**Exercise 3: Stored Procedures**

**Types of Procedures in PL/SQL**

There are two main types of procedures in PL/SQL: Anonymous Procedures (unnamed) and Stored Procedures (named).

A Stored Procedure is a named piece of code stored in memory for repeated use across multiple sessions, while an Anonymous Procedure has no name and exists only within the current session.

The key difference lies in scope:

Anonymous procedures are created, executed, and disappear within the same session.

Stored procedures persist beyond the session, allowing reuse anytime without redefining.

**Advantages of Stored Procedures**

Stored procedures enhance code reusability by allowing the same task to be executed multiple times without rewriting the code.

They improve security by encapsulating logic in a single place and controlling access.

Stored procedures boost performance because their execution plan is cached after the first run, making subsequent calls faster without repeated syntax checks.

**Example of a Stored Procedure**

Stored procedures are created using CREATE OR REPLACE PROCEDURE syntax, allowing updates without changing the procedure name.

Input parameters and output parameters are declared similarly, with IN for inputs and OUT for outputs; either IS or AS is used as part of the syntax.

An addition stored procedure example takes two input numbers, adds them, and stores the result in the output parameter.

Once created, the stored procedure is saved in the database and can be called anytime, across sessions, without redefining it.

Calling the procedure involves declaring variables, assigning values to inputs, invoking the procedure by name, and retrieving the output result.

|  |  |  |
| --- | --- | --- |
| Parameter Mode | Description | Usage |
| IN | Input parameter | Passes data into the procedure |
| OUT | Output parameter | Returns data from the procedure |
| IN OUT | Both input and output | Allows modification of input data within procedure |

**Example**

CREATE OR REPLACE PROCEDURE addition(

Num1 IN NUMBER,

Num2 IN NUMBER,

SUM1 OUT NUMBER

) IS

BEGIN

SUM1 := Num1 + Num2;

END;

/

DECLARE

a NUMBER(2);

b NUMBER(2);

c NUMBER(4);

BEGIN

a := &a;

b := &b;

addition(a, b, c);

DBMS\_OUTPUT.PUT\_LINE('Sum = ' || c);

END;

/

Output:

Enter value for a: old 6: a := &a;

new 6: a := 2 ;

Enter value for b: old 7: b := &b;

new 7: b := 3;

Sum = 5