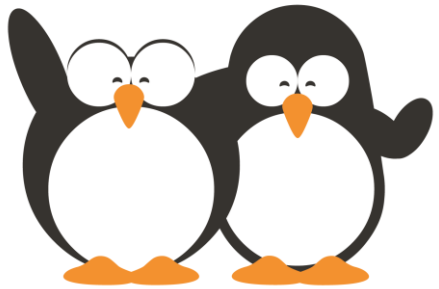
ÍNDICE

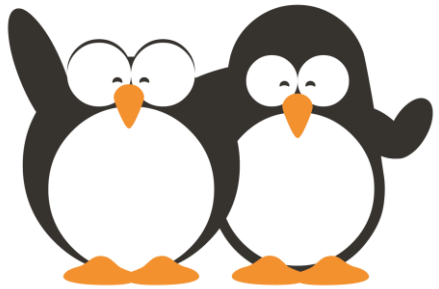


Procedimientos

Introducción

- Un procedimiento es un subprograma que ejecuta una acción específica y que no devuelve ningún valor. Un procedimiento tiene un nombre, un conjunto de parámetros (opcional) y un bloque de código.
- La sintaxis para declararlo es:

```
CREATE [OR REPLACE]
PROCEDURE <procedure_name> [(<param1> [IN|OUT|IN OUT] <type>,
                             <param2> [IN|OUT|IN OUT] <type>,
...)]
IS
 -- Declaración de variables locales
BEGIN
  -- Sentencias
[EXCEPTION]
  -- Sentencias control de excepcion
END [<procedure_name>];
```

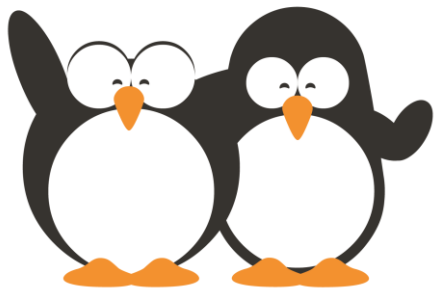


Procedimientos

Ejemplo

- En el ejemplo hemos creado un procedimiento que muestra los números desde el 1 hasta el valor pasado por parámetro.

```
create or replace procedure joc(i number) is
 num constant number := 2525;
begin
 case
   when i < num then
     dbms_output.put_line('El numero es mas pequeño, continua buscando...');
   when i > num then
     dbms_output.put_line('El numero es mas grande, continua buscando...');
   when i = num then
     dbms_output.put_line('HAS ACERTADO !!!!');
  end case;
end;
```



- Para llamar al procedimiento una vez creado simplemente debemos poner su nombre y, entre paréntesis el parámetro o parámetros necesarios.

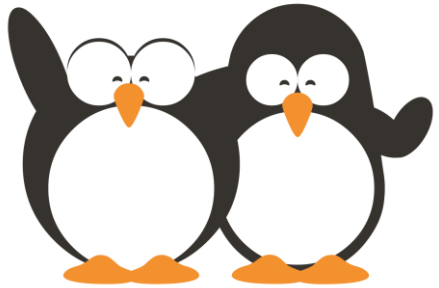
Procedimientos

Parámetros

- Para declarar los parámetros de un procedimiento debemos seguir la siguiente sintaxis.

  <param1> [**IN|OUT|IN OUT**] *<type>*

  - <param1> es el nombre del parámetro
  - <type> es el tipo de dato
  - [IN|OUT|IN OUT] → Indica la forma en la que pasamos el parámetro al procedimiento.
    - IN: El parámetro es de entrada, significa que la variable original (Fuera de la función) no se verá afectada.
    - OUT: El parámetro es de salida. El parámetro se usará para almacenar un valor de salida del procedimiento. No se puede emplear como parámetro de entrada.
    - IN OUT: El parámetro actúa como parámetro de entrada/salida, es decir, se emplea para pasar un valor al procedimiento y, además, como forma de almacenar un valor de salida.



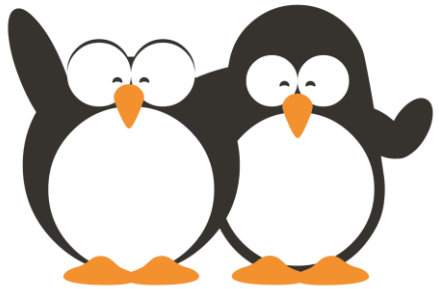
Funciones

Introducción

- Las funciones PL/SQL son unidades funcionales similares a los procedimientos, la principal diferencia radica en que las funciones devuelven un resultado tras su ejecución.
- Es necesario indicar el tipo de dato que la función va a devolver en la definición de la misma.
- Sintaxis:

```
CREATE [OR REPLACE]
FUNCTION <fn_name>[(<param1> IN <type>, <param2> IN <type>, ...)]
RETURN <return_type>
IS
  result <return_type>;
BEGIN

  return(result);
[EXCEPTION]
  -- Sentencias control de excepción
END [<fn_name>];
```

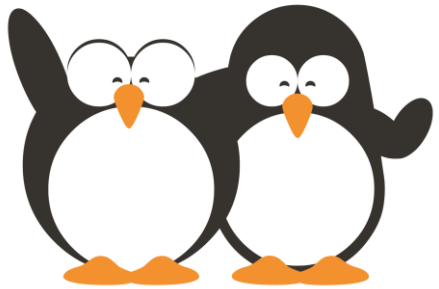


Funciones

Ejemplo

- En el ejemplo hemos programado una función que recibe dos números por parámetro y devuelve la suma de ambos.

```
CREATE OR REPLACE
FUNCTION sumarNumeros(num1 number, num2 number)
RETURN NUMBER
IS
   resultado NUMBER;
BEGIN
   resultado:=num1+num2;
   return(resultado);
END ;
```



Funciones

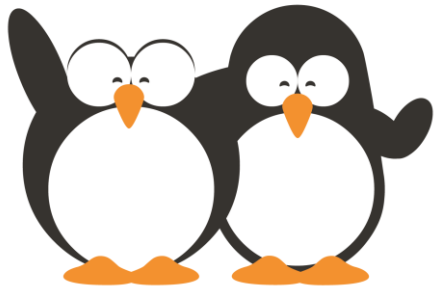
Llamadas a funciones

- Para llamar a una función se hace de forma similar a como llamábamos a los procedimientos, sin embargo, ahora podemos usar el dato que nos devuelve la función asignando a una variable o bien imprimiendo por pantalla.

```
Begin
  sumarNumeros(2,2);
End;
```

```
Declare
  suma Number;
Begin
  suma:=sumarNumeros(2,2);
end;
```

```
Begin
  DBMS_OUTPUT.PUT_LINE(sumarNumeros(2,2));
end;
```



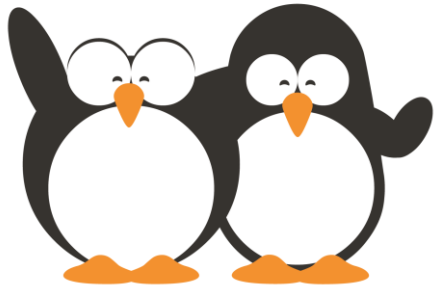
SQL en PL/SQL

Recuperar datos con SELECT

- Es posible recuperar datos de la base de datos y asignarlo a una variable dentro de nuestro código PL
- Sintaxis:

```
Select listaCampos
INTO variable1 [,...]
FROM tabla
WHERE condición;
```

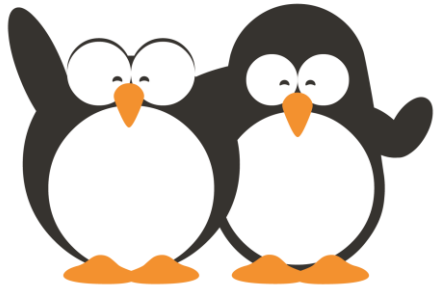
```
SET SERVEROUTPUT ON
-- Ejericio 6: Realizar en Pl/sql un programa que imprima por pantalla, la cantidad de empleados con cargo de director.
DECLARE
v_num NUMBER;
BEGIN
SELECT  count (*)INTO v_num FROM EMP WHERE oficio LIKE 'DIRECTOR';
DBMS_OUTPUT.PUT_LINE('La cantidad de empleados con cargo de director son: ' || v_num);
END;
```



CURSORES

- PL/SQL utiliza cursores para gestionar las instrucciones SELECT. Un cursor es un conjunto de registros devuelto por una instrucción SQL.
- Cursores implícitos. Cuando la consulta devuelve un único registro.

- Cursores explícitos. Son los cursores que son declarados y controlados por el programador. Se utilizan cuando la consulta devuelve un conjunto de registros.

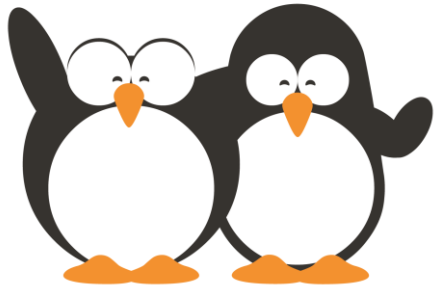


FORMATO DE UN CURSOR

- DECLARE CURSOR MICURSOR IS <consulta>

```
BEGIN
        OPEN MICURSOR; -- Abrir el cursor
        FETCH MICURSOR INTO VARIABLES_PLSQL -- Recorrer el cursor
        CLOSE MICURSOR; -- Cerrar el cursor
END; /
```

SECCIÓN DEL LIBRO 6.10 PAGINA 263



EJEMPLO DE CURSORES

```
set SERVEROUTPUT ON;
-- Programa que visualice el código y nombre de los clientes con código mayor a 104
declare
cursor cod_nombre is
select c.CLIENTE_COD, c.NOMBRE from cliente c WHERE c.CLIENTE_COD>104;
codigo cliente.cliente_cod%type;
nombre cliente.Nombre%type;
begin
 open cod_nombre;
 Loop
                fetch cod_nombre into codigo, nombre;
-- CADA FILA DEL CURSO LO GUARDAS EN LAS VARIABLES CODIGO Y NOMBRE
                exit when cod_nombre%notfound;
                DBMS_OUTPUT.PUT_LINE(codigo||' - '|| nombre);
 end loop;
 Close cod_nombre;
end;
```