**Procedimientos almacenados**

• Los procedimientos o funciones son **bloques PL/SQL** con nombre, que pueden recibir parámetros y pueden ser invocados desde distintos entornos: SQL\*PLUS, Oracle\*Forms, desde otros procedimientos y funciones y desde otras herramientas Oracle y aplicaciones.

• Los procedimientos y funciones llevan a cabo tareas específicas, y su mayor diferencia radica en que las funciones devuelven un valor.

**Sintaxis Procedimientos**

*CREATE [OR REPLACE} PROCEDURE [esquema].nombre-procedimiento*

*(nombre-parámetro {IN | OUT | IN OUT} tipo de dato, ..) {IS | AS}*

*Declaración de variables;*

*Declaración de constantes;*

*Declaración de cursores;*

*BEGIN*

*Cuerpo del subprograma PL/SQL;*

*EXCEPTION*

*Bloque de excepciones PL/SQL;*

*END;*

Descripción de la sintaxis:

• **Nombre-parámetro**: es el nombre que queramos dar al parámetro. Podemos utilizar múltiples parámetros. En caso de no necesitarlos, podemos omitir los paréntesis.

• **IN**: especifica que el parámetro es de entrada y que por tanto dicho parámetro tiene que tener un valor en el momento de llamar a la función o procedimiento. Si no se especifica nada, los parámetros son por defecto de tipo entrada.

• **OUT**: especifica que se trata de un parámetro de salida. Son parámetros cuyo valor es devuelto después de la ejecución el procedimiento al bloque PL/SQL que lo llamó. Las funciones PLSQL no admiten parámetros de salida.

• **IN OUT**: Son parámetros de entrada y salida a la vez.

•**Tipo-de-dato**: Indica el tipo de dato PLSQL que corresponde al parámetro (NUMBER, VARCHAR2, etc).

Ejemplo de creación de un procedimiento:

Los procedimientos y funciones de PL/SQL se comportan, igualmente, como los procedimientos y funciones en otros lenguajes de tercera generación, compartiendo muchas de sus propiedades. A los procedimientos y funciones, también, se les denominan, en conjunto, SUBPROGRAMAS.

CREATE OR REPLACE PROCEDURE imprime (mensaje varchar2)

as

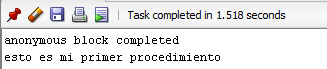
BEGIN

dbms\_output.put\_line (mensaje);

END;

Para invocar a este subprograma se hace de la siguiente forma:

**1era Forma**:

• **Desde un Sub Programa:**

SET serveroutput ON;

BEGIN

imprime ('esto es mi primer procedimiento');

END;

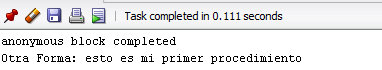
**2da Forma:**

• **Herramientas de desarrollo de aplicaciones de Oracle:**

SQL\*Plus, SQL Developer, SQL\*Dba, SQL\*Forms, SQL\*Menu, SQL\*ReportWriter, etc.

SET serveroutput ON;

EXECUTE imprime ('Otra Forma: esto es mi primer procedimiento');



* Procedimiento almacenado que actualiza el campo COMM de un determinado empleado con el valor de 120.

CREATE OR REPLACE PROCEDURE actualiza\_empleado

(w\_codigo\_emp IN emp.empno%TYPE)

AS

BEGIN

UPDATE EMP

SET COMM=120

WHERE empno=w\_codigo\_emp;

END actualiza\_empleado;

--EJECUTAR

EXECUTE actualiza\_empleado ('7369');

Crear un P.A. que Actualice el salario, fecha(HireDate) el cual recibirá como parámetro de entrada el código del empleado y el nuevo salario.

CREATE OR REPLACE PROCEDURE Actualiza\_SalarioA

(codemp NUMBER, new\_sal NUMBER)

AS

BEGIN

UPDATE EMP

SET SAL = new\_sal,HIREDATE = SYSDATE

WHERE EMPNO = codemp;

END Actualiza\_SalarioA;

**--EJECUTAR**

BEGIN

**Actualiza\_SalarioA(7369,2500);**

END;

Procedimiento almacenado que permite contratar un nuevo empleado.

CREATE OR REPLACE PROCEDURE contratar\_empleado

(w\_codigo\_emp IN emp.empno%TYPE,

w\_fecha\_alta IN emp.hiredate%TYPE,

w\_depart IN emp.deptno%TYPE)

AS

BEGIN

INSERT INTO EMP(empno, hiredate, deptno)

VALUES (w\_codigo\_emp, w\_fecha\_alta, w\_depart);

END contratar\_empleado;

En este procedimiento se ha definido el tipo de dato de los parámetros de entrada como del mismo tipo que los campos de la tabla “emp” , es decir:

nombreParametro IN nombreTabla.nombreColumna**%TYPE**.

Sería equivalente a poner:

w\_codigo\_emp number,

w\_depart varchar..

Llamadas a procedimientos

• **Desde un Sub Programa:**

BEGIN ...

*/\* llamada al procedimiento contratar\_empleado \*/*

contratar\_empleado (2645,'19/12/1999',30);

END;

• **Herramientas de desarrollo de aplicaciones de Oracle:**

SQL\*Plus, SQL Developer, SQL\*Dba, SQL\*Forms, SQL\*Menu, SQL\*ReportWriter, etc.

EXECUTE contratar\_empleado (2646,'19/12/1999',30);

Procedimiento almacenado que permite eliminar un empleado a través de su código

CREATE OR REPLACE PROCEDURE elimina\_empleado

(w\_codigo\_emp IN emp.empno%TYPE)

AS

BEGIN

DELETE FROM EMP

WHERE empno=w\_codigo\_emp;

END elimina\_empleado;

**EJECUTAR:**

EXECUTE elimina\_empleado ('2646');

P.A que permite obtener un empleado a través de su código

CREATE OR REPLACE PROCEDURE OBTENER\_EMP

(PARAME1 IN VARCHAR2)

AS

CODIGO VARCHAR2(30);

NOMBRE VARCHAR2(30);

BEGIN

SELECT EMPNO,ENAME INTO CODIGO,NOMBRE

FROM EMP WHERE EMPNO=PARAME1;

dbms\_output.put\_line('VALOR DE CODIGO: ' || codigo);

dbms\_output.put\_line('VALOR DE NOMBRE: ' || nombre);

END OBTENER\_EMP;

**EJECUTAR:**

set serveroutput on;

BEGIN

OBTENER\_EMP('7369');

END;

P.A que recibe como parámetro la inicial del nombre de un empleado y muestre los empleados que inicien con dicha inicial, además para visualizar los datos en consola utilice un cursor.

CREATE OR REPLACE PROCEDURE INICIAL\_EMP

(INICIAL IN VARCHAR2)

AS

CODIGO VARCHAR2(30);

NOMBRE VARCHAR2(30);

FECHA\_INGRESO DATE;

BEGIN

DECLARE

CURSOR c\_nuevocursor

IS

SELECT EMPNO,ENAME,HIREDATE INTO CODIGO,NOMBRE,FECHA\_INGRESO

FROM EMP WHERE ENAME LIKE INICIAL || '%';

BEGIN

FOR x IN c\_nuevocursor

LOOP

DBMS\_OUTPUT.put\_line (x.empno || ' ' || x.ename || ' ' || x.hiredate);

END LOOP;

END;

END INICIAL\_EMP;

**EJECUTAR**:

set serveroutput on;

BEGIN

INICIAL\_EMP('A');

END;

**Propuestos**

* Crear un P.A. que liste los empleados por nombre de Departamento
* Crear un P.A que liste los empleados cuyo salario este comprendido entre un rango de salarios.
* Crear un P.A que reciba como parámetro el tipo de operación, si es 1 muestre los empleados cuyo código sea impar y si es 0 los pares.
* Crear un P.A que ingrese registros a la tabla Departamento.
* Crear un P.A que elimine registros de la tabla Departamento.

**FUNCIONES**

**Sintaxis Funciones**

*CREATE [OR REPLACE] FUNCTION [esquema].nombre-función*

*(nombre-parámetro {IN | OUT | IN OUT} tipo-de-dato, ...)*

*RETURN tipo-de-dato {IS | AS}*

*Declaración de variables;*

*Declaración de constantes;*

*Declaración de cursores;*

*BEGIN*

*Cuerpo del subprograma PL/SQL;*

*EXCEPTION*

*Bloque de excepciones PL/SQL;*

*END;*

Ejemplo de creación de una función:

**Crear una función que retorne un bono especial por empleado, el cual será del 10% del salario**.

CREATE OR REPLACE FUNCTION bono\_emp(sal number)

return number

is

begin

return ((sal)\*0.10);

end bono\_emp;

**Consultar nuestra función**:

select empno,ename,hiredate,sal,bono\_emp(sal)

from emp;

**Crear una función que obtenga el salario de un determinado empleado.**

CREATE OR REPLACE FUNCTION obtener\_salario

(w\_codigo\_emp IN emp.empno%TYPE)

RETURN NUMBER

IS w\_salario emp.sal%TYPE;

BEGIN

SELECT sal INTO w\_salario

FROM emp

WHERE empno = w\_codigo\_emp;

RETURN w\_salario;

END obtener\_salario;

Cada función debe devolver un valor del tipo especificado utilizando la sentencia **RETURN.**

Llamadas a funciones

**• Desde un bloque anónimo**

**set serveroutput on;**

BEGIN

***DBMS\_OUTPUT.PUT\_LINE***('Salario cod\_emp 7369 '||obtener\_salario(7369));

END;

• **Desde una instrucción SQL**

SELECT obtener\_salario(7369) salario\_emp\_7369 FROM emp where rownum<=1;

**Documentación procedimientos y funciones**

• Para obtener los nombres de todos los procedimientos y funciones se puede consultar la VISTA **USER\_OBJECTS**

SELECT object\_name, object\_type FROM USER\_OBJECTS

WHERE object\_type IN ('PROCEDURE', 'FUNCTION');

• Para obtener el texto de un procedimiento o función almacenado se puede consultar la VISTA **USER\_SOURCE**

SELECT text FROM USER\_SOURCE

WHERE type = 'PROCEDURE'

AND name = 'CONTRATAR\_EMPLEADO';

Es necesario activar **SERVEROUTPUT** (SET SERVEROUTPUT ON) para ver las salidas desde procedimientos o funciones almacenados.

**Propuesto**:

* Crear una función que calcule el descuento de la tabla empleado, el cual se calculara sacando el 0.87 por ciento con respecto al salario.