

UNIVERSIDAD DE SEVILLA

IISSI

PROYECTO

---

# Modas Omaita

---

*Autores:*

Daniel González Corzo

Jesús Pineda Márquez

José Luis Mármol Romero

Roberto Hueso Gómez

22 de junio de 2017



Escuela Técnica Superior de  
Ingeniería Informática

# Índice

|  |          |
|--|----------|
| <b>1. Introducción</b>   | <b>3</b> |
| <b>2. Glosario de Términos</b>                                     | <b>3</b> |
| <b>3. Modelo de negocio</b>  |          |
| 3.1. Proceso de venta . . . . .                                    |          |
| 3.2. Proceso de compra al proveedor . . . . .                      |          |
| 3.3. Proceso de disponibilidad . . . . .                           |          |
| 3.4. Proceso de creación de socios . . . . .                       |          |
| <b>4. Visión General del Sistema</b>                               |          |
| 4.1. Descripción . . . . .   |          |
| <b>5. Catálogo de requisitos</b>                                   |          |
| 5.1. Requisitos generales . . . . .                                |          |
| 5.2. Requisitos de información . . . . .                           |          |
| 5.3. Requisitos funcionales . . . . .                              |          |
| 5.4. Requisitos no funcionales . . . . .                           |          |
| 5.5. Reglas de negocio . . . . .                                   |          |
| <b>6. Modelo Conceptual</b>  |          |
| 6.1. Diagramas de clases UML . . . . .                             |          |
| 6.2. Escenarios de prueba . . . . .                                |          |
| <b>7. Matrices de trazabilidad</b>                                 |          |
| 7.1. Pruebas de aceptación frente a requisitos . . . . .           |          |
| 7.2. Pruebas de aceptación frente a escenarios de prueba . . . . . |          |
| 7.3. Tipos de UML frente a Requisitos . . . . .                    |          |
| <b>8. Modelos relacionales</b>                                     |          |
| 8.1. 3FN Venta . . . . .   |          |
| 8.2. 3FN Traspaso - Pedido . . . . .                               |          |
| <b>9. Código SQL de la base de datos</b>                           |          |
| 9.1. Tablas . . . . .  |          |
| 9.2. Funciones y Procedures . . . . .                              |          |
| 9.3. Triggers . . . . .  |          |

|                                  |  |
|----------------------------------|--|
| 9.4. Pruebas . . . . .           |  |
| <b>10. Apéndice</b>              |  |
| 10.1. Actas de reunión . . . . . |  |

## 1. Introducción

Omaita Modas es una tienda situada en la localidad de Alcalá de Guadaira y más exactamente en la calle Pepe Luces nº20, la cual pertenece a una cadena de tiendas de ropa que se especializa en la venta de ropa y accesorios a un público maduro y femenino. Nuestro cliente busca ser una tienda puntera en su cadena gracias a que tiene a su disposición toda la atención del público de la localidad, ya que es la única de esta cadena en la misma. Actualmente nuestro cliente no pasa por la mejor situación en lo que a clientela se refiere, además de que carece de personal, por tanto, nuestro proyecto tiene como base ayudar a nuestro cliente en la gestión de la tienda con un sistema informático, el cual nos permita realizar pedidos a proveedores y visualizar productos en el stock de la otras tiendas de la cadena, y la creación de una página online donde sus clientes puedan visualizar y comprar un producto determinado en cualquier momento, con esto último se quiere conseguir ampliar la clientela de la tienda.



## 2. Glosario de Términos

| Término            | Descripción  |
|--------------------|--|
| Albarán de entrega | Documento obtenido en la recepción del pedido que verifica la correcta entrega del mismo.  |
| Almacén            | Almacén que tienen en común todas las tiendas de la cadena Omaita modas.   |
| Aviso              | Notificación o anuncio dado para comunicar de la falta o limitación.   |
| Consulta           | Actividad que se realiza para tratar de encontrar cualquier entidad almacenada en la base de datos.                                |
| Cadena             | Conjunto de tiendas relacionadas entre sí que ofrecen una mezcla estándar de productos, las cuales disponen de almacenes en común. |
| Categoría          | Tipo de producto que hay en la tienda.   |
| Emplazamiento      | Inmueble de la cadena, puede ser una tienda o el almacén   |
| Factura            | Documento en el que se plasman las compras realizadas además del número de referencia en la base de datos a la lista de compras.   |
| Pedido             | Petición de productos de un emplazamiento al proveedor.  |
| Proveedor          | Entidad económica a la cual varias empresas le compran el material que usarán en la misma o que luego lo venderán al por menor.    |
| Solicitud          | Petición de traspaso de un emplazamiento a otro  |
| Stock              | Conjunto de mercancías o productos que se tienen almacenados en espera de su venta o comercialización.                             |
| Stock mínimo       | 5 unidades de un producto almacenadas  |
| Traspaso           | Transferencia de uno o varios productos de un emplazamiento a otro   |

### 3. Modelo de negocio

#### 3.1. Proceso de venta

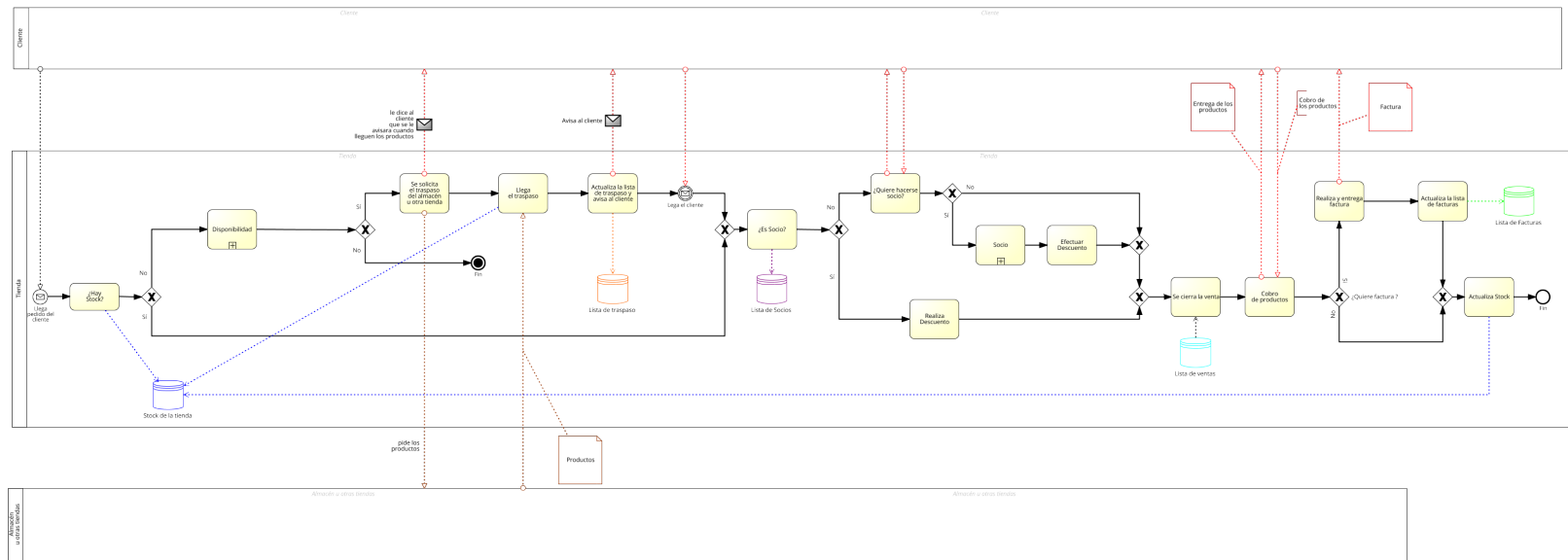
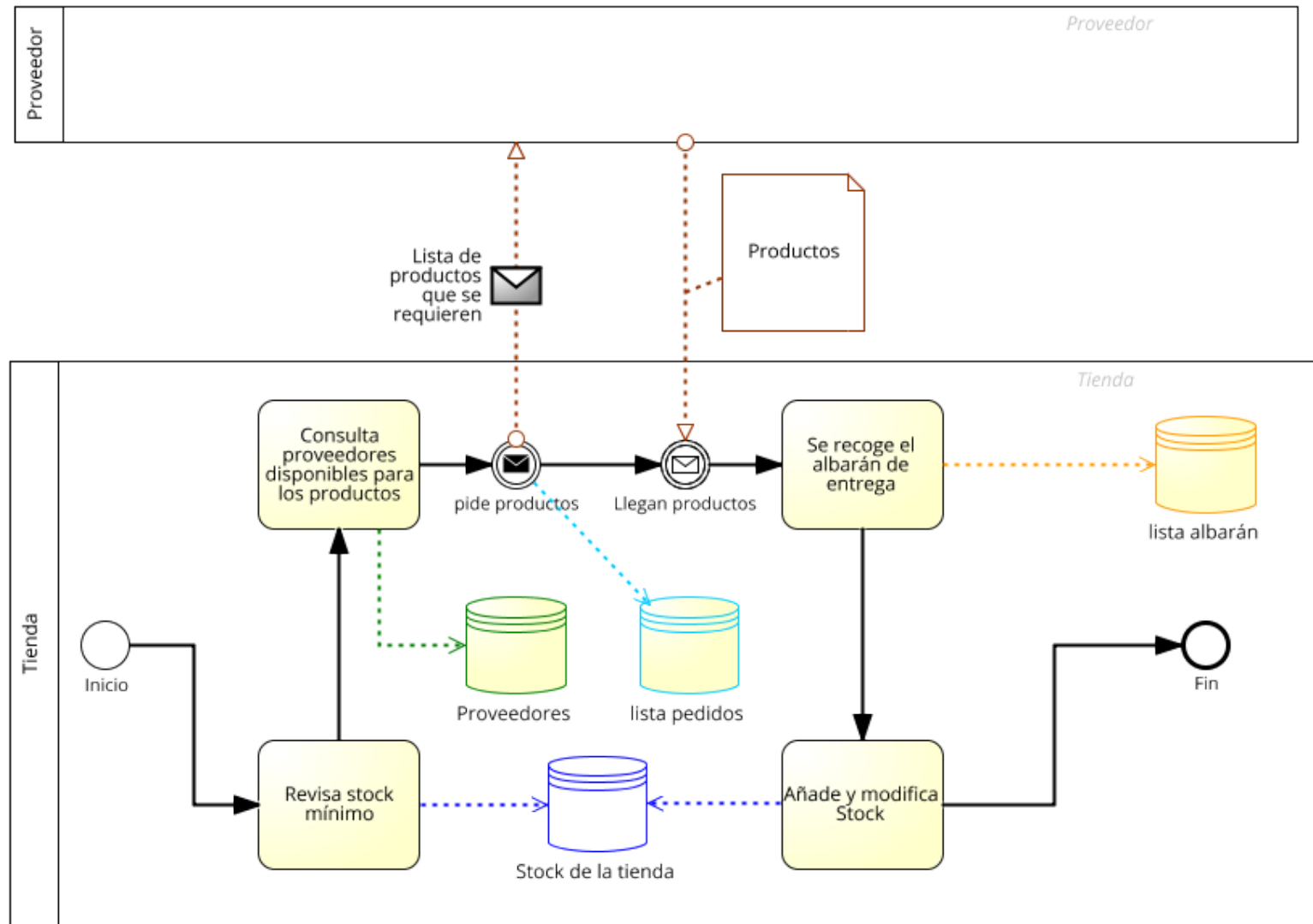


Figura 1: Proceso de venta



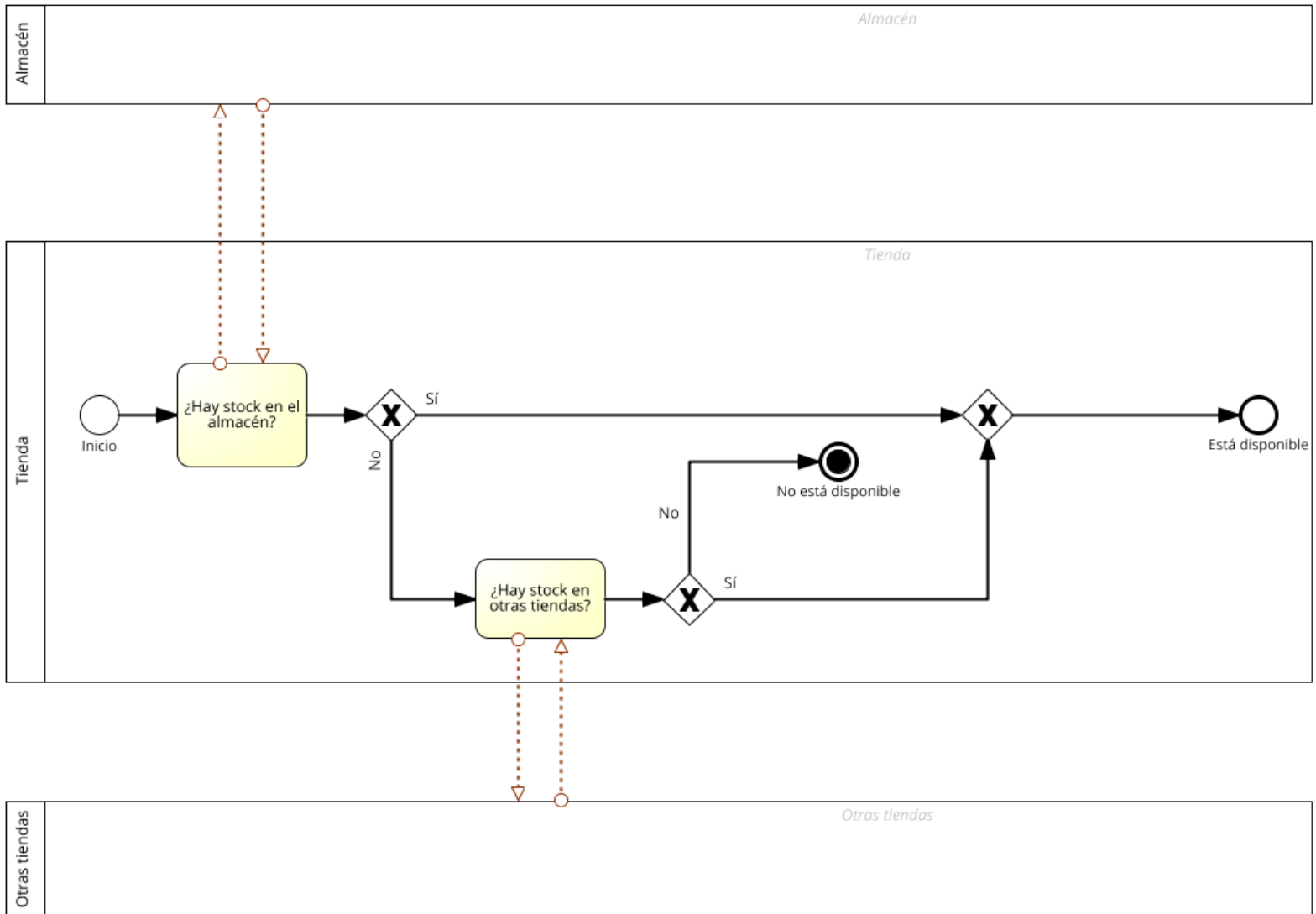
### 3.2. Proceso de compra al proveedor







### 3.3. Proceso de disponibilidad





### 3.4. Proceso de creación de socios

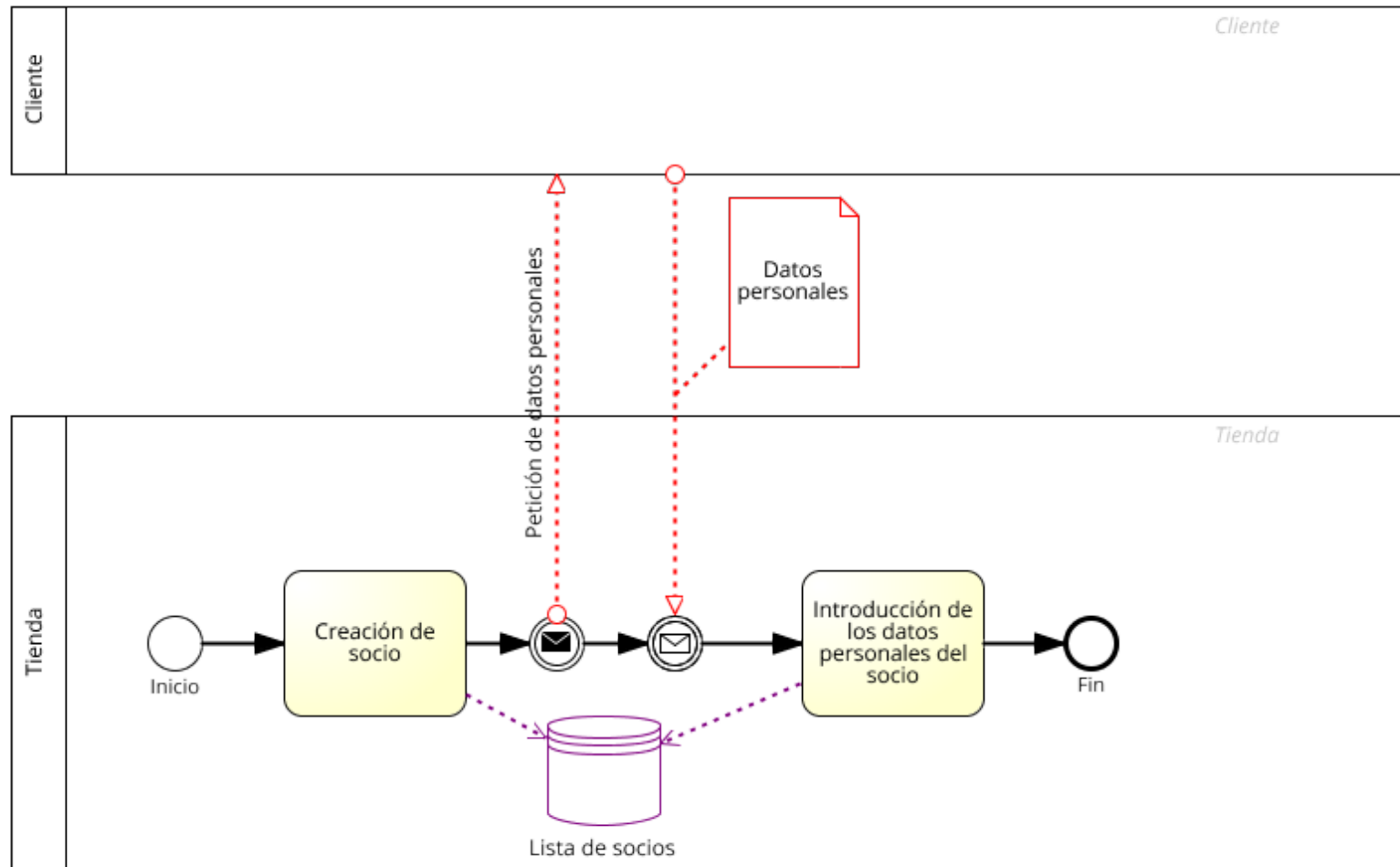


Figura 4: Proceso de creación de socios

## 4. Visión General del Sistema

### 4.1. Descripción

Para solucionar los problemas planteados de clientela y gestión, el sistema estará diseñado para permitir la gestión de las ventas, los productos, los pedidos, los traspasos y los clientes de manera más eficaz.

El objetivo principal de nuestro cliente es aumentar su clientela con la ayuda de una página web, esto lo haremos desarrollando un catálogo online de la tienda para que los clientes puedan consultar o comprar cualquier producto en cualquier momento, además de facilitar la gestión interna de la cadena.

Con todo esto principalmente lo que se quiere es aumentar los beneficios y hacer más eficiente la gestión de la cadena.

## 5. Catálogo de requisitos

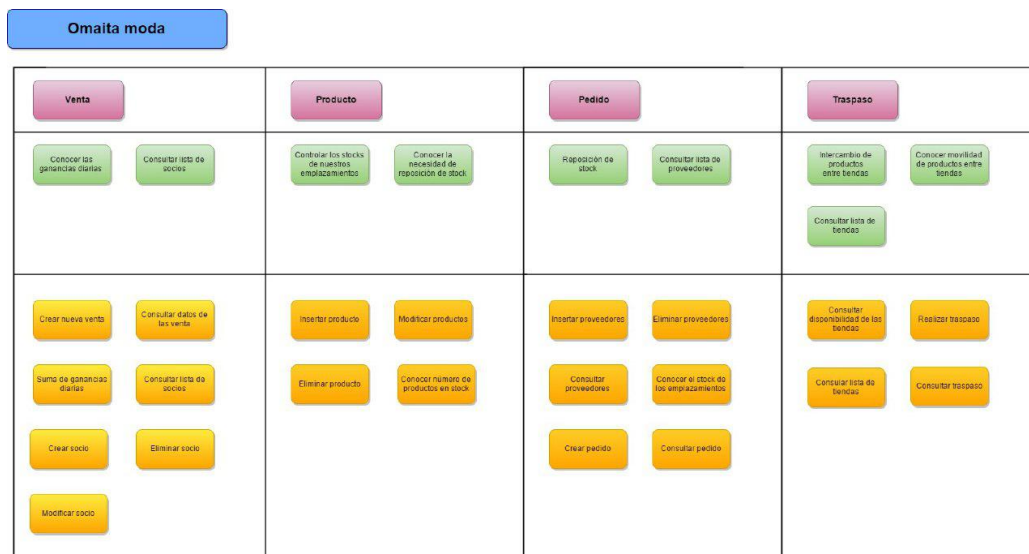


Figura 5: Mapa de requisitos

## 5.1. Requisitos generales

---

### RG-01 - Gestion ventas

- **Como** Propietario de la cadena
- **Quiero** poder gestionar y consultar las ventas de las tiendas
- **Para** llevar una buena gestión de ellas

---

### RG-02 - Gestion proveedores

- **Como** Propietario de la cadena
- **Quiero** poder gestionar y consultar los pedidos a proveedores
- **Para** llevar una buena gestión de ellos

---

### RG-03 - Traspasos

- **Como** Propietario de la cadena
- **Quiero** que haya un sistema de traspasos
- **Para** poder aprovechar al máximo el stock de las tiendas

---

### RG-04 - Catálogo

- **Como** Cliente
- **Quiero** poder ver una lista de los productos ofrecidos y disponibles
- **Para** realizar mis compras

## 5.2. Requisitos de información

---

### RI-01 - Información sobre ventas

- **Como** Propietario de la cadena
- **Quiero** que el sistema almacene la información correspondiente a las ventas, guardando así los siguientes datos
  - Fecha en la que se llevó acabo la venta.
  - La relación entre producto y venta que almacenará: el producto vendido, el precio y el IVA del mismo en el momento en el que se vendió.
- **Para** conocer los datos de las ventas realizadas.

### P-RI-01

1. Cuando se cree una nueva venta, si los datos introducidos son correctos, se almacenará en la lista de ventas.
2. En caso de que algún dato sea incorrecto no se almacenara.

---

### RI-02 - Información sobre facturas

- **Como** propietario de la cadena
- **Quiero** que el sistema almacene la información correspondiente a las facturas, guardando así los siguientes datos:
  - Fecha en la que se tramita la factura.
  - Número de la factura.
  - Un campo llamado devuelto, usado para las devoluciones, si el campo tiene el valor false, entonces podemos contabilizar esa factura sin problemas, si el campo tiene el valor true significará que el dinero fue devuelto al cliente y por la tanto no podríamos contabilizarlo.
- **Para** conocer los datos de las facturas expedidas.

## **P-RI-02**

1. Cuando se cree una nueva factura, si los datos introducidos son correctos, se almacenará en la lista de factura.
2. En caso de que algún dato sea incorrecto no se almacenara.

---

## **RI-03 - Información sobre socios**

- **Como** propietario de la cadena
- **Quiero** que el sistema almacene la información de los clientes que son socios, guardando los siguientes datos en él:
  - Nombre.
  - Apellidos.
  - DNI.
  - Dirección.
  - Fecha de Nacimiento.
  - Email.
- **Para** llevar un control de los clientes que son socios y así poderles aplicar descuentos en sus compras.

## **P-RI-03**

1. Cuando se cree una nueva factura, si los datos introducidos son correctos, se almacenará en la lista de factura.
2. En caso de que algún dato sea incorrecto (como por ejemplo un DNI duplicado) no se almacenara.

---

## **RI-04 - Información sobre productos**

- **Como** propietario de la cadena
- **Quiero** que los productos que venda la cadena se guarden con las siguientes características en el sistema:



- Nombre.
  - Una breve descripción.
  - La categoría del producto: abrigos, chaquetas, camisas, camisetas, jersey, vestidos, faldas, pantalones, calzado, accesorios y bisutería.
  - Precio del producto.
  - IVA actual, para controlar el IVA en todo momento
- **Para** conocer los datos de cada producto.

#### **P-RI-04**

1. Cuando se inserte un nuevo producto, si los datos introducidos son correctos, se almacenará en la lista de productos.
2. En caso de que algún dato sea incorrecto (como por ejemplo un precio negativo duplicado) no se almacenara.

---

#### **RI-05 - Stock del producto**

- **Como** propietario de la cadena
- **Quiero** saber el stock de cada producto en cada tienda de la cadena y del almacén
- **Para** controlar el stock de la cadena

#### **P-RI-05**

1. Actualizar la lista de stocks cuando se cree uno nuevo o se modifique alguno ya existente.

---

#### **RI-06 - Información sobre los emplazamientos**

- **Como** propietario de la cadena
- **Quiero** guardar la dirección de las tiendas y del almacén de la cadena, además de su teléfono de contacto
- **Para** conocer la localización de las mismas

## **P-RI-06**

1. Cuando se inserte un nuevo emplazamiento, si los datos introducidos son correctos, se almacenará en la lista de emplazamientos.
2. En caso de que algún dato sea incorrecto (como por ejemplo una dirección inexistente) no se almacenara.

---

## **RI-07 - Información sobre los proveedores**

- **Como** propietario de la cadena
- **Quiero** disponer de la siguiente información sobre proveedores:
  - Nombre.
  - CIF.
  - Número de teléfono.
  - Email.
- **Para** saber a qué proveedores puede realizar los pedidos

## **P-RI-07**

1. Cuando se inserte un nuevo proveedor, si los datos introducidos son correctos, se almacenará en la lista de proveedores.
2. En caso de que algún dato sea incorrecto no se almacenara.

---

## **RI-08 - Información sobre los pedidos**

- **Como** propietario de la cadena
- **Quiero** que el sistema almacene los pedidos guardando los siguientes datos:
  - Fecha en la que se realiza.
  - Una asociación que controla la cantidad de cada producto del pedido, el precio y el IVA.
- **Para** conocer los datos de los pedidos realizados.

## **P-RI-08**

1. Cuando se inserte un nuevo pedido, si los datos introducidos son correctos, se almacenará en la lista de pedidos.
2. En caso de que algún dato sea incorrecto no se almacenara.

---

## **RI-09 - Información sobre traspasos**

- **Como** propietario de la cadena
- **Quiero** que el sistema guarde los traspasos almacenando los siguientes datos:
  - Fecha de traspaso.
  - Asociación que controla la cantidad de cada producto en el traspaso.
- **Para** conocer los datos de los traspasos realizados.

## **P-RI-09**

1. Cuando se inserte un nuevo traspaso, si los datos introducidos son correctos, se almacenará en la lista de traspasos.
2. En caso de que algún dato sea incorrecto no se almacenara.

---

## **RI-10 - Información sobre el albarán**

- **Como** propietario de la cadena
- **Quiero** que el sistema almacene información correspondiente a los albaranes, guardando así los siguientes datos
  - Fecha de firma del albarán.
- **Para** tener un control de los pedidos recibidos los cuales son controlados por los albaranes.

## **P-RI-10**

1. Cuando se inserte un nuevo albarán, si los datos introducidos son correctos, se almacenará en la lista de albaranes.
2. En caso de que algún dato sea incorrecto no se almacenara.

---

**RI-11** - Información sobre solicitud

- **Como** propietario de la cadena
- **Quiero** que el sistema almacene información correspondiente a las solicitudes de traspaso, guardando así los siguientes datos
  - Fecha de solicitud.
  - Asociación que controla la cantidad de cada producto en la solicitud
- **Para** conocer los datos de las solicitudes realizadas.

**P-RI-11**

1. Cuando se inserte un nuevo pedido, si los datos introducidos son correctos, se almacenará en la lista de pedidos.
2. En caso de que algún dato sea incorrecto no se almacenara.

### 5.3. Requisitos funcionales

---

**RF-01** - Crear ventas

- **Como** empleado
- **Quiero** poder crear y consultar ventas realizadas
- **Para** poder así conocer las ganancias de la tienda

**P-RF-01**

1. Crear una nueva venta con nuevos datos y tener la posibilidad de acceder a los datos de las ventas ya realizadas.
2. No se creará la nueva venta si algún dato introducido no es válido.

---

**RF-02** - Actualizar stocks tras venta

- **Como** propietario de la cadena
- **Quiero** que tras una factura asociada a una venta el stock se actualice
- **Para** controlar el stock de manera correcta

**P-RF-02**

1. Tras realizar una factura el stock se actualiza en función de los parámetros de las ventas asociadas a dicha factura.

---

**RF-03** - Crear socios

- **Como** empleado
- **Quiero** poder crear socios en una lista y poder consultarla
- **Para** llevar así un control sobre estos y saber si se tiene que aplicar o no el descuento.

**P-RF-03**

1. Crear un socio rellenando un formulario con los datos del cliente en cuestión.
2. Poder acceder a la lista de los socios ya realizados.
3. No se creará el nuevo socio si algún dato introducido no es válido.

---

**RF-04** - Modificar socios

- **Como** empleado
- **Quiero** poder modificar la dirección y el email de un socio ya creado
- **Para** tener los datos correctos del socio.

**P-RF-04**

1. Cuando se cambie algún dato de un socio, este se actualizará en la lista de socios.
2. o se efectuará la actualización si alguno de los datos introducidos para el cambio no es valido.

---

#### **RF-05 - Eliminar socios**

- **Como** empleado
- **Quiero** poder eliminar un socio de la base de datos
- **Para** dejar de tener almacenados los datos de un cliente que desiste de su derecho de ser socio.

#### **P-RF-05**

1. Cuando se elimine un socio, la lista de socios se actualizará de forma de que el socio eliminado ya no aparezca en esta.

---

#### **RF-06 - Crear, modificar y eliminar productos**

- **Como** empleado
- **Quiero** poder insertar, modificar la descripción, el precio y el IVA de un producto, o eliminar un producto de mi base de datos
- **Para** poder así llevar una buena gestión de los productos.

#### **P-RF-06**

1. Insertar en la lista de productos un nuevo producto rellenando un formulario con los datos de este, y tras ello se actualizará dicha lista apareciendo en ella el nuevo producto.
2. Cuando se cambie algún dato de un producto existente, este se actualizará en la lista de productos.
3. Cuando se elimine un producto, la lista de productos se actualizará de forma de que el producto eliminado ya no aparezca en esta.

4. Si algunos de los datos introducidos a la hora de insertar o actualizar un producto es incorrecto no se realizará la operación.

---

#### **RF-07 - Crear y modificar stocks**

- **Como** empleado
- **Quiero** poder crear el stock de un producto y modificar su cantidad
- **Para** llevar un control sobre la disponibilidad de los productos según su emplazamiento.

#### **P-RF-07**

1. Cuando llegue un producto nuevo a la tienda se creará un stock de este con la cantidad que llegue del mismo.
2. Cuando se realice cualquier interacción con un producto de tal forma que modifique la cantidad que existe en la tienda del mismo, se actualizara el stock.
3. No se creará el nuevo stock si algún dato introducido no es válido, o si el producto al que se quiere realizar un stock ya posee uno en la tienda en cuestión.

---

#### **RF-08 - Crear, modificar y eliminar proveedores**

- **Como** propietario de la cadena
- **Quiero** poder insertar, modificar el nombre, el email y el teléfono, o eliminar proveedores en mi base de datos
- **Para** conocer los proveedores disponibles y tener sus datos actualizados.

#### **P-RF-08**

1. Insertar en la lista de proveedores un nuevo proveedor rellenando un formulario con los datos de este, y tras ello se actualizará dicha lista apareciendo en ella el nuevo proveedor.

2. Cuando se cambie algún dato de un proveedor existente, este se actualizará en la lista de proveedores.
3. Cuando se elimine un proveedor, la lista de proveedores se actualizará de forma de que el proveedor eliminado ya no aparezca en esta.
4. Si algunos de los datos introducidos a la hora de insertar o actualizar un producto es incorrecto no se realizará la operación.

---

#### **RF-09 - Crear pedidos**

- **Como** empleado
- **Quiero** poder crear pedidos
- **Para** tener constancia de todos los pedidos realizados por cada uno de mis emplazamientos

#### **P-RF-09**

1. Crear un nuevo pedido con los productos que sean necesario reponer.
2. No se podrá realizar un pedido si algún dato introducido no es válido.

---

#### **RF-10 - Actualizar stock tras pedido**

- **Como** propietario de la cadena
- **Quiero** que tras recibir el albarán que confirma la entrega del pedido el stock se actualice
- **Para** controlar correctamente el stock.

#### **P-RF-10**

1. Tras recibir el pedido realizado, se actualizará la cantidad del stock de los productos del pedido.

---

#### **RF-11 - Crear solicitud de traspaso**

- **Como** empleado



- **Quiero** poder crear una solicitud de traspaso
- **Para** así poder pedir a otra tienda productos que necesite.

#### **P-RF-11**

1. Poder realizar una solicitud de traspaso a otra tienda de un producto que sea necesario y esté disponible en la otra tienda.

---

#### **RF-12 - Crear traspasos**

- **Como** empleado
- **Quiero** poder crear traspasos
- **Para** responder a la necesidad de las solicitudes de traspaso.

#### **P-RF-12**

1. Responder a la otra tienda siempre que sea posible cumplir con sus peticiones de solicitud de traspaso, creando en el acto un traspaso con los datos de los productos traspasados.

---

#### **RF-13 - Actualizar stock tras traspaso**

- **Como** propietario de la cadena
- **Quiero** que cuando se realice un traspaso, se modifiquen de manera correcta los stocks de las tiendas implicadas
- **Para** así poder tener una buena gestión sobre nuestros stocks

#### **P-RF-13**

1. Cuando un traspaso se realiza de manera correcta ambas tiendas implicadas en el traspaso deberán actualizar la cantidad del stock de los productos implicados en el traspaso.

---

#### **RF-14 - Consultar traspasos**

- **Como** propietario de la cadena

- **Quiero** poder consultar todos los trasposos realizados por mis emplazamientos
- **Para** conocer la movilidad de los productos entre estos.

#### **P-RF-14**

1. Siempre se tendrá la posibilidad de acceder a los datos de los trasposos en los que esté implicado mi tienda.

---

#### **RF-15 - Crear y eliminar emplazamientos**

- **Como** propietario de la cadena
- **Quiero** poder añadir o eliminar emplazamientos en la base de datos
- **Para** saber que emplazamientos están en la cadena.

#### **P-RF-15**

1. Insertar en la lista de emplazamientos un nuevo emplazamiento rellenando un formulario con los datos de este, y tras ello se actualizará dicha lista apareciendo en ella el nuevo emplazamiento.
2. Cuando se elimine un emplazamiento, la lista de emplazamientos se actualizará de forma de que el emplazamiento eliminado ya nTras una devolución se marcará en la factura como que se ha realizado dicha acción modificando el campo correspondiente.
3. En el caso de que la devolución solo sea parcial y no total se creara una copia de la factura donde solo aparezcan los productos no devueltos y la antigua se marcara como devuelta.
4. En ambos casos se actualizará el stock de los productos devueltos.
5. Si se intenta realizar la devolución de una factura, la cual haya excedido el tiempo de devolución, no se realizará dicha devolución.o aparezca en esta.

---

#### **RF-16 - Modificar emplazamientos**

- **Como** propietario de la cadena
- **Quiero** poder modificar la dirección y el número de teléfono de un emplazamiento
- **Para** tener los datos actualizados de mis emplazamientos.

#### **P-RF-16**

1. Cuando se cambie algún dato de un emplazamiento existente, este se actualizará en la lista de emplazamientos.
2. Si al modificar algún emplazamiento se introduce algún dato no valido, no se actualizará la lista de emplazamientos.

---

#### **RF-17 - Devolución**

- **Como** propietario de la cadena
- **Quiero** que cuando se realiza una devolución el campo devuelto de las facturas pase a ser True, y si es necesario que el empleado cree de nuevo toda la venta y la factura pero sin los productos devueltos, esta factura deberá tener el campo devuelto como False
- **Para** tener un control sobre las devoluciones.

#### **P-RF-17**

1. Tras una devolución se marcará en la factura como que se ha realizado dicha acción modificando el campo correspondiente.
2. En el caso de que la devolución solo sea parcial y no total se creara una copia de la factura donde solo aparezcan los productos no devueltos y la antigua se marcara como devuelta.
3. En ambos casos se actualizará el stock de los productos devueltos.
4. Si se intenta realizar la devolución de una factura, la cual haya excedido el tiempo de devolución, no se realizará dicha devolución.

---

#### **RF-18 - Crear albarán**

- **Como** empleado
- **Quiero** crear una copia del albarán de entrega que me ha dado el proveedor al traer el pedido que habíamos hecho
- **Para** tener la información de los albaranes recogida en la base de datos.

#### **P-RF-18**

1. Cuando llegue cualquier pedido se guardará una copia de los albaranes

### **5.4. Requisitos no funcionales**

---

#### **RNF-01 - Acceso al sistema**

- **Como** propietario de la cadena
- **Quiero** que todos mis empleados tengan acceso al sistema
- **Para** facilitar la gestión de las tiendas y el control del sistema.

---

#### **RNF-02 - Protección de datos socios**

- **Como** propietario de la cadena
- **Quiero** que los datos de los socios permanezcan privados y seguros
- **Para** cumplir la Ley de Protección de Datos

---

#### **RNF-03 - Mantenimiento**

- **Como** propietario de la cadena
- **Quiero** realizar el mantenimiento del sistema al menos una vez al mes
- **Para** prevenir problemas mayores en el sistema.

## 5.5. Reglas de negocio

---

### RN-01 - Descuento a socios

- **Como** propietario de la cadena
- **Quiero** aplicar un 5 % de descuento en las ventas realizadas a socios
- **Para** recompensar su fidelidad

#### P-RN-01

1. Cuando se realice una venta, si esta está asociada a un socio , se aplicara el descuento en la factura.

---

### RN-02 - Tamaño mínimo de pedidos

- **Como** propietario de la cadena quiero
- **Quiero** que los pedidos sean como mínimo de 20 unidades por producto
- **Para** abaratar gastos de transporte.

#### P-RN-02

1. Cuando se realice un pedido, si la cantidad de algún producto no supera el mínimo no se realizará dicho pedido, en caso contrario se realizara sin ningún problema.

---

### RN-03 - Evitar traspaso si se provoca stock mínimo

- **Como** propietario de la cadena
- **Quiero** que las tiendas no puedan ofrecer el traspaso de un producto, si debido al traspaso el producto llega a su stock mínimo
- **Para** evitar que éstas se queden desabastecidas

#### P-RN-03

1. No se podrá responder a una solicitud de traspaso si en caso de realizar el traspaso el producto solicitado se nos quedaría en stock mínimo

---

**RN-04** - Política de devoluciones

- **Como** propietario de la cadena
- **Quiero** que solo se acepten devoluciones, con un plazo máximo de 30 días después de la compra
- **Para** dificultar la devolución de productos usados

**P-RN-04**

1. Si se intenta realizar una devolución dentro del plazo de devolución establecido por la tienda (en nuestro caso 30 días a partir del día de la compra) se permite realizar dicha devolución ya sea total o parcial.
2. En caso contrario no se podrá realizar la devolución.

---

**RN-05** - Aviso stock mínimo

- **Como** trabajador de una tienda
- **Quiero** obtener un aviso cuando el stock de un producto llegue al stock mínimo
- **Para** evitar el desabastecimiento de un producto

**P-RN-05**

1. Cuando un producto este bajo el stock mínimo saltara un mensaje de aviso para tener en cuenta su futura reposición

## 6. Modelo Conceptual

### 6.1. Diagramas de clases UML

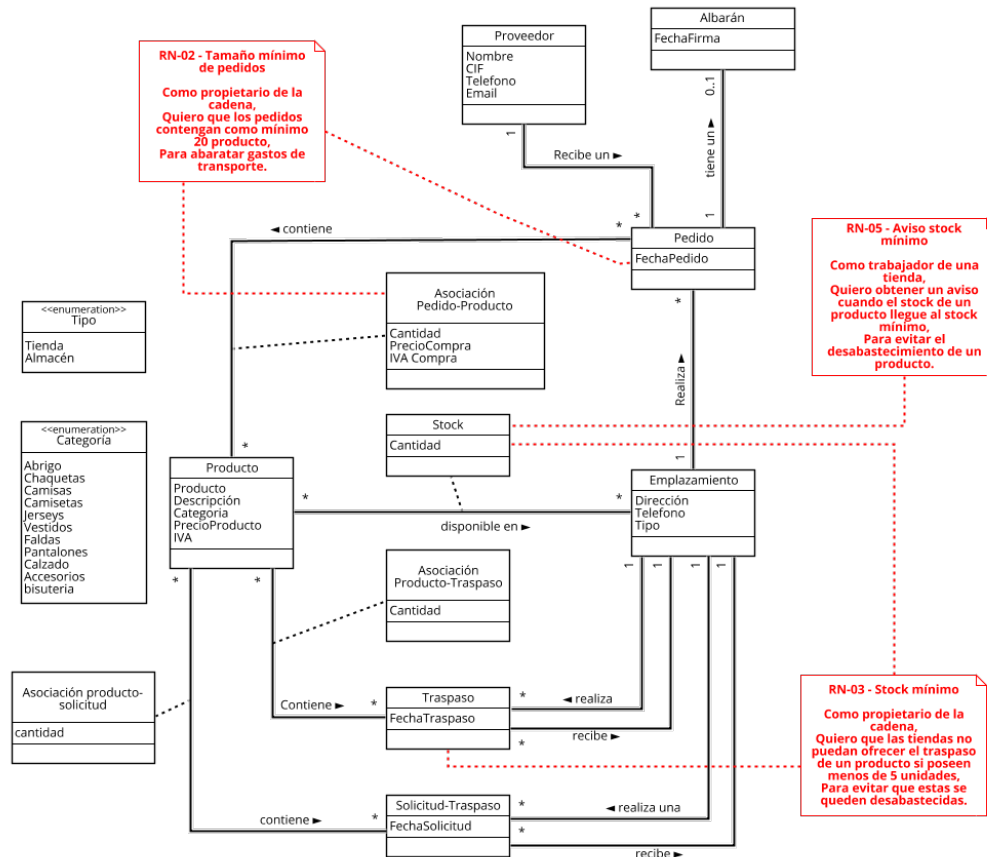


Figura 6: UML Traspaso / Pedido

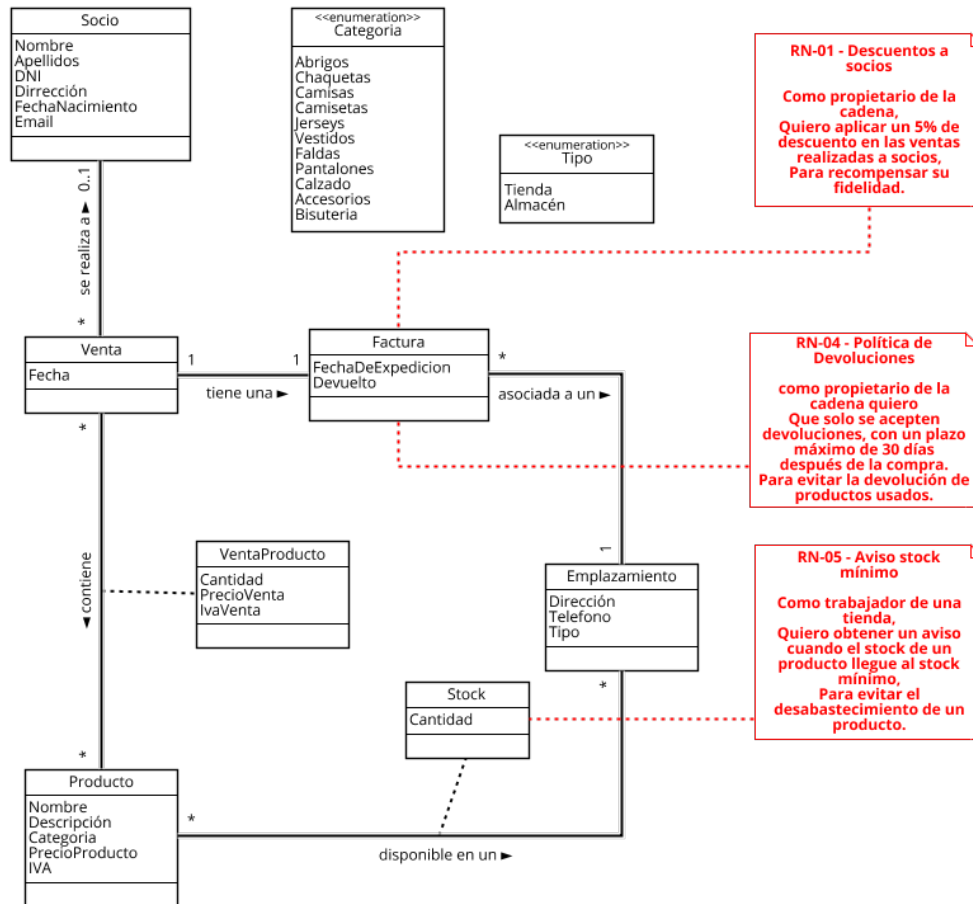


Figura 7: UML Venta



## 6.2. Escenarios de prueba

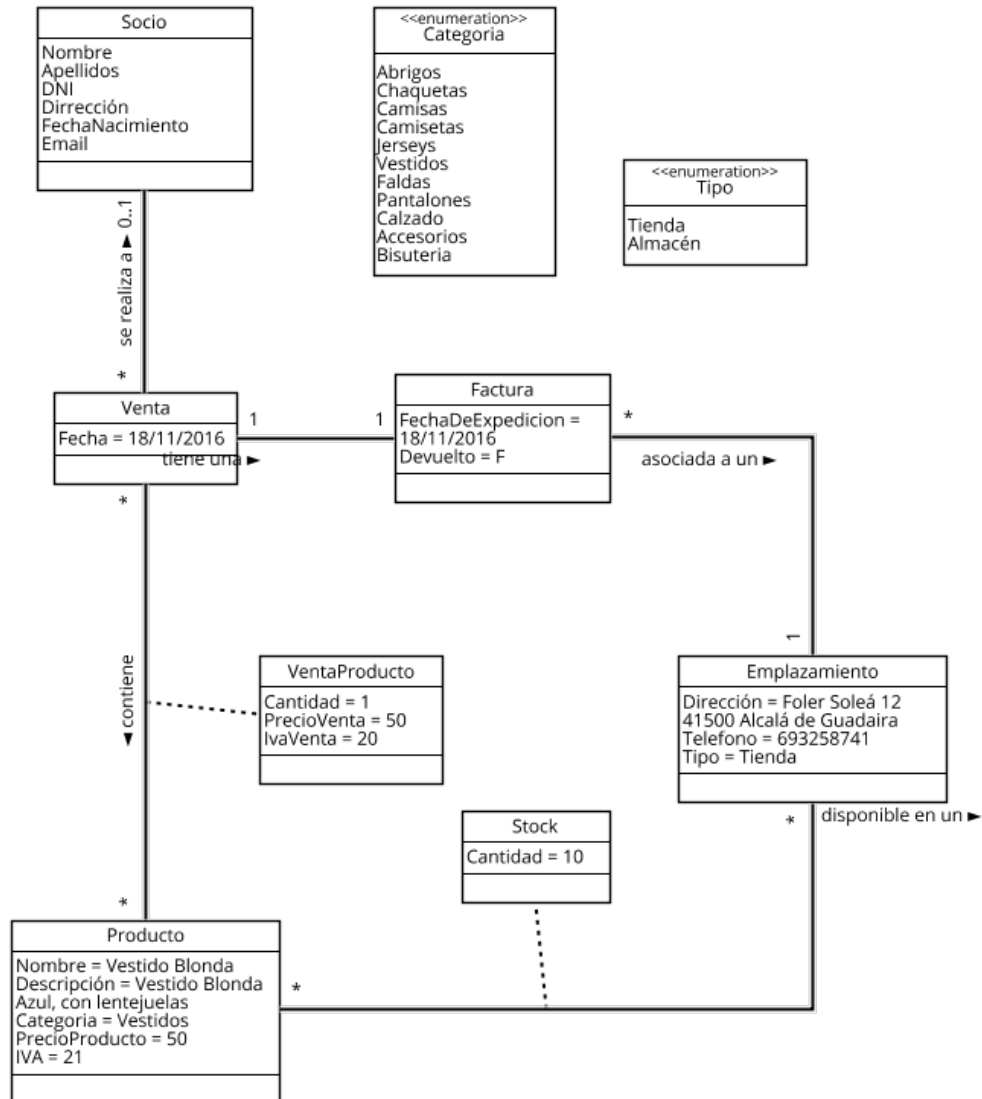


Figura 8: UML Venta NO Socio

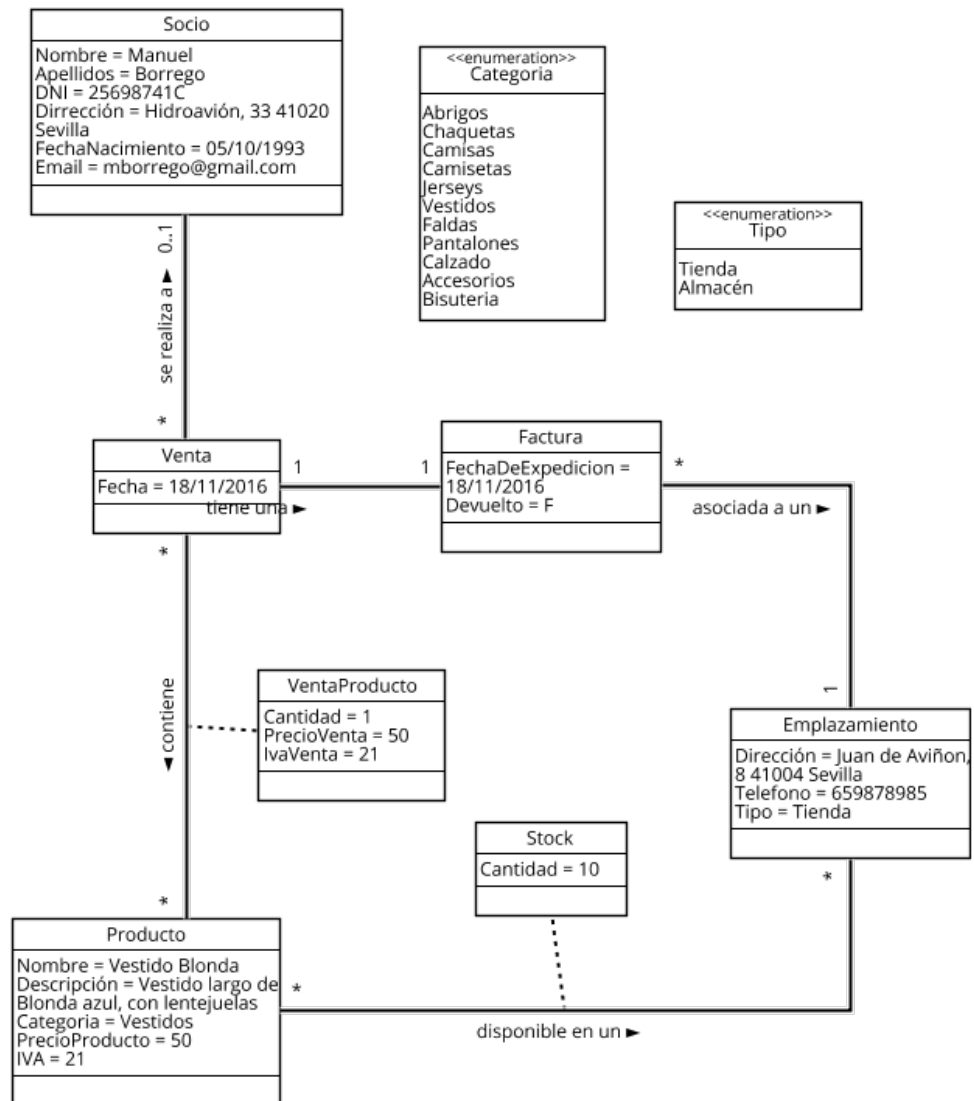


Figura 9: UML Venta Socio

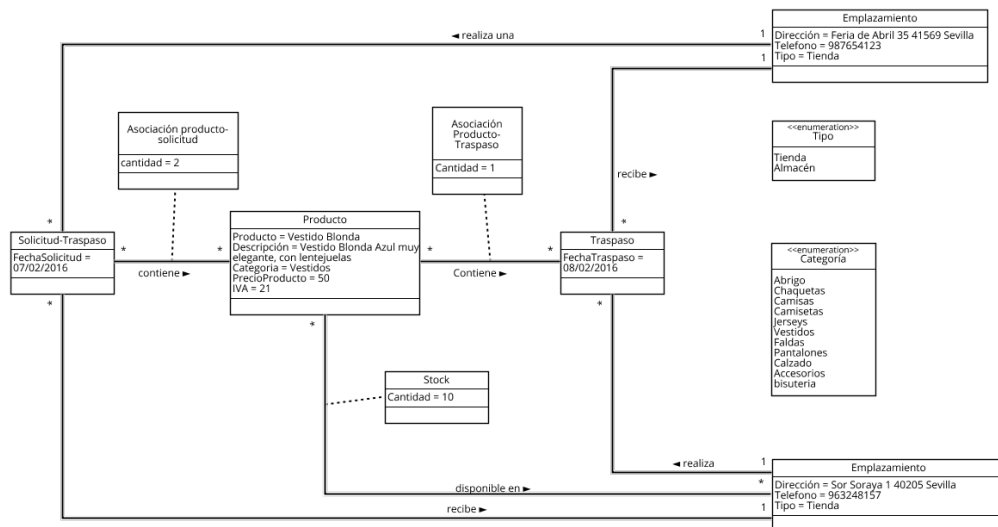


Figura 10: UML Traspaso Tienda - Tienda

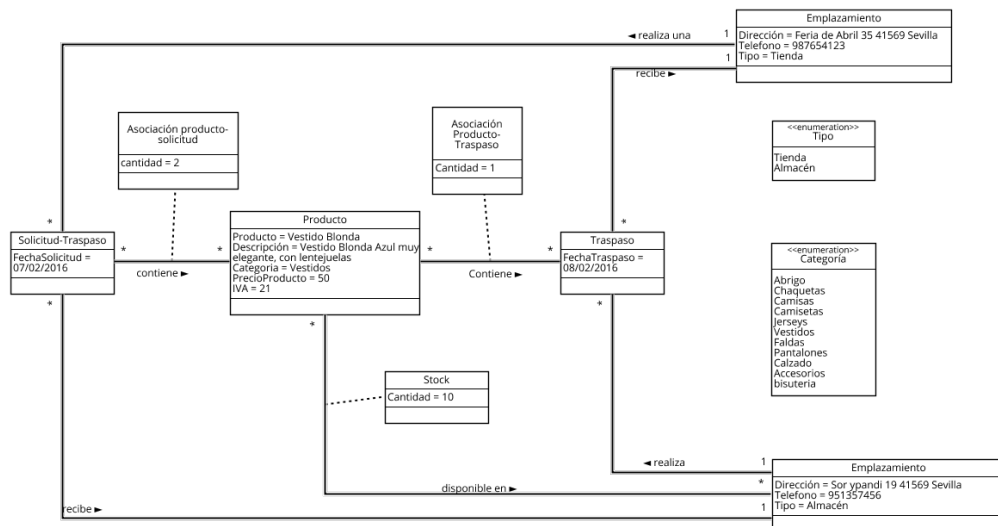


Figura 11: UML Traspaso Tienda - Almacén

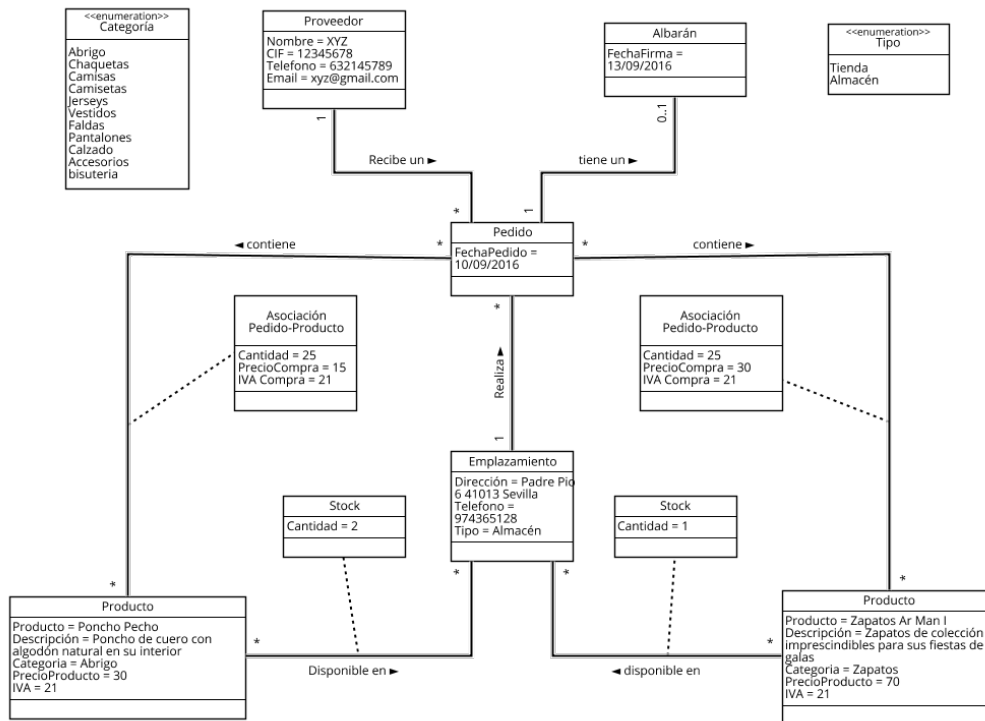


Figura 12: UML Pedido Tienda - Proveedor

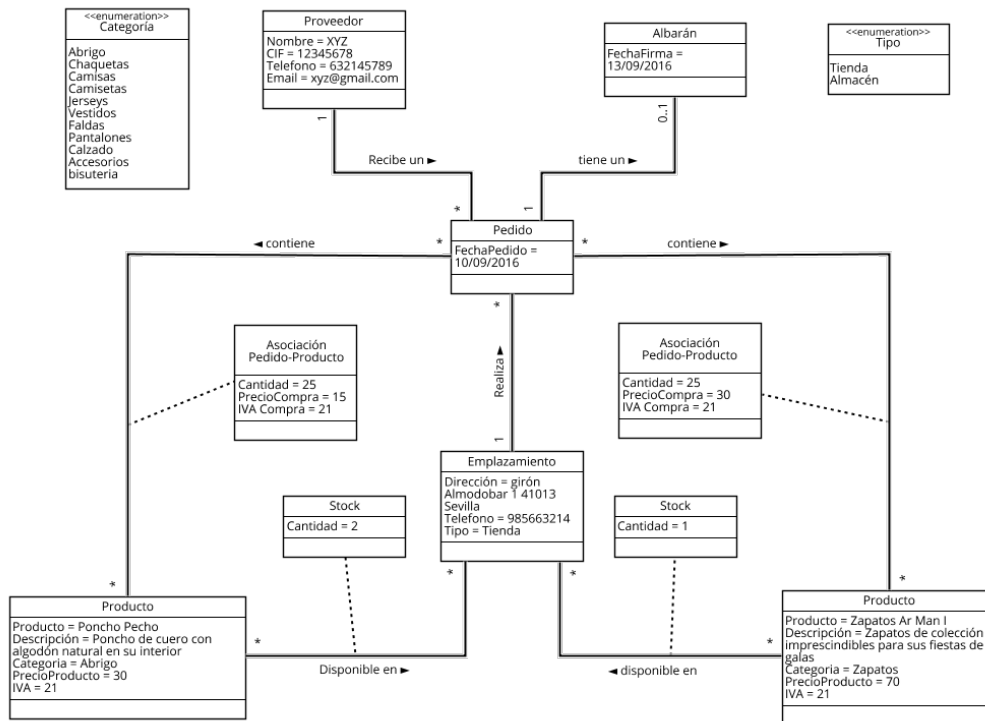


Figura 13: UML Pedido Almacén - Proveedor

## 7. Matrices de trazabilidad

### 7.1. Pruebas de aceptación frente a requisitos

Cuadro 1: Requisitos funcionales

|         |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| P-RF-01 | X  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| P-RF-02 |    | X  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| P-RF-03 |    |    | X  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| P-RF-04 |    |    |    | X  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| P-RF-05 |    |    |    |    | X  |    |    |    |    |    |    |    |    |    |    |    |    |    |
| P-RF-06 |    |    |    |    |    | X  |    |    |    |    |    |    |    |    |    |    |    |    |
| P-RF-07 |    |    |    |    |    |    | X  |    |    |    |    |    |    |    |    |    |    |    |
| P-RF-08 |    |    |    |    |    |    |    | X  |    |    |    |    |    |    |    |    |    |    |
| P-RF-09 |    |    |    |    |    |    |    |    | X  |    |    |    |    |    |    |    |    |    |
| P-RF-10 |    |    |    |    |    |    |    |    |    | X  |    |    |    |    |    |    |    |    |
| P-RF-11 |    |    |    |    |    |    |    |    |    |    | X  |    |    |    |    |    |    |    |
| P-RF-12 |    |    |    |    |    |    |    |    |    |    |    | X  |    |    |    |    |    |    |
| P-RF-13 |    |    |    |    |    |    |    |    |    |    |    |    | X  |    |    |    |    |    |
| P-RF-14 |    |    |    |    |    |    |    |    |    |    |    |    |    | X  |    |    |    |    |
| P-RF-15 |    |    |    |    |    |    |    |    |    |    |    |    |    |    | X  |    |    |    |
| P-RF-16 |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | X  |    |    |
| P-RF-17 |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | X  |    |
| P-RF-18 |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | X  |
|         | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |

Cuadro 2: Requisitos de información

|         |    |    |    |    |    |    |    |    |    |    |    |   |
|---------|----|----|----|----|----|----|----|----|----|----|----|---|
| P-RI-01 | X  |    |    |    |    |    |    |    |    |    |    |   |
| P-RI-02 |    | X  |    |    |    |    |    |    |    |    |    |   |
| P-RI-03 |    |    | X  |    |    |    |    |    |    |    |    |   |
| P-RI-04 |    |    |    | X  |    |    |    |    |    |    |    |   |
| P-RI-05 |    |    |    |    | X  |    |    |    |    |    |    |   |
| P-RI-06 |    |    |    |    |    | X  |    |    |    |    |    |   |
| P-RI-07 |    |    |    |    |    |    | X  |    |    |    |    |   |
| P-RI-08 |    |    |    |    |    |    |    | X  |    |    |    |   |
| P-RI-09 |    |    |    |    |    |    |    |    | X  |    |    |   |
| P-RI-10 |    |    |    |    |    |    |    |    |    | X  |    |   |
| P-RI-11 |    |    |    |    |    |    |    |    |    |    |    | X |
|         | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 |   |

**Cuadro 3: Reglas de negocio**

|                |           |           |           |           |           |
|----------------|-----------|-----------|-----------|-----------|-----------|
| <b>P-RN-01</b> | X         |           |           |           |           |
| <b>P-RN-02</b> |           | X         |           |           |           |
| <b>P-RN-03</b> |           |           | X         |           |           |
| <b>P-RN-04</b> |           |           |           | X         |           |
| <b>P-RN-05</b> |           |           |           |           | X         |
|                | <b>01</b> | <b>02</b> | <b>03</b> | <b>04</b> | <b>05</b> |

## 7.2. Pruebas de aceptación frente a escenarios de prueba

**Cuadro 4: Escenarios**

|              |         |         |         |         |         |         |         |         |         |         |         |
|--------------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| Escenario 01 | X       | X       | X       | X       | X       | X       |         |         |         |         |         |
| Escenario 02 | X       | X       | X       | X       | X       | X       |         |         |         |         |         |
| Escenario 03 |         |         |         | X       | X       | X       |         |         | X       |         |         |
| Escenario 04 |         |         |         | X       | X       | X       |         |         | X       |         |         |
| Escenario 05 |         |         |         | X       | X       | X       | X       | X       |         | X       | X       |
| Escenario 06 |         |         |         | X       | X       | X       | X       | X       |         | X       | X       |
|              | P-RI-01 | P-RI-02 | P-RI-03 | P-RI-04 | P-RI-05 | P-RI-06 | P-RI-07 | P-RI-08 | P-RI-09 | P-RI-10 | P-RI-11 |

**Cuadro 5: Escenarios**

|              |         |         |         |         |         |         |         |         |         |         |         |         |         |         |         |         |         |         |
|--------------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| Escenario 01 | X       | X       |         |         |         | X       |         |         |         |         |         |         |         |         | X       | X       | X       |         |
| Escenario 02 | X       | X       | X       | X       | X       | X       |         |         |         |         |         |         |         |         | X       | X       | X       |         |
| Escenario 03 |         |         |         |         |         | X       |         |         |         |         | X       | X       | X       | X       | X       | X       |         |         |
| Escenario 04 |         |         |         |         |         | X       | X       | X       | X       | X       | X       | X       | X       | X       | X       | X       |         |         |
| Escenario 05 |         |         |         |         |         | X       | X       | X       | X       | X       |         |         |         |         | X       | X       |         | X       |
| Escenario 06 |         |         |         |         |         | X       | X       | X       | X       | X       |         |         |         |         | X       | X       |         | X       |
|              | P-RF-01 | P-RF-02 | P-RF-03 | P-RF-04 | P-RF-05 | P-RF-06 | P-RF-07 | P-RF-08 | P-RF-09 | P-RF-10 | P-RF-11 | P-RF-12 | P-RF-13 | P-RF-14 | P-RF-15 | P-RF-16 | P-RF-17 | P-RF-18 |

**Cuadro 6: Escenarios**

|                     |                |                |                |                |                |
|---------------------|----------------|----------------|----------------|----------------|----------------|
| <b>Escenario 01</b> | X              |                |                | X              | X              |
| <b>Escenario 02</b> | X              |                |                | X              | X              |
| <b>Escenario 03</b> |                |                | X              |                | X              |
| <b>Escenario 04</b> |                |                | X              |                | X              |
| <b>Escenario 05</b> |                | X              |                |                | X              |
| <b>Escenario 06</b> |                | X              |                |                | X              |
|                     | <b>P-RN-01</b> | <b>P-RN-02</b> | <b>P-RN-03</b> | <b>P-RN-04</b> | <b>P-RN-05</b> |

### 7.3. Tipos de UML frente a Requisitos

Cuadro 7: Requisitos de informacion

|                          |           |           |           |           |           |           |           |           |           |           |           |
|--------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| <b>Venta</b>             | x         | x         | x         | x         | x         | x         |           |           |           |           |           |
| <b>Traspaso - Pedido</b> |           |           |           | x         | x         | x         | x         | x         | x         | x         | x         |
|                          | <b>01</b> | <b>02</b> | <b>03</b> | <b>04</b> | <b>05</b> | <b>06</b> | <b>07</b> | <b>08</b> | <b>09</b> | <b>10</b> | <b>11</b> |

Cuadro 8: Reglas de negocio

|                          |           |           |           |           |           |
|--------------------------|-----------|-----------|-----------|-----------|-----------|
| <b>Venta</b>             | x         |           |           | x         | x         |
| <b>Traspaso - Pedido</b> |           | x         | x         |           | x         |
|                          | <b>01</b> | <b>02</b> | <b>03</b> | <b>04</b> | <b>05</b> |

Cuadro 9: Requisitos Funcionales

|                         |           |           |           |           |           |           |           |           |           |           |           |           |           |           |           |           |           |           |
|-------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| <b>Venta</b>            | X         | X         | X         | X         | X         | X         | X         |           |           |           |           |           |           |           | X         | X         | X         |           |
| <b>Traspaso - Venta</b> |           |           |           |           |           | X         | X         | X         | X         | X         | X         | X         | X         | X         | X         | X         |           | X         |
|                         | <b>01</b> | <b>02</b> | <b>03</b> | <b>04</b> | <b>05</b> | <b>06</b> | <b>07</b> | <b>08</b> | <b>09</b> | <b>10</b> | <b>11</b> | <b>12</b> | <b>13</b> | <b>14</b> | <b>15</b> | <b>16</b> | <b>17</b> | <b>18</b> |

## 8. Modelos relacionales

Para poder pasar de UML(MC) a 3FN(MR), tenemos que ser capaces de normalizar tanto en 2FN como en 1FN, esto lo hemos comprobado de la siguiente manera: En cada tupla de la 3FN se le asigna a cada atributo un solo valor del dominio sobre el que está definido, es decir, no existen atributos con varios valores. Esto prueba que está en 1FN. Tras asegurar la 1FN, pasamos a la comprobación de la 2FN, para validar esta normalización tenemos que tener en cuenta que los atributos que no sean Primary Key o Foreign Key (atributos no primos) deben de ser dependientes de estas claves, esto lo cumplimos gracias a las foreign keys y primary keys creadas para las asociaciones, esta relación está explicada más adelante. Finalmente llegamos a la comprobación de la 3FN, para esta comprobación tenemos que tener en cuenta que la relación tiene que estar en 2FN y que la relación de tablas simplemente se haga mediante las primary y las foreign key, todo lo que no sea claves candidatas no puede estar en varias entidades asociadas a la vez.



## 8.1. 3FN Venta

- Socio Los atributos son directos en la 3FN, añadiendo que DNI será una primray key. Tiene una relación 0..1 - n con Venta, venta mantiene todos sus atributos de manera directa en la 3FN, añadiendo un ID\_VENTA como primary key y debido a la relación con socio una foreign key llamada DNI.
- Venta Tiene una relación n - m con Producto, se crea una tabla intermedia debido a esta relación que es la tabla VentaProducto, con los siguientes atributos: cantidad, precioVenta e ivaVenta. Además tiene ID\_VENTA y ID\_PRODUCTO como primary keys y foreigners keys, estas claves son provocadas por la relación n - m entre Venta y Producto.
- Producto La tabla producto mantiene todos sus atributos de manera directa en la 3FN y añade ID\_PRODUCTO que será su primary key.
- Venta Tiene una relación 1 - 1 con Factura, factura mantiene todos sus atributos de manera directa en la 3FN y añade un ID\_FACTURA como primary key, un ID\_VENTA como foreign key debido a la relación con Venta y un ID\_EMPLAZAMIENTO debido a la relación con Emplazamiento.
- Factura tiene una relación n - 1 con Emplazamiento, emplazamiento mantiene todos sus atributos de manera directa en la 3FN y añade un ID\_EMPLAZAMIENTO como primary key.
- Producto Tiene una relación n - m con Emplazamiento, esto da lugar a la creación de una tabla intermedia, para poder tratar la relación n - m, con los siguientes atributos: cantidad, ID\_PRODUCTO(primary key y foreign key) y ID\_EMPLAZAMIENTO(primary key y foreign key). Estas claves son provocadas por la relacion n - m entre Producto y Emplazamiento.

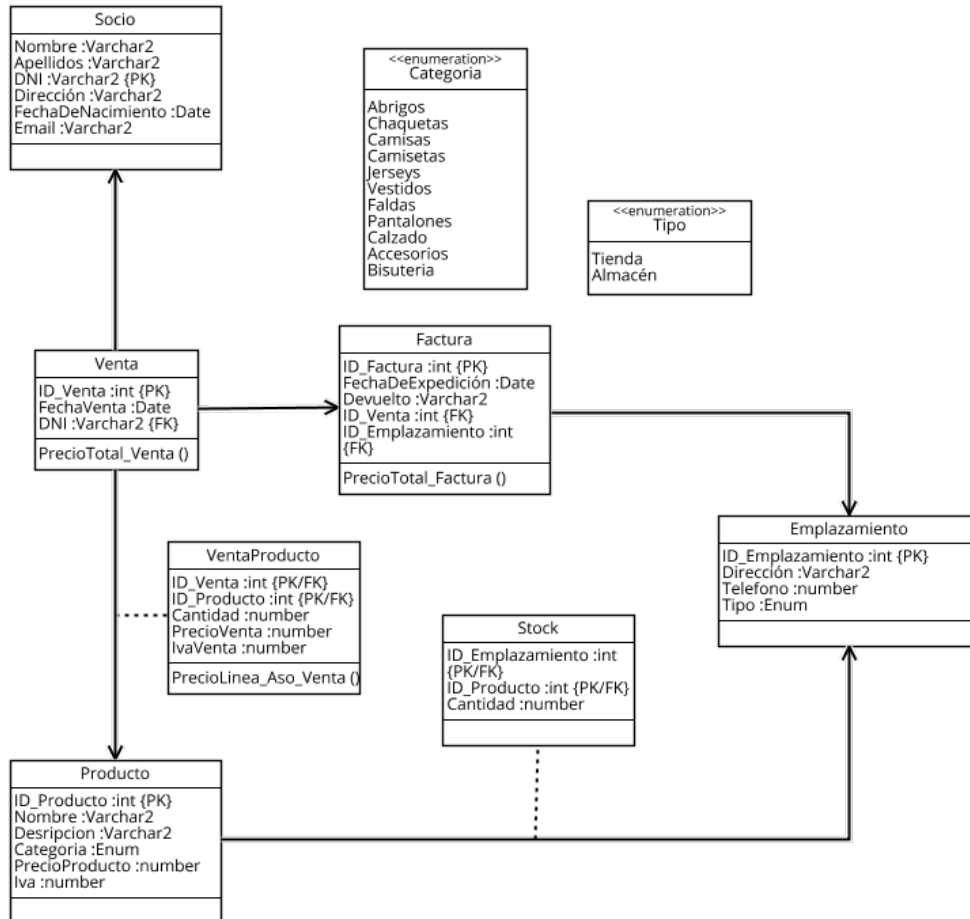


Figura 14: 3FN Venta

## 8.2. 3FN Traspaso - Pedido

Proveedor Tiene una relación 1 - n con Pedido, proveedor mantiene todos sus atributos de manera directa en la 3FN y añade un ID.PROVEEDOR como primary key.

Emplazamiento Tiene una relación 1 - n con Pedido, pedido mantiene todos sus atributos de manera directa en la 3FN, añade un ID\_PEDIDO como primary

key y un ID\_PROVEEDOR como foreign key debido a la relación con Proveedor.

Pedido Tiene una relación n - m con Producto, se crea una tabla intermedia debido a esta relación que es la tabla Asociación Pedido-Producto, con los siguientes atributos: cantidad, precioCompra, ivaCompra. Además tiene ID\_PEDIDO y ID\_PRODUCTO como primary keys y foreign keys, estas claves son provocadas por la relación n - m entre Pedido y Producto.

Albarán Tiene una relación 1 - 0..1 con Pedido, albarán mantiene todos sus atributos de manera directa en la 3FN y añade un ID\_ALBARAN como primary key.

Emplazamiento Tiene una relación 1 - m bidireccional con Traspaso, traspaso mantiene todos sus atributos de manera directa en la 3FN y añade un ID\_TRASPASO.

Emplazamiento Tiene una relación 1 - m bidireccional con Solicitud-Traspaso, solicitud-traspaso mantiene todos sus atributos de manera directa en la 3FN y añade un ID\_Solicitud.

Traspaso Tiene una relación n - m con Producto, esto da lugar a la creación de una tabla intermedia, para poder tratar la relación n-m, con los siguientes atributos: cantidad, ID\_Producto(primary key y foreign key) y ID\_Traspaso(primary key y foreign key). Estas claves son provocadas por la relación n - m entre Traspaso y Producto.

Solicitud-Traspaso Tiene una relación n - m con Producto, esto da lugar a la creación de una tabla intermedia, para poder tratar la relación n-m, con los siguientes atributos: cantidad, ID\_Producto(primary key y foreign key) y ID\_SOLICITUD(primary key y foreign key). Estas claves son provocadas por la relación n - m entre Solicitud-Traspaso y Producto.



```

9| DROP TABLE PROVEEDOR;
10| DROP TABLE STOCK;
11| DROP TABLE PRODUCTO;
12| DROP TABLE TRASPASO;
13| DROP TABLE SOLICITUD_TRASPASO;
14| DROP TABLE SOCIO;
15| DROP TABLE EMPLAZAMIENTO;

17|
18| CREATE TABLE EMPLAZAMIENTO(
19|     ID_Emplazamiento int PRIMARY KEY,
20|     Direccion VARCHAR2(100) NOT NULL,
21|     Telefono NUMBER(9),
22|     Tipo VARCHAR2(10) CHECK( Tipo IN('TIENDA','ALMACEN'))
23| );

25| CREATE TABLE SOCIO (
26|     Nombre VARCHAR2(25) NOT NULL,
27|     Apellidos VARCHAR2(50) NOT NULL,
28|     Direccion VARCHAR2(200) NOT NULL,
29|     FechaDeNacimiento DATE NOT NULL,
30|     Email VARCHAR2(50) NOT NULL,
31|     DNI VARCHAR2(9) PRIMARY KEY
32| );

33|
34| CREATE TABLE VENTA (
35|     ID_VENTA int PRIMARY KEY,
36|     FechaVenta DATE NOT NULL,
37|     DNI VARCHAR2(9),
38|     FOREIGN KEY(DNI) REFERENCES SOCIO
39| );

41| CREATE TABLE FACTURA (
42|     ID_FACTURA int PRIMARY KEY,
43|     FechaDeExpedicion DATE DEFAULT SYSDATE,
44|     Devuelto VARCHAR2(1) CHECK( Devuelto IN('F','T')),
45|     ID_Venta int,
46|     ID_Emplazamiento int,
47|     FOREIGN KEY (ID_Venta) REFERENCES VENTA,
48|     FOREIGN KEY (ID_Emplazamiento) REFERENCES Emplazamiento
49| );

51| CREATE TABLE PROVEEDOR (
52|     CIF VARCHAR2(9) PRIMARY KEY,
53|     Nombre VARCHAR2(75) NOT NULL,
54|     Telefono NUMBER(9) NOT NULL,
55|     Email VARCHAR2(50) NOT NULL
56| );

57| CREATE TABLE PEDIDO(
58|     ID_Pedido int PRIMARY KEY,
59|     FechaPedido DATE NOT NULL,
60|     ID_Emplazamiento int,
61|     CIF VARCHAR2(9),
62|     FOREIGN KEY (CIF) REFERENCES PROVEEDOR,
63|     FOREIGN KEY (ID_Emplazamiento) REFERENCES EMPLAZAMIENTO
64| );

65|
66| CREATE TABLE ALBARAN (
67|     ID_Albaran int NOT NULL,
68|     FechaFirma DATE NOT NULL,
69|     ID_Pedido INT PRIMARY KEY,
70|     FOREIGN KEY (ID_Pedido) REFERENCES PEDIDO);

73|
74| CREATE TABLE PRODUCTO(
75|     ID_Producto int PRIMARY KEY,
76|     Nombre VARCHAR2(50) NOT NULL,
77|     Descripcion VARCHAR2(300) NOT NULL,
78|     Categoria VARCHAR2(20) CHECK ( Categoria IN ('Abrigos','Chaquetas','Camisas',
79|         'Camisetas','Jerseys','Vestidos','Faldas',
80|         'Pantalones','Calzado','Accesorios','Bisuteria')),
81|     PrecioProducto NUMBER NOT NULL check(PrecioProducto>=0),
82|     IVA NUMBER NOT NULL check(IVA>=0 AND IVA<=1)
83| );

85| CREATE TABLE STOCK(

```

```

87     ID_Emplazamiento int,
88     ID_Producto int,
89     PRIMARY KEY (ID_Emplazamiento, ID_Producto),
90     Cantidad NUMBER(6) NOT NULL check(Cantidad>=0),
91     FOREIGN KEY (ID_Emplazamiento) REFERENCES EMPLAZAMIENTO,
92     FOREIGN KEY (ID_Producto) REFERENCES PRODUCTO
93 );
94
95 CREATE TABLE ASOCIACION_PEDIDO_PRODUCTO(
96     ID_PEDIDO int,
97     ID_PRODUCTO int,
98     PRIMARY KEY (ID_PEDIDO, ID_PRODUCTO),
99     Cantidad NUMBER(10) NOT NULL check(Cantidad>=20),
100    PrecioCompra NUMBER NOT NULL check(PrecioCompra>=0),
101    IVA NUMBER NOT NULL check(IVA>=0 AND IVA<=1),
102    FOREIGN KEY (ID_PEDIDO) REFERENCES PEDIDO,
103    FOREIGN KEY (ID_PRODUCTO) REFERENCES PRODUCTO
104 );
105
106 CREATE TABLE TRASPASO(
107     ID_Traspaso int PRIMARY KEY,
108     FechaTraspaso DATE NOT NULL,
109     ID_EmplazamientoSalida int,
110     ID_EmplazamientoEntrada int,
111     FOREIGN KEY (ID_EmplazamientoSalida) REFERENCES EMPLAZAMIENTO,
112     FOREIGN KEY (ID_EmplazamientoEntrada) REFERENCES EMPLAZAMIENTO
113 );
114
115 CREATE TABLE ASOCIACION_PRODUCTO_TRASPASO(
116     ID_Traspaso int,
117     ID_Producto int,
118     PRIMARY KEY (ID_Producto, ID_Traspaso),
119     Cantidad NUMBER(10) NOT NULL check(Cantidad>=0),
120     FOREIGN KEY (ID_Traspaso) REFERENCES TRASPASO,
121     FOREIGN KEY (ID_Producto) REFERENCES producto
122 );
123
124
125 CREATE TABLE SOLICITUD_TRASPASO(
126     ID_Solicitud int PRIMARY KEY,
127     FechaSolicitud DATE NOT NULL,
128     ID_EmplazamientoSalida int,
129     ID_EmplazamientoEntrada int,
130     FOREIGN KEY (ID_EmplazamientoSalida) REFERENCES EMPLAZAMIENTO,
131     FOREIGN KEY (ID_EmplazamientoEntrada) REFERENCES EMPLAZAMIENTO
132 );
133
134 CREATE TABLE ASOCIACION_PRODUCTO_SOLICITUD(
135     ID_Solicitud int,
136     ID_Producto int,
137     PRIMARY KEY (ID_Producto, ID_Solicitud),
138     Cantidad NUMBER(10) NOT NULL check(Cantidad>=0),
139     FOREIGN KEY (ID_Solicitud) REFERENCES SOLICITUD_TRASPASO,
140     FOREIGN KEY (ID_Producto) REFERENCES producto
141 );
142
143 CREATE TABLE ASOCIACION_VENTA_PRODUCTO(
144     ID_Venta int,
145     ID_Producto int,
146     PRIMARY KEY (ID_Venta, ID_Producto),
147     FOREIGN KEY (ID_Venta) REFERENCES VENTA,
148     FOREIGN KEY (ID_Producto) REFERENCES PRODUCTO,
149     Cantidad NUMBER(6) check(Cantidad>=0),
150     PrecioVenta NUMBER check(PrecioVenta>=0),
151     IvaVenta NUMBER check(IvaVenta>=0 AND IvaVenta<=1)
152 );
153
154 /* SECUENCIAS */
155
156 DROP SEQUENCE S_ID_Producto;
157 DROP SEQUENCE S_ID_Traspaso;
158 DROP SEQUENCE S_ID_Solicitud;
159 DROP SEQUENCE S_ID_Pedido;
160 DROP SEQUENCE S_ID_Albaran;
161 DROP SEQUENCE S_ID_Factura;
162 DROP SEQUENCE S_ID_Emplazamiento;

```

```

163 DROP SEQUENCE S_ID_Venta;
165
167
169 CREATE SEQUENCE S_ID_Emplazamiento START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
171 CREATE SEQUENCE S_ID_Venta START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
173 CREATE SEQUENCE S_ID_Pedido START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
175 CREATE SEQUENCE S_ID_Albaran START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
177 CREATE SEQUENCE S_ID_Producto START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
179 CREATE SEQUENCE S_ID-Traspaso START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
181 CREATE SEQUENCE S_ID_Solicitud START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
183 CREATE SEQUENCE S_ID_Factura START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
185
187 CREATE OR REPLACE TRIGGER crea_ID_Pedido
189 BEFORE INSERT ON Pedido
191 FOR EACH ROW
193 BEGIN
195 SELECT S_ID_Pedido.NEXTVAL INTO:NEW.ID_Pedido FROM DUAL;
197 END crea_ID_Pedido;
199
201 /
203
205 CREATE OR REPLACE TRIGGER crea_ID_Albaran
207 BEFORE INSERT ON ALBARAN
209 FOR EACH ROW
211 BEGIN
213 SELECT S_ID_Albaran.NEXTVAL INTO:NEW.ID_Albaran FROM DUAL;
215 END crea_ID_Albaran;
217
219 /
221
223 CREATE OR REPLACE TRIGGER Crea_ID_Producto
225 BEFORE INSERT ON PRODUCTO
227 FOR EACH ROW
229 BEGIN
231 SELECT S_ID_Producto.NEXTVAL INTO:NEW.ID_Producto FROM DUAL;
233 END Crea_Nuevo_Producto;
235
237 /
239
241 CREATE OR REPLACE TRIGGER Crea_ID-Traspaso
243 BEFORE INSERT ON TRASPASO
245 FOR EACH ROW
247 BEGIN
249 SELECT S_ID-Traspaso.NEXTVAL INTO:NEW.ID-Traspaso FROM DUAL;
251 END Crea_Nuevo-Traspaso;
253
255 /
257
259 CREATE OR REPLACE TRIGGER Crea_ID_Solicitud
261 BEFORE INSERT ON SOLICITUD_TRASPASO
263 FOR EACH ROW
265 BEGIN
267 SELECT S_ID_Solicitud.NEXTVAL INTO:NEW.ID_Solicitud FROM DUAL;
269 END Crea_Nuevo_Solicitud;
271
273 /
275
277 CREATE OR REPLACE TRIGGER crea_ID_Factura
279 BEFORE INSERT ON FACTURA
281 FOR EACH ROW
283 BEGIN
285 SELECT S_ID_Factura.NEXTVAL INTO:NEW.ID_Factura FROM DUAL;
287 END crea_ID_Factura;
289
291 /
293
295 CREATE OR REPLACE TRIGGER crea_ID_Emplazamiento
297 BEFORE INSERT ON EMPLAZAMIENTO
299 FOR EACH ROW
301 BEGIN
303 SELECT S_ID_Emplazamiento.NEXTVAL
305 INTO :NEW.ID_Emplazamiento FROM DUAL;
307 END crea_ID_Emplazamiento;
309
311 /

```

```

241 CREATE OR REPLACE TRIGGER crea_ID_Venta
    BEFORE INSERT ON Venta
    FOR EACH ROW
243 BEGIN
    SELECT S_ID_Venta.NEXTVAL
245 INTO :NEW.ID_Venta FROM DUAL;
END crea_ID_Venta;
247 /

```

sql/tablas.sql

i

## 9.2. Funciones y Procedures

```

1 CREATE OR REPLACE PROCEDURE Socio_Nuevo (
    p_Nombre IN SOCIO.Nombre%TYPE,
3    p_Apellidos IN SOCIO.Apellidos%TYPE,
    p_Direccion IN SOCIO.Direccion%TYPE,
5    p_FechaDeNacimiento IN SOCIO.FechaDeNacimiento%TYPE,
    p_Email IN SOCIO.Email%TYPE,
7    p_DNI IN SOCIO.DNI%TYPE)
IS BEGIN
9    INSERT INTO SOCIO
    VALUES (p_Nombre, p_Apellidos, p_Direccion, p_FechaDeNacimiento,
11         p_Email,p_DNI);
END Socio_Nuevo;
13
15 /
17 CREATE OR REPLACE PROCEDURE Producto_Nuevo(
    P_ID_Producto IN PRODUCTO.ID_Producto%TYPE,
    P_Nombre IN PRODUCTO.Nombre%TYPE,
19    P_Descripcion IN PRODUCTO.Descripcion%TYPE,
    P_Categoria IN PRODUCTO.Categoria%TYPE,
21    P_PrecioProducto IN PRODUCTO.PrecioProducto%TYPE,
    P_IVA IN PRODUCTO.IVA%TYPE)
23 IS BEGIN
    INSERT INTO PRODUCTO
25    VALUES(P_ID_Producto,P_Nombre,P_Descripcion,P_Categoria,
        P_PrecioProducto, P_IVA);
27 END Producto_Nuevo;
29 /
31 CREATE OR REPLACE PROCEDURE Stock_Nuevo (
    P_ID_Emplazamiento IN STOCK.ID_Emplazamiento%TYPE,
33    P_ID_Producto IN STOCK.ID_Producto%TYPE,
    P_Cantidad IN STOCK.Cantidad%TYPE)
35 IS BEGIN
    INSERT INTO STOCK
37    VALUES (P_ID_Emplazamiento, P_ID_Producto, P_Cantidad);
END Stock_Nuevo;
39
41 /
43 CREATE OR REPLACE PROCEDURE PRODUCTO_TRASPASO_Nuevo (
    p_ID_Traspaso IN ASOCIACION_PRODUCTO_TRASPASO.ID_Traspaso%TYPE,
    p_ID_Producto IN ASOCIACION_PRODUCTO_TRASPASO.ID_Producto%TYPE,
45    p_Cantidad IN ASOCIACION_PRODUCTO_TRASPASO.Cantidad%TYPE)
    IS BEGIN
47    INSERT INTO ASOCIACION_PRODUCTO_TRASPASO
    VALUES(p_ID_Traspaso,p_ID_Producto,p_Cantidad);
49 END PRODUCTO_TRASPASO_Nuevo;
51 /
53 CREATE OR REPLACE PROCEDURE PRODUCTO_SOLICITUD_Nuevo (
    p_ID_Solicitud IN ASOCIACION_PRODUCTO_SOLICITUD.ID_Solicitud%TYPE,
55    p_ID_Producto IN ASOCIACION_PRODUCTO_SOLICITUD.ID_Producto%TYPE,
    p_Cantidad IN ASOCIACION_PRODUCTO_SOLICITUD.Cantidad%TYPE)

```



```

57      IS BEGIN
58          INSERT INTO ASOCIACION_PRODUCTO_SOLICITUD
59          VALUES (p_ID_Solicitud, p_ID_Producto, p_Cantidad);
60      END PRODUCTO_SOLICITUD_Nuevo;
61
62  /
63
64  CREATE OR REPLACE PROCEDURE TRASPASO_Nuevo (
65      p_ID_Traspaso IN TRASPASO.ID_Traspaso %TYPE,
66      p_FechaTraspaso IN TRASPASO.FechaTraspaso %TYPE,
67      p_ID_EmplazamientoSalida IN TRASPASO.ID_EmplazamientoSalida %TYPE,
68      p_ID_EmplazamientoEntrada IN TRASPASO.ID_EmplazamientoEntrada %TYPE)
69  IS BEGIN
70      INSERT INTO TRASPASO
71      VALUES (p_ID_Traspaso, p_FechaTraspaso, p_ID_EmplazamientoSalida,
72              p_ID_EmplazamientoEntrada);
73  END TRASPASO_Nuevo;
74
75  /
76
77  CREATE OR REPLACE PROCEDURE SOLICITUD_Nuevo (
78      p_ID_Solicitud IN SOLICITUD_TRASPASO.ID_Solicitud %TYPE,
79      p_FechaSolicitud IN SOLICITUD_TRASPASO.FechaSolicitud %TYPE,
80      p_ID_EmplazamientoSalida IN SOLICITUD_TRASPASO.ID_EmplazamientoSalida %TYPE,
81      p_ID_EmplazamientoEntrada IN SOLICITUD_TRASPASO.ID_EmplazamientoEntrada %TYPE)
82  IS BEGIN
83      INSERT INTO SOLICITUD_TRASPASO
84      VALUES (p_ID_Solicitud, p_FechaSolicitud, p_ID_EmplazamientoSalida,
85              p_ID_EmplazamientoEntrada);
86  END SOLICITUD_Nuevo;
87
88  /
89
90
91  CREATE OR REPLACE PROCEDURE Emplazamiento_Nuevo(
92      P_ID_Emplazamiento IN EMPLAZAMIENTO.ID_Emplazamiento %TYPE,
93      P_Direccion IN EMPLAZAMIENTO.Direccion %TYPE,
94      P_Telefono IN EMPLAZAMIENTO.Telefono %TYPE,
95      P_Tipo IN EMPLAZAMIENTO.Tipo %TYPE)
96  IS BEGIN
97      INSERT INTO EMPLAZAMIENTO
98      VALUES (P_ID_Emplazamiento, P_Direccion, P_Telefono, P_Tipo);
99  END Emplazamiento_Nuevo;
100
101  /
102
103  CREATE OR REPLACE PROCEDURE Factura_Nueva(
104      p_ID IN FACTURA.ID_FACTURA %TYPE,
105      p_FechaDeExpedicion IN FACTURA.FechaDeExpedicion %TYPE,
106      p_Devuelto IN FACTURA.Devuelto %TYPE,
107      p_ID_Venta IN FACTURA.ID_Venta %TYPE,
108      p_ID_Emplazamiento IN FACTURA.ID_Emplazamiento %TYPE
109  )
110  IS BEGIN
111      INSERT INTO FACTURA VALUES(
112          p_ID, p_FechaDeExpedicion, p_Devuelto, p_ID_Venta,
113          p_ID_Emplazamiento);
114  END Factura_Nueva;
115
116  /
117
118  CREATE OR REPLACE PROCEDURE PROVEEDOR_Nuevo (
119      p_CIF IN PROVEEDOR.CIF %TYPE,
120      p_Nombre IN PROVEEDOR.Nombre %TYPE,
121      p_Telefono IN PROVEEDOR.Telefono %TYPE,
122      p_Email IN PROVEEDOR.Email %TYPE
123  ) IS BEGIN
124      INSERT INTO PROVEEDOR
125      VALUES (p_CIF, p_Nombre, p_Telefono, p_Email);
126  END PROVEEDOR_Nuevo;
127
128  /
129
130  CREATE OR REPLACE PROCEDURE ALBARAN_Nuevo (
131      p_ID_Albaran IN ALBARAN.ID_Albaran %TYPE,
132      p_FechaFirma IN ALBARAN.FechaFirma %TYPE,

```

```

135     p_ID_Pedido IN ALBARAN.ID_PEDIDO %TYPE
    )IS BEGIN
137     INSERT INTO ALBARAN
        VALUES ( p_ID_Albaran , p_FechaFirma , p_ID_Pedido);
139     END ALBARAN_Nuevo;

141 /

143 CREATE OR REPLACE PROCEDURE PEDIDO_Nuevo (
    p_ID_Pedido IN PEDIDO.ID_Pedido %TYPE,
145     p_FechaPedido IN PEDIDO.FechaPedido %TYPE,
    p_ID_Emplazamiento IN PEDIDO.ID_Emplazamiento %TYPE,
    p_CIF IN PEDIDO.CIF %TYPE
147 )IS BEGIN
    INSERT INTO PEDIDO
149     VALUES ( p_ID_Pedido ,p_FechaPedido ,
        p_ID_Emplazamiento , p_CIF);
151 END PEDIDO_Nuevo;

153 /

155 CREATE OR REPLACE PROCEDURE PEDIDO_PRODUCTO_Nuevo (
    p_ID_Producto IN ASOCIACION_PEDIDO_PRODUCTO.ID_Producto %TYPE,
157     p_ID_Pedido IN ASOCIACION_PEDIDO_PRODUCTO.ID_Pedido %TYPE,
    p_Cantidad IN ASOCIACION_PEDIDO_PRODUCTO.Cantidad %TYPE,
159     p_PrecioCompra IN ASOCIACION_PEDIDO_PRODUCTO.PrecioCompra %TYPE,
    p_IVA IN ASOCIACION_PEDIDO_PRODUCTO.IVA %TYPE)IS
161 BEGIN

163     INSERT INTO ASOCIACION_PEDIDO_PRODUCTO
        VALUES (p_ID_Producto , p_ID_Pedido , p_Cantidad , p_PrecioCompra ,
165         p_IVA);
    END PEDIDO_PRODUCTO_Nuevo;

167 /

169 /

171 CREATE OR REPLACE PROCEDURE Venta_Nueva(
    P_ID_Venta IN VENTA.ID_Venta %TYPE,
173     P_FechaVenta IN VENTA.FechaVenta %TYPE,
    P_DNI IN VENTA.DNI %TYPE)
175 IS BEGIN
    INSERT INTO VENTA
177     VALUES(P_ID_Venta , P_FechaVenta , P_DNI);
    END Venta_Nueva;

179 /

181 /

183 CREATE OR REPLACE PROCEDURE VENTA_PRODUCTO_Nueva(
    P_ID_Venta IN ASOCIACION_VENTA_PRODUCTO.ID_Venta %TYPE,
    P_ID_Producto IN ASOCIACION_VENTA_PRODUCTO.ID_Producto %TYPE,
185     P_Cantidad IN ASOCIACION_VENTA_PRODUCTO.Cantidad %TYPE,
    P_PrecioVenta IN ASOCIACION_VENTA_PRODUCTO.PrecioVenta %TYPE,
187     P_IvaVenta IN ASOCIACION_VENTA_PRODUCTO.IvaVenta %TYPE
    )
189 IS
    BEGIN
191     INSERT INTO ASOCIACION_VENTA_PRODUCTO
        VALUES(P_ID_Venta , P_ID_Producto ,P_Cantidad ,P_PrecioVenta ,P_IvaVenta);
193 END VENTA_PRODUCTO_Nueva;

195 /

197 CREATE OR REPLACE PROCEDURE MODIFICA_PROVEEDOR_NOMBRE
    (p_CIF IN PROVEEDOR.CIF %TYPE,
199     p_Nombre IN PROVEEDOR.Nombre %TYPE) IS
    BEGIN
201     UPDATE PROVEEDOR SET Nombre = p_Nombre WHERE p_CIF = CIF;
    END MODIFICA_PROVEEDOR_NOMBRE;

203 /

205 /

207 CREATE OR REPLACE PROCEDURE MODIFICA_PROVEEDOR_Telefono
    (p_CIF IN PROVEEDOR.CIF %TYPE,
    p_Telefono IN PROVEEDOR.Telefono %TYPE) IS
209 BEGIN
    UPDATE PROVEEDOR SET Telefono = p_Telefono WHERE p_CIF = CIF;

```

```

211 END MODIFICA_PROVEEDOR_Telefono;

213 /

215 CREATE OR REPLACE PROCEDURE MODIFICA_PROVEEDOR_EMAIL
216 (p_CIF IN PROVEEDOR.CIF %TYPE,
217 p_Email IN PROVEEDOR.Email %TYPE) IS
218 BEGIN
219 UPDATE PROVEEDOR SET Email = p_Email WHERE p_CIF = CIF;
220 END MODIFICA_PROVEEDOR_EMAIL;

221 /

223 CREATE OR REPLACE PROCEDURE MODIFICA_PRODUCTO_DESCRIPCION (
224 p_ID_Producto IN PRODUCTO.ID_Producto %TYPE,
225 p_Nombre IN PRODUCTO.Nombre %TYPE,
226 p_Descripcion IN PRODUCTO.Descripcion %TYPE,
227 p_Categoria IN PRODUCTO.Categoria %TYPE,
228 p_PrecioProducto IN PRODUCTO.PrecioProducto %TYPE,
229 p_IVA IN PRODUCTO.IVA %TYPE)
230 IS BEGIN
231 UPDATE PRODUCTO SET Descripcion = p_Descripcion WHERE p_ID_Producto = ID_Producto;
232 END MODIFICA_PRODUCTO_DESCRIPCION;

233 /

235 /

237 CREATE OR REPLACE PROCEDURE MODIFICA_PRODUCTO_PRECIO (
238 p_ID_Producto IN PRODUCTO.ID_Producto %TYPE,
239 p_Nombre IN PRODUCTO.Nombre %TYPE,
240 p_Descripcion IN PRODUCTO.Descripcion %TYPE,
241 p_Categoria IN PRODUCTO.Categoria %TYPE,
242 p_PrecioProducto IN PRODUCTO.PrecioProducto %TYPE,
243 p_IVA IN PRODUCTO.IVA %TYPE)
244 IS BEGIN
245 UPDATE PRODUCTO SET PrecioProducto = p_PrecioProducto WHERE p_ID_Producto = ID_Producto;
246 END MODIFICA_PRODUCTO_PRECIO;

247 /

249 /

251 CREATE OR REPLACE PROCEDURE MODIFICA_PRODUCTO_IVA (
252 p_ID_Producto IN PRODUCTO.ID_Producto %TYPE,
253 p_Nombre IN PRODUCTO.Nombre %TYPE,
254 p_Descripcion IN PRODUCTO.Descripcion %TYPE,
255 p_Categoria IN PRODUCTO.Categoria %TYPE,
256 p_PrecioProducto IN PRODUCTO.PrecioProducto %TYPE,
257 p_IVA IN PRODUCTO.IVA %TYPE)
258 IS BEGIN
259 UPDATE PRODUCTO SET IVA = p_IVA
260 WHERE p_ID_Producto = ID_Producto;
261 END MODIFICA_PRODUCTO_IVA;

262 /

263 /

265 CREATE OR REPLACE PROCEDURE MODIFICA_STOCK_CANTIDAD
266 (p_ID_Emplazamiento IN STOCK.ID_Emplazamiento %TYPE,
267 p_ID_Producto IN STOCK.ID_Producto %TYPE,
268 p_Cantidad IN STOCK.Cantidad %TYPE)
269 IS BEGIN
270 UPDATE STOCK SET Cantidad = p_Cantidad
271 WHERE p_ID_Emplazamiento = ID_Emplazamiento AND p_ID_PRODUCTO = ID_PRODUCTO;
272 END MODIFICA_STOCK_CANTIDAD;

273 /

275 create or replace PROCEDURE MODIFICA_SOCIO_DIRECCION(
276 m_DNI IN SOCIO.DNI %TYPE,
277 m_DIRECCION IN SOCIO.DIRECCION %TYPE)
278 IS BEGIN
279 UPDATE SOCIO SET DIRECCION = m_DIRECCION where m_DNI = DNI;
280 COMMIT WORK;
281 END MODIFICA_SOCIO_DIRECCION;

282 /

283 /

285 create or replace PROCEDURE MODIFICA_SOCIO_EMAIL(m_DNI IN SOCIO.DNI %TYPE,m_email IN SOCIO.EMAIL %TYPE)
286 IS BEGIN
287 UPDATE SOCIO SET EMAIL = m_email where m_DNI = DNI;

```

```

289 COMMIT WORK;
END MODIFICA_SOCIO_EMAIL;

291 /

293 CREATE OR REPLACE PROCEDURE MODIFICA_EMPLAZAMIENTO_DIR(
295     m_ID IN EMPLAZAMIENTO.ID_Emplazamiento%TYPE, m_Direccion IN EMPLAZAMIENTO.Direccion%TYPE)
IS BEGIN
297     UPDATE EMPLAZAMIENTO SET Direccion = m_Direccion where m_ID = ID_Emplazamiento;
END MODIFICA_EMPLAZAMIENTO_DIR;

299 /

301 CREATE OR REPLACE PROCEDURE MODIFICA_EMPLAZAMIENTO_TEL(
303     m_ID IN EMPLAZAMIENTO.ID_Emplazamiento%TYPE, m_Telefono IN EMPLAZAMIENTO.Telefono%TYPE)
IS BEGIN
305     UPDATE EMPLAZAMIENTO SET Telefono = m_Telefono where m_ID = ID_Emplazamiento;
END MODIFICA_EMPLAZAMIENTO_TEL;

307 /

309 CREATE OR REPLACE PROCEDURE MODIFICA_FACTURA_DEVUELTO
311     (m_ID_FACTURA IN FACTURA.ID_FACTURA%TYPE, m_Devuelto IN FACTURA.Devuelto%TYPE)
IS BEGIN
313     UPDATE FACTURA SET Devuelto = m_Devuelto where m_ID_FACTURA = ID_FACTURA;
END MODIFICA_FACTURA_DEVUELTO;

315 /

317 CREATE OR REPLACE PROCEDURE ELIMINA_PROVEEDOR(p_CIF IN PROVEEDOR.CIF%TYPE)
319     IS BEGIN
321     DELETE FROM PROVEEDOR WHERE p_CIF = CIF;
END ELIMINA_PROVEEDOR;

323 /

325 CREATE OR REPLACE PROCEDURE ELIMINA_PRODUCTO(p_ID_Producto IN PRODUCTO.ID_Producto%TYPE)
327     IS BEGIN
329     DELETE FROM PRODUCTO WHERE ID_Producto = p_ID_Producto;
END ELIMINA_PRODUCTO;

331 /

333 CREATE OR REPLACE PROCEDURE ELIMINA_EMPLAZAMIENTO(p_ID_Emplazamiento IN EMPLAZAMIENTO.
335     ID_Emplazamiento%TYPE)
IS BEGIN
337     DELETE FROM EMPLAZAMIENTO WHERE ID_Emplazamiento = p_ID_Emplazamiento;
END ELIMINA_EMPLAZAMIENTO;

339 /

339 CREATE OR REPLACE PROCEDURE ELIMINA_SOCIO(p_DNI IN SOCIO.DNI%TYPE)
341     IS BEGIN
343     DELETE FROM SOCIO WHERE DNI = p_DNI;
END ELIMINA_SOCIO;

345 /

345 CREATE OR REPLACE PROCEDURE ELIMINA_A_VENTA(e_ID_Producto IN Producto.ID_PRODUCTO%TYPE, e_ID_VENTA IN
347     VENTA.ID_VENTA%TYPE)
IS BEGIN
349     DELETE FROM ASOCIACION_VENTA_PRODUCTO WHERE ID_PRODUCTO = e_ID_Producto AND ID_VENTA = e_ID_VENTA;
END ELIMINA_A_VENTA;

351 /

353 /*FUNCIONES*/
355 /* Esta abajo con menos parametros
357 CREATE OR REPLACE FUNCTION precio_A_Venta_producto
359     (f_ID_Venta IN ASOCIACION_VENTA_PRODUCTO.ID_Venta%TYPE,
361     f_Cantidad IN ASOCIACION_VENTA_PRODUCTO.Cantidad%TYPE,
363     f_PrecioVenta IN ASOCIACION_VENTA_PRODUCTO.PrecioVenta%TYPE)
RETURN NUMBER is f_PrecioLinea ASOCIACION_VENTA_PRODUCTO.PRECIOLINEA%TYPE;
BEGIN
f_PrecioLinea := f_PrecioVenta * f_Cantidad;

```

```

363 RETURN(f_PrecioLinea);
END precio_A_Venta_producto;

365
/
367 */
/*
369 CREATE OR REPLACE FUNCTION precio_Venta
(f_ID_Venta IN VENTA.ID_Venta%TYPE)
371 RETURN NUMBER is f_PrecioTotal VENTA.PRECIOTOTAL%TYPE;
BEGIN
373 select SUM(precioLinea) into f_PrecioTotal from ASOCIACION_VENTA_PRODUCTO
where ID_Venta = f_ID_Venta;
375 RETURN(f_PrecioTotal);
END precio_Venta;

377
/
379 */
/* Esta abajo con menos parametros
381 CREATE OR REPLACE FUNCTION precio_A_Pedido_producto
(f_ID_Pedido IN ASOCIACION_PEDIDO_PRODUCTO.ID_Pedido%TYPE,
383 f_Cantidad IN ASOCIACION_PEDIDO_PRODUCTO.Cantidad%TYPE,
f_PrecioCompra IN ASOCIACION_PEDIDO_PRODUCTO.PrecioCompra%TYPE)
385 RETURN NUMBER is f_PrecioLinea ASOCIACION_PEDIDO_PRODUCTO.PrecioLinea%TYPE;
BEGIN
387 f_PrecioLinea := f_PrecioCompra * f_Cantidad;
RETURN(f_PrecioLinea);
389 END precio_A_Pedido_producto;

391
/
393 */
/*
395 CREATE OR REPLACE FUNCTION precio_Pedido
(f_ID_Pedido IN PEDIDO.ID_Pedido%TYPE)
RETURN NUMBER is f_PrecioTotal PEDIDO.PRECIOTOTAL%TYPE;
397 BEGIN
select SUM(PrecioLinea) into f_PrecioTotal from ASOCIACION_PEDIDO_PRODUCTO
399 where ID_Pedido = f_ID_Pedido;
RETURN(f_PrecioTotal);
401 END precio_Pedido;

403
/
405 */
CREATE OR REPLACE FUNCTION precioLinea_Aso_Pedido
407 (f_ID_PRODUCTO IN ASOCIACION_PEDIDO_PRODUCTO.ID_PRODUCTO%TYPE,f_ID_PEDIDO IN
ASOCIACION_PEDIDO_PRODUCTO.ID_PEDIDO%TYPE)
RETURN NUMBER is f_PrecioLinea ASOCIACION_PEDIDO_PRODUCTO.PRECIOCOMPRA%TYPE;
409 f_Cantidad ASOCIACION_PEDIDO_PRODUCTO.CANTIDAD%TYPE;
f_PrecioCompra ASOCIACION_PEDIDO_PRODUCTO.PRECIOCOMPRA%TYPE;
411 BEGIN
select Cantidad into f_Cantidad from ASOCIACION_PEDIDO_PRODUCTO where ID_Pedido = f_ID_PEDIDO AND
ID_PRODUCTO = f_ID_PRODUCTO;
413 select PrecioCompra into f_PrecioCompra from ASOCIACION_PEDIDO_PRODUCTO where ID_Pedido = f_ID_PEDIDO
AND ID_PRODUCTO = f_ID_PRODUCTO;
f_PrecioLinea := f_Cantidad * f_PrecioCompra;
415 RETURN(f_PrecioLinea);
END precioLinea_Aso_Pedido;

417
/
419
CREATE OR REPLACE FUNCTION precioLinea_Aso_Venta
421 (f_ID_PRODUCTO IN ASOCIACION_VENTA_PRODUCTO.ID_PRODUCTO%TYPE,f_ID_VENTA IN ASOCIACION_VENTA_PRODUCTO.
ID_VENTA%TYPE)
RETURN NUMBER is f_PrecioLinea ASOCIACION_VENTA_PRODUCTO.PRECIOVENTA%TYPE;
423 f_Cantidad ASOCIACION_VENTA_PRODUCTO.CANTIDAD%TYPE;
f_PrecioVenta ASOCIACION_VENTA_PRODUCTO.PRECIOVENTA%TYPE;
425 BEGIN
select Cantidad into f_Cantidad from ASOCIACION_VENTA_PRODUCTO where ID_VENTA = f_ID_VENTA AND
ID_PRODUCTO = f_ID_PRODUCTO;
427 select PrecioVenta into f_PrecioVenta from ASOCIACION_VENTA_PRODUCTO where ID_VENTA = f_ID_VENTA AND
ID_PRODUCTO = f_ID_PRODUCTO;
f_PrecioLinea := f_Cantidad * f_PrecioVenta;
429 RETURN(f_PrecioLinea);
END precioLinea_Aso_Venta;

431
/
433

```

```

435 CREATE OR REPLACE FUNCTION precioTotal_Venta
(f_ID_VENTA IN ASOCIACION_VENTA_PRODUCTO.ID_VENTA %TYPE)
RETURN NUMBER is f_precioTotal ASOCIACION_VENTA_PRODUCTO.PRECIOVENTA %TYPE;
437 precio_AUX ASOCIACION_VENTA_PRODUCTO.PRECIOVENTA %TYPE;

439 CURSOR all_prods
IS
441 SELECT id_venta,id_producto,cantidad
FROM ASOCIACION_VENTA_PRODUCTO
443 ORDER BY ID_producto;

445 m_id_venta ASOCIACION_VENTA_PRODUCTO.id_venta %TYPE;
m_id_producto Producto.id_producto %type;
447 m_cantidad ASOCIACION_VENTA_PRODUCTO.cantidad %TYPE;
BEGIN
449 f_precioTotal := 0;
OPEN all_prods;
451 LOOP
Fetch all_prods INTO m_id_venta,m_id_producto,m_cantidad;
453 EXIT WHEN all_prods %NOTFOUND;
if m_id_venta = f_id_venta
455 THEN
select precioVenta into precio_AUX from asociacion_venta_producto where id_venta = m_id_venta and
id_producto=m_id_producto;

457 f_precioTotal := (precio_AUX*m_cantidad) + f_precioTotal;
459 END IF;
END LOOP;
461 CLOSE all_prods;
RETURN(f_PrecioTotal);
463 END precioTotal_Venta;

465 /

467 CREATE OR REPLACE FUNCTION precioTotal_Factura
(f_ID_VENTA IN ASOCIACION_VENTA_PRODUCTO.ID_VENTA %TYPE)
469 RETURN NUMBER is f_precioTotal number(8,2);
precio_AUX ASOCIACION_VENTA_PRODUCTO.PRECIOVENTA %TYPE;
471 socio varchar2(9);
BEGIN
473 select precioTotal_Venta(f_ID_VENTA) into precio_AUX from dual;
select dni into socio from venta where id_Venta = f_id_Venta;
475 f_precioTotal := precio_AUX;
if socio is not null
477 then
f_precioTotal := f_precioTotal * 0.95;
479 END IF;
RETURN(f_PrecioTotal);
481 END precioTotal_Factura;

483 /

485 CREATE OR REPLACE FUNCTION precioTotal_Albaran
(f_ID_PEDIDO IN ASOCIACION_PEDIDO_PRODUCTO.ID_PEDIDO %TYPE)
487 RETURN NUMBER is f_precioTotal number(8,2);
precio_AUX ASOCIACION_PEDIDO_PRODUCTO.PRECIOCOMPRA %TYPE;
489 BEGIN
select precioTotal_Pedido(f_ID_PEDIDO) into precio_AUX from dual;
491 f_precioTotal := precio_AUX;
RETURN(f_PrecioTotal);
493 END precioTotal_Albaran;

495 /

497 CREATE OR REPLACE FUNCTION precioTotal_PEDIDO
(f_ID_PEDIDO IN ASOCIACION_PEDIDO_PRODUCTO.ID_PEDIDO %TYPE)
499 RETURN NUMBER is f_precioTotal ASOCIACION_PEDIDO_PRODUCTO.PRECIOCOMPRA %TYPE;
precio_AUX ASOCIACION_PEDIDO_PRODUCTO.PRECIOCOMPRA %TYPE;
501 CURSOR all_prods
IS
503 SELECT id_pedido,id_producto,cantidad
FROM ASOCIACION_PEDIDO_PRODUCTO
505 ORDER BY ID_producto;

507 m_id_pedido ASOCIACION_PEDIDO_PRODUCTO.id_pedido %TYPE;
509 m_id_producto Producto.id_producto %type;

```

```

m_cantidad ASOCIACION_PEDIDO_PRODUCTO.cantidad%TYPE;
511 BEGIN
    f_precioTotal := 0;
513 OPEN all_prods;
    LOOP
515     Fetch all_prods INTO m_id_pedido,m_id_producto,m_cantidad;
    EXIT WHEN all_prods%NOTFOUND;
517     if m_id_pedido = f_id_pedido
    THEN
519     select precioCompra into precio_AUX from ASOCIACION_PEDIDO_PRODUCTO where id_pedido = m_id_pedido
        and id_producto=m_id_producto;

521     f_precioTotal := (precio_AUX*m_cantidad) + f_precioTotal;
    END IF;
523 END LOOP;
    CLOSE all_prods;
525 RETURN(f_PrecioTotal);
    END precioTotal_Pedido;
527
/

```

sql/funciones\_y\_procedures.sql

### 9.3. Triggers

```

/* TRIGGERS */
2
/*
4 CREATE OR REPLACE TRIGGER descuento_socio
    BEFORE INSERT ON factura
    FOR EACH ROW
6 declare v_preciototal VENTA.PRECIOTOTAL%TYPE;
    v_DNI VENTA.dni%TYPE;
8 BEGIN
    select preciototal into v_preciototal from venta where id_venta = :NEW.id_venta;
10 select dni into v_dni from venta where id_venta = :NEW.id_venta;
    IF v_DNI IS NOT NULL then
12 UPDATE venta set preciototal = v_preciototal * 0.95 where id_venta = :NEW.id_venta;
    :New.preciototal := v_preciototal * 0.95;
14 else
    update venta set preciototal = v_preciototal where id_venta = :NEW.id_venta;
16 END IF;
END;
18
/
20
/*
/*
22 CREATE OR REPLACE TRIGGER stock_minimo
    AFTER INSERT OR UPDATE ON stock
24 FOR EACH ROW
BEGIN
26 IF :NEW.cantidad <0
    THEN
28 raise_application_error(-20601, :NEW.cantidad || 'No se puede realizar esta operacion');
    END IF;
30 END;

32
/
34
/*
34 CREATE OR REPLACE TRIGGER Inicializa_nueva_Venta
    BEFORE INSERT ON VENTA
36 FOR EACH ROW
BEGIN
38 :NEW.FechaVenta := SYSDATE;
    END Comprueba_Venta;
40
/
42
42 CREATE OR REPLACE TRIGGER Inicializa_nuevo_Pedido
44 BEFORE INSERT ON PEDIDO
    FOR EACH ROW
46 BEGIN

```

```

:NEW.FechaPedido := SYSDATE;
48 END Comprueba_Pedido;

50 /

52 CREATE OR REPLACE TRIGGER Inicializa_nueva_Factura
BEFORE INSERT ON Factura
54 FOR EACH ROW
BEGIN
56 :NEW.FechaDeExpedicion := SYSDATE;
END Inicializa_nueva_Factura;
58

60 /

62 CREATE OR REPLACE TRIGGER modifica_stock_venta
BEFORE INSERT ON FACTURA
64 FOR EACH ROW
DECLARE
66 e_ID_Emplazamiento Emplazamiento.ID_Emplazamiento %TYPE;
v_ID_Venta Venta.ID_VENTA %TYPE;

68
69 CURSOR all_prods
70 IS
71 SELECT id_venta, id_producto, cantidad
72 FROM ASOCIACION_VENTA_PRODUCTO
73 ORDER BY ID_producto;
74
75 m_id_venta ASOCIACION_VENTA_PRODUCTO.id_venta %TYPE;
76 m_id_producto Producto.id_producto %type;
77 m_cantidad ASOCIACION_VENTA_PRODUCTO.cantidad %TYPE;
78
79 BEGIN
80 select ID_Emplazamiento into e_ID_Emplazamiento from Emplazamiento where ID_Emplazamiento = :NEW.
ID_Emplazamiento;
81 select ID_Venta into v_ID_Venta from Venta where ID_Venta = :New.ID_Venta;
82
83 OPEN all_prods;
84 LOOP
85     Fetch all_prods INTO m_id_venta, m_id_producto, m_cantidad;
86     EXIT WHEN all_prods %NOTFOUND;
87     if m_id_venta = v_id_Venta
88     THEN
89         update stock set cantidad = cantidad - m_cantidad where id_emplazamiento = e_ID_Emplazamiento AND
90         id_producto = m_ID_Producto;
91     END IF;
92     END LOOP;
93     CLOSE all_prods;
94 END modifica_stock_venta;

96 /
/*
97 CREATE OR REPLACE TRIGGER inicializa_preciolinea_alv
98 BEFORE INSERT OR UPDATE ON ASOCIACION_VENTA_PRODUCTO
99 FOR EACH ROW
100 BEGIN
101     :NEW.preciolinea := :New.cantidad * :New.precioVenta;
102 END inicializa_preciolinea_alv;

104 /
*/
/*
105 CREATE OR REPLACE TRIGGER inicializa_preciototal_venta
106 BEFORE INSERT OR UPDATE ON ASOCIACION_VENTA_PRODUCTO
107 for each row
108 begin
109     UPDATE venta set preciototal = PRECIOTOTAL + (:New.cantidad * :New.precioVenta) where id_venta = :
110     New.id_venta;
111 END inicializa_preciototal_venta;

112
114 /
*/
115 CREATE OR REPLACE TRIGGER modifica_stock_pedido
116 BEFORE INSERT OR UPDATE ON ALBARAN
117 FOR EACH ROW
118 DECLARE

```



```

122 e_ID_Emplazamiento Emplazamiento.ID_Emplazamiento %TYPE;
    p_ID_PEDIDO PEDIDO.ID_PEDIDO %TYPE;

124 CURSOR all_prods
    IS
126 SELECT id_pedido, id_producto, cantidad
    FROM ASOCIACION_PEDIDO_PRODUCTO
128 ORDER BY ID_producto;

130 m_id_pedido ASOCIACION_PEDIDO_PRODUCTO.id_pedido %TYPE;
    m_id_producto Producto.id_producto %type;
132 m_cantidad ASOCIACION_PEDIDO_PRODUCTO.cantidad %TYPE;

134 BEGIN

136 select ID_Emplazamiento into e_ID_Emplazamiento from pedido where ID_pedido = :NEW.ID_Pedido;
    select ID_PEDIDO into p_id_pedido from pedido where ID_pedido = :NEW.ID_Pedido;
138 OPEN all_prods;
    LOOP
140     Fetch all_prods INTO m_id_pedido, m_id_producto, m_cantidad;
        EXIT WHEN all_prods %NOTFOUND;
142     if m_id_pedido = p_id_pedido
        THEN
144         update stock set cantidad = cantidad+m_cantidad where id_emplazamiento = e_ID_Emplazamiento AND
            id_producto = m_ID_Producto;
        END IF;
146     END LOOP;
        CLOSE all_prods;
148 END modifica_stock_pedido;

150 /

152 CREATE OR REPLACE TRIGGER modifica_stock_traspaso
    BEFORE INSERT OR UPDATE ON ASOCIACION_PRODUCTO_TRASPASO
154 FOR EACH ROW
    DECLARE
156 e_ID_Emplazamiento_salida Emplazamiento.ID_Emplazamiento %TYPE;
    e_ID_Emplazamiento_entrada Emplazamiento.ID_Emplazamiento %TYPE;
158 t_ID_traspaso traspaso.ID_traspaso %TYPE;

160 BEGIN
    select ID_Emplazamientosalida into e_ID_Emplazamiento_salida from traspaso where ID_traspaso = :NEW.
        ID_traspaso;
162 select ID_Emplazamientoentrada into e_ID_Emplazamiento_entrada from traspaso where ID_traspaso = :NEW
        .ID_traspaso;
    select ID_traspaso into t_id_traspaso from traspaso where ID_traspaso = :NEW.ID_traspaso;
164 update stock set cantidad = (cantidad- :NEW.cantidad) where id_emplazamiento =
        e_ID_Emplazamiento_salida AND id_producto = :NEW.id_producto;
    update stock set cantidad = (cantidad+ :NEW.cantidad) where id_emplazamiento =
        e_ID_Emplazamiento_entrada AND id_producto = :NEW.id_producto;
166 END modifica_stock_traspaso;

168 /

170 CREATE OR REPLACE TRIGGER Inicializa_Nuevo_Pedido
    BEFORE INSERT ON Pedido
    FOR EACH ROW
172 BEGIN
    :NEW.FechaPedido := SYSDATE;
174 END Inicializa_Nuevo_Pedido;

176 /

178 CREATE OR REPLACE TRIGGER Inicializa_Nueva_ST
180 BEFORE INSERT ON Solicitud_Traspaso
    FOR EACH ROW
182 BEGIN
    :NEW.FechaSolicitud := SYSDATE;
184 END Inicializa_Nueva_ST;

186 /

188 CREATE OR REPLACE TRIGGER Inicializa_Nuevo_Traspaso
190 BEFORE INSERT ON Traspaso
    FOR EACH ROW
    BEGIN
192 :NEW.FechaTraspaso := SYSDATE;

```

```

194 END Inicializa_Nuevo_Traspaso;
195 /
196
198 create or replace TRIGGER solicitud_stock_minimo
199 BEFORE INSERT ON asociacion_producto_solicitud
200 FOR EACH ROW
201 DECLARE
202 e_id_emplazamientoentrada SOLICITUD_TRASPASO.ID_EMPLAZAMIENTOENTRADA %TYPE;
203 e_cantidad Stock.cantidad %TYPE;
204 BEGIN
205     SELECT id_emplazamientoentrada into e_id_emplazamientoentrada from solicitud_traspaso where
206         id_solicitud = :NEW.id_solicitud;
207     select cantidad into e_cantidad from stock where id_emplazamiento = e_id_emplazamientoentrada and
208         id_producto = :NEW.id_producto;
209     if (e_cantidad - :NEW.cantidad) <=5
210 THEN raise_application_error(-20601, :NEW.cantidad || 'No se permite la solicitud, el otro
211     emplazamiento alcanzara su stock minimo');
212 END IF;
213 END;
214 /
215 /*
216 create or replace TRIGGER inicializa_preciolinea_alp
217 BEFORE INSERT OR UPDATE ON ASOCIACION_PEDIDO_PRODUCTO
218 FOR EACH ROW
219 BEGIN
220 :NEW.preciolinea := :New.cantidad * :New.preciocompra;
221 END inicializa_preciolinea_alp;
222 /
223 */
224 /*
225 create or replace TRIGGER inicializa_preciototal_albaran
226 BEFORE INSERT OR UPDATE ON Albaran
227 for each row
228 DECLARE
229 p_preciototal pedido.preciototal %TYPE;
230 begin
231 select preciototal into p_preciototal from pedido where id_pedido = :New.id_pedido;
232 :New.preciototal := p_preciototal;
233 END inicializa_preciototal_albaran;
234 /
235 */
236 /*
237 create or replace TRIGGER inicializa_preciototal_pedido
238 BEFORE INSERT OR UPDATE ON ASOCIACION_pedido_producto
239 for each row
240 begin
241 UPDATE pedido set preciototal = PRECIOTOTAL + (:New.cantidad * :New.preciocompra) where id_pedido =
242 :New.id_pedido;
243 END inicializa_preciototal_pedido;
244 /
245 */
246 create or replace trigger inicializa_precioventa_apv
247 before insert on asociacion_venta_producto
248 for each row
249 declare
250 p_precio producto.precioproducto %TYPE;
251 begin
252 select precioproducto into p_precio from producto where id_producto = :New.id_producto;
253 :NEW.precioventa := p_precio;
254 END inicializa_precioventa_apv;
255 /
256
258 create or replace trigger inicializa_IVA_apv
259 before insert on asociacion_venta_producto
260 for each row
261 declare
262 p_IVA PRODUCTO.IVA %TYPE;
263 begin
264 select Iva into p_IVA from producto where id_producto = :New.id_producto;
265 :NEW.IVAVENTA := p_IVA;

```

```

266| END inicializa_IVA_apv;
268| /
270| create or replace trigger inicializa_IVA_APP
271|   before insert on asociacion_pedido_producto
272|   for each row
273|   declare
274|     p_IVA PRODUCTO.IVA %TYPE;
275|   begin
276|     select IVA into p_IVA from producto where id_producto = :New.id_producto;
277|     :NEW.IVA := p_IVA;
278|   END inicializa_IVA_APP;
280| /
282| create or replace trigger devolucion
283|   before update on factura
284|   for each row
285|   begin
286|     if(sysdate - :old.fechadeexpedicion)>30 then
287|       raise_application_error(-20601, :NEW.fechadeexpedicion || 'No se permite la devolucion, han pasado
288|         mas de 30 dias');
289|     END IF;
290|   END devolucion;
291| /
292| /*
293| create or replace trigger mensaje
294|   before update of cantidad on stock
295|   for each row
296|   DECLARE
297|     lines dbms_output.chararr;
298|     num_lines number;
299|   BEGIN
300|
301|     if (:old.cantidad - :new.cantidad)<5 then
302|       -- enable the buffer with default size 20000
303|       dbms_output.enable;
304|
305|       dbms_output.put_line('Hello Reader!');
306|       dbms_output.put_line('Hope you have enjoyed the tutorials!');
307|       dbms_output.put_line('Have a great time exploring pl/sql!');
308|
309|       num_lines := 3;
310|
311|       dbms_output.get_lines(lines, num_lines);
312|
313|       FOR i IN 1..num_lines LOOP
314|         dbms_output.put_line(lines(i));
315|       END LOOP;
316|     end IF;
317|   END mensaje;
318| /
319| */

```

sql/triggers.sql

## 9.4. Pruebas

```

1| /* FUNCION AUXILIAR */
2| CREATE OR REPLACE FUNCTION EQUALS(salida BOOLEAN, salidaEsperada BOOLEAN)
3| RETURN VARCHAR2 AS
4| BEGIN
5|   IF (salida = salidaEsperada) THEN
6|     RETURN 'EXITO';
7|   ELSE
8|     RETURN 'FALLO';
9|   END IF;
10| END EQUALS;
11| /

```

```

14 CREATE OR REPLACE PROCEDURE PRINTR(nombre_prueba VARCHAR2, salida BOOLEAN,
                                     salidaEsperada BOOLEAN)
16 IS BEGIN
    DBMS_OUTPUT.put_line(nombre_prueba || ':' || EQUALS(salida, salidaEsperada));
18 END PRINTR;

20 /

22 /* DEFINICION PRUEBAS */
23 CREATE OR REPLACE PACKAGE PRUEBAS_SOCIO AS
24     PROCEDURE inicializar;
25     PROCEDURE insertar
26         (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_apellidos VARCHAR2,
27          i_DNI VARCHAR2, i_direccion VARCHAR2, i_nacimiento DATE, i_email VARCHAR2
28          ,salidaEsperada BOOLEAN);
29     PROCEDURE actualizar
30         (nombre_prueba VARCHAR2, a_DNI VARCHAR2, a_direccion VARCHAR2,
31          a_email VARCHAR2, salidaEsperada BOOLEAN);
32     PROCEDURE eliminar
33         (nombre_prueba VARCHAR2, e_DNI VARCHAR2, salidaEsperada BOOLEAN);
34 END PRUEBAS_SOCIO;

36 /

38 CREATE OR REPLACE PACKAGE PRUEBAS_PROVEEDOR AS
39     PROCEDURE inicializar;
40     PROCEDURE insertar
41         (nombre_prueba VARCHAR2, i_CIF VARCHAR2, i_nombre VARCHAR2,
42          i_telefono INT, i_email VARCHAR2, salidaEsperada BOOLEAN);
43     PROCEDURE actualizar
44         (nombre_prueba VARCHAR2, a_cif VARCHAR2, a_nombre VARCHAR2,
45          a_telefono INT, a_email VARCHAR2, salidaEsperada BOOLEAN);
46     PROCEDURE eliminar
47         (nombre_prueba VARCHAR2, e_CIF VARCHAR2, salidaEsperada BOOLEAN);
48 END PRUEBAS_PROVEEDOR;

50 /

52 CREATE OR REPLACE PACKAGE PRUEBAS_EMPLAZAMIENTO AS
53     PROCEDURE inicializar;
54     PROCEDURE insertar
55         (nombre_prueba VARCHAR2, i_direccion VARCHAR2, i_telefono INT,
56          i_tipo VARCHAR2, salidaEsperada BOOLEAN);
57     PROCEDURE actualizar
58         (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_direccion VARCHAR2,
59          a_telefono INT, salidaEsperada BOOLEAN);
60     PROCEDURE eliminar
61         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, salidaEsperada BOOLEAN);
62 END PRUEBAS_EMPLAZAMIENTO;

64 /

66 CREATE OR REPLACE PACKAGE PRUEBAS_ALBARAN AS
67     PROCEDURE inicializar;
68     PROCEDURE insertar
69         (nombre_prueba VARCHAR2, i_fecha DATE, i_id_pedido INT, salidaEsperada BOOLEAN);
70     PROCEDURE eliminar
71         (nombre_prueba VARCHAR2, e_id_albaran INT, salidaEsperada BOOLEAN);
72 END PRUEBAS_ALBARAN;

74 /

76 CREATE OR REPLACE PACKAGE PRUEBAS_PEDIDO AS
77     PROCEDURE inicializar;
78     PROCEDURE insertar
79         (nombre_prueba VARCHAR2, i_fecha DATE, i_id_emplazamiento INT, i_cif VARCHAR2, salidaEsperada
80          BOOLEAN);
81     PROCEDURE eliminar
82         (nombre_prueba VARCHAR2, e_id_pedido INT, salidaEsperada BOOLEAN);
83 END PRUEBAS_PEDIDO;

84 /

86 CREATE OR REPLACE PACKAGE PRUEBAS_PRODUCTO AS
87     PROCEDURE inicializar;
88     PROCEDURE insertar

```

```

90      (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_descripcion VARCHAR2,
      i_categoria VARCHAR2, i_precioProducto INT, i_iva INT, salidaEsperada BOOLEAN);
92  PROCEDURE actualizar
      (nombre_prueba VARCHAR2, a_id_producto INT, a_descripcion VARCHAR2,
      a_precioProducto INT, a_iva INT, salidaEsperada BOOLEAN);
94  PROCEDURE eliminar
      (nombre_prueba VARCHAR2, e_id_producto INT, salidaEsperada BOOLEAN);
96  END PRUEBAS_PRODUCTO;

98  /

100 CREATE OR REPLACE PACKAGE PRUEBAS_VENTA AS
      PROCEDURE inicializar;
102  PROCEDURE insertar
      (nombre_prueba VARCHAR2, i_fecha DATE, i_dni VARCHAR2,
      salidaEsperada BOOLEAN);
104  PROCEDURE eliminar
      (nombre_prueba VARCHAR2, e_id_venta INT, salidaEsperada BOOLEAN);
106  /*procedure descuento
      (nombre_prueba VARCHAR2, v_id_venta int,v_preciototal number,v_dni_socio varchar2,
108  salidaEsperada BOOLEAN);*/
      END PRUEBAS_VENTA;

110  /

112 CREATE OR REPLACE PACKAGE PRUEBAS_SOLICITUD_TRASPASO AS
      PROCEDURE inicializar;
114  PROCEDURE insertar
      (nombre_prueba VARCHAR2, i_fechaSolicitud DATE, i_id_emplazamientoSalida INT,
      i_id_emplazamientoEntrada INT, salidaEsperada BOOLEAN);
116  PROCEDURE eliminar
      (nombre_prueba VARCHAR2, e_id_solicitudTraspaso INT, salidaEsperada BOOLEAN);
118  END PRUEBAS_SOLICITUD_TRASPASO;

120  /

122 /

124 CREATE OR REPLACE PACKAGE PRUEBAS_FACTURA AS
      PROCEDURE inicializar;
126  PROCEDURE insertar
      (nombre_prueba VARCHAR2, i_fechaDeExpedicion DATE, i_devuelto VARCHAR2,
      i_id_venta INT, i_id_emplazamiento INT, salidaEsperada BOOLEAN);
128  PROCEDURE actualizar
      (nombre_prueba VARCHAR2, a_id_factura INT, a_devuelto VARCHAR2,
      salidaEsperada BOOLEAN);
130  PROCEDURE eliminar
      (nombre_prueba VARCHAR2, e_id_factura INT, salidaEsperada BOOLEAN);
132  END PRUEBAS_FACTURA;

134  /

136 /

138 CREATE OR REPLACE PACKAGE PRUEBAS_TRASPASO AS
      PROCEDURE inicializar;
140  PROCEDURE insertar
      (nombre_prueba VARCHAR2, i_fechaTraspaso DATE, i_id_emplazamientoSalida INT,
      i_id_emplazamientoEntrada INT, salidaEsperada BOOLEAN);
142  PROCEDURE eliminar
      (nombre_prueba VARCHAR2, e_id_Traspaso INT, salidaEsperada BOOLEAN);
144  END PRUEBAS_TRASPASO;

146  /

148 /

150 CREATE OR REPLACE PACKAGE PRUEBAS_A_VENTA_PRODUCTO AS
      PROCEDURE inicializar;
152  PROCEDURE insertar
      (nombre_prueba VARCHAR2, i_id_venta INT, i_id_producto INT,
      i_cantidad NUMBER, i_precioVenta NUMBER, i_ivaVenta NUMBER,
      salidaEsperada BOOLEAN);
154  PROCEDURE eliminar
      (nombre_prueba VARCHAR2, e_id_venta INT, e_id_producto INT, salidaEsperada BOOLEAN);
156  END PRUEBAS_A_VENTA_PRODUCTO;

158  /

160 /

162 CREATE OR REPLACE PACKAGE PRUEBAS_STOCK AS
      PROCEDURE inicializar;
164  PROCEDURE insertar
      (nombre_prueba VARCHAR2, i_id_emplazamiento INT, i_id_producto INT,

```

```

166         i_cantidad INT, salidaEsperada BOOLEAN);
167     PROCEDURE actualizar
168         (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_id_producto INT,
169          a_cantidad INT, salidaEsperada BOOLEAN);
170     PROCEDURE eliminar
171         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT,
172          salidaEsperada BOOLEAN);
173     PROCEDURE stock_correcto
174         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
175          INT, salidaEsperada BOOLEAN);
176     PROCEDURE stock_correcto_p
177         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
178          INT, salidaEsperada BOOLEAN);
179     PROCEDURE stock_correcto_t
180         (nombre_prueba VARCHAR2, e_id_emplazamiento envia int, e_id_emplazamiento_recibe int,
181          stock_previo envia int, stock_previo_recibe int,
182          cantidad int, e_id_producto int, salidaEsperada BOOLEAN);
183     END PRUEBAS_STOCK;
184
185 /
186
187 CREATE OR REPLACE PACKAGE PRUEBAS_A_PRODUCTO_SOLICITUD AS
188     PROCEDURE inicializar;
189     PROCEDURE insertar
190         (nombre_prueba VARCHAR2, i_id_producto INT, i_id_Solicitud INT,
191          i_cantidad NUMBER, salidaEsperada BOOLEAN);
192     PROCEDURE eliminar
193         (nombre_prueba VARCHAR2, e_id_Solicitud INT, e_id_producto INT, salidaEsperada BOOLEAN);
194     END PRUEBAS_A_PRODUCTO_SOLICITUD;
195
196 /
197
198 CREATE OR REPLACE PACKAGE PRUEBAS_A_PRODUCTO_TRASPASO AS
199     PROCEDURE inicializar;
200     PROCEDURE insertar
201         (nombre_prueba VARCHAR2, i_id_producto INT, i_id_Traspaso INT,
202          i_cantidad NUMBER, salidaEsperada BOOLEAN);
203     PROCEDURE eliminar
204         (nombre_prueba VARCHAR2, e_id_Traspaso INT, e_id_producto INT, salidaEsperada BOOLEAN);
205     END PRUEBAS_A_PRODUCTO_TRASPASO;
206
207 /
208
209 CREATE OR REPLACE PACKAGE PRUEBAS_A_PEDIDO_PRODUCTO AS
210     PROCEDURE inicializar;
211     PROCEDURE insertar
212         (nombre_prueba VARCHAR2, i_id_pedido INT, i_id_producto INT,
213          i_cantidad NUMBER, i_precioCompra number, i_iva INT,
214          salidaEsperada BOOLEAN);
215     PROCEDURE eliminar
216         (nombre_prueba VARCHAR2, e_id_pedido INT, e_id_producto INT,
217          salidaEsperada BOOLEAN);
218     END PRUEBAS_A_PEDIDO_PRODUCTO;
219
220 /
221
222 CREATE OR REPLACE PACKAGE PRUEBAS_FUNCIONES AS
223     PROCEDURE precioLinea_A_Pedido
224         (nombre_prueba VARCHAR2, ID_PRODUCTO NUMBER, ID_PEDIDO NUMBER, esperado number,
225          salidaEsperada BOOLEAN);
226     PROCEDURE precioLinea_A_Venta
227         (nombre_prueba VARCHAR2, ID_PRODUCTO NUMBER, ID_VENTA NUMBER, esperado number, salidaEsperada
228          BOOLEAN);
229     PROCEDURE precio_Venta
230         (nombre_prueba VARCHAR2, ID_Venta NUMBER, esperado number, salidaEsperada BOOLEAN);
231     PROCEDURE precio_Factura
232         (nombre_prueba VARCHAR2, ID_Venta NUMBER, esperado number, salidaEsperada BOOLEAN);
233     PROCEDURE precio_Pedido
234         (nombre_prueba VARCHAR2, ID_PEDIDO NUMBER, esperado number, salidaEsperada BOOLEAN);
235     PROCEDURE precio_Albaran
236         (nombre_prueba VARCHAR2, ID_PEDIDO NUMBER, esperado number, salidaEsperada BOOLEAN);
237     END PRUEBAS_FUNCIONES;
238
239 /
240
241 /* CUERPOS DE PRUEBAS */
242
243 CREATE OR REPLACE PACKAGE BODY PRUEBAS_SOCIO AS

```

```

238 PROCEDURE inicializar AS
239 BEGIN
240     DELETE FROM SOCIO;
241 END inicializar;
242
243 PROCEDURE insertar
244     (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_apellidos VARCHAR2,
245      i_DNI VARCHAR2, i_direccion VARCHAR2, i_nacimiento DATE, i_email VARCHAR2
246      ,salidaEsperada BOOLEAN) AS
247     salida BOOLEAN := true;
248     actual socio%ROWTYPE;
249 BEGIN
250     INSERT INTO socio VALUES (i_nombre, i_apellidos, i_direccion, i_nacimiento,
251                                i_email, i_DNI);
252
253     SELECT * INTO actual FROM SOCIO WHERE DNI = i_DNI;
254
255     IF (actual.nombre<>i_nombre) OR (actual.apellidos<>i_apellidos) OR (actual.DNI<>i_DNI) OR (
256     actual.direccion<>i_direccion) OR (actual.fechadenacimiento<>i_nacimiento) OR (actual.email<>
257     i_email) THEN
258         salida := false;
259     END IF;
260
261     PRINTR(nombre_prueba, salida, salidaEsperada);
262
263     EXCEPTION
264     WHEN OTHERS THEN
265         PRINTR(nombre_prueba, false, salidaEsperada);
266         ROLLBACK;
267 END insertar;
268
269 PROCEDURE actualizar
270     (nombre_prueba VARCHAR2, a_DNI VARCHAR2, a_direccion VARCHAR2,
271      a_email VARCHAR2, salidaEsperada BOOLEAN) AS
272     salida BOOLEAN := true;
273     actual socio%ROWTYPE;
274 BEGIN
275     UPDATE SOCIO SET DIRECCION = a_direccion, EMAIL = a_email where DNI = a_DNI;
276
277     SELECT * INTO actual FROM SOCIO WHERE DNI = a_DNI;
278
279     IF (actual.direccion<>a_direccion) OR (actual.email<>a_email) THEN
280         salida := false;
281     END IF;
282
283     PRINTR(nombre_prueba, salida, salidaEsperada);
284
285     EXCEPTION
286     WHEN OTHERS THEN
287         PRINTR(nombre_prueba, false, salidaEsperada);
288         ROLLBACK;
289 END actualizar;
290
291 PROCEDURE eliminar
292     (nombre_prueba VARCHAR2, e_DNI VARCHAR2, salidaEsperada BOOLEAN) AS
293     salida BOOLEAN := true;
294     cantidad INT;
295 BEGIN
296     DELETE FROM SOCIO WHERE DNI = e_DNI;
297
298     SELECT COUNT(*) INTO cantidad FROM SOCIO WHERE DNI = e_DNI;
299
300     IF (cantidad<>0) THEN
301         salida := false;
302     END IF;
303
304     PRINTR(nombre_prueba, salida, salidaEsperada);
305
306     EXCEPTION
307     WHEN OTHERS THEN
308         PRINTR(nombre_prueba, false, salidaEsperada);
309         ROLLBACK;
310 END eliminar;
311 END PRUEBAS_SOCIO;
312 /

```

```

312 CREATE OR REPLACE PACKAGE BODY PRUEBAS_PROVEEDOR AS
314
316     PROCEDURE inicializar AS
318     BEGIN
319         DELETE FROM PROVEEDOR;
320     END inicializar;
321
322     PROCEDURE insertar
323     (nombre_prueba VARCHAR2, i_CIF VARCHAR2, i_nombre VARCHAR2,
324      i_telefono INT, i_email VARCHAR2, salidaEsperada BOOLEAN) AS
325     salida BOOLEAN := true;
326     actual proveedor %ROWTYPE;
327     BEGIN
328         INSERT INTO PROVEEDOR VALUES (i_CIF, i_nombre, i_telefono, i_email);
329
330         SELECT * INTO actual FROM PROVEEDOR WHERE CIF = i_CIF;
331
332         IF (actual.nombre <> i_nombre) OR (actual.telefono <> i_telefono) OR (actual.CIF <> i_CIF) OR (
333         actual.email <> i_email) THEN
334             salida := false;
335         END IF;
336
337         PRINTR(nombre_prueba, salida, salidaEsperada);
338
339         EXCEPTION
340         WHEN OTHERS THEN
341             PRINTR(nombre_prueba, false, salidaEsperada);
342             ROLLBACK;
343     END insertar;
344
345     PROCEDURE actualizar
346     (nombre_prueba VARCHAR2, a_cif VARCHAR2, a_nombre VARCHAR2,
347      a_telefono INT, a_email VARCHAR2, salidaEsperada BOOLEAN) AS
348     salida BOOLEAN := true;
349     actual proveedor %ROWTYPE;
350     BEGIN
351         UPDATE PROVEEDOR SET NOMBRE = a_nombre, TELEFONO = a_telefono, EMAIL = a_email where CIF
352         = a_cif;
353
354         SELECT * INTO actual FROM PROVEEDOR WHERE CIF = a_cif;
355
356         IF (actual.nombre <> a_nombre) OR
357            (actual.telefono <> a_telefono) OR
358            (actual.email <> a_email) THEN
359             salida := false;
360         END IF;
361
362         PRINTR(nombre_prueba, salida, salidaEsperada);
363
364         EXCEPTION
365         WHEN OTHERS THEN
366             PRINTR(nombre_prueba, false, salidaEsperada);
367             ROLLBACK;
368     END actualizar;
369
370     PROCEDURE eliminar
371     (nombre_prueba VARCHAR2, e_CIF VARCHAR2, salidaEsperada BOOLEAN) AS
372     salida BOOLEAN := true;
373     cantidad INT;
374     BEGIN
375         DELETE FROM PROVEEDOR WHERE CIF = e_CIF;
376
377         SELECT COUNT(*) INTO cantidad FROM PROVEEDOR WHERE CIF = e_cif;
378
379         IF (cantidad <> 0) THEN
380             salida := false;
381         END IF;
382
383         PRINTR(nombre_prueba, salida, salidaEsperada);
384
385         EXCEPTION
386         WHEN OTHERS THEN
387             PRINTR(nombre_prueba, false, salidaEsperada);
388             ROLLBACK;
389     END eliminar;
390 END PRUEBAS_PROVEEDOR;

```



```

388 /
390 CREATE OR REPLACE PACKAGE BODY PRUEBAS_EMPLAZAMIENTO AS
392     PROCEDURE inicializar AS
394     BEGIN
395         DELETE FROM EMPLAZAMIENTO;
396     END inicializar;
398     PROCEDURE insertar
399         (nombre_prueba VARCHAR2, i_direccion VARCHAR2, i_telefono INT,
400          i_tipo VARCHAR2, salidaEsperada BOOLEAN) AS
401     salida BOOLEAN := true;
402     actual emplazamiento%ROWTYPE;
403     w_ID_Emplazamiento INT;
404     BEGIN
405         INSERT INTO emplazamiento VALUES(null,i_direccion, i_telefono, i_tipo);
406
407         w_ID_Emplazamiento := S_ID_Emplazamiento.currval;
408         SELECT * INTO actual FROM EMPLAZAMIENTO WHERE ID_EMPLAZAMIENTO = w_ID_Emplazamiento;
409
410         IF (actual.direccion<>i_direccion) OR (actual.telefono<>i_telefono) OR (actual.tipo<>i_tipo)
411         THEN
412             salida := false;
413         END IF;
414
415         PRINTR(nombre_prueba, salida, salidaEsperada);
416
417         EXCEPTION
418         WHEN OTHERS THEN
419             PRINTR(nombre_prueba, false, salidaEsperada);
420             ROLLBACK;
421     END insertar;
422
423     PROCEDURE actualizar
424         (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_direccion VARCHAR2,
425          a_telefono INT, salidaEsperada BOOLEAN)AS
426     salida BOOLEAN := true;
427     actual emplazamiento%ROWTYPE;
428     BEGIN
429         UPDATE emplazamiento SET DIRECCION = a_direccion, TELEFONO = a_telefono where
430         ID_EMPLAZAMIENTO = a_id_emplazamiento;
431
432         SELECT * INTO actual FROM EMPLAZAMIENTO WHERE ID_EMPLAZAMIENTO = a_id_emplazamiento;
433
434         IF (actual.direccion<>a_direccion) OR
435            (actual.telefono<>a_telefono) THEN
436             salida := false;
437         END IF;
438
439         PRINTR(nombre_prueba, salida, salidaEsperada);
440
441         EXCEPTION
442         WHEN OTHERS THEN
443             PRINTR(nombre_prueba, false, salidaEsperada);
444             ROLLBACK;
445     END actualizar;
446
447     PROCEDURE eliminar
448         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, salidaEsperada BOOLEAN) AS
449     salida BOOLEAN := true;
450     cantidad INT;
451     BEGIN
452         DELETE FROM EMPLAZAMIENTO WHERE ID_EMPLAZAMIENTO = e_id_emplazamiento;
453
454         SELECT COUNT(*) INTO cantidad FROM EMPLAZAMIENTO WHERE ID_EMPLAZAMIENTO =
455         e_id_emplazamiento;
456
457         IF (cantidad<>0) THEN
458             salida := false;
459         END IF;
460
461         PRINTR(nombre_prueba, salida, salidaEsperada);
462
463         EXCEPTION

```

```

462         WHEN OTHERS THEN
463             PRINTR(nombre_prueba, false, salidaEsperada);
464             ROLLBACK;
465     END eliminar;
466 END PRUEBAS_EMPLAZAMIENTO;
467
468 /
469
470 CREATE OR REPLACE PACKAGE BODY PRUEBAS_PRODUCTO AS
471
472     PROCEDURE inicializar AS
473     BEGIN
474         DELETE FROM PRODUCTO;
475     END inicializar;
476
477     PROCEDURE insertar
478     (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_descripcion VARCHAR2,
479      i_categoria VARCHAR2, i_precioProducto INT, i_iva INT, salidaEsperada BOOLEAN) AS
480     salida BOOLEAN := true;
481     actual_producto %ROWTYPE;
482     w_ID_Producto INT;
483     BEGIN
484         INSERT INTO producto VALUES(null,i_nombre, i_descripcion, i_categoria, i_precioProducto,
485         i_iva);
486
487         w_ID_Producto := S_ID_Producto.currval;
488         SELECT * INTO actual FROM PRODUCTO WHERE ID_Producto = w_ID_Producto;
489
490         IF (actual.nombre<>i_nombre) OR (actual.descripcion<>i_descripcion) OR (actual.categoria<>
491         i_categoria) OR (actual.precioProducto<>i_precioProducto) OR (actual.iva<>i_iva) THEN
492             salida := false;
493         END IF;
494
495         PRINTR(nombre_prueba, salida, salidaEsperada);
496
497     EXCEPTION
498     WHEN OTHERS THEN
499         PRINTR(nombre_prueba, false, salidaEsperada);
500         ROLLBACK;
501     END insertar;
502
503     PROCEDURE actualizar
504     (nombre_prueba VARCHAR2, a_id_producto INT, a_descripcion VARCHAR2,
505      a_precioProducto INT, a_iva INT, salidaEsperada BOOLEAN) AS
506     salida BOOLEAN := true;
507     actual_producto %ROWTYPE;
508     BEGIN
509         UPDATE producto SET DESCRIPCION = a_descripcion, PRECIOPRODUCTO = a_precioProducto, IVA =
510         a_iva
511         where ID_Producto = a_id_producto;
512
513         SELECT * INTO actual FROM PRODUCTO WHERE ID_Producto = a_id_producto;
514         IF (actual.descripcion<>a_descripcion) OR
515         (actual.precioProducto<>a_precioProducto) OR
516         (actual.iva<>a_iva) THEN
517             salida := false;
518         END IF;
519
520         PRINTR(nombre_prueba, salida, salidaEsperada);
521
522     EXCEPTION
523     WHEN OTHERS THEN
524         PRINTR(nombre_prueba, false, salidaEsperada);
525         ROLLBACK;
526     END actualizar;
527
528     PROCEDURE eliminar
529     (nombre_prueba VARCHAR2, e_id_producto INT, salidaEsperada BOOLEAN) AS
530     salida BOOLEAN := true;
531     cantidad INT;
532     BEGIN
533         DELETE FROM PRODUCTO WHERE ID_Producto = e_id_producto;
534
535         SELECT COUNT(*) INTO cantidad FROM PRODUCTO WHERE ID_Producto = e_id_producto;
536         IF (cantidad<>0) THEN
537             salida := false;
538         END IF;

```

```

536         PRINTR(nombre_prueba, salida, salidaEsperada);
537     EXCEPTION
538     WHEN OTHERS THEN
540         PRINTR(nombre_prueba, false, salidaEsperada);
541         ROLLBACK;
542     END eliminar;
543 END PRUEBAS_PRODUCTO;
544
545 /
546
547 CREATE OR REPLACE PACKAGE BODY PRUEBAS_FACTURA AS
548
549     PROCEDURE inicializar AS
550     BEGIN
551         DELETE FROM FACTURA;
552     END inicializar;
553
554     PROCEDURE insertar
555     (nombre_prueba VARCHAR2, i_fechaDeExpedicion DATE, i_devuelto VARCHAR2,
556      i_id_venta INT, i_id_emplazamiento INT, salidaEsperada BOOLEAN) AS
557     salida BOOLEAN := true;
558     actual factura%ROWTYPE;
559     w_ID_factura INT;
560     BEGIN
561         INSERT INTO factura VALUES(null, i_fechaDeExpedicion, i_devuelto, i_id_venta,
562                                     i_id_emplazamiento);
563
564         w_ID_factura := S_ID_Factura.currval;
565         SELECT * INTO actual FROM factura WHERE ID_factura = w_ID_factura;
566
567         IF (actual.fechaDeExpedicion<>i_fechaDeExpedicion) OR (actual.devuelto<>i_devuelto) OR (
568             actual.id_venta<>i_id_venta) OR (actual.id_emplazamiento<>i_id_emplazamiento) THEN
569             salida := false;
570         END IF;
571
572         PRINTR(nombre_prueba, salida, salidaEsperada);
573
574     EXCEPTION
575     WHEN OTHERS THEN
576         PRINTR(nombre_prueba, false, salidaEsperada);
577         ROLLBACK;
578     END insertar;
579
580     PROCEDURE actualizar
581     (nombre_prueba VARCHAR2, a_id_factura INT, a_devuelto VARCHAR2,
582      salidaEsperada BOOLEAN) AS
583     salida BOOLEAN := true;
584     actual factura%ROWTYPE;
585     BEGIN
586         UPDATE factura SET DEVUELTO = a_devuelto where ID_factura = a_id_factura;
587
588         SELECT * INTO actual FROM factura WHERE ID_factura = a_id_factura;
589         IF (actual.devuelto<>a_devuelto) THEN
590             salida := false;
591         END IF;
592
593         PRINTR(nombre_prueba, salida, salidaEsperada);
594
595     EXCEPTION
596     WHEN OTHERS THEN
597         PRINTR(nombre_prueba, false, salidaEsperada);
598         ROLLBACK;
599     END actualizar;
600
601     PROCEDURE eliminar
602     (nombre_prueba VARCHAR2, e_id_factura INT, salidaEsperada BOOLEAN) AS
603     salida BOOLEAN := true;
604     cantidad INT;
605     BEGIN
606         DELETE FROM factura WHERE ID_factura = e_id_factura;
607
608         SELECT COUNT(*) INTO cantidad FROM factura WHERE ID_factura = e_id_factura;
609         IF (cantidad<>0) THEN
610             salida := false;
611         END IF;

```

```

610         PRINTR(nombre_prueba, salida, salidaEsperada);
612     EXCEPTION
614     WHEN OTHERS THEN
615         PRINTR(nombre_prueba, false, salidaEsperada);
616         ROLLBACK;
617     END eliminar;
618 END PRUEBAS_factura;
620 /
622 CREATE OR REPLACE PACKAGE BODY PRUEBA_STOCK AS
623     PROCEDURE inicializar AS
624     BEGIN
625         DELETE FROM stock;
626     END inicializar;
628     PROCEDURE insertar
629     (nombre_prueba VARCHAR2, i_id_emplazamiento INT, i_id_producto INT,
630      i_cantidad INT, salidaEsperada BOOLEAN) AS
631     salida BOOLEAN := true;
632     actual stock%ROWTYPE;
633     BEGIN
634         INSERT INTO stock VALUES(i_id_emplazamiento, i_id_producto, i_cantidad);
636         SELECT * INTO actual FROM stock WHERE ID_Emplazamiento = i_id_emplazamiento and ID_Producto =
        i_id_producto;
638         IF (actual.id_emplazamiento<>i_id_emplazamiento) OR (actual.id_producto<>i_id_producto) OR (
        actual.cantidad<>i_cantidad) THEN
639             salida := false;
640         END IF;
642         PRINTR(nombre_prueba, salida, salidaEsperada);
644     EXCEPTION
645     WHEN OTHERS THEN
646         PRINTR(nombre_prueba, false, salidaEsperada);
647         ROLLBACK;
648     END insertar;
650     PROCEDURE actualizar
651     (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_id_producto INT,
652      a_cantidad INT, salidaEsperada BOOLEAN) AS
653     salida BOOLEAN := true;
654     actual stock%ROWTYPE;
655     BEGIN
656         UPDATE stock SET CANTIDAD = a_cantidad where ID_Emplazamiento = a_id_emplazamiento and
        ID_Producto = a_id_producto;
658         SELECT * INTO actual FROM stock WHERE ID_Emplazamiento = a_id_emplazamiento and
        ID_Producto = a_id_producto;
659         IF (actual.cantidad<>a_cantidad) THEN
660             salida := false;
661         END IF;
662         PRINTR(nombre_prueba, salida, salidaEsperada);
664     EXCEPTION
665     WHEN OTHERS THEN
666         PRINTR(nombre_prueba, false, salidaEsperada);
667         ROLLBACK;
668     END actualizar;
670     PROCEDURE eliminar
671     (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT,
672      salidaEsperada BOOLEAN) AS
673     salida BOOLEAN := true;
674     cantidad INT;
675     BEGIN
676         DELETE FROM stock WHERE ID_Emplazamiento = e_id_emplazamiento and ID_Producto =
        e_id_producto;
678         SELECT COUNT(*) INTO cantidad FROM stock WHERE ID_Emplazamiento = e_id_emplazamiento and
        ID_Producto = e_id_producto;
680         IF (cantidad<>0) THEN

```

```

682         salida := false;
        END IF;

684         PRINTR(nombre_prueba, salida, salidaEsperada);

686         EXCEPTION
        WHEN OTHERS THEN
688             PRINTR(nombre_prueba, false, salidaEsperada);
            ROLLBACK;
690     END eliminar;

692 PROCEDURE stock_correcto
    (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
    INT, salidaEsperada BOOLEAN) AS
694     salida BOOLEAN := true;
    stock_nuevo int;
696     BEGIN
        select cantidad into stock_nuevo from stock where id_emplazamiento = e_id_emplazamiento and
        id_producto= e_id_producto;
698         if (stock_previo-cantidad)<> stock_nuevo then
            salida := false;
700         END IF;

702         PRINTR(nombre_prueba, salida, salidaEsperada);

704         EXCEPTION
        WHEN OTHERS THEN
706             PRINTR(nombre_prueba, false, salidaEsperada);
            ROLLBACK;
708     END stock_correcto;

710 PROCEDURE stock_correcto_p
    (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
    INT, salidaEsperada BOOLEAN) AS
712     salida BOOLEAN := true;
    stock_nuevo int;
714     BEGIN
        select cantidad into stock_nuevo from stock where id_emplazamiento = e_id_emplazamiento and
        id_producto= e_id_producto;
716         if (stock_previo+cantidad)<> stock_nuevo then
            salida := false;
718         END IF;

720         PRINTR(nombre_prueba, salida, salidaEsperada);

722         EXCEPTION
        WHEN OTHERS THEN
724             PRINTR(nombre_prueba, false, salidaEsperada);
            ROLLBACK;
726     end stock_correcto_p;

728 PROCEDURE stock_correcto_t
    (nombre_prueba VARCHAR2, e_id_emplazamiento_envia int, e_id_emplazamiento_recibe int,
    stock_previo_envia int, stock_previo_recibe int,
    cantidad int, e_id_producto int, salidaEsperada BOOLEAN) AS
730     salida BOOLEAN := true;
    stock_nuevo_envia int;
    stock_nuevo_recibe int;
732     BEGIN
        select cantidad into stock_nuevo_envia from stock where id_emplazamiento =
        e_id_emplazamiento_envia and id_producto = e_id_producto;
        select cantidad into stock_nuevo_recibe from stock where id_emplazamiento =
        e_id_emplazamiento_recibe and id_producto = e_id_producto;
734         if (stock_previo_envia-cantidad)<>stock_nuevo_envia or (stock_previo_recibe+cantidad)<>
        stock_nuevo_recibe then
            salida := false;
736         END IF;
        PRINTR(nombre_prueba, salida, salidaEsperada);
738         EXCEPTION
        WHEN OTHERS THEN
740             PRINTR(nombre_prueba, false, salidaEsperada);
            ROLLBACK;
742     end stock_correcto_t;

744     end PRUEBA_STOCK;

```

```

750 /
752
754 CREATE OR REPLACE PACKAGE BODY PRUEBAS_VENTA AS
756     PROCEDURE inicializar AS
757     BEGIN
758         DELETE FROM VENTA;
759     END inicializar;
760
761     PROCEDURE insertar
762     (nombre_prueba VARCHAR2, i_fecha DATE, i_dni VARCHAR2,
763      salidaEsperada BOOLEAN) AS
764     salida BOOLEAN := true;
765     actual_venta%ROWTYPE;
766     w_ID_Venta INT;
767     BEGIN
768         INSERT INTO venta VALUES(null, i_fecha, i_dni);
769
770         w_ID_Venta := S_ID_Venta.currval;
771         SELECT * INTO actual FROM venta WHERE ID_Venta = w_ID_Venta;
772
773         IF (actual.fechaVenta <> i_fecha) OR (actual.dni <> i_dni) THEN
774             salida := false;
775         END IF;
776
777         PRINTR(nombre_prueba, salida, salidaEsperada);
778
779         EXCEPTION
780         WHEN OTHERS THEN
781             PRINTR(nombre_prueba, false, salidaEsperada);
782             ROLLBACK;
783     END insertar;
784
785     PROCEDURE eliminar
786     (nombre_prueba VARCHAR2, e_id_venta INT, salidaEsperada BOOLEAN) AS
787     salida BOOLEAN := true;
788     cantidad INT;
789     BEGIN
790         DELETE FROM venta WHERE ID_Venta = e_id_venta;
791
792         SELECT COUNT(*) INTO cantidad FROM venta WHERE ID_Venta = e_id_venta;
793         IF (cantidad <> 0) THEN
794             salida := false;
795         END IF;
796
797         PRINTR(nombre_prueba, salida, salidaEsperada);
798
799         EXCEPTION
800         WHEN OTHERS THEN
801             PRINTR(nombre_prueba, false, salidaEsperada);
802             ROLLBACK;
803     END eliminar;
804
805     /*
806     procedure descuento
807     (nombre_prueba VARCHAR2, v_id_venta int, v_preciototal number, v_dni_socio varchar2,
808      salidaEsperada BOOLEAN) as
809     salida BOOLEAN := true;
810     p_preciototal number;
811     begin
812         select preciototal into p_preciototal from venta where id_venta = v_id_venta;
813         if (v_dni_socio <> null) then
814             if (v_preciototal * 0.95 <> p_preciototal) then
815                 salida := false;
816             END IF;
817         END IF;
818
819         printr(nombre_prueba, salida, salidaEsperada);
820         EXCEPTION
821         WHEN OTHERS THEN
822             PRINTR(nombre_prueba, false, salidaEsperada);
823             ROLLBACK;
824     end descuento;*/
825 END PRUEBAS_VENTA;
826 /

```

```

826 CREATE OR REPLACE PACKAGE BODY PRUEBAS_PEDIDO AS
828     PROCEDURE inicializar AS
829     BEGIN
830         DELETE FROM pedido;
831     END inicializar;
832
833     PROCEDURE insertar
834     (nombre_prueba VARCHAR2, i_fecha DATE,
835      i_id_emplazamiento INT, i_cif VARCHAR2, salidaEsperada BOOLEAN) AS
836     salida BOOLEAN := true;
837     actual_pedido %ROWTYPE;
838     w_ID_pedido INT;
839     BEGIN
840         INSERT INTO pedido VALUES(null, i_fecha, i_id_emplazamiento, i_cif);
841
842         w_ID_pedido := S_ID_Pedido.currval;
843         SELECT * INTO actual FROM pedido WHERE ID_Pedido = w_ID_pedido;
844
845         IF (actual.fechaPedido<>i_fecha) OR (actual.id_emplazamiento<>i_id_emplazamiento) OR (actual.
846            cif<>i_cif) THEN
847             salida := false;
848         END IF;
849
850         PRINTR(nombre_prueba, salida, salidaEsperada);
851
852         EXCEPTION
853         WHEN OTHERS THEN
854             PRINTR(nombre_prueba, false, salidaEsperada);
855             ROLLBACK;
856     END insertar;
857
858     PROCEDURE eliminar
859     (nombre_prueba VARCHAR2, e_id_pedido INT, salidaEsperada BOOLEAN) AS
860     salida BOOLEAN := true;
861     cantidad INT;
862     BEGIN
863         DELETE FROM pedido WHERE ID_Pedido = e_id_pedido;
864
865         SELECT COUNT(*) INTO cantidad FROM pedido WHERE ID_Pedido = e_id_pedido;
866         IF (cantidad<>0) THEN
867             salida := false;
868         END IF;
869
870         PRINTR(nombre_prueba, salida, salidaEsperada);
871
872         EXCEPTION
873         WHEN OTHERS THEN
874             PRINTR(nombre_prueba, false, salidaEsperada);
875             ROLLBACK;
876     END eliminar;
877 END PRUEBAS_PEDIDO;
878 /
879
880 CREATE OR REPLACE PACKAGE BODY PRUEBAS_TRASPASO AS
882     PROCEDURE inicializar AS
883     BEGIN
884         DELETE FROM traspaso;
885     END inicializar;
886
887     PROCEDURE insertar
888     (nombre_prueba VARCHAR2, i_fechaTraspaso DATE, i_id_emplazamientoSalida INT,
889      i_id_emplazamientoEntrada INT, salidaEsperada BOOLEAN) AS
890     salida BOOLEAN := true;
891     actual_traspaso %ROWTYPE;
892     w_ID_traspaso INT;
893     BEGIN
894         INSERT INTO traspaso VALUES(null, i_fechaTraspaso, i_id_emplazamientoSalida,
895            i_id_emplazamientoEntrada);
896
897         w_ID_traspaso := S_ID_traspaso.currval;
898         SELECT * INTO actual FROM traspaso WHERE ID_traspaso = w_ID_traspaso;
899
900         IF (actual.fechaTraspaso<>i_fechaTraspaso) OR (actual.id_emplazamientoSalida<>
901            i_id_emplazamientoSalida) OR (actual.id_emplazamientoEntrada<>i_id_emplazamientoEntrada) THEN

```

```

900         salida := false;
901     END IF;
902
903     PRINTR(nombre_prueba, salida, salidaEsperada);
904
905     EXCEPTION
906     WHEN OTHERS THEN
907         PRINTR(nombre_prueba, false, salidaEsperada);
908         ROLLBACK;
909 END insertar;
910
911 PROCEDURE eliminar
912 (nombre_prueba VARCHAR2, e_id_Traspaso INT, salidaEsperada BOOLEAN) AS
913     salida BOOLEAN := true;
914     cantidad INT;
915 BEGIN
916     DELETE FROM traspaso WHERE ID_traspaso = e_id_traspaso;
917
918     SELECT COUNT(*) INTO cantidad FROM traspaso WHERE ID_traspaso = e_id_traspaso;
919     IF (cantidad <> 0) THEN
920         salida := false;
921     END IF;
922
923     PRINTR(nombre_prueba, salida, salidaEsperada);
924
925     EXCEPTION
926     WHEN OTHERS THEN
927         PRINTR(nombre_prueba, false, salidaEsperada);
928         ROLLBACK;
929 END eliminar;
930 END PRUEBAS_TRASPASO;
931
932 /
933
934 CREATE OR REPLACE PACKAGE BODY PRUEBAS_Solicitud_Traspaso AS
935
936     PROCEDURE inicializar AS
937     BEGIN
938         DELETE FROM Solicitud_Traspaso;
939     END inicializar;
940
941     PROCEDURE insertar
942     (nombre_prueba VARCHAR2, i_fechaSolicitud DATE, i_id_emplazamientoSalida INT,
943      i_id_emplazamientoEntrada INT, salidaEsperada BOOLEAN) AS
944     salida BOOLEAN := true;
945     actual Solicitud_Traspaso%ROWTYPE;
946     w_ID_Solicitud INT;
947 BEGIN
948     INSERT INTO Solicitud_Traspaso VALUES(null, i_fechaSolicitud, i_id_emplazamientoSalida,
949      i_id_emplazamientoEntrada);
950
951     w_ID_Solicitud := S_ID_Solicitud.currval;
952     SELECT * INTO actual FROM Solicitud_Traspaso WHERE ID_Solicitud = w_ID_Solicitud;
953
954     IF (actual.fechaSolicitud <> i_fechaSolicitud) OR (actual.id_emplazamientoSalida <>
955      i_id_emplazamientoSalida) OR (actual.id_emplazamientoEntrada <> i_id_emplazamientoEntrada) THEN
956         salida := false;
957     END IF;
958
959     PRINTR(nombre_prueba, salida, salidaEsperada);
960
961     EXCEPTION
962     WHEN OTHERS THEN
963         PRINTR(nombre_prueba, false, salidaEsperada);
964         ROLLBACK;
965 END insertar;
966
967 PROCEDURE eliminar
968 (nombre_prueba VARCHAR2, e_id_solicitudTraspaso INT, salidaEsperada BOOLEAN) AS
969     salida BOOLEAN := true;
970     cantidad INT;
971 BEGIN
972     DELETE FROM Solicitud_Traspaso WHERE ID_Solicitud = e_id_solicitudTraspaso;
973
974     SELECT COUNT(*) INTO cantidad FROM Solicitud_Traspaso WHERE ID_Solicitud =
975     e_id_solicitudTraspaso;
976     IF (cantidad <> 0) THEN

```



```

974         salida := false;
975     END IF;
976
977     PRINTR(nombre_prueba, salida, salidaEsperada);
978
979     EXCEPTION
980     WHEN OTHERS THEN
981         PRINTR(nombre_prueba, false, salidaEsperada);
982         ROLLBACK;
983     END eliminar;
984 END PRUEBAS_Solicitud_Traspaso;
985
986 /
987
988 CREATE OR REPLACE PACKAGE BODY PRUEBAS_Albaran AS
989
990     PROCEDURE inicializar AS
991     BEGIN
992         DELETE FROM Albaran;
993     END inicializar;
994
995     PROCEDURE insertar
996     (nombre_prueba VARCHAR2, i_fecha DATE,
997      i_id_pedido INT, salidaEsperada BOOLEAN) AS
998     salida BOOLEAN := true;
999     actual Albaran%ROWTYPE;
1000     w_ID_Albaran INT;
1001     BEGIN
1002         INSERT INTO Albaran VALUES(null, i_fecha, i_id_pedido);
1003
1004         w_ID_Albaran := S_ID_Albaran.currval;
1005         SELECT * INTO actual FROM Albaran WHERE ID_Albaran = w_ID_Albaran;
1006
1007         IF (actual.fechaFirma<>i_fecha) OR (actual.id_pedido<>i_id_pedido) THEN
1008             salida := false;
1009         END IF;
1010
1011         PRINTR(nombre_prueba, salida, salidaEsperada);
1012
1013         EXCEPTION
1014         WHEN OTHERS THEN
1015             PRINTR(nombre_prueba, false, salidaEsperada);
1016             ROLLBACK;
1017     END insertar;
1018
1019     PROCEDURE eliminar
1020     (nombre_prueba VARCHAR2, e_id_albaran INT, salidaEsperada BOOLEAN) AS
1021     salida BOOLEAN := true;
1022     cantidad INT;
1023     BEGIN
1024         DELETE FROM Albaran WHERE ID_Albaran = e_id_albaran;
1025
1026         SELECT COUNT(*) INTO cantidad FROM Albaran WHERE ID_Albaran = e_id_albaran;
1027         IF (cantidad<>0) THEN
1028             salida := false;
1029         END IF;
1030
1031         PRINTR(nombre_prueba, salida, salidaEsperada);
1032
1033         EXCEPTION
1034         WHEN OTHERS THEN
1035             PRINTR(nombre_prueba, false, salidaEsperada);
1036             ROLLBACK;
1037     END eliminar;
1038 END PRUEBAS_Albaran;
1039
1040 /
1041
1042 CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_PRODUCTO_TRASPASO AS
1043
1044     PROCEDURE inicializar AS
1045     BEGIN
1046         DELETE FROM asociacion_Producto_traspaso;
1047     END inicializar;
1048
1049     PROCEDURE insertar
1050     (nombre_prueba VARCHAR2, i_id_producto INT, i_id_Traspaso INT,

```

```

1052         i_cantidad NUMBER, salidaEsperada BOOLEAN) AS
1053     salida BOOLEAN := true;
1054     actual asociacion_Producto_traspaso %ROWTYPE;
1055     BEGIN
1056         INSERT INTO asociacion_Producto_traspaso VALUES(i_id_Traspaso, i_id_producto, i_cantidad);
1057
1058         SELECT * INTO actual FROM asociacion_Producto_traspaso WHERE ID_Producto = i_id_producto and
1059             ID_Traspaso = i_id_Traspaso;
1060
1061         IF (actual.id_producto<>i_id_producto) OR (actual.id_Traspaso<>i_id_Traspaso) OR (actual.
1062             cantidad<>i_cantidad) THEN
1063             salida := false;
1064         END IF;
1065
1066         PRINTR(nombre_prueba, salida, salidaEsperada);
1067
1068         EXCEPTION
1069         WHEN OTHERS THEN
1070             PRINTR(nombre_prueba, false, salidaEsperada);
1071             ROLLBACK;
1072     END insertar;
1073
1074     PROCEDURE eliminar
1075     (nombre_prueba VARCHAR2, e_id_Traspaso INT, e_id_producto INT, salidaEsperada BOOLEAN) AS
1076     salida BOOLEAN := true;
1077     cantidad INT;
1078     BEGIN
1079         DELETE FROM asociacion_Producto_traspaso WHERE ID_Producto = e_id_producto and
1080             ID_Traspaso = e_id_Traspaso;
1081
1082         SELECT COUNT(*) INTO cantidad FROM asociacion_Producto_traspaso WHERE ID_Producto =
1083             e_id_producto and ID_Traspaso = e_id_Traspaso;
1084         IF (cantidad<>0) THEN
1085             salida := false;
1086         END IF;
1087
1088         PRINTR(nombre_prueba, salida, salidaEsperada);
1089
1090         EXCEPTION
1091         WHEN OTHERS THEN
1092             PRINTR(nombre_prueba, false, salidaEsperada);
1093             ROLLBACK;
1094     END eliminar;
1095 END PRUEBAS_A_PRODUCTO_TRASPASO;
1096
1097 /
1098
1099 CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_PRODUCTO_SOLICITUD AS
1100
1101     PROCEDURE inicializar AS
1102     BEGIN
1103         DELETE FROM asociacion_Producto_Solicitud;
1104     END inicializar;
1105
1106     PROCEDURE insertar
1107     (nombre_prueba VARCHAR2, i_id_producto INT, i_id_Solicitud INT,
1108         i_cantidad NUMBER, salidaEsperada BOOLEAN) AS
1109     salida BOOLEAN := true;
1110     actual asociacion_Producto_Solicitud %ROWTYPE;
1111     BEGIN
1112         INSERT INTO asociacion_Producto_Solicitud VALUES(i_id_Solicitud, i_id_producto, i_cantidad);
1113
1114         SELECT * INTO actual FROM asociacion_Producto_Solicitud WHERE ID_Producto = i_id_producto and
1115             ID_Solicitud = i_id_Solicitud;
1116
1117         IF (actual.id_producto<>i_id_producto) OR (actual.id_Solicitud<>i_id_Solicitud) OR (actual.
1118             cantidad<>i_cantidad) THEN
1119             salida := false;
1120         END IF;
1121
1122         PRINTR(nombre_prueba, salida, salidaEsperada);
1123
1124         EXCEPTION
1125         WHEN OTHERS THEN
1126             PRINTR(nombre_prueba, false, salidaEsperada);
1127             ROLLBACK;
1128     END insertar;

```

```

1122
1123 PROCEDURE eliminar
1124     (nombre_prueba VARCHAR2, e_id_Solicitud INT, e_id_producto INT, salidaEsperada BOOLEAN) AS
1125     salida BOOLEAN := true;
1126     cantidad INT;
1127     BEGIN
1128         DELETE FROM asociacion_Producto_Solicitud WHERE ID_Producto = e_id_producto and
1129         ID_Solicitud = e_id_Solicitud;
1130
1131         SELECT COUNT(*) INTO cantidad FROM asociacion_Producto_Solicitud WHERE ID_Producto =
1132         e_id_producto and ID_Solicitud = e_id_Solicitud;
1133         IF (cantidad<>0) THEN
1134             salida := false;
1135         END IF;
1136
1137         PRINTR(nombre_prueba, salida, salidaEsperada);
1138
1139         EXCEPTION
1140         WHEN OTHERS THEN
1141             PRINTR(nombre_prueba, false, salidaEsperada);
1142             ROLLBACK;
1143     END eliminar;
1144 END PRUEBAS_A_PRODUCTO_SOLICITUD;
1145
1146 /
1147
1148 CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_PEDIDO_PRODUCTO AS
1149
1150     PROCEDURE inicializar AS
1151     BEGIN
1152         DELETE FROM asociacion_Pedido_Producto;
1153     END inicializar;
1154
1155     PROCEDURE insertar
1156     (nombre_prueba VARCHAR2, i_id_pedido INT, i_id_producto INT,
1157     i_cantidad NUMBER, i_precioCompra number, i_iva INT,
1158     salidaEsperada BOOLEAN) AS
1159     salida BOOLEAN := true;
1160     actual asociacion_Pedido_Producto%ROWTYPE;
1161     BEGIN
1162         INSERT INTO asociacion_Pedido_Producto VALUES(i_id_pedido, i_id_producto, i_cantidad,
1163         i_precioCompra, i_iva);
1164
1165         SELECT * INTO actual FROM asociacion_Pedido_Producto WHERE ID_pedido = i_id_pedido and
1166         ID_producto = i_id_producto;
1167
1168         IF (actual.id_pedido<>i_id_pedido) OR (actual.id_producto<>i_id_producto) OR (actual.cantidad
1169         <>i_cantidad) OR (actual.precioCompra<>i_precioCompra) OR (actual.iva<>i_iva) THEN
1170             salida := false;
1171         END IF;
1172
1173         PRINTR(nombre_prueba, salida, salidaEsperada);
1174
1175         EXCEPTION
1176         WHEN OTHERS THEN
1177             PRINTR(nombre_prueba, false, salidaEsperada);
1178             ROLLBACK;
1179     END insertar;
1180
1181     PROCEDURE eliminar
1182     (nombre_prueba VARCHAR2, e_id_pedido INT, e_id_producto INT,
1183     salidaEsperada BOOLEAN) AS
1184     salida BOOLEAN := true;
1185     cantidad INT;
1186     BEGIN
1187         DELETE FROM asociacion_Pedido_Producto WHERE ID_pedido = e_id_pedido and ID_producto =
1188         e_id_producto;
1189
1190         SELECT COUNT(*) INTO cantidad FROM asociacion_Pedido_Producto WHERE ID_pedido =
1191         e_id_pedido and ID_producto = e_id_producto;
1192         IF (cantidad<>0) THEN
1193             salida := false;
1194         END IF;
1195
1196         PRINTR(nombre_prueba, salida, salidaEsperada);
1197
1198         EXCEPTION

```

```

1192         WHEN OTHERS THEN
1193             PRINTR(nombre_prueba, false, salidaEsperada);
1194             ROLLBACK;
1195     END eliminar;
1196
1197 END PRUEBAS_A_PEDIDO_PRODUCTO;
1198
1199 /
1200
1201 CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_VENTA_PRODUCTO AS
1202
1203     PROCEDURE inicializar AS
1204     BEGIN
1205         DELETE FROM asociacion_Venta_Producto;
1206     END inicializar;
1207
1208     PROCEDURE insertar
1209     (nombre_prueba VARCHAR2, i_id_venta INT, i_id_producto INT,
1210      i_cantidad NUMBER, i_precioVenta NUMBER, i_ivaVenta NUMBER,
1211      salidaEsperada BOOLEAN) AS
1212     salida BOOLEAN := true;
1213     actual asociacion_Venta_Producto %ROWTYPE;
1214     BEGIN
1215         INSERT INTO asociacion_Venta_Producto VALUES(i_id_venta, i_id_producto, i_cantidad,
1216             i_precioVenta, i_ivaVenta);
1217
1218         SELECT * INTO actual FROM asociacion_Venta_Producto WHERE ID_Venta = i_id_venta and
1219             ID_producto = i_id_producto;
1220
1221         IF (actual.id_venta <> i_id_venta) OR (actual.id_producto <> i_id_producto) OR (actual.cantidad <>
1222             i_cantidad) OR (actual.precioVenta <> i_precioVenta) OR (actual.ivaVenta <> i_ivaVenta) THEN
1223             salida := false;
1224         END IF;
1225
1226         PRINTR(nombre_prueba, salida, salidaEsperada);
1227
1228     EXCEPTION
1229     WHEN OTHERS THEN
1230         PRINTR(nombre_prueba, false, salidaEsperada);
1231         ROLLBACK;
1232     END insertar;
1233
1234     PROCEDURE eliminar
1235     (nombre_prueba VARCHAR2, e_id_venta INT, e_id_producto INT, salidaEsperada BOOLEAN) AS
1236     salida BOOLEAN := true;
1237     cantidad INT;
1238     BEGIN
1239         DELETE FROM asociacion_Venta_Producto WHERE ID_Venta = e_id_venta and ID_producto =
1240             e_id_producto;
1241
1242         SELECT COUNT(*) INTO cantidad FROM asociacion_Venta_Producto WHERE ID_Venta = e_id_venta
1243             and ID_producto = e_id_producto;
1244         IF (cantidad <> 0) THEN
1245             salida := false;
1246         END IF;
1247
1248         PRINTR(nombre_prueba, salida, salidaEsperada);
1249
1250     EXCEPTION
1251     WHEN OTHERS THEN
1252         PRINTR(nombre_prueba, false, salidaEsperada);
1253         ROLLBACK;
1254     END eliminar;
1255 END PRUEBAS_A_VENTA_PRODUCTO;
1256
1257 /
1258
1259 CREATE OR REPLACE PACKAGE BODY PRUEBAS_FUNCIONES AS
1260
1261     PROCEDURE precioLinea_A_Pedido
1262     (nombre_prueba VARCHAR2, ID_PRODUCTO NUMBER, ID_PEDIDO NUMBER, esperado number, salidaEsperada
1263      BOOLEAN) AS
1264     salida BOOLEAN := true;
1265     precio number;
1266     BEGIN

```

```

1264         SELECT precioLinea_Aso_Pedido(ID_PRODUCTO, ID_PEDIDO) INTO precio FROM dual;
1266         IF (precio <> esperado) THEN
1268             salida := false;
1268         END IF;
1270         PRINTR(nombre_prueba, salida, salidaEsperada);
1272         EXCEPTION
1274         WHEN OTHERS THEN
1276             PRINTR(nombre_prueba, false, salidaEsperada);
1276             ROLLBACK;
1276         END PRECIOLINEA_A_PEDIDO;
1278
1278     PROCEDURE precioLinea_A_Venta
1279     (nombre_prueba VARCHAR2, ID_PRODUCTO NUMBER, ID_VENTA NUMBER, esperado number, salidaEsperada
1280     BOOLEAN) AS
1280         salida BOOLEAN := true;
1280     precio number;
1280     BEGIN
1282         SELECT precioLinea_Aso_VENTA(ID_PRODUCTO, ID_VENTA) INTO precio FROM dual;
1284         IF (precio <> esperado) THEN
1286             salida := false;
1286         END IF;
1288         PRINTR(nombre_prueba, salida, salidaEsperada);
1290         EXCEPTION
1292         WHEN OTHERS THEN
1294             PRINTR(nombre_prueba, false, salidaEsperada);
1294             ROLLBACK;
1294         END PRECIOLINEA_A_VENTA;
1296
1296     PROCEDURE PRECIO_VENTA
1297     (nombre_prueba VARCHAR2, ID_Venta NUMBER, esperado number, salidaEsperada BOOLEAN) AS
1298         salida BOOLEAN := true;
1298     precio number;
1298     BEGIN
1300         SELECT precioTotal_Venta(ID_VENTA) INTO precio FROM dual;
1302         IF (precio <> esperado) THEN
1304             salida := false;
1304         END IF;
1306         PRINTR(nombre_prueba, salida, salidaEsperada);
1308         EXCEPTION
1310         WHEN OTHERS THEN
1312             PRINTR(nombre_prueba, false, salidaEsperada);
1312             ROLLBACK;
1312         END PRECIO_VENTA;
1314
1314     PROCEDURE precio_Factura
1315     (nombre_prueba VARCHAR2, ID_Venta NUMBER, esperado number, salidaEsperada BOOLEAN) AS
1316         salida BOOLEAN := true;
1316     precio number;
1316     BEGIN
1320         SELECT precioTotal_Factura(ID_VENTA) INTO precio FROM dual;
1322         IF (precio <> esperado) THEN
1324             salida := false;
1324         END IF;
1326         PRINTR(nombre_prueba, salida, salidaEsperada);
1328         EXCEPTION
1330         WHEN OTHERS THEN
1332             PRINTR(nombre_prueba, false, salidaEsperada);
1332             ROLLBACK;
1332         END precio_Factura;
1334
1334     PROCEDURE precio_Pedido
1335     (nombre_prueba VARCHAR2, ID_PEDIDO NUMBER, esperado number, salidaEsperada BOOLEAN) AS
1336         salida BOOLEAN := true;
1336     precio number;
1336     BEGIN

```

```

1340      SELECT precioTotal_Pedido(ID_PEDIDO) INTO precio FROM dual;
1342      IF (precio<>esperado) THEN
1344          salida := false;
1346      END IF;

      PRINTR(nombre_prueba, salida, salidaEsperada);

      EXCEPTION
1348      WHEN OTHERS THEN
          PRINTR(nombre_prueba, false, salidaEsperada);
1350      ROLLBACK;
      END precio_Pedido;

1352  PROCEDURE precio_Albaran
1354      (nombre_prueba VARCHAR2, ID_PEDIDO NUMBER, esperado number, salidaEsperada BOOLEAN) AS
1356      salida BOOLEAN := true;
1358      precio number;
      BEGIN
1360          SELECT precioTotal_Albaran(ID_PEDIDO) INTO precio FROM dual;
1362          IF (precio<>esperado) THEN
              salida := false;
          END IF;

          PRINTR(nombre_prueba, salida, salidaEsperada);

          EXCEPTION
1366          WHEN OTHERS THEN
              PRINTR(nombre_prueba, false, salidaEsperada);
              ROLLBACK;
1370      END precio_Albaran;

1372  END PRUEBAS_FUNCIONES;
/
1374  DROP SEQUENCE S_ID_Producto;
1376  DROP SEQUENCE S_ID-Traspaso;
1378  DROP SEQUENCE S_ID-Solicitud;
1380  DROP SEQUENCE S_ID-Pedido;
1382  DROP SEQUENCE S_ID-Albaran;
1384  DROP SEQUENCE S_ID-Factura;
1386  DROP SEQUENCE S_ID-Emplazamiento;
1388  DROP SEQUENCE S_ID-Venta;

1390  CREATE SEQUENCE S_ID-Emplazamiento START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1392  CREATE SEQUENCE S_ID-Venta START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1394  CREATE SEQUENCE S_ID-Pedido START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1396  CREATE SEQUENCE S_ID-Albaran START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1398  CREATE SEQUENCE S_ID-Producto START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1400  CREATE SEQUENCE S_ID-Traspaso START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1402  CREATE SEQUENCE S_ID-Solicitud START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1404  CREATE SEQUENCE S_ID-Factura START WITH 1 INCREMENT BY 1 MAXVALUE 9999;

1406  /* EXECUTE DE PRUEBAS */
1408  -- Inicializacion
1410  EXECUTE pruebas_albaran.inicializar();
1412  EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.inicializar();
      EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.inicializar();
      EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.inicializar();
      EXECUTE PRUEBAS_A_VENTA_PRODUCTO.inicializar();
      EXECUTE PRUEBAS_FACTURA.inicializar();
      EXECUTE pruebas_venta.inicializar();
      EXECUTE pruebas_pedido.inicializar();
      EXECUTE pruebas_proveedor.inicializar();
      EXECUTE PRUEBAS_STOCK.inicializar();
      EXECUTE pruebas_producto.inicializar();
      EXECUTE PRUEBAS_TRASPASO.inicializar();
      EXECUTE PRUEBAS_SOLICITUD_TRASPASO.inicializar();
      EXECUTE PRUEBAS_SOCIO.inicializar();
      EXECUTE pruebas_emplazamiento.inicializar();
      --SOCIO
1410  EXECUTE PRUEBAS_SOCIO.inicializar();
      EXECUTE PRUEBAS_SOCIO.insertar('Socio Insertar', 'Daniel', 'Gonzalez', '54148787G', 'Avenida del
      pepinillo', TO_DATE('10102015', 'DDMMYYYY'), 'dani@us.es', true);
1412  EXECUTE PRUEBAS_SOCIO.insertar('Socio Insertar', 'Jose Luis', 'Marmol Romero', '29613400A', 'Calle
      Virgen', TO_DATE('10101996', 'DDMMYYYY'), 'jlmr@us.es', true);

```

```

EXECUTE PRUEBAS_SOCIO.actualizar('Socio Actualizar', '54148787G', 'Avenida del pepinillo234', '
dani@us.es', true);
1414 EXECUTE PRUEBAS_SOCIO.eliminar('Socio Eliminar', '54148787G', true);

1416 --PROVEEDOR
EXECUTE PRUEBAS_PROVEEDOR.inicializar();
1418 EXECUTE PRUEBAS_PROVEEDOR.insertar('Proveedor Insertar', 'B54120214', 'Pimex', 954852320, 'pimex@pi.
es', true);
EXECUTE PRUEBAS_PROVEEDOR.insertar('Proveedor Insertar', 'B54129999', 'Pimex34', 954896666, '
pimex34@pi.es', true);
1420 EXECUTE PRUEBAS_PROVEEDOR.actualizar('Proveedor Actualizar', 'B54120214', 'Piexs', 111111111, 'a@pi.
es', true);
EXECUTE PRUEBAS_PROVEEDOR.eliminar('Proveedor Eliminar', 'B54120214', true);
1422

--PRODUCTO
1424 EXECUTE PRUEBAS_PRODUCTO.inicializar();
COMMIT WORK;
1426 EXECUTE PRUEBAS_PRODUCTO.insertar('Producto con IVA invalido','Chaleco de lana','por una gran
diseñadora','Jerseys',46.95,2,false);
EXECUTE PRUEBAS_PRODUCTO.insertar('Producto con categoria invalida','Chaleco de lana','por una gran
diseñadora','Ropa',46.95,0.21,false);
1428 EXECUTE PRUEBAS_PRODUCTO.insertar('Producto con precio invalido','Chaleco de lana','por una gran
diseñadora','Jerseys',-2,0.21,false);

1430 EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion','Zapatos cutres','unos zaparos cutres pero
calentitos','Calzado',12.5,0.21,true);
EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion','Cool shoes','Zapatos de cuero italiano','
Calzado',5.95,0.21,true);
1432 EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion','Sport Shoes','Zapatos de plastico','Calzado'
,6.95,0.21,true);
EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion','Brown Sandals','Sandalias de playa marrones',
'Calzado',7.95,0.21,true);
1434 EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion','Chupa de cuero','gran imitacion de la chupa
de Han Solo','Calzado',500,0.21,true);
EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion','Chaleco de lana','por una gran diseñadora','
Jerseys',46.95,0.21,true);
1436 EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion','Sombrero de paja','el gran sombrero de
Mugiwara','Accesorios',20,0.21,true);
EXECUTE PRUEBAS_PRODUCTO.actualizar('Producto actualizacion',S_ID_Producto.currval-2,'Unos zapatos
cutres pero calentitos, estan rotos',10.5,0.21,true);
1438 EXECUTE PRUEBAS_PRODUCTO.eliminar('Producto eliminar',S_ID_Producto.currval,true);

--EMPLAZAMIENTO
1440 EXECUTE PRUEBAS_EMPLAZAMIENTO.inicializar();
COMMIT WORK;
1442 EXECUTE PRUEBAS_EMPLAZAMIENTO.insertar('Emplazamiento con tipo invalido', 'Calle de la piruleta 25',
958632666,'Emplazamiento', false);

1444 EXECUTE PRUEBAS_EMPLAZAMIENTO.insertar('Emplazamiento insercion', 'Calle de la piruleta', 954147741,'
TIENDA', true);
EXECUTE PRUEBAS_EMPLAZAMIENTO.insertar('Emplazamiento insercion', 'Avenida Reina Mercedes',
958632147,'TIENDA', true);
1446 EXECUTE PRUEBAS_EMPLAZAMIENTO.insertar('Emplazamiento insercion', 'Calle de la piruleta 24',
958632147,'TIENDA', true);
EXECUTE PRUEBAS_EMPLAZAMIENTO.insertar('Emplazamiento insercion', 'Calle de la piruleta 25',
958632666,'ALMACEN', true);
1448 EXECUTE PRUEBAS_EMPLAZAMIENTO.actualizar('Emplazamiento actualizacion',S_ID_Emplazamiento.currval-2,'
Avenida del pepinillo2', 954147871, true);
EXECUTE PRUEBAS_EMPLAZAMIENTO.eliminar('Emplazamiento eliminar', S_ID_Emplazamiento.currval, true);
1450

--STOCK
EXECUTE PRUEBA_STOCK.inicializar();
1452 EXECUTE PRUEBA_STOCK.insertar('Stock insercion',S_ID_Emplazamiento.currval-2,S_ID_Producto.currval
-1,10,true);
EXECUTE PRUEBA_STOCK.insertar('Stock insercion',S_ID_Emplazamiento.currval-2,S_ID_Producto.currval
-2,20,true);
1454 EXECUTE PRUEBA_STOCK.insertar('Stock insercion',S_ID_Emplazamiento.currval-2,S_ID_Producto.currval
-3,30,true);
EXECUTE PRUEBA_STOCK.insertar('Stock insercion',S_ID_Emplazamiento.currval-1,S_ID_Producto.currval
-1,11,true);
1456 EXECUTE PRUEBA_STOCK.insertar('Stock insercion',S_ID_Emplazamiento.currval-1,S_ID_Producto.currval
-2,22,true);
EXECUTE PRUEBA_STOCK.insertar('Stock insercion',S_ID_Emplazamiento.currval-1,S_ID_Producto.currval
-3,33,true);
1458 EXECUTE PRUEBA_STOCK.actualizar('Stock actualizacion',S_ID_Emplazamiento.currval-2,S_ID_Producto.
currval -2,30,true);
EXECUTE PRUEBA_STOCK.eliminar('Stock eliminar',S_ID_Emplazamiento.currval-2,S_ID_Producto.currval-3,
true);
1460

```

```

1462 --VENTA
EXECUTE pruebas_venta.inicializar();
EXECUTE pruebas_venta.insertar('Venta insercion',sysdate,'29613400A',true);
1464 EXECUTE pruebas_venta.insertar('Venta insercion',sysdate,null,true);
EXECUTE pruebas_venta.insertar('Venta insercion',sysdate,null,true);
1466 EXECUTE pruebas_venta.eliminar('Venta eliminar',S_ID_venta.currval,true);
/*
1468 EXECUTE pruebas_venta.insertar('Venta insercion',0,sysdate,null,true);
EXECUTE PRUEBAS_A_VENTA_PRODUCTO.insertar('Venta-Producto insercion',s_id_venta.currval,s_id_producto
    .currval-1,28,10.5,0.21,0,true);
1470 INSERT INTO FACTURA VALUES(0,0,sysdate,'F',4,3);
*/
1472 --EXECUTE Pruebas_factura.insertar('Factura insercion',0,sysdate,'F',s_id_venta.currval,
    S_ID_Emplazamiento.currval-1,true);
--select S_ID_Emplazamiento.currval-1 from dual;
1474 --ASOCIACION_VENTA_PRODUCTO
EXECUTE PRUEBAS_A_VENTA_PRODUCTO.inicializar();
1476 EXECUTE PRUEBAS_A_VENTA_PRODUCTO.insertar('Venta-Producto insercion',s_id_venta.currval-1,
    s_id_producto.currval-2,3,10.5,0.21,true);
EXECUTE Pruebas_A_venta_producto.insertar('Venta-Producto insercion',s_id_venta.currval-1,
    s_id_producto.currval-1,2,46.95,0.21,true);
1478 EXECUTE PRUEBAS_A_VENTA_PRODUCTO.insertar('Venta-Producto insercion',s_id_venta.currval-2,
    s_id_producto.currval-2,3,10.5,0.21,true);
EXECUTE Pruebas_A_venta_producto.insertar('Venta-Producto insercion',s_id_venta.currval-2,
    s_id_producto.currval-1,2,46.95,0.21,true);
1480 EXECUTE PRUEBAS_A_VENTA_PRODUCTO.eliminar('Venta-Producto eliminar',s_id_venta.currval-2,
    s_id_producto.currval-1,true);

1482 --FACTURA
EXECUTE PRUEBAS_factura.inicializar();
1484 EXECUTE Pruebas_factura.insertar('Factura insercion',sysdate,'F',s_id_venta.currval-2,
    S_ID_Emplazamiento.currval-1,true);
EXECUTE Pruebas_factura.insertar('Factura insercion',sysdate,'F',s_id_venta.currval-1,
    S_ID_Emplazamiento.currval-2,true);
1486 EXECUTE Pruebas_factura.actualizar('Factura actualizar',s_id_factura.currval,'T',true);
--EXECUTE Pruebas_venta.descuento('Descuento factura-venta',s_id_venta.currval-2,125.4,'29613400A',
    true);
1488 --EXECUTE Pruebas_venta.descuento('Descuento factura-venta',s_id_venta.currval-1,125.4,null,true);
EXECUTE Prueba_stock.stock_correcto('Actualizacion de stock tras venta',s_id_emplazamiento.currval-1,
    s_id_producto.currval-2,22,3,true);
1490 EXECUTE Prueba_stock.stock_correcto('Actualizacion de stock tras venta',s_id_emplazamiento.currval-1,
    s_id_producto.currval-2,21,3,false);

1492 --PEDIDO
EXECUTE PRUEBAS_PEDIDO.inicializar();
1494 EXECUTE PRUEBAS_PEDIDO.insertar('Pedido insercion',sysdate,S_ID_Emplazamiento.currval-1,'B54129999',
    true);
EXECUTE PRUEBAS_PEDIDO.insertar('Pedido insercion',sysdate,S_ID_Emplazamiento.currval-2,'B54129999',
    true);
1496 EXECUTE PRUEBAS_PEDIDO.insertar('Pedido insercion',sysdate,S_ID_Emplazamiento.currval-2,'B54129999',
    true);
EXECUTE PRUEBAS_PEDIDO.eliminar('Pedido eliminar',S_ID_Pedido.currval,true);
1498 --PEDIDO_PRODUCTO
1500 EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.inicializar();
1502 EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.insertar('PEDIDO_PRODUCTO insercion',S_ID_Pedido.currval-1,
    s_id_producto.currval-1,21,10.5,0.21,true);
EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.insertar('PEDIDO_PRODUCTO insercion',S_ID_Pedido.currval-2,
    s_id_producto.currval-1,22,46.95,0.21,true);
1504 EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.insertar('PEDIDO_PRODUCTO insercion',S_ID_Pedido.currval-1,
    s_id_producto.currval-2,22,46.95,0.21,true);
EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.eliminar('PEDIDO_PRODUCTO eliminar',S_ID_Pedido.currval-1,
    s_id_producto.currval-2,true);
1506 --ALBARAN
1508 EXECUTE PRUEBAS_ALBARAN.inicializar();
EXECUTE PRUEBAS_ALBARAN.insertar('Albaran insercion',sysdate,S_ID_Pedido.currval-1,true);
1510 EXECUTE PRUEBAS_STOCK.stock_correcto_p('Actualizacion de stock tras comprar',s_id_emplazamiento.
    currval-2,s_id_producto.currval-1,8,21,true);
EXECUTE PRUEBAS_STOCK.stock_correcto_p('Actualizacion de stock tras comprar',s_id_emplazamiento.
    currval-2,s_id_producto.currval-1,8,15,false);
1512 EXECUTE PRUEBAS_ALBARAN.insertar('Albaran insercion',sysdate,S_ID_Pedido.currval-2,true);
EXECUTE PRUEBAS_ALBARAN.eliminar('Albaran eliminacion',S_ID_Albaran.currval,true);
1514 --SOLICITUD_TRASPASO
1516 EXECUTE PRUEBAS_SOLICITUD_TRASPASO.inicializar();

```



```

EXECUTE PRUEBAS_SOLICITUD_TRASPASO.insertar('Solicitud traspaso Insertar', sysdate,
    S_ID_Emplazamiento.currval-1, S_ID_Emplazamiento.currval-2, true);
1518 EXECUTE PRUEBAS_SOLICITUD_TRASPASO.eliminar('Solicitud traspaso Eliminar', S_ID_Solicitud.currval,
    true);
EXECUTE PRUEBAS_SOLICITUD_TRASPASO.insertar('Solicitud traspaso Insertar', sysdate,
    S_ID_Emplazamiento.currval-1, S_ID_Emplazamiento.currval-2, true);
1520 --ASOC_PRODUCTO_SOLICITUD
EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.inicializar();
1522 EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.insertar('A-Producto-Solicitud Insertar', S_ID_Producto.currval
    -2, S_ID_Solicitud.currval, 3, true);
EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.eliminar('A-Producto-Solicitud Eliminar', S_ID_Solicitud.currval
    , S_ID_Producto.currval-2, true);
1524 EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.insertar('A-Producto-Solicitud Insertar', S_ID_Producto.currval
    -2, S_ID_Solicitud.currval, 3, true);
COMMIT WORK;
1526 EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.insertar('Prueba de Trigger solicitud_stock_minimo',
    S_ID_Producto.currval-1, S_ID_Solicitud.currval, 28, false);

1528 --TRASPASO
EXECUTE PRUEBAS_TRASPASO.inicializar();
1530 EXECUTE PRUEBAS_TRASPASO.insertar('Traspaso insercion',sysdate,S_ID_Emplazamiento.currval-1,
    S_ID_Emplazamiento.currval-2,true);
EXECUTE PRUEBAS_TRASPASO.insertar('Traspaso insercion',sysdate,S_ID_Emplazamiento.currval-1,
    S_ID_Emplazamiento.currval-3,true);
1532 EXECUTE PRUEBAS_TRASPASO.eliminar('Traspaso eliminar',S_ID_traspaso.currval,true);

1534 --ASOC_PRODUCTO_TRASPASO
EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.inicializar();
1536 EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.insertar('A-Producto-Traspaso Insertar', S_ID_Producto.currval-1,
    S_ID_traspaso.currval-1, 4, true);
EXECUTE PRUEBA_STOCK.STOCK_CORRECTO_T('Actualizacion de stock tras traspaso',s_ID_emplazamiento.
    currval-1, s_ID_emplazamiento.currval-2,33,29,4,s_ID_producto.currval-1,true);
1538 EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.eliminar('A-Producto-Traspaso Eliminar', S_ID_traspaso.currval-1,
    S_ID_Producto.currval-1, true);
--Pruebas de funciones
1540

1542 EXECUTE PRUEBAS_FUNCIONES.precioLinea_A_Pedido('Funcion precio linea aso-pedido',9,2,220.5,true);
EXECUTE PRUEBAS_FUNCIONES.precioLinea_A_Venta('Funcion precio linea aso-venta',9,2,93.9,true);
1544 EXECUTE PRUEBAS_FUNCIONES.precio_Venta('Funcion calculo precio total venta',1,31.5,true);
EXECUTE PRUEBAS_FUNCIONES.precio_Factura('Funcion calculo precio total factura con socio',1,29.93,
    true);
1546 EXECUTE PRUEBAS_FUNCIONES.precio_Factura('Funcion calculo precio total factura sin socio',2,125.4,
    true);
EXECUTE PRUEBAS_FUNCIONES.precio_Pedido('Funcion calculo precio total pedido',2,220.5,true);
1548 EXECUTE PRUEBAS_FUNCIONES.precio_Albaran('Funcion calculo precio total albaran',2,220.5,true);
-- Tiene que dar 220.5
1550 --SELECT precioLinea_Aso_Pedido(9,2) FROM DUAL;

1552 -- Tiene que dar 220.5
--select preciototal_pedido(2) from dual;
1554
-- Tiene que dar 220.5
1556 --select preciototal_albaran(2) from dual;

1558 -- Tiene que dar 31.5
--select preciototal_venta(2) from dual;
1560
-- Tiene que dar 31.5*0.95 = 29.93
1562 --select preciototal_factura(1) from dual;

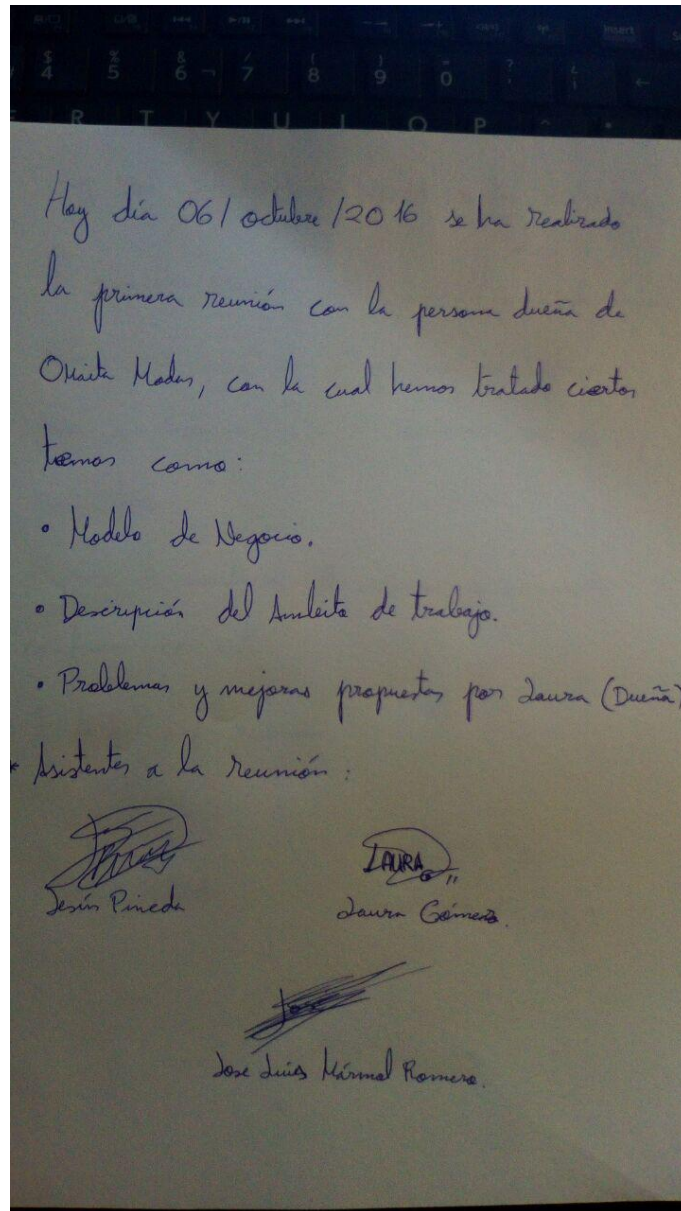
1564 --Tienda que dar 93.9
--SELECT precioLinea_Aso_Venta(9,2) FROM DUAL;
1566
-- Select que muestra los productos ofrecidos y disponibles
1568 --SELECT nombre,id_emplazamiento from producto inner join stock on stock.ID_PRODUCTO = producto.
    id_producto;

```

sql/pruebas.sql

## 10. Apéndice

### 10.1. Actas de reunión



Hoy Día 20/ Octubre /2016 se ha realizado la segunda reunión con Laura (Dona de Ornato Rodas) para enseñarle el proceso del trabajo para que todo estuviese de acuerdo con sus expectativas.

Después de mostrarle el avance se encuentra conforme con este.

\* Asistentes a la reunión:

Jesús Pineda

Laura Gómez