**ERROR HANDLING.**

**Scenario 1:** Handle exceptions during fund transfers between accounts.

* + Question: Write a stored procedure SafeTransferFunds that transfers funds between two accounts. Ensure that if any error occurs (e.g., insufficient funds), an appropriate error message is logged and the transaction is rolled back.

**SOLUTION:**

**Query:**

CREATE TABLE ErrorLogs (

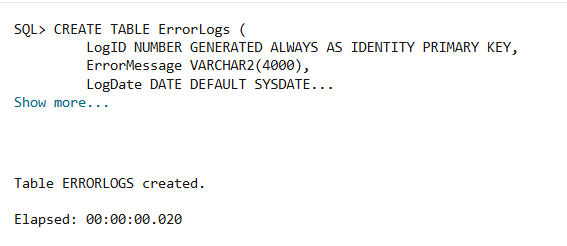
    LogID NUMBER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,

    ErrorMessage VARCHAR2(4000),

    LogDate DATE DEFAULT SYSDATE

);

**Output:**



**Query:**

CREATE OR REPLACE PROCEDURE SafeTransferFunds (

    p\_FromAccountID NUMBER,

    p\_ToAccountID NUMBER,

    p\_Amount NUMBER

) IS

    v\_FromBalance NUMBER;

BEGIN

    SAVEPOINT StartTransfer;

    SELECT Balance INTO v\_FromBalance FROM Accounts WHERE AccountID = p\_FromAccountID FOR UPDATE;

    IF v\_FromBalance < p\_Amount THEN

        RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient funds in source account.');

    END IF;

    UPDATE Accounts

    SET Balance = Balance - p\_Amount,

        LastModified = SYSDATE

    WHERE AccountID = p\_FromAccountID;

    UPDATE Accounts

    SET Balance = Balance + p\_Amount,

        LastModified = SYSDATE

    WHERE AccountID = p\_ToAccountID;

    INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

    VALUES (6, p\_FromAccountID, SYSDATE, p\_Amount, 'DEBIT');

    INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

    VALUES (4, p\_ToAccountID, SYSDATE, p\_Amount, 'CREDIT');

    COMMIT;

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        INSERT INTO ErrorLogs (ErrorMessage)

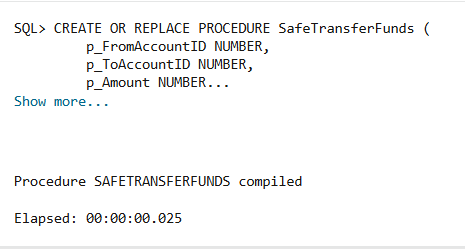
        VALUES ('TRANSACTION FAILED!!');

    WHEN OTHERS THEN

    ROLLBACK TO StartTransfer;

END;

**Output:**



**Query:**

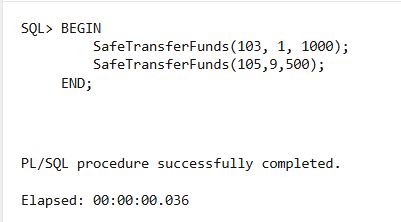
BEGIN

    SafeTransferFunds(103, 1, 1000);

    SafeTransferFunds(505,9,500);

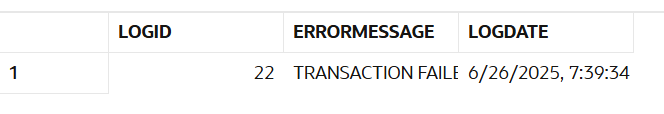
END;

**Output:**



**Query:** select \* from errorlogs;

**Output:**



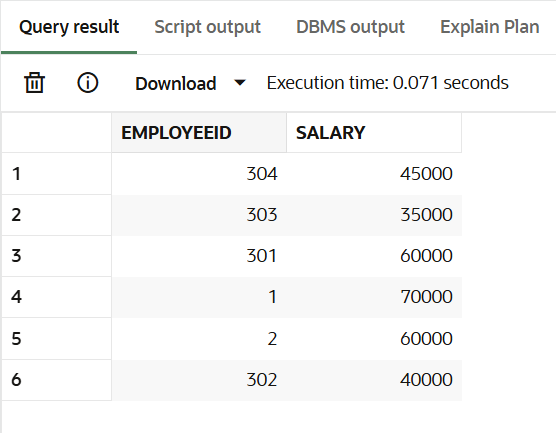
**Scenario 2**: Manage errors when updating employee salaries.

* + Question: Write a stored procedure UpdateSalary that increases the salary of an employee by a given percentage. If the employee ID does not exist, handle the exception and log an error message.

**SOLUTION:**

**Query:** select employeeid,salary from employee;

**Output:**



**Query:**

create or replace procedure updatesalary(e\_id IN number,percentage IN number)

AS

e\_salary number;

BEGIN

    select salary into e\_salary from employee where employeeid=e\_id;

    e\_salary:=e\_salary\*((100+percentage)/100);

    update employee set salary=e\_salary where employeeid=e\_id;

    dbms\_output.put\_line('THE SALARY OF EMPLOYEE '||e\_id||' IS UPDATED..');

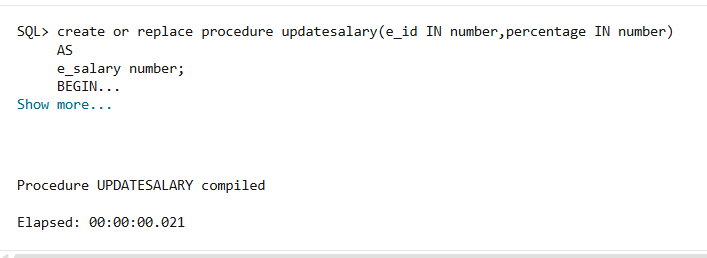
    EXCEPTION

    when no\_data\_found then

    dbms\_output.put\_line('THE GIVEN EMPLOYEEID IS NOT FOUND');

END;

**Output:**



**Query:**

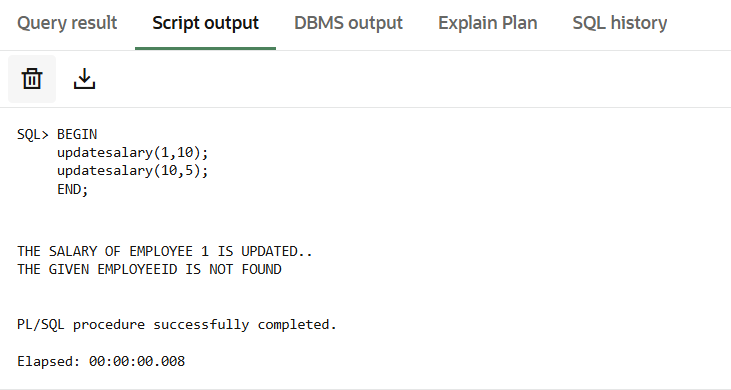
BEGIN

    updatesalary(1,10);

    updatesalary(10,5);

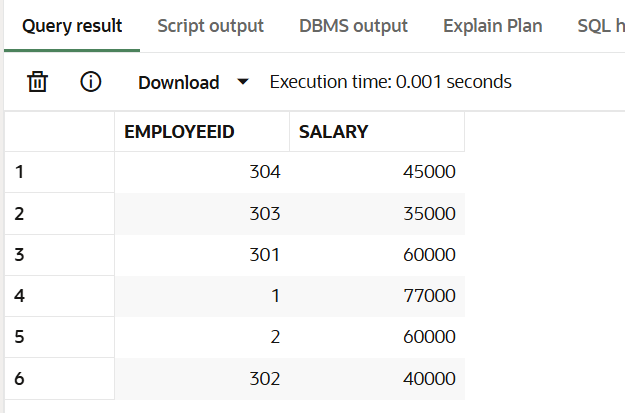
END;

**Output:**



**Query**: select employeeid,salary from employee where employeeid=10;

**Output:**



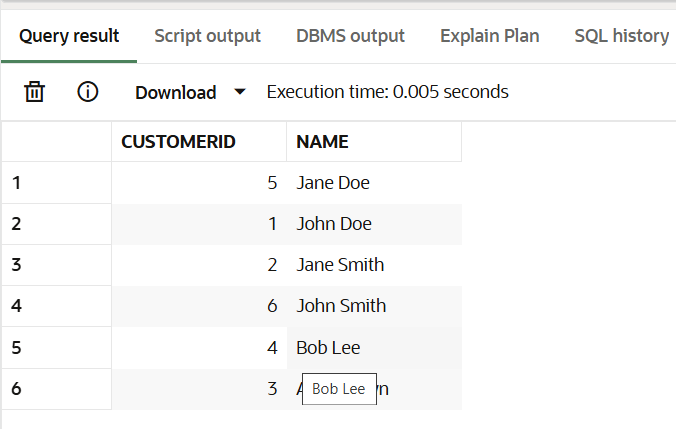
**Scenario 3**: Ensure data integrity when adding a new customer.

* + Question: Write a stored procedure AddNewCustomer that inserts a new customer into the Customers table. If a customer with the same ID already exists, handle the exception by logging an error and preventing the insertion.

**SOLUTION:**

**Query:** select customerid,name from customers;

**Output:**



**Query:**

create or replace procedure addcustomer(

    c\_id in customers.customerid%type,

    c\_name in customers.name%type,

    c\_dob in customers.dob%type,

    c\_balance in customers.balance%type,

    c\_lastmodify in customers.lastmodified%type

)

AS

BEGIN

    insert into customers(customerid,name,dob,balance,lastmodified) values(c\_id,c\_name,c\_dob,c\_balance,c\_lastmodify);

    dbms\_output.put\_line('ROW INSERTED SUCCESSFULLY');

    EXCEPTION

    WHEN DUP\_VAL\_ON\_INDEX THEN

    dbms\_output.put\_line('FALIED TO INSERT CUSID: '||c\_id||' AS IT ALREADY EXIST');

END;

**Output:**



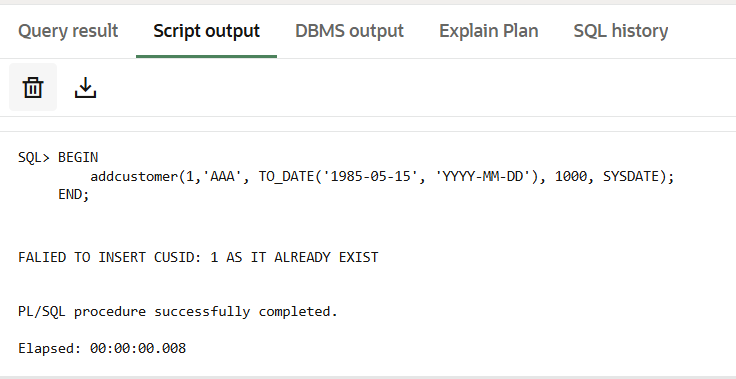
**Query:**

BEGIN

    addcustomer(1,'AAA', TO\_DATE('1985-05-15', 'YYYY-MM-DD'), 1000, SYSDATE);

END;

**Output:**



**Query:** select customerid,name from customers;

**Output:**

