**PLSQL\_Exercises**

**(All exercises are included)**

**Exercise 1: Control Structures**

**Scenario 1:**

**INPUT**

BEGIN

    for res in (

        select c.name,c.customerID,l.loanID,l.InterestRate

        from loans l join customers c on l.customerID=c.customerID

        where trunc(months\_between(sysdate,c.dob)/12) > 60 ) loop

        Update loans set InterestRate = res.InterestRate -1

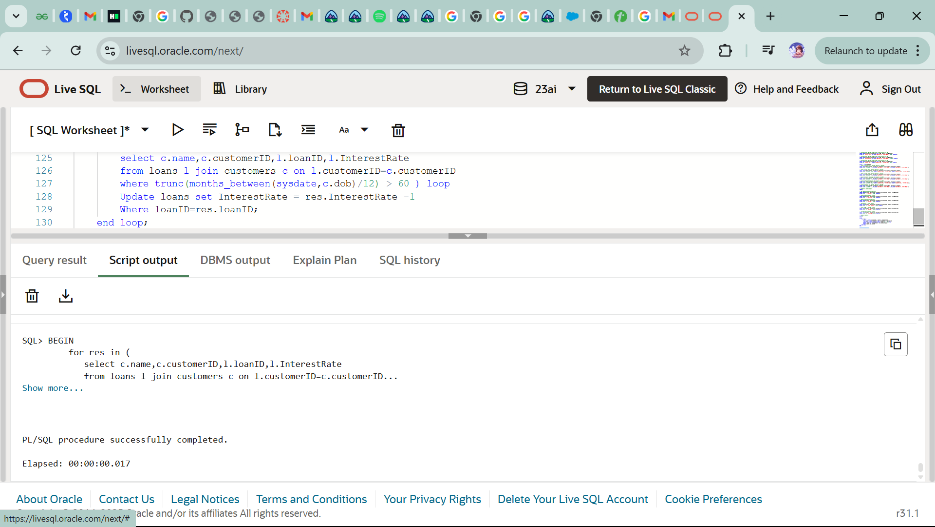
        Where loanID=res.loanID;

    end loop;

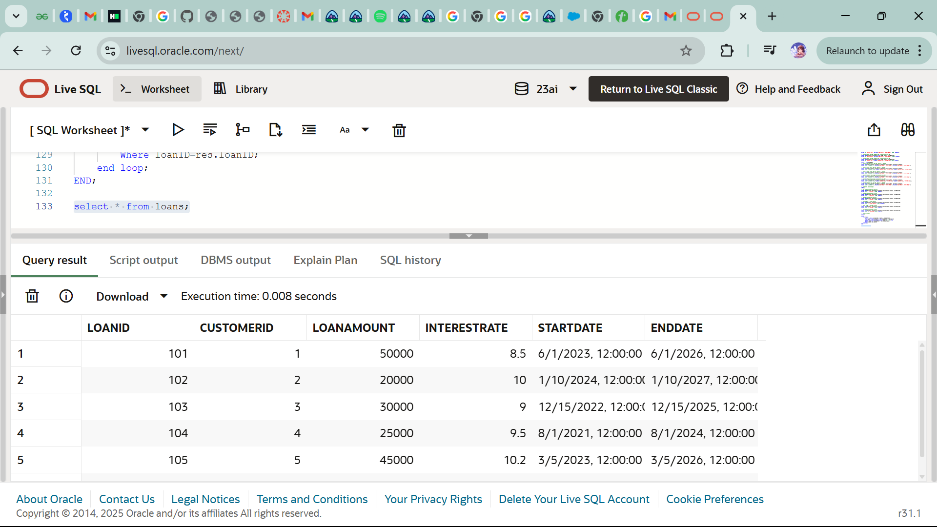
END;

**OUTPUT**

--procedure is created and executed:



--after executing procedure table LOANS:



**Scenario 2:**

**INPUT**

ALTER TABLE CUSTOMERS

ADD ISVIP VARCHAR(1);

SELECT \* FROM CUSTOMERS;

BEGIN

    for res in( select name , customerID, Balance from customers) loop

    if (res.Balance>10000) then

        Update customers set ISVIP='Y'

        where customerID=res.customerID;

    else

        Update customers set ISVIP='N'

        where customerID=res.customerID;

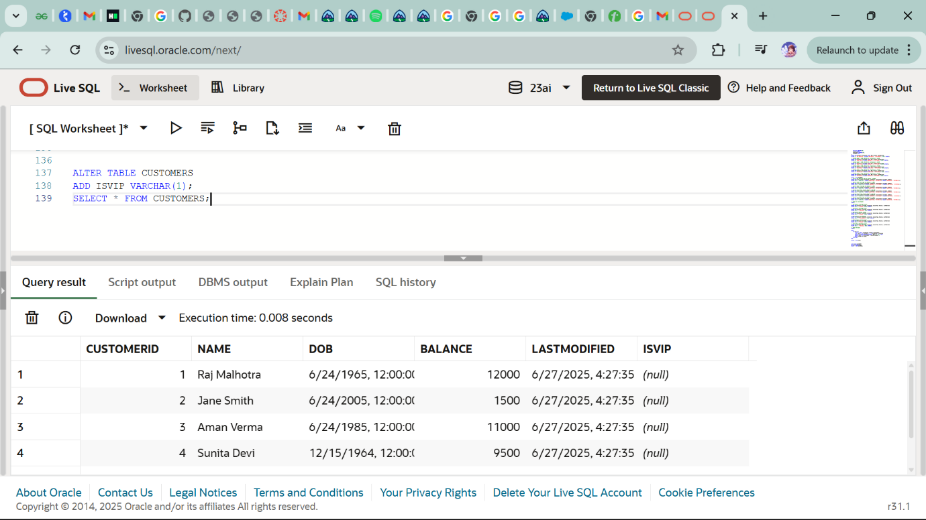
    end if;

    end loop;

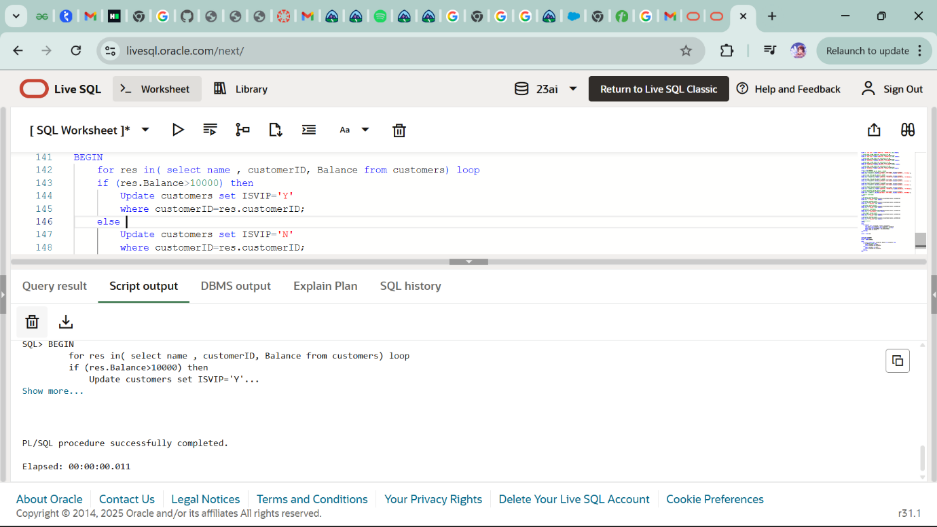
END;

**OUTPUT**

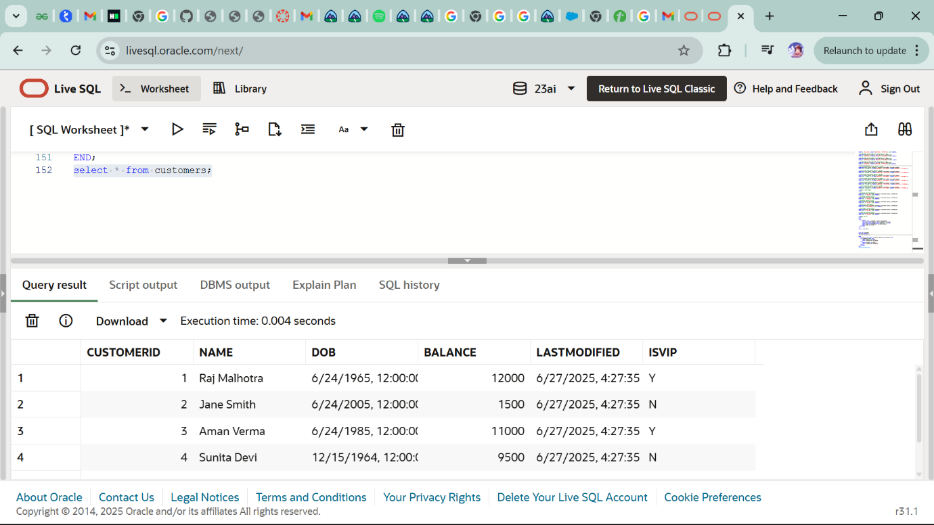
--ISVIP column is added in CUSTOMERS table:



--procedure is created and executed:



--after procedure execution table CUSTOMERS:



**Scenario 3:**

**INPUT**

SET SERVEROUTPUT ON;

BEGIN

    for res in (select c.name,c.customerID,l.loanID,l.EndDate

    from loans l join customers c on l.customerID=c.customerID

    where l.EndDate between sysdate and sysdate+30

    ) loop

    Dbms\_output.put\_line('Reminder: Dear ' || res.NAME ||

                             ', your loan is due on ' || TO\_CHAR(res.ENDDATE, 'DD-MON-YYYY') ||

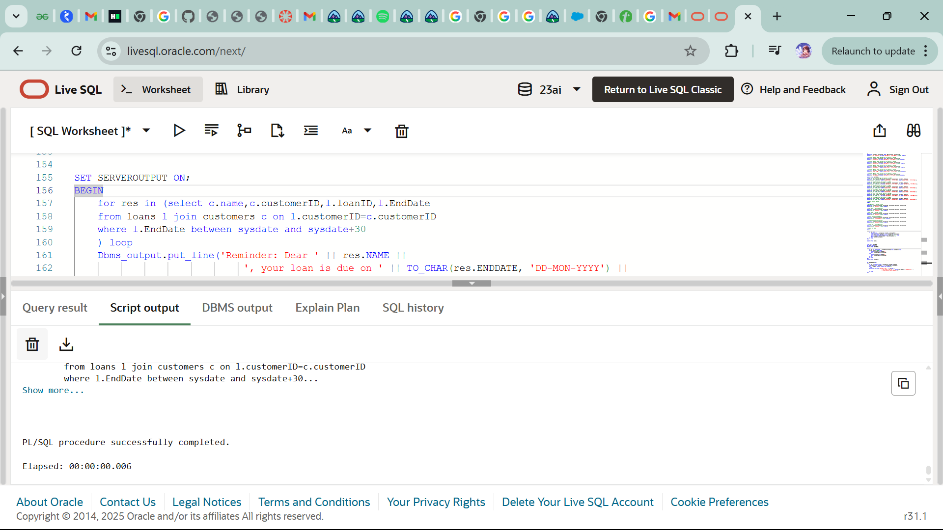
                             '. Please ensure timely payment.');

    end loop;

END;

**OUTPUT**

--procedure is created and executed:



**Exercise 3: Stored Procedures**

**Scenario 1:**

**INPUT**

create or replace procedure ProcessMonthlyInterest is

BEGIN

    update accounts set balance=(0.01\*balance)+balance;

END;

--procedure calling

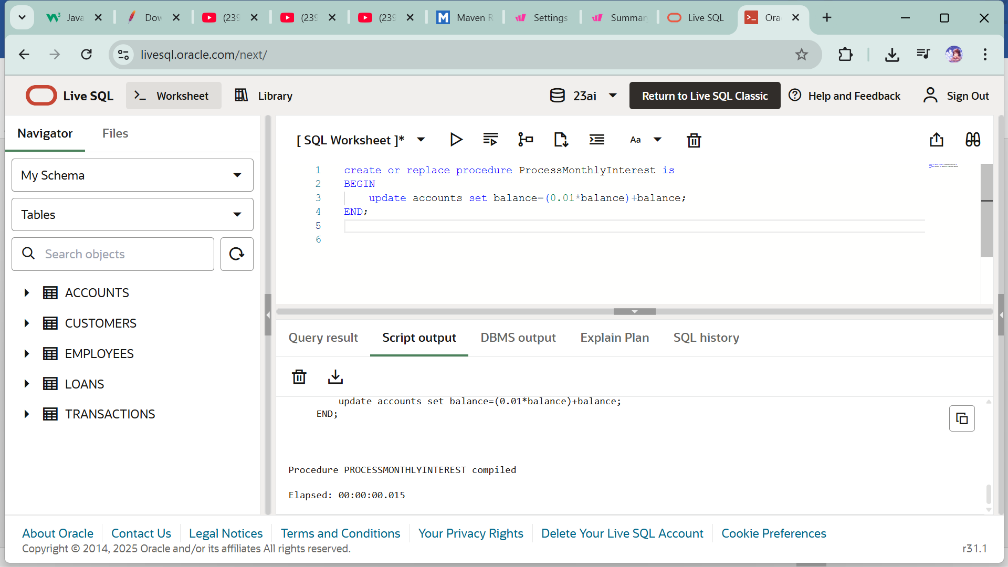
BEGIN

    ProcessMonthlyInterest;

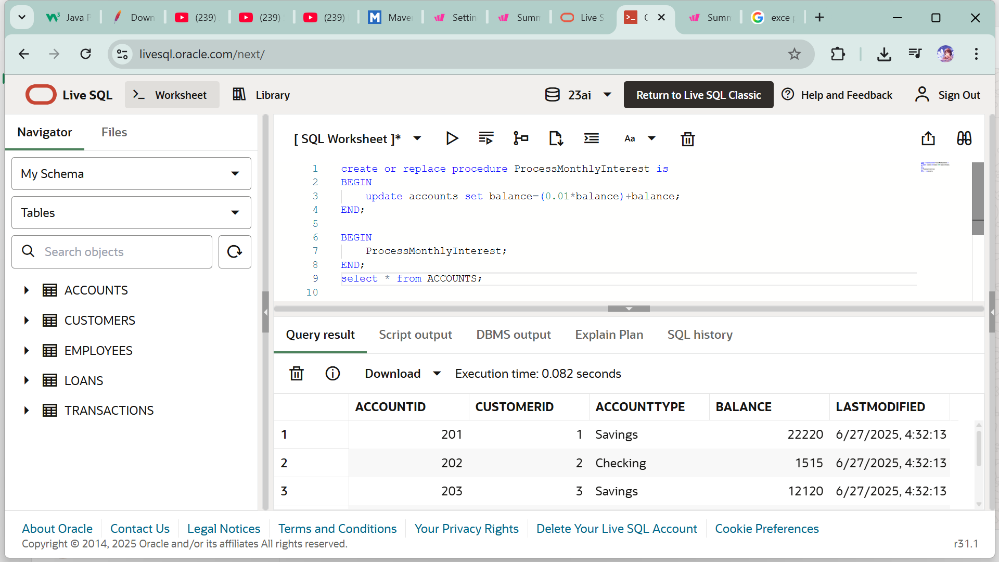
END;

**OUTPUT**

--creating and exceuting the procedure:



--after exceuting the procedure our ACCOUNTS table:



**Scenario 2:**

**INPUT**

create or replace procedure UpdateEmployeeBonus(Employee\_Department IN varchar2, bonus\_percentage IN number) is

BEGIN

    update employees set salary= salary + bonus\_percentage

    where department=Employee\_Department;

END;

--procedure calling

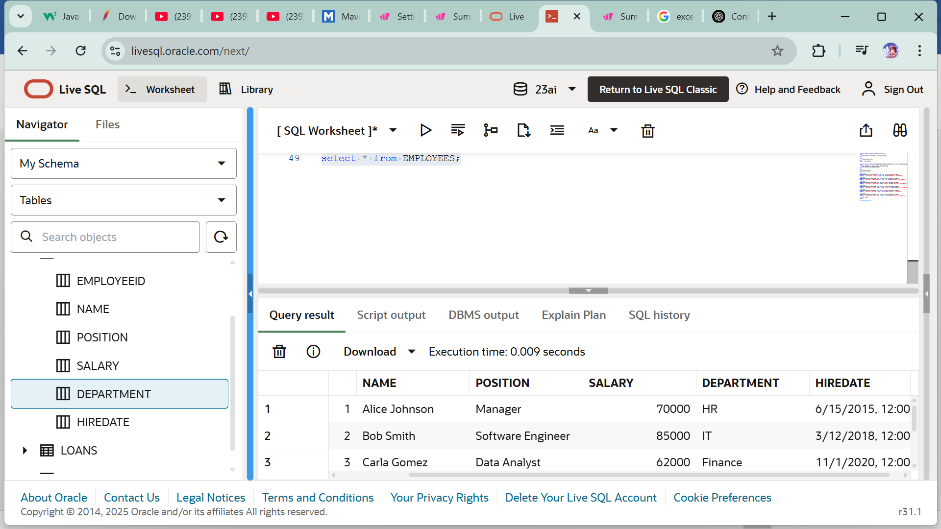
BEGIN

    UpdateEmployeeBonus('HR',500);

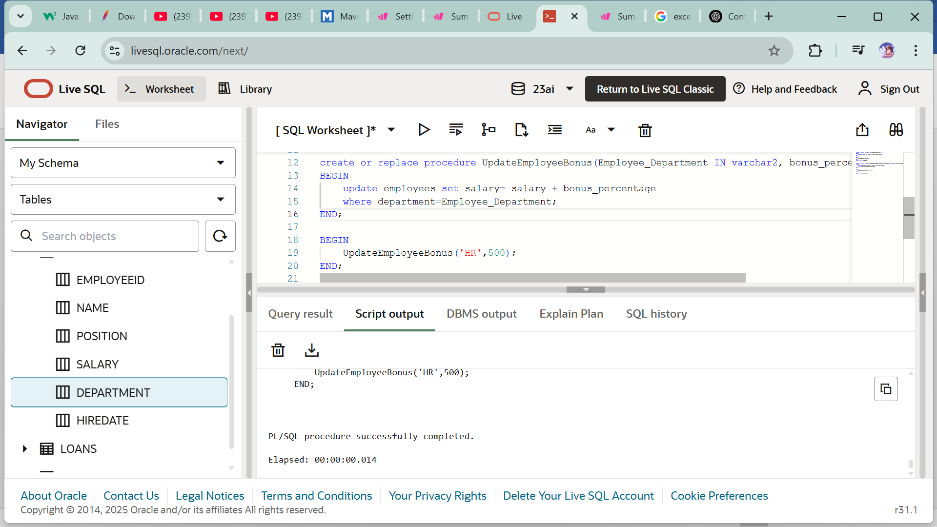
END;

**OUTPUT**

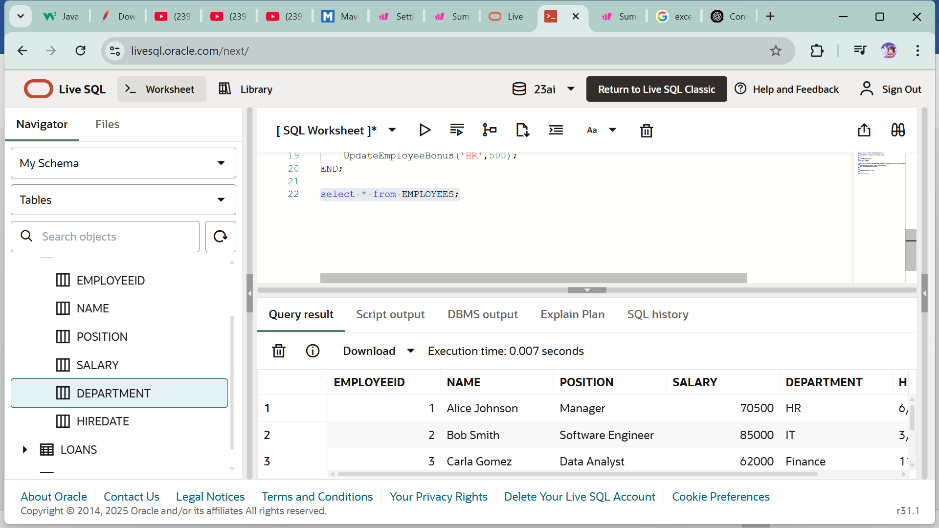
--before exceuting the procedure our EMPLOYEES table:



--creating and executing procedure:



--after exceuting the procedure our EMPLOYEES table:



**Scenario 3:**

create or replace procedure TransferFunds(from\_account IN number, to\_account IN number, amount IN number) IS

v\_balance number;

BEGIN

    select balance into v\_balance from accounts where ACCOUNTID=from\_account for update;

    if v\_balance < amount then

        raise\_application\_error(-20001,'Insufficient balance of source account');

    end if;

        update accounts set balance = balance-amount,lastmodified=sysdate where accountID=from\_account;

        update accounts set balance = balance+amount,lastmodified=sysdate where accountID=to\_account;

END;

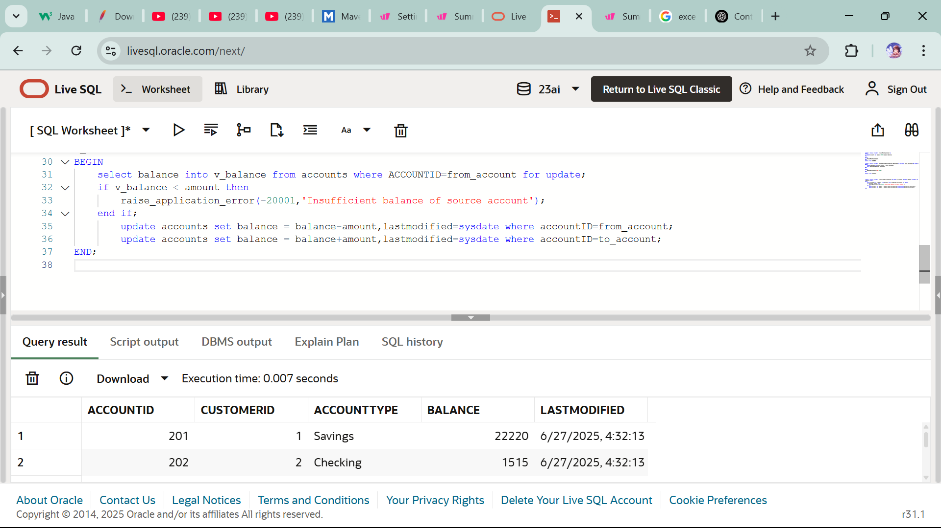
BEGIN

    TransferFunds(201,202,220);

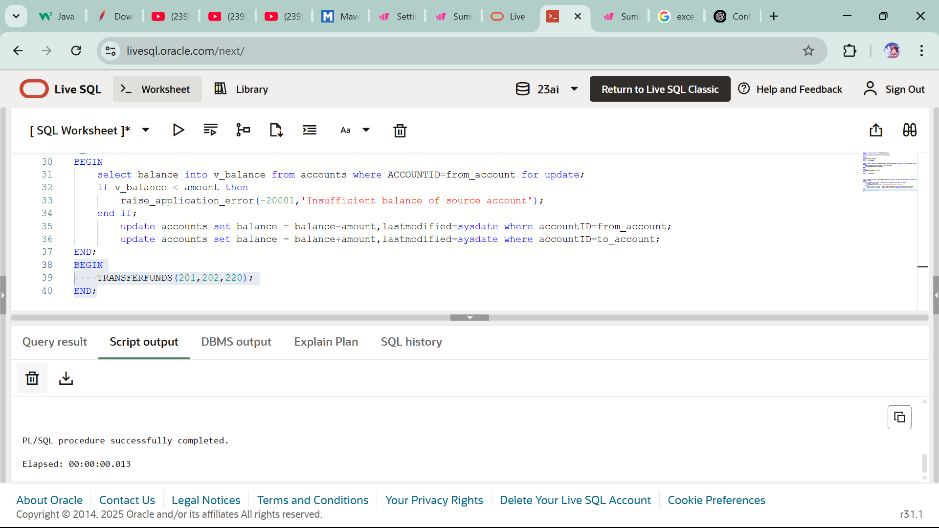
END;

**OUTPUT**

--before exceuting the procedure our ACCOUNTS table:



--procedure is created and executed



--after exceuting the procedure our ACCOUNTS table:

