

SQL: DCL y extensión procedimental, casos prácticos II.

Desarrollo de Aplicaciones
Multiplataforma, Desarrollo de
Aplicaciones Web.

Módulo 02: Bases de datos



Actividad

SQL: DCL y extensión procedimental, casos prácticos II.

Objetivos

- Aprender la sintaxis básica de PL/SQL.
- Programar procedimientos y funciones.
- Controlar errores en PL/SQL.
- Diseñar y programar disparadores.
- Desarrollar cursores implícitos y explícitos.

1. Cestas de Navidad

Ejercicio 1.

Un importante centro de distribución nos ha pedido ayuda realizar el control de las cestas de navidad.



Se deberán crear las siguientes tablas (escoger el tipo y longitud de campos que consideréis apropiado):

Cestas



- Código (Clave primaria)
- Nombre
- Precio
- Formato.
- Peso

Productos Cesta

- Código Cesta (Clave primaria)
- Código Producto (Clave primaria)
- Nombre
- Descripción
- Categoría
- Unidades.
- Precio.

Crea las tablas y añade información a cada una de ellas. Comprueba su funcionamiento.

Aclaración: *En realidad deberíamos realizar el ejercicio con tres tablas, Cestas, productos y otra con cestasproductos, pero lo simplificamos en dos para no complicar el ejercicio.*

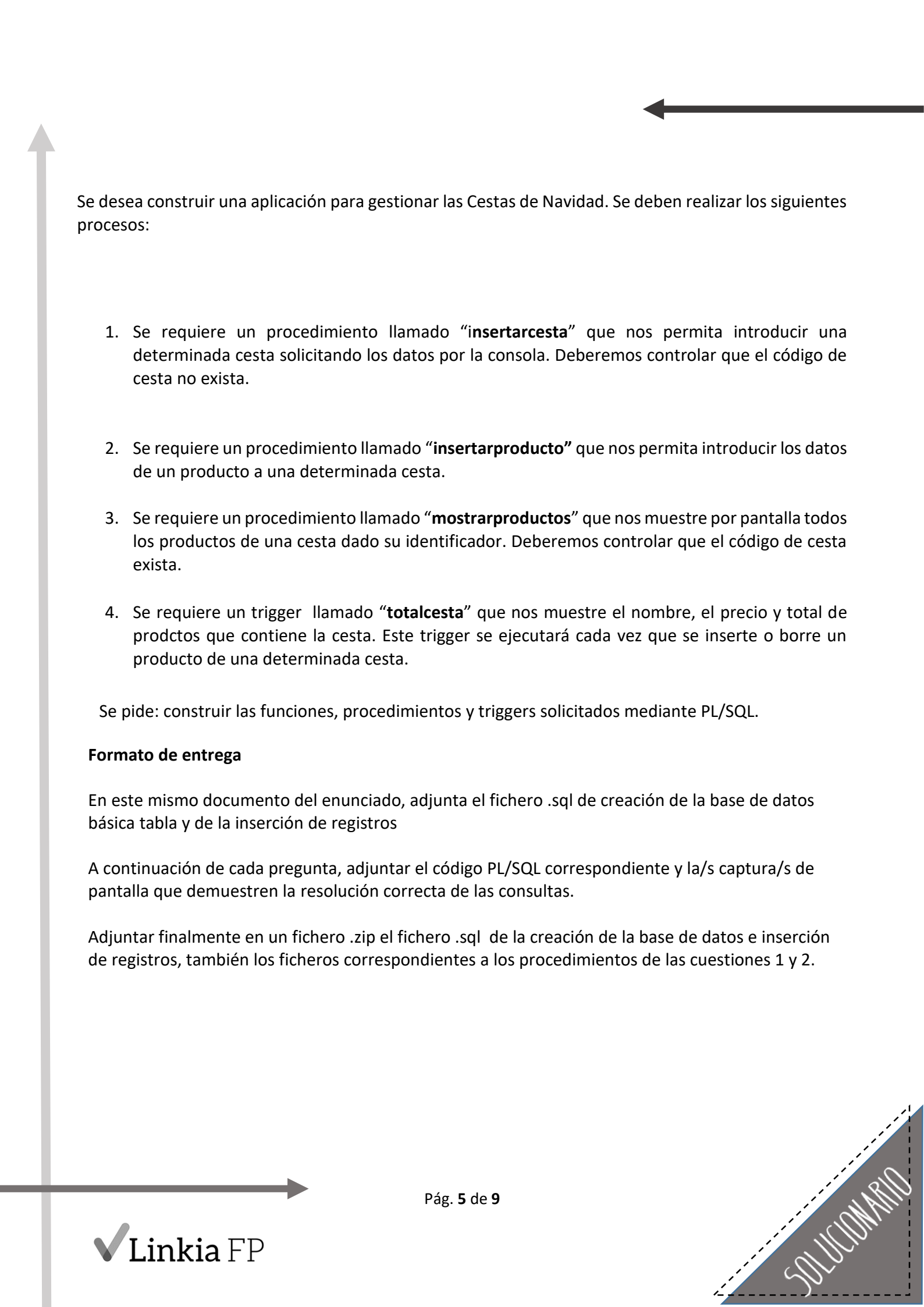


```
create table cestas (  
codigo varchar2(5) primary key,  
nombre varchar2(50),  
precio NUMBER(4),  
formato varchar2(50),  
peso FLOAT  
);
```

```
create table productos_cesta (  
codigo_cesta varchar2(5),  
codigo_producto varchar2(5) primary key,  
nombre varchar2(50),  
descripcion varchar(100),  
categoria varchar2(50),  
unidades number,  
precio float,  
constraint fk_codigo_cesta foreign key (codigo_cesta) references cestas(codigo)  
);
```

```
insert into cestas values ('1', 'bar pepe', 300, 'delux', 50);  
insert into productos_cesta values ('1', '1', 'jamón', 'jamón serrano', 'extra', 1, 80);
```

```
select * from cestas;  
select * from productos_cesta;
```



Se desea construir una aplicación para gestionar las Cestas de Navidad. Se deben realizar los siguientes procesos:

1. Se requiere un procedimiento llamado **“insertarcesta”** que nos permita introducir una determinada cesta solicitando los datos por la consola. Deberemos controlar que el código de cesta no exista.
2. Se requiere un procedimiento llamado **“insertarproducto”** que nos permita introducir los datos de un producto a una determinada cesta.
3. Se requiere un procedimiento llamado **“mostrarproductos”** que nos muestre por pantalla todos los productos de una cesta dado su identificador. Deberemos controlar que el código de cesta exista.
4. Se requiere un trigger llamado **“totalcesta”** que nos muestre el nombre, el precio y total de productos que contiene la cesta. Este trigger se ejecutará cada vez que se inserte o borre un producto de una determinada cesta.

Se pide: construir las funciones, procedimientos y triggers solicitados mediante PL/SQL.

Formato de entrega

En este mismo documento del enunciado, adjunta el fichero .sql de creación de la base de datos básica tabla y de la inserción de registros

A continuación de cada pregunta, adjuntar el código PL/SQL correspondiente y la/s captura/s de pantalla que demuestren la resolución correcta de las consultas.

Adjuntar finalmente en un fichero .zip el fichero .sql de la creación de la base de datos e inserción de registros, también los ficheros correspondientes a los procedimientos de las cuestiones 1 y 2.



```
/*-----PROCEDIMIENTO PARA PODER ENTRAR PRODUCTOS EN "CESTAS"-----*/

create or replace procedure insertarcesta (
v_codigo cestas.codigo%type,
v_nombre cestas.nombre%type,
v_precio cestas.precio%type,
v_formato cestas.formato%type,
v_peso cestas.peso%type
)
IS
num_error number;
msg_error Varchar2(255);
Begin
insert into cestas(codigo, nombre, precio, formato, peso) values (v_codigo, v_nombre, v_precio,
    v_formato, v_peso);
commit;
exception
when dup_VAL_ON_INDEX then
dbms_output.put_line('elcodigo introducido ya existe' || SQLERRM);
END;
/

/*----bloque anonimo para llamar al procedimiento insertarcesta----*/
accept v_codigo prompt'codigo cesta';
accept v_nombre prompt'nombre cesta';
accept v_precio prompt'precio cesta';
accept v_formato prompt'formato cesta';
accept v_peso prompt'peso cesta';
declare
v_codigo cestas.codigo%type:='&v_codigo';
v_nombre cestas.nombre%type:='&v_nombre';
v_precio cestas.precio%type:='&v_precio';
v_formato cestas.formato%type:='&v_formato';
v_peso cestas.peso%type:='&v_peso';
BEGIN
insertarcesta(v_codigo, v_nombre, v_precio, v_formato, v_peso);
END;
/
```



```
/*----PROCEDIMIENTO PARA PRODUCTOS_CESTA----*/
```

```
create or replace procedure insertarproducto(  
v_codigo_cesta productos_cesta.codigo_cesta%type,  
v_codigo_producto productos_cesta.codigo_producto%type,  
v_nombre productos_cesta.nombre%type,  
v_descripcion productos_cesta.decripcion%type,  
v_categoria productos_cesta.categoria%type,  
v_unidades productos_cesta.unidades%type,  
v_precio productos_cesta.precio%type  
)  
Is  
Num_error number;  
MSG_erro varchar2(255);  
BEGIN  
INSERT INTO productos_cesta(codigo_cesta, codigo_producto, nombre, decripcion, categoria,  
unidades, precio)  
VALUES (v_codigo_cesta, v_codigo_producto, v_nombre, v_descripcion, v_categoria, v_unidades,  
v_precio);  
commit;  
EXCEPTION  
WHEN DUP_VAL_ON_INDEX THEN  
dbms_output.put_line('el codigo esta repetido');  
END;  
/
```

```
/*bloque anonimo para el procedimiento insertarproducto*/
```

```
accept v_codigo_cesta prompt'inserta codigo cesta';  
accept v_codigo_producto prompt'insert codigo producto';  
accept v_nombre prompt'nombre producto';  
accept v_descripcion prompt'descripcion del producto';  
accept v_categoria prompt'categoria producto';  
accept v_unidades prompt'undades del producto';  
accept v_precio prompt'precio producto';  
DECLARE  
v_codigo_cesta productos_cesta.codigo_cesta%type:='&v_codigo_cesta';  
v_codigo_producto productos_cesta.codigo_producto%type:='&v_codigo_producto';  
v_nombre productos_cesta.nombre%type:='&v_nombre';  
v_descripcion productos_cesta.decripcion%type:='&v_descripcion';  
v_categoria productos_cesta.categoria%type:='&v_categoria';  
v_unidades productos_cesta.unidades%type:='&v_unidades';  
v_precio productos_cesta.precio%type:='&v_precio';
```

```

BEGIN
insertarproducto(v_codigo_cesta, v_codigo_producto, v_nombre, v_descripcion, v_categoria,
    v_unidades, v_precio);
END;



/*-----PRODECIDENTOS PARA MOSTRAR-----*/

/*-----PROCEDURE PARA MOSTRAR LOS PRODUCTOS DE UNA CESTA ENTRADA POR CONSOLA-----*/
create or replace procedure p_mostraproductos (
p_mostraproductos productos_cesta.codigo_cesta%TYPE
)
IS
cursor c_mostraproductos IS
SELECT nombre, unidades FROM productos_cesta WHERE codigo_cesta = p_mostraproductos;
v_nombre productos_cesta.nombre%type;
v_unidades productos_cesta.unidades%type;
begin
open c_mostraproductos;
LOOP
FETCH c_mostraproductos INTO v_nombre, v_unidades;
EXIT WHEN c_mostraproductos%notfound;
dbms_output.put_line(v_nombre || '-----' || v_unidades);
END LOOP;
IF c_mostraproductos%rowcount = 0 THEN
dbms_output.put_line('no tienes productos');
END IF;
CLOSE c_mostraproductos;
END;

EXECUTE p_mostraproductos(1);

/*-----PROCEDURE PARA MOSTRAR LA CESTA ENTRADA POR CONSOLA-----*/
create or replace procedure p_mostracestas (p_mostracestas productos_cesta.unidades%Type) IS
cursor c_mostracestas IS
select c.nombre, p.unidades FROM cestas c
left join productos_cesta p
on codigo_cesta = codigo
WHERE unidades = p_mostracestas;
v_nombre cestas.nombre%type;
v_unidades productos_cesta.unidades%type;
BEGIN
open c_mostracestas;
loop
FETCH c_mostracestas INTO v_nombre, v_unidades;

```

```

exit when c_mostracestas%notfound;
DBMS_OUTPUT.PUT_LINE(v_nombre || '-----' || v_unidades);
end loop;
If c_mostracestas%rowcount = 0 THEN
dbms_output.put_line('no tienes cestas con estas unidades');
end IF;
close c_mostracestas;
END;

execute p_mostracestas(2);

/*-----TRIGGER ARA MOSTRAR EL TOTAL DE LA CESTA MODIFICADA INSERT/DELETE-----*/

/*cursor para mostrar nombre, precio y total productos de la cesta*/
set SERVEROUTPUT ON;
create or replace trigger totalcesta
AFTER
DELETE OR INSERT OR UPDATE OF nombre on cestas
FOR EACH ROW
DECLARE
v_nombre cestas.nombre%TYPE;
v_precio cestas.precio%TYPE;
v_unidades productos_cesta.unidades%TYPE;
CURSOR c_totalcesta IS
select c.nombre, c.precio, p.unidades FROM cestas c
left join productos_cesta p
On codigo = codigo_cesta;
begin
open c_totalcesta;
LOOP
FETCH c_totalcesta INTO v_nombre, v_precio, v_unidades;
EXIT WHEN c_totalcesta%notfound;
dbms_output.put_line(v_nombre || '-----' || v_precio || '-----' || v_unidades);
END LOOP;
close c_totalcesta;
END;

```

