

DeliverUS

Proyecto de Introducción a la Ingeniería del Software y los
Sistemas de Información I

Contenido

DeliverUS	0
1. Introducción al problema	
	2
2. Glosario de términos	
	3
3. Visión general del sistema	
	4
3.1. Requisitos generales	4
3.2. Usuarios del sistema	5
4. Catálogo de requisitos	
	6
4.1. Mapa de historias de usuario	6
4.2. Requisitos de información	7
4.3. Reglas de negocio	9
4.4. Requisitos funcionales	10
4.5. Requisitos no funcionales	12
5. Pruebas de aceptación	
	14
5.1. Reglas de aceptación de reglas de negocio	14
6. Modelado conceptual	16
6.1. Escenarios de prueba	17
7. Matriz de trazabilidad	18
8. Modelado Relacional en 3FN	19
9. Modelo Tecnológico	20
9.1 Creación de tablas, restricciones	20
9.2 Creación de funciones, procedimientos	23
9.3 Poblado de la base de datos	25
9.4 Cursores y listados de consultas	27
9.5 Creación de triggers	29

1. Introducción al problema

Hoy día el ritmo de vida que seguimos es demasiado “rápido”, cada vez las personas disponen de menos tiempo libre, esto hace que sacar tiempo para hacer la comida o desplazarse a comer a nuestros restaurantes favoritos sea prácticamente imposible. Nuestro objetivo es facilitar esa tarea y tener la comida lista en un par de clicks. Para ello se necesita de una aplicación que tenga como objetivo mitigar dicho problema, es aquí donde surge la empresa DeliverUS.

Nuestra aplicación está diseñada para ofrecer servicios a aquellos quienes están interesados en realizar pedidos a los restaurantes.

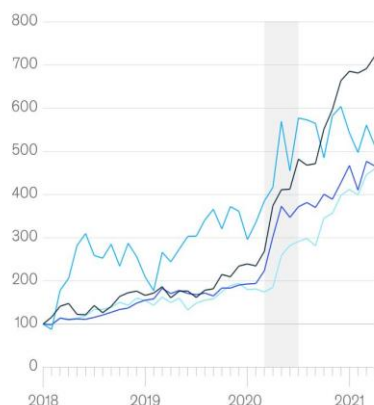
Para impulsar el negocio y mejorar el servicio a los clientes, se han mantenido una serie de entrevistas entre las partes interesadas, ya sean propietarios de los restaurantes o clientes y se han acordado los objetivos y requisitos generales que se describen en este documento.

A continuación, se desarrollarán los requisitos del sistema atendiendo a las necesidades por parte de los usuarios del sistema y se proporcionará un glosario con conceptos clave para poder tener una comunicación fluida con el cliente.

Gráfica del crecimiento de las ventas a través de plataformas de reparto:

Gráfica del nivel de estrés de las personas en sus trabajos:

Crecimiento
normalizado
de las ventas
de plataformas
de reparto, índice
(enero de 2018 = 100)



Fuente: Edison Trends

Canadá
Australia
Estados Unidos
Reino Unido

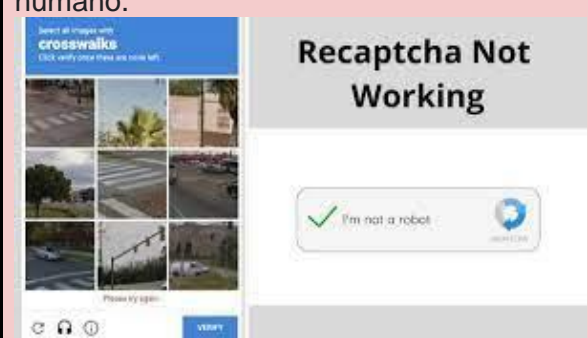
Los empleados, cada vez más estresados

Porcentaje de empleados que aseguran haber experimentado estrés durante gran parte del día anterior



Basado en encuestas a al menos 1,000 empleados por país en más de 100 países.
Fuente: Gallup

2. Glosario de términos

Término	Descripción
DeliverUS	Empresa que se dedica a gestionar pedidos de clientes a distintos restaurantes.
Devolución	Proceso en el cual se realiza el reembolso del coste del pedido.
Login	Proceso que controla el acceso individual a un sistema informático mediante la identificación del usuario utilizando credenciales provistas por el usuario.
Pedido	Encargo de género que realiza el cliente a un restaurante.
reCAPTCHA	<p>Es un sistema diseñado para diferenciar entre humanos y ordenadores, que se utiliza para que los bots no puedan completar formularios de forma malintencionada en nombre de un ser humano.</p>  The image shows a reCAPTCHA interface. On the left, there is a grid of nine small images, each containing a crosswalk. Above the grid, the text reads 'Select all images with crosswalks' and 'Click verify once these are done.' Below the grid, it says 'Please try again.' and 'verify'. On the right, there is a grey box with the text 'Recaptcha Not Working' in bold. Below this, there is a green checkmark icon and the text 'I'm not a robot'. To the right of this is a circular arrow icon with the text 'refresh'.

3. Visión general del sistema

3.1. Requisitos generales

OBJ-1: Usuarios:

Quiero distinguir entre los distintos tipos de usuarios que va a tener la aplicación.

Para realizar un correcto seguimiento sobre ellos mediante el login.

OBJ-2: Restaurantes:

Quiero tener registrados los restaurantes de la aplicación.

Para realizar un correcto seguimiento sobre ellos.

OBJ-3: Productos:

Quiero tener registrados los productos de los diversos restaurantes de la aplicación.

Para realizar un correcto seguimiento sobre ellos.

OBJ-4: Pedidos:

Quiero tener registrados los pedidos que se hacen dentro de la aplicación.

Para realizar un correcto seguimiento sobre ellos.

3.2. Usuarios del sistema

Usuarios	Descripción
Gerente de un restaurante	Se encarga de gestionar el estado de un restaurante, desde si está en funcionamiento hasta los productos que dicho restaurante ofrece.
Cliente	Sólo puede hacer pedidos dentro del sistema.

4. Catálogo de requisitos

4.1. Mapa de historias de usuario

Control de pedidos

Como gerente de un restaurante **quiero** que cada vez que se realice un pedido correspondiente a mi restaurante me llegue una alerta, **para** llevar un control de los pedidos.

Mejor experiencia

Como cliente **quiero** disponer de una respuesta inmediata, fluidez en la aplicación y un buen soporte ante los posibles problemas que surjan, **para** disfrutar de la mejor experiencia.

Experiencia y atención

Como gerente **quiero** poder visualizar, controlar y seguir sugerencias, quejas, pedidos etc... **para** poder dar la mejor experiencia y soporte al usuario.

4.2. Requisitos de información

RI-1: Usuarios:

Quiero que se distingan 2 tipos de usuarios: los gerentes de los restaurantes y clientes. Para ellos, la información requerida es:

- Nombre.
- Apellidos.
- Email.
- Teléfono.
- Dirección.
- Código postal.

Para un mejor seguimiento de los usuarios dentro de la aplicación mediante el login con su usuario y contraseña correspondientes.

RI-2: Restaurantes:

Quiero que se almacene la siguiente información sobre los restaurantes:

- Nombre.
- Descripción.
- Dirección.
- Código postal.
- URL.
- Correo electrónico.
- Número de teléfono.
- Gastos de envío (por defecto para los pedidos realizados a cada restaurante)
- Tiempo medio de servicio en minutos (que se calculará a partir del registro de pedidos) y estado.
- El estado de un restaurante representa si está aceptando pedidos, si no está disponible o si está cerrado temporal o permanentemente.
- El gerente que gestiona cada restaurante.
- Su categoría. Existen categorías de restaurantes predefinidas en el sistema.

Para la correcta gestión de los distintos restaurantes que se ofrecen.

RI-3: Productos:

Quiero almacenar la siguiente información con respecto a los productos vendidos por los restaurantes:

- Nombre.
- Descripción.
- Precio.
- Disponibilidad.

Para disponer de un conocimiento y control pleno sobre los productos que se ofrecen.

RI-4: Pedidos:

Quiero almacenar la siguiente información sobre los pedidos realizados por los clientes:

- Fecha de creación (cuando el cliente realiza el pedido).
- Fecha de inicio (cuando un restaurante acepta el pedido).
- Fecha de envío (cuando el pedido sale del restaurante).
- Fecha de entrega (cuando el cliente recibe el pedido).
- Precio total de los productos incluidos.
- La dirección donde debe ser entregado.
- Los gastos de envío.
- El sistema tiene que almacenar la cantidad de cada producto incluido en el pedido y el precio unitario de cada producto en el momento de hacer el pedido.
- Cada pedido incluirá un conjunto de productos de un restaurante concreto. Los pedidos no pueden incluir productos de más de un restaurante.

Para una gestión completa y correcta de los mismos.

4.3. Reglas de negocio

RN-1 Política de envío

- El envío será gratuito para aquellos pedidos mayores de 10 euros.

RN-2 Limitaciones en pedidos

- Un pedido sólo puede incluir productos de un restaurante, es decir, no se pueden combinar productos de distintos restaurantes.
- No se puede realizar un pedido a un restaurante que esté cerrado y/o marcado como inactivo.

RN-3 Política de pago

- Solamente se podrá pagar con efectivo (dando la cantidad justa de dinero sin la posibilidad de recibir cambio), con PayPal o con tarjeta.

RN-4 Política de reembolso

- Si el tiempo de entrega del pedido supera la hora, el cliente recibirá el reembolso.

4.4. Requisitos funcionales

RF-1 Estado de los restaurantes

Como cliente.

Quiero poder ver el estado de todos los restaurantes.

RF-2 Total pedido de un usuario

Como cliente.

Quiero poder ver el coste total de mi pedido, esto es , la suma del coste de todos los productos de mi pedido además de los gastos de envío.

RF-3 Número de unidades vendidas de un determinado producto

Como gerente.

Quiero saber cuántas unidades se han vendido de un determinado producto mediante un filtrado por el Id de dicho producto.

RF-4 Tiempo de reparto del pedido

Como cliente.

Quiero saber cuánto tiempo exactamente ha tardado mi pedido en llegar, calculándose como una resta entre la fecha de llegada y la fecha de envío del pedido.

RF-5 Lista de restaurantes filtrados

Como cliente.

Quiero poder ver una lista con los restaurantes que sean filtrados según su categoría.

RF-6 Eliminar un pedido

Quiero poder eliminar un pedido en caso de que ocurra algún error en la creación de este.

4.5. Requisitos no funcionales

RNF-1 Tiempo de respuesta del sistema

Quiero que el tiempo de respuesta en atención al cliente sea menor a 1 minuto si hay más de 500 usuarios concurrentes,

para mejorar la experiencia del cliente.

RNF-2 Interfaz de los usuarios de la aplicación

Quiero que haya un tipo de interfaz diferente según el tipo de usuario del sistema, es decir, que cada usuario tenga un apartado específico dentro de la aplicación,

para mejorar la experiencia del usuario

RNF-3 Mayor nivel de seguridad

Quiero que la cuenta sea verificada con una foto del DNI, selfie y resolver un reCAPTCHA,

para aumentar la seguridad de los usuarios.

RNF-4 Tiempo de funcionamiento del sistema

Quiero que el sistema esté activo de manera ininterrumpida,

para mejorar la experiencia del usuario.

RNF-5 Compatibilidad

Quiero que la aplicación funcione tanto en ordenador como en dispositivos móviles,

para mejorar la experiencia del usuario.

RNF-6 Atención al cliente

Quiero que el sistema ofrezca atención al cliente de 6:00 a 24:00 horas, festivos incluidos.

para mejorar la experiencia del usuario.

5. Pruebas de aceptación

5.1. Reglas de aceptación de reglas de negocio

PA-01 Prueba de aceptación de las políticas de envíos:

- *El envío será gratuito para aquellos pedidos que superen los 10 euros. Si el cliente realiza un pedido donde la suma total de los productos supera los 10 euros, se le notificará con un mensaje donde se le informa de que los gastos de envío son gratuitos.*
- *Si el envío tarda más de una hora, no se cobrará al cliente. En el momento en el que se finaliza el pedido se pone en marcha un contador, si este llegase a 0 y el pedido no ha sido entregado, el cliente recibirá un mensaje donde se le informa de que recibirá un reembolso por el retraso en la entrega del pedido*

PA-02 Prueba de aceptación de las limitaciones en pedidos:

- *El pedido solo puede incluir productos de un restaurante. En el momento en el que el cliente esté realizando el pedido, si añade a la lista un producto el cual pertenece a un restaurante distinto al de los productos anteriores, recibirá una alerta en la que se le notifica de que no puede finalizar el pedido y que deberá eliminar el producto de la lista.*
- *No se pueda realizar un pedido de un restaurante que se encuentre cerrado y/o marcado como inactivo. Si el cliente realiza un pedido cuyos productos pertenecen a un restaurante el cual se encuentra cerrado o inactivo, recibirá una alerta donde se le notificará de ello.*

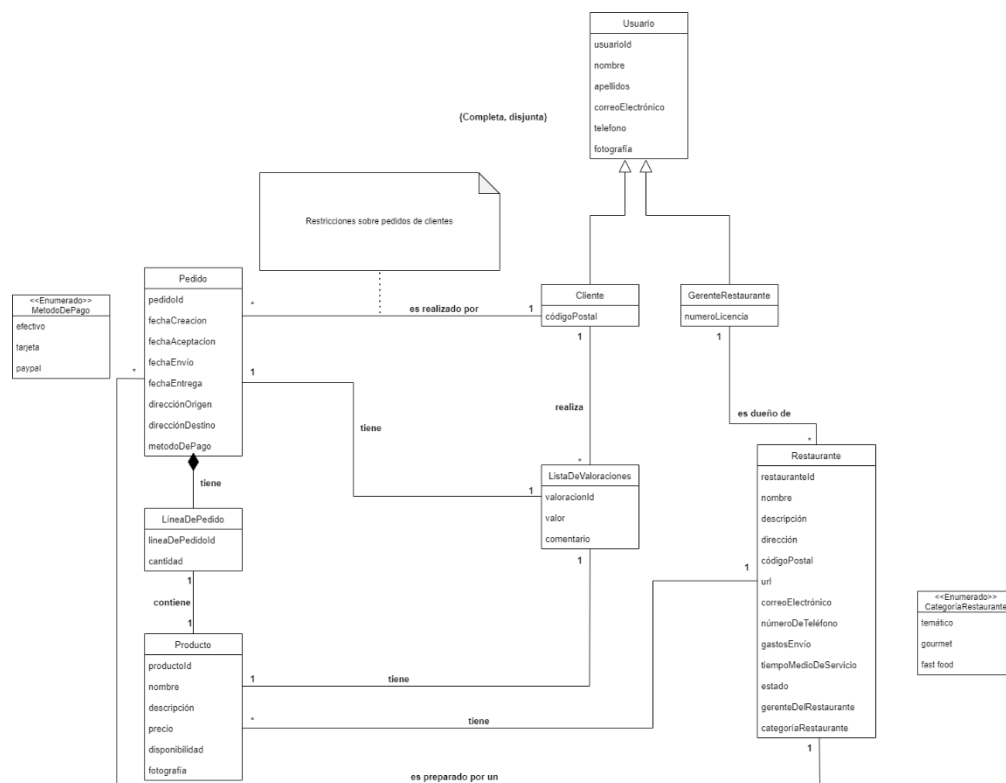
PA-03 Prueba de aceptación de las políticas de pago:

- *Solamente se podrá pagar con efectivo, con PayPal o con tarjeta. Al finalizar el pedido el cliente dispondrá de un apartado en el cual pueda elegir el método de pago. Si éste elige efectivo se deberá notificar al cliente con un mensaje en el cual se le recuerde que el dinero que se le entrega al repartidor debe ser la cantidad justa, sin posibilidad de recibir cambio.*

PA-04 Prueba de aceptación de las políticas de reembolso:

- *Si existiese algún error en el pedido, el cliente recibirá un reembolso. Si al realizar el pedido el cliente puede finalizarlo sin problemas, y en su gestión se encuentran algunos de ellos como, por ejemplo, restaurante inactivo o cerrado, producto que ya no está en carta etc.... El cliente recibirá un reembolso por el valor de la compra.*
- *Si el tiempo de entrega del pedido supera la hora, el cliente recibirá el reembolso. Si el pedido tarda más de una hora en llegar, se aplicará un reembolso del coste del pedido, en el caso de que se haya elegido como método de pago el efectivo, el repartidor no le pedirá el dinero.*

6. Modelado conceptual

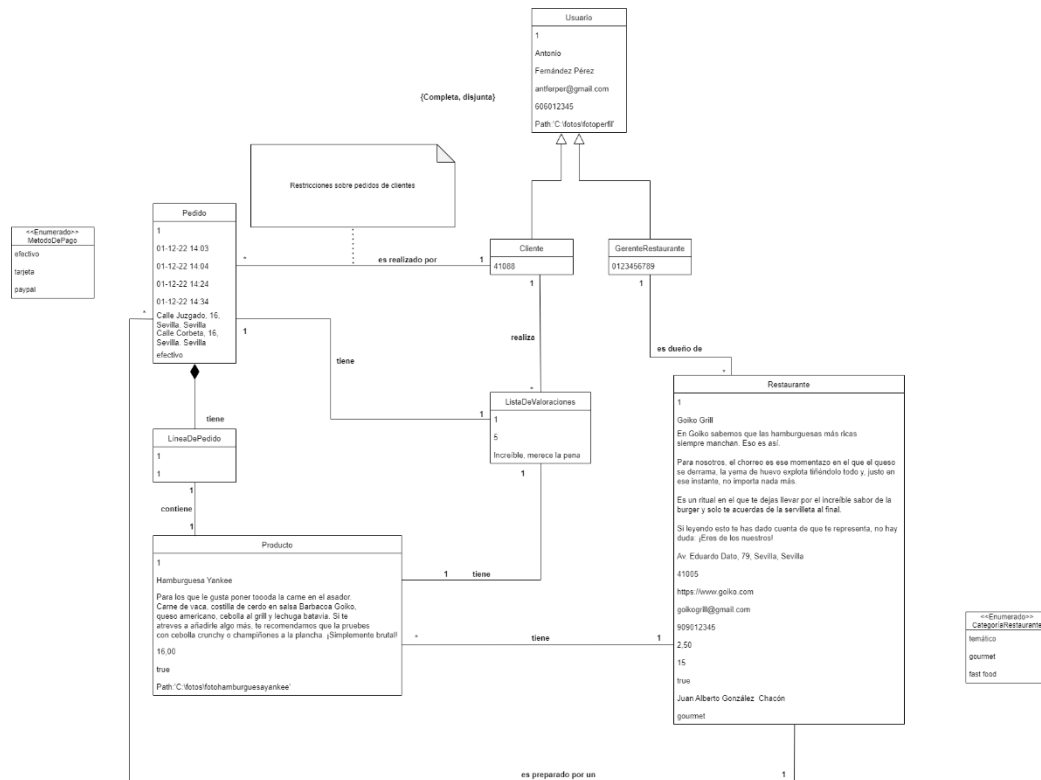


Por errores con el formato, se exponen las siguientes restricciones que no se podían apreciar en el UML:

Restricciones sobre pedidos de clientes:

- R-1: Un pedido sólo puede incluir productos de un restaurante, es decir, no se pueden combinar productos de distintos restaurantes.
- R-2: No se puede realizar un pedido a un restaurante que esté cerrado y/o marcado como inactivo.

6.1. Escenarios de prueba

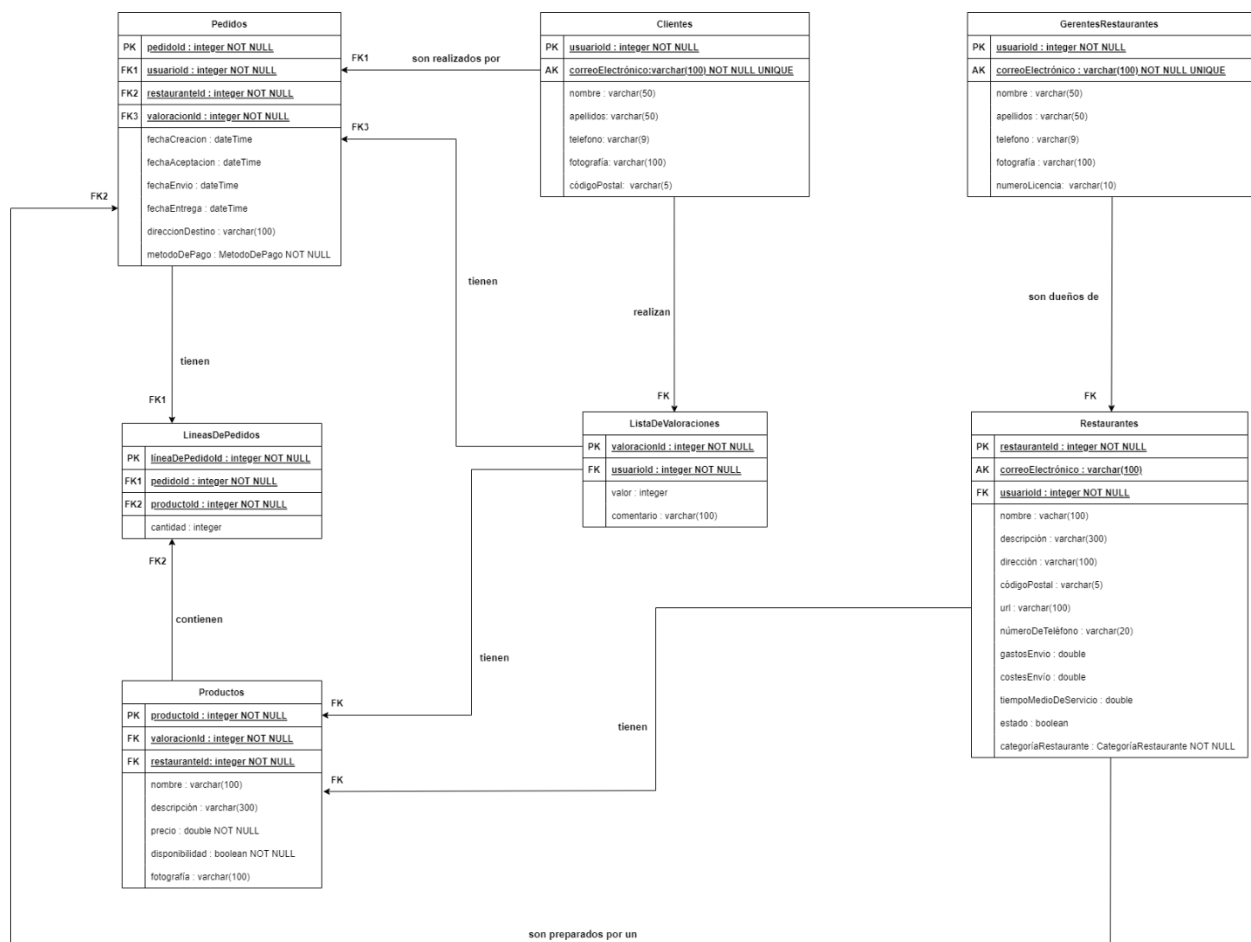


En este escenario de prueba, el usuario *Antonio Fernández Pérez* realiza un pedido al restaurante “*Goiko*”, en el diagrama se observan los datos almacenados de cada clase desde los datos del usuario hasta los detalles del pedido, así como los detalles del restaurante y su posterior valoración al pedido.

7. Matriz de trazabilidad

	Usuario	Cliente	GerenteRestaurante	Pedido	LineaDePedido	Producto	ListaDeValoraciones	Restaurante	es realizado por	tiene	realiza	es dueño de	tiene	es preparado por un	contiene	tiene	tiene	R-1	R-2
RI-1	X	X	X						X		X	X							
RI-2	X		X	X				X				X		X			X	X	X
RI-3				X	X	X	X						X		X		X		
RI-4	X	X		X	X		X	X	X	X				X		X		X	X
RN-1	X			X															
RN-2				X		X		X	X								X	X	X
RN-3	X	X		X															
RN-4	X	X		X															

8. Modelado Relacional en 3FN



Hemos seguido la estrategia de hacer una tabla por subclase ya que era la más conveniente según los apuntes vistos en clase de teoría.

9. Modelo Tecnológico

9.1 Creación de tablas, restricciones

```
DROP TABLE IF EXISTS lineasDePedidos;
DROP TABLE IF EXISTS productos;
DROP TABLE IF EXISTS pedidos;
DROP TABLE IF EXISTS listaDeValoraciones;
DROP TABLE IF EXISTS restaurantes;
DROP TABLE IF EXISTS gerentesRestaurantes;
DROP TABLE IF EXISTS clientes;
```

```
CREATE TABLE clientes(
    clienteId INT NOT NULL AUTO_INCREMENT,
    correoElectronico VARCHAR(100) NOT NULL UNIQUE,
    nombre VARCHAR(50) NOT NULL,
    apellidos VARCHAR(50) NOT NULL,
    telefono VARCHAR(9) NOT NULL,
    fotografia VARCHAR(100) NOT NULL,
    codigoPostal VARCHAR(5) NOT NULL,
    PRIMARY KEY(clienteId),
    UNIQUE(correoElectronico)
);
```

```
CREATE TABLE gerentesRestaurantes(
    gerenteId INT NOT NULL AUTO_INCREMENT,
    correoElectronico VARCHAR(100) NOT NULL UNIQUE,
    nombre VARCHAR(50) NOT NULL,
    apellidos VARCHAR(50) NOT NULL,
    telefono VARCHAR(9) NOT NULL,
    fotografia VARCHAR(100) NOT NULL,
    numeroLicencia VARCHAR(10),
    PRIMARY KEY(gerenteId),
    UNIQUE(correoElectronico)
);
```

```
CREATE TABLE restaurantes(
    restauranteId INT NOT NULL AUTO_INCREMENT,
    nombre VARCHAR(100),
    descripcion VARCHAR(300),
    direccion VARCHAR(100),
    codigoPostal VARCHAR(5),
    url VARCHAR(100),
    numeroDeTelefono VARCHAR(20),
    gastosEnvio DOUBLE NOT NULL,
    tiempoMedioDeServicio INT,
    estado BOOLEAN,
    categoriaRestaurante ENUM('temático','gourmet','fast food'),
    correoElectronico VARCHAR(100) NOT NULL UNIQUE, -- Clave alternativa
    gerenteId INT NOT NULL,
    PRIMARY KEY(restauranteId),
    UNIQUE(correoElectronico),
```

```

FOREIGN KEY (gerenteId) REFERENCES gerentesRestaurantes(gerenteId)

);

CREATE TABLE listaDeValoraciones(
    valoracionId INT NOT NULL AUTO_INCREMENT,
    valor INT,
    comentario VARCHAR(300),
    clienteId INT NOT NULL,
    PRIMARY KEY(valoracionId) ,
    FOREIGN KEY(clienteId) REFERENCES clientes(clienteId),
    CONSTRAINT numeroDeValoracionInvalido CHECK (valor >= 0 AND valor <=
5)
);

CREATE TABLE pedidos(
    pedidoId INT NOT NULL AUTO_INCREMENT,
    fechaCreacion DATETIME NOT NULL,
    fechaAceptacion DATETIME NOT NULL,
    fechaEnvio DATETIME NOT NULL,
    fechaEntrega DATETIME NOT NULL,
    direccionDestino VARCHAR(100) NOT NULL,
    metodoDePago ENUM('tarjeta','efectivo','paypal'),
    clienteId INT NOT NULL,
    restauranteId INT NOT NULL,
    valoracionId INT NOT NULL,
    PRIMARY KEY(pedidoId),
    FOREIGN KEY(clienteId) REFERENCES clientes(clienteId),
    FOREIGN KEY(restauranteId) REFERENCES restaurantes(restauranteId),
    FOREIGN KEY(valoracionId) REFERENCES
listaDeValoraciones(valoracionId),
    CONSTRAINT fechaIncorrecta CHECK
(TIMEDIFF(fechaCreacion,fechaAceptacion) < '00:00:00' AND
TIMEDIFF(fechaAceptacion,fechaEnvio) < '00:00:00' AND
TIMEDIFF(fechaEnvio,fechaEntrega) < 0)
);

CREATE TABLE productos(
    productoId INT NOT NULL AUTO_INCREMENT,
    nombre VARCHAR(100),
    descripcion VARCHAR(300),
    precio DOUBLE NOT NULL,
    disponibilidad BOOLEAN NOT NULL,
    fotografia VARCHAR(100),
    valoracionId INT NOT NULL,
    restauranteId INT NOT NULL,
    PRIMARY KEY(productoId) ,
    FOREIGN KEY(valoracionId) REFERENCES
listaDeValoraciones(valoracionId),
    FOREIGN KEY(restauranteId) REFERENCES restaurantes(restauranteId),
    CONSTRAINT precioInvalido CHECK(precio >= 0)
);

CREATE TABLE lineasDePedidos(
    lineaDePedidoId INT NOT NULL AUTO_INCREMENT,
    cantidad INT NOT NULL,

```

```
pedidoId INT NOT NULL,  
productoId INT NOT NULL,  
PRIMARY KEY(lineaDePedidoId),  
FOREIGN KEY(pedidoId) REFERENCES pedidos(pedidoId) ON DELETE CASCADE,  
FOREIGN KEY(productoId) REFERENCES productos(productoId),  
CONSTRAINT cantidadInvalida CHECK(cantidad >= 0)  
);
```

9.2 Creación de funciones, procedimientos

```
-- RF-1:
SELECT nombre,estado FROM restaurantes
;

-- RF-2:

DELIMITER //
CREATE OR REPLACE FUNCTION fObtenerTotalPedido(identificador INT) RETURNS
DOUBLE
BEGIN
    DECLARE precio DOUBLE;
    SET precio = (
        SELECT SUM(productos.precio * lineasdepedidos.cantidad)
        FROM productos
        JOIN lineasdepedidos ON
        (productos.productoId=lineasdepedidos.productoId)
        JOIN pedidos ON (lineasdepedidos.pedidoId=pedidos.pedidoId)
        WHERE pedidos.pedidoId=identificador
    );
    IF(precio < 10) THEN
        SET precio = (
            SELECT SUM(productos.precio * lineasdepedidos.cantidad) +
restaurantes.gastosEnvio
            FROM pedidos
            JOIN lineasdepedidos ON (pedidos.pedidoId=lineasdepedidos.pedidoId)
            JOIN productos ON (lineasdepedidos.productoId=productos.productoId)
            JOIN restaurantes ON
            (productos.restauranteId=restaurantes.restauranteId)
            WHERE pedidos.pedidoId=identificador
        );
    END IF;
    RETURN precio;
END //
DELIMITER ;

-- RF-3:
DELIMITER //
CREATE OR REPLACE FUNCTION fUnidadesVendidas(idProducto INT) RETURNS INT
BEGIN
    RETURN (
        SELECT SUM(lineasdepedidos.cantidad)
        FROM lineasdepedidos
        JOIN productos ON (lineasdepedidos.productoId=productos.productoId)
        WHERE productos.productoId=idProducto
    );
END //
DELIMITER ;

-- RF-4:
DELIMITER //
CREATE OR REPLACE FUNCTION fTiempoReparto(identificador INT) RETURNS TIME
BEGIN
    RETURN (
```



```

        SELECT TIMEDIFF(pedidos.fechaEntrega , pedidos.fechaEnvio) FROM
pedidos WHERE identificador=pedidos.pedidoId
    );
END //
DELIMITER ;

-- RF-5:
DELIMITER //
CREATE OR REPLACE FUNCTION fRestaurantesPorCategorias(categoria
VARCHAR(100)) RETURNS VARCHAR(100)
BEGIN
    RETURN(
        SELECT restaurantes.nombre FROM restaurantes
        WHERE restaurantes.categoriaRestaurante = categoria
    );
END //
DELIMITER ;

-- Si hubiese mas de un restaurante de una categoría dada tendríamos que
hacerlo con un select

-- SELECT restaurantes.nombre FROM restaurantes
-- WHERE restaurantes.categoriaRestaurante = 'fast food'

-- RF6
DELIMITER //
CREATE OR REPLACE PROCEDURE pEliminarPedido(pedId INT)
BEGIN
    DELETE FROM pedidos WHERE pedidoId=pedId;
END //

DELIMITER ;
CALL pEliminarPedido(1);

```

9.3 Poblado de la base de datos

```
DELETE FROM lineasdepedidos;
DELETE FROM productos;
DELETE FROM pedidos;
DELETE FROM listaDevaloraciones;
DELETE FROM restaurantes;
DELETE FROM gerentesRestaurantes;
DELETE FROM clientes;
```

```
INSERT INTO clientes(correoElectronico,nombre
,apellidos,telefono,fotografia,codigoPostal) VALUES
('Manuelito@hotmail.com','Manuel','Lopez
Fernandez','625478980','path=C:\fotos\fotomanuel.png','41704'),
('Martgh@hotmail.com','Marta','Gutiérrez
Herrera','615278931','path=C:\fotos\foto1.png','41500'),
('Jorgelvp@hotmail.com','Jorge','López
Vega','667778546','path=C:\fotos\fotojlv.png','41711'),
('Gonzalosp@hotmail.com','Gonzalo','Suárez
Potros','611236790','path=C:\fotos\gonzalofoto.png','41704'),
('Irenelc@hotmail.com','Irene','Hernández
Guillén','655478921','path=C:\fotos\fotodeliverirene.png','41015')
;
```

```
INSERT INTO gerentesRestaurantes(correoElectronico,nombre
,apellidos,telefono,fotografia,numeroLicencia) VALUES
('DiegoLeon@anatolia.com','Diego','Leon
Fernandez','611098890','path=C:\fotos\fotodiegorestaurante.png','9834758984
'),
('LauraEstrada@goiko.com','Laura','Estrada
Lopez','622123321','path=C:\fotos\fotolauranegocio.png','1045675897'),
('MiguelSoto@Ándalee.com','Miguel','Soto
Hidalgo','633456654','path=C:\fotos\fotomiguelrestaurante.png','0098789867'
)
;
```

```
INSERT INTO
restaurantes(gerenteId,nombre,descripcion,direccion,codigoPostal,url,numero
DeTelefono,gastosEnvio,tiempoMedioDeServicio,
estado,categoriaRestaurante,correoElectronico) VALUES
(1,'Anatolia','Restaurante Sevillano que lleva muchas generaciones
trabajando en perfeccionar su platos','Calle Uruguay
nº2','41704','www.anatolia.es','955678876',3.5,
30,FALSE,'gourmet','anatoliarestaurante@hotmail.com'),
(2,'Goiko','Para los que le gusta poner toooda la carne en el
asador','Calle Heliopoldo
nº3','41500','www.goiko.es','955678123',1.99,25,TRUE,'fast
food','goiko@hotmail.com'),
(3,'Ándalee','Si quieres probar la auténtica esencia mexicana este es
tu restaurante, los mejores sabores y platos de nuestra maravillosa
tierra','Calle SpidyGonzález
nº5','41250','www.andlee.es','955366689',2.5,35,TRUE,'temático','andleeeme
xicano@hotmail.com')
;
```

```
INSERT INTO listaDeValoraciones(clienteId,valor,comentario) VALUES
```

```

        (1,3,'Resturante nada especial pero que cumple con lo esperado'),
        (2,5,'Una maravilla de restaurante, volvería sin duda'),
        (3,1,'Comida de mala calidad y mucho tiempo de espera, no lo
recomiendo'),
        (4,4,'Me ha sorprendido gratamente este restaurante, lo recomiendo si
estas algo ajustado de dinero y quieres comer en cantidad'),
        (5,3,'Me ha gustado')
    ;

```

INSERT INTO

```

pedidos(clienteId,restauranteId,valoracionId,fechaCreacion,fechaAceptacion,
fechaEnvio,fechaEntrega,direccionDestino,metodoDePago) VALUES
    (1,3,1,'2022-05-19 14:30','2022-05-19 14:33','2022-05-19
15:00','2022-05-19 15:15','Calle Parsero,3','tarjeta'),
    (2,2,4,'2022-08-29 20:00','2022-08-29 20:23','2022-08-29
20:30','2022-08-29 21:00','Calle Argentina,33','paypal'),
    (3,2,2,'2022-10-30 13:30','2022-10-30 13:33','2022-10-30
13:50','2022-10-30 14:15','Calle Almanzor,9','efectivo'),
    (4,1,3,'2022-11-30 22:00','2022-11-30 22:03','2022-11-30
22:15','2022-11-30 22:30','Calle Romeo y Julieta,15','tarjeta'),
    (3,1,5,'2022-09-10 15:05','2022-09-10 15:15','2022-09-10
15:30','2022-09-10 16:00','Calle Saborio,28','paypal')
;

```

INSERT INTO

```

productos(valoracionId,restauranteId,nombre,descripcion,precio,disponibilid
ad,fotografia) VALUES
    (1,2,'LA TITAN','Hamburguesa de ternera con queso roquefort y
atún',14.99,TRUE,'/desktop/foto1.png'),
    (2,1,'FETUCCINI','Plato de pasta a la boloñesa con
pimienta',10.99,TRUE,'/desktop/foto2.png'),
    (3,3,'BURRITO','Burrito mixto de ternera y pollo completo con salsa
yogur',9.50,TRUE,'/desktop/foto3.png'),
    (4,2,'YANKEE','Hamburguesa de pollo con bacon,queso cheddar y cebolla
crunchi',16.99,TRUE,'/desktop/foto4.png'),
    (5,1,'LA VELETA','Plato especial de la casa, chuleta de cerdo ahumada
con salsa a la pimienta',32.99,TRUE,'/desktop/foto5.png')
;

```

INSERT INTO lineasdepedidos

```

(pedidoId,productoId,cantidad) VALUES
    (2,4,2),
    (1,2,1),
    (3,2,3),
    (4,5,1),
    (5,3,1)
;

```

9.4 Cursores y listados de consultas

```
--      Cursor

-- Navega sobre la tabla productos e incrementar el precio del producto en
1€
DELIMITER //

DROP PROCEDURE IF EXISTS precioProducto//
CREATE PROCEDURE precioProducto()
BEGIN

    DECLARE var_id INTEGER;
    DECLARE var_precio DOUBLE;
    DECLARE var_nombre VARCHAR(100);
    DECLARE var_final INTEGER DEFAULT 0;

    DECLARE CURSOR1 CURSOR FOR SELECT productoId,nombre,precio FROM
productos;

    DECLARE CONTINUE HANDLER FOR NOT FOUND SET var_final =1;

    OPEN CURSOR1;

    bucle: LOOP

        FETCH cursor1 INTO var_id, var_nombre, var_precio;

        IF var_final = 1 THEN
            LEAVE bucle;
        END IF;

        UPDATE productos SET precio = var_precio + 1.00 WHERE productoId =
var_id;

        SELECT
            var_nombre AS 'nombre',
            var_precio AS 'precio',
            precio AS 'Incremento'
        FROM productos WHERE productoId = var_id;

    END LOOP bucle;
    CLOSE CURSOR1;

END//
DELIMITER ;

CALL precioProducto();

-- Lista de consultas

-- RF1
SELECT nombre,estado FROM restaurantes
;
```

```
-- RF2
SELECT DISTINCT fObtenerTotalPedido(5) FROM pedidos;

-- RF3
SELECT DISTINCT fUnidadesVendidas(2) FROM lineasdepedidos;

-- RF4
SELECT DISTINCT fTiempoReparto(1) FROM restaurantes;

-- RF5
SELECT DISTINCT fRestaurantesPorCategorias('fast food') FROM restaurantes;
```

9.5 Creación de triggers

```
-- RN2 Pedidos a restaurantes que esten cerrados

DROP TRIGGER IF EXISTS triggerRestaurantesCerrados;

DELIMITER //
CREATE OR REPLACE TRIGGER triggerRestaurantesCerrados
BEFORE UPDATE ON pedidos
FOR EACH ROW
BEGIN
    DECLARE state BOOLEAN;
    SET state = (SELECT restaurantes.estado FROM pedidos JOIN restaurantes
ON (pedidos.restauranteId=restaurantes.restauranteId));
    IF (state=FALSE)
        THEN SIGNAL SQLSTATE '45001' SET MESSAGE_TEXT = 'No se puede realizar
un pedido a un restaurante cerrado';
        END IF;
END //
DELIMITER ;

INSERT INTO pedidos
(clienteId, restauranteId, valoracionId, fechaCreacion, fechaAceptacion, fechaEn
vio, fechaEntrega, direccionDestino, metodoDePago)
VALUES(1,1,7, '2022-05-19 11:30', '2022-05-19 11:33', '2022-05-19
12:00', '2022-05-19 12:15', 'Calle Guaranil', 'tarjeta');
```

-- RN-3 Politica de pago

-- Vamos a hacer un insert de una valoracion para tener 6 ids y poder
probar el trigger

```
DROP TRIGGER IF EXISTS triggerPoliticaPago;

INSERT INTO listadevaloraciones (clienteId, valor, comentario) VALUES
(1,1, 'Restaurante mediocre');
```

DELIMITER //

```
CREATE OR REPLACE TRIGGER triggerPoliticaPago
BEFORE INSERT ON pedidos
FOR EACH ROW
BEGIN
    IF (NEW.metodoDePago != 'tarjeta' AND NEW.metodoDePago != 'efectivo'
AND NEW.metodoDePago != 'paypal' ) THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'MÉTODO DE PAGO
INCORRECTO';
        END IF;
END //
```

```
INSERT INTO pedidos
(clienteId, restauranteId, valoracionId, fechaCreacion, fechaAceptacion, fechaEn
vio, fechaEntrega, direccionDestino, metodoDePago)
VALUES(1,1,6, '2022-10-19 11:30', '2022-10-19 11:33', '2022-10-19
12:00', '2022-10-19 12:15', 'Calle Alcaraz', 'bitcoin'); -- si pones cualquier
otro método de pago que no sean los establecidos, sale un error que no es el
mensaje de texto
```