

UNIVERSIDAD DE SEVILLA

IISSI

PROYECTO

Modas Omaita

Autores:

Daniel González Corzo

Jesús Pineda Márquez

José Luis Mármol Romero

Roberto Hueso Gómez

20 de junio de 2017



Escuela Técnica Superior de
Ingeniería Informática

Índice

1. Introducción	3
2. Glosario de Términos	3
3. Modelo de negocio	
3.1. Proceso de venta	
3.2. Proceso de compra al proveedor	
3.3. Proceso de disponibilidad	
3.4. Proceso de creación de socios	
4. Visión General del Sistema	
4.1. Descripción	
4.2. Historias de usuario	
5. Catálogo de requisitos	
5.1. Requisitos de información	
5.2. Requisitos funcionales	
5.3. Requisitos no funcionales	
5.4. Reglas de negocio	
6. Pruebas de aceptación	
7. Modelo Conceptual	
7.1. Diagramas de clases UML	
7.2. Escenarios de prueba	
8. Matrices de trazabilidad	
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	
9. Modelos relacionales	
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 Lucas nº20, la cual pertenece a una cadena de tiendas de ropa que se especializa en la venta de ropa y accesorios a un público maduro y femenino. Nuestro cliente busca ser una tienda puntera en su cadena gracias a que tiene a su disposición toda la atención del público de la localidad, ya que es la única de esta cadena en la misma. Actualmente nuestro cliente no pasa por la mejor situación en lo que a clientela se refiere, además de que carece de personal, por tanto, nuestro proyecto tiene como base ayudar a nuestro cliente en la gestión de la tienda con un sistema informático, el cual nos permita realizar pedidos a proveedores y visualizar productos en el stock de la otras tiendas de la cadena, y la creación de una página online donde sus clientes puedan visualizar y comprar un producto determinado en cualquier momento, con esto último se quiere conseguir ampliar la clientela de la tienda.



2. Glosario de Términos

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

3. Modelo de negocio

3.1. Proceso de venta

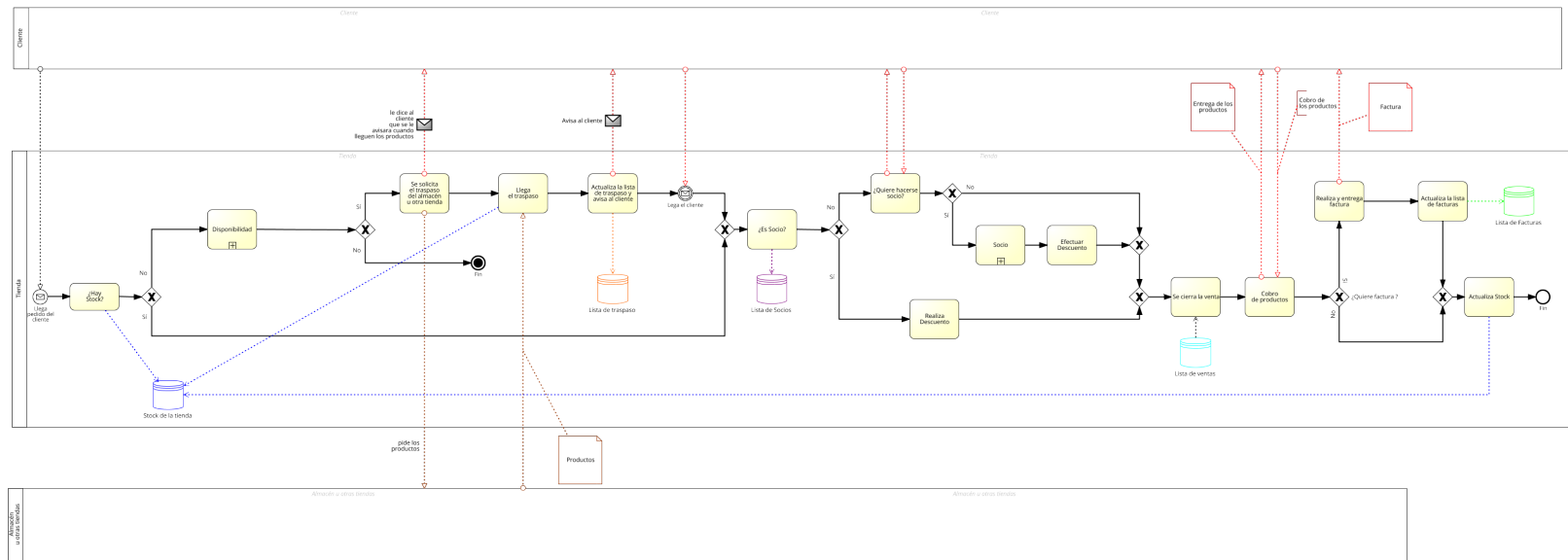
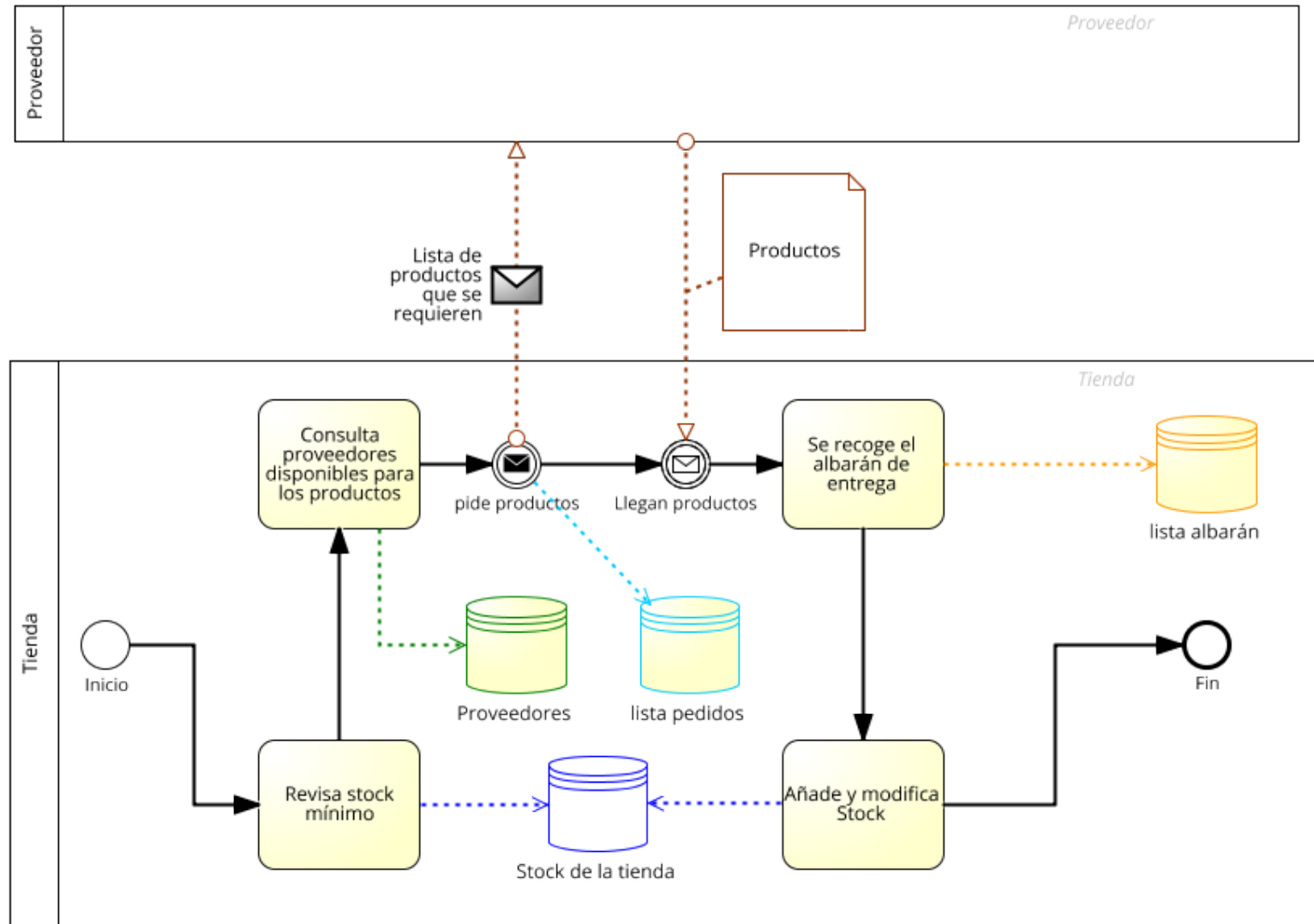
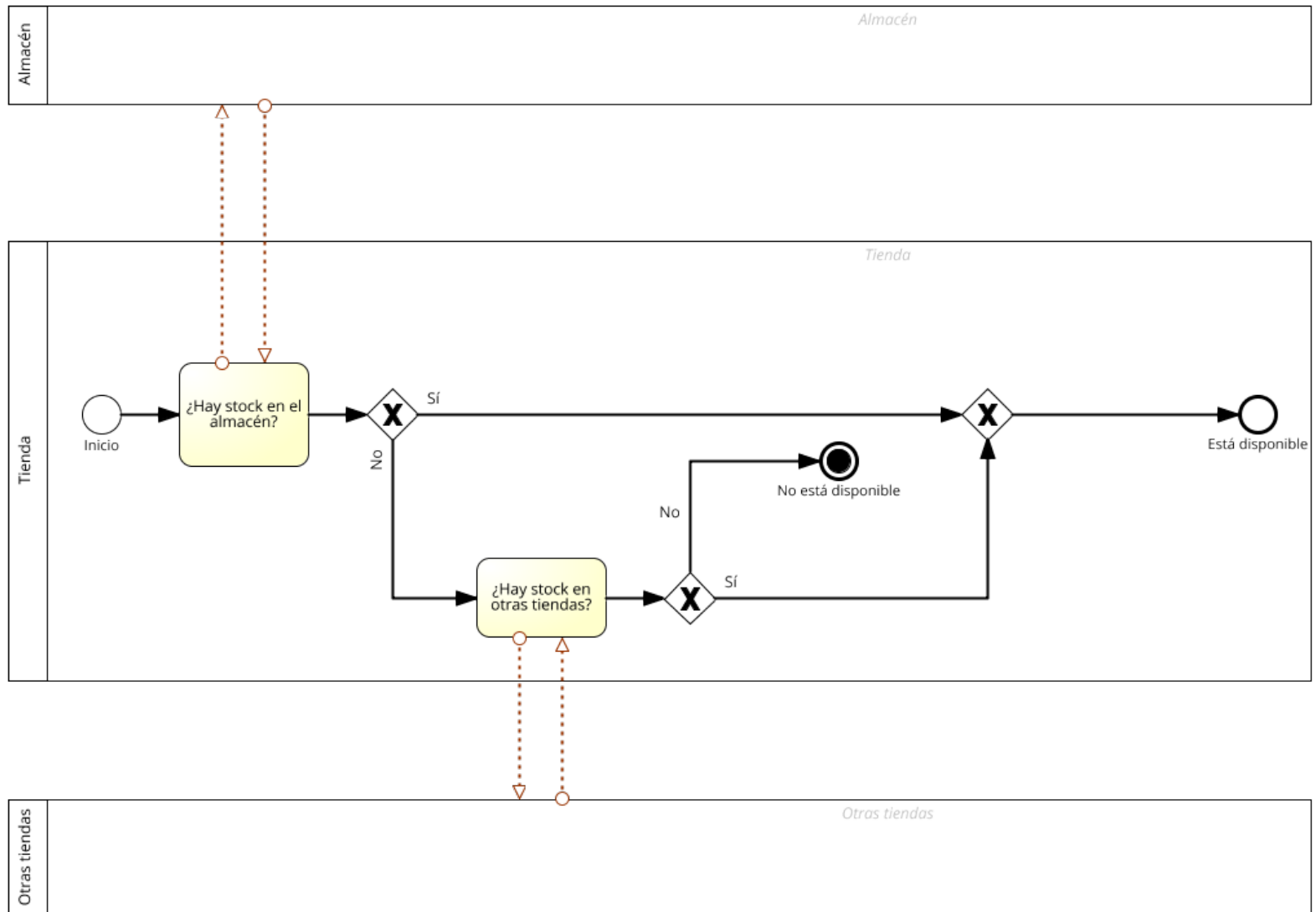


Figura 1: Proceso de venta

3.2. Proceso de compra al proveedor

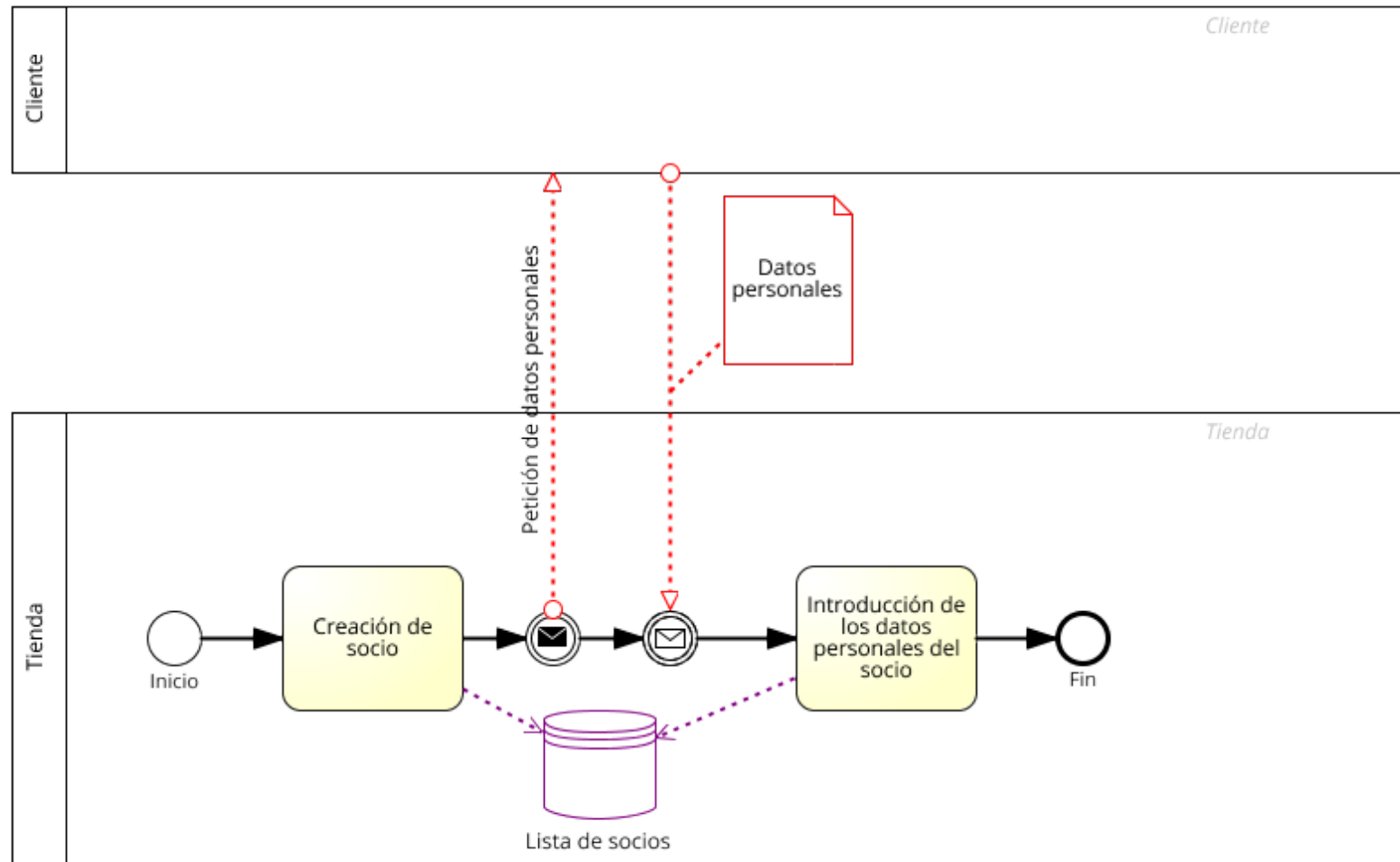


3.3. Proceso de disponibilidad



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

5. Catálogo de requisitos

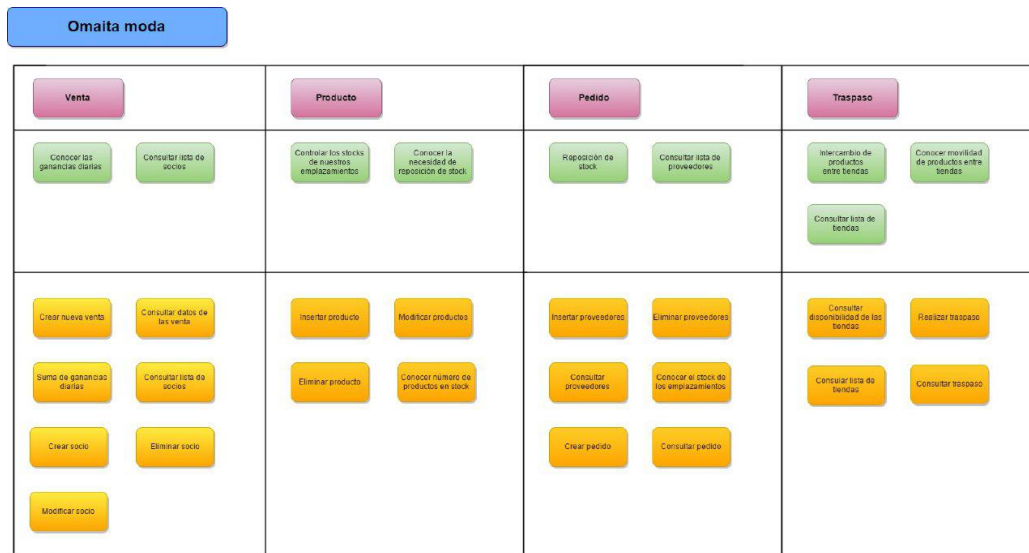


Figura 5: Mapa de requisitos

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

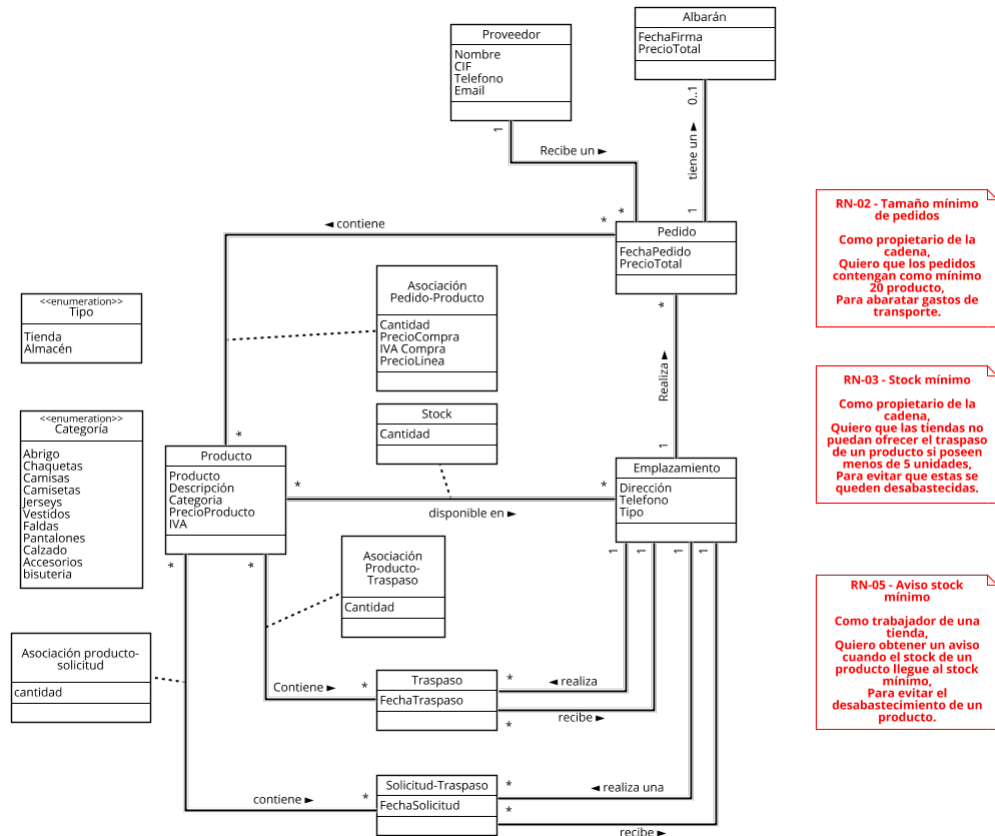


Figura 6: UML Traspaso / Pedido

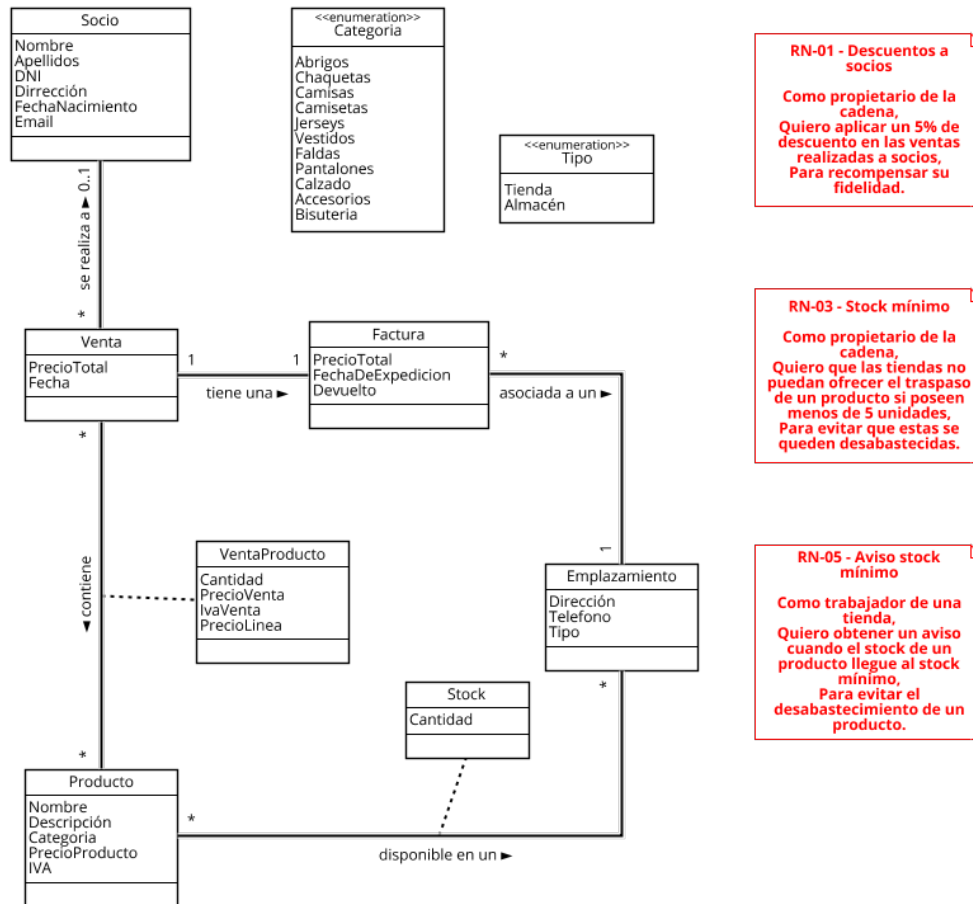


Figura 7: UML Venta

7.2. Escenarios de prueba

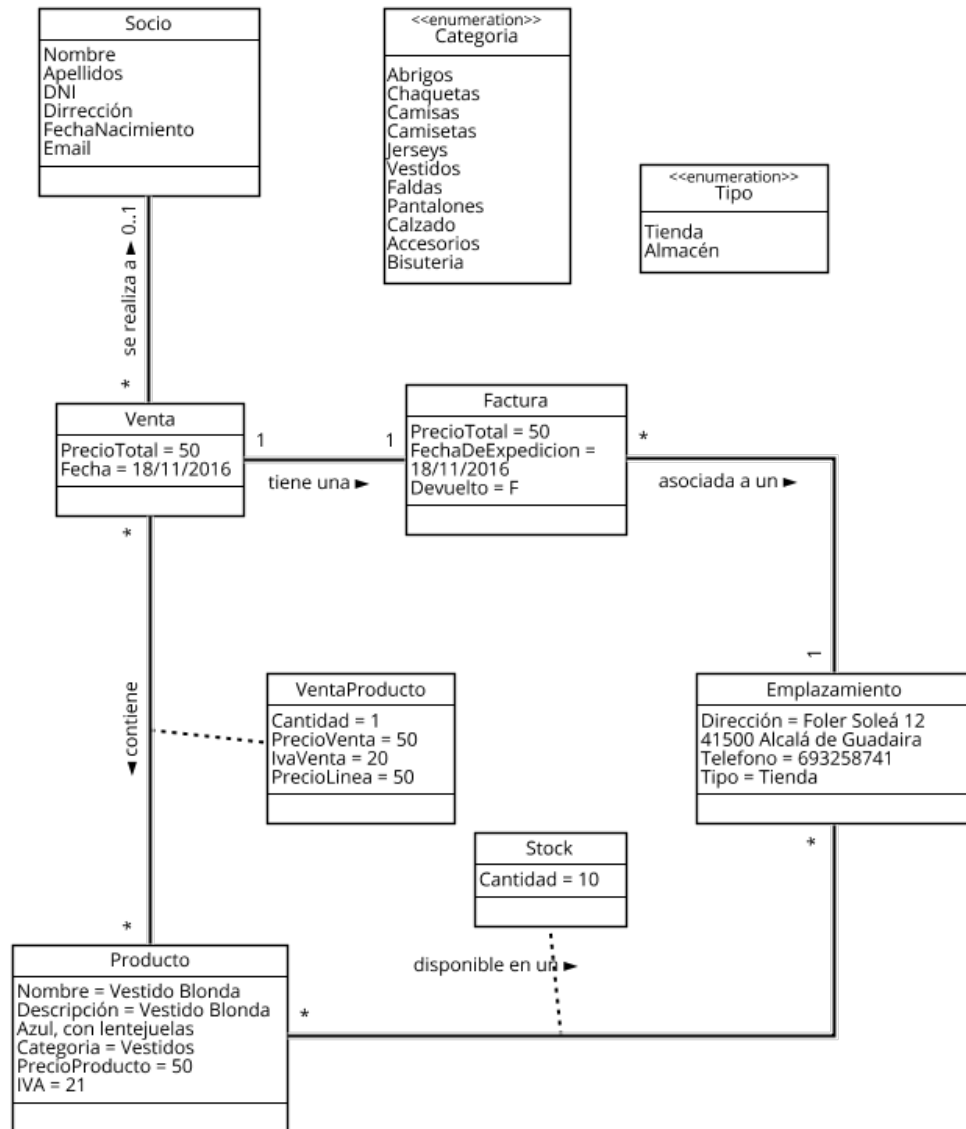


Figura 8: UML Venta NO Socio

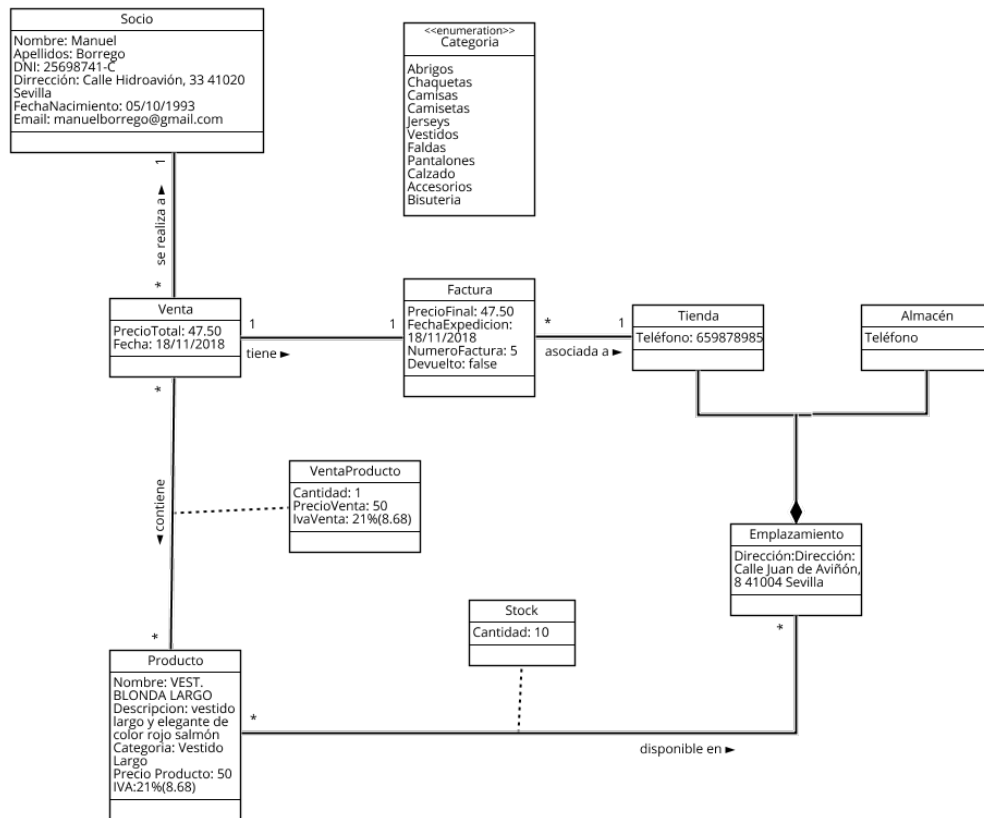


Figura 9: UML Venta Socio

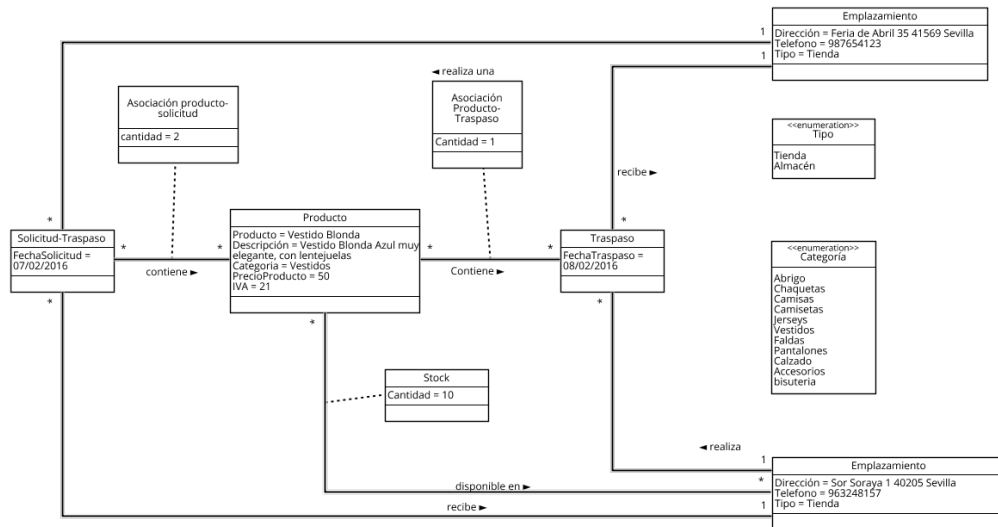


Figura 10: UML Traspaso Tienda - Tienda

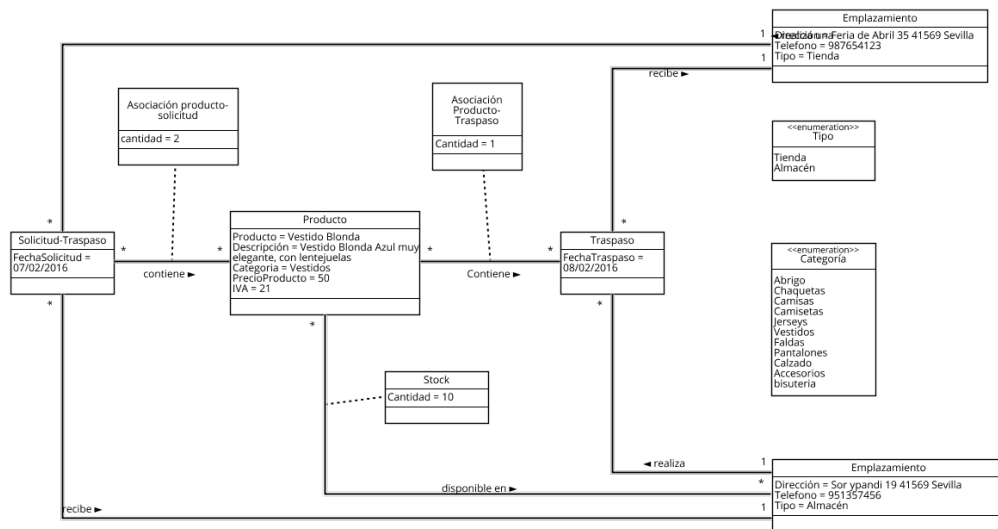


Figura 11: UML Traspaso Tienda - Almacén

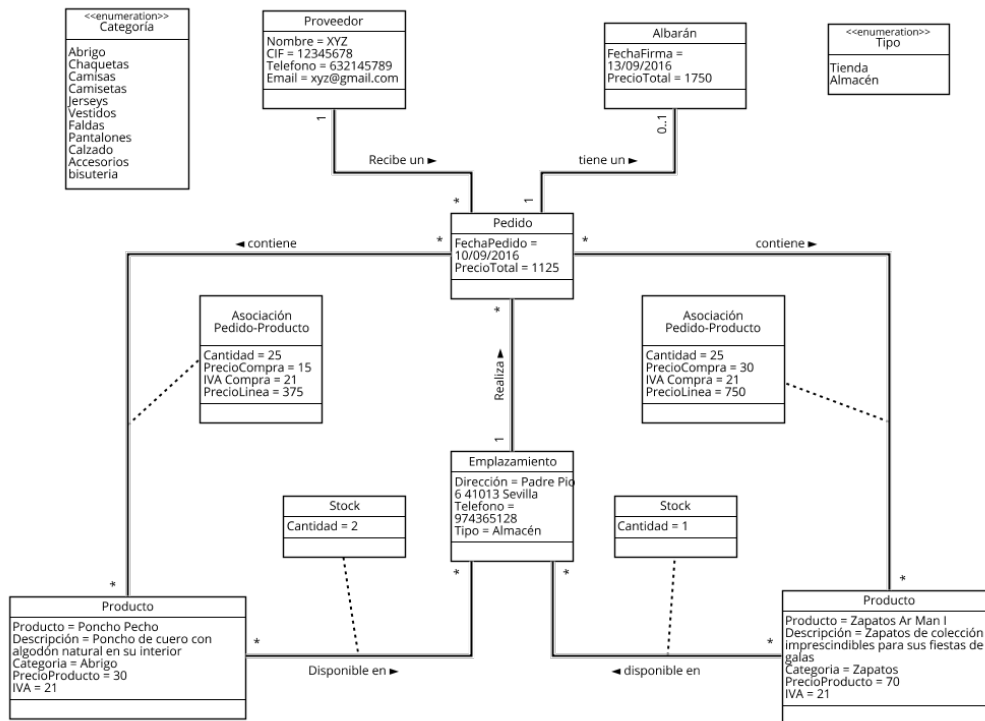


Figura 12: UML Pedido Tienda - Proveedor

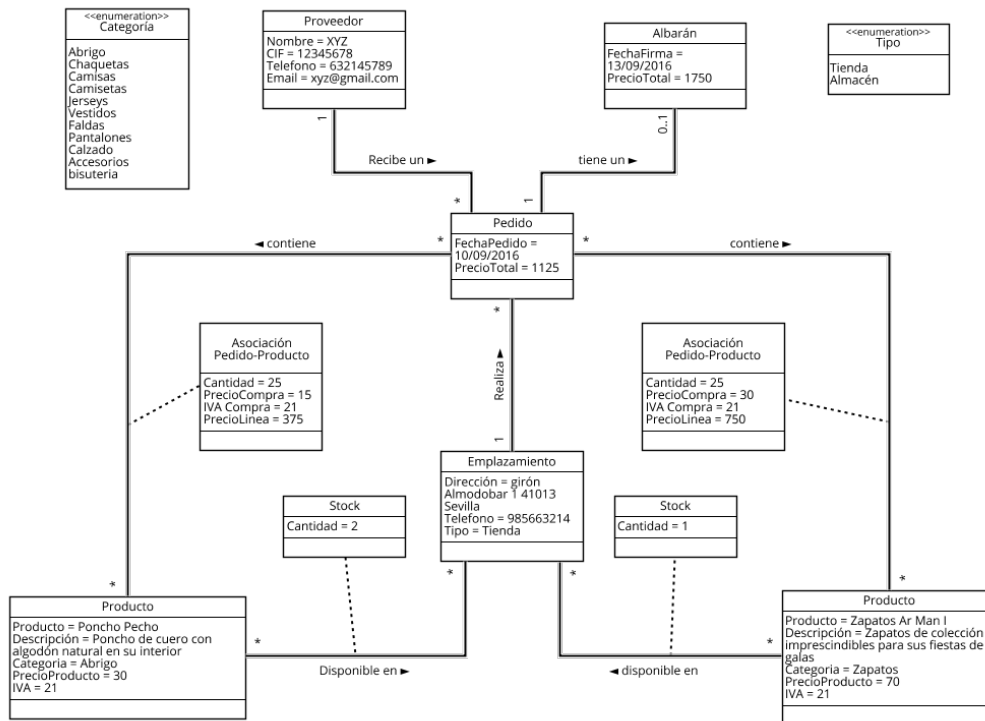


Figura 13: UML Pedido Almacén - Proveedor

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

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

8.2. Pruebas de aceptación frente a escenarios de prueba

Cuadro 4: Escenarios

Escenario 01				X		X	X		X		X
Escenario 02				X		X	X		X		X
Escenario 03				X	X			X	X		
Escenario 04				X	X			X	X		
Escenario 05	X	X		X	X						
Escenario 06	X	X	X	X	X					X	
	01	02	03	04	05	06	07	08	09	10	11

8.3. Tipos de UML frente a Requisitos

Cuadro 5: Requisitos de informacion

Venta	x	x	x	x	x	x					
Traspaso - Pedido				x	x	x	x	x	x	x	x
	01	02	03	04	05	06	07	08	09	10	11

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

label description

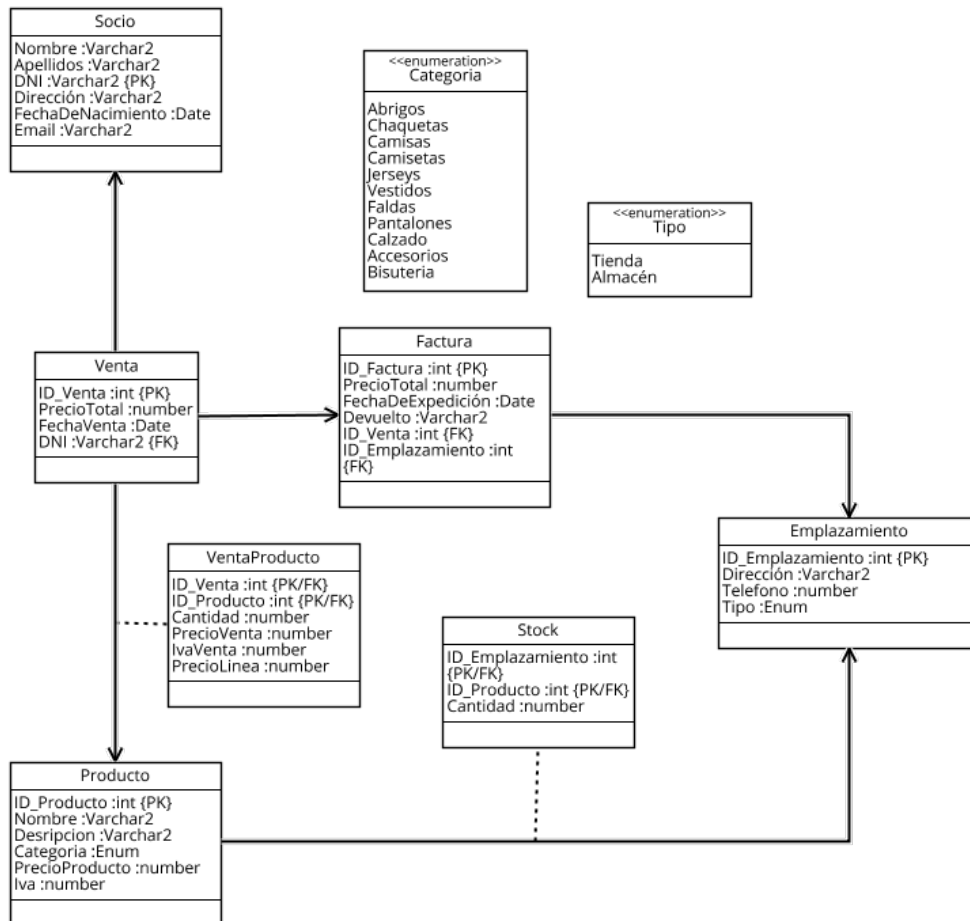


Figura 14: 3FN Venta

9.0.2. 3FN Traspaso - Pedido

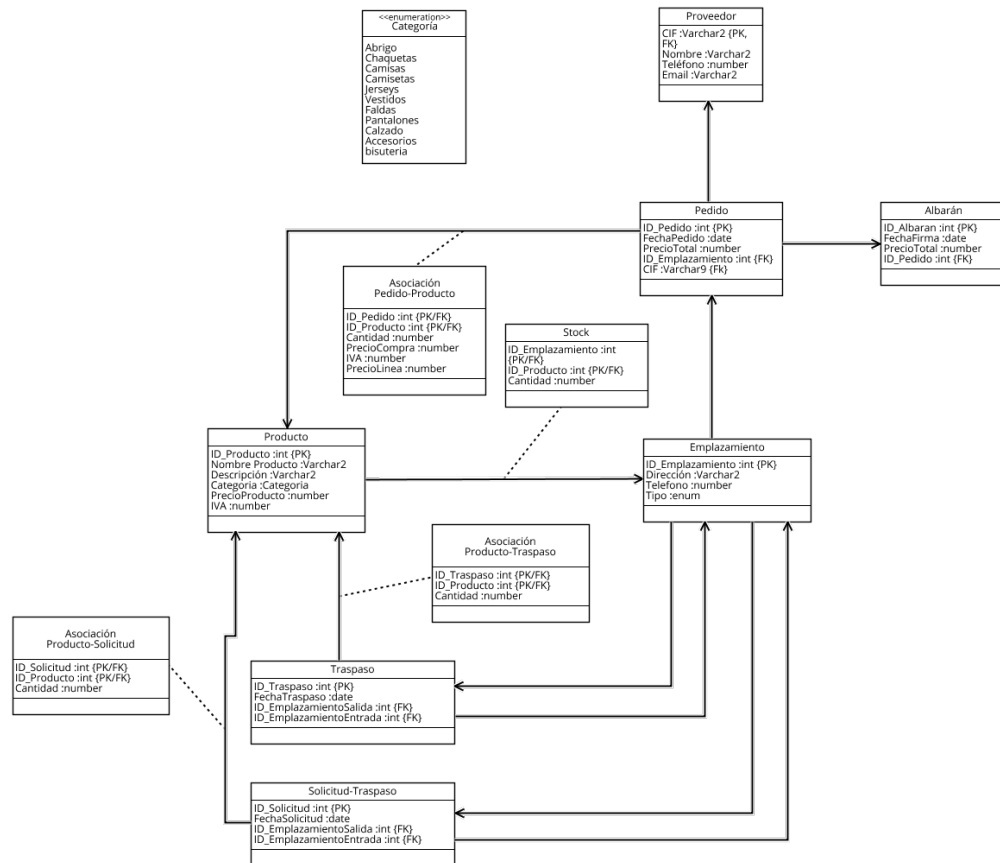


Figura 15: 3FN Traspaso - Pedido

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
240     BEGIN
241         SELECT S_ID_Emplazamiento.NEXTVAL
242         INTO :NEW.ID_Emplazamiento FROM DUAL;
243     END crea_ID_Emplazamiento;
244 /
245
246 CREATE OR REPLACE TRIGGER crea_ID_Venta
247 BEFORE INSERT ON Venta
248 FOR EACH ROW
249 BEGIN
250     SELECT S_ID_Venta.NEXTVAL
251     INTO :NEW.ID_Venta FROM DUAL;
252 END crea_ID_Venta;
253 /

```

sql/tablas.sql

i

10.2. Funciones y Procedures

```

1 CREATE OR REPLACE PROCEDURE Socio_Nuevo (
2     p_Nombre IN SOCIO.Nombre%TYPE,
3     p_Apellidos IN SOCIO.Apellidos%TYPE,
4     p_Direccion IN SOCIO.Direccion%TYPE,
5     p_FechaDeNacimiento IN SOCIO.FechaDeNacimiento%TYPE,
6     p_Email IN SOCIO.Email%TYPE,
7     p_DNI IN SOCIO.DNI%TYPE)
8 IS BEGIN
9     INSERT INTO SOCIO
10     VALUES (p_Nombre, p_Apellidos, p_Direccion, p_FechaDeNacimiento,
11             p_Email,p_DNI);
12 END Socio_Nuevo;
13 /
14
15 CREATE OR REPLACE PROCEDURE Producto_Nuevo(
16     P_ID_Producto IN PRODUCTO.ID_Producto%TYPE,
17     P_Nombre IN PRODUCTO.Nombre%TYPE,
18     P_Descripcion IN PRODUCTO.Descripcion%TYPE,
19     P_Categoria IN PRODUCTO.Categoria%TYPE,
20     P_PrecioProducto IN PRODUCTO.PrecioProducto%TYPE,
21     P_IVA IN PRODUCTO.IVA%TYPE)
22 IS BEGIN
23     INSERT INTO PRODUCTO
24     VALUES(P_ID_Producto,P_Nombre,P_Descripcion,P_Categoria,
25             P_PrecioProducto, P_IVA);
26 END Producto_Nuevo;
27 /
28
29 CREATE OR REPLACE PROCEDURE Stock_Nuevo (
30     P_ID_Emplazamiento IN STOCK.ID_Emplazamiento%TYPE,
31     P_ID_Producto IN STOCK.ID_Producto%TYPE,
32     P_Cantidad IN STOCK.Cantidad%TYPE)
33 IS BEGIN
34     INSERT INTO STOCK
35     VALUES (P_ID_Emplazamiento, P_ID_Producto, P_Cantidad);
36 END Stock_Nuevo;
37 /
38
39 CREATE OR REPLACE PROCEDURE PRODUCTO_TRASPASO_Nuevo (
40     p_ID_Traspaso IN ASOCIACION_PRODUCTO_TRASPASO.ID_Traspaso%TYPE,
41     p_ID_Producto IN ASOCIACION_PRODUCTO_TRASPASO.ID_Producto%TYPE,
42     p_Cantidad IN ASOCIACION_PRODUCTO_TRASPASO.Cantidad%TYPE)
43 IS BEGIN
44     INSERT INTO ASOCIACION_PRODUCTO_TRASPASO
45     VALUES(p_ID_Traspaso,p_ID_Producto,p_Cantidad);
46 END PRODUCTO_TRASPASO_Nuevo;
47 /

```

```

51 /
53 CREATE OR REPLACE PROCEDURE PRODUCTO_SOLICITUD_Nuevo (
55     p_ID_Solicitud IN ASOCIACION_PRODUCTO_SOLICITUD.ID_Solicitud%TYPE,
56     p_ID_Producto IN ASOCIACION_PRODUCTO_SOLICITUD.ID_Producto%TYPE,
57     p_Cantidad IN ASOCIACION_PRODUCTO_SOLICITUD.Cantidad%TYPE)
58 IS BEGIN
59     INSERT INTO ASOCIACION_PRODUCTO_SOLICITUD
60     VALUES(p_ID_Solicitud,p_ID_Producto,p_Cantidad);
61 END PRODUCTO_SOLICITUD_Nuevo;
62 /
63 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 CREATE OR REPLACE PROCEDURE SOLICITUD_Nuevo (
77     p_ID_Solicitud IN SOLICITUD_TRASPASO.ID_Solicitud%TYPE,
78     p_FechaSolicitud IN SOLICITUD_TRASPASO.FechaSolicitud%TYPE,
79     p_ID_EmplazamientoSalida IN SOLICITUD_TRASPASO.ID_EmplazamientoSalida%TYPE,
80     p_ID_EmplazamientoEntrada IN SOLICITUD_TRASPASO.ID_EmplazamientoEntrada%TYPE)
81 IS BEGIN
82     INSERT INTO SOLICITUD_TRASPASO
83     VALUES(p_ID_Solicitud, p_FechaSolicitud, p_ID_EmplazamientoSalida,
84     p_ID_EmplazamientoEntrada);
85 END SOLICITUD_Nuevo;
86 /
87 CREATE OR REPLACE PROCEDURE Emplazamiento_Nuevo(
89     P_ID_Emplazamiento IN EMPLAZAMIENTO.ID_Emplazamiento%TYPE,
90     P_Direccion IN EMPLAZAMIENTO.Direccion%TYPE,
91     P_Telefono IN EMPLAZAMIENTO.Telefono%TYPE,
92     P_Tipo IN EMPLAZAMIENTO.Tipo%TYPE)
93 IS BEGIN
94     INSERT INTO EMPLAZAMIENTO
95     VALUES(P_ID_Emplazamiento,P_Direccion,P_Telefono,P_Tipo);
96 END Emplazamiento_Nuevo;
97 /
98 CREATE OR REPLACE PROCEDURE Factura_Nueva(
100     p_ID IN FACTURA.ID_FACTURA%TYPE,
101     p_PrecioTotal IN FACTURA.PrecioTotal%TYPE,
102     p_FechaDeExpedicion IN FACTURA.FechaDeExpedicion%TYPE,
103     p_Devuelto IN FACTURA.Devuelto%TYPE,
104     p_ID_Venta IN FACTURA.ID_Venta%TYPE,
105     p_ID_Emplazamiento IN FACTURA.ID_Emplazamiento%TYPE
106 )
107 IS BEGIN
108     INSERT INTO FACTURA VALUES(
109     p_ID, p_PrecioTotal, p_FechaDeExpedicion, p_Devuelto, p_ID_Venta,
110     p_ID_Emplazamiento);
111 END Factura_Nueva;
112 /
113 CREATE OR REPLACE PROCEDURE PROVEEDOR_Nuevo (
115     p_CIF IN PROVEEDOR.CIF%TYPE,
116     p_Nombre IN PROVEEDOR.Nombre%TYPE,
117     p_Telefono IN PROVEEDOR.Telefono%TYPE,
118     p_Email IN PROVEEDOR.Email%TYPE
119 )IS BEGIN
120     INSERT INTO PROVEEDOR

```

```

127 VALUES (p_CIF, p_Nombre, p_Telefono, p_Email);
128 END PROVEEDOR_Nuevo;
129
130 /
131
132 CREATE OR REPLACE PROCEDURE ALBARAN_Nuevo (
133     p_ID_Albaran IN ALBARAN.ID_Albaran%TYPE,
134     p_FechaFirma IN ALBARAN.FechaFirma%TYPE,
135     p_PrecioTotal IN ALBARAN.PrecioTotal%TYPE,
136     p_ID_Pedido IN ALBARAN.ID_PEDIDO%TYPE
137 )IS BEGIN
138     INSERT INTO ALBARAN
139     VALUES ( p_ID_Albaran, p_FechaFirma, p_PrecioTotal, p_ID_Pedido);
140 END ALBARAN_Nuevo;
141
142 /
143
144 CREATE OR REPLACE PROCEDURE PEDIDO_Nuevo (
145     p_ID_Pedido IN PEDIDO.ID_Pedido%TYPE,
146     p_FechaPedido IN PEDIDO.FechaPedido%TYPE,
147     p_PrecioTotal IN PEDIDO.PrecioTotal%TYPE,
148     p_ID_Emplazamiento IN PEDIDO.ID_Emplazamiento%TYPE,
149     p_CIF IN PEDIDO.CIF%TYPE
150 )IS BEGIN
151     INSERT INTO PEDIDO
152     VALUES ( p_ID_Pedido ,p_FechaPedido , p_PrecioTotal ,
153             p_ID_Emplazamiento, p_CIF);
154 END PEDIDO_Nuevo;
155
156 /
157
158 CREATE OR REPLACE PROCEDURE PEDIDO_PRODUCTO_Nuevo (
159     p_ID_Producto IN ASOCIACION_PEDIDO_PRODUCTO.ID_Producto%TYPE,
160     p_ID_Pedido IN ASOCIACION_PEDIDO_PRODUCTO.ID_Pedido%TYPE,
161     p_Cantidad IN ASOCIACION_PEDIDO_PRODUCTO.Cantidad%TYPE,
162     p_PrecioCompra IN ASOCIACION_PEDIDO_PRODUCTO.PrecioCompra%TYPE,
163     p_IVA IN ASOCIACION_PEDIDO_PRODUCTO.IVA%TYPE,
164     p_PrecioLinea IN ASOCIACION_PEDIDO_PRODUCTO.precioLinea%TYPE)IS
165 BEGIN
166
167     INSERT INTO ASOCIACION_PEDIDO_PRODUCTO
168     VALUES (p_ID_Producto, p_ID_Pedido , p_Cantidad , p_PrecioCompra,
169             p_IVA, p_PrecioLinea);
170 END PEDIDO_PRODUCTO_Nuevo;
171
172 /
173
174
175 CREATE OR REPLACE PROCEDURE Venta_Nueva(
176     P_ID_Venta IN VENTA.ID_Venta%TYPE,
177     P_PrecioTotal IN VENTA.PrecioTotal%TYPE,
178     P_FechaVenta IN VENTA.FechaVenta%TYPE,
179     P_DNI IN VENTA.DNI%TYPE)
180 IS BEGIN
181     INSERT INTO VENTA
182     VALUES(P_ID_Venta, P_PrecioTotal, P_FechaVenta, P_DNI);
183 END Venta_Nueva;
184
185 /
186
187 CREATE OR REPLACE PROCEDURE VENTA_PRODUCTO_Nueva(
188     P_ID_Venta IN ASOCIACION_VENTA_PRODUCTO.ID_Venta%TYPE,
189     P_ID_Producto IN ASOCIACION_VENTA_PRODUCTO.ID_Producto%TYPE,
190     P_Cantidad IN ASOCIACION_VENTA_PRODUCTO.Cantidad%TYPE,
191     P_PrecioVenta IN ASOCIACION_VENTA_PRODUCTO.PrecioVenta%TYPE,
192     P_IvaVenta IN ASOCIACION_VENTA_PRODUCTO.IvaVenta%TYPE,
193     P_PrecioLinea IN ASOCIACION_VENTA_PRODUCTO.PrecioLinea%TYPE
194 )
195 IS
196 BEGIN
197     INSERT INTO ASOCIACION_VENTA_PRODUCTO
198     VALUES(P_ID_Venta, P_ID_Producto,P_Cantidad,P_PrecioVenta,P_IvaVenta,P_PrecioLinea);
199 END VENTA_PRODUCTO_Nueva;
200
201 /
202
203 CREATE OR REPLACE PROCEDURE MODIFICA_PROVEEDOR_NOMBRE

```

```

205 (p_CIF IN PROVEEDOR.CIF %TYPE,
206 p_Nombre IN PROVEEDOR.Nombre %TYPE) IS
207 BEGIN
208 UPDATE PROVEEDOR SET Nombre = p_Nombre WHERE p_CIF = CIF;
209 END MODIFICA_PROVEEDOR_NOMBRE;
210
211 /
212
213 CREATE OR REPLACE PROCEDURE MODIFICA_PROVEEDOR_Telefono
214 (p_CIF IN PROVEEDOR.CIF %TYPE,
215 p_Telefono IN PROVEEDOR.Telefono %TYPE) IS
216 BEGIN
217 UPDATE PROVEEDOR SET Telefono = p_Telefono WHERE p_CIF = CIF;
218 END MODIFICA_PROVEEDOR_Telefono;
219
220 /
221
222 CREATE OR REPLACE PROCEDURE MODIFICA_PROVEEDOR_EMAIL
223 (p_CIF IN PROVEEDOR.CIF %TYPE,
224 p_Email IN PROVEEDOR.Email %TYPE) IS
225 BEGIN
226 UPDATE PROVEEDOR SET Email = p_Email WHERE p_CIF = CIF;
227 END MODIFICA_PROVEEDOR_EMAIL;
228
229 /
230
231 CREATE OR REPLACE PROCEDURE MODIFICA_PRODUCTO_DESCRIPCION (
232 p_ID_Producto IN PRODUCTO.ID_Producto %TYPE,
233 p_Nombre IN PRODUCTO.Nombre %TYPE,
234 p_Descripcion IN PRODUCTO.Descripcion %TYPE,
235 p_Categoria IN PRODUCTO.Categoria %TYPE,
236 p_PrecioProducto IN PRODUCTO.PrecioProducto %TYPE,
237 p_IVA IN PRODUCTO.IVA %TYPE)
238 IS BEGIN
239 UPDATE PRODUCTO SET Descripcion = p_Descripcion WHERE p_ID_Producto = ID_Producto;
240 END MODIFICA_PRODUCTO_DESCRIPCION;
241
242 /
243
244 CREATE OR REPLACE PROCEDURE MODIFICA_PRODUCTO_PRECIO (
245 p_ID_Producto IN PRODUCTO.ID_Producto %TYPE,
246 p_Nombre IN PRODUCTO.Nombre %TYPE,
247 p_Descripcion IN PRODUCTO.Descripcion %TYPE,
248 p_Categoria IN PRODUCTO.Categoria %TYPE,
249 p_PrecioProducto IN PRODUCTO.PrecioProducto %TYPE,
250 p_IVA IN PRODUCTO.IVA %TYPE)
251 IS BEGIN
252 UPDATE PRODUCTO SET PrecioProducto = p_PrecioProducto WHERE p_ID_Producto = ID_Producto;
253 END MODIFICA_PRODUCTO_PRECIO;
254
255 /
256
257 CREATE OR REPLACE PROCEDURE MODIFICA_PRODUCTO_IVA (
258 p_ID_Producto IN PRODUCTO.ID_Producto %TYPE,
259 p_Nombre IN PRODUCTO.Nombre %TYPE,
260 p_Descripcion IN PRODUCTO.Descripcion %TYPE,
261 p_Categoria IN PRODUCTO.Categoria %TYPE,
262 p_PrecioProducto IN PRODUCTO.PrecioProducto %TYPE,
263 p_IVA IN PRODUCTO.IVA %TYPE)
264 IS BEGIN
265 UPDATE PRODUCTO SET IVA = p_IVA
266 WHERE p_ID_Producto = ID_Producto;
267 END MODIFICA_PRODUCTO_IVA;
268
269 /
270
271 CREATE OR REPLACE PROCEDURE MODIFICA_STOCK_CANTIDAD
272 (p_ID_Emplazamiento IN STOCK.ID_Emplazamiento %TYPE,
273 p_ID_Producto IN STOCK.ID_Producto %TYPE,
274 p_Cantidad IN STOCK.Cantidad %TYPE)
275 IS BEGIN
276 UPDATE STOCK SET Cantidad = p_Cantidad
277 WHERE p_ID_Emplazamiento = ID_Emplazamiento AND p_ID_PRODUCTO = ID_PRODUCTO;
278 END MODIFICA_STOCK_CANTIDAD;
279
280 /

```



```

281 create or replace PROCEDURE MODIFICA_SOCIO_DIRECCION(
282     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,

```

```

357 f_PrecioVenta IN ASOCIACION_VENTA_PRODUCTO.PrecioVenta%TYPE)
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
CREATE OR REPLACE FUNCTION precio_Venta
(f_ID_Venta IN VENTA.ID_Venta%TYPE)
RETURN NUMBER is f_PrecioTotal VENTA.PRECIOTOTAL %TYPE;
367
BEGIN
select SUM(precioLinea) into f_PrecioTotal from ASOCIACION_VENTA_PRODUCTO
369 where ID_Venta = f_ID_Venta;
RETURN(f_PrecioTotal);
371
END precio_Venta;
373
/
375
/
377
CREATE OR REPLACE FUNCTION precio_A_Pedido_producto
(f_ID_Pedido IN ASOCIACION_PEDIDO_PRODUCTO.ID_Pedido%TYPE,
379 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
CREATE OR REPLACE FUNCTION precio_Pedido
(f_ID_Pedido IN PEDIDO.ID_Pedido%TYPE)
391 RETURN NUMBER is f_PrecioTotal PEDIDO.PRECIOTOTAL %TYPE;
BEGIN
393 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
CREATE OR REPLACE FUNCTION ganancias_mensuales(fechainicio IN factura.fechadeexpedicion%TYPE,fechafin
IN factura.fechadeexpedicion%TYPE )
401 return number is f_preciototal factura.preciototal %TYPE;
v_preciototal factura.preciototal %TYPE;
403 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;
END;

11 /
13
15 CREATE OR REPLACE TRIGGER descuento_socio
    BEFORE INSERT ON factura
    FOR EACH ROW
17   declare v_preciototal VENTA.PRECIOTOTAL%TYPE;
        v_DNI VENTA.dni%TYPE;
19 BEGIN
    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;
    IF v_DNI IS NOT NULL then
23     UPDATE venta set preciototal = v_preciototal * 0.95 where id_venta = :NEW.id_venta;
        :New.preciototal := v_preciototal * 0.95;
25   else
        update venta set preciototal = v_preciototal where id_venta = :NEW.id_venta;
27   END IF;
END;

29 /
31
33 CREATE OR REPLACE TRIGGER stock_minimo
    AFTER INSERT OR UPDATE ON stock
    FOR EACH ROW
35 BEGIN
    IF :NEW.cantidad <=0
37   THEN
        raise_application_error(-20601, :NEW.cantidad || 'No se puede realizar esta operacion');
39   END IF;
END;

41 /
43
45 CREATE OR REPLACE TRIGGER Inicializa_nueva_venta
    BEFORE INSERT ON VENTA
    FOR EACH ROW
47 BEGIN
    :NEW.PrecioTotal := 0;
49   :NEW.FechaVenta := SYSDATE;
    END Comprueba_venta;
51 /
53
55 CREATE OR REPLACE TRIGGER Inicializa_nuevo_Pedido
    BEFORE INSERT ON PEDIDO
    FOR EACH ROW
57 BEGIN
    :NEW.PrecioTotal := 0;
59   :NEW.FechaPedido := SYSDATE;
    END Comprueba_Pedido;
61 /
63
65 CREATE OR REPLACE TRIGGER Inicializa_nueva_Factura
    BEFORE INSERT ON Factura
    FOR EACH ROW
67 BEGIN
    :NEW.FechaDeExpedicion := SYSDATE;
69   END Inicializa_nueva_Factura;
71 /
73
75 CREATE OR REPLACE TRIGGER modifica_stock_venta
    BEFORE INSERT ON FACTURA
    FOR EACH ROW
77 DECLARE
    e_ID_Emplazamiento Emplazamiento.ID_Emplazamiento%TYPE;
79   v_ID_venta Venta.ID_venta%TYPE;

81   CURSOR all_prods
    IS
83   SELECT id_venta, id_producto, cantidad

```

```

85 FROM ASOCIACION_VENTA_PRODUCTO
ORDER BY ID_producto;

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

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

107 /

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

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

125 /

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

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

141 m_id_pedido ASOCIACION_PEDIDO_PRODUCTO.id_pedido%TYPE;
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;
select ID_PEDIDO into p_id_pedido from pedido where ID_pedido = :NEW.ID_Pedido;
149 OPEN all_prods;
LOOP
151 Fetch all_prods INTO m_id_pedido,m_id_producto,m_cantidad;
EXIT WHEN all_prods%NOTFOUND;
153 if m_id_pedido = p_id_pedido
THEN
155 update stock set cantidad = cantidad+m_cantidad where id_emplazamiento = e_ID_Emplazamiento AND
id_producto =m_ID_Producto;
END IF;

```

```

157     END LOOP;
        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
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
:NEW.FechaSolicitud := SYSDATE;
195 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
e_id_emplazamientoentrada SOLICITUD_TRASPASO.ID_EMPLAZAMIENTOENTRADA %TYPE;
213 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
228     BEGIN
229         :NEW.preciolinea := :New.cantidad * :New.preciocompra;
230     END inicializa_preciolinea_alp;
231 /

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

243 /

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

253 /

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

265 /

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

277 /

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

289 /

```

sql/triggers.sql

10.4. Pruebas

```

/* FUNCION AUXILIAR */
2 CREATE OR REPLACE FUNCTION EQUALS(salida BOOLEAN, salidaEsperada BOOLEAN)
RETURN VARCHAR2 AS

```

```

4| BEGIN
6|     IF (salida = salidaEsperada) THEN
8|         RETURN 'EXITO';
10|     ELSE
12|         RETURN 'FALLO';
14|     END IF;
16| END EQUALS;
18| /
20|
22| CREATE OR REPLACE PROCEDURE PRINTR(nombre_prueba VARCHAR2, salida BOOLEAN,
24|                                     salidaEsperada BOOLEAN)
26| IS BEGIN
28|     DBMS_OUTPUT.put_line(nombre_prueba || ':' || EQUALS(salida, salidaEsperada));
30| END PRINTR;
32| /
34|
36| /* DEFINICION PRUEBAS */
38| CREATE OR REPLACE PACKAGE PRUEBAS_SOCIO AS
40|     PROCEDURE inicializar;
42|     PROCEDURE insertar
44|         (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_apellidos VARCHAR2,
46|          i_DNI VARCHAR2, i_direccion VARCHAR2, i_nacimiento DATE, i_email VARCHAR2
48|          ,salidaEsperada BOOLEAN);
50|     PROCEDURE actualizar
52|         (nombre_prueba VARCHAR2, a_DNI VARCHAR2, a_direccion VARCHAR2,
54|          a_email VARCHAR2, salidaEsperada BOOLEAN);
56|     PROCEDURE eliminar
58|         (nombre_prueba VARCHAR2, e_DNI VARCHAR2, salidaEsperada BOOLEAN);
60| END PRUEBAS_SOCIO;
62| /
64|
66| CREATE OR REPLACE PACKAGE PRUEBAS_PROVEEDOR AS
68|     PROCEDURE inicializar;
70|     PROCEDURE insertar
72|         (nombre_prueba VARCHAR2, i_CIF VARCHAR2, i_nombre VARCHAR2,
74|          i_telefono INT, i_email VARCHAR2, salidaEsperada BOOLEAN);
76|     PROCEDURE actualizar
78|         (nombre_prueba VARCHAR2, a_cif VARCHAR2, a_nombre VARCHAR2,
80|          a_telefono INT, a_email VARCHAR2, salidaEsperada BOOLEAN);
82|     PROCEDURE eliminar
84|         (nombre_prueba VARCHAR2, e_CIF VARCHAR2, salidaEsperada BOOLEAN);
86| END PRUEBAS_PROVEEDOR;
88| /
90|
92| CREATE OR REPLACE PACKAGE PRUEBAS_EMPLAZAMIENTO AS
94|     PROCEDURE inicializar;
96|     PROCEDURE insertar
98|         (nombre_prueba VARCHAR2, i_direccion VARCHAR2, i_telefono INT,
100|          i_tipo VARCHAR2, salidaEsperada BOOLEAN);
102|     PROCEDURE actualizar
104|         (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_direccion VARCHAR2,
106|          a_telefono INT, salidaEsperada BOOLEAN);
108|     PROCEDURE eliminar
110|         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, salidaEsperada BOOLEAN);
112| END PRUEBAS_EMPLAZAMIENTO;
114| /
116|
118| CREATE OR REPLACE PACKAGE PRUEBAS_ALBARAN AS
120|     PROCEDURE inicializar;
122|     PROCEDURE insertar
124|         (nombre_prueba VARCHAR2, i_fecha DATE,
126|          i_precio NUMBER, i_id_pedido INT, salidaEsperada BOOLEAN);
128|     PROCEDURE eliminar
130|         (nombre_prueba VARCHAR2, e_id_albaran INT, salidaEsperada BOOLEAN);
132| END PRUEBAS_ALBARAN;
134| /
136|
138| CREATE OR REPLACE PACKAGE PRUEBAS_PEDIDO AS
140|     PROCEDURE inicializar;
142|     PROCEDURE insertar
144|         (nombre_prueba VARCHAR2, i_fecha DATE, i_precio NUMBER,

```

```

82         i_id_emplazamiento INT, i_cif VARCHAR2, salidaEsperada BOOLEAN);
PROCEDURE eliminar
84     (nombre_prueba VARCHAR2, e_id_pedido INT, salidaEsperada BOOLEAN);
END PRUEBAS_PEDIDO;

86 /

88 CREATE OR REPLACE PACKAGE PRUEBAS_PRODUCTO AS
PROCEDURE inicializar;
90 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
94     (nombre_prueba VARCHAR2, a_id_producto INT, a_descripcion VARCHAR2,
    a_precioProducto INT, a_iva INT, salidaEsperada BOOLEAN);
96 PROCEDURE eliminar
    (nombre_prueba VARCHAR2, e_id_producto INT, salidaEsperada BOOLEAN);
98 END PRUEBAS_PRODUCTO;

100 /

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

/

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

124 /

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

138 /

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

148 /

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

```



```

158     PROCEDURE eliminar
        (nombre_prueba VARCHAR2, e_id_venta INT, e_id_producto INT, salidaEsperada BOOLEAN);
160 END PRUEBAS_A_VENTA_PRODUCTO;
162 /
162 CREATE OR REPLACE PACKAGE PRUEBA_STOCK AS
164     PROCEDURE inicializar;
        PROCEDURE insertar
166         (nombre_prueba VARCHAR2, i_id_emplazamiento INT, i_id_producto INT,
            i_cantidad INT, salidaEsperada BOOLEAN);
168     PROCEDURE actualizar
        (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_id_producto INT,
170         a_cantidad INT, salidaEsperada BOOLEAN);
        PROCEDURE eliminar
172         (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT,
            salidaEsperada BOOLEAN);
174     PROCEDURE stock_correcto
        (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
176         INT, salidaEsperada BOOLEAN);
        PROCEDURE stock_correcto_p
        (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
178         INT, salidaEsperada BOOLEAN);
        PROCEDURE stock_correcto_t
        (nombre_prueba VARCHAR2, e_id_emplazamiento_envia int, e_id_emplazamiento_recibe int,
180         stock_previo_envia int, stock_previo_recibe int,
            cantidad int, e_id_producto int, salidaEsperada BOOLEAN);
182 END PRUEBA_STOCK;
184 /
184 CREATE OR REPLACE PACKAGE PRUEBAS_A_PRODUCTO_SOLICITUD AS
186     PROCEDURE inicializar;
        PROCEDURE insertar
188         (nombre_prueba VARCHAR2, i_id_producto INT, i_id_Solicitud INT,
            i_cantidad NUMBER, salidaEsperada BOOLEAN);
190     PROCEDURE eliminar
        (nombre_prueba VARCHAR2, e_id_Solicitud INT, e_id_producto INT, salidaEsperada BOOLEAN);
192 END PRUEBAS_A_PRODUCTO_SOLICITUD;
194 /
196 CREATE OR REPLACE PACKAGE PRUEBAS_A_PRODUCTO_TRASPASO AS
198     PROCEDURE inicializar;
        PROCEDURE insertar
200         (nombre_prueba VARCHAR2, i_id_producto INT, i_id_Traspaso INT,
            i_cantidad NUMBER, salidaEsperada BOOLEAN);
202     PROCEDURE eliminar
        (nombre_prueba VARCHAR2, e_id_Traspaso INT, e_id_producto INT, salidaEsperada BOOLEAN);
204 END PRUEBAS_A_PRODUCTO_TRASPASO;
206 /
206 CREATE OR REPLACE PACKAGE PRUEBAS_A_PEDIDO_PRODUCTO AS
208     PROCEDURE inicializar;
        PROCEDURE insertar
210         (nombre_prueba VARCHAR2, i_id_pedido INT, i_id_producto INT,
            i_cantidad NUMBER, i_precioCompra INT, i_iva INT, i_precioLinea INT,
212         salidaEsperada BOOLEAN);
        PROCEDURE eliminar
214         (nombre_prueba VARCHAR2, e_id_pedido INT, e_id_producto INT,
            salidaEsperada BOOLEAN);
216 END PRUEBAS_A_PEDIDO_PRODUCTO;
218 /
220 /* CUERPOS DE PRUEBAS */
222 CREATE OR REPLACE PACKAGE BODY PRUEBAS_SOCIO AS
224     PROCEDURE inicializar AS
        BEGIN
226         DELETE FROM SOCIO;
        END inicializar;
228
        PROCEDURE insertar
230         (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_apellidos VARCHAR2,

```

```

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

```

```

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

```

```

382      DELETE FROM EMPLAZAMIENTO;
END inicializar;

384  PROCEDURE insertar
      (nombre_prueba VARCHAR2, i_direccion VARCHAR2, i_telefono INT,
386      i_tipo VARCHAR2, salidaEsperada BOOLEAN) AS
      salida BOOLEAN := true;
388      actual emplazamiento%ROWTYPE;
      w_ID_Emplazamiento INT;
390  BEGIN
      INSERT INTO emplazamiento VALUES(null,i_direccion, i_telefono, i_tipo);

392      w_ID_Emplazamiento := S_ID_Emplazamiento.currval;
      SELECT * INTO actual FROM EMPLAZAMIENTO WHERE ID_EMPLAZAMIENTO = w_ID_Emplazamiento;
394
      IF (actual.direccion<>i_direccion) OR (actual.telefono<>i_telefono) OR (actual.tipo<>i_tipo)
396  THEN
          salida := false;
398  END IF;

400      PRINTR(nombre_prueba, salida, salidaEsperada);

402      EXCEPTION
      WHEN OTHERS THEN
404          PRINTR(nombre_prueba, false, salidaEsperada);
          ROLLBACK;
406  END insertar;

408  PROCEDURE actualizar
      (nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_direccion VARCHAR2,
410      a_telefono INT, salidaEsperada BOOLEAN) AS
      salida BOOLEAN := true;
412      actual emplazamiento%ROWTYPE;
      BEGIN
414          UPDATE emplazamiento SET DIRECCION = a_direccion, TELEFONO = a_telefono where
              ID_EMPLAZAMIENTO = a_id_emplazamiento;

416          SELECT * INTO actual FROM EMPLAZAMIENTO WHERE ID_EMPLAZAMIENTO = a_id_emplazamiento;

418          IF (actual.direccion<>a_direccion) OR
              (actual.telefono<>a_telefono) THEN
420              salida := false;
          END IF;

422          PRINTR(nombre_prueba, salida, salidaEsperada);

424          EXCEPTION
      WHEN OTHERS THEN
426              PRINTR(nombre_prueba, false, salidaEsperada);
              ROLLBACK;
428  END actualizar;

430  PROCEDURE eliminar
      (nombre_prueba VARCHAR2, e_id_emplazamiento INT, salidaEsperada BOOLEAN) AS
432      salida BOOLEAN := true;
      cantidad INT;
434  BEGIN
436      DELETE FROM EMPLAZAMIENTO WHERE ID_EMPLAZAMIENTO = e_id_emplazamiento;

438      SELECT COUNT(*) INTO cantidad FROM EMPLAZAMIENTO WHERE ID_EMPLAZAMIENTO =
          e_id_emplazamiento;

440      IF (cantidad<>0) THEN
          salida := false;
442  END IF;

444      PRINTR(nombre_prueba, salida, salidaEsperada);

446      EXCEPTION
      WHEN OTHERS THEN
448          PRINTR(nombre_prueba, false, salidaEsperada);
          ROLLBACK;
450  END eliminar;
END PRUEBAS_EMPLAZAMIENTO;
452
454 /

```

```

456 CREATE OR REPLACE PACKAGE BODY PRUEBAS_PRODUCTO AS
457
458     PROCEDURE inicializar AS
459     BEGIN
460         DELETE FROM PRODUCTO;
461     END inicializar;
462
463     PROCEDURE insertar
464     (nombre_prueba VARCHAR2, i_nombre VARCHAR2, i_descripcion VARCHAR2,
465      i_categoria VARCHAR2, i_precioProducto INT, i_iva INT, salidaEsperada BOOLEAN) AS
466     salida BOOLEAN := true;
467     actual producto%ROWTYPE;
468     w_ID_Producto INT;
469     BEGIN
470         INSERT INTO producto VALUES(null,i_nombre, i_descripcion, i_categoria, i_precioProducto,
471         i_iva);
472
473         w_ID_Producto := S_ID_Producto.currval;
474         SELECT * INTO actual FROM PRODUCTO WHERE ID_Producto = w_ID_Producto;
475
476         IF (actual.nombre<>i_nombre) OR (actual.descripcion<>i_descripcion) OR (actual.categoria<>
477         i_categoria) OR (actual.precioProducto<>i_precioProducto) OR (actual.iva<>i_iva)THEN
478             salida := false;
479         END IF;
480
481         PRINTR(nombre_prueba, salida, salidaEsperada);
482
483         EXCEPTION
484         WHEN OTHERS THEN
485             PRINTR(nombre_prueba, false, salidaEsperada);
486             ROLLBACK;
487     END insertar;
488
489     PROCEDURE actualizar
490     (nombre_prueba VARCHAR2, a_id_producto INT , a_descripcion VARCHAR2,
491      a_precioProducto INT, a_iva INT, salidaEsperada BOOLEAN)AS
492     salida BOOLEAN := true;
493     actual producto%ROWTYPE;
494     BEGIN
495         UPDATE producto SET DESCRIPCION = a_descripcion, PRECIOPRODUCTO = a_precioProducto, IVA =
496         a_iva
497         where ID_Producto = a_id_producto;
498
499         SELECT * INTO actual FROM PRODUCTO WHERE ID_Producto = a_id_producto;
500         IF (actual.descripcion<>a_descripcion) OR
501         (actual.precioProducto<>a_precioProducto) OR
502         (actual.iva<>a_iva) THEN
503             salida := false;
504         END IF;
505
506         PRINTR(nombre_prueba, salida, salidaEsperada);
507
508         EXCEPTION
509         WHEN OTHERS THEN
510             PRINTR(nombre_prueba, false, salidaEsperada);
511             ROLLBACK;
512     END actualizar;
513
514     PROCEDURE eliminar
515     (nombre_prueba VARCHAR2, e_id_producto INT, salidaEsperada BOOLEAN) AS
516     salida BOOLEAN := true;
517     cantidad INT;
518     BEGIN
519         DELETE FROM PRODUCTO WHERE ID_Producto = e_id_producto;
520
521         SELECT COUNT(*) INTO cantidad FROM PRODUCTO WHERE ID_Producto = e_id_producto;
522         IF (cantidad<>0) THEN
523             salida := false;
524         END IF;
525
526         PRINTR(nombre_prueba, salida, salidaEsperada);
527
528         EXCEPTION
529         WHEN OTHERS THEN
530             PRINTR(nombre_prueba, false, salidaEsperada);
531             ROLLBACK;
532     END eliminar;

```

```

530 END PRUEBAS_PRODUCTO;
531 /
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;

```

```

604     END eliminar;
END PRUEBAS_factura;

606 /

608 CREATE OR REPLACE PACKAGE BODY PRUEBA_STOCK AS
PROCEDURE inicializar AS
610 BEGIN
612     DELETE FROM stock;
END inicializar;

614 PROCEDURE insertar
(nombre_prueba VARCHAR2, i_id_emplazamiento INT, i_id_producto INT,
616 i_cantidad INT, salidaEsperada BOOLEAN) AS
salida BOOLEAN := true;
618 actual stock%ROWTYPE;
BEGIN
620     INSERT INTO stock VALUES(i_id_emplazamiento, i_id_producto, i_cantidad);

622     SELECT * INTO actual FROM stock WHERE ID_Emplazamiento = i_id_emplazamiento and ID_Producto =
i_id_producto;

624     IF (actual.id_emplazamiento<>i_id_emplazamiento) OR (actual.id_producto<>i_id_producto) OR (
actual.cantidad<>i_cantidad) THEN
        salida := false;
626     END IF;

628     PRINTR(nombre_prueba, salida, salidaEsperada);

630     EXCEPTION
632     WHEN OTHERS THEN
        PRINTR(nombre_prueba, false, salidaEsperada);
        ROLLBACK;
634 END insertar;

636 PROCEDURE actualizar
(nombre_prueba VARCHAR2, a_id_emplazamiento INT, a_id_producto INT,
638 a_cantidad INT, salidaEsperada BOOLEAN) AS
salida BOOLEAN := true;
640 actual stock%ROWTYPE;
BEGIN
642     UPDATE stock SET CANTIDAD = a_cantidad where ID_Emplazamiento = a_id_emplazamiento and
ID_Producto = a_id_producto;

644     SELECT * INTO actual FROM stock WHERE ID_Emplazamiento = a_id_emplazamiento and
ID_Producto = a_id_producto;
        IF (actual.cantidad<>a_cantidad) THEN
646             salida := false;
        END IF;

648     PRINTR(nombre_prueba, salida, salidaEsperada);

650     EXCEPTION
652     WHEN OTHERS THEN
        PRINTR(nombre_prueba, false, salidaEsperada);
        ROLLBACK;
654 END actualizar;

656 PROCEDURE eliminar
(nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT,
658 salidaEsperada BOOLEAN) AS
salida BOOLEAN := true;
660 cantidad INT;
662 BEGIN
        DELETE FROM stock WHERE ID_Emplazamiento = e_id_emplazamiento and ID_Producto =
e_id_producto;

664     SELECT COUNT(*) INTO cantidad FROM stock WHERE ID_Emplazamiento = e_id_emplazamiento and
ID_Producto = e_id_producto;
        IF (cantidad<>0) THEN
666             salida := false;
        END IF;

668     PRINTR(nombre_prueba, salida, salidaEsperada);

670     EXCEPTION
672     WHEN OTHERS THEN

```

```

674         PRINTR(nombre_prueba, false, salidaEsperada);
675         ROLLBACK;
676     END eliminar;

677
678     PROCEDURE stock_correcto
679     (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
680     INT, salidaEsperada BOOLEAN) AS
681     salida BOOLEAN := true;
682     stock_nuevo int;
683     BEGIN
684         select cantidad into stock_nuevo from stock where id_emplazamiento = e_id_emplazamiento and
685         id_producto= e_id_producto;
686         if (stock_previo-cantidad)<> stock_nuevo then
687             salida := false;
688         END IF;
689
690         PRINTR(nombre_prueba, salida, salidaEsperada);
691
692     EXCEPTION
693     WHEN OTHERS THEN
694         PRINTR(nombre_prueba, false, salidaEsperada);
695         ROLLBACK;
696
697     END stock_correcto;
698
699     PROCEDURE stock_correcto_p
700     (nombre_prueba VARCHAR2, e_id_emplazamiento INT, e_id_producto INT, stock_previo int, cantidad
701     INT, salidaEsperada BOOLEAN) AS
702     salida BOOLEAN := true;
703     stock_nuevo int;
704     BEGIN
705         select cantidad into stock_nuevo from stock where id_emplazamiento = e_id_emplazamiento and
706         id_producto= e_id_producto;
707         if (stock_previo+cantidad)<> stock_nuevo then
708             salida := false;
709         END IF;
710
711         PRINTR(nombre_prueba, salida, salidaEsperada);
712
713     EXCEPTION
714     WHEN OTHERS THEN
715         PRINTR(nombre_prueba, false, salidaEsperada);
716         ROLLBACK;
717
718     end stock_correcto_p;
719
720     PROCEDURE stock_correcto_t
721     (nombre_prueba VARCHAR2, e_id_emplazamiento_envia int, e_id_emplazamiento_recibe int,
722     stock_previo_envia int, stock_previo_recibe int,
723     cantidad int, e_id_producto int, salidaEsperada BOOLEAN) AS
724     salida BOOLEAN := true;
725     stock_nuevo_envia int;
726     stock_nuevo_recibe int;
727     BEGIN
728         select cantidad into stock_nuevo_envia from stock where id_emplazamiento =
729         e_id_emplazamiento_envia and id_producto = e_id_producto;
730         select cantidad into stock_nuevo_recibe from stock where id_emplazamiento =
731         e_id_emplazamiento_recibe and id_producto = e_id_producto;
732         if (stock_previo_envia-cantidad)<>stock_nuevo_envia or (stock_previo_recibe+cantidad)<>
733         stock_nuevo_recibe then
734             salida := false;
735         END IF;
736         PRINTR(nombre_prueba, salida, salidaEsperada);
737
738     EXCEPTION
739     WHEN OTHERS THEN
740         PRINTR(nombre_prueba, false, salidaEsperada);
741         ROLLBACK;
742
743     end stock_correcto_t;
744
745 END PRUEBA_STOCK;
746
747 /
748
749 CREATE OR REPLACE PACKAGE BODY PRUEBAS_VENTA AS
750
751     PROCEDURE inicializar AS
752     BEGIN

```



```

744      DELETE FROM VENTA;
745  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      w_ID_Venta := S_ID_Venta.currval;
755      SELECT * INTO actual FROM venta WHERE ID_Venta = w_ID_Venta;

756      IF (actual.PrecioTotal <> i_PrecioVenta) OR (actual.fechaVenta <> i_fecha) OR (actual.dni <> i_dni)
757      THEN
758          salida := false;
759      END IF;

760      PRINTR(nombre_prueba, salida, salidaEsperada);

761      EXCEPTION
762      WHEN OTHERS THEN
763          PRINTR(nombre_prueba, false, salidaEsperada);
764          ROLLBACK;
765  END insertar;

766  PROCEDURE eliminar
767      (nombre_prueba VARCHAR2, e_id_venta INT, salidaEsperada BOOLEAN) AS
768      salida BOOLEAN := true;
769      cantidad INT;
770  BEGIN
771      DELETE FROM venta WHERE ID_Venta = e_id_venta;

772      SELECT COUNT(*) INTO cantidad FROM venta WHERE ID_Venta = e_id_venta;
773      IF (cantidad <> 0) THEN
774          salida := false;
775      END IF;

776      PRINTR(nombre_prueba, salida, salidaEsperada);

777      EXCEPTION
778      WHEN OTHERS THEN
779          PRINTR(nombre_prueba, false, salidaEsperada);
780          ROLLBACK;
781  END eliminar;

782  procedure descuento
783      (nombre_prueba VARCHAR2, v_id_venta int, v_preciototal number, v_dni_socio varchar2,
784      salidaEsperada BOOLEAN) as
785      salida BOOLEAN := true;
786      p_preciototal number;
787  begin
788      select preciototal into p_preciototal from venta where id_venta = v_id_venta;
789      if (v_dni_socio <> null) then
790          if (v_preciototal * 0.95 <> p_preciototal) then
791              salida := false;
792          END IF;
793      END IF;

794      printr(nombre_prueba, salida, salidaEsperada);
795      EXCEPTION
796      WHEN OTHERS THEN
797          PRINTR(nombre_prueba, false, salidaEsperada);
798          ROLLBACK;
799  end descuento;
800  END PRUEBAS_VENTA;
801  /
802  CREATE OR REPLACE PACKAGE BODY PRUEBAS_PEDIDO AS
803  BEGIN
804      PROCEDURE inicializar AS
805      BEGIN
806          DELETE FROM pedido;
807      END inicializar;

```

```

818 PROCEDURE insertar
      (nombre_prueba VARCHAR2, i_fecha DATE, i_precio NUMBER,
820       i_id_emplazamiento INT, i_cif VARCHAR2, salidaEsperada BOOLEAN) AS
      salida BOOLEAN := true;
822      actual_pedido %ROWTYPE;
      w_ID_pedido INT;
824      BEGIN
          INSERT INTO pedido VALUES(null, i_fecha, i_precio, i_id_emplazamiento, i_cif);

826          w_ID_pedido := S_ID_Pedido.currval;
          SELECT * INTO actual FROM pedido WHERE ID_Pedido = w_ID_pedido;

828          IF (actual.fechaPedido<>i_fecha) OR (actual.precioTotal<>i_precio) OR (actual.
            id_emplazamiento<>i_id_emplazamiento) OR (actual.cif<>i_cif) THEN
              salida := false;
830              END IF;
832          PRINTR(nombre_prueba, salida, salidaEsperada);

834          EXCEPTION
          WHEN OTHERS THEN
836              PRINTR(nombre_prueba, false, salidaEsperada);
              ROLLBACK;
838          END insertar;
840
842 PROCEDURE eliminar
      (nombre_prueba VARCHAR2, e_id_pedido INT, salidaEsperada BOOLEAN) AS
844      salida BOOLEAN := true;
      cantidad INT;
846      BEGIN
          DELETE FROM pedido WHERE ID_Pedido = e_id_pedido;

848          SELECT COUNT(*) INTO cantidad FROM pedido WHERE ID_Pedido = e_id_pedido;
          IF (cantidad<>0) THEN
850              salida := false;
          END IF;
852          PRINTR(nombre_prueba, salida, salidaEsperada);

854          EXCEPTION
          WHEN OTHERS THEN
856              PRINTR(nombre_prueba, false, salidaEsperada);
              ROLLBACK;
858          END eliminar;
860 END PRUEBAS_PEDIDO;
862
864 /
866
868 CREATE OR REPLACE PACKAGE BODY PRUEBAS_TRASPASO AS
870
872     PROCEDURE inicializar AS
874     BEGIN
876         DELETE FROM traspaso;
878     END inicializar;
880
882     PROCEDURE insertar
      (nombre_prueba VARCHAR2, i_fechaTraspaso DATE, i_id_emplazamientoSalida INT,
874       i_id_emplazamientoEntrada INT, salidaEsperada BOOLEAN) AS
      salida BOOLEAN := true;
876      actual_traspaso %ROWTYPE;
      w_ID_traspaso INT;
878      BEGIN
          INSERT INTO traspaso VALUES(null, i_fechaTraspaso, i_id_emplazamientoSalida,
            i_id_emplazamientoEntrada);

880          w_ID_traspaso := S_ID_traspaso.currval;
          SELECT * INTO actual FROM traspaso WHERE ID_traspaso = w_ID_traspaso;

882          IF (actual.fechaTraspaso<>i_fechaTraspaso) OR (actual.id_emplazamientoSalida<>
            i_id_emplazamientoSalida) OR (actual.id_emplazamientoEntrada<>i_id_emplazamientoEntrada) THEN
              salida := false;
884              END IF;
886          PRINTR(nombre_prueba, salida, salidaEsperada);

888          EXCEPTION
          WHEN OTHERS THEN
890

```

```

892         PRINTR(nombre_prueba, false, salidaEsperada);
893         ROLLBACK;
894     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         SELECT COUNT(*) INTO cantidad FROM traspaso WHERE ID_traspaso = e_id_traspaso;
903         IF (cantidad <> 0) THEN
904             salida := false;
905         END IF;

906         PRINTR(nombre_prueba, salida, salidaEsperada);

907         EXCEPTION
908         WHEN OTHERS THEN
909             PRINTR(nombre_prueba, false, salidaEsperada);
910             ROLLBACK;
911     END eliminar;
912 END PRUEBAS_TRASPASO;
913
914 /
915
916 CREATE OR REPLACE PACKAGE BODY PRUEBAS_Solicitud_Traspaso AS
917
918     PROCEDURE inicializar AS
919     BEGIN
920         DELETE FROM Solicitud_Traspaso;
921     END inicializar;

922     PROCEDURE insertar
923     (nombre_prueba VARCHAR2, i_fechaSolicitud DATE, i_id_emplazamientoSalida INT,
924     i_id_emplazamientoEntrada INT, salidaEsperada BOOLEAN) AS
925     salida BOOLEAN := true;
926     actual Solicitud_Traspaso%ROWTYPE;
927     w_ID_Solicitud INT;
928     BEGIN
929         INSERT INTO Solicitud_Traspaso VALUES(null, i_fechaSolicitud, i_id_emplazamientoSalida,
930         i_id_emplazamientoEntrada);

931         w_ID_Solicitud := S_ID_Solicitud.currval;
932         SELECT * INTO actual FROM Solicitud_Traspaso WHERE ID_Solicitud = w_ID_Solicitud;

933         IF (actual.fechaSolicitud <> i_fechaSolicitud) OR (actual.id_emplazamientoSalida <>
934         i_id_emplazamientoSalida) OR (actual.id_emplazamientoEntrada <> i_id_emplazamientoEntrada) THEN
935             salida := false;
936         END IF;

937         PRINTR(nombre_prueba, salida, salidaEsperada);

938         EXCEPTION
939         WHEN OTHERS THEN
940             PRINTR(nombre_prueba, false, salidaEsperada);
941             ROLLBACK;
942     END insertar;

943     PROCEDURE eliminar
944     (nombre_prueba VARCHAR2, e_id_solicitudTraspaso INT, salidaEsperada BOOLEAN) AS
945     salida BOOLEAN := true;
946     cantidad INT;
947     BEGIN
948         DELETE FROM Solicitud_Traspaso WHERE ID_Solicitud = e_id_solicitudTraspaso;

949         SELECT COUNT(*) INTO cantidad FROM Solicitud_Traspaso WHERE ID_Solicitud =
950         e_id_solicitudTraspaso;
951         IF (cantidad <> 0) THEN
952             salida := false;
953         END IF;

954         PRINTR(nombre_prueba, salida, salidaEsperada);

955         EXCEPTION
956         WHEN OTHERS THEN

```

```

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

```

```

1042      SELECT * INTO actual FROM asociacion_Producto_traspaso WHERE ID_Producto = i_id_producto and
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;
END IF;

1048      PRINTR(nombre_prueba, salida, salidaEsperada);

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

1056      PROCEDURE eliminar
(nombre_prueba VARCHAR2, e_id_Traspaso INT, e_id_producto INT, salidaEsperada BOOLEAN) AS
1058      salida BOOLEAN := true;
cantidad INT;
1060      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
1066      salida := false;
END IF;

1068      PRINTR(nombre_prueba, salida, salidaEsperada);

1070      EXCEPTION
1072      WHEN OTHERS THEN
PRINTR(nombre_prueba, false, salidaEsperada);
1074      ROLLBACK;
END eliminar;
1076      END PRUEBAS_A_PRODUCTO_TRASPASO;
/
1078
1080      CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_PRODUCTO_SOLICITUD AS

1082      PROCEDURE inicializar AS
BEGIN
1084      DELETE FROM asociacion_Producto_Solicitud;
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
1092      INSERT INTO asociacion_Producto_Solicitud VALUES(i_id_Solicitud, i_id_producto, i_cantidad);

1094      SELECT * INTO actual FROM asociacion_Producto_Solicitud WHERE ID_Producto = i_id_producto and
ID_Solicitud = i_id_Solicitud;

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

1100      PRINTR(nombre_prueba, salida, salidaEsperada);

1102      EXCEPTION
1104      WHEN OTHERS THEN
PRINTR(nombre_prueba, false, salidaEsperada);
1106      ROLLBACK;
END insertar;

1108      PROCEDURE eliminar
(nombre_prueba VARCHAR2, e_id_Solicitud INT, e_id_producto INT, salidaEsperada BOOLEAN) AS
1110      salida BOOLEAN := true;
cantidad INT;
1112      BEGIN

```

```

1114         DELETE FROM asociacion_Producto_Solicitud WHERE ID_Producto = e_id_producto and
ID_Solicitud = e_id_Solicitud;

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

        PRINTR(nombre_prueba, salida, salidaEsperada);

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

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

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

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

        PRINTR(nombre_prueba, salida, salidaEsperada);

1154     EXCEPTION
        WHEN OTHERS THEN
1156         PRINTR(nombre_prueba, false, salidaEsperada);
            ROLLBACK;
1158     END insertar;

1160     PROCEDURE eliminar
        (nombre_prueba VARCHAR2, e_id_pedido INT, e_id_producto INT,
1162         salidaEsperada BOOLEAN) AS
        salida BOOLEAN := true;
1164         cantidad INT;
        BEGIN
1166             DELETE FROM asociacion_Pedido_Producto WHERE ID_pedido = e_id_pedido and ID_producto =
e_id_producto;

1168             SELECT COUNT(*) INTO cantidad FROM asociacion_Pedido_Producto WHERE ID_pedido =
e_id_pedido and ID_producto = e_id_producto;
1170             IF (cantidad<>0) THEN
                salida := false;
1172             END IF;

1174             PRINTR(nombre_prueba, salida, salidaEsperada);

1176     EXCEPTION
        WHEN OTHERS THEN
1178         PRINTR(nombre_prueba, false, salidaEsperada);
            ROLLBACK;
1180     END eliminar;
END PRUEBAS_A_PEDIDO_PRODUCTO;
1182

```

```

1184 /
1186 CREATE OR REPLACE PACKAGE BODY PRUEBAS_A_VENTA_PRODUCTO AS
1188     PROCEDURE inicializar AS
1189     BEGIN
1190         DELETE FROM asociacion_Venta_Producto;
1191     END inicializar;
1192
1193     PROCEDURE insertar
1194     (nombre_prueba VARCHAR2, i_id_venta INT, i_id_producto INT,
1195      i_cantidad NUMBER, i_precioVenta NUMBER, i_ivaVenta NUMBER,
1196      i_precioLinea NUMBER, salidaEsperada BOOLEAN) AS
1197     salida BOOLEAN := true;
1198     actual asociacion_Venta_Producto %ROWTYPE;
1199     BEGIN
1200         INSERT INTO asociacion_Venta_Producto VALUES(i_id_venta, i_id_producto, i_cantidad,
1201         i_precioVenta, i_ivaVenta, i_precioLinea);
1202
1203         SELECT * INTO actual FROM asociacion_Venta_Producto WHERE ID_Venta = i_id_venta and
1204         ID_producto = i_id_producto;
1205
1206         IF (actual.id_venta <> i_id_venta) OR (actual.id_producto <> i_id_producto) OR (actual.cantidad <>
1207         i_cantidad) OR (actual.precioVenta <> i_precioVenta) OR (actual.ivaVenta <> i_ivaVenta) OR (actual.
1208         precioLinea <> i_precioLinea) THEN
1209             salida := false;
1210         END IF;
1211
1212         PRINTR(nombre_prueba, salida, salidaEsperada);
1213
1214         EXCEPTION
1215         WHEN OTHERS THEN
1216             PRINTR(nombre_prueba, false, salidaEsperada);
1217             ROLLBACK;
1218     END insertar;
1219
1220     PROCEDURE eliminar
1221     (nombre_prueba VARCHAR2, e_id_venta INT, e_id_producto INT, salidaEsperada BOOLEAN) AS
1222     salida BOOLEAN := true;
1223     cantidad INT;
1224     BEGIN
1225         DELETE FROM asociacion_Venta_Producto WHERE ID_Venta = e_id_venta and ID_producto =
1226         e_id_producto;
1227
1228         SELECT COUNT(*) INTO cantidad FROM asociacion_Venta_Producto WHERE ID_Venta = e_id_venta
1229         and ID_producto = e_id_producto;
1230         IF (cantidad <> 0) THEN
1231             salida := false;
1232         END IF;
1233
1234         PRINTR(nombre_prueba, salida, salidaEsperada);
1235
1236         EXCEPTION
1237         WHEN OTHERS THEN
1238             PRINTR(nombre_prueba, false, salidaEsperada);
1239             ROLLBACK;
1240     END eliminar;
1241 END PRUEBAS_A_VENTA_PRODUCTO;
1242
1243 /
1244
1245 DROP SEQUENCE S_ID_Producto;
1246 DROP SEQUENCE S_ID-Traspaso;
1247 DROP SEQUENCE S_ID_Solicitud;
1248 DROP SEQUENCE S_ID_Pedido;
1249 DROP SEQUENCE S_ID-Albaran;
1250 DROP SEQUENCE S_ID-Factura;
1251 DROP SEQUENCE S_ID-Emplazamiento;
1252 DROP SEQUENCE S_ID_Venta;
1253
1254 CREATE SEQUENCE S_ID-Emplazamiento START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1255 CREATE SEQUENCE S_ID_Venta START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1256 CREATE SEQUENCE S_ID_Pedido START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1257 CREATE SEQUENCE S_ID-Albaran START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1258 CREATE SEQUENCE S_ID_Producto START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1259 CREATE SEQUENCE S_ID-Traspaso START WITH 1 INCREMENT BY 1 MAXVALUE 9999;
1260 CREATE SEQUENCE S_ID_Solicitud START WITH 1 INCREMENT BY 1 MAXVALUE 9999;

```

```

1254 CREATE SEQUENCE S_ID_Factura START WITH 1 INCREMENT BY 1 MAXVALUE 9999;

1256 /* 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 disenadora','
Jerseys',46.95,21,true);
EXECUTE PRUEBAS_PRODUCTO.insertar('Producto insercion','Sombrero de paja','el gran sombrero de
Mugiwara','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 PRUEBA_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);

```

```

1364 EXECUTE PRUEBAS_ALBARAN.insertar('Albaran insercion',sysdate,0,S_ID_Pedido.currval-2,true);
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);

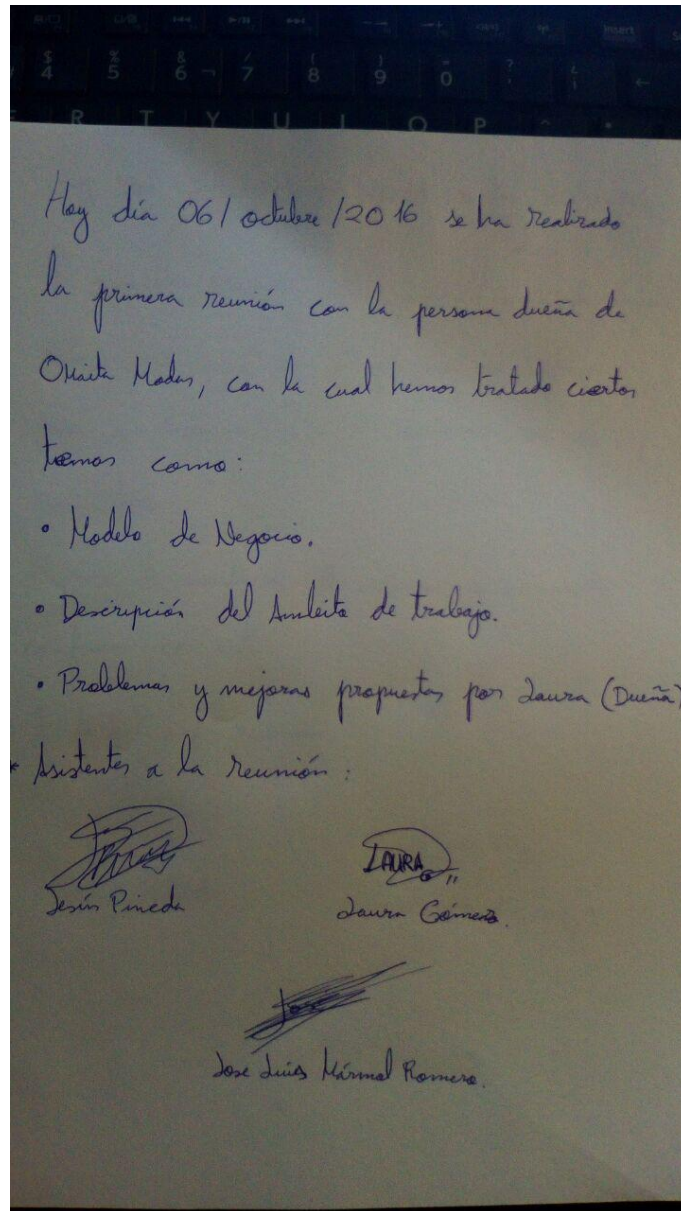
1382 --ASOC_PRODUCTO_TRASPASO
EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.inicializar();
1384 EXECUTE PRUEBAS_A_PRODUCTO_TRASPASO.insertar('A-Producto-Traspaso Insertar', S_ID_Producto.currval-1,
S_ID_traspaso.currval-1, 4, true);
EXECUTE PRUEBAS_STOCK.STOCK_CORRECTO_T('Actualizacion de stock tras traspaso',s_ID_emplazamiento.
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 Ornato 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