TransferFunds(1, 2, 500);

END;

SELECT \* FROM Accounts;

**Output :**

****

**After:**

Transfer of 500 from Account 1 to Account 2 completed.

Savings - 510 ← 1010 – 500

Checking - 2000 ← 1500 + 500

