Reto Base de datos QA

Tienda Virtual Don pepe (Ejercicio C)

Don pepe quiere que sus clientes puedan realizar compras desde sus casas. El junto a su esposa tienen una cantidad domiciliarios conocidos que se encargan de llevar los pedidos a los clientes.

A continuación se muestra la conversación que se tuvo con don pepe:

- ¡Veee mijo! yo quiero que mas gente me compre los producticos, cuando llegá un vecino nuevo a la cuadra yo lo apunto en un cuadernito.
- Entiendo don pepe, y no le gustaría que le comprarán por internet?
- Ehhh mijo pues no es mala idea y que hago con mi clientela?
- Pues don pepe hacemos un video tutorial para usar la aplicación, y le pedimos una información a sus clientes indicando sus datos personales (ID, cedula, Nombre, Dirección, Teléfono, email y password) a través de un formulario de registro. Una vez registrado podrá acceder a la realización de pedidos con su email y su password.
- ¡ eeeee yo no te creo! Asi de fácil? Como motilando calvos?
- Don pepe ojala fuera así de sencillo dejeme le cuento mejor, Los productos que oferta el supermercado deben estar divididos en diversas categorías. Los datos necesarios para cada categoría son: nombre de la categoría, condiciones de almacenamiento (frío, congelado, seco) y observaciones. Tambien debemos detallar la información de los productos (nombre, marca, origen, dimensiones (volumen y peso), una fotografía, la categoría y unidades disponibles).¡ no mijo eso me va salir muy caro con tanto detalle!
- don pepe todo lo contrario va aumentar mucho sus ganancias espereme le cuento algo mas, la aplicación permitirá visualizar un listado de productos ordenado por categoría, permitiendo seleccionar los productos que desee comprar mediante una caja de texto donde se indicará el número de unidades seleccionadas. La aplicación llevará la cuenta (cesta de la compra) de los productos que el cliente ha ido seleccionando. La aplicación permitirá también efectuar un pedido con todos los productos que lleve almacenados en su cesta de la compra. Los datos del pedido son: código del pedido, fecha del pedido, cliente, dirección de entrega, productos pedidos, importe total del pedido y datos de pago (número de tarjeta y fecha de caducidad)®.

Para poder generar un pedido se deberán dar dos situaciones:

- El cliente deberá pertenecer a una zona (Código Postal) donde existan domiciliarios. Un domiciliario se identifica mediante un nombre, número de matrícula de la furgoneta y zona donde reparte.
- Debe haber unidades suficientes por cada producto para satisfacer las demandas de cada pedido.

Una vez generado el pedido se mostrará al usuario una página con los datos de su pedido, se restarán del stock las unidades pedidas y se emitirá una nota de entrega a los responsables de almacén para que sirvan ese pedido.

1. Realizar el modelo E-R

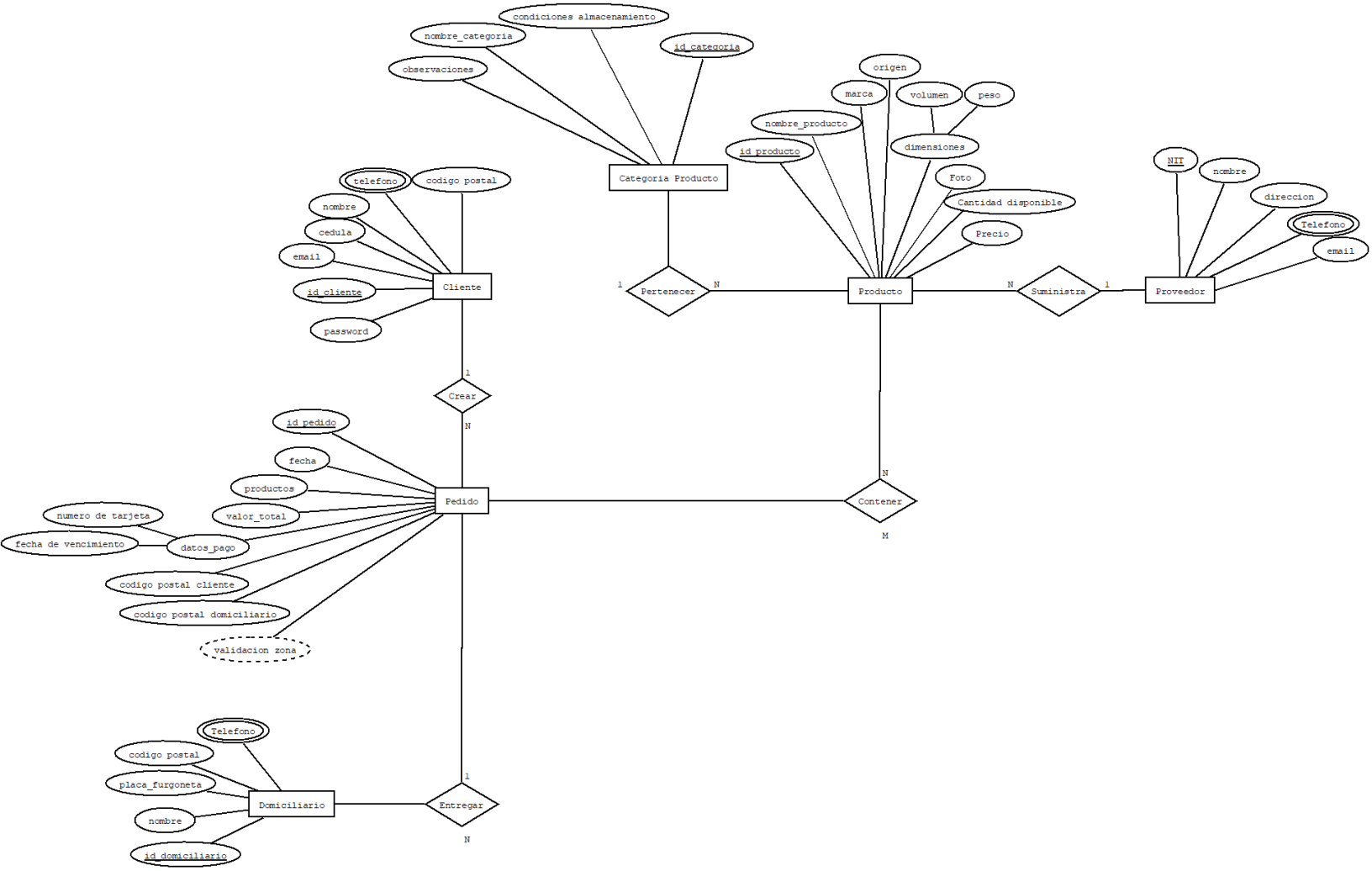


Figura 1: Modelo entidad-relacion

Para la generación del modelo E-R se tiene en cuenta el caso de la tienda y se toman en cuenta las indicaciones que se dieron durante la conversación presente en el enunciado.

Inicialmente se encuentra que para el modelo es necesario tener las siguientes entidades:

- Cliente: Es la entidad encargada de registrar la información de los clientes que se registran a la aplicación y que serán los encargados de realizar los pedidos.
- Proveedor: Es la entidad encargada de suministrar la tienda los productos a vender.

- Producto: Es la entidad encargada de registrar la información de los productos que se venderan en la tienda.
- Categoria de Producto: Es la entidad encargada de categorizar los productos en la tienda y llevar un control de las observaciones y los métodos de almacenamiento de los mismos.
- Pedido: Es la entidad encargada de llevar el registro de las solicitudes de pedidos generadas por el cliente con el fin de llevar un control y validar si es posible continuar con la entrega del pedido o este debe ser cancelado.
- Domiciliario: Es la entidad encargada de entregar los pedidos a los clientes, aquí se tiene control de sus datos y el código postal de la zona donde realiza las entregas.

Para estas entidades se identifican los siguientes atributos:

- Cliente:

- Cuenta con los atributos: Id_cliente(Llave primaria), nombre, cedula, email, password, teléfono(Atributo multivaluado) y código postal.

- Proveedor:

- Cuenta con los atributos: NIT(Llave primaria), nombre, direccion, teléfono(atributo multivalor) e Email.

- Producto:

- Cuenta con los atributos id_producto(llave primaria), nombre, marca, origen, dimensiones(atributo compuesto), foto, cantidad disponible y precio.

- Categoria de producto:

- Cuenta con los atributos id_categoria(llave primaria), nombre, observaciones del producto y condiciones de almacenamiento(Atributo compuesto)

- Pedido:

- Cuenta con los atributos id_pedido(llave primaria), fecha, productos, datos pago(atributo compuesto), valor total, código postal cliente, código postal domiciliario y validación zona atributo derivado con el cual se hará la validación de que se pueda realizar el pedido con los codigos postales de cliente y domiciliario.

- Domiciliario:

- Cuenta con los atributos id_domiciliario(llave primaria), nombre, placa furgoneta, teléfono(atributo multivaluado) y código postal

Para las relaciones entre entidades se tiene:

- Una relación Crear de 1 a N entre las entidades Cliente y Pedido teniendo en cuenta que un cliente puede crear uno o muchos pedidos pero un pedido solo puede ser creado por un cliente.
- Una relación Entregar de 1 a N entre las entidades Domiciliario y Pedido teniendo en cuenta que un pedido es entregado por un domiciliario pero un domiciliario puede entregar uno o muchos pedidos.
- Una relación Pertenecer de 1 a N entre las entidades Categoría Producto y Producto ya que un producto solo puede pertenecer a una categoría pero una categoría puede tener uno o muchos productos.

- Una relación Suministrar entre las entidades Proveedor y Producto ya que un proveedor suministra uno o muchos producto pero un producto es suministrado por un solo proveedor.
- Una relación de muchos a muchos entre las entidades Pedido y Producto ya que un producto puede estar en uno o muchos pedidos y un pedido puede tener uno o muchos productos.

2. **Realizar el modelo relacional**

Para comenzar, procedemos a convertir las entidades generadas y sus atributos en tablas.

Cliente
id_cliente
nombre_cliente
cedula_cliente
telefono_cliente
email_cliente
codigo_postal_cliente
password_cliente

Proveedor
NIT
nombre_proveedor
direccion_proveedor
telefono_proveedor
email_proveedor

Domiciliario
id_domiciiliario
nombre_domiciliario
placa_furgoneta_domiciliario
codigo_postal_zona_domiciliario
telefono_domicliario

Producto
id_producto
nombre_producto
marca_producto
origen_producto
dimensiones_producto
foto_producto
cantidad_producto
precio_producto

Categoria_producto
id_categoria
nombre_categoria
condiciones_almacenamiento_producto
observaciones_producto

Pedido
id_pedido
fecha_pedido
codigo_postal_cliente
codigo_postal_domiciliario
productos_pedido
valor_total_pedido
datos_pago_producto
direccion_pedido
validacion_zona_entrega

Una vez realizadas las tablas generamos las tablas intermedias que representan los atributos multivaluados.

Contacto_cliente
id_cliente
telefono_cliente

Cliente
id_cliente
nombre_cliente
cedula_cliente
email_cliente
codigo_postal_cliente
password_cliente

Producto
id_producto
nombre_producto
marca_producto
origen_producto
dimensiones_producto
foto_producto
cantidad_producto
precio_producto

Categoria_producto
id_categoria
nombre_categoria
id_condiciones_almacenamiento
observaciones_producto

Condiciones_almacenamiento
id_condiciones_almacenamiento
condicion_almacenamiento

Pedido
id_pedido
fecha_pedido
codigo_postal_cliente
codigo_postal_domiciliario
productos_pedido
valor_total_pedido
datos_pago_producto
direccion_pedido
validacion_zona_entrega

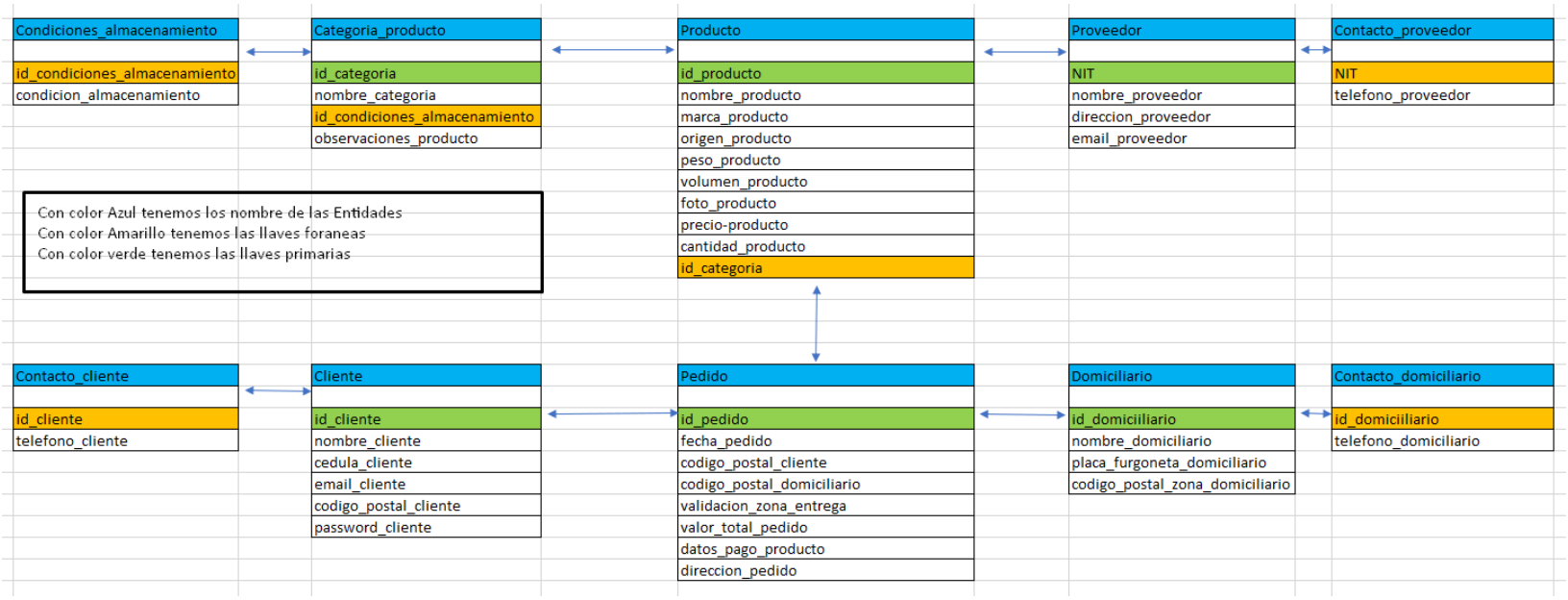
Proveedor
NIT
nombre_proveedor
direccion_proveedor
email_proveedor

Domiciliario
id_domiciiliario
nombre_domiciliario
placa_furgoneta_domiciliario
codigo_postal_zona_domiciliario

Contacto_proveedor
NIT
telefono_proveedor

Contacto_domiciliario
id_domiciiliario
telefono_domiciliario

Posteriormente validamos las llaves foráneas de cada tabla y procedemos a realizar las conexiones entre tablas:



3. **Escribir con sentencias SQL toda la definición de la base de datos.**

Teniendo en cuenta las tablas creadas previamente se procede a crear cada una de las tablas usando comandos SQL, para las los ID de las llaves primarias se utilizara un identificador único consistente en una o varias letras y un numero.

Inicialmente realizamos la creación de la base de datos, aquí mostraremos paso a paso la creación de la base de datos y adicionalmente se añadirá al repositorio el Script SQL con los comandos necesarios para su ejecución.

```
CREATE DATABASE IF NOT EXISTS TiendaDonPepe;


USE TiendaDonPepe;
```

Creacion base de datos

Posteriormente se realiza la creación de cada una de las tablas requeridas para la base de datos incluyendo una tabla intermedia debido a que entre la entidad Producto y la entidad Pedido se encuentra una relación de N-M y las tablas para guardar los atributos multivaluados de contacto y tipo de almacenamiento.

```sql
-- Creacion tabla Cliente
CREATE TABLE Cliente (
  Id_Cliente VARCHAR(6) PRIMARY KEY,
  Nombre_Cliente VARCHAR(60),
  Cedula_Cliente VARCHAR(30),
  Password_Cliente VARCHAR(30),
  Email_Cliente VARCHAR(80),
  Direccion_Cliente VARCHAR(130),
  Codigo_Postal_Cliente VARCHAR(30)
);
```

```sql
-- Creacion tabla Contacto_Cliente
CREATE TABLE Contacto_Cliente (
  Id_Cliente VARCHAR(6) PRIMARY KEY,
  Telefono_Cliente INT,
  CONSTRAINT IdCliente_Contacto FOREIGN KEY (Id_Cliente)
    REFERENCES Cliente (Id_Cliente)
);
```

Creacion tablas cliente y contacto_cliente

```sql
-- Creacion tabla Domiciliario
CREATE TABLE Domiciliario (
  Id_Domiciliario VARCHAR(6) PRIMARY KEY,
  Placa_Furgoneta_Domiciliario VARCHAR(90),
  Nombre_Domiciliario VARCHAR(30),
  Codigo_Postal_Domiciliario VARCHAR(30)
);
```

```sql
-- Creacion tabla Contacto_Domiciliario
CREATE TABLE Contacto_Domiciliario (
  Id_Domiciliario VARCHAR(6) PRIMARY KEY,
  Telefono_Domiciliario INT,
  CONSTRAINT IdDomiciliario_contacto FOREIGN KEY (Id_Domiciliario)
    REFERENCES Domiciliario (Id_Domiciliario)
);
```

Creacion tablas domiciliario y contacto_domiciliario

```sql
-- Creacion tabla Proveedor
CREATE TABLE Proveedor (
  NIT VARCHAR(6) PRIMARY KEY,
  Nombre_Proveedor VARCHAR(80),
  Direccion_Proveedor VARCHAR(130),
  Email_Proveedor VARCHAR(80)
);
```

```sql
-- Creacion tabla  Contacto_Proveedor
CREATE TABLE Contacto_Proveedor (
  NIT VARCHAR(6) PRIMARY KEY,
  Telefono_Proveedor INT,
  CONSTRAINT IdProveedor_contacto FOREIGN KEY (NIT)
    REFERENCES Proveedor (NIT)
);
```

Creacion tablas proveedor y contacto_proveedor

```sql
-- Creacion tabla Condiciones Almacenamiento
CREATE TABLE Condiciones_Almacenamiento (
  Id_Almacenamiento VARCHAR(6) PRIMARY KEY,
  Condiciones_Almacenamiento VARCHAR(30)
);
```

Creacion tablas condiciones_almacenamiento y categoría_producto

```sql
-- Creacion Tabla Categoria Producto
CREATE TABLE Categoria_Producto (
  Id_Categoria_Producto VARCHAR(6) PRIMARY KEY,
  Observaciones_Producto VARCHAR(500),
  Id_condiciones_Almacenamiento VARCHAR(6),
  CONSTRAINT IdAlmacenamiento_Categoria FOREIGN KEY (Id_condiciones_Almacenamiento)
    REFERENCES Condiciones_Almacenamiento (Id_Almacenamiento)
);
```

```sql
-- Creacion tabla Producto
CREATE TABLE Producto (
  ID_Producto VARCHAR(6) PRIMARY KEY,
  Id_Categoria_Producto VARCHAR(6),
  NIT VARCHAR(6),
  Id_Pedido VARCHAR(6),
  Foto_Producto VARCHAR(100),
  Unidad_Disponible_Producto INT,
  Volumen_Producto VARCHAR(30),
  Peso_Producto VARCHAR(30),
  Nombre_Producto VARCHAR(80),
  Precio_Producto VARCHAR(8),
  CONSTRAINT IdCategoria_Producto FOREIGN KEY (Id_Categoria_Producto)
    REFERENCES Categoria_Producto (Id_Categoria_Producto),
  CONSTRAINT NIT_Producto FOREIGN KEY (NIT)
    REFERENCES Proveedor (NIT)
);
```

Creacion tabla producto

```sql
-- Creacion tabla Pedido
CREATE TABLE Pedido (
  Id_Pedido VARCHAR(6) PRIMARY KEY,
  Id_Cliente VARCHAR(6),
  Id_Domiciliario VARCHAR(6),
  Fecha_Pedido VARCHAR(150),
  Unidades_Producto_Pedido INT,
  Valor_Total_Pedido VARCHAR(30),
  Numero_Tarjeta_Pago VARCHAR(30),
  Caducidad_Tarjeta_Pago VARCHAR(30),
  Direccion_Pedido VARCHAR(130),
  Codigo_Postal_Cliente VARCHAR(30),
  Codigo_Postal_Domiciliario VARCHAR(30),
  Verificacion_Zona_Pedido VARCHAR(30),
  CONSTRAINT IdCLiente_Pedido FOREIGN KEY (Id_Cliente)
    REFERENCES Cliente (Id_Cliente),
  CONSTRAINT IdDomiciliario_Pedido FOREIGN KEY (Id_Domiciliario)
    REFERENCES Domiciliario (Id_Domiciliario)
);
```

Creacion tabla pedido

```sql
-- Creacion tabla Producto_Pedido
CREATE TABLE Producto_Pedido (
  Id_Producto VARCHAR(6),
  Id_Pedido VARCHAR(6),
  PRIMARY KEY (Id_Producto,Id_Pedido),
  foreign key(Id_Pedido) references Pedido(Id_Pedido),
  foreign key(Id_Producto) references Producto(Id_Producto)
);
```

Creacion tabla intermedia producto_pedido

- **Escribir consultas que me permitan ver la información de cada tabla o de varias tablas (10).**

- **Primera consulta:**

```
-- Mostrar productos disponibles que tengan un precio mayor a 50
SELECT *
FROM Producto
WHERE Precio_Producto > 50;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ID_Producto	Id_Categoria_Producto	NIT	Foto_Producto	Unidad_Disponible_Producto	Volumen_Producto	Peso_Producto	Nombre_Producto	Precio_Producto
Prod10	Cat10	P10	http://lorempixel.com/640/200/nature/	1899	2 Aovb	7 atFb	Duck	57,38
Prod13	Cat13	P13	http://lorempixel.com/g/1366/768/fashion/	5340	8 NDYb	9.57 tJab	Koshihikari Rice	79,44
Prod14	Cat14	P14	http://lorempixel.com/640/480/people/	8996	6.91 xJkb	2 dPab	Pine Nut	76,00
Prod15	Cat15	P15	http://lorempixel.com/g/1920/1200/animals/	6885	6 tObb	4 ijub	Star Anise	74,64
Prod16	Cat16	P16	http://lorempixel.com/1366/768/sports/	2922	46.23 NQb	02.7 iWab	Feijoa	69,34
Prod19	Cat19	P19	http://lorempixel.com/1920/1200/city/	8333	1.82 kqDb	07.64 aStb	Rhubarb	73,43
Prod21	Cat21	P21	http://lorempixel.com/g/640/350/city/	3655	3 MpMb	6.58 dVbb	Dried Chinese Broccoli	76,75
Prod22	Cat22	P22	http://lorempixel.com/720/348/transport/	4652	44 yhPb	4 KYZb	Zucchini	94,77
Prod23	Cat23	P23	http://lorempixel.com/1024/768/people/	3283	6.62 Vtbb	9 xfab	Barramundi	98,87
Prod28	Cat28	P28	http://lorempixel.com/640/200/city/	1419	19 hGb	8 Ewab	Green Tea	91,15
Prod3	Cat3	P3	http://lorempixel.com/640/480/animals/	9849	23 FbCb	88 tgqb	Starfruit	80,30
Prod32	Cat32	P32	http://lorempixel.com/g/1024/768/cats/	9864	1 ilPb	5 mzb	Achacha	74,68
Prod34	Cat34	P34	http://lorempixel.com/1024/768/nightlife/	9838	9 GlQb	58.1 UDWb	Beetroot	73,57
Prod35	Cat35	P35	http://lorempixel.com/g/1600/1200/city/	9095	3.21 dGab	36 ENhb	Redfish	88,87
Prod36	Cat36	P36	http://lorempixel.com/g/640/200/fashion/	3572	22.5 nHab	9 mpab	Apple Juice	77,73
Prod37	Cat37	P37	http://lorempixel.com/1366/768/abstract/	2118	3.8 Kwxb	3 OLb	White Wine Vinegar	88,93
Prod39	Cat39	P39	http://lorempixel.com/g/1366/768/cats/	7513	7 uSqb	2.83 tejb	Canola Oil	93,76
Prod4	Cat4	P4	http://lorempixel.com/1366/768/city/	6864	75.8 aMBb	4.24 VHRb	Mastic	65,18
Prod43	Cat43	P43	http://lorempixel.com/1366/768/nightlife/	9029	6.23 zRab	0.3 RDab	Vanilla Beans	93,14

Con esta consulta podremos validar los productos disponibles en la tabla de “Producto” cuyo precio sea mayor a 50.

- **Segunda consulta**

```
-- Mostrar la unidades disponibles de un producto
SELECT Unidad_Disponible_Producto FROM Producto WHERE ID_Producto = 'Prod25';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Unidad_Disponible_Producto
8802

Usando esta consulta podemos verificar la cantidad de productos disponibles usando el id del producto.

- **Tercera consulta**

```
--     Seleccionar los productos que cumplan con ciertas condiciones de almacenamiento
SELECT Nombre_Producto FROM Producto
JOIN Categoria_Producto ON Producto.Id_Categoria_Producto = Categoria_Producto.Id_Categoria_Producto
JOIN Condiciones_Almacenamiento ON Categoria_Producto.Id_condiciones_Almacenamiento = Condiciones_Almacenamiento.Id_Almacenamient
WHERE Condiciones_Almacenamiento = 'Seco';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Nombre_Producto
Water
Kumera
Parrotfish
Cream Cheese
Watercress
Tapioca Flour
Chives
Incaberries
Parsley
Passionfruit
Pumpkin
Cheddar
Soy
Chives
Morwong
Pork
Marjoram
Bamboo Shoots

Con esta consulta podemos validar los productos en la lista que requieran una condición de almacenamiento especifica.

- **Cuarta consulta**

```
-- Seleccionar los productos que tienen un precio superior a 20 y que esten disponibles en la categoria de "Bebida"
SELECT * FROM Producto
JOIN Categoria_Producto ON Producto.Id_Categoria_Producto = Categoria_Producto.Id_Categoria_Producto
WHERE Precio_Producto > 20 AND Observaciones_Producto = 'Bebida';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ID_Producto	Id_Categoria_Producto	NIT	Foto_Producto	Unidad_Disponible_Producto	Volumen_Producto	Peso_Producto	Nombre_Producto	Precio_Producto	Id_Categoria_Producto	Observaciones_Producto	Id_condicion
Prod19	Cat19	P19	http://lorempixel.com/1920/1200/city/	8333	1.82 kqDb	07.64 aStb	Rhubarb	73,43	Cat19	Bebida	Cond47
Prod28	Cat28	P28	http://lorempixel.com/640/200/city/	1419	19 hGb	8 Ewab	Green Tea	91,15	Cat28	Bebida	Cond34
Prod3	Cat3	P3	http://lorempixel.com/640/480/animals/	9849	23 FbCb	88 tgqb	Starfruit	80,30	Cat3	Bebida	Cond25
Prod32	Cat32	P32	http://lorempixel.com/g/1024/768/cats/	9864	1 ilPb	5 mzb	Achacha	74,68	Cat32	Bebida	Cond9
Prod50	Cat50	P50	http://lorempixel.com/720/348/city/	1310	3 qLgb	60 Wmrb	Kenchur	70,93	Cat50	Bebida	Cond39

Esta consulta nos permite ver los productos que estén disponibles en la categoría “Bebida” y cuyo precio sea mayor a 20.

- **Quinta consulta**

```
-- Obtener el nombre y el correo electrónico de todos los proveedores cuyo nombre contenga caracteres especificos
SELECT Nombre_Proveedor, Email_Proveedor
FROM Proveedor
WHERE Nombre_Proveedor LIKE '%Pu%';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Nombre_Proveedor	Email_Proveedor
Puente S.L.	eva.varela@yahoo.com
Puga Farías S.A.	lorenzo.alarcon@hotmail.com
Pulido Hermanos	alfonso.valladares@hotmail.com
Amaya, Puga y Verdugo Asociados	cristian.collado@yahoo.com

Esta consulta nos permite validar el nombre y el email de los proveedores cuyo nombre contenga caracteres específicos.

- **Sexta Consulta**

```
-- Obtener el nombre y la cantidad disponible de los productos de una categoría específica usando su Id
SELECT Nombre_Producto, Unidad_Disponible_Producto
FROM Producto
JOIN Categoria_Producto
ON Producto.Id_Categoria_Producto = Categoria_Producto.Id_Categoria_Producto
WHERE Categoria_Producto.Id_Categoria_Producto = 'Cat15';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Nombre_Producto	Unidad_Disponible_Producto
Limes	2019

Con esta consulta podemos verificar el nombre y la cantidad de un producto utilizando el id de la categoría que le corresponde.

- **Séptima consulta**

```
-- Obtener el nombre y el numero telefonico de todos los domiciliarios
SELECT Nombre_Domiciliario, Telefono_Domiciliario
FROM Contacto_Domiciliario
JOIN Domiciliario
ON Contacto_Domiciliario.Id_Domiciliario = Domiciliario.Id_Domiciliario;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Nombre_Domiciliario	Telefono_Domiciliario
Adela Márquez Baca	7
Virginia Zelaya Velázquez	30349912
Homero Ulibarri Puente	8902
Claudio Cabrera Ortega	230990
Julio Mena Anaya	532
Natalia Castañeda Tórrez	57576030
Armando Mireles Zaragoza	722059627
Lucia Becerra Santacruz	8097957
Jerónimo Barela Montoya	4865413
Juana Delacrúz Serna	162148
Gabriela Sedillo Barreto	73287564
Mónica Sanabria Pedroza	47673750
Jorge Luis Roque Iglesias	7078
Vicente Sosa Sotelo	61905
Elisa Santillán Santacruz	9
Juan Castellanos Montaño	7416
Sr. Ignacio Villa Herrera	740340
Reina Ontiveros Reynoso	4556717

Con esta consulta podemos validar el nombre y el número telefónico de los domiciliarios registrados.

- **Octava consulta**

```sql
-- Ver las obsertvaciones de un producto, el nombre del proveedor que lo suministra y el precio del producto
SELECT Categoria_Producto.Observaciones_Producto, Proveedor.Nombre_Proveedor, Producto.Precio_Producto
FROM Producto
JOIN Categoria_Producto
ON Producto.Id_Categoria_Producto = Categoria_Producto.Id_Categoria_Producto
JOIN Proveedor
ON Producto.NIT = Proveedor.NIT;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Observaciones_Producto	Nombre_Proveedor	Precio_Producto
Alimento no perecedero	Aguilera Valdés S.A.	38,07
Producto de aseo	Gastélum Pelayo S.A.	58,32
Alimento perecedero	Curiel S.A.	45,51
Producto de aseo	Estévez, Soto y Medina Asociados	60,25
Bebida	Cortés Núñez Hermanos	96,23
Producto de aseo	Mota Apodaca e Hijos	69,21
Alimento perecedero	Limón Ramón S.A.	68,93
Alimento perecedero	Banda S.A.	80,65
Bebida	Carrasco Hermanos	60,32
Bebida	Ozuna S.L.	48,62
Alimento no perecedero	Yáñez y Cadena	28,24
Bebida	Henríquez e Hijos	86,88
Alimento perecedero	Sarabia Reséndez S.L.	20,49
Alimento perecedero	Jiménez y Orellana	38,44
Alimento no perecedero	Rentería, de Jesús y Caldera Aso...	79,60
Alimento no perecedero	Duarte, Preciado y Carbajal Asoci...	5,98
Alimento no perecedero	Brito y Gurule	35,10
Alimento no perecedero	Estévez, Carrion y Ruiz Asociados	50,55

Con esta consulta podemos validar el nombre del proveedor, el precio y las observaciones de un producto.

- **Novena consulta**

```sql
-- Ver la información de los clientes que tengan una dirección que contenga la palabra "puerta"
SELECT *
FROM Cliente
WHERE Direccion_Cliente LIKE '%puerta%';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Id_Cliente	Nombre_Cliente	Cedula_Cliente	Password_Cliente	Email_Cliente	Direccion_Cliente	Codigo_Postal_Cliente
C1	Carla Macías Collazo	702-15-5045	2zv76np3f700	juancarlos.guzman@yahoo.com	Puerta 904 Paseo Eloisa, 7 Puerta 789, Pamplo...	65845
C10	Carlota Valenzuela Galindo	825-84-7388	fmi0wsz1t	leticia.perea@yahoo.com	Esc. 656 Barrio Enrique 20 Puerta 328, Jerez de...	24038
C11	Sancho Solís Prieto	704-30-7716	8x3qw280y90aga5	arturo.osorio@gmail.com	Esc. 912 Vía Pública Hernán Calderón 06 Puerta...	33397
C16	María Elena Lozada Carreón	287-42-7246	830tty07w8y	luz.rosales@gmail.com	Glorieta Concepción, 11 Puerta 682, Ceuta, An...	63358
C19	Sta. Rafael Varela Delafuente	020-65-9916	6yvdwv35w3gitav	diego.quinonez@gmail.com	Puerta 092 Gran Subida María del Carmen Ocam...	31932
C20	Sra. Alfredo Borrego Cadena	877-82-0280	1wxrnj9ax0277me	hugo.paez@yahoo.com	Esc. 126 Apartamento Irene 46 Puerta 031, Ru...	78010
C21	José Emilio Tamayo Pabón	605-24-2595	yv64fgmy8	mariano.gomez@hotmail.com	Rambla Cristobal, 6 Puerta 523, Vigo, And 89529	40386
C25	María Teresa Leiva Lugo	241-74-6608	40nj87d7h8	angela.perales@hotmail.com	Puerta 908 Extrarradio Marco Antonio Tello, 28,...	88129
C26	Gabriela Malave Ruelas	447-06-9276	foba32b2d	nicolas.cepeda@hotmail.com	Puerta 179 Torrente Alfredo 7, Jerez de la Fron...	13637
C27	Antonio Zamora Guzmán	248-15-7118	3tcipu7a	horacio.heredia@yahoo.com	Manzana Blanca, 9 Puerta 049, Torrevieja, Man...	32260
C28	Sta. Sancho Enríquez Mascar...	145-79-6495	2a9tifsl	isabel.delrio@gmail.com	Puerta 039 Vía Pública Gabriel, 56 Puerta 568, S...	02181
C29	María Arteaga Gaytán	226-62-4141	x0a2c21cdr	elisa.samaniego@gmail.com	Puerta 968 Grupo Alfonso, 9, Cádiz, Man 60129	50051
C31	Sra. Estela Parra Salcido	768-61-0051	qdjvz48mmjt	magdalena.rosas@yahoo.com	Puerta Timoteo Cavazos 81, Parla, Vas 64072	81483
C33	Ramona Vergara Centeno	585-13-6271	wqkofnw5coaqyf	cristina.altamirano@yahoo.com	Puerta 871 Carretera Timoteo Valle 56 Puerta 7...	34481
C39	Débora Trujillo Cazares	734-16-1407	f989b09ujoa	sancho.rocha@yahoo.com	Partida Guillermina Atencio s/n. Puerta 356, Parl...	16407
C40	Alejandra Pagan Centeno	039-65-9200	y7dzuq2hcbw27y	jorgeluis.montanez@yahoo.com	Vía Pública Vicente Quesada s/n. Puerta 549, T...	90507
C41	Rocio Márquez Aparicio	386-42-6075	178txvdon0f	leticia.chavarria@yahoo.com	Puerta 765 Puerta Javier, 4, Sagunto, Cat 43340	87995
C42	Gonzalo Barrientos Barraza	235-73-7432	hmxc66kj80wxw0	concepcion.tapia@yahoo.com	Extrarradio Alejandra Valdivia, 6 Puerta 834, Or...	36783

- **Decima consulta**

```sql
-- Ver clientes que hayan realizado un pedido superior a 50000
SELECT Cliente.*
FROM Cliente
JOIN Pedido ON Cliente.Id_Cliente = Pedido.Id_Cliente
WHERE Pedido.Valor_Total_Pedido > '50000';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id_Cliente	Nombre_Cliente	Cedula_Cliente	Password_Cliente	Email_Cliente	Direccion_Cliente	Codigo_Postal_Cliente
C1	Carla Macías Collazo	702-15-5045	2zv76np3f700	juancarlos.guzman@yahoo.com	Puerta 904 Paseo Eloisa, 7 Puerta 789, Pamplo...	65845
C30	Rosario Carmona Holguín	072-73-3391	903tddwwdi	ramon.ortiz@hotmail.com	Muelle Javier Gamboa, 2 Esc. 477, Roquetas de...	81590
C31	Sra. Estela Parra Salcido	768-61-0051	qdjvz48mmjt	magdalena.rosas@yahoo.com	Puerta Timoteo Cavazos 81, Parla, Vas 64072	81483
C39	Débora Trujillo Cazares	734-16-1407	f989b09ujoa	sancho.rocha@yahoo.com	Partida Guillermina Atencio s/n. Puerta 356, Parl...	16407

Con esta consulta podemos validar que clientes han realizado pedidos de un monto superior al indicado.

- **Generar de 4 a 6 vistas donde se evidencie lo más importante de cada ejercicio (haga una selección muy responsable de la información realmente importante según el contexto).**

- **Primera vista**

```sql
-- Vista de la imformacion completa de los clientes
CREATE VIEW Vista_Clientes_Contactos AS
SELECT c.Id_Cliente, c.Nombre_Cliente, c.Cedula_Cliente, cc.Telefono_Cliente, c.Email_Cliente, c.Direc
FROM Cliente c
JOIN Contacto_Cliente cc ON c.Id_Cliente = cc.Id_Cliente;
SELECT * FROM tiendadonpepe.Vista_Clientes_Contactos;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id_Cliente	Nombre_Cliente	Cedula_Cliente	Telefono_Cliente	Email_Cliente	Direccion_Cliente	Codigo_Postal_Cliente
C1	Julio César Tórrez Venegas	708-96-9659	6528058	patricia.pena@yahoo.com	Riera Benjamín, 6 Puerta 502, Alcorcón, Ara 75...	32040
C10	Antonia Rivero Montoya	199-33-7537	7	clemente.chavez@gmail.com	Puerta 465 Puerta Leonor, 18, Cuenta, And 44...	13377
C11	Sta. Felipe Castellanos Gamez	233-92-1066	9944	cristian.terrazas@hotmail.com	Esc. 832 Partida Gilberto García, 4, Talavera de...	31243
C12	Marcela Serrato Huerta	502-72-5802	92	marialuisa.salazar@hotmail.com	Puerta 256 Arroyo Manuel s/n., Reus, Vas 29231	05438
C13	Diego Villa Méndez	561-22-7472	84	emilio.miranda@hotmail.com	Chalet María Elena 77, Baracaldo, Ast 70277	23081
C14	Olivia Barraza Briones	200-22-3822	4119085	susana.rincon@gmail.com	Polígono Esteban Blanco, 87, Sabadell, And 90833	91711
C15	Marcos Marroquín Domínguez	363-39-7046	90196	soledad.toledo@hotmail.com	Esc. 361 Barranco María Elena 16, Cartagena, ...	36025
C16	Federico Salcedo Alcala	675-80-3605	9	mateo.caban@yahoo.com	Prolongación Pablo Ozuna 80, Madrid, Leo 01060	31075
C17	Sr. Javier Camacho Guerrero	773-22-8639	6	rosario.diaz@gmail.com	Puerta 062 Torrente Lourdes, 73, Rivas-Vaciam...	92064
C18	Nicolás Olivárez Botello	636-65-3642	4246184	veronica.cardona@yahoo.com	Esc. 527 Partida Mariano, 7 Puerta 480, Segovi...	88269
C19	María Eugenia Arenas Luevano	568-39-0588	2188	magdalena.castaneda@yahoo...	Rambla Lorenzo Cortez 9 Puerta 296, Sanlúcar ...	67216
C2	Marisol Jaimes Borrego	555-67-9228	6543	estela.olivarez@yahoo.com	Puerta 179 Terrenos Manuel, 39, Avilés, Can 4...	31554
C20	Rafael Montes Caldera	307-47-3432	41	elisa.aguilera@hotmail.com	Cuesta Jaime Rodríguez, 62, Mérida, Cat 49698	49732
C21	Reina Colunga Acuña	025-41-6348	2337617	david.sauceda@hotmail.com	Esc. 088 Subida Horacio Martínez, 22 Esc. 753, ...	59864
C22	Ariadna Rolón Cornejo	559-84-1016	585	gustavo.pizarro@hotmail.com	Esc. 113 Arroyo Verónica Mata, 1 Puerta 684, ...	30573
C23	Ana Noriega Lovato	212-64-1236	86665309	mariateresa.mendez@yahoo.com	Terrenos Marco Antonio Solorzano 93, Sanlúcar...	48155
C24	Sra. Esteban Ferrer Villa	460-46-6713	4839	gonzalo.mateo@yahoo.com	Esc. 738 Barrio Pedro Henríquez 7 Puerta 242, ...	95208
C25	Verónica Bétancourt Delgado	439-54-7790	20	cesar.torrez@gmail.com	Rampa Pilar Zamora, 8 Puerta 708, Motril, Mad ...	61786
C26	Barbara Pineda Carrera	617-78-3995	85133	andrea.angulo@yahoo.com	Puerta 989 Aldea Ana Quintana 00 Puerta 080, ...	81587
C27	Reina Altamirano Galván	041-91-6604	999	daniel.vazquez@gmail.com	Torrente Gabriel Rivas 68 Puerta 319, Elche, M...	68221

Esta vista nos permite ver la información de los clientes de la tienda con sus números telefónicos y dirección de email.

- **Segunda vista**

```
CREATE VIEW Productos_Por_Proveedor AS
SELECT pr.Nombre_Proveedor, p.Nombre_Producto, p.Unidad_Disponible_Producto, p.Precio_Producto
FROM Proveedor pr
JOIN Producto p ON pr.NIT = p.NIT;
SELECT * FROM tiendadonpepe.Productos_Por_Proveedor;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Nombre_Proveedor	Nombre_Producto	Unidad_Disponible_Producto	Precio_Producto
Matos, Solano y Echevarría Asociados	Feijoa	4888	1,67
Bernal S.L.	Duck	1899	57,38
Luna y Heredia	Sour Dough Bread	5131	15,96
Raya e Hijos	Beef Stock	4496	13,28
Serrato, Caraballo y Leiva Asociados	Koshihikari Rice	5340	79,44
López y Alemán	Pine Nut	8996	76,00
Merino S.A.	Star Anise	6885	74,64
Nieves, Casárez y Zaragoza Asociados	Feijoa	2922	69,34
Benítez, Carranza y Raya Asociados	Red Wine Vinegar	6137	29,88
Duran, Menchaca y Gracia Asociados	Ham	3233	11,71
León Archuleta S.A.	Rhubarb	8333	73,43
Jurado, Leyva y Guillen Asociados	White Flour	6504	10,94
Bravo e Hijos	Wholewheat Flour	9314	30,75
Peralta y Mayorga	Dried Chinese Br...	3655	76,75
Bañuelos y Chávez	Zucchini	4652	94,77
Valdés S.L.	Barramundi	3283	98,87
Palacios Hermanos	Custard Apples	1214	17,64
Mateo y Rosado	Feta	6790	27,55
Almanza, Chapa y Montes Asociados	Tamari	1757	36,93
Rodrígez, Chapa y Barragán Asociados	Mahi Mahi	2314	23,78

En esta vista podemos validar el nombre del proveedor, el nombre del producto, las unidades disponibles en la tienda y su precio por unidad, de esta manera se puede validar el stock de un producto y su proveedor.

- **Tercera vista**

```
CREATE VIEW Vista_Productos_Categorias AS
SELECT p.ID_Producto, p.Nombre_Producto, p.Precio_Producto, c.Id_Categoria_Producto, c.Observaciones_Producto
FROM Producto p
INNER JOIN Categoria_Producto c ON p.Id_Categoria_Producto = c.Id_Categoria_Producto;
SELECT * FROM tiendadonpepe.Vista_Productos_Categorias;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ID_Producto	Nombre_Producto	Precio_Producto	Id_Categoria_Producto	Observaciones_Producto
Prod1	Feijoa	1,67	Cat1	Producto de aseo
Prod10	Duck	57,38	Cat10	Alimento perecedero
Prod11	Sour Dough Bread	15,96	Cat11	Bebida
Prod12	Beef Stock	13,28	Cat12	Alimento perecedero
Prod13	Koshihikari Rice	79,44	Cat13	Alimento no perecedero
Prod14	Pine Nut	76,00	Cat14	Alimento perecedero
Prod15	Star Anise	74,64	Cat15	Producto de aseo
Prod16	Feijoa	69,34	Cat16	Producto de aseo
Prod17	Red Wine Vinegar	29,88	Cat17	Producto de aseo
Prod18	Ham	11,71	Cat18	Alimento perecedero
Prod19	Rhubarb	73,43	Cat19	Bebida
Prod2	White Flour	10,94	Cat2	Producto de aseo
Prod20	Wholewheat Flour	30,75	Cat20	Producto de aseo
Prod21	Dried Chinese Br...	76,75	Cat21	Alimento perecedero
Prod22	Zucchini	94,77	Cat22	Alimento no perecedero
Prod23	Barramundi	98,87	Cat23	Producto de aseo
Prod24	Custard Apples	17,64	Cat24	Producto de aseo
Prod25	Feta	27,55	Cat25	Producto de aseo
Prod26	Tamari	36,93	Cat26	Producto de aseo

Esta vista nos permite visualizar información de un producto para conocer su nombre, precio y observaciones del tipo de producto.

- **Cuarta vista**

```
CREATE VIEW Vista_Validacion_Inventario AS
SELECT ID_Producto, Nombre_Producto, Unidad_Disponible_Producto, Precio_Producto
FROM Producto
WHERE Unidad_Disponible_Producto > 0;
SELECT * FROM tiendadonpepe.Vista_Validacion_Inventario;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ID_Producto	Nombre_Producto	Unidad_Disponible_Producto	Precio_Producto
Prod1	Feijoa	4888	1,67
Prod10	Duck	1899	57,38
Prod11	Sour Dough Bread	5131	15,96
Prod12	Beef Stock	4496	13,28
Prod13	Koshihikari Rice	5340	79,44
Prod14	Pine Nut	8996	76,00
Prod15	Star Anise	6885	74,64
Prod16	Feijoa	2922	69,34
Prod17	Red Wine Vinegar	6137	29,88
Prod18	Ham	3233	11,71
Prod19	Rhubarb	8333	73,43
Prod2	White Flour	6504	10,94
Prod20	Wholewheat Flour	9314	30,75
Prod21	Dried Chinese Br...	3655	76,75
Prod22	Zucchini	4652	94,77
Prod23	Barramundi	3283	98,87
Prod24	Custard Apples	1214	17,64
Prod25	Feta	6790	27,55
Prod26	Tamari	1757	36,93

Esta vista nos permite visualizar completamente el inventario de productos disponibles para vender, pudiendo visualizar el stock restante y el precio por unidad de cada producto.

- **Quinta vista**

```
CREATE VIEW Pedidos_Y_Detalles AS
SELECT Pedido.Id_Pedido, Pedido.Fecha_Pedido, Cliente.Nombre_Cliente, Domiciliario.Nombre_Domiciliario, Pedido.Valor_Total_Pedido, Pedido.Direccion_Pedido
FROM Pedido
JOIN Cliente ON Pedido.Id_Cliente = Cliente.Id_Cliente
JOIN Domiciliario ON Pedido.Id_Domiciliario = Domiciliario.Id_Domiciliario;
SELECT * FROM tiendadonpepe.Pedidos_Y_Detalles;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id_Pedido	Fecha_Pedido	Nombre_Cliente	Nombre_Domiciliario	Valor_Total_Pedido	Direccion_Pedido
Ped1	com.github.javafaker.DateAndTime@757b8c47	Julio César Tórrez Venegas	María Cristina Loera Pulido	202.79	Barranco Raquel Acuña 9 Esc. 966
Ped10	com.github.javafaker.DateAndTime@757b8c47	Antonia Rivero Montoya	Lucia Sanches Cantú	168.60	Municipio María Soledad Preciado 80
Ped11	com.github.javafaker.DateAndTime@757b8c47	Sta. Felipe Castellanos Gamez	Samuel Guillen Villagómez	184.74	Conjunto Santiago 77 Puerta 385
Ped12	com.github.javafaker.DateAndTime@757b8c47	Marcela Serrato Huerta	Pilar Villareal Sisneros	232.15	Vía Pública Esteban, 1
Ped14	com.github.javafaker.DateAndTime@757b8c47	Olivia Barraza Briones	Sancho Ortiz Mesa	65.44	Rampa Inés Pedraza, 30
Ped15	com.github.javafaker.DateAndTime@757b8c47	Marcos Marroquín Domínguez	Laura Rojo Delgado	404.79	Riera Arturo Fernández 0
Ped16	com.github.javafaker.DateAndTime@757b8c47	Federico Salcedo Alcala	Elvira Vega Escamilla	128.56	Camino Mateo Cardona 50 Esc. 141
Ped17	com.github.javafaker.DateAndTime@757b8c47	Sr. Javier Camacho Guerrero	Esteban Matos Soliz	434.60	Explanada Bernardo Rangel 5
Ped18	com.github.javafaker.DateAndTime@757b8c47	Nicolás Olivárez Botello	Anita Sedillo Correa	58.16	Ramal Ramón Valencia 37 Esc. 617
Ped19	com.github.javafaker.DateAndTime@757b8c47	María Eugenia Arenas Luevano	Margarita Castillo Cedillo	249.87	Mercado Dorotea Corona, 6 Puerta 476
Ped2	com.github.javafaker.DateAndTime@757b8c47	Marisol Jaimes Borrego	Sta. Jerónimo Lozada Nor...	285.50	Calleja Bernardo Vélez s/n.
Ped20	com.github.javafaker.DateAndTime@757b8c47	Rafael Montes Caldera	José Reynoso Pineda	125.73	Huerta Gloria Alvarez, 1 Esc. 556
Ped21	com.github.javafaker.DateAndTime@757b8c47	Reina Colunga Acuña	Andrea Arellano Olivárez	60.31	Solar José Eduardo Tijerina 42
Ped22	com.github.javafaker.DateAndTime@757b8c47	Ariadna Rolón Cornejo	Benito Negrete Archuleta	457.62	Gran Subida Guillermina Amador s/n. P...
Ped23	com.github.javafaker.DateAndTime@757b8c47	Ana Noriega Lovato	Hernán Ramón Manzanares	73.7	Plaza Claudio Negrón 87
Ped24	com.github.javafaker.DateAndTime@757b8c47	Sra. Esteban Ferrer Villa	Sta. Miguel Melgar Ybarra	483.13	Senda Gabriel Fonseca 57

Esta vista nos permite visualizar en su totalidad la información correspondiente a los pedidos realizados.

- **Generar al menos 4 procedimientos almacenados**.

- **Primer procedimiento**

```
-- Procedimiento para regitrar clientes nuevos
DELIMITER //
CREATE PROCEDURE Registrar_Cliente(
    IN Id_Cliente VARCHAR(6),
    IN Nombre_Cliente VARCHAR(60),
    IN Cedula_Cliente VARCHAR(30),
    IN Password_Cliente VARCHAR(30),
    IN Email_Cliente VARCHAR(80),
    IN Direccion_Cliente VARCHAR(130),
    IN Codigo_Postal_Cliente VARCHAR(30)
)
BEGIN
    INSERT INTO Cliente(Id_Cliente, Nombre_Cliente, Cedula_Cliente, Password_Cliente, Email_Cliente, Direccion_Cliente, Codigo_Postal_Cliente)
    VALUES(Id_Cliente, Nombre_Cliente, Cedula_Cliente, Password_Cliente, Email_Cliente, Direccion_Cliente, Codigo_Postal_Cliente);
END//
DELIMITER ;
CALL Registrar_cliente ('C60', 'Andres perez', '8374892', 'djsfghlashdg', 'aperez@gmail.com', 'Calle falsa 123', '47663');
SELECT * FROM tiendadonpepe.cliente;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Id_Cliente	Nombre_Cliente	Cedula_Cliente	Password_Cliente	Email_Cliente	Direccion_Cliente	Codigo_Postal_Cliente
C50	Jacobo Domínquez Delao	057-78-9958	71ydkcx8	juliocesar.zamora@gmail.com	Chalet Luz, 66 Esc. 491, Parla, Gal 16564	56803
C6	Mercedes Aranda Cadena	161-83-2318	r443qsm16zm1g0	julia.escalante@hotmail.com	Puerta 719 Sección Berta León 30, Baracaldo, N...	21811
C60	Andres perez	8374892	djsfghlashdg	aperez@gmail.com	Calle falsa 123	47663

Con este procedimiento podemos registrar un nuevo cliente en la tabla de clientes agregando la información requerida.

- **Segundo procedimiento**

```
-- Procedimiento para ingresar nuevos domiciliarios
DELIMITER //
CREATE PROCEDURE Registrar_Domiciliario(
    IN Id_Domiciliario VARCHAR(6),
    IN Placa_Furgoneta_Domiciliario VARCHAR(90),
    IN Nombre_Domiciliario VARCHAR(30),
    IN Codigo_Postal_Domiciliario VARCHAR(30)
)
BEGIN
    INSERT INTO Domiciliario(Id_Domiciliario, Placa_Furgoneta_Domiciliario, Nombre_Domiciliario, Codigo_Postal_Domiciliario)
    VALUES(Id_Domiciliario, Placa_Furgoneta_Domiciliario, Nombre_Domiciliario, Codigo_Postal_Domiciliario);

END//
DELIMITER ;
CALL Registrar_Domiciliario ('D60', 'GQD-712', 'djsfghlashdg', 'Andres lopez', '47663');
SELECT * FROM tiendadonpepe.domiciliario;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Id_Domiciliario	Placa_Furgoneta_Domiciliario	Nombre_Domiciliario	Codigo_Postal_Domiciliario
D5	HYR-689	Marisol Girón Noriega	01880
D50	JQY-708	Felipe Girón Calderón	62181
D6	DCY-636	Gloria Ferrer Velásquez	11558
D60	GQD-712	djsfghlashdg	Andres lopez
D7	OVZ-609	Sr. Mateo Villaseñor Balderas	34183

Este procedimiento nos permite agregar domiciliarios a la tabla domiciliario.

- **Tercer procedimiento**

```
CREATE PROCEDURE Modificar_Cliente(
  IN p_Id_Cliente VARCHAR(6),
  IN p_Nombre_Cliente VARCHAR(60),
  IN p_Cedula_Cliente VARCHAR(30),
  IN p_Password_Cliente VARCHAR(30),
  IN p_Email_Cliente VARCHAR(80),
  IN p_Direccion_Cliente VARCHAR(130),
  IN p_Codigo_Postal_Cliente VARCHAR(30)
)
BEGIN
  UPDATE Cliente
  SET Nombre_Cliente = p_Nombre_Cliente,
    Cedula_Cliente = p_Cedula_Cliente,
    Password_Cliente = p_Password_Cliente,
    Email_Cliente = p_Email_Cliente,
    Direccion_Cliente = p_Direccion_Cliente,
    Codigo_Postal_Cliente = p_Codigo_Postal_Cliente
  WHERE Id_Cliente = p_Id_Cliente;
END//
DELIMITER ;
CALL Modificar_Cliente('C60','Juan lopez', '73463287', 'dgfdsfgh', 'juanlopez@hotmail.com', 'Calle de la alegria 123', '234532');
SELECT * FROM tiendadonpepe.cliente;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Id_Cliente	Nombre_Cliente	Cedula_Cliente	Password_Cliente	Email_Cliente	Direccion_Cliente	Codigo_Postal_Cliente
C5	Germán Córdova Cruz	476-61-4647	qc1adg0kzhlzvo	pablo.adorno@gmail.com	Senda Ernesto Delafuente s/n. Esc. 958, Grana...	13526
C50	Jacobo Domínquez Delao	057-78-9958	71ydkcx8	juliocesar.zamora@gmail.com	Chalet Luz, 66 Esc. 491, Parla, Gal 16564	56803
C6	Mercedes Aranda Cadena	161-83-2318	r443qsm16zm1g0	julia.escalante@hotmail.com	Puerta 719 Sección Berta León 30, Baracaldo, N...	21811
C60	Juan lopez	73463287	dgfdsfgh	juanlopez@hotmail.com	Calle de la alegria 123	234532

Con este procedimiento podemos modificar la información de uno de los clientes registrados para actualizar el registro ya presente.

- **Cuarto procedimiento**

```
-- Procedimiento para modificar un domiciliario registrado
DELIMITER //
CREATE PROCEDURE Modificar_Domiciliario(
  IN p_Id_Domiciliario VARCHAR(6),
  IN p_Placa_Furgoneta_Domiciliario VARCHAR(90),
  IN p_Nombre_Domiciliario VARCHAR(30),
  IN p_Codigo_Postal_Domiciliario VARCHAR(30)
)
BEGIN
  UPDATE Domiciliario
  SET Placa_Furgoneta_Domiciliario = p_Placa_Furgoneta_Domiciliario,
    Nombre_Domiciliario = p_Nombre_Domiciliario,
    Codigo_Postal_Domiciliario = p_Codigo_Postal_Domiciliario
  WHERE Id_Domiciliario = p_Id_Domiciliario;
END//
DELIMITER ;
CALL Modificar_Domiciliario ('D60', 'AAA-000', 'Pepe lopez', '09263');
SELECT * FROM tiendadonpepe.domiciliario;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Id_Domiciliario	Placa_Furgoneta_Domiciliario	Nombre_Domiciliario	Codigo_Postal_Domiciliario
D50	JQY-708	Felipe Girón Calderón	62181
D6	DCY-636	Gloria Ferrer Velásquez	11558
D60	AAA-000	Pepe lopez	09263

Con este procedimiento podemos modificar un registro de un domiciliario ya creado con anterioridad.

- **Generar al menos 4 triggers**

- **Primer Trigger**

Para empezar crearemos las tablas necesarias para las notificaciones de los trigger.

```
298  CREATE TABLE Notificaciones (
299      Id_Notificacion INT AUTO_INCREMENT  PRIMARY KEY,
300      Mensaje_Notificacion VARCHAR(150),
301      Fecha_Notificacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP
302  );
306  CREATE TABLE IF NOT EXISTS Registros_Cliente (
307  Usuario_Cambio VARCHAR(90) NOT NULL,
308  Cambio_Realizado VARCHAR(150) NOT NULL,
309  Fecha_Cambio DATETIME NOT NULL,
310  PRIMARY KEY (Usuario_Cambio, Cambio_Realizado, Fecha_Cambio)
311  );
```

En estas tablas guardaremos las notificaciones generadas por los trigger. Ahora creamos el primer trigger.

```
321  -- Notificar la existencia de un nuevo pedido
322  DELIMITER //
323  CREATE TRIGGER Nuevo_Pedido
324  AFTER INSERT ON Pedido
325  FOR EACH ROW
326  BEGIN
327      DECLARE nuevoMensaje VARCHAR(150);
328      SET nuevoMensaje = CONCAT('Un nuevo ha sido creado con el ID: ', NEW.Id_Pedido);
329      INSERT INTO Notificaciones (Mensaje_Notificacion) VALUES (nuevoMensaje);
330  END
331  //DELIMITER ;
332  INSERT INTO `tiendadonpepe`.`pedido`
333  (`Id_Pedido`,`Id_Cliente`,`Id_Domiciliario`,`Fecha_Pedido`,`Unidades_Producto_Pedido`,`Valor_Total_Pedido`,`Numero_Tarjeta_
334  VALUES
335  ('Ped60', 'C35', 'D25', '15-02-2023', '6', '6735', '3333-3333-3333-3333','12-25', 'Calle falsa 123','45736','45736','Si');
336  SELECT * FROM tiendadonpepe.notificaciones;
337
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Id_Notificacion	Mensaje_Notificacion	Fecha_Notificacion
1	Un nuevo ha sido creado con el ID: Ped60	2023-02-19 16:00:45

Este trigger genera un mensaje en la tabla de Notificaciones cada vez que un nuevo pedido es creado y agregado a la tabla, para realizar la prueba insertamos un pedido nuevo y comprobamos la tabla.

- **Segundo trigger**

```
-- Trigger para registrar el ingreso de nuevos clientes
DELIMITER //
CREATE TRIGGER Nuevo_Cliente
AFTER INSERT ON Cliente
FOR EACH ROW
BEGIN
 INSERT INTO Registros_Cliente (Usuario_Cambio, Cambio_Realizado, Fecha_Cambio)
 VALUES (USER(), "Agrego un nuevo cliente en", NOW());
END//
DELIMITER ;
CALL Registrar_Cliente('C61','Carlo pedro martinez', '73424287', 'dgfdgh', 'carlopedromartinez@gmail.com', 'Calle de la alegria 13', '23452');
SELECT * FROM tiendadonpepe.registros_cliente;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Usuario_Cambio	Cambio_Realizado	Fecha_Cambio
root@localhost	Agrego un nuevo cliente en	2023-02-19 16:08:24
NULL	NULL	NULL

Este trigger genera un mensaje en la tabla de Registros_Cliente cada vez que un nuevo cliente es agregado a la tabla, para realizar la prueba hacemos uso del procedimiento creado anteriormente.

- **Tercer trigger**

```
-- Trigger para validar que un usuario no haya sido previamente registrado
DELIMITER //
CREATE TRIGGER check_existing_client BEFORE INSERT ON Cliente
FOR EACH ROW
BEGIN
  IF EXISTS (SELECT * FROM Cliente WHERE Id_Cliente = NEW.Id_Cliente OR Email_Cliente = NEW.Email_Cliente) THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El cliente ya existe.';
  END IF;
END;
//DELIMITER ;
CALL Registrar_Cliente('C61','Carlo pedro martinez', '73424287', 'dgfdgh', 'carlopedromartinez@gmail.com', 'Calle de la alegria 13', '23452')
```

Output

Action Output

#	Time	Action	Message
1	16:09:34	CREATE TRIGGER check_existing_client BEFORE INSERT ON Cliente FOR EACH ROW BEGIN IF EXISTS (SELECT * FROM Cliente WHERE Id_Cliente...	0 row(s) affected
2	16:09:34	CALL Registrar_Cliente('C61','Carlo pedro martinez', '73424287', 'dgfdgh', 'carlopedromartinez@gmail.com', 'Calle de la alegria 13', '23452')	Error Code: 1644. El cliente ya existe.

Este trigger nos mostrara una notificación cada vez que se quiere agregar un registro a la tabla Cliente que ya exista dentro de la misma con el fin de evitar registros duplicados. Para hacer la prueba usamos el mismo procedimiento y registro usado en el ejemplo anterior.

- **Cuarto trigger**

```
-- Trigger para descontar del stock
DELIMITER //
CREATE TRIGGER Restar_Stock
AFTER INSERT ON Producto_Pedido
FOR EACH ROW
UPDATE Producto
JOIN Producto_Pedido ON Producto.ID_Producto = Producto_Pedido.Id_Producto
JOIN Pedido ON Pedido.Id_Pedido = Producto_Pedido.Id_Pedido
SET Producto.Unidad_Disponible_Producto = Producto.Unidad_Disponible_Producto - Pedido.Unidades_Producto_Pedido
WHERE Producto_Pedido.Id_Producto = NEW.Id_Producto AND Producto_Pedido.Id_Pedido = NEW.Id_Pedido;
//
DELIMITER ;
```

Inicialmente creamos el trigger que nos va a disminuir el stock de un producto al confirmase que se recibió un pedido de este.

ID_Producto	Id_Categoria_Producto	NIT	Foto_Producto	Unidad_Disponible_Producto	Volumen_Producto	Peso_Producto	Nombre_Producto	Precio_Producto
Prod1	Cat1	P1	http://lorempixel.com/1280/1024/business/	4844	78 asIb	0 sSnb	Feijoa	1,67
Prod10	Cat10	P10	http://lorempixel.com/640/200/nature/	1899	2 Aovb	7 atFb	Duck	57,38
Prod11	Cat11	P11	http://lorempixel.com/640/350/transport/	5131	48 IYvb	00.9 rdab	Sour Dough Bread	15,96
Prod12	Cat12	P12	http://lorempixel.com/640/200/transport/	4496	99.3 prNb	91.8 Ueb	Beef Stock	13,28
Prod13	Cat13	P13	http://lorempixel.com/g/1366/768/fashion/	5340	8 NDYb	9.57 tJab	Koshihikari Rice	79,44
Prod14	Cat14	P14	http://lorempixel.com/640/480/people/	8996	6.91 xJkb	2 dPab	Pine Nut	76,00
Prod15	Cat15	P15	http://lorempixel.com/g/1920/1200/animals/	6885	6 tObb	4 ijub	Star Anise	74,64
Prod16	Cat16	P16	http://lorempixel.com/1366/768/sports/	2922	46.23 NQb	02.7 iWab	Feijoa	69,34
Prod17	Cat17	P17	http://lorempixel.com/g/720/348/abstract/	6137	5 oQbb	7.5 nlab	Red Wine Vinegar	29,88
Prod18	Cat18	P18	http://lorempixel.com/640/200/animals/	3233	9 VQVb	88.25 Gifb	Ham	11,71
Prod19	Cat19	P19	http://lorempixel.com/1920/1200/city/	8333	1.82 kqDb	07.64 aStb	Rhubarb	73,43

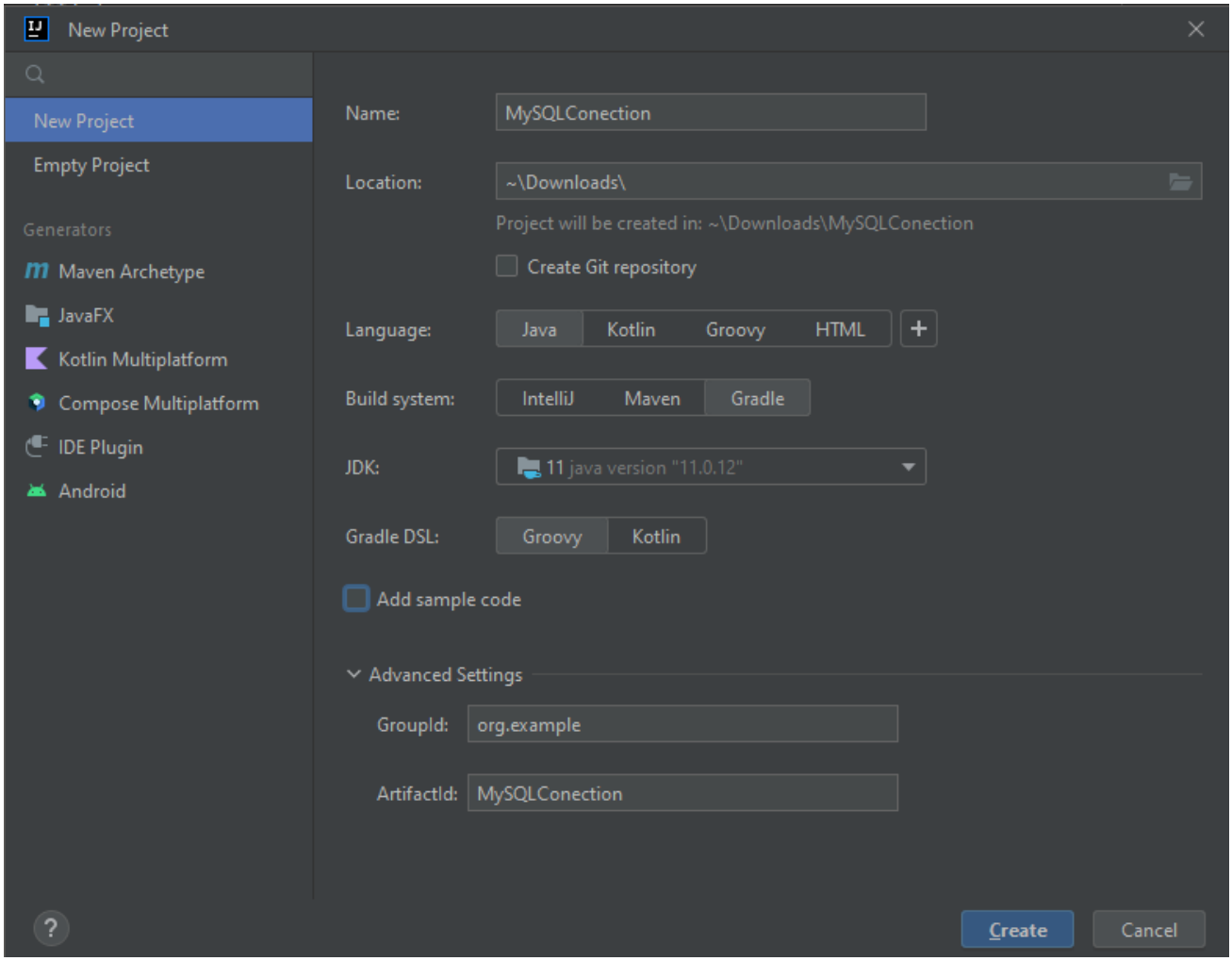
Validamos en la tabla Producto las unidades disponibles del producto antes de generar un pedido nuevo.

ID_Producto	Id_Categoria_Producto	NIT	Foto_Producto	Unidad_Disponible_Producto	Volumen_Producto	Peso_Producto	Nombre_Producto	Precio_Producto
Prod1	Cat1	P1	http://lorempixel.com/1280/1024/business/	4796	78 asIb	0 sSnb	Feijoa	1,67
Prod10	Cat10	P10	http://lorempixel.com/640/200/nature/	1899	2 Aovb	7 atFb	Duck	57,38
Prod11	Cat11	P11	http://lorempixel.com/640/350/transport/	5131	48 IYvb	00.9 rdab	Sour Dough Bread	15,96
Prod12	Cat12	P12	http://lorempixel.com/640/200/transport/	4496	99.3 prNb	91.8 Ueb	Beef Stock	13,28
Prod13	Cat13	P13	http://lorempixel.com/g/1366/768/fashion/	5340	8 NDYb	9.57 tJab	Koshihikari Rice	79,44
Prod14	Cat14	P14	http://lorempixel.com/640/480/people/	8996	6.91 xJkb	2 dPab	Pine Nut	76,00
Prod15	Cat15	P15	http://lorempixel.com/g/1920/1200/animals/	6885	6 tObb	4 ijub	Star Anise	74,64
Prod16	Cat16	P16	http://lorempixel.com/1366/768/sports/	2922	46.23 NQb	02.7 iWab	Feijoa	69,34
Prod17	Cat17	P17	http://lorempixel.com/g/720/348/abstract/	6137	5 oQbb	7.5 nlab	Red Wine Vinegar	29,88
Prod18	Cat18	P18	http://lorempixel.com/640/200/animals/	3233	9 VQVb	88.25 Gifb	Ham	11,71
Prod19	Cat19	P19	http://lorempixel.com/1920/1200/city/	8333	1.82 kqDb	07.64 aStb	Rhubarb	73,43

Una vez se ha generado un nuevo pedido las unidades disponibles del producto han disminuido dependiendo de la cantidad solicitada.

- **Poblar la base de datos (50 registros por tabla) utilizando una conexión desde Java.**

Para poder realizar esto primero hacemos la creación de un proyecto desde el IDE Intellij.



Una vez nombrado el proyecto procedemos agregando las dependencias requeridas para este proyecto, en este caso serán las dependencias de MySQL connector y Java Faker.

```
dependencies {
    testImplementation 'org.junit.jupiter:junit-jupiter-api:5.8.1'
    testRuntimeOnly 'org.junit.jupiter:junit-jupiter-engine:5.8.1'
    // https://mvnrepository.com/artifact/com.mysql/mysql-connector-j
    implementation group: 'com.mysql', name: 'mysql-connector-j', version: '8.0.32'
// https://mvnrepository.com/artifact/com.github.javafaker/javafaker
    implementation group: 'com.github.javafaker', name: 'javafaker', version: '1.0.2'
}
```

Una vez agregadas las dependencias al proyecto procedemos a crear una clase donde se guardaran las constantes de conexión necesarias.

```java
package com.sofkau.integration.database.mysql;

no usages
public class MySqlConstants {
    no usages
    public static final String My_SQL_JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";
    no usages
    public static final String CONNECTION_STRING="jdbc:mysql://%s/%s?user=%s&password=%s";
}
```

Teniendo las constantes creadas procedemos a generar una interfaz donde crearemos los métodos necesarios para la conexión a la DB y el envio de los registros.

```java
import java.sql.SQLException;

no usages
public interface DataBase {

    no usages
    public void configureDataBaseConnection();

    no usages
    public void executeSqlStatement();

    no usages
    public ResultSet getResultset();

    no usages
    public void close();

    no usages
    public void printResultset() throws SQLException;

}
```

Una vez terminada la creación de la interfaz procedemos a crear una nueva clase donde se implementaran los métodos necesarios para la conexión y el envio de los registros.

```
no usages
public class MySqlOperation implements DataBase {

    no usages
    private Connection connection= null;
    no usages
    private Statement statement= null;
    no usages
    private Resultset resultset=null;
    no usages
    @Override
    public void configureDataBaseConnection() {

    }

    no usages
    @Override
    public void executeSqlStatement() {

    }

    no usages
    @Override
    public ResultSet getResultset() {
        return null;
    }

    no usages
    @Override
    public void close() {

    }

    no usages
    @Override
    public void printResultset() throws SQLException {

    }
}
```

```
public class MySqlOperation implements DataBase {

    no usages
    private Connection connection= null;
    no usages
    private Statement statement= null;
    no usages
    private Resultset resultset=null;
    no usages
    private String sqlStatement;
    no usages
    private String server;
    no usages
    private String dataBaseName;
    no usages
    private String user;
    no usages
    private String password;
```

Se realiza la implementación de los métodos para continuar con la conexión a la DB.

```
@Override
public void configureDataBaseConnection() {
    try {
        Class.forName(My_SQL_JDBC_DRIVER);
        connection = DriverManager.getConnection(
                String.format(CONNECTION_STRING,
                        this.server,
                        this.dataBaseName,
                        this.user,
                        this.password));
        statement = connection.createStatement();
    } catch (Exception e) {
        System.out.println(e.getMessage());
    }
}
```

Se genera el metodo que nos permitira la conexion a la DB.

El método siguiente nos permitirá enviar las sentencias SQL a la DB.

```
no usages
@Override
public void executeSqlStatement() {
    try {
        configureDataBaseConnection();
        resultSet = statement.executeQuery(sqlStatement);
    } catch (Exception e) {
        System.out.println(e.getMessage());
    }
}

11 usages
@Override
public void executeSqlStatementVoid() {
    try{
        configureDataBaseConnection();
        statement.execute(sqlStatement);
    }catch(Exception e){
        System.out.println(e.getMessage());
    }

}
```

Haremos uso de ellos para realizar consultas y el envio de los registros a la DB.

Seguimos con la implementación de los métodos.

```
1 usage
@Override
public void close() {
    try {
        if (resultSet != null) {
            resultSet.close();
        }
        if (statement != null) {
            statement.close();
        }
        if (connection != null) {
            connection.close();
        }
    } catch (Exception e) {
        System.out.println(e.getMessage());
    }
}
```

Nos permitira cerrar la conexion

Una vez finalizamos ya podemos empezar el envio de registros a la DB, para esto haremos uso de Java faker para enviar los datos de forma automática.

Inicialmente crearemos constantes que tengan los campos de cada registro con el fin de agregarlos haciendo uso de String.format

```java
public class Main {
    1 usage
    private static final String SERVER = "localhost";
    12 usages
    private static final String DATA_BASE_NAME = "TiendaDonPepe";
    1 usage
    private static final String USER = "root";
    1 usage
    private static final String PASSWORD = "Revolution1";
    1 usage
    private static final String INSERT_CLIENTE = String.format("INSERT INTO %s.Cliente VALUES (%s, %s, %s, %s, %s, %s, %s)", DATA_BASE_NAME, "'C%d'", "'%s'", "'%s'", "'%s'", "'%s'", "'%s'", "'%s'
    1 usage
    private static final String INSERT_CONTACTO_CLIENTE = String.format("INSERT INTO %s.Contacto_Cliente VALUES (%s, %s)", DATA_BASE_NAME, "'C%d'", "'%s'");
    1 usage
    private static final String INSERT_DOMICLIARIO = String.format("INSERT INTO %s.Domiciliario VALUES (%s, %s, %s, %s)", DATA_BASE_NAME, "'D%d'", "'%s'", "'%s'", "'%s'");
    1 usage
    private static final String INSERT_CONTACTO_DOMICLIARIO = String.format("INSERT INTO %s.Contacto_Domiciliario VALUES (%s, %s)", DATA_BASE_NAME, "'D%d'", "'%s'");
    1 usage
    private static final String INSERT_PROVEEDOR = String.format("INSERT INTO %s.Proveedor VALUES (%s, %s, %s, %s)", DATA_BASE_NAME, "'P%d'", "'%s'", "'%s'", "'%s'");
    1 usage
    private static final String INSERT_CONTACTO_PROOVEDOR = String.format("INSERT INTO %s.Contacto_Proveedor VALUES (%s, %s)", DATA_BASE_NAME, "'P%d'", "'%s'");
    1 usage
    private static final String INSERT_COND_ALMACENAMIENTO = String.format("INSERT INTO %s.Condiciones_Almacenamiento VALUES (%s, %s)", DATA_BASE_NAME, "'Cond%d'", "'%s'");
    1 usage
    private static final String INSERT_PRODUCTO = String.format("INSERT INTO %s.Producto VALUES (%s, %s, %s, %s, %s ,%s ,%s ,%s,%s)", DATA_BASE_NAME, "'Prod%d'", "'Cat%s'", "'P%s'","'%s'","'%s'",
    1 usage
    private static final String INSERT_PEDIDO = String.format("INSERT INTO %s.Pedido VALUES (%s, %s, %s, %s, %s, %s ,%s ,%s ,%s,%s,%s,%s)", DATA_BASE_NAME, "'Ped%d'", "'C%s'", "'D%s'", "'%s'","'%
    1 usage
    private static final String INSERT_CATEGORIA = String.format("INSERT INTO %s.Categoria_Producto VALUES (%s, %s, %s)", DATA_BASE_NAME, "'Cat%d'", "'%s'", "'Cond%s'");
    1 usage
    private static final String INSERT_PROD_PEDIDO = String.format("INSERT INTO %s.Producto_Pedido VALUES (%s, %s)", DATA_BASE_NAME, "'Prod%d'", "'Ped%s'");
```

Una vez tenemos las constantes creadas usaremos Java faker para poblar las tablas de la DB.

```java
public static void insertClientes() {
    Faker faker = new Faker(new Locale( language: "es"));
    for (int i = 1; i <= 50; i++) {
        int id = i;
        String nombre = faker.name().fullName();
        String cedula = faker.idNumber().valid();
        String direccion = faker.address().fullAddress();
        String password = faker.internet().password();
        String codigoPostal = faker.address().zipCode();
        String email = faker.internet().emailAddress();
        String query = String.format(INSERT_CLIENTE, id, nombre, cedula, password, email, direccion, codigoPostal);
        mySqlOperation.setSqlStatement(query);
        System.out.println(query);
        mySqlOperation.executeSqlStatementVoid();
    }
}
1 usage
public static void insertContactoCliente() {
    Faker faker = new Faker(new Locale( language: "es"));
    for (int i = 1; i <= 50; i++) {
        int id = i;
        int telefono = (int) faker.number().randomNumber();
        String query = String.format(INSERT_CONTACTO_CLIENTE, id, telefono);
        mySqlOperation.setSqlStatement(query);
        System.out.println(query);
        mySqlOperation.executeSqlStatementVoid();
    }

}
```

```java
public static void insertDomiciliario() {
    Faker faker = new Faker(new Locale( language: "es"));
    for (int i = 1; i <= 50; i++) {
        int id = i;
        String nombre = faker.name().fullName();
        String placa = faker.regexify("[A-Z]{3}-[0-9]{3}");
        String codigoPostal = faker.address().zipCode();
        String query = String.format(INSERT_DOMICLIARIO, id, placa, nombre, codigoPostal);
        mySqlOperation.setSqlStatement(query);
        System.out.println(query);
        mySqlOperation.executeSqlStatementVoid();
    }
}
```

1 usage

```java
public static void insertContactoDomiciliario() {
    Faker faker = new Faker(new Locale( language: "es"));
    for (int i = 1; i <= 50; i++) {
        int id = i;
        int telefono = (int) faker.number().randomNumber();
        String query = String.format(INSERT_CONTACTO_DOMICLIARIO, id, telefono);
        mySqlOperation.setSqlStatement(query);
        System.out.println(query);
        mySqlOperation.executeSqlStatementVoid();
    }
}
```

1 usage

```java
public static void insertProveedor() {
    Faker faker = new Faker(new Locale( language: "es"));
    for (int i = 1; i <= 50; i++) {
        int id = i;
        String nombre = faker.company().name();
        String direccion = faker.address().fullAddress();
        String email = faker.internet().emailAddress();
        String query = String.format(INSERT_PROVEEDOR, id, nombre, direccion, email);
        mySqlOperation.setSqlStatement(query);
        System.out.println(query);
        mySqlOperation.executeSqlStatementVoid();
    }
}
```

```java
public static void insertContactoProveedor() {
    Faker faker = new Faker(new Locale( language: "es"));
    for (int i = 1; i <= 50; i++) {
        int id = i;
        int telefono = (int) faker.number().randomNumber();
        String query = String.format(INSERT_CONTACTO_PROOVEDOR, id, telefono);
        mySqlOperation.setSqlStatement(query);
        System.out.println(query);
        mySqlOperation.executeSqlStatementVoid();
    }
}
1 usage
public static void insertCondicionalmacenamiento() {
    Faker faker = new Faker(new Locale( language: "es"));
    for (int i = 1; i <= 50; i++) {
        int id = i;
        String almacenamiento = faker.options().option( ...options: "Frio", "Seco", "Congelado");
        String query = String.format(INSERT_COND_ALMACENAMIENTO, id, almacenamiento);
        mySqlOperation.setSqlStatement(query);
        System.out.println(query);
        mySqlOperation.executeSqlStatementVoid();
    }
}
1 usage
public static void insertProducto() {
    Faker faker = new Faker(new Locale( language: "es"));
    for (int i = 1; i <= 50; i++) {
        int id = i;
        String foto = faker.internet().image();
        int unidadesProducto= faker.number().numberBetween(1000, 10000);
        String volumen= faker.regexify("\\d{1,2}(\\.\\d{1,2})? [a-zA-Z]{2,3}\\b");
        String peso = faker.regexify("\\d{1,2}(\\.\\d{1,2})? [a-zA-Z]{2,3}\\b");
        String nombre= faker.food().ingredient();
        String precio= faker.commerce().price();
        String query = String.format(INSERT_PRODUCTO, id, id, id, foto, unidadesProducto, volumen, peso, nombre,precio);
        mySqlOperation.setSqlStatement(query);
        System.out.println(query);
        mySqlOperation.executeSqlStatementVoid();
    }
}
```

Ejecutamos desde Main y podemos validar que los datos son agregados satisfactoriamente.

```
INSERT INTO TiendaDonPepe.Proveedor VALUES ('P48', 'Salcido Botello S.A.', 'V◆a Juan 8 Puerta 185, L◆rida, Com 15715', 'mariajose.terrazas@yahoo.com')
INSERT INTO TiendaDonPepe.Proveedor VALUES ('P49', 'T◆llez Camarillo e Hijos', 'Esc. 519 Subida Gustavo Salda◆a 8, Badalona, Leo 59790', 'jeronimo.arroyo@gmail.co
INSERT INTO TiendaDonPepe.Proveedor VALUES ('P50', 'Brito Alvarez Hermanos', 'Poblado Jos◆ Alvarado, 16, Badalona, Leo 41228', 'diana.patino@gmail.com')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P1', '43')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P2', '262')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P3', '886')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P4', '8884334')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P5', '83030020')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P6', '613')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P7', '7908')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P8', '767965')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P9', '5366')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P10', '9488730')
INSERT INTO TiendaDonPepe.Contacto_Proveedor VALUES ('P11', '3')
```

- **Al terminar el ejercicio responda ¿ Está conforme con el resultado obtenido según el contexto o cree que hubiera obtenido un mejor resultado con una base de datos no relacional?**

Como la tienda es pequeña me parece que es mas practico utilizar una base de datos relacional ya que no se manejaran grandes cantidades de información, si fuera para un negocio mas grande como un almacén de cadena o Grandes Superficies si consideraría útil el tener una DB no relacional.