**Error Handling**

**Scenario 1:**

CREATE OR REPLACE PROCEDURE SafeTransferFunds(

  from\_id IN NUMBER,

  to\_id   IN NUMBER,

  amt     IN NUMBER

) AS

  bal NUMBER;

BEGIN

  SELECT Balance INTO bal

  FROM Accounts

  WHERE AccountID = from\_id FOR UPDATE;

  IF bal < amt THEN

    RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient funds for transfer');

  END IF;

  UPDATE Accounts SET Balance = Balance - amt WHERE AccountID = from\_id;

  UPDATE Accounts SET Balance = Balance + amt WHERE AccountID = to\_id;

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

  COMMIT;

EXCEPTION

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('Transfer failed: ' || SQLERRM);

    ROLLBACK;

END;

/

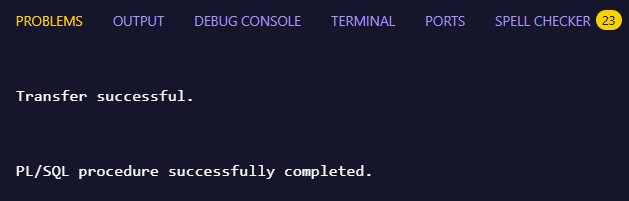
**Valid TestCase:**

BEGIN

  SafeTransferFunds(1,2, 500);

END;

/

**Output:**

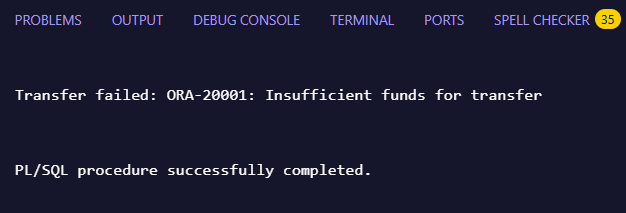
**Invalid TestCase:**

BEGIN

  SafeTransferFunds(1, 2, 99999);

END;

/

**Output:**

**Scenario 2:**

CREATE OR REPLACE PROCEDURE UpdateSalary(

  emp\_id IN NUMBER,

  pct    IN NUMBER

) AS

BEGIN

  UPDATE Employees

  SET Salary = Salary + (Salary \* pct / 100)

  WHERE EmployeeID = emp\_id;

  IF SQL%ROWCOUNT = 0 THEN

    RAISE\_APPLICATION\_ERROR(-20002, 'Employee ID not found.');

  END IF;

  DBMS\_OUTPUT.PUT\_LINE('Salary updated Successfully!');

  COMMIT;

EXCEPTION

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

    ROLLBACK;

END;

/

BEGIN

  UpdateSalary(1, 10);

END;

/

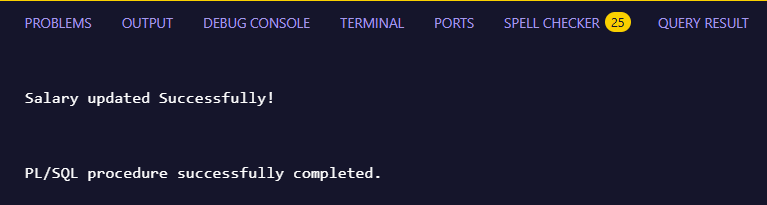
**Valid TestCase:**

BEGIN

  UpdateSalary(999, 15);

END;

/

**Output:**

**Invalid TestCase:**

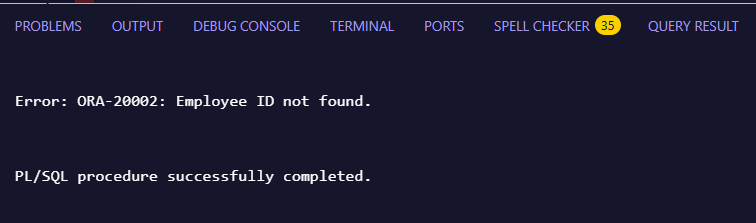
BEGIN

  UpdateSalary(999, 15);

END;

/

**Output:**

****

**Scenario 3:**

CREATE OR REPLACE PROCEDURE AddNewCustomer(

  cust\_id IN NUMBER,

  cust\_name IN VARCHAR2,

  dob IN DATE,

  bal IN NUMBER

) AS

BEGIN

  INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)

  VALUES (cust\_id, cust\_name, dob, bal, SYSDATE);

  DBMS\_OUTPUT.PUT\_LINE('New customer added Successfully!');

  COMMIT;

EXCEPTION

  WHEN DUP\_VAL\_ON\_INDEX THEN

    DBMS\_OUTPUT.PUT\_LINE('Error: Customer ID ' || cust\_id || ' already exists.');

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

    ROLLBACK;

END;

/

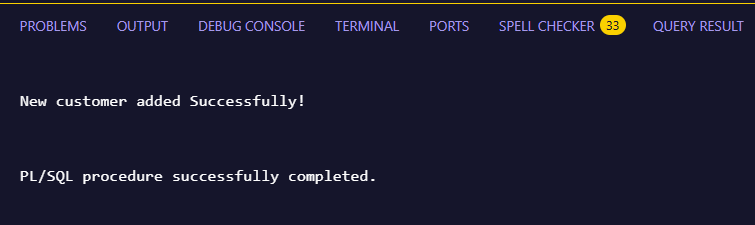
**Valid TestCase:**

BEGIN

  AddNewCustomer(3, 'Elsa', TO\_DATE('1999-04-15', 'YYYY-MM-DD'), 50000);

END;

/

**Output:**

**Invalid TestCase:**

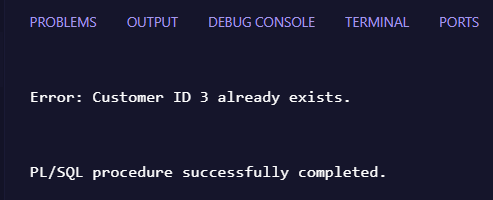
BEGIN

  AddNewCustomer(3, 'Anna', TO\_DATE('1988-03-15', 'YYYY-MM-DD'), 30000);

END;

/

**Output:**

****