Oracle SQL*Plus

File Edit Search Options Help

```
PL/SQL procedure successfully completed.

SQL> CREATE OR REPLACE PROCEDURE SafeTransferFunds(
  2      p_from_account_id NUMBER,
  3      p_to_account_id NUMBER,
  4      p_amount NUMBER
  5  ) AS
  6      v_from_balance NUMBER;
  7      v_to_balance NUMBER;
  8  BEGIN
  9      -- Lock accounts to prevent concurrent modifications
 10      SELECT Balance INTO v_from_balance FROM Accounts
 11      WHERE AccountID = p_from_account_id FOR UPDATE;
 12
 13      SELECT Balance INTO v_to_balance FROM Accounts
 14      WHERE AccountID = p_to_account_id FOR UPDATE;
 15
 16      -- Check for sufficient funds
 17      IF v_from_balance < p_amount THEN
 18          RAISE_APPLICATION_ERROR(-20001, 'Insufficient funds in the source account.');
 19      END IF;
 20
 21      -- Deduct from source account
 22      UPDATE Accounts
 23      SET Balance = Balance - p_amount
 24      WHERE AccountID = p_from_account_id;
 25
 26      -- Add to destination account
 27      UPDATE Accounts
 28      SET Balance = Balance + p_amount
 29      WHERE AccountID = p_to_account_id;
 30
 31      -- Commit transaction
 32      COMMIT;
 33
 34      DBMS_OUTPUT.PUT_LINE('Transfer completed successfully.');
 35
 36  EXCEPTION
 37      WHEN NO_DATA_FOUND THEN
 38          ROLLBACK;
 39          DBMS_OUTPUT.PUT_LINE('Account not found. Transfer aborted.');
 40      WHEN OTHERS THEN
 41          ROLLBACK;
 42          DBMS_OUTPUT.PUT_LINE('An error occurred during the transfer: ' || SQLERRM);
 43  END SafeTransferFunds;
 44  /

Procedure created.
```

Oracle SQL*Plus

File Edit Search Options Help

```
SQL> CREATE OR REPLACE PROCEDURE UpdateSalary(
  2      p_employee_id NUMBER,
  3      p_percentage NUMBER
  4  ) AS
  5      v_current_salary NUMBER;
  6  BEGIN
  7      -- Get current salary
  8      SELECT Salary INTO v_current_salary FROM Employees
  9      WHERE EmployeeID = p_employee_id;
 10
 11      -- Update salary
 12      UPDATE Employees
 13      SET Salary = Salary * (1 + p_percentage / 100)
 14      WHERE EmployeeID = p_employee_id;
 15
 16      -- Commit transaction
 17      COMMIT;
 18
 19      DBMS_OUTPUT.PUT_LINE('Salary updated successfully.');
 20
 21  EXCEPTION
 22      WHEN NO_DATA_FOUND THEN
 23          DBMS_OUTPUT.PUT_LINE('Employee ID not found. Salary update aborted.');
 24      WHEN OTHERS THEN
 25          DBMS_OUTPUT.PUT_LINE('An error occurred while updating salary: ' || SQLERRM);
 26  END UpdateSalary;
 27  /

Procedure created.

SQL> CREATE OR REPLACE PROCEDURE AddNewCustomer(
  2      p_customer_id NUMBER,
  3      p_name VARCHAR2,
  4      p_dob DATE,
  5      p_balance NUMBER
  6  ) AS
  7  BEGIN
  8      -- Insert new customer
  9      INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)
 10      VALUES (p_customer_id, p_name, p_dob, p_balance, SYSDATE);
 11
 12      -- Commit transaction
 13      COMMIT;
 14
 15      DBMS_OUTPUT.PUT_LINE('Customer added successfully.');
 16
 17  EXCEPTION
 18      WHEN DUP_VAL_ON_INDEX THEN
 19          DBMS_OUTPUT.PUT_LINE('Customer ID already exists. Insertion aborted.');
```

Oracle SQL*Plus

File Edit Search Options Help

```
  6  BEGIN
  7      -- Get current salary
  8      SELECT Salary INTO v_current_salary FROM Employees
  9      WHERE EmployeeID = p_employee_id;
 10
 11      -- Update salary
 12      UPDATE Employees
 13      SET Salary = Salary * (1 + p_percentage / 100)
 14      WHERE EmployeeID = p_employee_id;
 15
 16      -- Commit transaction
 17      COMMIT;
 18
 19      DBMS_OUTPUT.PUT_LINE('Salary updated successfully.');
 20
 21  EXCEPTION
 22      WHEN NO_DATA_FOUND THEN
 23          DBMS_OUTPUT.PUT_LINE('Employee ID not found. Salary update aborted.');
 24      WHEN OTHERS THEN
 25          DBMS_OUTPUT.PUT_LINE('An error occurred while updating salary: ' || SQLERRM);
 26  END UpdateSalary;
 27  /

Procedure created.

SQL> CREATE OR REPLACE PROCEDURE AddNewCustomer(
  2      p_customer_id NUMBER,
  3      p_name VARCHAR2,
  4      p_dob DATE,
  5      p_balance NUMBER
  6  ) AS
  7  BEGIN
  8      -- Insert new customer
  9      INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)
 10      VALUES (p_customer_id, p_name, p_dob, p_balance, SYSDATE);
 11
 12      -- Commit transaction
 13      COMMIT;
 14
 15      DBMS_OUTPUT.PUT_LINE('Customer added successfully.');
 16
 17  EXCEPTION
 18      WHEN DUP_VAL_ON_INDEX THEN
 19          DBMS_OUTPUT.PUT_LINE('Customer ID already exists. Insertion aborted.');
 20      WHEN OTHERS THEN
 21          DBMS_OUTPUT.PUT_LINE('An error occurred while adding a customer: ' || SQLERRM);
 22  END AddNewCustomer;
 23  /

Procedure created.
```