

UNIVERSIDAD DE SEVILLA

IISSI

PROYECTO

---

# Modas Omaita

---

*Autores:*

Daniel González Corzo

Jesús Pineda Márquez

José Luis Mármol Romero

Roberto Hueso Gómez

14 de mayo 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 . . . . .	
4.2. Historias de usuario . . . . .	
<b>5. Catálogo de requisitos</b>	
5.1. Requisitos de información . . . . .	
5.2. Requisitos funcionales . . . . .	
5.3. Requisitos no funcionales . . . . .	
5.4. Reglas de negocio . . . . .	
<b>6. Pruebas de aceptación</b>	
<b>7. Modelo Conceptual</b>	
7.1. Diagramas de clases UML . . . . .	
7.2. Escenarios de prueba . . . . .	
<b>8. Matrices de trazabilidad</b>	
8.1. Pruebas de aceptación frente a requisitos . . . . .	
8.2. Pruebas de aceptación frente a escenarios de prueba . . . . .	
8.3. Tipos de UML frente a Requisitos . . . . .	
<b>9. Modelos relacionales</b>	
9.0.1. 3FN Venta . . . . .	
9.0.2. 3FN Traspaso - Pedido . . . . .	

## **10.Código SQL de la base de datos**

- 10.1. Tablas . . . . .
- 10.2. Funciones y Procedures . . . . .
- 10.3. Triggers . . . . .
- 10.4. Pruebas . . . . .

## **11.Apéndice**

- 11.1. 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

### 3.1. Proceso de venta

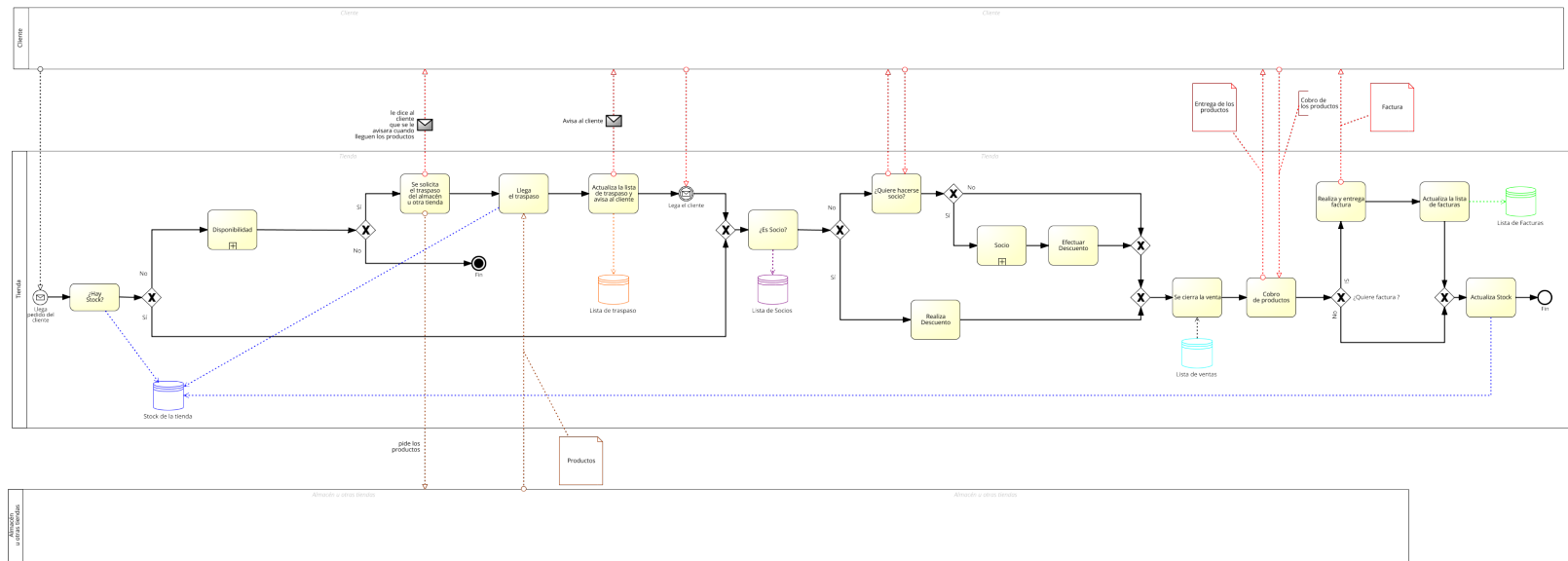
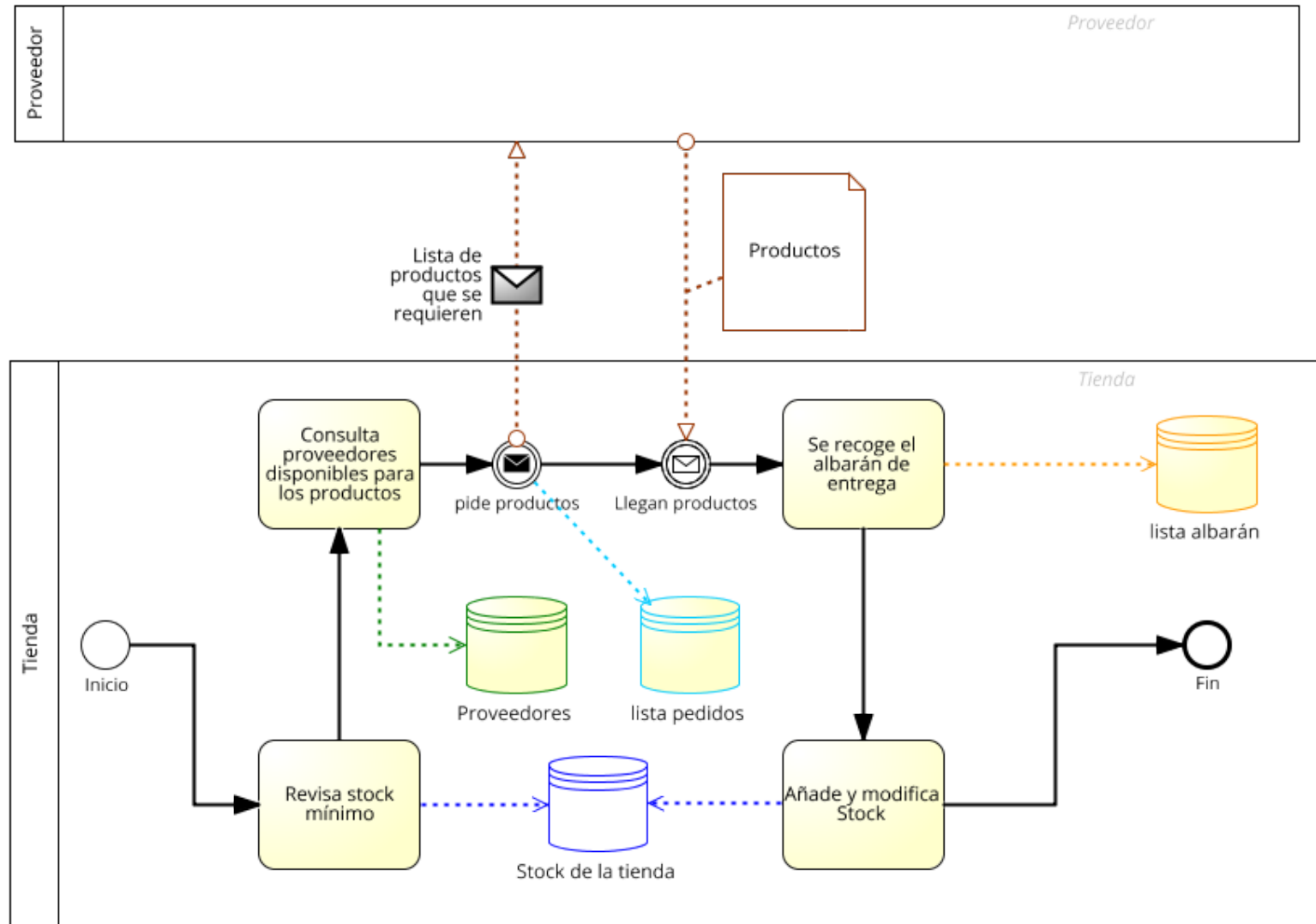


Figura 1: Proceso de venta



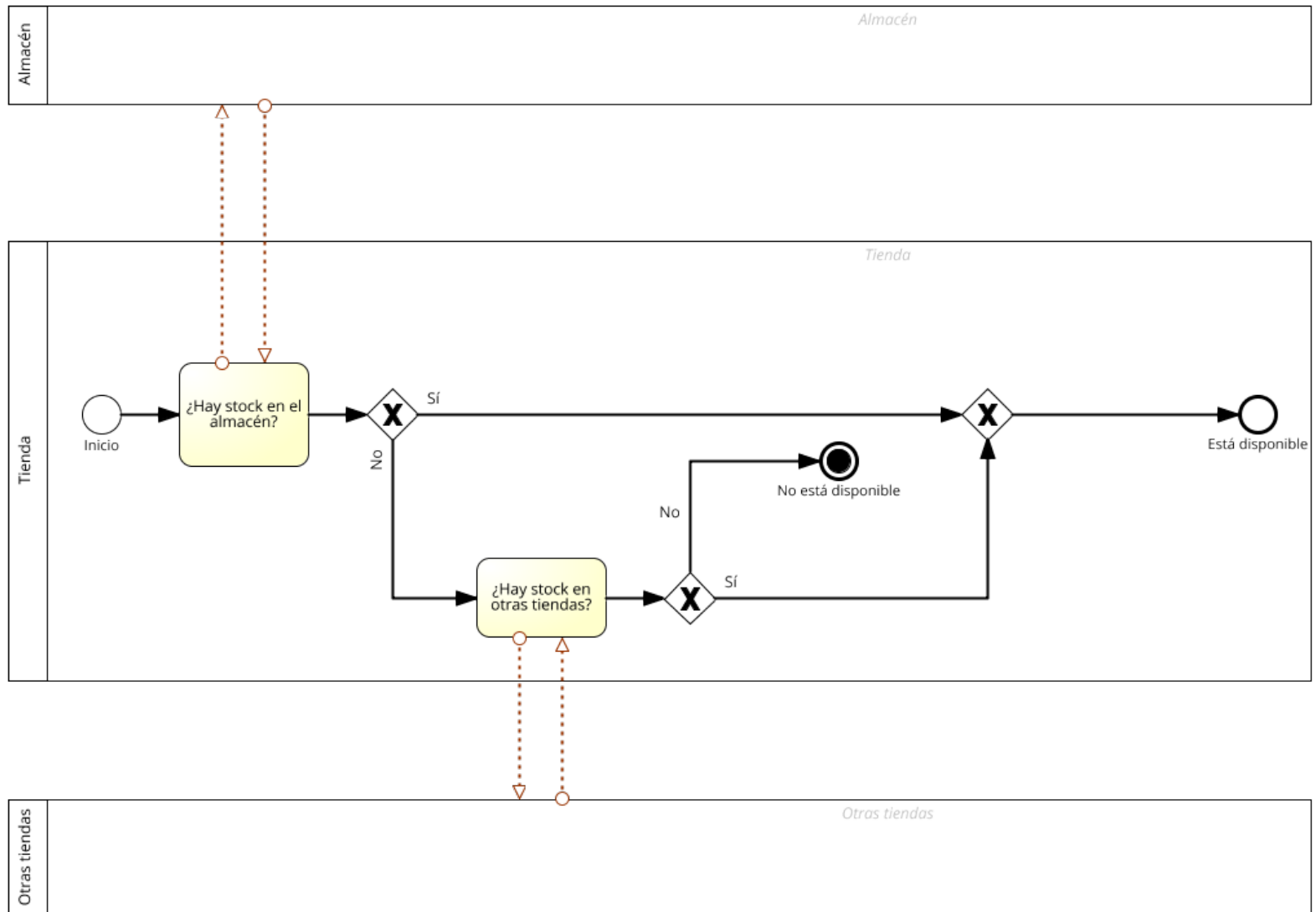
### 3.2. Proceso de compra al proveedor







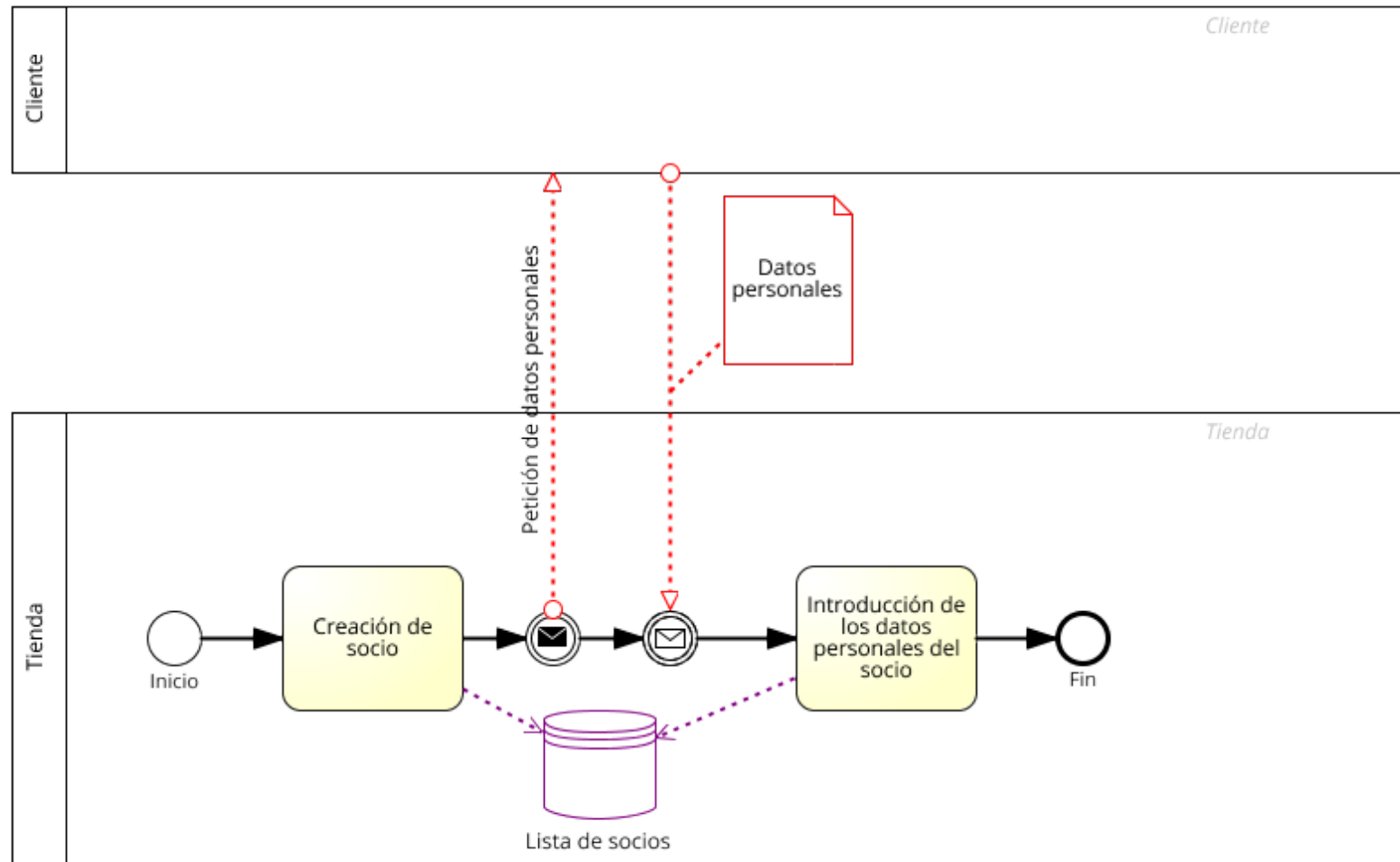
### 3.3. Proceso de disponibilidad



.png



### 3.4. Proceso de creación de socios



.png

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.

## 4.2. Historias de usuario

ID	Historia
HU-1	<ul style="list-style-type: none"><li>▪ <b>Como</b> trabajador</li><li>▪ <b>Quiero</b> conocer las ventas diarias del negocio</li><li>▪ <b>Para</b> hacer ajustes a las ofertas del mismo</li></ul>
HU-2	<ul style="list-style-type: none"><li>▪ <b>Como</b> propietario de la cadena</li><li>▪ <b>Quiero</b> conocer el stock disponible en cada negocio</li><li>▪ <b>Para</b> organizar los pedidos a proveedores</li></ul>
HU-3	<ul style="list-style-type: none"><li>▪ <b>Como</b> propietario de la cadena</li><li>▪ <b>Quiero</b> conocer las ganancias mensuales</li><li>▪ <b>Para</b> tener un balance mensual</li></ul>
HU-4	<ul style="list-style-type: none"><li>▪ <b>Como</b> cliente</li><li>▪ <b>Quiero</b> ver una lista de los productos ofrecidos y disponibles</li><li>▪ <b>Para</b> realizar mis compras</li></ul>

## 5. Catálogo de requisitos

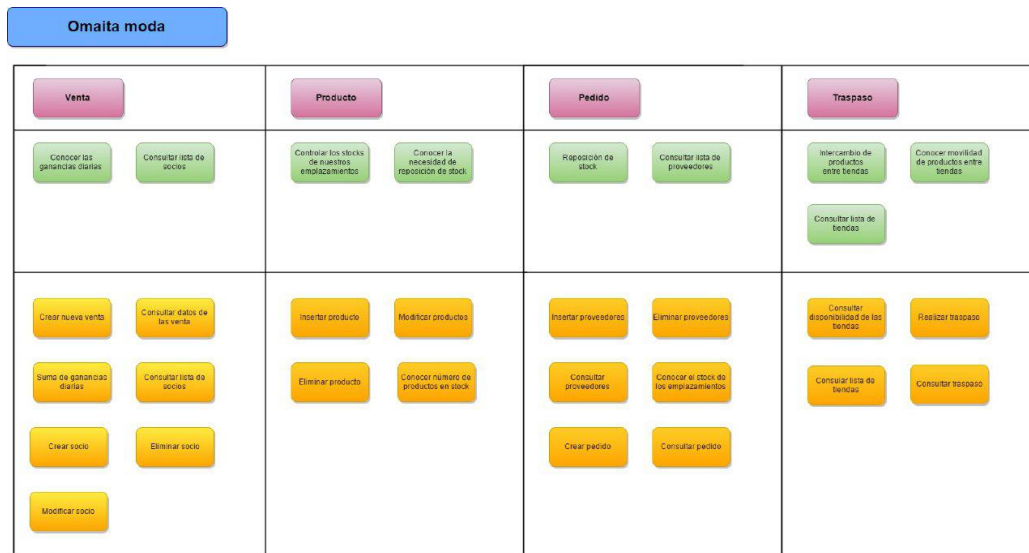


Figura 5: Mapa de requisitos

### 5.1. 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
  - Precio total de la venta.
  - 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.

---

## 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:
  - Precio final de la venta.
  - 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.

---

## 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.

---

## 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.
- 

#### **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
- 

#### **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
- 

#### **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

---

#### 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.
  - Precio total del pedido.
  - Una asociación que controla la cantidad de cada producto del pedido, el precio y el IVA.
- **Para** conocer los datos de los pedidos realizados.

---

#### 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.

---

#### 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.
  - Precio total.
- **Para** tener un control de los pedidos recibidos los cuales son controlados por los albaranes.

---

#### 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.

## 5.2. Requisitos funcionales

---

#### RF-01 - Crear ventas

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

---

#### 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

---

**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.

---

**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.

---

**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.

---

**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.

---

**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.
- 

#### **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.
- 

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

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

#### **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.
- 

#### **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.

---

**RF-12 - Crear traspasos**

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

---

**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

---

**RF-14 - Consultar traspasos**

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

---

**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.

---

**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.
- 

#### **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.
- 

#### **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.
- 

### **5.3. 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.

### 5.4. 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
- 

#### 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.
- 

#### 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



---

**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

---

**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

## 6. Pruebas de aceptación

**PA-01 Ventas:**

- Se realiza una venta y con una consulta vemos que la fecha y el precio total es el esperado.

**PA-02 Facturas y Devoluciones:**

- Se realiza una factura y con una consulta vemos que el precio final de la venta, la fecha, el número de la factura y el IVA es el esperado.
- Un cliente devuelve un producto dentro del plazo de 30 días y por tanto se devuelve el dinero al cliente, realizamos una consulta sobre la factura del producto devuelto y vemos la factura original con el campo devuelto como True.
- Tras la situación anterior el trabajador crea una nueva factura con todos los productos originales de la compra menos con el que se ha devuelto, entonces se realiza una consulta a la factura y vemos como se ha eliminado correctamente el producto devuelto y no se ha contabilizado en el precio de la factura. (Situación que solo se da si el producto devuelto pertenece a una factura de más productos)

- Un cliente desea devolver un producto tras 30 días de su compra, los artículos no pueden ser devueltos debido a que ha pasado más de 30 días desde su compra.

#### **PA-03 Socios:**

- Se añade un nuevo socio, se actualiza la lista de socios y este aparece con todos los siguientes datos: nombre, apellidos, DNI, dirección, fecha de nacimiento y email.
- Se elimina un socio, se actualiza la lista de socios y en ella no aparece el socio.
- Se modifica los datos de un socio, se actualiza la lista de socios y este aparece con sus datos modificados.
- Se intenta añadir un nuevo socio, pero le faltan datos, el sistema no permite añadirlo.
- Se intenta añadir un nuevo socio, pero ya está en la base de datos y el sistema no nos lo permite.

#### **PA-04 Productos:**

- Se añade un nuevo producto, se actualiza la lista de productos y este aparece con todos los siguientes datos: nombre, descripción, categoría, precio del producto e IVA.
- Se elimina un producto, se actualiza la lista de productos y en ella no aparece el producto eliminado.
- Se modifica los datos de un producto, se actualiza la lista de productos y este aparece con sus datos modificados.
- Se intenta añadir un producto nuevo, pero le faltan datos, el sistema no permite añadirlo.
- Se desea añadir un producto que ya está en la base de datos y el sistema no nos lo permite.

#### **PA-05 Stock:**

- Si el stock de un producto es de 10 y un cliente quiere comprar 11, el stock es superado y el sistema no permite esa compra.
- Si el stock de un producto es de 10 y un cliente quiere comprar 3, se le permite la venta y se modifica el stock en la base de datos.
- Se realiza un traspaso entre 2 emplazamientos, hacemos 2 consultas y comprobamos que el stock de cada emplazamiento es el correcto tras la transacción.
- El stock de un producto es de 6 y un cliente realiza una compra de ese producto, entonces se muestra un aviso de que el stock de ese producto ha alcanzado el stock mínimo.
- Consultamos el stock de la tienda y después recibimos el albarán del pedido que habíamos realizado, consultamos de nuevo el stock y vemos que se ha actualizado correctamente.

#### **PA-06 Proveedores:**

- Se registra un proveedor nuevo, se actualiza la lista de proveedores y aparece el proveedor con los siguientes datos: nombre, CIF, número de teléfono y email.
- Se modifican los datos de un proveedor, se actualiza la lista de proveedores y sale el proveedor con los datos modificados.
- Se elimina un proveedor, al actualizar la lista de los proveedores, el proveedor eliminado ya no aparece.
- Se intenta añadir un proveedor que ya está en la base de datos y el sistema no lo permite.

#### **PA-07 Pedidos y albaranes:**

- El almacén ha llegado a stock mínimo de algunos de sus productos y se realiza un pedido al proveedor o proveedores, tras recibir los productos del proveedor hacemos una consulta para ver el albarán está en orden y que los productos del pedido son los que se solicitaron.
- Se realiza una consulta sobre el pedido que se había realizado y vemos que se muestran los siguientes datos: fecha en la que se realizó, precio total y emplazamiento que realizó el pedido.

- Se intenta realizar un pedido de 15 unidades y salta un mensaje de error, ya que el pedido tiene que ser de 20 unidades mínimo.
- Se crea un albarán con los datos recibidos del albarán de entrega y hacemos una consulta para ver que se muestra correctamente.

#### **PA-08 Traspasos y solicitudes:**

- Falta stock de un producto y se solicita a otro emplazamiento, el otro emplazamiento realiza el traspaso y se reciben los productos. Hacemos una consulta para la solicitud y otra consulta para traspaso y vemos que los productos son correctos, además de que aparecen las respectivas fechas y los emplazamientos de la relación.
- Se intenta solicitar un traspaso de 3 unidades de un producto a una tienda, pero la tienda a la que se solicita el traspaso solo posee 6 unidades del producto por lo que salta un mensaje de que no se puede solicitar el traspaso a esta tienda.

#### **PA-09 Emplazamientos:**

- Se añade una nueva tienda, se realiza una consulta y vemos que se ha añadido correctamente.
- Se cambia la localización del almacén, se realiza una consulta y vemos que se ha cambiado correctamente.
- Se elimina una tienda, se realiza una consulta y vemos que se ha eliminado correctamente y ya no aparece en la lista de emplazamientos.
- Se realiza una consulta sobre el stock de una tienda y este se muestra correctamente.

#### **PA-10 Descuentos:**

- Un socio va a gastar un total de 100 euros en nuestros productos, al ser socio se le aplica un descuento del 5 %, por tanto, se le rebajan 5 euros en la factura.
- Un cliente va a gastar un total de 50 euros, al no ser socio no se le aplica el descuento y tiene que pagar los 50 euros completos.

**PA-11 Tamaño mínimo pedidos:**

- Se desea realizar un pedido de 10 productos y de cada producto se requieren 20 unidades, por tanto, se puede hacer el pedido.
- Se desea realizar el pedido de un producto con una cantidad equivalente a 15, este pedido no se podría llevar a cabo debido a que no llega a las 20 unidades necesarias.

## 7. Modelo Conceptual

### 7.1. Diagramas de clases UML

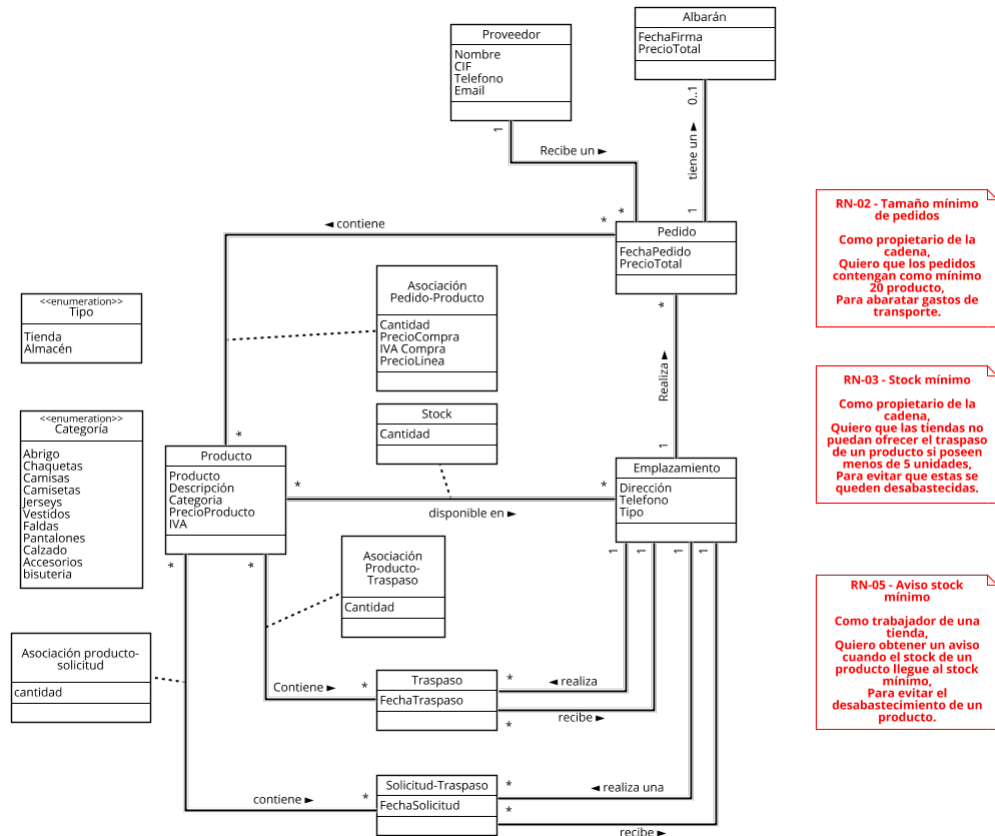


Figura 6: UML Traspaso / Pedido



## 7.2. Escenarios de prueba

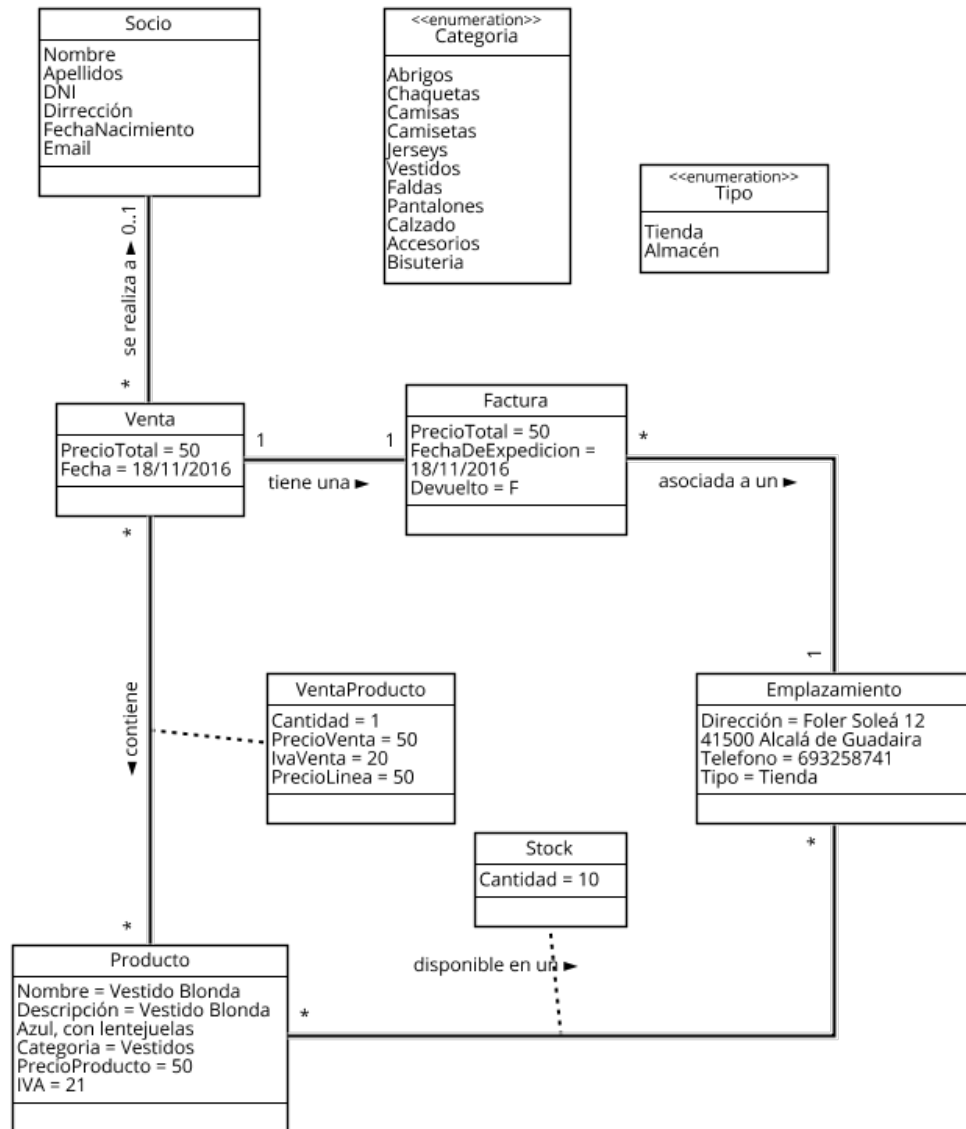


Figura 8: UML Venta NO Socio



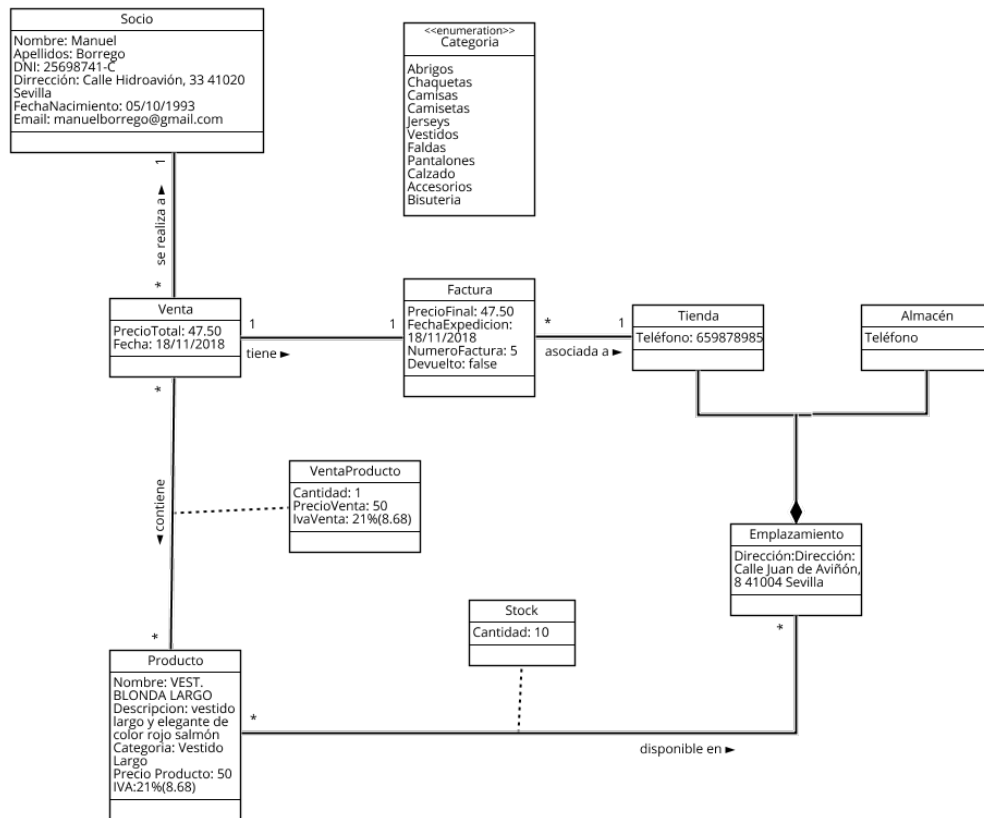


Figura 9: UML Venta Socio

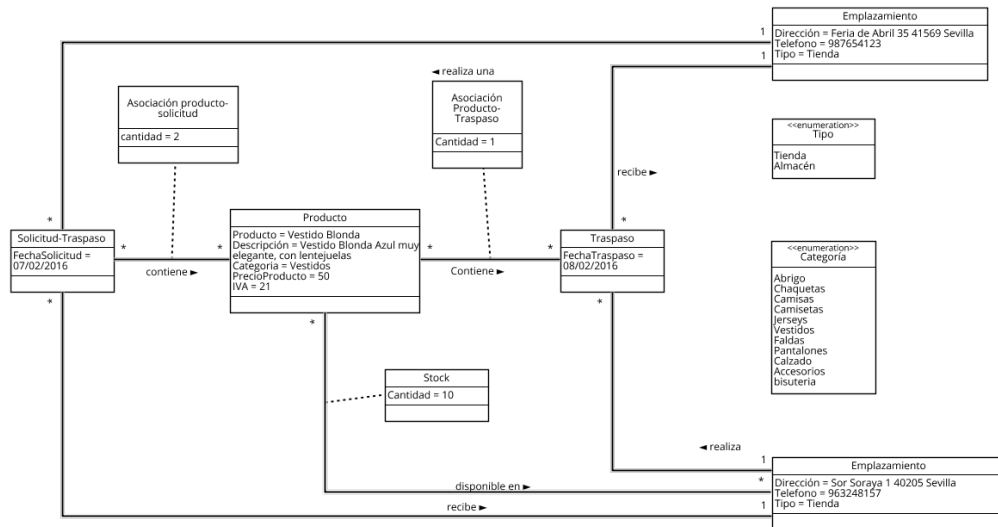


Figura 10: UML Traspaso Tienda - Tienda

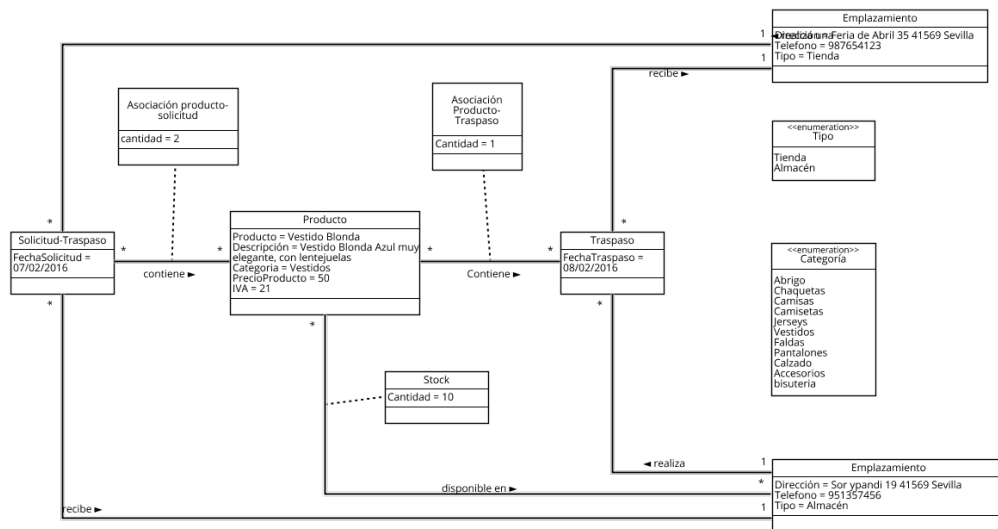


Figura 11: UML Traspaso Tienda - Almacén

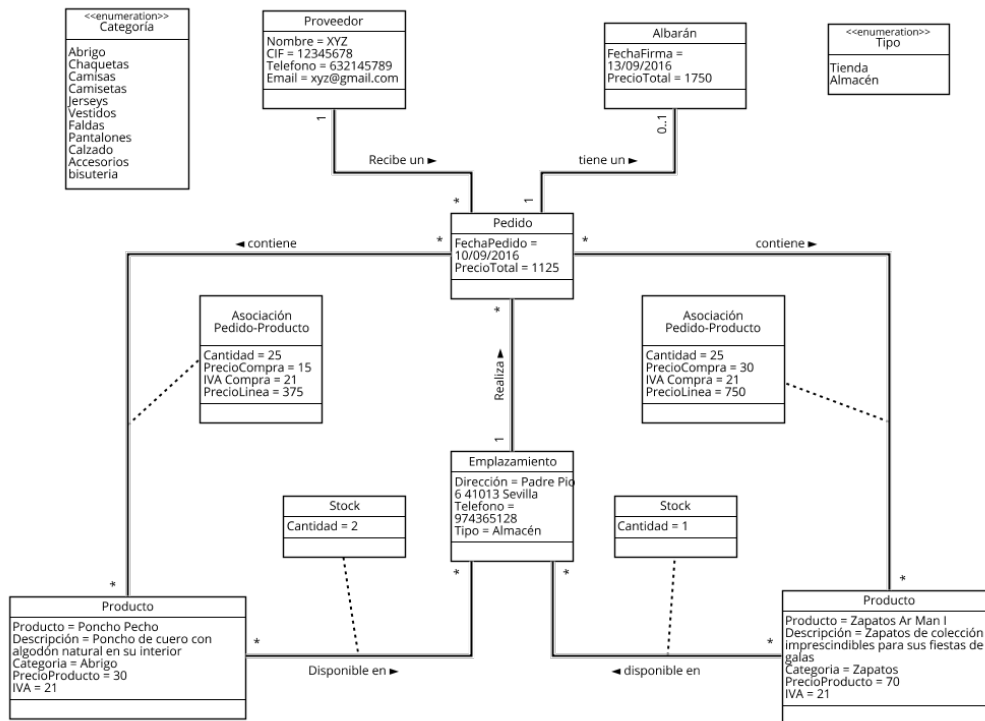


Figura 12: UML Pedido Tienda - Proveedor

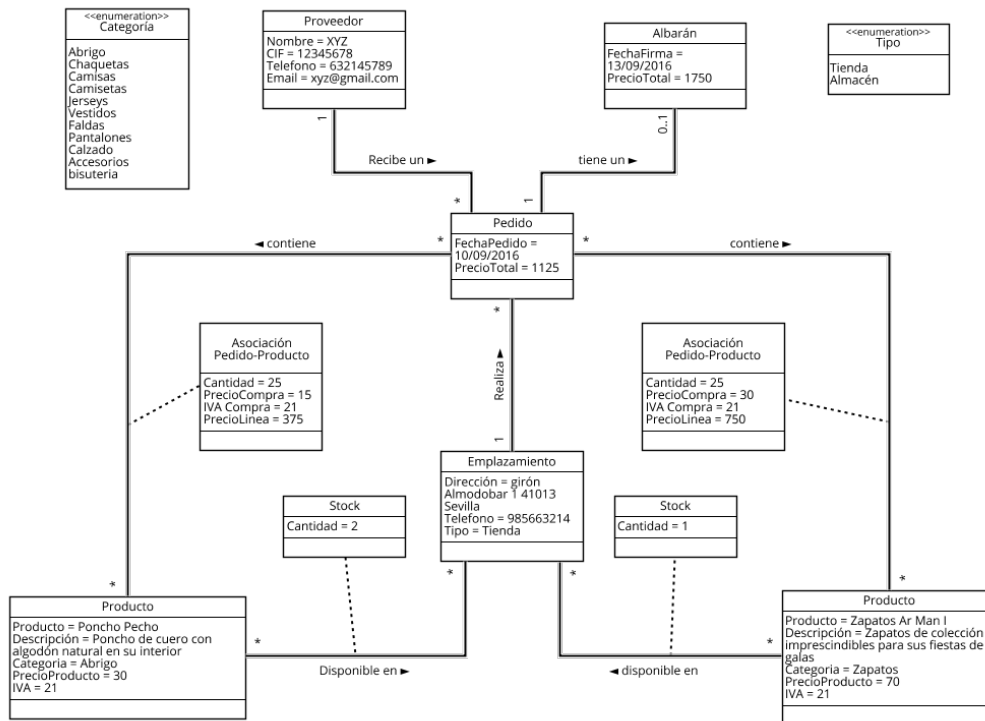


Figura 13: UML Pedido Almacén - Proveedor

## 8. Matrices de trazabilidad

### 8.1. Pruebas de aceptación frente a requisitos

Cuadro 1: Requisitos funcionales

PA-01	x																	
PA-02																	x	
PA-03			x	x	x													
PA-04						x												
PA-05		x					x			x			x					
PA-06								x										
PA-07									x	x								x
PA-08											x	x		x				
PA-09							x									x	x	
PA-10																		
PA-11																		
	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18

Cuadro 2: Requisitos de informacion

PA-01	x											
PA-02		x										
PA-03			x									
PA-04				x								
PA-05					x							
PA-06							x					
PA-07								x		x		
PA-08									x		x	
PA-09							x					
PA-10												
PA-11												
	01	02	03	04	05	06	07	08	09	10	11	

Cuadro 3: Reglas de negocio

<b>PA-01</b>					x
<b>PA-02</b>	x			x	
<b>PA-03</b>	x				
<b>PA-04</b>		x		x	
<b>PA-05</b>			x		x
<b>PA-06</b>		x			
<b>PA-07</b>		x			x
<b>PA-08</b>			x		
<b>PA-09</b>			x	x	
<b>PA-10</b>	x				
<b>PA-11</b>		x			
	<b>01</b>	<b>02</b>	<b>03</b>	<b>04</b>	<b>05</b>

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

Cuadro 4: Escenarios

<b>Escenario 01</b>				X		X	X		X		X
<b>Escenario 02</b>				X		X	X		X		X
<b>Escenario 03</b>				X	X			X	X		
<b>Escenario 04</b>				X	X			X	X		
<b>Escenario 05</b>	X	X		X	X						
<b>Escenario 06</b>	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>

## 8.3. Tipos de UML frente a Requisitos

Cuadro 5: 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 6: Reglas de negocio

Venta	x			x	x
Traspaso - Pedido		x	x		x
	01	02	03	04	05

Cuadro 7: Requisitos Funcionales

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

## 9. Modelos relacionales

### 9.0.1. 3FN Venta

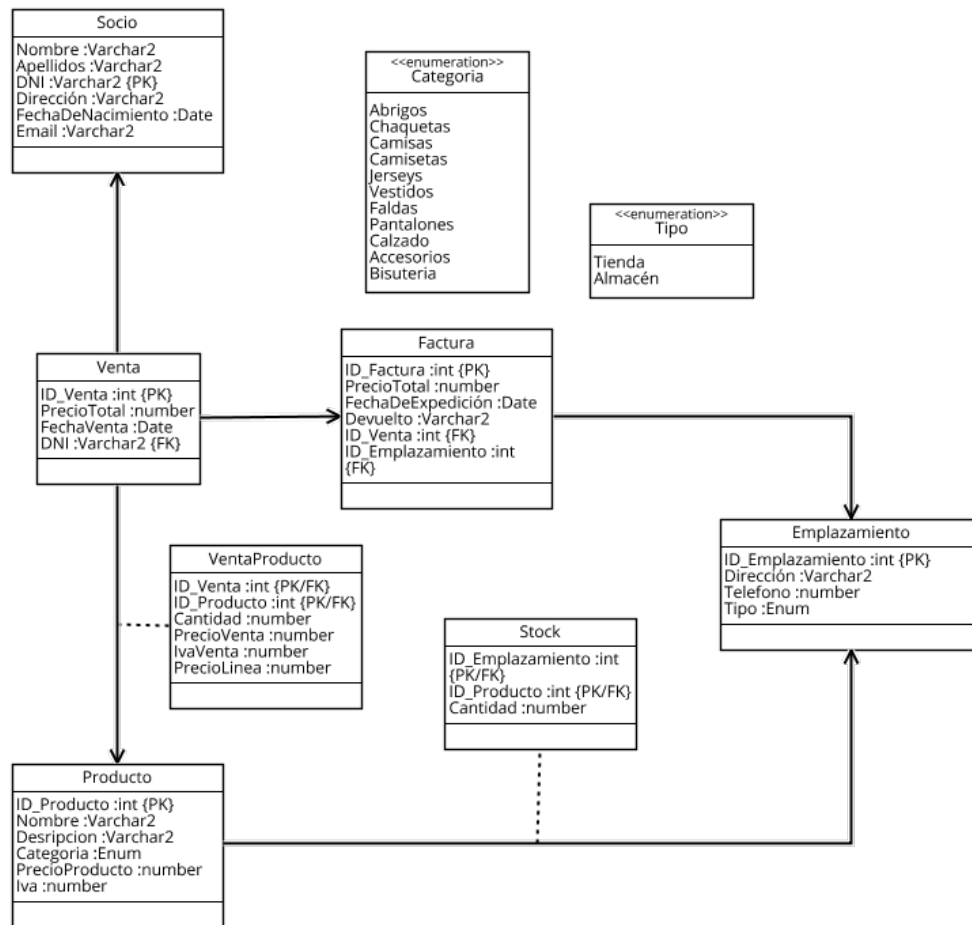


Figura 14: 3FN Venta



### 9.0.2. 3FN Traspaso - Pedido

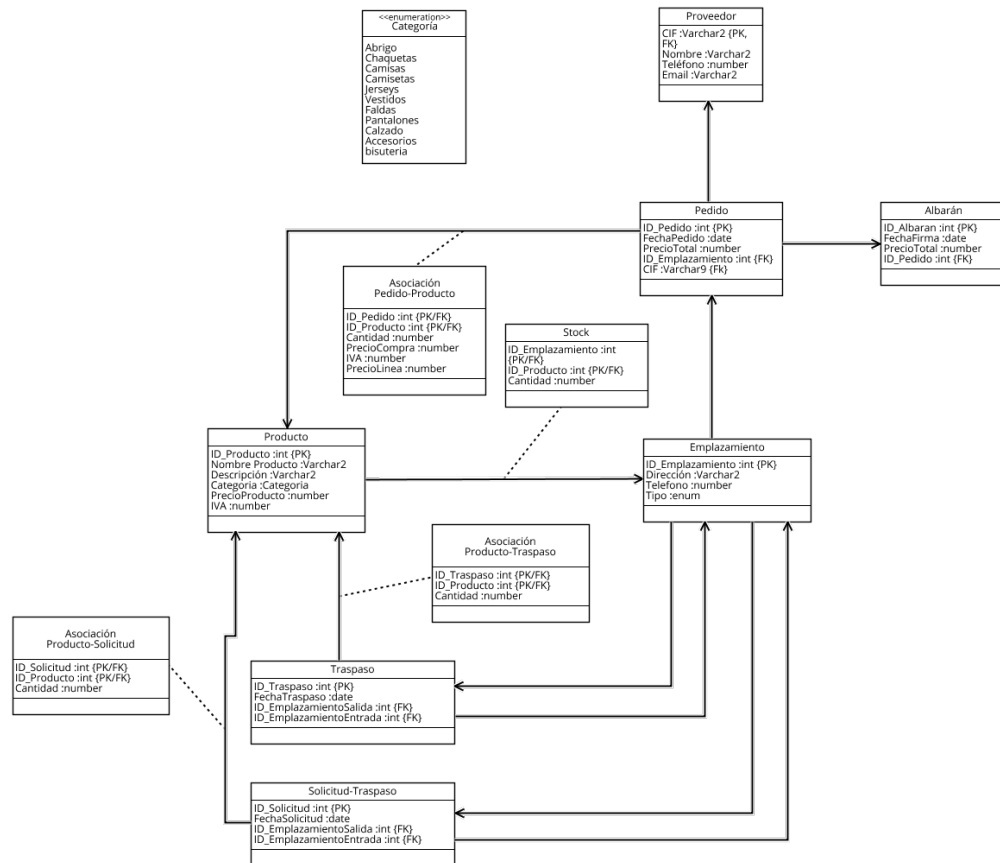


Figura 15: 3FN Traspaso - Pedido

## 10. Código SQL de la base de datos

## 10.1. Tablas

```
1 DROP TABLE ALBARAN;
2 DROP TABLE ASOCIACION_PRODUCTO_TRASPASO;
3 DROP TABLE ASOCIACION_PRODUCTO_SOLICITUD;
4 DROP TABLE ASOCIACION_PEDIDO_PRODUCTO;
5 DROP TABLE ASOCIACION_VENTA_PRODUCTO;
6 DROP TABLE FACTURA;
7 DROP TABLE VENTA;
```

```

9 DROP TABLE PEDIDO;
DROP TABLE PROVEEDOR;
DROP TABLE STOCK;
11 DROP TABLE PRODUCTO;
DROP TABLE TRASPASO;
13 DROP TABLE SOLICITUD_TRASPASO;
DROP TABLE SOCIO;
15 DROP TABLE EMPLAZAMIENTO;

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

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

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

41
CREATE TABLE FACTURA (
43     ID_FACTURA int PRIMARY KEY,
    PrecioTotal Number,
45     FechaDeExpedicion DATE DEFAULT SYSDATE,
    Devuelto VARCHAR2(1) CHECK( Devuelto IN('F','T')),
47     ID_Venta int,
    ID_Emplazamiento int,
49     FOREIGN KEY (ID_Venta) REFERENCES VENTA,
    FOREIGN KEY (ID_Emplazamiento) REFERENCES Emplazamiento
51 );

53 CREATE TABLE PROVEEDOR (
    CIF VARCHAR2(9) PRIMARY KEY,
55     Nombre VARCHAR2(75) NOT NULL,
    Telefono NUMBER(9) NOT NULL,
57     Email VARCHAR2(50) NOT NULL
);

59
CREATE TABLE PEDIDO(
61     ID_Pedido int PRIMARY KEY,
    FechaPedido DATE NOT NULL,
63     PrecioTotal NUMBER NOT NULL,
    ID_Emplazamiento int,
65     CIF VARCHAR2(9),
    FOREIGN KEY (CIF) REFERENCES PROVEEDOR,
67     FOREIGN KEY (ID_Emplazamiento) REFERENCES EMPLAZAMIENTO
);

69
CREATE TABLE ALBARAN (
71     ID_Albaran int NOT NULL,
    FechaFirma DATE NOT NULL,
73     PrecioTotal NUMBER NOT NULL,
    ID_Pedido INT PRIMARY KEY,
75     FOREIGN KEY (ID_Pedido) REFERENCES PEDIDO);

77
CREATE TABLE PRODUCTO(
79     ID_Producto int PRIMARY KEY,
    Nombre VARCHAR2(50) NOT NULL,
81     Descripcion VARCHAR2(300) NOT NULL,
    Categoria VARCHAR2(20) CHECK ( Categoria IN ('Abrigo','Chaquetas','Camisas'
83     , 'Camisetas','Jerseys','Vestidos','Faldas'
    , 'Pantalones','Calzado','Accesorios','Bisuteria')),

```

```

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

```

```

163 DROP SEQUENCE S_ID_Producto;
165 DROP SEQUENCE S_ID_Traspaso;
165 DROP SEQUENCE S_ID_Solicitud;
165 DROP SEQUENCE S_ID_Pedido;
167 DROP SEQUENCE S_ID_Albaran;
167 DROP SEQUENCE S_ID_Factura;
169 DROP SEQUENCE S_ID_Emplazamiento;
169 DROP SEQUENCE S_ID_Venta;

171

173
175 CREATE SEQUENCE S_ID_Emplazamiento START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
175 CREATE SEQUENCE S_ID_Venta START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
177 CREATE SEQUENCE S_ID_Pedido START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
177 CREATE SEQUENCE S_ID_Albaran START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
179 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;
181 CREATE SEQUENCE S_ID_Factura START WITH 1 INCREMENT BY 1 MAXVALUE 9999;

183 CREATE OR REPLACE TRIGGER crea_ID_Pedido
185 BEFORE INSERT ON Pedido
185 FOR EACH ROW
187 BEGIN
187 SELECT S_ID_Pedido.NEXTVAL INTO:NEW.ID_Pedido FROM DUAL;
189 END crea_ID_Pedido;

189 /

191
193 CREATE OR REPLACE TRIGGER crea_ID_Albaran
193 BEFORE INSERT ON ALBARAN
193 FOR EACH ROW
195 BEGIN
195 SELECT S_ID_Albaran.NEXTVAL INTO:NEW.ID_Albaran FROM DUAL;
197 END crea_ID_Albaran;

197 /

199 /

201
203 CREATE OR REPLACE TRIGGER Crea_Nuevo_Producto
203 BEFORE INSERT ON PRODUCTO
203 FOR EACH ROW
205 BEGIN
205 SELECT S_ID_Producto.NEXTVAL INTO:NEW.ID_Producto FROM DUAL;
207 END Crea_Nuevo_Producto;

207 /

209
211 CREATE OR REPLACE TRIGGER Crea_Nuevo_Traspaso
211 BEFORE INSERT ON TRASPASO
211 FOR EACH ROW
213 BEGIN
213 SELECT S_ID_Traspaso.NEXTVAL INTO:NEW.ID_Traspaso FROM DUAL;
215 END Crea_Nuevo_Traspaso;

215 /

217 /

219
221 CREATE OR REPLACE TRIGGER Crea_Nuevo_Solicitud
221 BEFORE INSERT ON SOLICITUD_TRASPASO
221 FOR EACH ROW
223 BEGIN
223 SELECT S_ID_Solicitud.NEXTVAL INTO:NEW.ID_Solicitud FROM DUAL;
225 END Crea_Nuevo_Solicitud;

225 /

227
229 CREATE OR REPLACE TRIGGER crea_ID_Factura
229 BEFORE INSERT ON FACTURA
229 FOR EACH ROW
231 BEGIN
231 SELECT S_ID_Factura.NEXTVAL INTO:NEW.ID_Factura FROM DUAL;
233 END crea_ID_Factura;

233 /

235 /

237
237 CREATE OR REPLACE TRIGGER crea_ID_Emplazamiento
237 BEFORE INSERT ON EMPLAZAMIENTO

```

```

239 | FOR EACH ROW
      | BEGIN
241 |     SELECT S_ID_Emplazamiento.NEXTVAL
      |     INTO :NEW.ID_Emplazamiento FROM DUAL;
243 | END crea_ID_Emplazamiento;
      | /
245 |
247 | CREATE OR REPLACE TRIGGER crea_ID_Venta
      | BEFORE INSERT ON Venta
      | FOR EACH ROW
249 | BEGIN
      |     SELECT S_ID_Venta.NEXTVAL
251 |     INTO :NEW.ID_Venta FROM DUAL;
      | END crea_ID_Venta;
253 | /

```

sql/tablas.sql

## 10.2. Funciones y Procedures

```

1 | CREATE OR REPLACE PROCEDURE Socio_Nuevo (
      |     p_Nombre IN SOCIO.Nombre%TYPE,
      |     p_Apellidos IN SOCIO.Apellidos%TYPE,
      |     p_Direccion IN SOCIO.Direccion%TYPE,
      |     p_FechaDeNacimiento IN SOCIO.FechaDeNacimiento%TYPE,
      |     p_Email IN SOCIO.Email%TYPE,
      |     p_DNI IN SOCIO.DNI%TYPE)
      | IS BEGIN
      |     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,
      |     P_Descripcion IN PRODUCTO.Descripcion%TYPE,
      |     P_Categoria IN PRODUCTO.Categoria%TYPE,
      |     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,
      |     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,
      |     p_Cantidad IN ASOCIACION_PRODUCTO_TRASPASO.Cantidad%TYPE)
45 | IS BEGIN
      |     INSERT INTO ASOCIACION_PRODUCTO_TRASPASO
47 |     VALUES (p_ID_Traspaso, p_ID_Producto, p_Cantidad);
49 | END PRODUCTO_TRASPASO_Nuevo;
51 | /

```

```

53 CREATE OR REPLACE PROCEDURE PRODUCTO_SOLICITUD_Nuevo (
54     p_ID_Solicitud IN ASOCIACION_PRODUCTO_SOLICITUD.ID_Solicitud%TYPE,
55     p_ID_Producto IN ASOCIACION_PRODUCTO_SOLICITUD.ID_Producto%TYPE,
56     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_PrecioTotal IN FACTURA.PrecioTotal%TYPE,
106     p_FechaDeExpedicion IN FACTURA.FechaDeExpedicion%TYPE,
107     p_Devuelto IN FACTURA.Devuelto%TYPE,
108     p_ID_Venta IN FACTURA.ID_Venta%TYPE,
109     p_ID_Emplazamiento IN FACTURA.ID_Emplazamiento%TYPE
110 )
111 IS BEGIN
112     INSERT INTO FACTURA VALUES(
113     p_ID, p_PrecioTotal, p_FechaDeExpedicion, p_Devuelto, p_ID_Venta,
114     p_ID_Emplazamiento);
115 END Factura_Nueva;
116
117 /
118
119 CREATE OR REPLACE PROCEDURE PROVEEDOR_Nuevo (
120     p_CIF IN PROVEEDOR.CIF%TYPE,
121     p_Nombre IN PROVEEDOR.Nombre%TYPE,
122     p_Telefono IN PROVEEDOR.Telefono%TYPE,
123     p_Email IN PROVEEDOR.Email%TYPE
124 ) IS BEGIN
125     INSERT INTO PROVEEDOR
126     VALUES (p_CIF, p_Nombre, p_Telefono, p_Email);
127 END PROVEEDOR_Nuevo;

```

```

129 /
131
133 CREATE OR REPLACE PROCEDURE ALBARAN_Nuevo (
135     p_ID_Albaran IN ALBARAN.ID_Albaran %TYPE,
137     p_FechaFirma IN ALBARAN.FechaFirma %TYPE,
139     p_PrecioTotal IN ALBARAN.PrecioTotal %TYPE,
141     p_ID_Pedido IN ALBARAN.ID_PEDIDO %TYPE
143 ) IS BEGIN
145     INSERT INTO ALBARAN
147     VALUES ( p_ID_Albaran, p_FechaFirma, p_PrecioTotal, p_ID_Pedido);
149 END ALBARAN_Nuevo;
151 /
153
155 CREATE OR REPLACE PROCEDURE PEDIDO_Nuevo (
157     p_ID_Pedido IN PEDIDO.ID_Pedido %TYPE,
159     p_FechaPedido IN PEDIDO.FechaPedido %TYPE,
161     p_PrecioTotal IN PEDIDO.PrecioTotal %TYPE,
163     p_ID_Emplazamiento IN PEDIDO.ID_Emplazamiento %TYPE,
165     p_CIF IN PEDIDO.CIF %TYPE
167 ) IS BEGIN
169     INSERT INTO PEDIDO
171     VALUES ( p_ID_Pedido ,p_FechaPedido , p_PrecioTotal ,
173         p_ID_Emplazamiento , p_CIF);
175 END PEDIDO_Nuevo;
177 /
179
181 CREATE OR REPLACE PROCEDURE PEDIDO_PRODUCTO_Nuevo (
183     p_ID_Producto IN ASOCIACION_PEDIDO_PRODUCTO.ID_Producto %TYPE,
185     p_ID_Pedido IN ASOCIACION_PEDIDO_PRODUCTO.ID_Pedido %TYPE,
187     p_Cantidad IN ASOCIACION_PEDIDO_PRODUCTO.Cantidad %TYPE,
189     p_PrecioCompra IN ASOCIACION_PEDIDO_PRODUCTO.PrecioCompra %TYPE,
191     p_IVA IN ASOCIACION_PEDIDO_PRODUCTO.IVA %TYPE,
193     p_PrecioLinea IN ASOCIACION_PEDIDO_PRODUCTO.preciolinea %TYPE) IS
195 BEGIN
197     INSERT INTO ASOCIACION_PEDIDO_PRODUCTO
199     VALUES (p_ID_Producto, p_ID_Pedido , p_Cantidad , p_PrecioCompra,
201         p_IVA, p_PrecioLinea);
203 END PEDIDO_PRODUCTO_Nuevo;
205 /
207
209 CREATE OR REPLACE PROCEDURE Venta_Nueva(
211     P_ID_Venta IN VENTA.ID_Venta %TYPE,
213     P_PrecioTotal IN VENTA.PrecioTotal %TYPE,
215     P_FechaVenta IN VENTA.FechaVenta %TYPE,
217     P_DNI IN VENTA.DNI %TYPE)
219 IS BEGIN
221     INSERT INTO VENTA
223     VALUES(P_ID_Venta, P_PrecioTotal, P_FechaVenta, P_DNI);
225 END Venta_Nueva;
227 /
229
231 CREATE OR REPLACE PROCEDURE VENTA_PRODUCTO_Nueva(
233     P_ID_Venta IN ASOCIACION_VENTA_PRODUCTO.ID_Venta %TYPE,
235     P_ID_Producto IN ASOCIACION_VENTA_PRODUCTO.ID_Producto %TYPE,
237     P_Cantidad IN ASOCIACION_VENTA_PRODUCTO.Cantidad %TYPE,
239     P_PrecioVenta IN ASOCIACION_VENTA_PRODUCTO.PrecioVenta %TYPE,
241     P_IvaVenta IN ASOCIACION_VENTA_PRODUCTO.IvaVenta %TYPE,
243     P_PrecioLinea IN ASOCIACION_VENTA_PRODUCTO.PrecioLinea %TYPE
245 )
247 IS
249 BEGIN
251     INSERT INTO ASOCIACION_VENTA_PRODUCTO
253     VALUES(P_ID_Venta, P_ID_Producto,P_Cantidad,P_PrecioVenta,P_IvaVenta,P_PrecioLinea);
255 END VENTA_PRODUCTO_Nueva;
257 /
259
261 CREATE OR REPLACE PROCEDURE MODIFICA_PROVEEDOR_NOMBRE
263 (p_CIF IN PROVEEDOR.CIF %TYPE,
265 p_Nombre IN PROVEEDOR.Nombre %TYPE) IS

```

```

207 BEGIN
    UPDATE PROVEEDOR SET Nombre = p_Nombre WHERE p_CIF = CIF;
209 END MODIFICA_PROVEEDOR_NOMBRE;

211 /

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

219 /

221 CREATE OR REPLACE PROCEDURE MODIFICA_PROVEEDOR_EMAIL
    (p_CIF IN PROVEEDOR.CIF%TYPE,
     p_Email IN PROVEEDOR.Email%TYPE) IS
223 BEGIN
    UPDATE PROVEEDOR SET Email = p_Email WHERE p_CIF = CIF;
225 END MODIFICA_PROVEEDOR_EMAIL;

227 /

229 CREATE OR REPLACE PROCEDURE MODIFICA_PRODUCTO_DESCRIPCION (
231     p_ID_Producto IN PRODUCTO.ID_Producto%TYPE,
     p_Nombre IN PRODUCTO.Nombre%TYPE,
233     p_Descripcion IN PRODUCTO.Descripcion%TYPE,
     p_Categoria IN PRODUCTO.Categoria%TYPE,
235     p_PrecioProducto IN PRODUCTO.PrecioProducto%TYPE,
     p_IVA IN PRODUCTO.IVA%TYPE)
237 IS BEGIN
    UPDATE PRODUCTO SET Descripcion = p_Descripcion WHERE p_ID_Producto = ID_Producto;
239 END MODIFICA_PRODUCTO_DESCRIPCION;

241 /

243 CREATE OR REPLACE PROCEDURE MODIFICA_PRODUCTO_PRECIO (
     p_ID_Producto IN PRODUCTO.ID_Producto%TYPE,
245     p_Nombre IN PRODUCTO.Nombre%TYPE,
     p_Descripcion IN PRODUCTO.Descripcion%TYPE,
247     p_Categoria IN PRODUCTO.Categoria%TYPE,
     p_PrecioProducto IN PRODUCTO.PrecioProducto%TYPE,
249     p_IVA IN PRODUCTO.IVA%TYPE)
    IS BEGIN
251     UPDATE PRODUCTO SET PrecioProducto = p_PrecioProducto WHERE p_ID_Producto = ID_Producto;
    END MODIFICA_PRODUCTO_PRECIO;
253 /

255 CREATE OR REPLACE PROCEDURE MODIFICA_PRODUCTO_IVA (
257     p_ID_Producto IN PRODUCTO.ID_Producto%TYPE,
     p_Nombre IN PRODUCTO.Nombre%TYPE,
259     p_Descripcion IN PRODUCTO.Descripcion%TYPE,
     p_Categoria IN PRODUCTO.Categoria%TYPE,
261     p_PrecioProducto IN PRODUCTO.PrecioProducto%TYPE,
     p_IVA IN PRODUCTO.IVA%TYPE)
263 IS BEGIN
    UPDATE PRODUCTO SET IVA = p_IVA
265     WHERE p_ID_Producto = ID_Producto;
    END MODIFICA_PRODUCTO_IVA;
267 /

269 CREATE OR REPLACE PROCEDURE MODIFICA_STOCK_CANTIDAD
271     (p_ID_Emplazamiento IN STOCK.ID_Emplazamiento%TYPE,
     p_ID_Producto IN STOCK.ID_Producto%TYPE,
273     p_Cantidad IN STOCK.Cantidad%TYPE)
    IS BEGIN
275     UPDATE STOCK SET Cantidad = p_Cantidad
     WHERE p_ID_Emplazamiento = ID_Emplazamiento AND p_ID_PRODUCTO = ID_PRODUCTO;
277 END MODIFICA_STOCK_CANTIDAD;

279 /

281 create or replace PROCEDURE MODIFICA_SOCIO_DIRECCION(
     m_DNI IN SOCIO.DNI%TYPE,

```



```

283      m_DIRECCION IN SOCIO.DIRECCION %TYPE)
284      IS BEGIN
285      UPDATE SOCIO SET DIRECCION = m_DIRECCION where m_DNI = DNI;
286      COMMIT WORK;
287 END MODIFICA_SOCIO_DIRECCION;

289 /

291 create or replace PROCEDURE MODIFICA_SOCIO_EMAIL(m_DNI IN SOCIO.DNI %TYPE, m_email IN SOCIO.EMAIL %TYPE)
292 IS BEGIN
293     UPDATE SOCIO SET EMAIL = m_email where m_DNI = DNI;
294     COMMIT WORK;
295 END MODIFICA_SOCIO_EMAIL;

297 /

299 CREATE OR REPLACE PROCEDURE MODIFICA_EMPLAZAMIENTO_DIR(
300     m_ID IN EMPLAZAMIENTO.ID_Emplazamiento %TYPE, m_Direccion IN EMPLAZAMIENTO.Direccion %TYPE)
301 IS BEGIN
302     UPDATE EMPLAZAMIENTO SET Direccion = m_Direccion where m_ID = ID_Emplazamiento;
303 END MODIFICA_EMPLAZAMIENTO_DIR;

305 /

307 CREATE OR REPLACE PROCEDURE MODIFICA_EMPLAZAMIENTO_TEL(
308     m_ID IN EMPLAZAMIENTO.ID_Emplazamiento %TYPE, m_Telefono IN EMPLAZAMIENTO.Telefono %TYPE)
309 IS BEGIN
310     UPDATE EMPLAZAMIENTO SET Telefono = m_Telefono where m_ID = ID_Emplazamiento;
311 END MODIFICA_EMPLAZAMIENTO_TEL;

313 /

315 CREATE OR REPLACE PROCEDURE MODIFICA_FACTURA_DEVUELTO
316 (m_ID_FACTURA IN FACTURA.ID_FACTURA %TYPE, m_Devuelto IN FACTURA.Devuelto %TYPE)
317 IS BEGIN
318     UPDATE FACTURA SET Devuelto = m_Devuelto where m_ID_FACTURA = ID_FACTURA;
319 END MODIFICA_FACTURA_DEVUELTO;

321 /

323 CREATE OR REPLACE PROCEDURE ELIMINA_PROVEEDOR(p_CIF IN PROVEEDOR.CIF %TYPE)
324 IS BEGIN
325     DELETE FROM PROVEEDOR WHERE p_CIF = CIF;
326 END ELIMINA_PROVEEDOR;

329 /

331 CREATE OR REPLACE PROCEDURE ELIMINA_PRODUCTO(p_ID_Producto IN PRODUCTO.ID_Producto %TYPE)
332 IS BEGIN
333     DELETE FROM PRODUCTO WHERE ID_Producto = p_ID_Producto;
334 END ELIMINA_PRODUCTO;

335 /

337 CREATE OR REPLACE PROCEDURE ELIMINA_EMPLAZAMIENTO(p_ID_Emplazamiento IN EMPLAZAMIENTO.
338     ID_Emplazamiento %TYPE)
339 IS BEGIN
340     DELETE FROM EMPLAZAMIENTO WHERE ID_Emplazamiento = p_ID_Emplazamiento;
341 END ELIMINA_EMPLAZAMIENTO;

343 /

345 CREATE OR REPLACE PROCEDURE ELIMINA_SOCIO(p_DNI IN SOCIO.DNI %TYPE)
346 IS BEGIN
347     DELETE FROM SOCIO WHERE DNI = p_DNI;
348 END ELIMINA_SOCIO;

349 /

351 /*FUNCIONES*/

353 CREATE OR REPLACE FUNCTION precio_A_Venta_producto
354 (f_ID_Venta IN ASOCIACION_VENTA_PRODUCTO.ID_Venta %TYPE,
355  f_Cantidad IN ASOCIACION_VENTA_PRODUCTO.Cantidad %TYPE,
356  f_PrecioVenta IN ASOCIACION_VENTA_PRODUCTO.PrecioVenta %TYPE)
357 RETURN NUMBER is f_PrecioLinea ASOCIACION_VENTA_PRODUCTO.PRECIOLINEA %TYPE;

```

```

359 BEGIN
    f_PrecioLinea := f_PrecioVenta * f_Cantidad;
361 RETURN(f_PrecioLinea);
    END precio_A_Venta_producto;
363 /
365
367 CREATE OR REPLACE FUNCTION precio_Venta
    (f_ID_Venta IN VENTA.ID_Venta%TYPE)
    RETURN NUMBER is f_PrecioTotal VENTA.PRECIOTOTAL%TYPE;
369 BEGIN
    select SUM(precioLinea) into f_PrecioTotal from ASOCIACION_VENTA_PRODUCTO
371 where ID_Venta = f_ID_Venta;
    RETURN (f_PrecioTotal);
373 END precio_Venta;
375 /
377
379 CREATE OR REPLACE FUNCTION precio_A_Pedido_producto
    (f_ID_Pedido IN ASOCIACION_PEDIDO_PRODUCTO.ID_Pedido%TYPE,
    f_Cantidad IN ASOCIACION_PEDIDO_PRODUCTO.Cantidad%TYPE,
    f_PrecioCompra IN ASOCIACION_PEDIDO_PRODUCTO.PrecioCompra%TYPE)
381 RETURN NUMBER is f_PrecioLinea ASOCIACION_PEDIDO_PRODUCTO.PrecioLinea%TYPE;
    BEGIN
383 f_PrecioLinea := f_PrecioCompra * f_Cantidad;
    RETURN (f_PrecioLinea);
385 END precio_A_Pedido_producto;
387 /
389
391 CREATE OR REPLACE FUNCTION precio_Pedido
    (f_ID_Pedido IN PEDIDO.ID_Pedido%TYPE)
    RETURN NUMBER is f_PrecioTotal PEDIDO.PRECIOTOTAL%TYPE;
393 BEGIN
    select SUM(PrecioLinea) into f_PrecioTotal from ASOCIACION_PEDIDO_PRODUCTO
    where ID_Pedido = f_ID_Pedido;
395 RETURN (f_PrecioTotal);
    END precio_Pedido;
397 /
399
401 CREATE OR REPLACE FUNCTION ganancias_mensuales(fechainicio IN factura.fechadeexpedicion%TYPE, fechafin
    IN factura.fechadeexpedicion%TYPE )
    return number is f_preciototal factura.preciototal%TYPE;
403 v_preciototal factura.preciototal%TYPE;
    p_preciototal albaran.preciototal%TYPE;
    begin
405 select sum(preciototal) into v_preciototal from factura where fechadeexpedicion between fechainicio
        and fechafin;
    select sum(preciototal) into p_preciototal from albaran where fechafirma between fechainicio and
        fechafin;
407 f_preciototal := v_preciototal - p_preciototal;
    return (f_preciototal);
409 end ganancias_mensuales;
411 /

```

sql/funciones\_y\_procedures.sql

## 10.3. Triggers

```

1  /* TRIGGERS */
3  CREATE OR REPLACE TRIGGER pedido_minimo
    BEFORE INSERT or UPDATE ON ASOCIACION_PEDIDO_PRODUCTO
5  FOR EACH ROW
    BEGIN
7  IF :NEW.cantidad < 20
    THEN raise_application_error(-20600, :NEW.cantidad || 'No se pueden pedir menos de 20 unidades de
        un producto');
9  END IF;

```

```

11 END;
12 /
13
14 CREATE OR REPLACE TRIGGER descuento_socio
15 BEFORE INSERT ON factura
16 FOR EACH ROW
17 declare v_preciototal VENTA.PRECIOTOTAL %TYPE;
18 v_DNI VENTA.dni %TYPE;
19 BEGIN
20     select preciototal into v_preciototal from venta where id_Venta = :NEW.id_venta;
21     select dni into v_dni from venta where id_venta = :NEW.id_Venta;
22     IF v_DNI IS NOT NULL then
23         UPDATE venta set preciototal = v_preciototal * 0.95 where id_Venta = :NEW.id_venta;
24         :New.preciototal := v_preciototal * 0.95;
25     else
26         update venta set preciototal = v_preciototal where id_venta = :NEW.id_venta;
27     END IF;
28 END;
29 /
30
31 CREATE OR REPLACE TRIGGER stock_minimo
32 AFTER INSERT OR UPDATE ON stock
33 FOR EACH ROW
34 BEGIN
35     IF :NEW.cantidad <= 0
36     THEN
37         raise_application_error(-20601, :NEW.cantidad || 'No se puede realizar esta operacion');
38     END IF;
39 END;
40 /
41
42 /
43
44 CREATE OR REPLACE TRIGGER Inicializa_nueva_Venta
45 BEFORE INSERT ON VENTA
46 FOR EACH ROW
47 BEGIN
48     :NEW.PrecioTotal := 0;
49     :NEW.FechaVenta := SYSDATE;
50 END Comprueba_Venta;
51 /
52
53 CREATE OR REPLACE TRIGGER Inicializa_nuevo_Pedido
54 BEFORE INSERT ON PEDIDO
55 FOR EACH ROW
56 BEGIN
57     :NEW.PrecioTotal := 0;
58     :NEW.FechaPedido := SYSDATE;
59 END Comprueba_Pedido;
60 /
61
62 /
63
64 CREATE OR REPLACE TRIGGER Inicializa_nueva_Factura
65 BEFORE INSERT ON Factura
66 FOR EACH ROW
67 BEGIN
68     :NEW.FechaDeExpedicion := SYSDATE;
69 END Inicializa_nueva_Factura;
70 /
71
72 /
73
74 CREATE OR REPLACE TRIGGER modifica_stock_venta
75 BEFORE INSERT ON FACTURA
76 FOR EACH ROW
77 DECLARE
78     e_ID_Emplazamiento Emplazamiento.ID_Emplazamiento %TYPE;
79 v_ID_Venta Venta.ID_VENTA %TYPE;
80
81 CURSOR all_prods
82 IS
83 SELECT id_venta, id_producto, cantidad
84 FROM ASOCIACION_VENTA_PRODUCTO
85 ORDER BY ID_producto;

```

```

87 m_id_venta ASOCIACION_VENTA_PRODUCTO.id_venta%TYPE;
88 m_id_producto Producto.id_producto%type;
89 m_cantidad ASOCIACION_VENTA_PRODUCTO.cantidad%TYPE;

91 BEGIN
92 select ID_Emplazamiento into e_ID_Emplazamiento from Emplazamiento where ID_Emplazamiento = :NEW.
   ID_Emplazamiento;
93 select ID_Venta into v_ID_Venta from Venta where ID_Venta = :New.ID_Venta;

95 OPEN all_prods;
96 LOOP
97     Fetch all_prods INTO m_id_venta,m_id_producto,m_cantidad;
98     EXIT WHEN all_prods%NOTFOUND;
99     if m_id_venta = v_id_Venta
100     THEN
101         update stock set cantidad = cantidad-m_cantidad where id_emplazamiento = e_ID_Emplazamiento AND
           id_producto =m_ID_Producto;
102     END IF;
103 END LOOP;
104 CLOSE all_prods;
105 END modifica_stock_venta;

107 /

109 CREATE OR REPLACE TRIGGER inicializa_preciolinea_alv
110 BEFORE INSERT OR UPDATE ON ASOCIACION_VENTA_PRODUCTO
111 FOR EACH ROW
112 BEGIN
113     :NEW.preciolinea := :New.cantidad * :New.precioVenta;
114 END inicializa_preciolinea_alv;
115
117 /

119 CREATE OR REPLACE TRIGGER inicializa_preciototal_venta
120 BEFORE INSERT OR UPDATE ON ASOCIACION_VENTA_PRODUCTO
121 for each row
122 begin
123     UPDATE venta set preciototal = PRECIOTOTAL + (:New.cantidad * :New.precioVenta) where id_venta = :
       New.id_venta;
124 END inicializa_preciototal_venta;
125 /

127 CREATE OR REPLACE TRIGGER modifica_stock_pedido
128 BEFORE INSERT OR UPDATE ON ALBARAN
129 FOR EACH ROW
130 DECLARE
131 e_ID_Emplazamiento Emplazamiento.ID_Emplazamiento%TYPE;
132 p_ID_PEDIDO PEDIDO.ID_PEDIDO%TYPE;

135 CURSOR all_prods
136 IS
137 SELECT id_pedido,id_producto,cantidad
138 FROM ASOCIACION_PEDIDO_PRODUCTO
139 ORDER BY ID_producto;

141 m_id_pedido ASOCIACION_PEDIDO_PRODUCTO.id_pedido%TYPE;
142 m_id_producto Producto.id_producto%type;
143 m_cantidad ASOCIACION_PEDIDO_PRODUCTO.cantidad%TYPE;

145 BEGIN
147 select ID_Emplazamiento into e_ID_Emplazamiento from pedido where ID_pedido = :NEW.ID_Pedido;
148 select ID_PEDIDO into p_id_pedido from pedido where ID_pedido = :NEW.ID_Pedido;
149 OPEN all_prods;
150 LOOP
151     Fetch all_prods INTO m_id_pedido,m_id_producto,m_cantidad;
152     EXIT WHEN all_prods%NOTFOUND;
153     if m_id_pedido = p_id_pedido
154     THEN
155         update stock set cantidad = cantidad+m_cantidad where id_emplazamiento = e_ID_Emplazamiento AND
           id_producto =m_ID_Producto;
156     END IF;
157 END LOOP;
158 CLOSE all_prods;
159 END modifica_stock_pedido;

```

```

161 /
163 CREATE OR REPLACE TRIGGER modifica_stock_traspaso
BEFORE INSERT OR UPDATE ON ASOCIACION_PRODUCTO_TRASPASO
165 FOR EACH ROW
DECLARE
167 e_ID_Emplazamiento_salida Emplazamiento.ID_Emplazamiento%TYPE;
e_ID_Emplazamiento_entrada Emplazamiento.ID_Emplazamiento%TYPE;
169 t_ID_traspaso traspaso.ID_traspaso%TYPE;

171 BEGIN
select ID_Emplazamientosalida into e_ID_Emplazamiento_salida from traspaso where ID_traspaso = :NEW.
ID_traspaso;
173 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;
175 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;

177 END modifica_stock_traspaso;
179 /
181 CREATE OR REPLACE TRIGGER Inicializa_Nuevo_Pedido
BEFORE INSERT ON Pedido
FOR EACH ROW
183 BEGIN
:NEW.PrecioTotal := 0;
185 :NEW.FechaPedido := SYSDATE;
END Inicializa_Nuevo_Pedido;
187 /
189 /
191 CREATE OR REPLACE TRIGGER Inicializa_Nueva_ST
BEFORE INSERT ON Solicitud_Traspaso
193 FOR EACH ROW
BEGIN
195 :NEW.FechaSolicitud := SYSDATE;
END Inicializa_Nueva_ST;
197 /
199 /
201 CREATE OR REPLACE TRIGGER Inicializa_Nuevo_Traspaso
BEFORE INSERT ON Traspaso
FOR EACH ROW
203 BEGIN
:NEW.FechaTraspaso := SYSDATE;
205 END Inicializa_Nuevo_Traspaso;

207 /
209 create or replace TRIGGER solicitud_stock_minimo
BEFORE INSERT ON asociacion_producto_solicitud
211 FOR EACH ROW
DECLARE
213 e_id_emplazamientoentrada SOLICITUD_TRASPASO.ID_EMPLAZAMIENTOENTRADA%TYPE;
e_cantidad Stock.cantidad%TYPE;
215 BEGIN
217 SELECT id_emplazamientoentrada into e_id_emplazamientoentrada from solicitud_traspaso where
id_solicitud = :NEW.id_solicitud;
select cantidad into e_cantidad from stock where id_emplazamiento = e_id_emplazamientoentrada and
id_producto = :NEW.id_producto;
219 if (e_cantidad - :NEW.cantidad) <=5
THEN raise_application_error(-20601, :NEW.cantidad || 'No se permite la solicitud, el otro
emplazamiento alcanzara su stock minimo');
221 END IF;
END;
223 /
225 create or replace TRIGGER inicializa_preciolinea_alp
BEFORE INSERT OR UPDATE ON ASOCIACION_PEDIDO_PRODUCTO
227 FOR EACH ROW
BEGIN
229 :NEW.preciolinea := :New.cantidad * :New.preciocompra;

```

```

231 END inicializa_preciolinea_alp;
/

233 create or replace TRIGGER inicializa_preciototal_albaran
  BEFORE INSERT OR UPDATE ON Albaran
235   for each row
  DECLARE
237   p_preciototal pedido.preciototal %TYPE;
  begin
239   select preciototal into p_preciototal from pedido where id_pedido = :New.id_pedido;
    :New.preciototal := p_preciototal;
241 END inicializa_preciototal_albaran;

243 /

245 create or replace TRIGGER inicializa_preciototal_pedido
  BEFORE INSERT OR UPDATE ON ASOCIACION_pedido_producto
247   for each row
  begin
249   UPDATE pedido set preciototal = PRECIOTOTAL + (:New.cantidad * :New.preciocompra) where id_pedido =
    :New.id_pedido;
  END inicializa_preciototal_pedido;

251 /

253 create or replace trigger inicializa_precioventa_apv
255   before insert on asociacion_venta_producto
  for each row
257   declare
  p_precio producto.precioproducto %TYPE;
259   begin
  select precioproducto into p_precio from producto where id_producto = :New.id_producto;
261   :NEW.precioventa := p_precio;
  END inicializa_precioventa_apv;

263 /

265 create or replace trigger inicializa_IVA_apv
267   before insert on asociacion_venta_producto
  for each row
269   declare
  p_IVA PRODUCTO.IVA %TYPE;
271   begin
  select Iva into p_IVA from producto where id_producto = :New.id_producto;
273   :NEW.IVAVENTA := p_IVA;
  END inicializa_IVA_apv;

275 /

277 create or replace trigger inicializa_IVA_APP
279   before insert on asociacion_pedido_producto
  for each row
281   declare
  p_IVA PRODUCTO.IVA %TYPE;
283   begin
  select IVA into p_IVA from producto where id_producto = :New.id_producto;
285   :NEW.IVA := p_IVA;
  END inicializa_IVA_APP;

287 /

```

sql/triggers.sql

## 10.4. Pruebas

```

/* FUNCION AUXILIAR */
2 CREATE OR REPLACE FUNCTION EQUALS(salida BOOLEAN, salidaEsperada BOOLEAN)
  RETURN VARCHAR2 AS
4 BEGIN
  IF (salida = salidaEsperada) THEN
6     RETURN 'EXITO';

```

```

8         ELSE
          RETURN 'FALLO';
10     END IF;
11 END EQUALS;
12 /
13
14 CREATE OR REPLACE PROCEDURE PRINTR(nombre_prueba VARCHAR2, salida BOOLEAN,
                                     salidaEsperada BOOLEAN)
15 IS BEGIN
16     DBMS_OUTPUT.put_line(nombre_prueba || ':' || EQUALS(salida, salidaEsperada));
17 END PRINTR;
18 /
19
20 /* DEFINICION PRUEBAS */
21 CREATE OR REPLACE PACKAGE PRUEBAS_SOCIO AS
22 PROCEDURE inicializar;
23 PROCEDURE insertar
24     (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_apellidos VARCHAR2,
25      i_DNI VARCHAR2, i_direccion VARCHAR2, i_nacimiento DATE, i_email VARCHAR2
26      ,salidaEsperada BOOLEAN);
27 PROCEDURE actualizar
28     (nombre_prueba VARCHAR2, a_DNI VARCHAR2, a_direccion VARCHAR2,
29      a_email VARCHAR2, salidaEsperada BOOLEAN);
30 PROCEDURE eliminar
31     (nombre_prueba VARCHAR2, e_DNI VARCHAR2, salidaEsperada BOOLEAN);
32 END PRUEBAS_SOCIO;
33 /
34
35 CREATE OR REPLACE PACKAGE PRUEBAS_PROVEEDOR AS
36 PROCEDURE inicializar;
37 PROCEDURE insertar
38     (nombre_prueba VARCHAR2, i_CIF VARCHAR2, i_nombre VARCHAR2,
39      i_telefono INT, i_email VARCHAR2, salidaEsperada BOOLEAN);
40 PROCEDURE actualizar
41     (nombre_prueba VARCHAR2, a_cif VARCHAR2, a_nombre VARCHAR2,
42      a_telefono INT, a_email VARCHAR2, salidaEsperada BOOLEAN);
43 PROCEDURE eliminar
44     (nombre_prueba VARCHAR2, e_CIF VARCHAR2, salidaEsperada BOOLEAN);
45 END PRUEBAS_PROVEEDOR;
46 /
47
48 CREATE OR REPLACE PACKAGE PRUEBAS_EMPLAZAMIENTO AS
49 PROCEDURE inicializar;
50 PROCEDURE insertar
51     (nombre_prueba VARCHAR2, i_direccion VARCHAR2, i_telefono INT,
52      i_tipo VARCHAR2, salidaEsperada BOOLEAN);
53 PROCEDURE actualizar
54     (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_direccion VARCHAR2,
55      a_telefono INT, salidaEsperada BOOLEAN);
56 PROCEDURE eliminar
57     (nombre_prueba VARCHAR2, e_id_emplazamiento INT, salidaEsperada BOOLEAN);
58 END PRUEBAS_EMPLAZAMIENTO;
59 /
60
61 CREATE OR REPLACE PACKAGE PRUEBAS_ALBARAN AS
62 PROCEDURE inicializar;
63 PROCEDURE insertar
64     (nombre_prueba VARCHAR2, i_fecha DATE,
65      i_precio NUMBER, i_id_pedido INT, salidaEsperada BOOLEAN);
66 PROCEDURE eliminar
67     (nombre_prueba VARCHAR2, e_id_albaran INT, salidaEsperada BOOLEAN);
68 END PRUEBAS_ALBARAN;
69 /
70
71 CREATE OR REPLACE PACKAGE PRUEBAS_PEDIDO AS
72 PROCEDURE inicializar;
73 PROCEDURE insertar
74     (nombre_prueba VARCHAR2, i_fecha DATE, i_precio NUMBER,
75      i_id_emplazamiento INT, i_cif VARCHAR2, salidaEsperada BOOLEAN);
76 PROCEDURE eliminar
77     (nombre_prueba VARCHAR2, e_id_pedido INT, salidaEsperada BOOLEAN);

```

```

84 END PRUEBAS_PEDIDO;

86 /

88 CREATE OR REPLACE PACKAGE PRUEBAS_PRODUCTO AS
    PROCEDURE inicializar;
    PROCEDURE insertar
        (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_descripcion VARCHAR2,
92         i_categoria VARCHAR2, i_precioProducto INT, i_iva INT, salidaEsperada BOOLEAN);
    PROCEDURE actualizar
        (nombre_prueba VARCHAR2, a_id_producto INT, a_descripcion VARCHAR2,
94         a_precioProducto INT, a_iva INT, salidaEsperada BOOLEAN);
    PROCEDURE eliminar
        (nombre_prueba VARCHAR2, e_id_producto INT, salidaEsperada BOOLEAN);
96 END PRUEBAS_PRODUCTO;

100 /

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

112 /

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

122 /

124 /

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

134 /

136 /

138 /

140 CREATE OR REPLACE PACKAGE PRUEBAS_TRASPASO AS
    PROCEDURE inicializar;
    PROCEDURE insertar
        (nombre_prueba VARCHAR2, i_fechaTraspaso DATE, i_id_emplazamientoSalida INT,
142         i_id_emplazamientoEntrada INT, salidaEsperada BOOLEAN);
    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;
    PROCEDURE insertar
        (nombre_prueba VARCHAR2, i_id_venta INT, i_id_producto INT,
152         i_cantidad NUMBER, i_precioVenta NUMBER, i_ivaVenta NUMBER,
        i_precioLinea NUMBER, salidaEsperada BOOLEAN);
    PROCEDURE eliminar
        (nombre_prueba VARCHAR2, e_id_venta INT, e_id_producto INT, salidaEsperada BOOLEAN);
154 END PRUEBAS_A_VENTA_PRODUCTO;

```



```

160 /
162
164 CREATE OR REPLACE PACKAGE PRUEBA_STOCK AS
    PROCEDURE inicializar;
    PROCEDURE insertar
166         (nombre_prueba VARCHAR2, i_id_emplazamiento INT, i_id_producto INT,
168          i_cantidad INT, salidaEsperada BOOLEAN);
    PROCEDURE actualizar
170         (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_id_producto INT,
172          a_cantidad INT, salidaEsperada BOOLEAN);
    PROCEDURE eliminar
174         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT,
176          salidaEsperada BOOLEAN);
    PROCEDURE stock_correcto
178         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
180          INT, salidaEsperada BOOLEAN);
    PROCEDURE stock_correcto_p
182         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
184          INT, salidaEsperada BOOLEAN);
    PROCEDURE stock_correcto_t
186         (nombre_prueba VARCHAR2, e_id_emplazamiento_envia int, e_id_emplazamiento_recibe int,
188          stock_previo_envia int, stock_previo_recibe int,
190          cantidad int, e_id_producto int, salidaEsperada BOOLEAN);
192 END PRUEBA_STOCK;
194 /
196
198 CREATE OR REPLACE PACKAGE PRUEBAS_A_PRODUCTO_SOLICITUD AS
    PROCEDURE inicializar;
    PROCEDURE insertar
200         (nombre_prueba VARCHAR2, i_id_producto INT, i_id_Solicitud INT,
202          i_cantidad NUMBER, salidaEsperada BOOLEAN);
    PROCEDURE eliminar
204         (nombre_prueba VARCHAR2, e_id_Solicitud INT, e_id_producto INT, salidaEsperada BOOLEAN);
206 END PRUEBAS_A_PRODUCTO_SOLICITUD;
208 /
210
212 CREATE OR REPLACE PACKAGE PRUEBAS_A_PRODUCTO_TRASPASO AS
    PROCEDURE inicializar;
    PROCEDURE insertar
214         (nombre_prueba VARCHAR2, i_id_producto INT, i_id_Traspaso INT,
216          i_cantidad NUMBER, salidaEsperada BOOLEAN);
    PROCEDURE eliminar
218         (nombre_prueba VARCHAR2, e_id_Traspaso INT, e_id_producto INT, salidaEsperada BOOLEAN);
220 END PRUEBAS_A_PRODUCTO_TRASPASO;
222 /
224
226 CREATE OR REPLACE PACKAGE PRUEBAS_A_PEDIDO_PRODUCTO AS
    PROCEDURE inicializar;
    PROCEDURE insertar
228         (nombre_prueba VARCHAR2, i_id_pedido INT, i_id_producto INT,
230          i_cantidad NUMBER, i_precioCompra INT, i_iva INT, i_precioLinea INT,
232          salidaEsperada BOOLEAN);
    PROCEDURE eliminar
234         (nombre_prueba VARCHAR2, e_id_pedido INT, e_id_producto INT,
236          salidaEsperada BOOLEAN);
238 END PRUEBAS_A_PEDIDO_PRODUCTO;
240 /
242
244 /* CUERPOS DE PRUEBAS */
246
248 CREATE OR REPLACE PACKAGE BODY PRUEBAS_SOCIO AS
    PROCEDURE inicializar AS
    BEGIN
250         DELETE FROM SOCIO;
252     END inicializar;
254
    PROCEDURE insertar
256         (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_apellidos VARCHAR2,
258          i_DNI VARCHAR2, i_direccion VARCHAR2, i_nacimiento DATE, i_email VARCHAR2
260          ,salidaEsperada BOOLEAN) AS
262     salida BOOLEAN := true;

```

```

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

```

```

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

```

```

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

```

```

458 BEGIN
459     DELETE FROM PRODUCTO;
460 END inicializar;

462 PROCEDURE insertar
463     (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_descripcion VARCHAR2,
464      i_categoria VARCHAR2, i_precioProducto INT, i_iva INT, salidaEsperada BOOLEAN) AS
465 salida BOOLEAN := true;
466 actual_producto %ROWTYPE;
467 w_ID_Producto INT;
468 BEGIN
469     INSERT INTO producto VALUES(null,i_nombre, i_descripcion, i_categoria, i_precioProducto,
470      i_iva);

471     w_ID_Producto := S_ID_Producto.currval;
472     SELECT * INTO actual FROM PRODUCTO WHERE ID_Producto = w_ID_Producto;

473     IF (actual.nombre<>i_nombre) OR (actual.descripcion<>i_descripcion) OR (actual.categoria<>
474      i_categoria) OR (actual.precioProducto<>i_precioProducto) OR (actual.iva<>i_iva) THEN
475         salida := false;
476     END IF;

477     PRINTR(nombre_prueba, salida, salidaEsperada);

478     EXCEPTION
479     WHEN OTHERS THEN
480         PRINTR(nombre_prueba, false, salidaEsperada);
481         ROLLBACK;
482 END insertar;

484 PROCEDURE actualizar
485     (nombre_prueba VARCHAR2, a_id_producto INT, a_descripcion VARCHAR2,
486      a_precioProducto INT, a_iva INT, salidaEsperada BOOLEAN) AS
487 salida BOOLEAN := true;
488 actual_producto %ROWTYPE;
489 BEGIN
490     UPDATE producto SET DESCRIPCION = a_descripcion, PRECIOPRODUCTO = a_precioProducto, IVA =
491      a_iva
492      where ID_Producto = a_id_producto;

493     SELECT * INTO actual FROM PRODUCTO WHERE ID_Producto = a_id_producto;
494     IF (actual.descripcion<>a_descripcion) OR
495      (actual.precioProducto<>a_precioProducto) OR
496      (actual.iva<>a_iva) THEN
497         salida := false;
498     END IF;

499     PRINTR(nombre_prueba, salida, salidaEsperada);

500     EXCEPTION
501     WHEN OTHERS THEN
502         PRINTR(nombre_prueba, false, salidaEsperada);
503         ROLLBACK;
504 END actualizar;

506 PROCEDURE eliminar
507     (nombre_prueba VARCHAR2, e_id_producto INT, salidaEsperada BOOLEAN) AS
508 salida BOOLEAN := true;
509 cantidad INT;
510 BEGIN
511     DELETE FROM PRODUCTO WHERE ID_Producto = e_id_producto;

512     SELECT COUNT(*) INTO cantidad FROM PRODUCTO WHERE ID_Producto = e_id_producto;
513     IF (cantidad<>0) THEN
514         salida := false;
515     END IF;

516     PRINTR(nombre_prueba, salida, salidaEsperada);

517     EXCEPTION
518     WHEN OTHERS THEN
519         PRINTR(nombre_prueba, false, salidaEsperada);
520         ROLLBACK;
521 END eliminar;
522 END PRUEBAS_PRODUCTO;
523 /

```

```

532 CREATE OR REPLACE PACKAGE BODY PRUEBAS_FACTURA AS
533
534     PROCEDURE inicializar AS
535     BEGIN
536         DELETE FROM FACTURA;
537     END inicializar;
538
539     PROCEDURE insertar
540     (nombre_prueba VARCHAR2, i_precioTotal INT, i_fechaDeExpedicion DATE, i_devuelto VARCHAR2,
541      i_id_venta INT, i_id_emplazamiento INT, salidaEsperada BOOLEAN) AS
542     salida BOOLEAN := true;
543     actual factura%ROWTYPE;
544     w_ID_factura INT;
545     BEGIN
546         INSERT INTO factura VALUES(null,i_precioTotal, i_fechaDeExpedicion, i_devuelto, i_id_venta,
547          i_id_emplazamiento);
548
549         w_ID_factura := S_ID_Factura.currval;
550         SELECT * INTO actual FROM factura WHERE ID_factura = w_ID_factura;
551
552         IF (actual.precioTotal<>i_precioTotal) OR (actual.fechaDeExpedicion<>i_fechaDeExpedicion) OR
553         (actual.devuelto<>i_devuelto) OR (actual.id_venta<>i_id_venta) OR (actual.id_emplazamiento<>
554         i_id_emplazamiento) THEN
555             salida := false;
556         END IF;
557
558         PRINTR(nombre_prueba, salida, salidaEsperada);
559
560         EXCEPTION
561         WHEN OTHERS THEN
562             PRINTR(nombre_prueba, false, salidaEsperada);
563             ROLLBACK;
564     END insertar;
565
566     PROCEDURE actualizar
567     (nombre_prueba VARCHAR2, a_id_factura INT, a_devuelto VARCHAR2,
568      salidaEsperada BOOLEAN) AS
569     salida BOOLEAN := true;
570     actual factura%ROWTYPE;
571     BEGIN
572         UPDATE factura SET DEVUELTO = a_devuelto where ID_factura = a_id_factura;
573
574         SELECT * INTO actual FROM factura WHERE ID_factura = a_id_factura;
575         IF (actual.devuelto<>a_devuelto) THEN
576             salida := false;
577         END IF;
578
579         PRINTR(nombre_prueba, salida, salidaEsperada);
580
581         EXCEPTION
582         WHEN OTHERS THEN
583             PRINTR(nombre_prueba, false, salidaEsperada);
584             ROLLBACK;
585     END actualizar;
586
587     PROCEDURE eliminar
588     (nombre_prueba VARCHAR2, e_id_factura INT, salidaEsperada BOOLEAN) AS
589     salida BOOLEAN := true;
590     cantidad INT;
591     BEGIN
592         DELETE FROM factura WHERE ID_factura = e_id_factura;
593
594         SELECT COUNT(*) INTO cantidad FROM factura WHERE ID_factura = e_id_factura;
595         IF (cantidad<>0) THEN
596             salida := false;
597         END IF;
598
599         PRINTR(nombre_prueba, salida, salidaEsperada);
600
601         EXCEPTION
602         WHEN OTHERS THEN
603             PRINTR(nombre_prueba, false, salidaEsperada);
604             ROLLBACK;
605     END eliminar;
606 END PRUEBAS_factura;

```

```

606 /
608 CREATE OR REPLACE PACKAGE BODY PRUEBA_STOCK AS
610     PROCEDURE inicializar AS
611     BEGIN
612         DELETE FROM stock;
613     END inicializar;
614
615     PROCEDURE insertar
616     (nombre_prueba VARCHAR2, i_id_emplazamiento INT, i_id_producto INT,
617      i_cantidad INT, salidaEsperada BOOLEAN) AS
618     salida BOOLEAN := true;
619     actual stock%ROWTYPE;
620     BEGIN
621         INSERT INTO stock VALUES(i_id_emplazamiento, i_id_producto, i_cantidad);
622
623         SELECT * INTO actual FROM stock WHERE ID_Emplazamiento = i_id_emplazamiento and ID_Producto =
624         i_id_producto;
625
626         IF (actual.id_emplazamiento<>i_id_emplazamiento) OR (actual.id_producto<>i_id_producto) OR (
627         actual.cantidad<>i_cantidad) THEN
628             salida := false;
629         END IF;
630
631         PRINTR(nombre_prueba, salida, salidaEsperada);
632
633     EXCEPTION
634     WHEN OTHERS THEN
635         PRINTR(nombre_prueba, false, salidaEsperada);
636         ROLLBACK;
637     END insertar;
638
639     PROCEDURE actualizar
640     (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_id_producto INT,
641      a_cantidad INT, salidaEsperada BOOLEAN) AS
642     salida BOOLEAN := true;
643     actual stock%ROWTYPE;
644     BEGIN
645         UPDATE stock SET CANTIDAD = a_cantidad where ID_Emplazamiento = a_id_emplazamiento and
646         ID_Producto = a_id_producto;
647
648         SELECT * INTO actual FROM stock WHERE ID_Emplazamiento = a_id_emplazamiento and
649         ID_Producto = a_id_producto;
650         IF (actual.cantidad<>a_cantidad) THEN
651             salida := false;
652         END IF;
653
654         PRINTR(nombre_prueba, salida, salidaEsperada);
655
656     EXCEPTION
657     WHEN OTHERS THEN
658         PRINTR(nombre_prueba, false, salidaEsperada);
659         ROLLBACK;
660     END actualizar;
661
662     PROCEDURE eliminar
663     (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT,
664      salidaEsperada BOOLEAN) AS
665     salida BOOLEAN := true;
666     cantidad INT;
667     BEGIN
668         DELETE FROM stock WHERE ID_Emplazamiento = e_id_emplazamiento and ID_Producto =
669         e_id_producto;
670
671         SELECT COUNT(*) INTO cantidad FROM stock WHERE ID_Emplazamiento = e_id_emplazamiento and
672         ID_Producto = e_id_producto;
673         IF (cantidad<>0) THEN
674             salida := false;
675         END IF;
676
677         PRINTR(nombre_prueba, salida, salidaEsperada);
678
679     EXCEPTION
680     WHEN OTHERS THEN
681         PRINTR(nombre_prueba, false, salidaEsperada);
682         ROLLBACK;
683     END eliminar;

```

```

678 PROCEDURE stock_correcto
    (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
679 INT, salidaEsperada BOOLEAN) AS
680     salida BOOLEAN := true;
681     stock_nuevo int;
682 BEGIN
683     select cantidad into stock_nuevo from stock where id_emplazamiento = e_id_emplazamiento and
684     id_producto= e_id_producto;
685     if (stock_previo-cantidad)<> stock_nuevo then
686         salida := false;
687     END IF;
688
689     PRINTR(nombre_prueba, salida, salidaEsperada);
690
691     EXCEPTION
692     WHEN OTHERS THEN
693         PRINTR(nombre_prueba, false, salidaEsperada);
694         ROLLBACK;
695
696 END stock_correcto;
697
698 PROCEDURE stock_correcto_p
    (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
699 INT, salidaEsperada BOOLEAN) AS
700     salida BOOLEAN := true;
701     stock_nuevo int;
702 BEGIN
703     select cantidad into stock_nuevo from stock where id_emplazamiento = e_id_emplazamiento and
704     id_producto= e_id_producto;
705     if (stock_previo+cantidad)<> stock_nuevo then
706         salida := false;
707     END IF;
708
709     PRINTR(nombre_prueba, salida, salidaEsperada);
710
711     EXCEPTION
712     WHEN OTHERS THEN
713         PRINTR(nombre_prueba, false, salidaEsperada);
714         ROLLBACK;
715 end stock_correcto_p;
716
717 PROCEDURE stock_correcto_t
    (nombre_prueba VARCHAR2, e_id_emplazamiento_envia int, e_id_emplazamiento_recibe int,
718 stock_previo_envia int, stock_previo_recibe int,
719 cantidad int, e_id_producto int, salidaEsperada BOOLEAN) AS
720     salida BOOLEAN := true;
721     stock_nuevo_envia int;
722     stock_nuevo_recibe int;
723 BEGIN
724     select cantidad into stock_nuevo_envia from stock where id_emplazamiento =
725     e_id_emplazamiento_envia and id_producto = e_id_producto;
726     select cantidad into stock_nuevo_recibe from stock where id_emplazamiento =
727     e_id_emplazamiento_recibe and id_producto = e_id_producto;
728     if (stock_previo_envia-cantidad)<>stock_nuevo_envia or (stock_previo_recibe+cantidad)<>
729     stock_nuevo_recibe then
730         salida := false;
731     END IF;
732     PRINTR(nombre_prueba, salida, salidaEsperada);
733
734     EXCEPTION
735     WHEN OTHERS THEN
736         PRINTR(nombre_prueba, false, salidaEsperada);
737         ROLLBACK;
738 end stock_correcto_t;
739
740 END PRUEBA_STOCK;
741
742 /
743
744 CREATE OR REPLACE PACKAGE BODY PRUEBAS_VENTA AS
745
746     PROCEDURE inicializar AS
747     BEGIN
748         DELETE FROM VENTA;
749     END inicializar;

```



```

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

```

```

822 salida BOOLEAN := true;
823 actual_pedido %ROWTYPE;
824 w_ID_pedido INT;
825 BEGIN
826     INSERT INTO pedido VALUES(null, i_fecha, i_precio, i_id_emplazamiento, i_cif);
827
828     w_ID_pedido := S_ID_Pedido.currval;
829     SELECT * INTO actual FROM pedido WHERE ID_Pedido = w_ID_pedido;
830
831     IF (actual.fechaPedido<>i_fecha) OR (actual.precioTotal<>i_precio) OR (actual.
832 id_emplazamiento<>i_id_emplazamiento) OR (actual.cif<>i_cif) THEN
833         salida := false;
834     END IF;
835
836     PRINTR(nombre_prueba, salida, salidaEsperada);
837
838     EXCEPTION
839     WHEN OTHERS THEN
840         PRINTR(nombre_prueba, false, salidaEsperada);
841         ROLLBACK;
842 END insertar;
843
844 PROCEDURE eliminar
845     (nombre_prueba VARCHAR2, e_id_pedido INT, salidaEsperada BOOLEAN) AS
846     salida BOOLEAN := true;
847     cantidad INT;
848 BEGIN
849     DELETE FROM pedido WHERE ID_Pedido = e_id_pedido;
850
851     SELECT COUNT(*) INTO cantidad FROM pedido WHERE ID_Pedido = e_id_pedido;
852     IF (cantidad<>0) THEN
853         salida := false;
854     END IF;
855
856     PRINTR(nombre_prueba, salida, salidaEsperada);
857
858     EXCEPTION
859     WHEN OTHERS THEN
860         PRINTR(nombre_prueba, false, salidaEsperada);
861         ROLLBACK;
862 END eliminar;
863 END PRUEBAS_PEDIDO;
864
865 /
866
867 CREATE OR REPLACE PACKAGE BODY PRUEBAS_TRASPASO AS
868
869     PROCEDURE inicializar AS
870     BEGIN
871         DELETE FROM traspaso;
872     END inicializar;
873
874     PROCEDURE insertar
875     (nombre_prueba VARCHAR2, i_fechaTraspaso DATE, i_id_emplazamientoSalida INT,
876 i_id_emplazamientoEntrada INT, salidaEsperada BOOLEAN) AS
877     salida BOOLEAN := true;
878     actual_traspaso %ROWTYPE;
879     w_ID_traspaso INT;
880 BEGIN
881     INSERT INTO traspaso VALUES(null, i_fechaTraspaso, i_id_emplazamientoSalida,
882 i_id_emplazamientoEntrada);
883
884     w_ID_traspaso := S_ID_traspaso.currval;
885     SELECT * INTO actual FROM traspaso WHERE ID_traspaso = w_ID_traspaso;
886
887     IF (actual.fechaTraspaso<>i_fechaTraspaso) OR (actual.id_emplazamientoSalida<>
888 i_id_emplazamientoSalida) OR (actual.id_emplazamientoEntrada<>i_id_emplazamientoEntrada) THEN
889         salida := false;
890     END IF;
891
892     PRINTR(nombre_prueba, salida, salidaEsperada);
893
894     EXCEPTION
895     WHEN OTHERS THEN
896         PRINTR(nombre_prueba, false, salidaEsperada);
897         ROLLBACK;
898 END insertar;

```

```

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

```

```

970 END PRUEBAS_Solicitud_Traspaso;
971 /
972 CREATE OR REPLACE PACKAGE BODY PRUEBAS_Albaran AS
973
974     PROCEDURE inicializar AS
975     BEGIN
976         DELETE FROM Albaran;
977     END inicializar;
978
979     PROCEDURE insertar
980     (nombre_prueba VARCHAR2, i_fecha DATE,
981      i_precio NUMBER, i_id_pedido INT, salidaEsperada BOOLEAN) AS
982     salida BOOLEAN := true;
983     actual Albaran%ROWTYPE;
984     w_ID_Albaran INT;
985     BEGIN
986         INSERT INTO Albaran VALUES(null, i_fecha, i_precio, i_id_pedido);
987
988         w_ID_Albaran := S_ID_Albaran.currval;
989         SELECT * INTO actual FROM Albaran WHERE ID_Albaran = w_ID_Albaran;
990
991         IF (actual.fechaFirma<>i_fecha) OR (actual.id_pedido<>i_id_pedido) THEN
992             salida := false;
993         END IF;
994
995         PRINTR(nombre_prueba, salida, salidaEsperada);
996
997         EXCEPTION
998         WHEN OTHERS THEN
999             PRINTR(nombre_prueba, false, salidaEsperada);
1000             ROLLBACK;
1001     END insertar;
1002
1003     PROCEDURE eliminar
1004     (nombre_prueba VARCHAR2, e_id_albaran INT, salidaEsperada BOOLEAN) AS
1005     salida BOOLEAN := true;
1006     cantidad INT;
1007     BEGIN
1008         DELETE FROM Albaran WHERE ID_Albaran = e_id_albaran;
1009
1010         SELECT COUNT(*) INTO cantidad FROM Albaran WHERE ID_Albaran = e_id_albaran;
1011         IF (cantidad<>0) THEN
1012             salida := false;
1013         END IF;
1014
1015         PRINTR(nombre_prueba, salida, salidaEsperada);
1016
1017         EXCEPTION
1018         WHEN OTHERS THEN
1019             PRINTR(nombre_prueba, false, salidaEsperada);
1020             ROLLBACK;
1021     END eliminar;
1022 END PRUEBAS_Albaran;
1023 /
1024
1025 CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_PRODUCTO_TRASPASO AS
1026
1027     PROCEDURE inicializar AS
1028     BEGIN
1029         DELETE FROM asociacion_Producto_traspaso;
1030     END inicializar;
1031
1032     PROCEDURE insertar
1033     (nombre_prueba VARCHAR2, i_id_producto INT, i_id_Traspaso INT,
1034      i_cantidad NUMBER, salidaEsperada BOOLEAN) AS
1035     salida BOOLEAN := true;
1036     actual asociacion_Producto_traspaso%ROWTYPE;
1037     BEGIN
1038         INSERT INTO asociacion_Producto_traspaso VALUES(i_id_Traspaso, i_id_producto, i_cantidad);
1039
1040         SELECT * INTO actual FROM asociacion_Producto_traspaso WHERE ID_Producto = i_id_producto and
1041         ID_Traspaso = i_id_Traspaso;

```

```

1044         IF (actual.id_producto<>i_id_producto) OR (actual.id_traspaso<>i_id_traspaso) OR (actual.
cantidad<>i_cantidad) THEN
1046             salida := false;
1048         END IF;

1048         PRINTR(nombre_prueba, salida, salidaEsperada);

1050         EXCEPTION
1052         WHEN OTHERS THEN
            PRINTR(nombre_prueba, false, salidaEsperada);
            ROLLBACK;
1054     END insertar;

1056     PROCEDURE eliminar
(nombre_prueba VARCHAR2, e_id_traspaso INT, e_id_producto INT, salidaEsperada BOOLEAN) AS
1058         salida BOOLEAN := true;
1060         cantidad INT;
        BEGIN
            DELETE FROM asociacion_producto_traspaso WHERE ID_producto = e_id_producto and
1062             ID_traspaso = e_id_traspaso;

            SELECT COUNT(*) INTO cantidad FROM asociacion_producto_traspaso WHERE ID_producto =
1064             e_id_producto and ID_traspaso = e_id_traspaso;
            IF (cantidad<>0) THEN
                salida := false;
1066             END IF;

            PRINTR(nombre_prueba, salida, salidaEsperada);

1068         EXCEPTION
1070         WHEN OTHERS THEN
            PRINTR(nombre_prueba, false, salidaEsperada);
            ROLLBACK;
1072     END eliminar;
1074 END PRUEBAS_A_PRODUCTO_TRASPASO;
1076
1078 /
1080 CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_PRODUCTO_SOLICITUD AS
1082     PROCEDURE inicializar AS
        BEGIN
            DELETE FROM asociacion_producto_solicitud;
1084     END inicializar;

1086     PROCEDURE insertar
(nombre_prueba VARCHAR2, i_id_producto INT, i_id_solicitud INT,
1088         i_cantidad NUMBER, salidaEsperada BOOLEAN) AS
        salida BOOLEAN := true;
1090         actual asociacion_producto_solicitud%ROWTYPE;
        BEGIN
            INSERT INTO asociacion_producto_solicitud VALUES(i_id_solicitud, i_id_producto, i_cantidad);
1092
            SELECT * INTO actual FROM asociacion_producto_solicitud WHERE ID_producto = i_id_producto and
1094             ID_solicitud = i_id_solicitud;

            IF (actual.id_producto<>i_id_producto) OR (actual.id_solicitud<>i_id_solicitud) OR (actual.
1096             cantidad<>i_cantidad) THEN
                salida := false;
1098             END IF;

            PRINTR(nombre_prueba, salida, salidaEsperada);

1100         EXCEPTION
1102         WHEN OTHERS THEN
            PRINTR(nombre_prueba, false, salidaEsperada);
            ROLLBACK;
1104     END insertar;
1106
1108     PROCEDURE eliminar
(nombre_prueba VARCHAR2, e_id_solicitud INT, e_id_producto INT, salidaEsperada BOOLEAN) AS
1110         salida BOOLEAN := true;
1112         cantidad INT;
        BEGIN
            DELETE FROM asociacion_producto_solicitud WHERE ID_producto = e_id_producto and
1114             ID_solicitud = e_id_solicitud;

```

```

1116         SELECT COUNT(*) INTO cantidad FROM asociacion_Producto_Solicitud WHERE ID_Producto =
1117         e_id_producto and ID_Solicitud = e_id_Solicitud;
1118         IF (cantidad<>0) THEN
1119             salida := false;
1120         END IF;

1120     PRINTR(nombre_prueba, salida, salidaEsperada);

1122     EXCEPTION
1123     WHEN OTHERS THEN
1124         PRINTR(nombre_prueba, false, salidaEsperada);
1125         ROLLBACK;
1126     END eliminar;
1127 END PRUEBAS_A_PRODUCTO_SOLICITUD;
1128
1129 /
1130
1131 CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_PEDIDO_PRODUCTO AS
1132
1133     PROCEDURE inicializar AS
1134     BEGIN
1135         DELETE FROM asociacion_Pedido_Producto;
1136     END inicializar;

1138     PROCEDURE insertar
1139     (nombre_prueba VARCHAR2, i_id_pedido INT, i_id_producto INT,
1140     i_cantidad NUMBER, i_precioCompra INT, i_iva INT, i_precioLinea INT,
1141     salidaEsperada BOOLEAN) AS
1142     salida BOOLEAN := true;
1143     actual asociacion_Pedido_Producto%ROWTYPE;
1144     BEGIN
1145         INSERT INTO asociacion_Pedido_Producto VALUES(i_id_pedido, i_id_producto, i_cantidad,
1146         i_precioCompra, i_iva, i_precioLinea);

1148         SELECT * INTO actual FROM asociacion_Pedido_Producto WHERE ID_pedido = i_id_pedido and
1149         ID_producto = i_id_producto;

1150         IF (actual.id_pedido<>i_id_pedido) OR (actual.id_producto<>i_id_producto) OR (actual.cantidad
1151         <>i_cantidad) OR (actual.precioCompra<>i_precioCompra) OR (actual.iva<>i_iva) THEN
1152             salida := false;
1153         END IF;

1154         PRINTR(nombre_prueba, salida, salidaEsperada);

1156     EXCEPTION
1157     WHEN OTHERS THEN
1158         PRINTR(nombre_prueba, false, salidaEsperada);
1159         ROLLBACK;
1160     END insertar;

1162     PROCEDURE eliminar
1163     (nombre_prueba VARCHAR2, e_id_pedido INT, e_id_producto INT,
1164     salidaEsperada BOOLEAN) AS
1165     salida BOOLEAN := true;
1166     cantidad INT;
1167     BEGIN
1168         DELETE FROM asociacion_Pedido_Producto WHERE ID_pedido = e_id_pedido and ID_producto =
1169         e_id_producto;

1170         SELECT COUNT(*) INTO cantidad FROM asociacion_Pedido_Producto WHERE ID_pedido =
1171         e_id_pedido and ID_producto = e_id_producto;
1172         IF (cantidad<>0) THEN
1173             salida := false;
1174         END IF;

1175         PRINTR(nombre_prueba, salida, salidaEsperada);

1177     EXCEPTION
1178     WHEN OTHERS THEN
1179         PRINTR(nombre_prueba, false, salidaEsperada);
1180         ROLLBACK;
1181     END eliminar;
1182 END PRUEBAS_A_PEDIDO_PRODUCTO;
1183
1184 /
1185
1186 CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_VENTA_PRODUCTO AS

```

```

1186 PROCEDURE inicializar AS
1187 BEGIN
1188     DELETE FROM asociacion_Venta_Producto;
1189 END inicializar;

1192 PROCEDURE insertar
1193 (nombre_prueba VARCHAR2, i_id_venta INT, i_id_producto INT,
1194  i_cantidad NUMBER, i_precioVenta NUMBER, i_ivaVenta NUMBER,
1195  i_precioLinea NUMBER, salidaEsperada BOOLEAN) AS
1196 salida BOOLEAN := true;
1197 actual asociacion_Venta_Producto %ROWTYPE;
1198 BEGIN
1199     INSERT INTO asociacion_Venta_Producto VALUES(i_id_venta, i_id_producto, i_cantidad,
1200     i_precioVenta, i_ivaVenta, i_precioLinea);

1201     SELECT * INTO actual FROM asociacion_Venta_Producto WHERE ID_Venta = i_id_venta and
1202     ID_producto = i_id_producto;

1203     IF (actual.id_venta<>i_id_venta) OR (actual.id_producto<>i_id_producto) OR (actual.cantidad<>
1204     i_cantidad) OR (actual.precioVenta<>i_precioVenta) OR (actual.ivaVenta<>i_ivaVenta) OR (actual.
1205     precioLinea<>i_precioLinea) THEN
1206         salida := false;
1207     END IF;

1208     PRINTR(nombre_prueba, salida, salidaEsperada);

1209     EXCEPTION
1210     WHEN OTHERS THEN
1211         PRINTR(nombre_prueba, false, salidaEsperada);
1212         ROLLBACK;
1213 END insertar;

1214 PROCEDURE eliminar
1215 (nombre_prueba VARCHAR2, e_id_venta INT, e_id_producto INT, salidaEsperada BOOLEAN) AS
1216 salida BOOLEAN := true;
1217 cantidad INT;
1218 BEGIN
1219     DELETE FROM asociacion_Venta_Producto WHERE ID_Venta = e_id_venta and ID_producto =
1220     e_id_producto;

1221     SELECT COUNT(*) INTO cantidad FROM asociacion_Venta_Producto WHERE ID_Venta = e_id_venta
1222     and ID_producto = e_id_producto;
1223     IF (cantidad<>0) THEN
1224         salida := false;
1225     END IF;

1226     PRINTR(nombre_prueba, salida, salidaEsperada);

1227     EXCEPTION
1228     WHEN OTHERS THEN
1229         PRINTR(nombre_prueba, false, salidaEsperada);
1230         ROLLBACK;
1231 END eliminar;
1232 END PRUEBAS_A_VENTA_PRODUCTO;

1233 /

1234 DROP SEQUENCE S_ID_Producto;
1235 DROP SEQUENCE S_ID_Traspaso;
1236 DROP SEQUENCE S_ID_Solicitud;
1237 DROP SEQUENCE S_ID_Pedido;
1238 DROP SEQUENCE S_ID_Albaran;
1239 DROP SEQUENCE S_ID_Factura;
1240 DROP SEQUENCE S_ID_Emplazamiento;
1241 DROP SEQUENCE S_ID_Venta;

1242 CREATE SEQUENCE S_ID_Emplazamiento START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1243 CREATE SEQUENCE S_ID_Venta START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1244 CREATE SEQUENCE S_ID_Pedido START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1245 CREATE SEQUENCE S_ID_Albaran START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1246 CREATE SEQUENCE S_ID_Producto START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1247 CREATE SEQUENCE S_ID_Traspaso START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1248 CREATE SEQUENCE S_ID_Solicitud START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1249 CREATE SEQUENCE S_ID_Factura START WITH 1 INCREMENT BY 1 MAXVALUE 9999;

1250 /* EXECUTE DE PRUEBAS */

```

```

-- Inicializacion
1258 EXECUTE pruebas_albaran.inicializar();
EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.inicializar();
1260 EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.inicializar();
EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.inicializar();
1262 EXECUTE PRUEBAS_A_VENTA_PRODUCTO.inicializar();
EXECUTE PRUEBAS_FACTURA.inicializar();
1264 EXECUTE pruebas_venta.inicializar();
EXECUTE pruebas_pedido.inicializar();
1266 EXECUTE pruebas_proveedor.inicializar();
EXECUTE PRUEBAS_STOCK.inicializar();
1268 EXECUTE pruebas_producto.inicializar();
EXECUTE PRUEBAS_TRASPASO.inicializar();
1270 EXECUTE PRUEBAS_SOLICITUD_TRASPASO.inicializar();
EXECUTE PRUEBAS_SOCIO.inicializar();
1272 EXECUTE pruebas_emplazamiento.inicializar();
--SOCIO
1274 EXECUTE PRUEBAS_SOCIO.inicializar();
EXECUTE PRUEBAS_SOCIO.insertar('Socio Insertar', 'Daniel', 'Gonzalez', '54148787G', 'Avenida del
pepinillo', TO_DATE('10102015', 'DDMMYYYY'), 'dani@us.es', true);
1276 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);
1278 EXECUTE PRUEBAS_SOCIO.eliminar('Socio Eliminar', '54148787G', true);

1280 --PROVEEDOR
EXECUTE PRUEBAS_PROVEEDOR.inicializar();
1282 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);
1284 EXECUTE PRUEBAS_PROVEEDOR.actualizar('Proveedor Actualizar', 'B54120214', 'Piexs', 111111111, 'a@pi.
es', true);
EXECUTE PRUEBAS_PROVEEDOR.eliminar('Proveedor Eliminar', 'B54120214', true);

1286 --PRODUCTO
1288 EXECUTE PRUEBAS_PRODUCTO.inicializar();
EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion', 'Zapatos cutres', 'unos zaparos cutres pero
calentitos', 'Calzado', 12.5, 21, true);
1290 EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion', 'Cool shoes', 'Zapatos de cuero italiano', '
Calzado', 5.95, 21, true);
EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion', 'Sport Shoes', 'Zapatos de plastico', 'Calzado'
, 6.95, 21, true);
1292 EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion', 'Brown Sandals', 'Sandalias de playa marrones',
'Calzado', 7.95, 21, true);
EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion', 'Chupa de cuero', 'gran imitacion de la chupa
de Han Solo', 'Calzado', 500, 21, true);
1294 EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion', 'Chaleco de lana', 'por una gran diseñadora', '
Jerseys', 46.95, 21, true);
EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion', 'Sombrero de paja', 'el gran sombrero de
Mugivara', 'Accesorios', 20, 21, true);
1296 EXECUTE PRUEBAS_PRODUCTO.actualizar('Producto actualizacion', S_ID_Producto.currval-2, 'Unos zapatos
cutres pero calentitos, estan rotos', 10.5, 21, true);
EXECUTE PRUEBAS_PRODUCTO.eliminar('Producto elminar', S_ID_Producto.currval, true);

1298 --EMPLAZAMIENTO
1300 EXECUTE PRUEBAS_EMPLAZAMIENTO.inicializar();
EXECUTE PRUEBAS_EMPLAZAMIENTO.insertar('Emplazamiento insercion', 'Calle de la piruleta', 954147741, '
TIENDA', true);
1302 EXECUTE PRUEBAS_EMPLAZAMIENTO.insertar('Emplazamiento insercion', 'Avenida Reina Mercedes',
958632147, 'TIENDA', true);
EXECUTE PRUEBAS_EMPLAZAMIENTO.insertar('Emplazamiento insercion', 'Calle de la piruleta 24',
958632147, 'TIENDA', true);
1304 EXECUTE PRUEBAS_EMPLAZAMIENTO.insertar('Emplazamiento insercion', 'Calle de la piruleta 25',
958632666, 'ALMACEN', true);
EXECUTE PRUEBAS_EMPLAZAMIENTO.actualizar('Emplazamiento actualizacion', S_ID_Emplazamiento.currval-2, '
Avenida del pepinillo2', 954147871, true);
1306 EXECUTE PRUEBAS_EMPLAZAMIENTO.eliminar('Emplazamiento eliminar', S_ID_Emplazamiento.currval, true);

1308 --STOCK
EXECUTE PRUEBAS_STOCK.inicializar();
1310 EXECUTE PRUEBAS_STOCK.insertar('Stock insercion', S_ID_Emplazamiento.currval-2, S_ID_Producto.currval
-1, 10, true);
EXECUTE PRUEBAS_STOCK.insertar('Stock insercion', S_ID_Emplazamiento.currval-2, S_ID_Producto.currval
-2, 20, true);

```



```

1312 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);
1314 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);
1316 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);
1318
--VENTA
1320 EXECUTE pruebas_venta.inicializar();
EXECUTE pruebas_venta.insertar('Venta insercion',0,sysdate,'29613400A',true);
1322 EXECUTE pruebas_venta.insertar('Venta insercion',0,sysdate,null,true);
EXECUTE pruebas_venta.insertar('Venta insercion',0,sysdate,null,true);
1324 EXECUTE pruebas_venta.eliminar('Venta eliminar',S_ID_venta.currval,true);

1326 --ASOCIACION_VENTA_PRODUCTO
EXECUTE PRUEBAS_A_VENTA_PRODUCTO.inicializar();
1328 EXECUTE PRUEBAS_A_VENTA_PRODUCTO.insertar('Venta-Producto insercion',s_id_venta.currval-1,
      s_id_producto.currval-2,3,10.5,21,31.5,true);
EXECUTE PRUEBAS_A_venta_producto.insertar('Venta-Producto insercion',s_id_venta.currval-1,
      s_id_producto.currval-1,2,46.95,21,93.9,true);
1330 EXECUTE PRUEBAS_A_VENTA_PRODUCTO.insertar('Venta-Producto insercion',s_id_venta.currval-2,
      s_id_producto.currval-2,3,10.5,21,31.5,true);
EXECUTE PRUEBAS_A_venta_producto.insertar('Venta-Producto insercion',s_id_venta.currval-2,
      s_id_producto.currval-1,2,46.95,21,93.9,true);
1332
--FACTURA
1334 EXECUTE PRUEBAS_factura.inicializar();
EXECUTE PRUEBAS_factura.insertar('Factura insercion',119.13,sysdate,'F',s_id_venta.currval-2,
      S_ID_Emplazamiento.currval-1,true);
1336 EXECUTE PRUEBAS_factura.insertar('Factura insercion',125.4,sysdate,'F',s_id_venta.currval-1,
      S_ID_Emplazamiento.currval-2,true);
EXECUTE PRUEBAS_factura.actualizar('Factura actualizar',s_id_factura.currval,'T',true);
1338 EXECUTE PRUEBAS_venta.descuento('Descuento factura-venta',s_id_venta.currval-2,125.4,'29613400A',true
      );
EXECUTE PRUEBAS_venta.descuento('Descuento factura-venta',s_id_venta.currval-1,125.4,null,true);
1340 EXECUTE Prueba_stock.stock_correcto('Actualizacion de stock tras venta',s_id_emplazamiento.currval-1,
      s_id_producto.currval-2,22,3,true);
EXECUTE Prueba_stock.stock_correcto('Actualizacion de stock tras venta',s_id_emplazamiento.currval-1,
      s_id_producto.currval-2,21,3,false);
1342 EXECUTE Prueba_stock.stock_correcto('Actualizacion de stock tras venta',s_id_emplazamiento.currval-1,
      s_id_producto.currval-1,11,2,true);

1344 --PEDIDO
EXECUTE PRUEBAS_PEDIDO.inicializar();
1346 EXECUTE PRUEBAS_PEDIDO.insertar('Pedido insercion',sysdate,0,S_ID_Emplazamiento.currval-1,'B54129999'
      ,true);
EXECUTE PRUEBAS_PEDIDO.insertar('Pedido insercion',sysdate,0,S_ID_Emplazamiento.currval-2,'B54129999'
      ,true);
1348 EXECUTE PRUEBAS_PEDIDO.insertar('Pedido insercion',sysdate,0,S_ID_Emplazamiento.currval-2,'B54129999'
      ,true);
EXECUTE PRUEBAS_PEDIDO.eliminar('Pedido eliminar',S_ID_Pedido.currval,true);
1350
--PEDIDO_PRODUCTO
1352 EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.inicializar();
EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.insertar('PEDIDO_PRODUCTO insercion',S_ID_Pedido.currval-1,
      s_id_producto.currval-1,21,10.5,21,220.5,true);
1354 EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.insertar('PEDIDO_PRODUCTO insercion',S_ID_Pedido.currval-2,
      s_id_producto.currval-1,22,46.95,21,0,true);
EXECUTE PRUEBAS_A_PEDIDO_PRODUCTO.insertar('PEDIDO_PRODUCTO insercion',S_ID_Pedido.currval-1,
      s_id_producto.currval-2,22,46.95,21,0,true);
1356
--ALBARAN
1358 EXECUTE PRUEBAS_ALBARAN.inicializar();
1360 EXECUTE PRUEBAS_ALBARAN.insertar('Albaran insercion',sysdate,0,S_ID_Pedido.currval-1,true);
EXECUTE PRUEBA_STOCK.stock_correcto_p('Actualizacion de stock tras comprar',s_ID_emplazamiento.
      currval-2,s_ID_producto.currval-1,8,21,true);
1362 EXECUTE PRUEBA_STOCK.stock_correcto_p('Actualizacion de stock tras comprar',s_ID_emplazamiento.
      currval-2,s_ID_producto.currval-1,8,15,false);
EXECUTE PRUEBAS_ALBARAN.insertar('Albaran insercion',sysdate,0,S_ID_Pedido.currval-2,true);
1364 EXECUTE PRUEBAS_ALBARAN.eliminar('Albaran eliminacion',S_ID_Albaran.currval,true);

```

```

1366 --SOLICITUD_TRASPASO
EXECUTE PRUEBAS_SOLICITUD_TRASPASO.inicializar();
1368 EXECUTE PRUEBAS_SOLICITUD_TRASPASO.insertar('Solicitud traspaso Insertar', sysdate,
      S_ID_Emplazamiento.currval-1, S_ID_Emplazamiento.currval-2, true);
EXECUTE PRUEBAS_SOLICITUD_TRASPASO.eliminar('Solicitud traspaso Eliminar', S_ID_Solicitud.currval,
      true);
1370 EXECUTE PRUEBAS_SOLICITUD_TRASPASO.insertar('Solicitud traspaso Insertar', sysdate,
      S_ID_Emplazamiento.currval-1, S_ID_Emplazamiento.currval-2, true);
--ASOC_PRODUCTO_SOLICITUD
1372 EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.inicializar();
EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.insertar('A-Producto-Solicitud Insertar', S_ID_Producto.currval
      -1, S_ID_Solicitud.currval, 3, true);
1374 EXECUTE PRUEBAS_A_PRODUCTO_SOLICITUD.eliminar('A-Producto-Solicitud Eliminar', S_ID_Solicitud.currval
      , S_ID_Producto.currval-1, true);

1376 --TRASPASO
EXECUTE PRUEBAS_TRASPASO.inicializar();
1378 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);
1380 EXECUTE PRUEBAS_TRASPASO.eliminar('Traspaso eliminar',S_ID_traspaso.currval,true);

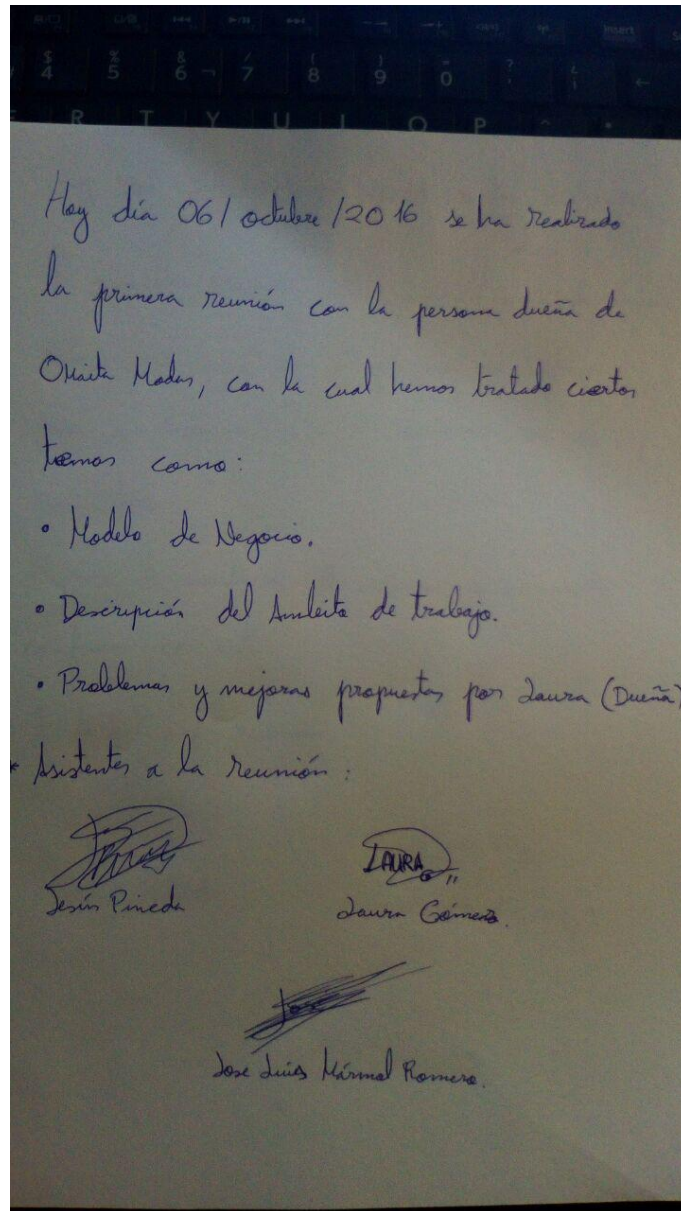
1382 --ASOC_PRODUCTO_TRASPASO
EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.inicializar();
1384 EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.insertar('A-Producto-Traspaso Insertar', S_ID_Producto.currval-1,
      S_ID_traspaso.currval-1, 4, true);
EXECUTE PRUEBA_STOCK.STOCK_CORRECTO_T('Actualizacion de stock tras traspaso',s_ID_emplazamiento.
      currval-1, s_ID_emplazamiento.currval-2,31,29,4,s_ID_producto.currval-1,true);
1386 EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.eliminar('A-Producto-Traspaso Actualizar', S_ID_traspaso.currval
      -1, S_ID_Producto.currval-1, true);

```

sql/pruebas.sql

## 11. Apéndice

### 11.1. Actas de reunión



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

Después de mostrarle el avance se encuentra conforme con este.

\* Asistentes a la reunión:

Jesús Pineda

Laura Gómez