

Índice

Objetivo de la práctica	2
Descripción del problema	2
Ejercicio 1 : Definir el modelo de datos	2
Ejercicio 2 : Procedimiento para crear un producto	2
Ejercicio 3 : Entrada y salida de productos	3
Ejercicio 4 : Vista de existencias (1 punto)	4
Ejercicio 5 : Salida del almacén respetando existencias (1 punto)	4
Instrucciones de entrega	4

Objetivo de la práctica

En esta práctica el alumno utilizará las funcionalidades de PLSQL para automatizar algunas operaciones y para realizar comprobaciones sobre los datos. Estas operaciones y comprobaciones no pueden realizarse solo con SQL.

La última versión de esta práctica está disponible en [este enlace](#).

Descripción del problema

Una compañía necesita automatizar su almacén

- De cada producto se almacena su identificador, su nombre y su *stock*.
- En cada entrada de producto al almacén, se apunta
 - El proveedor de esta entrada
 - La fecha de entrada
 - El producto
 - La cantidad de producto
 - El precio pagado al proveedor por unidad de producto
- De cada salida de producto del almacén se apunta
 - El cliente al que se envía
 - La fecha de salida
 - El producto, cantidad de producto y precio por unidad que paga el proveedor

Ejercicio 1 : Definir el modelo de datos

Crea las tablas y funciones necesarias para soportar la descripción del problema. Las tablas y atributos concretos no son importantes, ya que el profesor consultará los datos a partir de la vista definida más adelante.

Ejercicio 2 : Procedimiento para crear un producto

Crea un procedimiento de nombre `CREAR_PRODUCTO` (listado 1), que inserte un nuevo producto en la base de datos. Los productos tendrán un identificador basado en la secuencia `SECUENCIA_PRODUCTO_ID`, que aumentará de uno en uno.

```
create or replace procedure CREAR_PRODUCTO (nombreproducto IN varchar(255),  
      ↪ idproducto OUT number)  
as  
begin  
    -- CONSIGUE EL NUEVO ID  
    -- INSERTA EL PRODUCTO CON ESE ID  
end;
```

Listado 1: Creación del procedimiento `CREAR_PRODUCTO`

El procedimiento devolverá en su segundo parámetro el identificador del producto recién creado.

Crea una vista de nombre V_PRODUCTOS para que el profesor pueda consultar los productos en el catálogo (listado 2)

```
create or replace view V_PRODUCTOS(nombreproducto,idproducto) as
...
```

Listado 2: Creación de la vista V_PRODUCTOS

Ejercicio 3 : Entrada y salida de productos

- Crea un procedimiento ENTRADA_PRODUCTO que actualice la base de datos cuando llegue un producto.
- Crea un procedimiento SALIDA_PRODUCTO que actualice la base de datos cuando se envíe un producto.
- En ninguno de los dos casos la fecha de la entrada o la salida puede ser posterior a la actual (error -20103).
- Si el producto no existe, se lanzará el error -20102.

```
create or replace procedure ENTRADA_PRODUCTO(
    idproducto IN number,
    idproveedor IN number,
    cantidad IN number,
    preciopagado IN number,
    fecha IN timestamp default systimestamp)
as
begin
    ...
end;

create or replace procedure SALIDA_PRODUCTO(
    idproducto IN number,
    idcliente IN number,
    cantidad IN number,
    preciocobrado IN number,
    fecha IN timestamp default systimestamp)
as
begin
    ...
end;
```

Listado 3: Entrada y salida de productos

Para poder corregirse, debe existir la función EXISTENCIAS_PRODUCTO que informe del *stock* de un producto en un momento concreto. Se tendrán en cuenta las entradas y salidas de producto hasta la fecha indicada (fecha incluida). Un producto que no ha tenido entradas o salidas tiene un *stock* de cero. Un producto que no existe tiene *stock* -1.

```
create or replace function EXISTENCIAS_PRODUCTO(idproducto IN number, fecha IN
    ↪ timestamp default sysdate) return number
as
begin
    ...
end;
```

Listado 4: Función de existencias

Ejercicio 4 : Vista de existencias (1 punto)

Crea la vista V_EXISTENCIAS. En esta vista se listan todos los productos existentes y su stock **en el momento actual**. Un producto que nunca ha tenido entradas o salidas debe tener un *stock* de cero. Un producto que no se ha comprado nunca a un proveedor tiene un *ultimopreciocompra* a NULL. Un producto que nunca se ha vendido a un cliente tiene *ultimoprecioventa* a NULL.

```
create or replace view EXISTENCIAS(idproducto,existencias,ultimopreciocompra,
    ↪ ultimoprecioventa) as
...
```

Listado 5: Vista de existencias

Ejercicio 5 : Salida del almacén respetando existencias (1 punto)

Crea un procedimiento SALIDA_PRODUCTO_CON_STOCK. Realizará el mismo proceso que SALIDA_PRODUCTO, pero en el caso de que no haya existencias suficientes lanzará un error con el mensaje Rotura de stock y número -20101

```
create or replace procedure SALIDA_PRODUCTO_CON_STOCK(
    idproducto IN number,
    idcliente IN number,
    cantidad IN number,
    preciocobrado IN number,
    fecha IN timestamp default systimestamp)
as
begin
    ...
    RAISE_APPLICATION_ERROR(-20101,'Rotura de stock');
    ...
end;
```

Listado 6: Entrada y salida de productos

Instrucciones de entrega

Se creará un único fichero SQL para todos los apartados.

- Este fichero se corregirá de forma semiautomática, por lo que es necesario seguir la nomenclatura propuesta en el ejercicio.

- El fichero se cargará en un usuario recién creado con permisos necesarios para crear todos los elementos necesarios (tablas, vistas, funciones, secuencias...)
- Si tiene **errores** de compilación podría no corregirse. Si no se siguen los **nombres de objetos** pedidos podría no corregirse.

La autoría del trabajo puede ser por parejas. A pesar de ello, cada alumno debe subir al moodle una copia del trabajo.

Sube el documento a [la tarea correspondiente en el aula virtual](#). Presta atención al plazo de entrega (con fecha y hora).