**Exercise 3: Stored Procedures**

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

* + **Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

PROGRAM:

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE Accounts

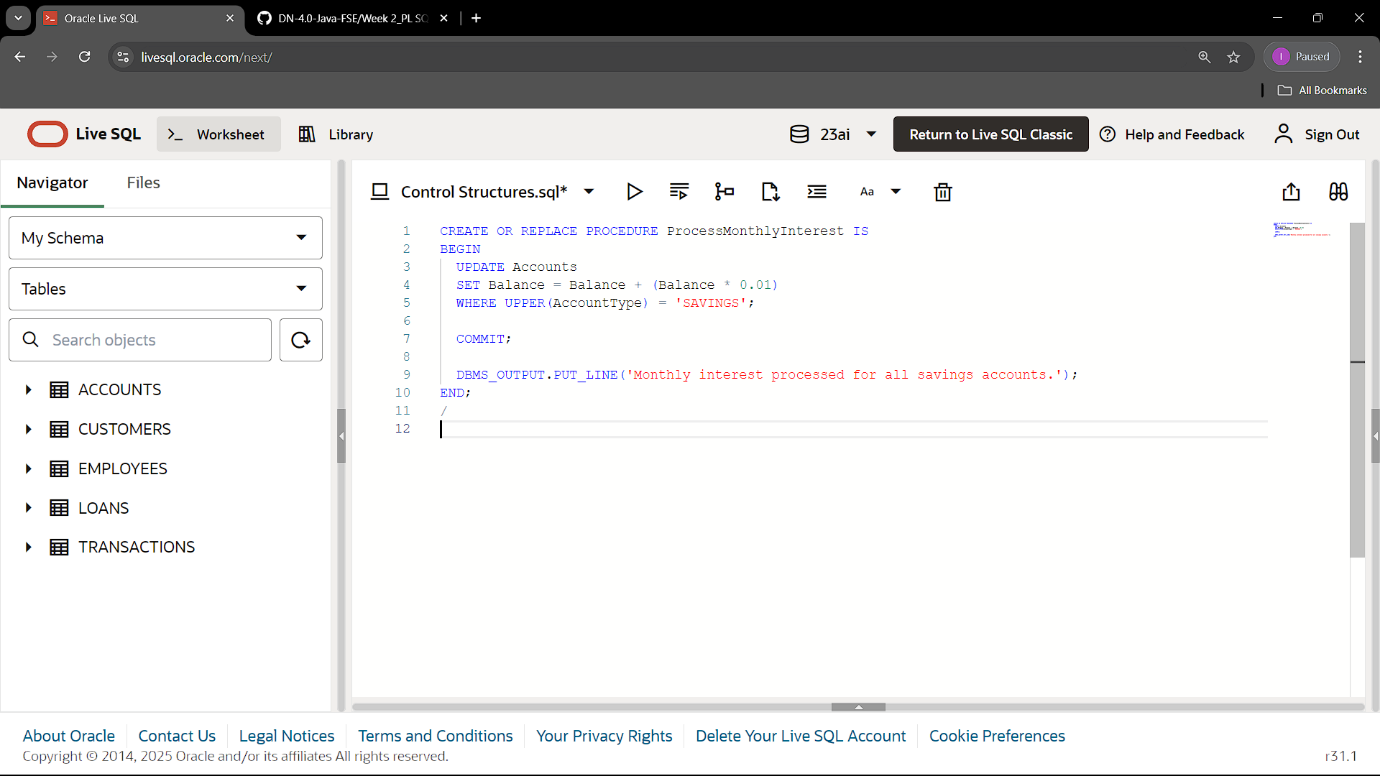
SET Balance = Balance + (Balance \* 0.01)

WHERE UPPER(AccountType) = 'SAVINGS';

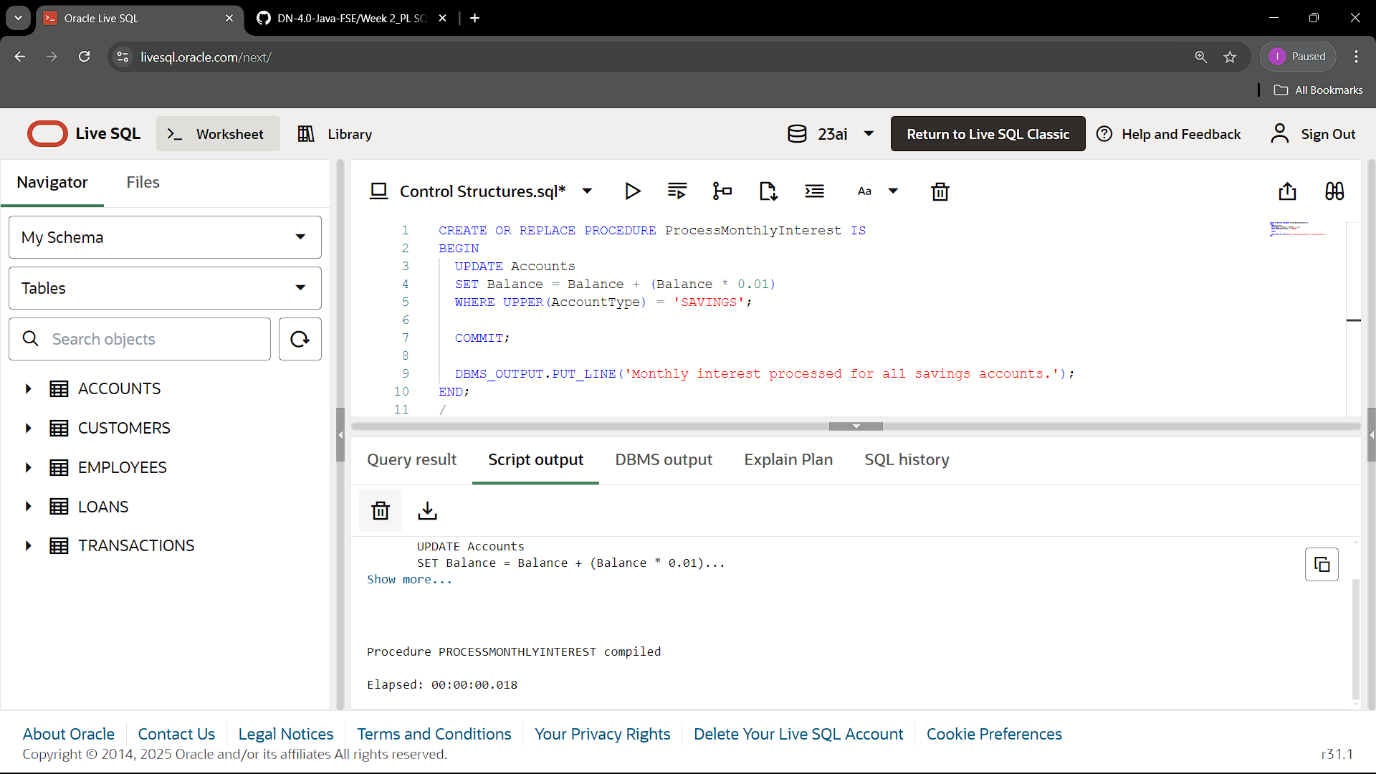
COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Monthly interest processed for all savings accounts.');

END;



SCRIPT OUTPUT:



HOW TO CALL IT:

SET SERVEROUTPUT ON;

BEGIN

ProcessMonthlyInterest;

END;

**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

* + **Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

PROGRAM TO CREATE A PROCEDURE:

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

  p\_department IN VARCHAR2,

  p\_bonus\_percent IN NUMBER

) IS

BEGIN

  UPDATE Employees

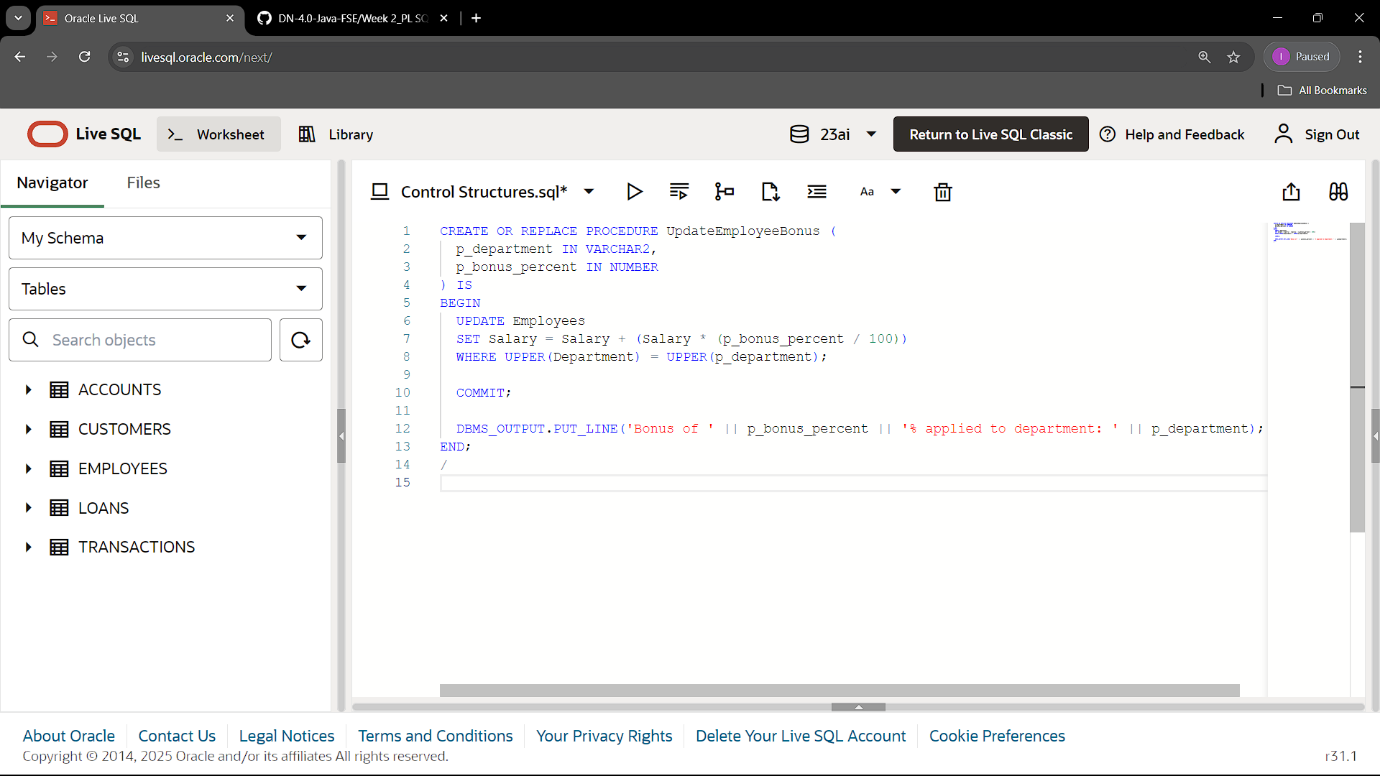
  SET Salary = Salary + (Salary \* (p\_bonus\_percent / 100))

  WHERE UPPER(Department) = UPPER(p\_department);

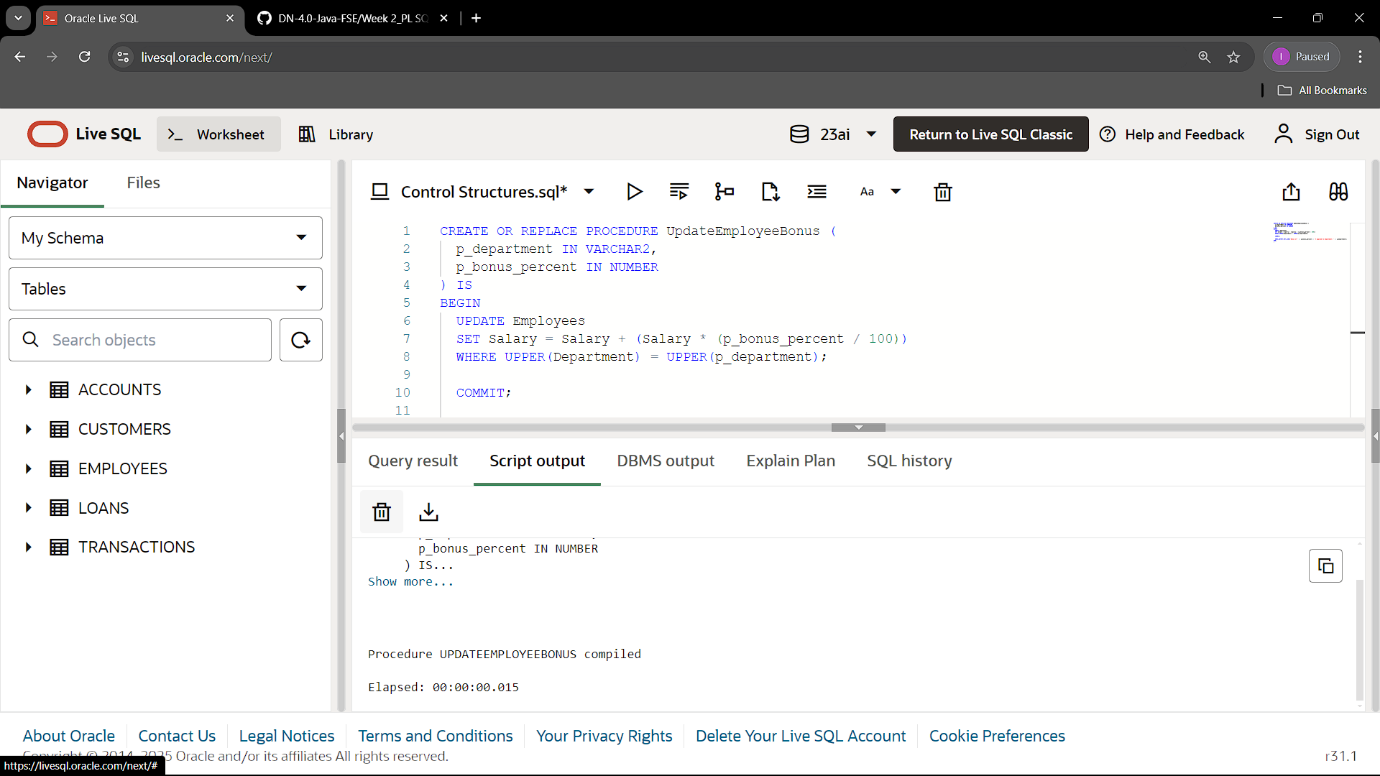
  COMMIT;

  DBMS\_OUTPUT.PUT\_LINE('Bonus of ' || p\_bonus\_percent || '% applied to department: ' || p\_department);

END;



SCRIPT OUTPUT:



HOW TO CALL IT:

SET SERVEROUTPUT ON;

BEGIN

UpdateEmployeeBonus('Finance', 15);

END;

**Scenario 3:** Customers should be able to transfer funds between their accounts.

* + **Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

CODE TO CREATE A PROCEDURE:

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_source\_account\_id IN NUMBER,

p\_destination\_account\_id IN NUMBER,

p\_amount IN NUMBER

) IS

v\_source\_balance NUMBER;

BEGIN

SELECT Balance

INTO v\_source\_balance

FROM Accounts

WHERE AccountID = p\_source\_account\_id

FOR UPDATE;

IF v\_source\_balance < p\_amount THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance in source account.');

END IF;

UPDATE Accounts

SET Balance = Balance - p\_amount

WHERE AccountID = p\_source\_account\_id;

UPDATE Accounts

SET Balance = Balance + p\_amount

WHERE AccountID = p\_destination\_account\_id;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Transferred ' || p\_amount ||

' from Account ' || p\_source\_account\_id ||

' to Account ' || p\_destination\_account\_id);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

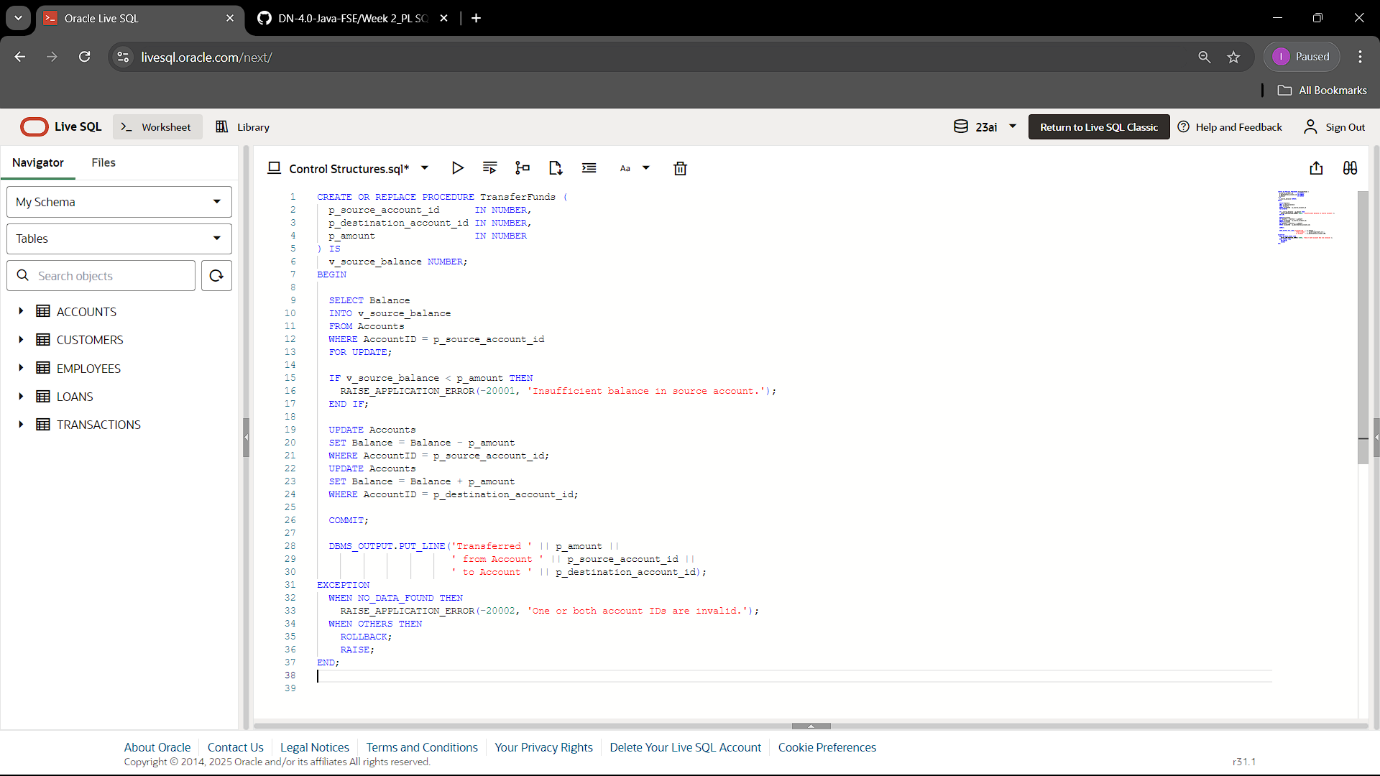
RAISE\_APPLICATION\_ERROR(-20002, 'One or both account IDs are invalid.');

WHEN OTHERS THEN

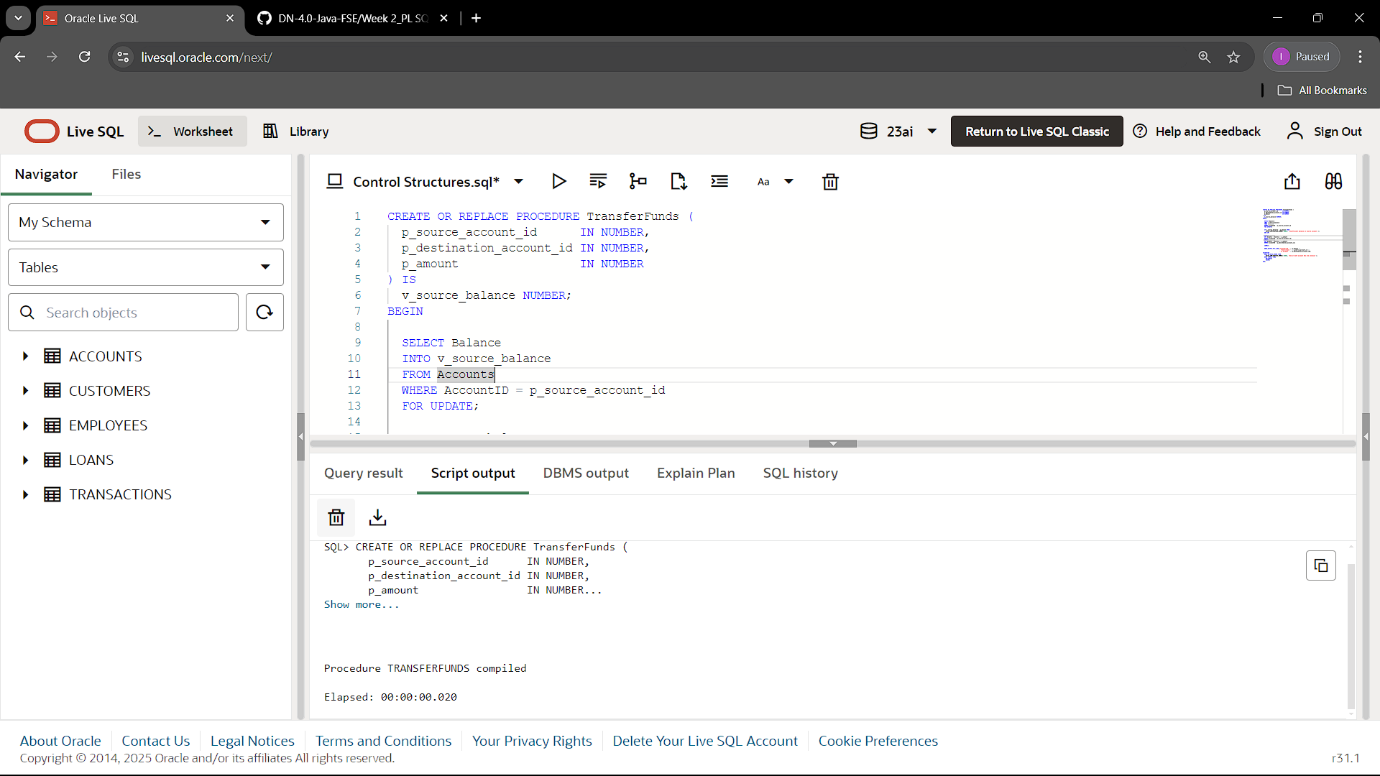
ROLLBACK;

RAISE;

END;



SCRIPT OUTPUT:



HOW TO CALL IT:

SET SERVEROUTPUT ON;

BEGIN

TransferFunds(101, 202, 500);

END;