**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.

**CODE:**

CREATE TABLE accounts (

    account\_id NUMBER PRIMARY KEY,

    account\_holder VARCHAR2(100),

    account\_type VARCHAR2(20), -- 'Savings', 'Checking', etc.

    balance NUMBER(12, 2)

);

-- Sample accounts

INSERT INTO accounts VALUES (101, 'Alice', 'Savings', 5000);

INSERT INTO accounts VALUES (102, 'Bob', 'Checking', 3000);

INSERT INTO accounts VALUES (103, 'Charlie', 'Savings', 10000);

INSERT INTO accounts VALUES (104, 'Naveen', 'Savings', 1000);

COMMIT;

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

    UPDATE accounts

    SET balance = balance / 1.01

    WHERE account\_type = 'Savings';

    COMMIT;

END;

/

-- Process interest

BEGIN

    ProcessMonthlyInterest;

END;

/

--To Display the result

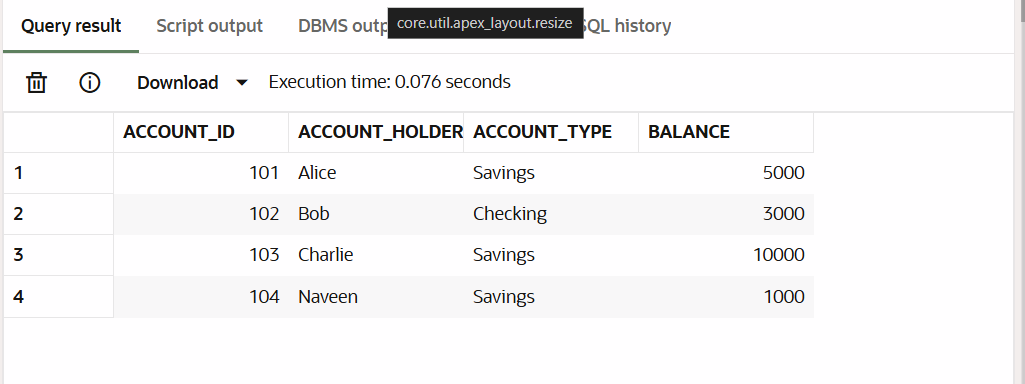
SELECT account\_id, account\_holder, account\_type, balance

FROM accounts

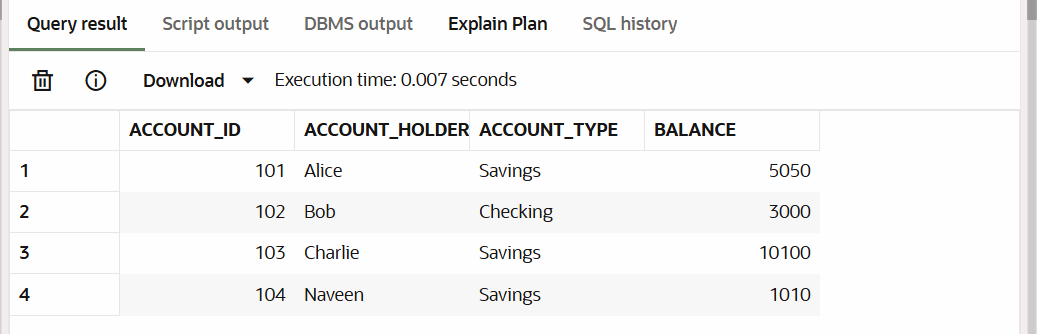
ORDER BY account\_id;

**OUTPUT: (SCREENSHOTS)**

* **Before Update -**

****

* **After Update –**

****

**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.

**CODE:**

CREATE TABLE employees (

    employee\_id NUMBER PRIMARY KEY,

    name VARCHAR2(100),

    department\_id NUMBER,

    salary NUMBER(10, 2)

);

-- Sample employees

INSERT INTO employees VALUES (1, 'David', 10, 40000);

INSERT INTO employees VALUES (2, 'Eva', 10, 42000);

INSERT INTO employees VALUES (3, 'Frank', 20, 39000);

COMMIT;

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

    p\_dept\_id IN NUMBER,

    p\_bonus\_percent IN NUMBER

) IS

BEGIN

    UPDATE employees

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

    WHERE department\_id = p\_dept\_id;

    COMMIT;

END;

/

-- Give 10% bonus to dept 10

BEGIN

    UpdateEmployeeBonus(10, 10);

END;

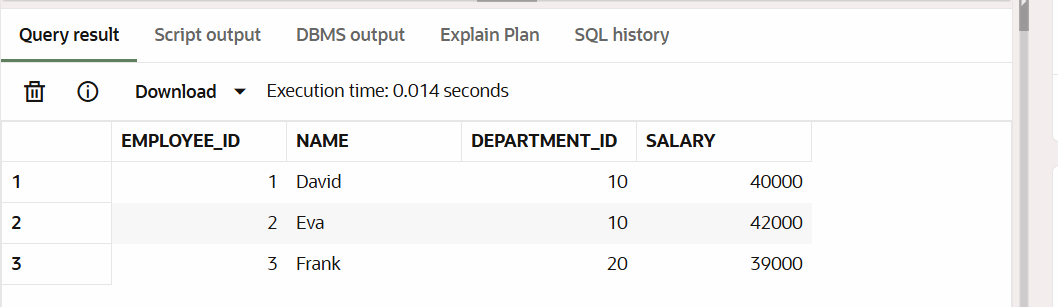
/

--To display the table

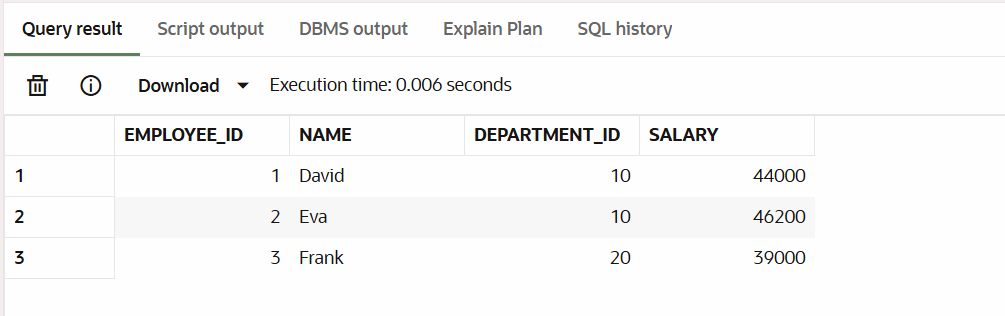
SELECT \* FROM EMPLOYEES;

**OUTPUT: (SCREENSHOTS)**

* **Before Update –**

****

* **After Update –**

****

**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:**

CREATE OR REPLACE PROCEDURE TransferFunds(

    p\_from\_account IN NUMBER,

    p\_to\_account IN NUMBER,

    p\_amount IN NUMBER

) IS

    v\_balance NUMBER;

BEGIN

    -- Lock the row and get balance

    SELECT balance INTO v\_balance

    FROM accounts

    WHERE account\_id = p\_from\_account

    FOR UPDATE;

    IF v\_balance < p\_amount THEN

        RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance');

    END IF;

    -- Debit source

    UPDATE accounts

    SET balance = balance - p\_amount

    WHERE account\_id = p\_from\_account;

    -- Credit destination

    UPDATE accounts

    SET balance = balance + p\_amount

    WHERE account\_id = p\_to\_account;

    COMMIT;

END;

-- Transfer ₹500 from account 101 to 102

BEGIN

    TransferFunds(101, 102, 500);

END;

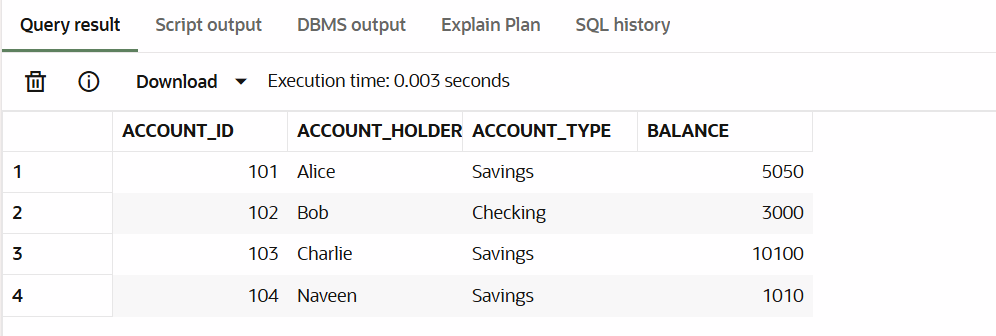
/

--To Display the table

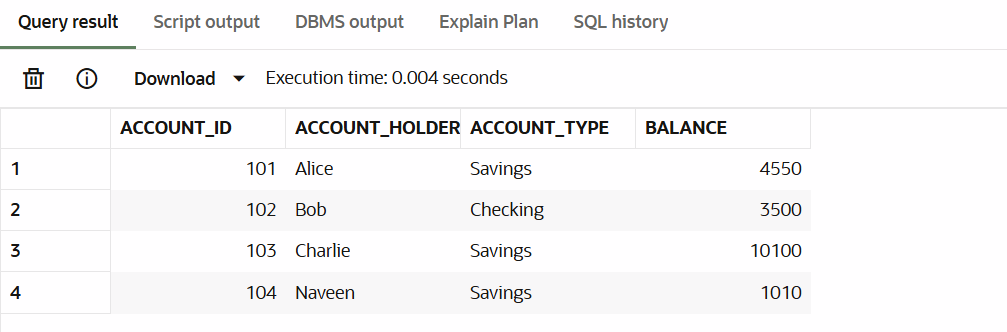
SELECT \* FROM ACCOUNTS;

**OUTPUT: (SCREENSHOTS)**

* **Before Update –**

****

* **After Update –**

****