Stored Procedures

Program:

**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.

Procedure:

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

    UPDATE Bank

    SET amount = amount + (amount \* 0.01)

    WHERE account\_type = 'saving';

    COMMIT;

END;

/

Running it:

BEGIN

    processMonthlyInterest;

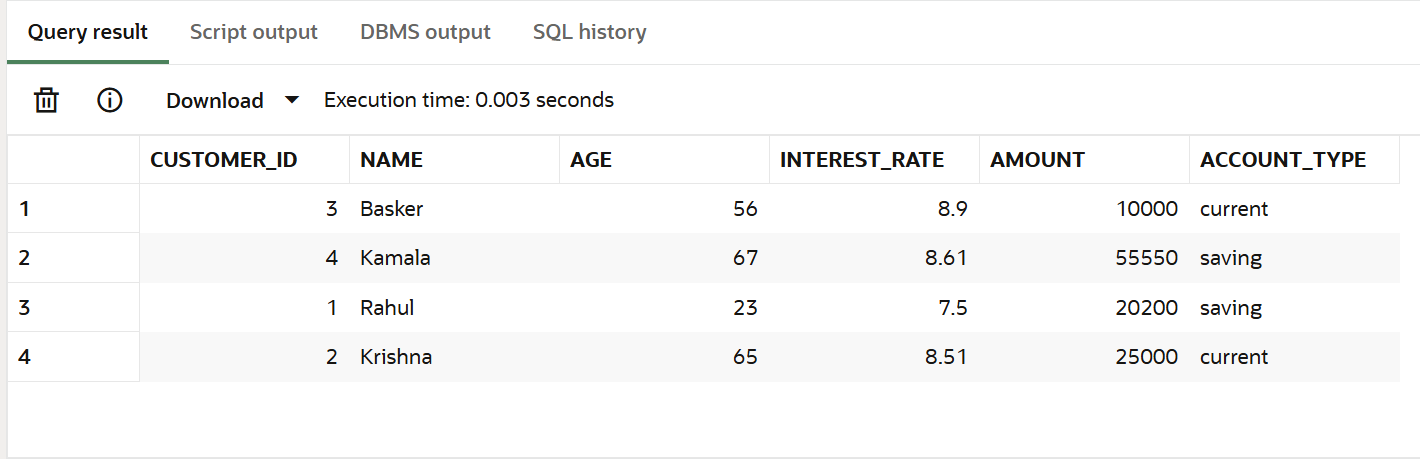
END;

/

Table:

SELECT \*

FROM Bank;



**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.

**Bonus.**sql:

CREATE OR REPLACE PROCEDURE Bonus(

    n\_id in number,

    n\_bonus in number

)IS

BEGIN

    UPDATE Employee

    SET salary = salary + (salary \* (n\_bonus/100))

    WHERE id = n\_id;

    COMMIT;

END;

/

Run.sql

BEGIN

Bonus(102 , 3);

END;

/

Table before :

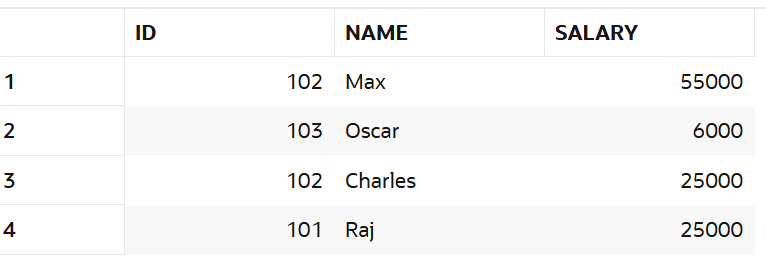
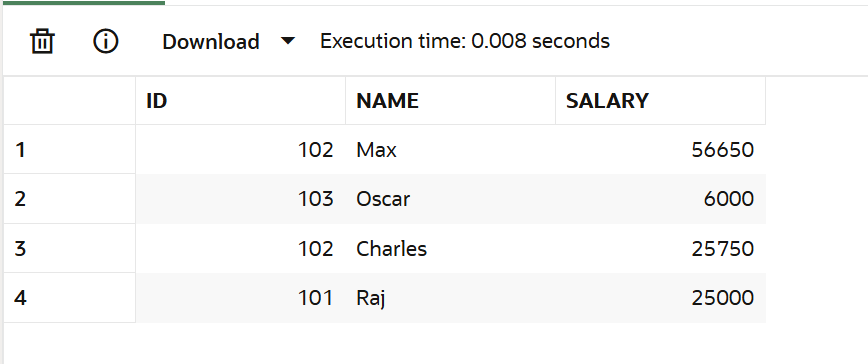


Table After:



Scenario 3:

Customers should be able to transfer funds between their accounts. 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.

Program:

CREATE OR REPLACE PROCEDURE TRANSFERFUND(

    n\_account\_number IN number,

    n\_account\_to IN NUMBER

)IS

BEGIN

    UPDATE ACCOUNT

    SET AMOUNT = AMOUNT + (SELECT AMOUNT FROM ACCOUNT WHERE ACCOUNT.ACCOUNT\_NUMBER = n\_account\_number AND amount > 0 )

    WHERE account\_number = n\_account\_to;

    COMMIT;

END;

/

Table Before:

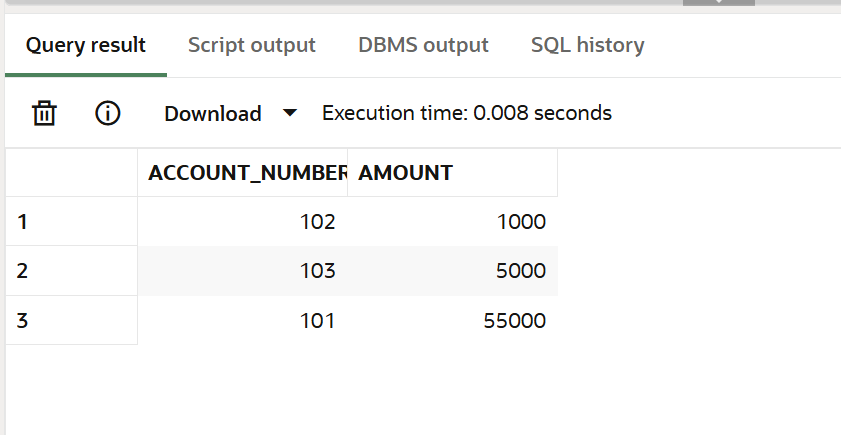


Table After:

