Stored Procedures Part - I

- Jayendra Khatod

Objectives

- Distinguish anonymous PL/SQL blocks from named PL/SQL blocks (subprograms)
- Describe subprograms
- Describe PL/SQL blocks and subprograms
- Describe the uses of procedures
- Create procedures
- Differentiate between formal and actual parameters
- List the features of different parameter modes – IN, OUT and IN-OUT
- Create procedures with parameters

Overview of Subprograms

A subprogram:

- **Is a named PL/SQL block that can accept parameters and be invoked from a calling environment**
- **Is based on standard PL/SQL block structure**
- **Provides modularity, reusability, extensibility, and maintainability**
- **Is of two types: Procedures and Functions**

Block Structure for Anonymous PL/SQL Blocks

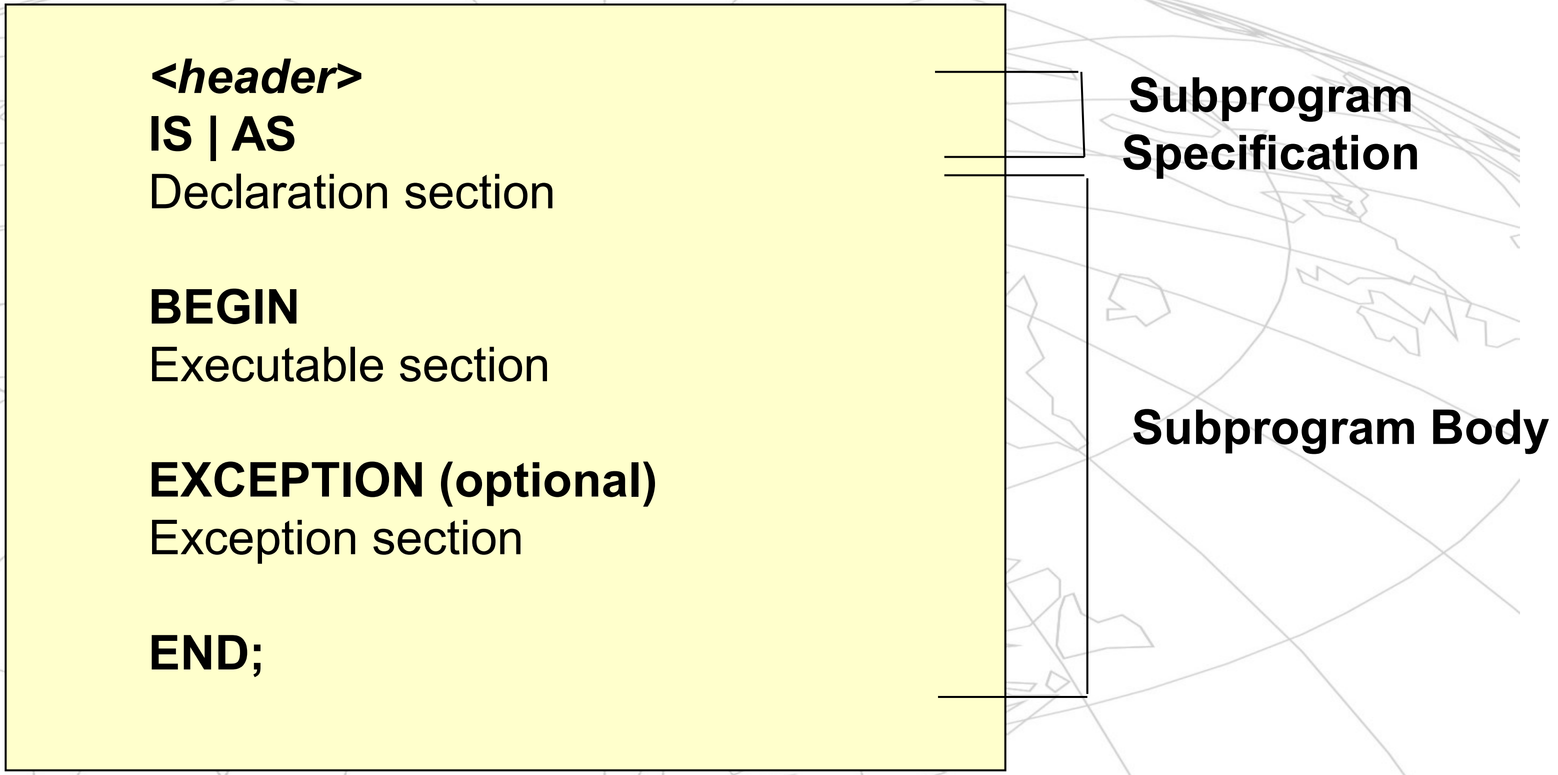
DECLARE (optional)
Declare PL/SQL objects to be used within this block

BEGIN (mandatory)
Define the executable statements

EXCEPTION (optional)
Define the actions that take place if an error or exception arises

END; (mandatory)

Block Structure for PL/SQL Subprograms



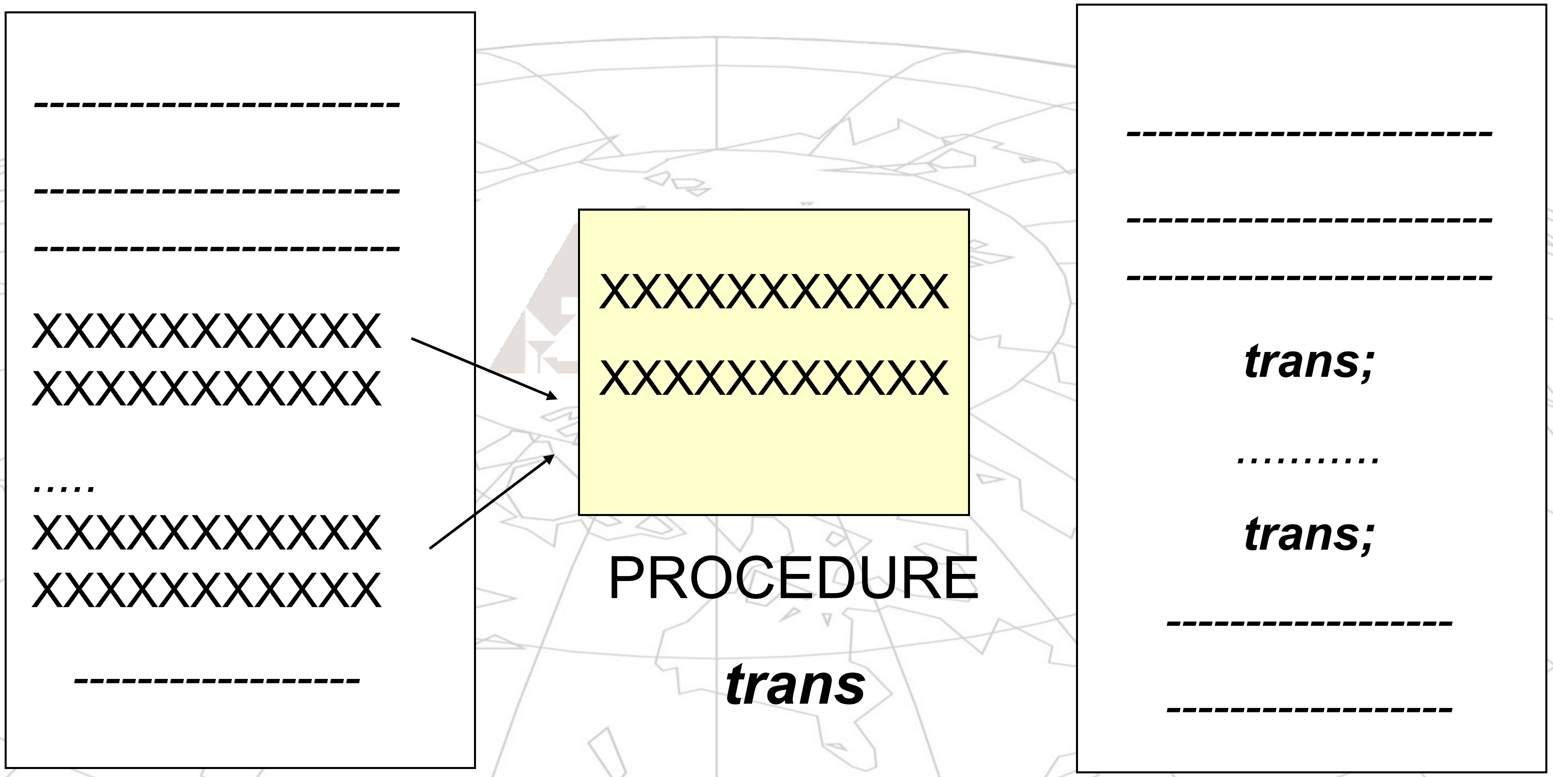
```
<header>
IS | AS
Declaration section

BEGIN
Executable section

EXCEPTION (optional)
Exception section

END;
```

PL/SQL Subprograms



Main Program

*Repeating code transferred to the subprogram **trans***

Main Program

What Is a Procedure?

- **A procedure is a type of subprogram that performs an action.**
- **A procedure can be stored in the database, as a schema object, for repeated execution.**

Syntax for Creating Procedures

```
CREATE [OR REPLACE] PROCEDURE procedure_name
 [(parameter1 [mode1] datatype1,
  parameter2 [mode2] datatype2,
  . . .)]

  IS|AS

BEGIN
…
```

- The REPLACE option indicates that if the procedure exists, it will be dropped and replaced with the new version created by the statement.
- PL/SQL block starts with either BEGIN or the declaration of local variables and ends with either END or END *procedure_name.*

Formal Versus Actual Parameters

- **Formal parameters:**

  Variables declared in the parameter list of a subprogram specification

  Example:

  CREATE PROCEDURE raise_sal(p_id NUMBER, p_amount NUMBER)

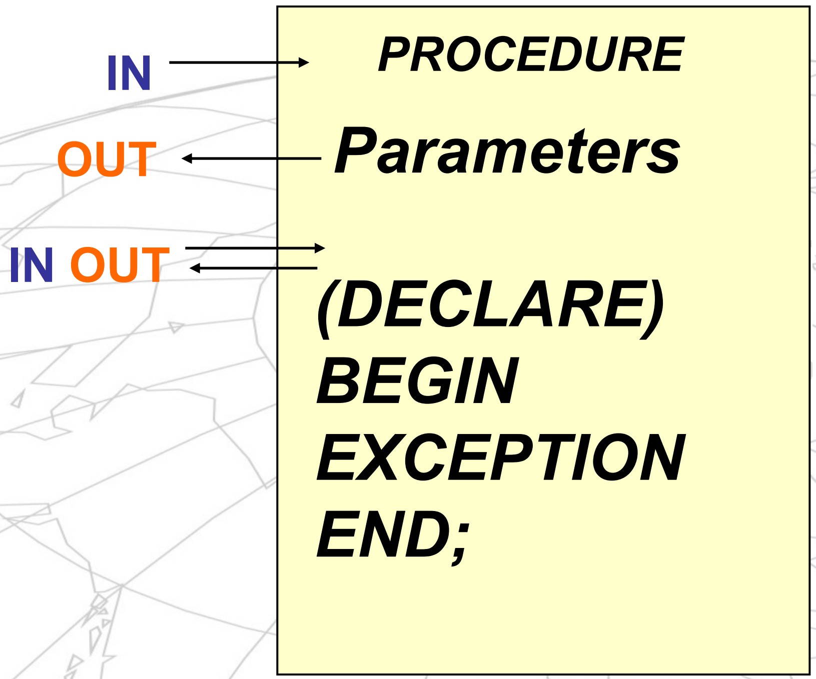
- **Actual parameters:**

  Variables or expressions referenced in the parameter list of a subprogram call

  Example:

  raise_sal(v_id, 2000);

Procedural Parameter Modes



Procedural Parameters

When you create the procedure, the formal parameter defines the value used in the executable section of the PL/SQL block, whereas the actual parameter is referenced when invoking the procedure

Procedural Parameter Modes

IN	OUT	IN OUT
Default mode	Must be specified	Must be specified
Value is passed into subprogram	Returned to calling environment	Passed into subprogram; returned to calling environment
Formal parameter acts as a constant	Uninitialized variable	Initialized variable
Actual parameter can be a literal, expression, constant, or initialized variable	Must be a variable	Must be a variable
Can be assigned a default value	Cannot be assigned a default value	Cannot be assigned a default value

IN Parameters: Example

```
CREATE OR REPLACE PROCEDURE
  raise_salary
    (p_id IN emp.empno%TYPE)
   IS
   BEGIN
      UPDATE emp
      SET sal = sal * 1.10
      WHERE empno = p_id;
   END raise_salary;
   /
```

OUT Parameters : Example

```
CREATE OR REPLACE PROCEDURE query_emp
   (p_id IN employees.employee_id%TYPE,
   p_name OUT employees.last_name%TYPE,
   p_salary OUT employees.salary%TYPE,
   p_comm OUT employees.commission_pct%TYPE)
IS
BEGIN
    SELECT last_name, salary, commission_pct
    INTO p_name, p_salary, p_comm
    FROM employees
    WHERE employee_id = p_id;
END query_emp;
```

Viewing OUT Parameters

```
Declare
v_name varchar2(50);
v_salary number;
v_comm number;

Begin

query_emp(100, v_name, v_salary,v_comm );
dbms_output.put_line (v_name||'  '||v_salary || ' '
   ||v_comm );

end;
```

IN OUT Parameters : Example

```
CREATE OR REPLACE PROCEDURE
   format_phone
  (p_phone_no IN OUT VARCHAR2)
IS
BEGIN
    p_phone_no := '(' ||
   SUBSTR(p_phone_no,1,3) ||
    ')' || SUBSTR(p_phone_no,4,3) ||
    '-' || SUBSTR(p_phone_no,7);
END format_phone;
/
```

Viewing IN OUT Parameters

```
DECLARE
v_phone_no  varchar2(50) := ‘9890777890';
BEGIN
format_phone (v_phone_no);
dbms_output.put_line (v_phone_no);
END;
```

Summary

- **Procedures can serve as building blocks for an application.**
- **Create procedures Syntax**
- **Difference between formal and actual parameters**
- **There are three parameter modes as**

  IN

  OUT

  IN-OUT

Thank You !