**STORED PROCEDURE.**

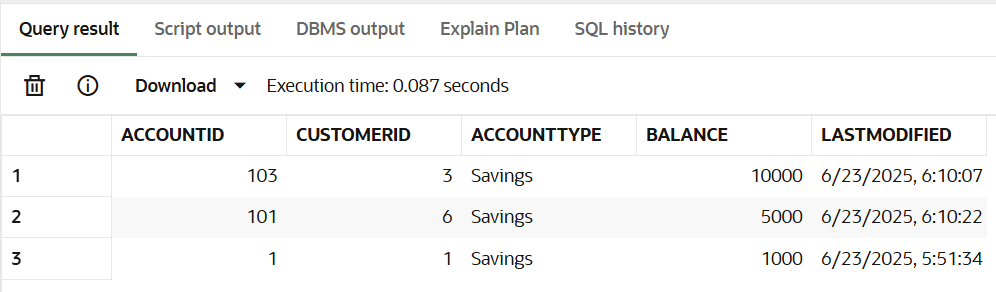
**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

* + 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.

**SOLUTION:**

**Query:** select \* from accounts WHERE accounttype='Savings';

**Output:**



**Query:**

create or replace procedure ProcessMonthlyInterest

as

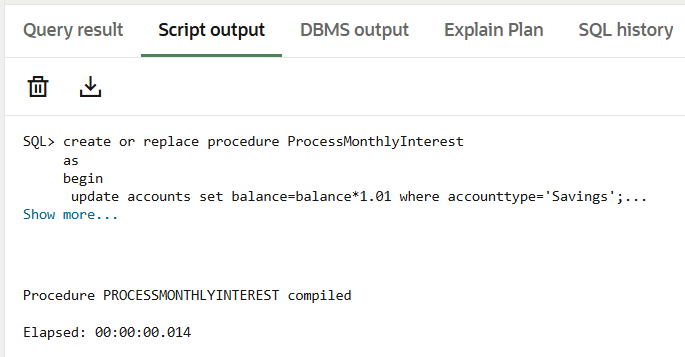
begin

    update accounts set balance=balance\*1.01 where accounttype='Savings';

    dbms\_output.put\_line('UPDATED ALL SAVINGS ACCOUNT..');

end;

**Output:**



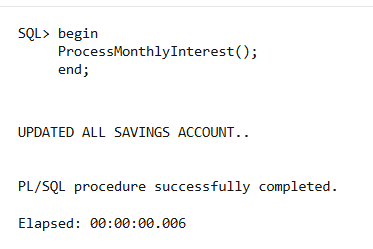
**Query:**

begin

    ProcessMonthlyInterest();

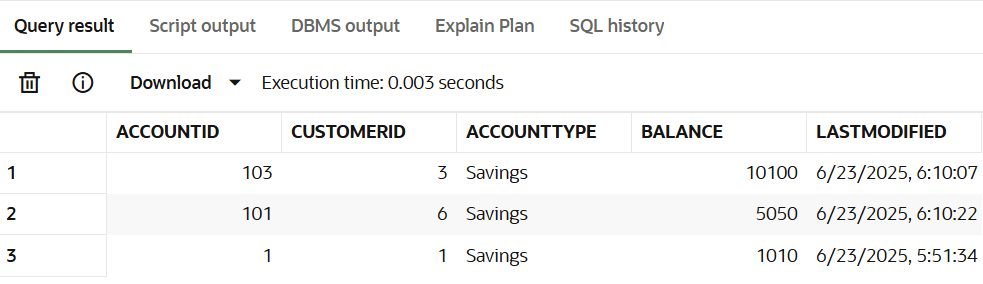
end;

**Output:**



**Query:** select \* from accounts where accounttype='Savings';

**Output:**



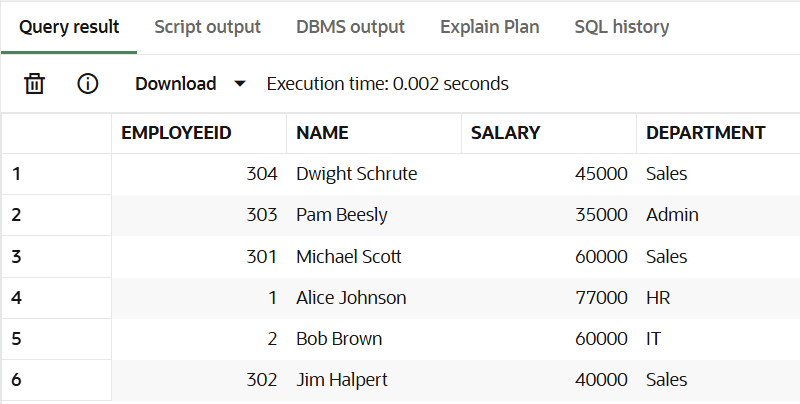
**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

* + 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.

**SOLUTION:**

**Query:** select Employeeid,name,salary,department from employee;

**Output:**



**Query:**

create or replace procedure UpdateEmployeeBonus(dept in varchar,percentage in number)

as

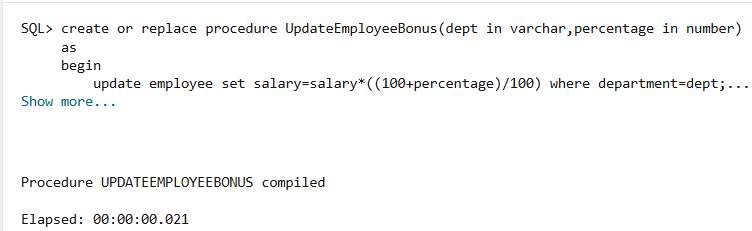
begin

    update employee set salary=salary\*((100+percentage)/100) where department=dept;

    dbms\_output.put\_line('SALARY UPDATED WITH BONUS FOR '||dept||' DEPARTMENT!!!');

end;

**Output:**



**Query:**

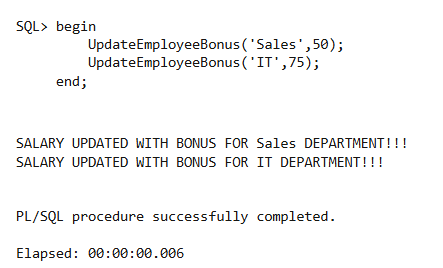
begin

    UpdateEmployeeBonus('Sales',50);

    UpdateEmployeeBonus('IT',75);

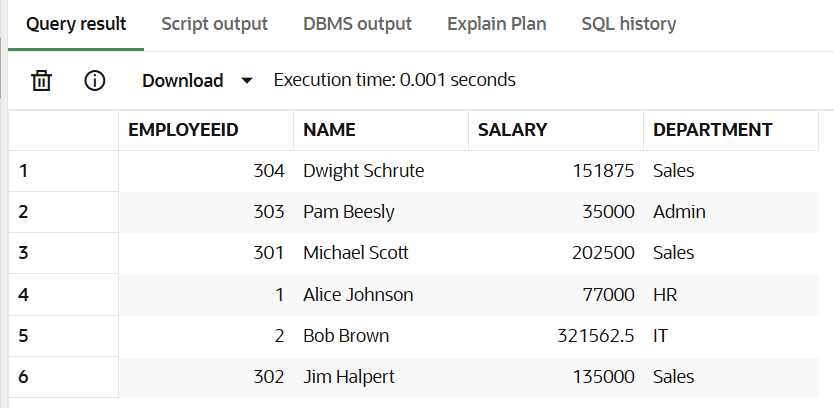
end;

**Output:**



**Query:** select Employeeid,name,salary,department from employee;

**Output:**



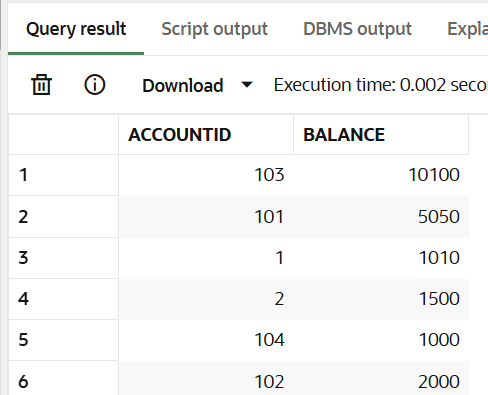
**Scenario 3:** Customers should be able to transfer funds between their accounts.

* + 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.

**SOLUTION:**

**Query:** select accountid,balance from accounts;

**Output:**



**Query:**

create or replace procedure TransferFunds(

sourceid in accounts.accountid%type,

receiverid in accounts.accountid%type,

amount number)

as

checkbalance number;

begin

    select balance into checkbalance from accounts where accountid=sourceid;

    if checkbalance>amount then

    update accounts set balance=balance-amount where accountid=sourceid;

    update accounts set balance=balance+amount where accountid=receiverid;

    dbms\_output.put\_line('TRANSACTION COMPLETED');

    dbms\_output.put\_line('SENDERID: '||sourceid);

    dbms\_output.put\_line('RECEIVERID: '||receiverid);

    dbms\_output.put\_line('AMOUNT: '||amount);

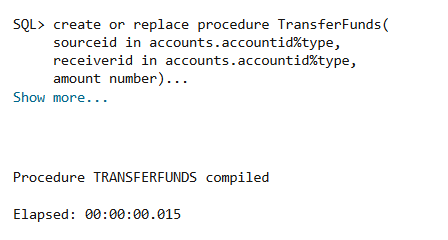
    else

    dbms\_output.put\_line('NO SUFFICIENT BALANCE IN SOURCE ACCOUNT WITH ID:'||sourceid);

    end if;

end;

**Output:**



**Query:**

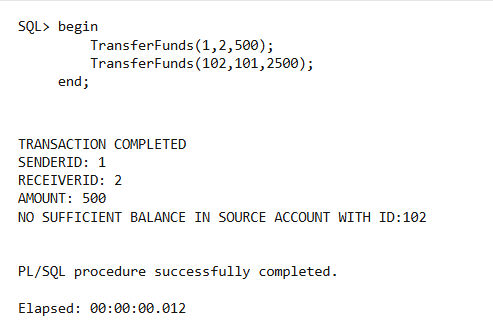
begin

    TransferFunds(1,2,500);

    TransferFunds(102,101,2500);

end;

**Output:**



**Query:** select accountid,balance from accounts;

**Output:**

