**PL/SQL EXERCISES**

**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.

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

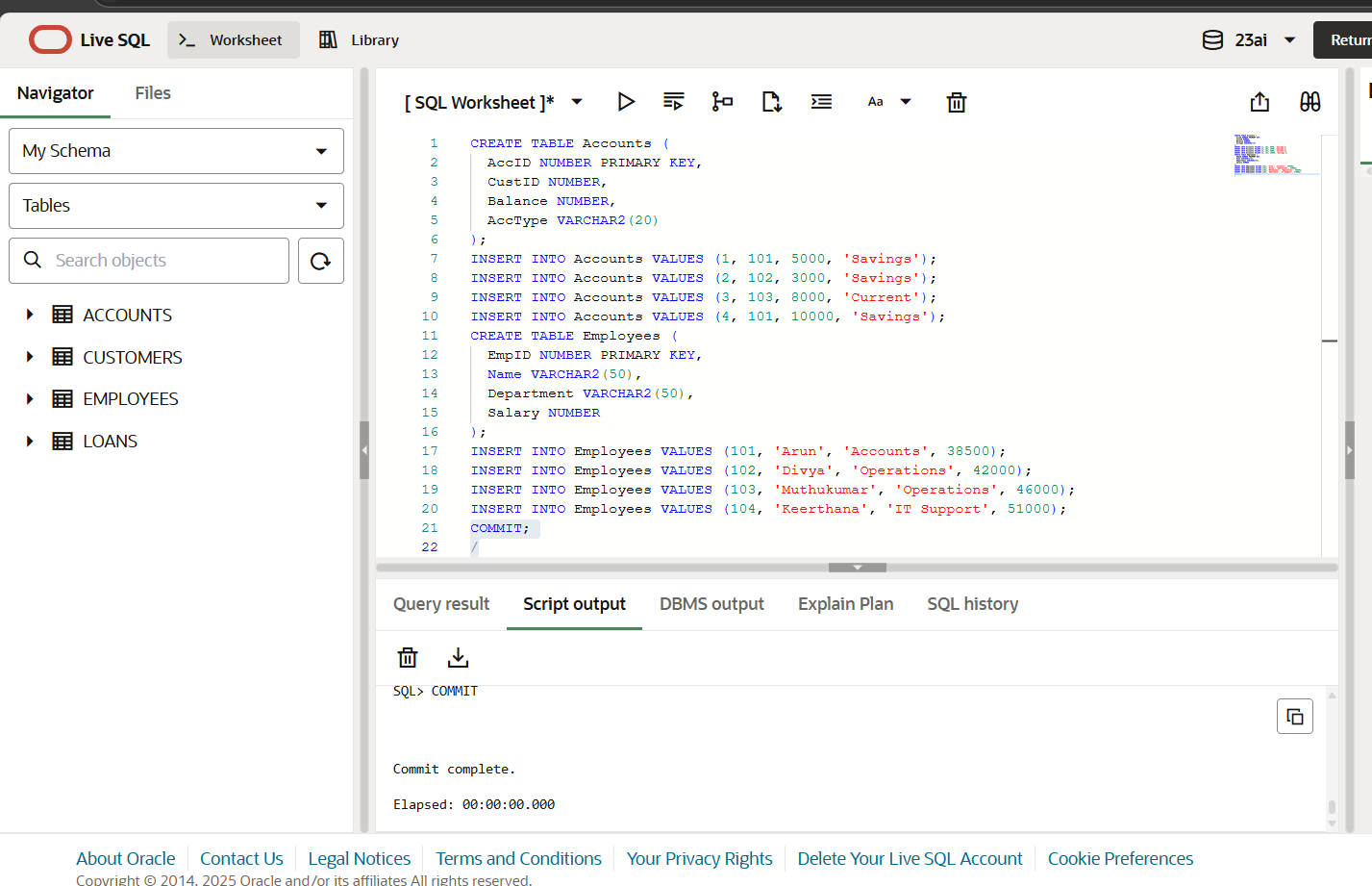
* + **Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

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

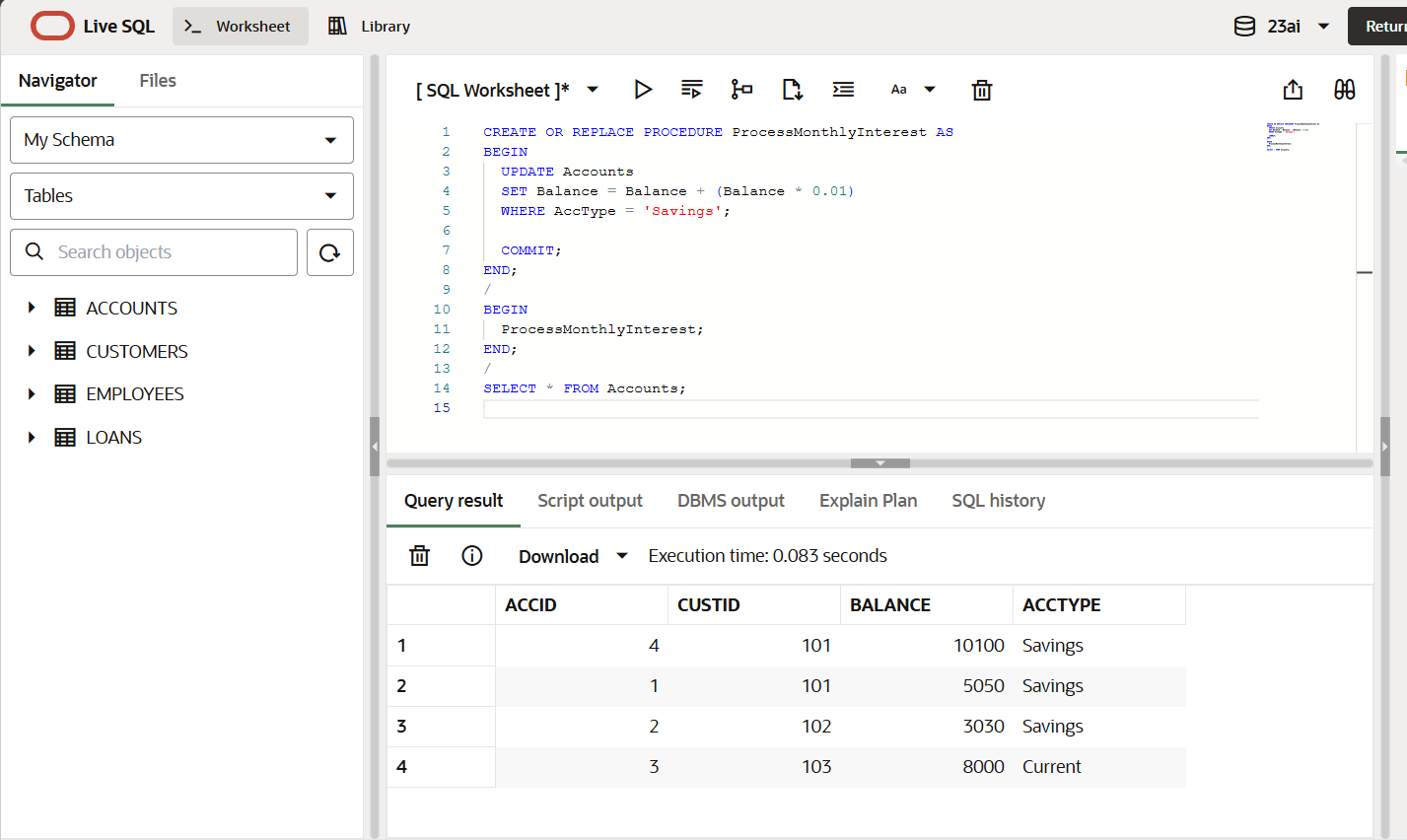
* + **Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

**IMPLEMENTATION IN LIVE SQL :**

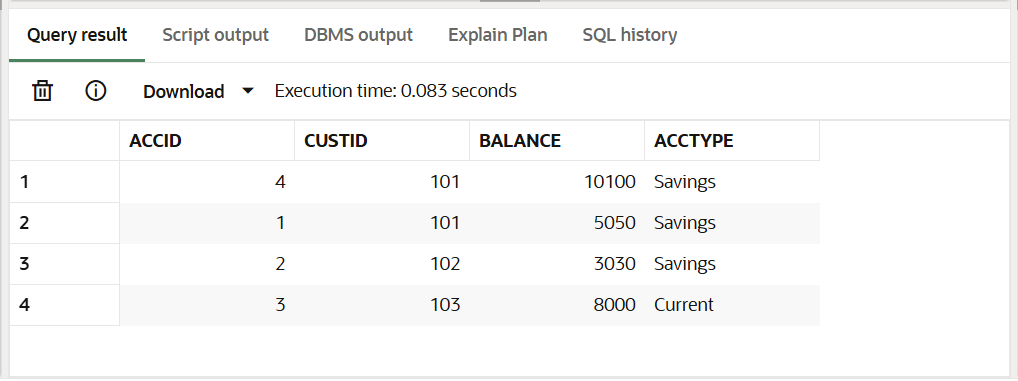
**Creation of Tables and Data stored in Database:**

****

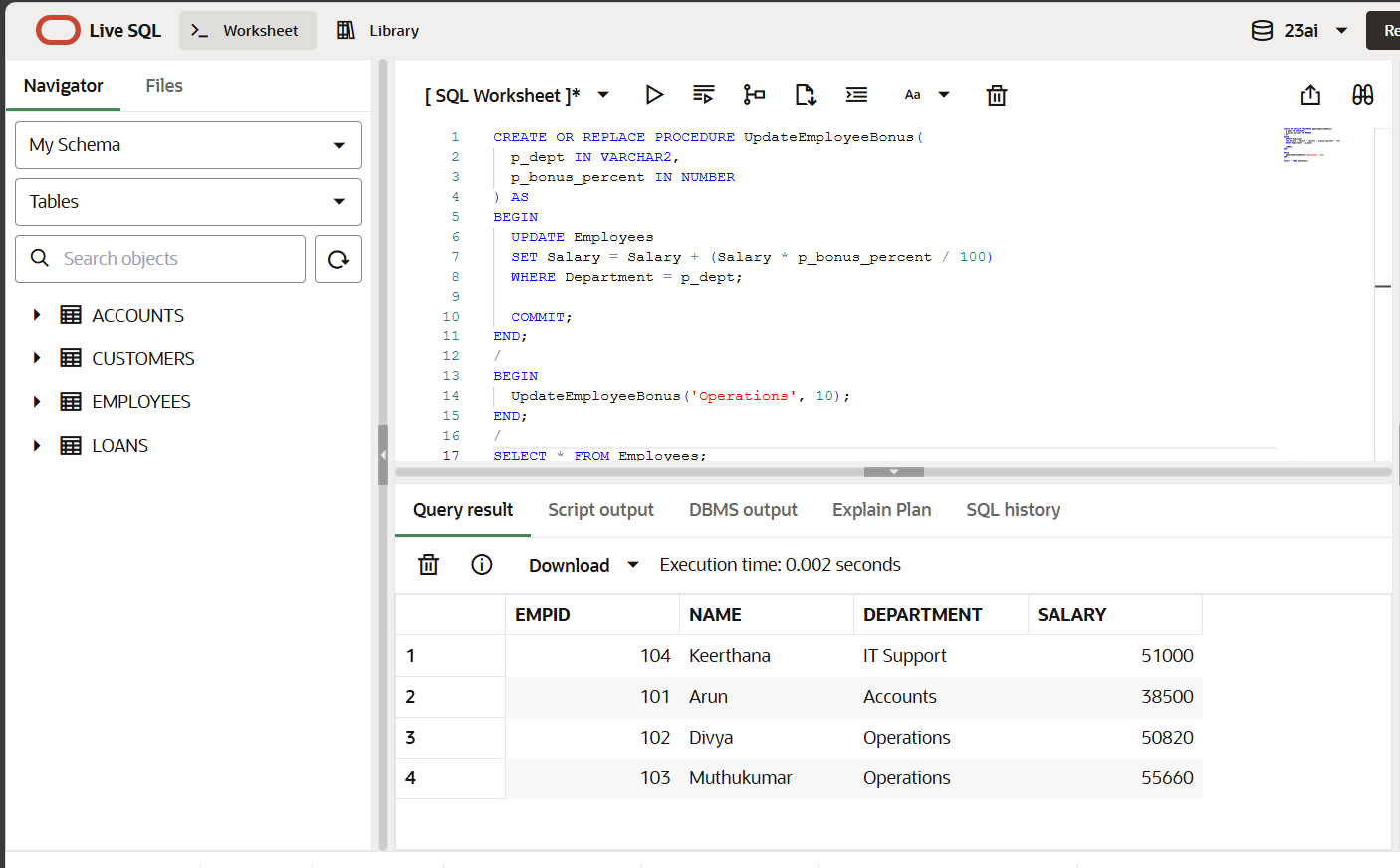
**Scenario 1 Implementation:**

****

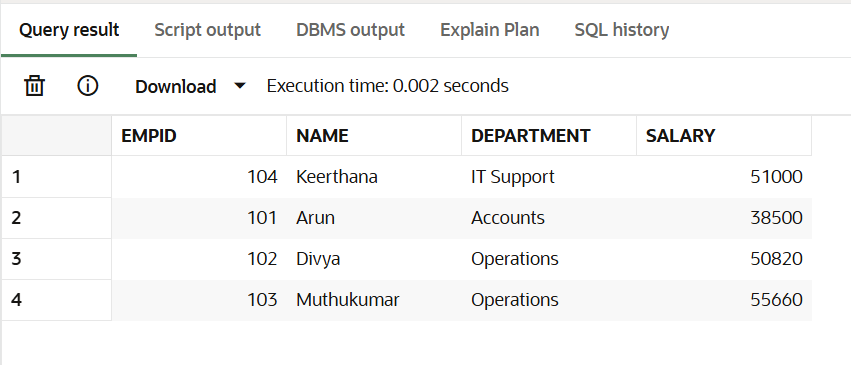
**Scenario 1 Output:**

****

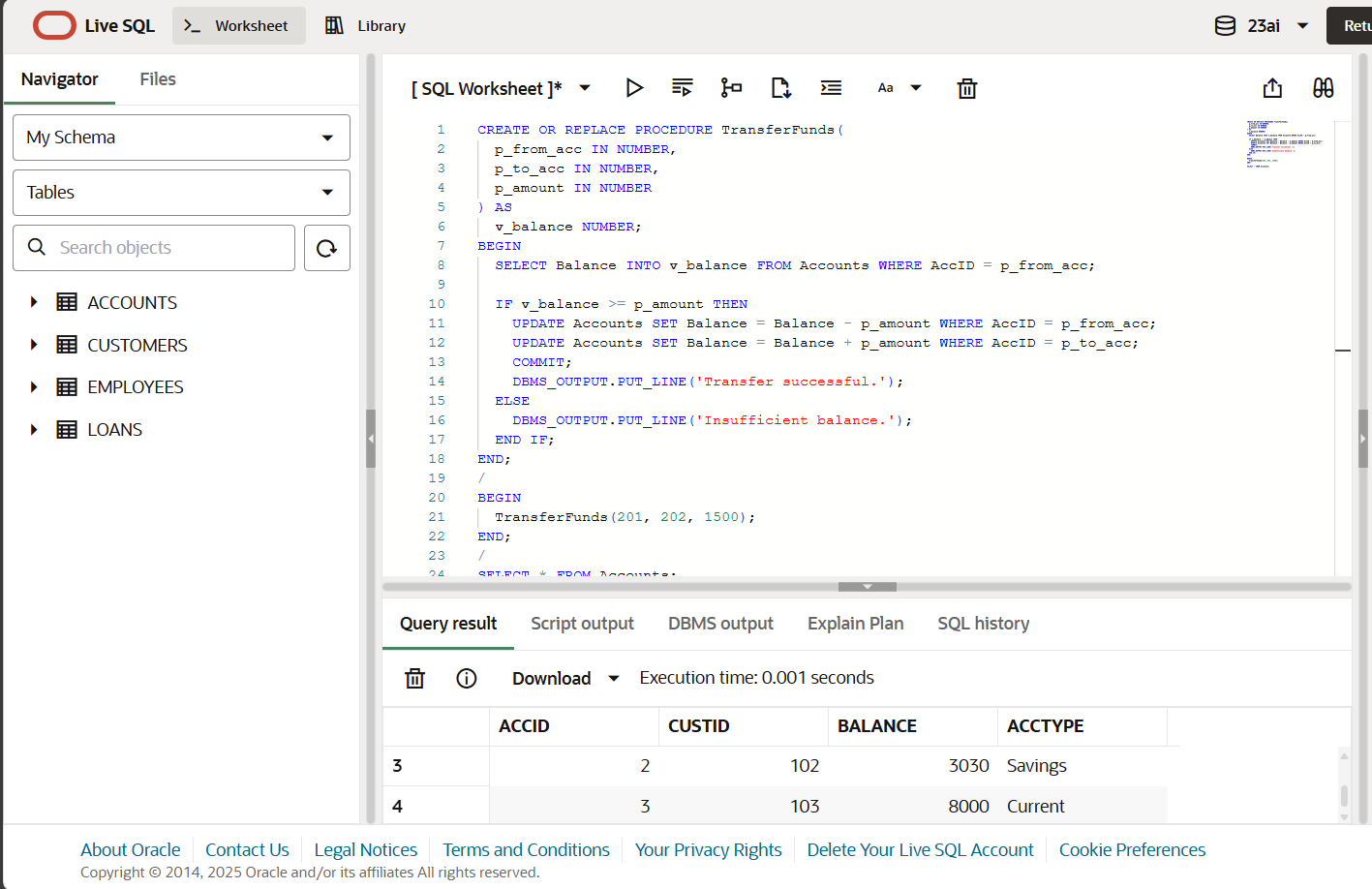
**Scenario 2 Implementation:**

****

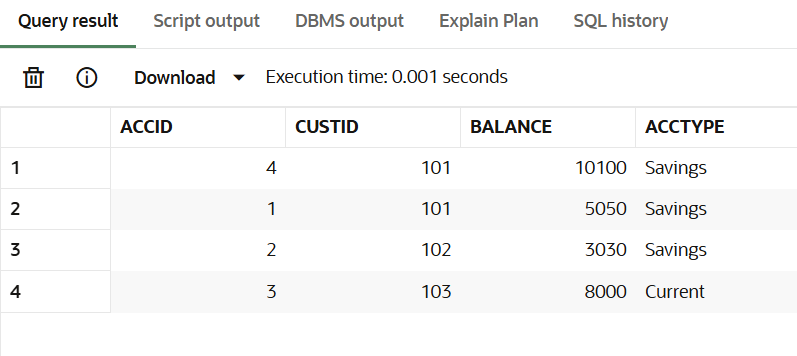
**Scenario 2 Output:**

****

**Scenario 3 Implementation:**

****

**Scenario 3 Output:**

****

**PROGRAM :**

**Creation of tables and sample data :**

CREATE TABLE Accounts (

AccID NUMBER PRIMARY KEY,

CustID NUMBER,

Balance NUMBER,

AccType VARCHAR2(20)

);

INSERT INTO Accounts VALUES (201, 5001, 8450, 'Savings');

INSERT INTO Accounts VALUES (202, 5002, 6200, 'Savings');

INSERT INTO Accounts VALUES (203, 5003, 10250, 'Current');

INSERT INTO Accounts VALUES (204, 5001, 9700, 'Savings');

CREATE TABLE Employees (

EmpID NUMBER PRIMARY KEY,

Name VARCHAR2(50),

Department VARCHAR2(50),

Salary NUMBER

);

INSERT INTO Employees VALUES (101, 'Arun Prakash', 'Accounts', 38500);

INSERT INTO Employees VALUES (102, 'Divya Lakshmi', 'Operations', 42000);

INSERT INTO Employees VALUES (103, 'Muthukumar', 'Operations', 46000);

INSERT INTO Employees VALUES (104, 'Keerthana R.', 'IT Support', 51000);

COMMIT;

/

**Scenario 1 :**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01)

WHERE AccType = 'Savings';

COMMIT;

END;

/

BEGIN

ProcessMonthlyInterest;

END;

/

SELECT \* FROM Accounts;

**Scenario 2 :**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

p\_dept IN VARCHAR2,

p\_bonus\_percent IN NUMBER

) AS

BEGIN

UPDATE Employees

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

WHERE Department = p\_dept;

COMMIT;

END;

/

BEGIN

UpdateEmployeeBonus('Operations', 8);

END;

/

SELECT \* FROM Employees;

**Scenario 3 :**

CREATE OR REPLACE PROCEDURE TransferFunds(

  p\_from\_acc IN NUMBER,

  p\_to\_acc IN NUMBER,

  p\_amount IN NUMBER

) AS

  v\_balance NUMBER;

BEGIN

  SELECT Balance INTO v\_balance FROM Accounts WHERE AccID = p\_from\_acc;

  IF v\_balance >= p\_amount THEN

    UPDATE Accounts SET Balance = Balance - p\_amount WHERE AccID = p\_from\_acc;

    UPDATE Accounts SET Balance = Balance + p\_amount WHERE AccID = p\_to\_acc;

    COMMIT;

    DBMS\_OUTPUT.PUT\_LINE('Transfer successful.');

  ELSE

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

  END IF;

END;

/

BEGIN

  TransferFunds(201, 202, 1500);

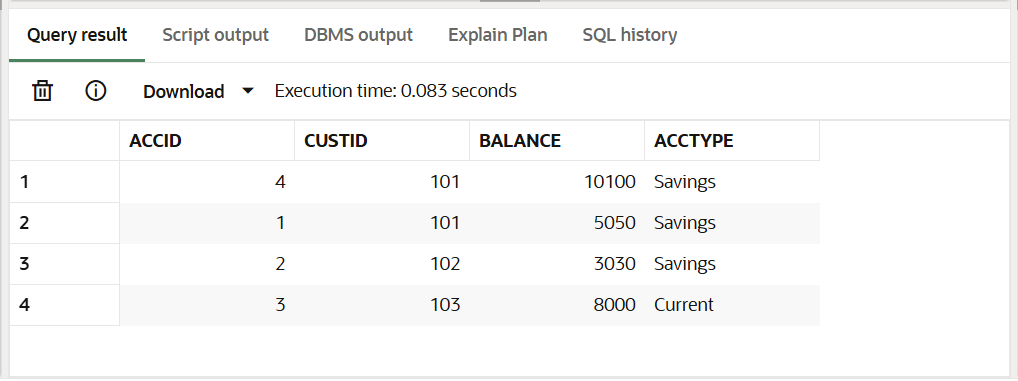
END;

/

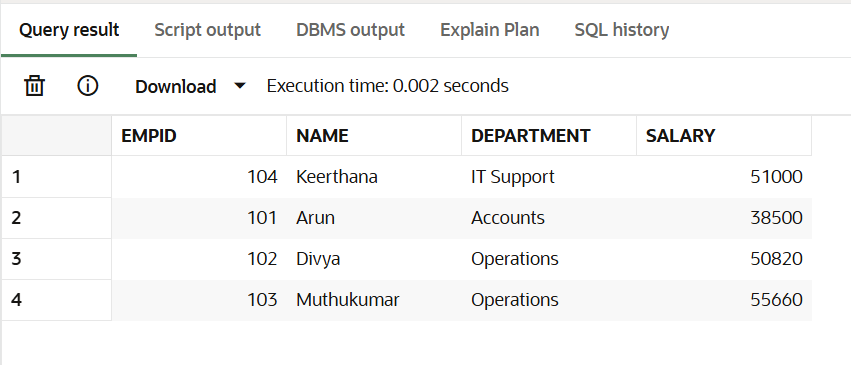
SELECT \* FROM Accounts;

**OUTPUT :**

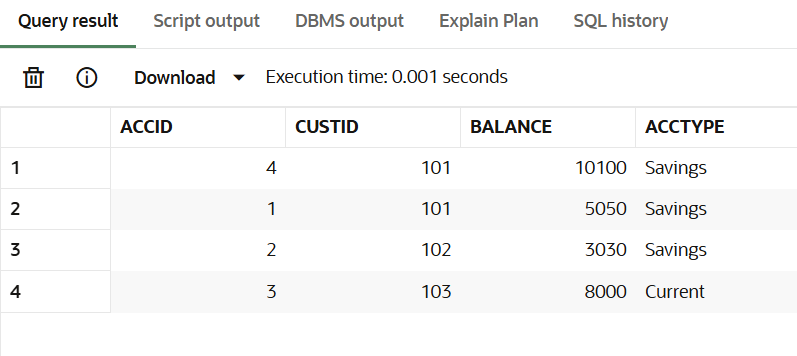
**Scenario 1:**

****

**Scenario 2 :**

****

**Scenario 3 :**

****