Índice

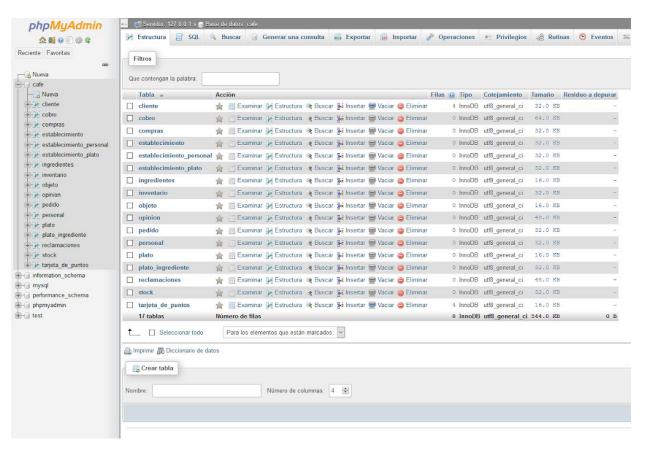
Captura de la inserción inicial Tablas Valores Iniciales Tabla cliente Tabla cobro Tabla compras Tabla establecimiento Tabla establecimiento personal Tabla establecimiento plato Tabla ingredientes Tabla inventario Tabla objeto Tabla opinion Tabla pedido Tabla personal Tabla plato Tabla plato ingrediente Tabla reclamaciones Tabla stock Tabla tarjeta de puntos Capturas de Triggers actualizacionStockCuandoCobro El tipo de pedido ha de ser "local" o "Domicilio" Comprobamos que los platos pertenecen al establecimiento de ese pedido Obtenemos la cantidad de ingredientes que tienen todos los platos, y lo restamos al stock del establecimiento. Se actualiza el Stock. <u>anadirCompra</u> comprobar alergeno ingrediente fecha cliente comprobar puestos comprobar tipo ingrediente

precio plato

comprobacion_valoracion
verificarTipoCompra

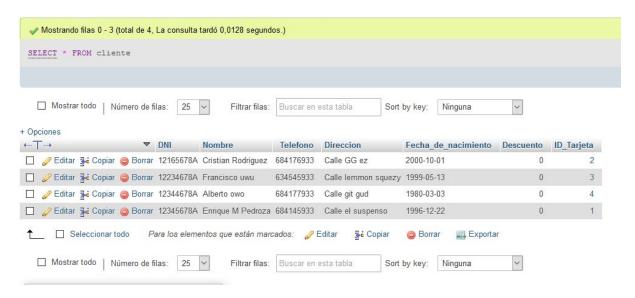
1. Captura de la inserción inicial

1.1. Tablas

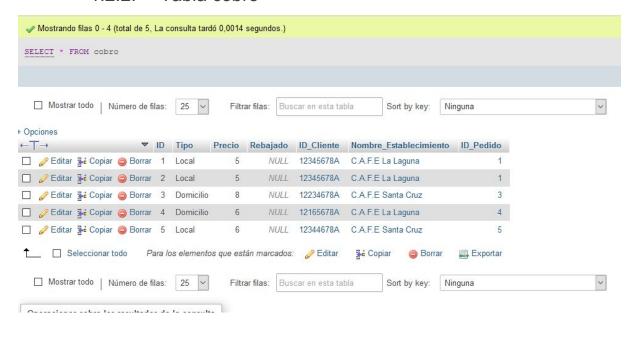


1.2. Valores Iniciales

1.2.1. Tabla cliente



1.2.2. Tabla cobro



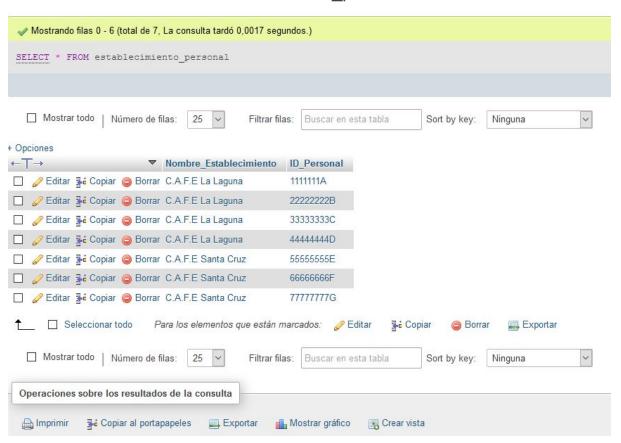
1.2.3. Tabla compras



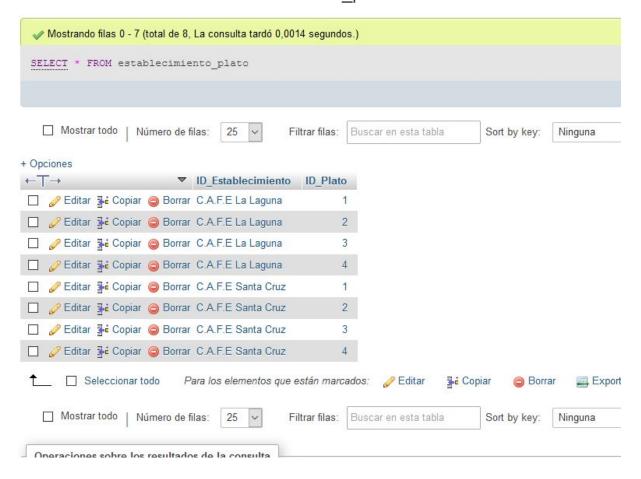
1.2.4. Tabla establecimiento



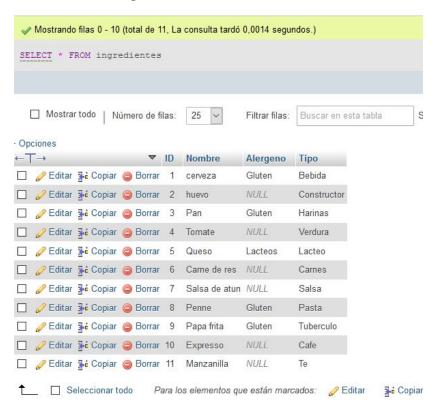
1.2.5. Tabla establecimiento personal



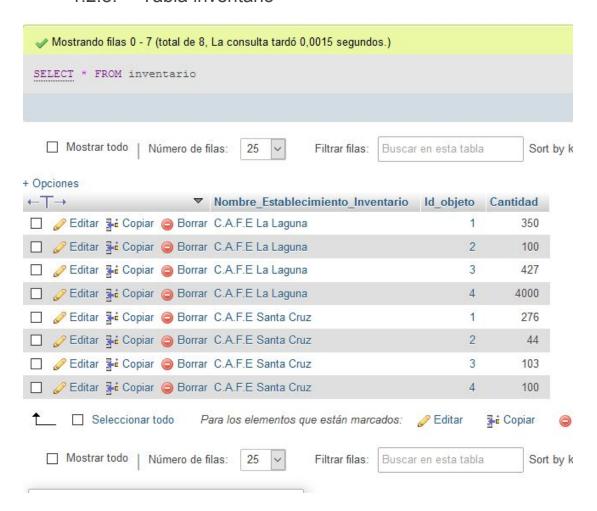
1.2.6. Tabla establecimiento plato



1.2.7. Tabla ingredientes



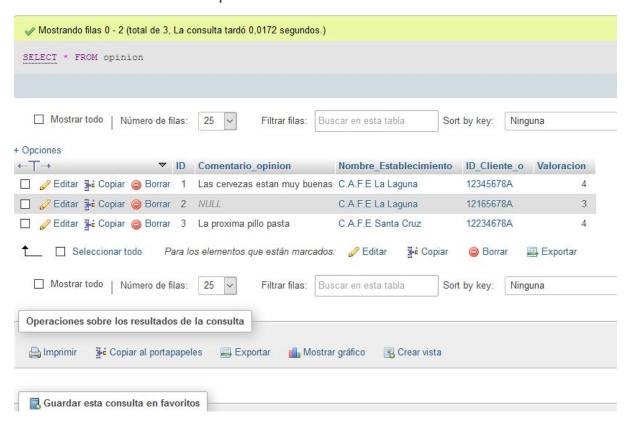
1.2.8. Tabla inventario



1.2.9. Tabla objeto



1.2.10. Tabla opinion



1.2.11. Tabla pedido



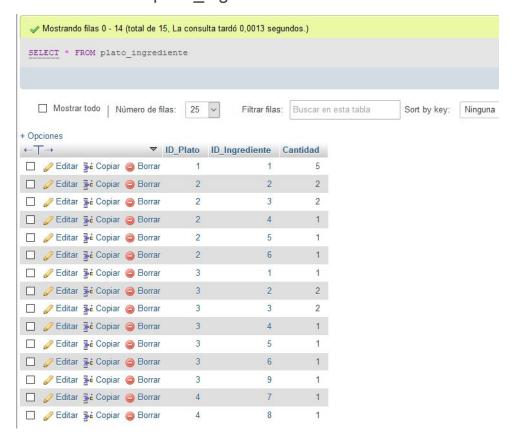
1.2.12. Tabla personal



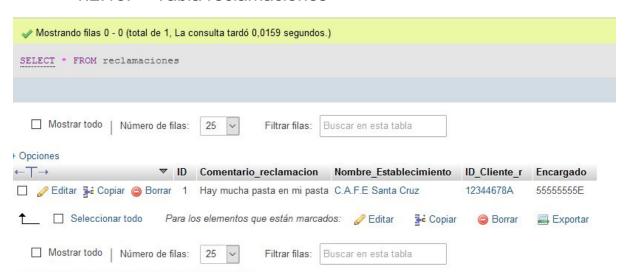
1.2.13. Tabla plato



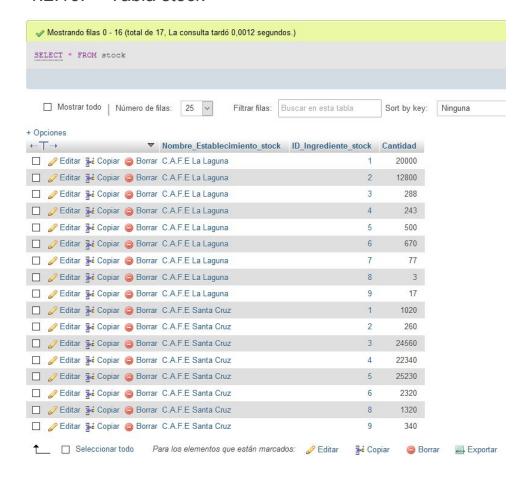
1.2.14. Tabla plato ingrediente



1.2.15. Tabla reclamaciones



1.2.16. Tabla stock



1.2.17. Tabla tarjeta_de_puntos



2. Capturas de Triggers

2.1. actualizacionStockCuandoCobro

<u>Información:</u> Este Trigger actualiza la información cuando se realiza un cobro. Un cobro se realiza cuando un cliente realiza un pedido y paga ese pedido. En el proceso, este Trigger comprueba:

- 1. El tipo de pedido ha de ser "local" o "Domicilio"
- 2. Comprobamos que los platos pertenecen al establecimiento de ese pedido
- 3. Obtenemos la cantidad de ingredientes que tienen todos los platos, y lo restamos al stock del establecimiento. Se actualiza el Stock.
- 4. Ponemos el precio del pedido, que es la suma de los platos
- 5. Actualizamos el precio del producto con el de los puntos.

2.1.1. El tipo de pedido ha de ser "local" o "Domicilio"

Primero insertamos un pedido que sea "Local":

```
√ 1 fila insertada.

La Id de la fila insertada es: 6 (La consulta tardó 0,4176 segundos.)

INSERT INTO cobro (Tipo, Precio, Rebajado, ID_Cliente, Nombre_Establecimiento, ID_Pedido) VALUES ("Local", 10, NULL, "12345678A", "C.A.F.E La Laguna", 5)

| Comparison of the consultation o
```

A continuación, otro que sea a "Domicilio":

```
√ 1 fila insertada.

La Id de la fila insertada es: 7 (La consulta tardó 0.4919 segundos.)

INSERI INTO cobro (Tipo, Precio, Rebajado, ID_Cliente, Nombre_Establecimiento, ID_Pedido) VALUES ("Domicilio", 10, NULL, "12345678A", "C.A.F.E La Laguna", 5)

| Comparison of the Compar
```

Si introducimos otro que no sea "local" o "Domicilio", nos mostrará un error personalizado. En este caso, intentamos introducir "Glovo":

```
Error

consulta SOL: Copiar

INSERT INTO cobro (Tipo, Precio, Rebajado, ID_Cliente, Nombre_Establecimiento, ID_Pedido) VALUES ("Glovo", 10, NULL, "12945678A", "C.A.F.E La Laguna", 5)

MySOL ha dicho: 
$1644 - El tipo de un pedido solo puede ser Local o Domicilio
```

2.1.2.Comprobamos que los platos pertenecen al establecimiento de ese pedido

<u>Información</u>: Definimos un plato como un conjunto o combinación de alimentos. Cada establecimiento puede ofrecer diferentes tipos de platos (Unos hamburguesas, otros pueden tener ensaladas, sopas, etc). Los platos que tiene cada establecimiento, se almacenan en la tabla establecimiento_plato, que contiene la relaciones entre los establecimientos y sus platos. Este Trigger comprueba, que el identificador, de los platos que pide el usuario, permanezcan en el establecimiento donde realizó el pedido.

Primero comprobamos un plato que pertenece al establecimiento. El pedido 4, contiene el plato 4, que está SOLO en C.A.F.E. La Laguna:



Por lo tanto, si ahora añadido un cobro de un Pedido 4, desde "C.A.F.E La Laguna", el cobro se ejecutará con éxito:

```
√ 1 fila insertada.

La Id de la fila insertada es: 12 (La consulta tardó 0,5435 segundos.)

INSERT INTO cobro (Tipo, Precio, Rebajado, ID_Cliente, Nombre_Establecimiento, ID_Pedido) VALUES ("Local", 11, NULL, "12345678A", "C.A.F.E La Laguna", 4)

INTO cobro (Tipo, Precio, Rebajado, ID_Cliente, Nombre_Establecimiento, ID_Pedido) VALUES ("Local", 11, NULL, "12345678A", "C.A.F.E La Laguna", 4)

**The color of the c
```

Sin embargo, recordemos que en la primera captura, podemos ver como el plato 4 solo se encuentra en La Laguna. Si lo pedimos en Santa Cruz, nos saldrá un error:

```
Error

consulta SQL: Copiar

INSERT INTO cobro (Tipo, Frecio, Rebajado, ID_Cliente, Nombre_Establecimiento, ID_Fedido) VALUES ("Local", 11, NULL, "12345678A", "C.A.F.E Snta Cruz", 4)

MySQL ha dicho: 

#1644 - Error un plato no se encuentra en el establecimiento o no hay asociado un plato a ningún establecimiento
```

2.1.3. Obtenemos la cantidad de ingredientes que tienen todos los platos, y lo restamos al stock del establecimiento. Se actualiza el Stock.

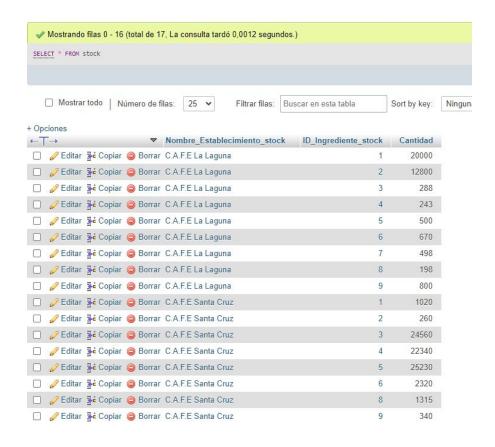
Vamos a realizar una compra de una hamburguesa combo. La cual se compone de los siguientes ingredientes:

- → 2 de Huevo (ID = 2)
- → 2 de Pan (ID = 3)
- \rightarrow 1 de Tomate (ID = 4)
- → 1 de Queso (ID = 5)
- → 1 de Carne de Res (ID = 6)

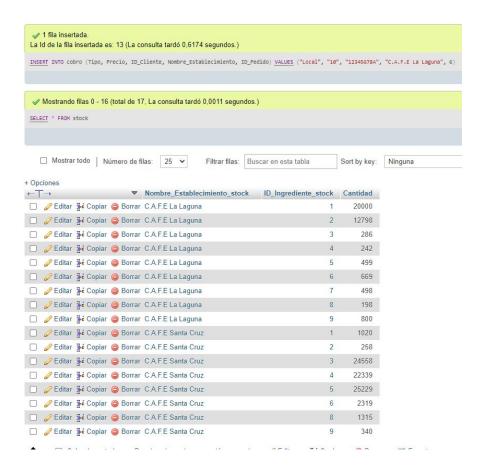
Realizamos primero un select de la tabla de Stock, ejecutamos la consulta de cobro, y hacemos nuevamente otra consulta select sobre Stock, y veremos que sea han reducido las cantidades de esos ingredientes en la tabla Stock:

```
SELECT * FROM stock;
INSERT INTO cobro (Tipo, Precio, ID_Cliente, Nombre_Establecimiento,
ID_Pedido) VALUES ("Local", "10", "12345678A", "C.A.F.E La Laguna", 6);
SELECT * FROM stock;
```

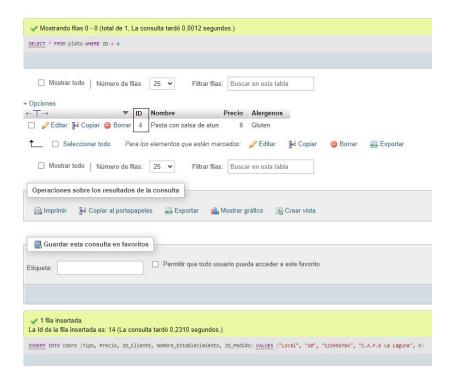
La tabla stock ANTES de que se active el Trigger:



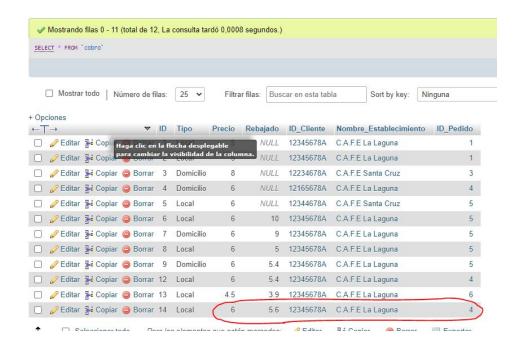
La tabla stock DESPUÉS de realizar el cobro:



A continuación, el Trigger pone el precio del pedido como la suma de todos los platos. Vamos a ver el pedido 4, el cual corresponde al Plato 4. El cual tiene un precio de 6 euros. Al realizar la inserción en la tabla de Cobros, deberá de aparecer el costo del plato, que son 6 euros.



Y ahora, si vemos nuevamente la tabla cobros, vemos que el último cobro tiene un precio de 6, que es el precio del plato:

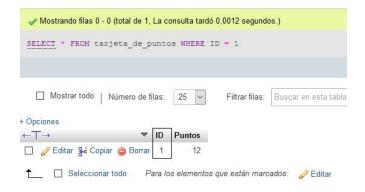


A continuación, queda actualizar el nuevo precio y añadirlo (*PrecioRebajado*). Recordemos que tenemos un sistema de puntos, por lo que podemos realizar un descuento a los usuarios según los puntos que tengan. Estos se almacenan en la tabla de *tarjeta de puntos*, y se aplican directamente al comprar cualquier producto por el cliente:

Está puesto a relación 1 a 10. Es decir, por cada euro gastado obtenemos un punto. Y por cada diez puntos, obtenemos un euro.

Como podemos ver en la siguiente consulta, el cliente 1, tiene 12 puntos, por lo tanto, es un ahorro de 1.2 euros en el pedido, si pedimos una hamburguesa combo (Que cuesta 6 euros). Entonces, nos queda disponible: 6€ - 1.2€ = 4.8 €. Además, recordemos que añadimos 6 puntos por los 6 euros.

Resumen: El nuevo precio rebajado es de 4.8 euros, y nos quedamos con 6 puntos. Recordemos que el ID del cliente es 1. A continuación, una captura de los puntos:



Vemos la inserción en la tabla de cobros:

```
✓ 1 fila insertada.

La ld de la fila insertada es: 15 (La consulta tardó 0,3059 segundos.)

INSERT INTO cobro (TIPO, ID_Cliente, Nombre_Establecimiento, ID_Pedido) VALUES ("Local", "12345678A", "C.A.F.E La Laguna", 4)

Output

Description

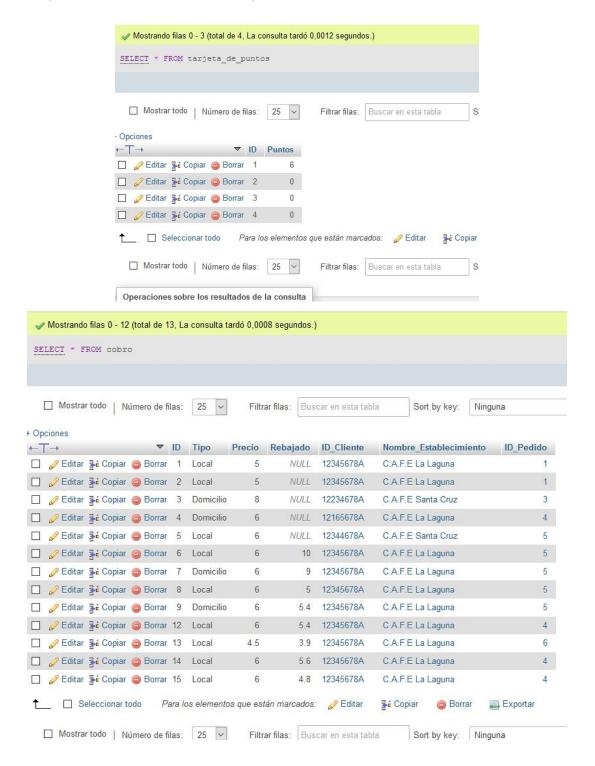
VALUES ("Local", "12345678A", "C.A.F.E La Laguna", 4)

Output

Description

Ou
```

Si a continuación, hacemos una captura a la tarjeta de puntos, veremos como tenemos 6 puntos, y además, el nuevo precio rebajado es de 4.8 euros:



2.2. anadirCompra

Lo que hace este Trigger, es que, recordemos que la tabla compras contiene las compras del Inventario y el Stock. La diferencia entre ambos es la siguiente:

- → <u>Inventario</u>: Consiste en objetos, los cuales no están pensado para venderse, sino para ofrecer servicios. Además, no son un solo uso. Estos pueden ser: Mesas, sillas, ordenadores, freidoras, máquinas de café, etc.
- → <u>Stock</u>: Todo lo relacionado con el producto que se va a vender. En este caso, nos referimos a los ingredientes de los alimentos: Pan, zanahoria, cebolla, huevos, etc.

Lo primero que hace el Trigger es comprobar si la compra es "I" o "S", de inventario y stock respectivamente. Vamos a introducir la F, y veremos que es rechazado:

```
Error

consulta SQL: Copiar

INSERT INTO compras (Item, Tipo, Cantidad, Nombre_est) VALUES (4, "F", 300, "C.A.F.E Santa Cruz")

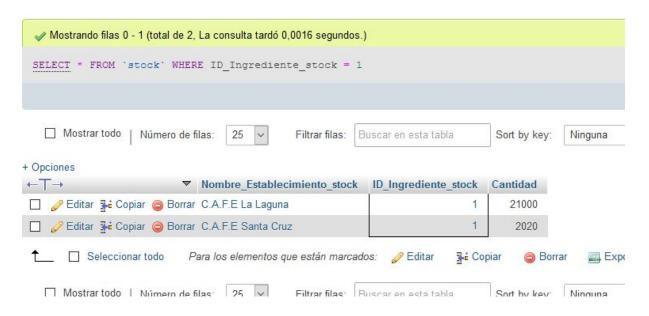
MySQL ha dicho: 

#1644 - Error, el tipo de compra solo puede ser "S" (Stock) o "I" (Inventario)
```

A continuación, probaremos utilizando la "I" o la "S". En el caso de que sea la "S" de Stock, veremos si existe o no ese ingrediente. En el caso de que exista pues le sumamos a la cantidad actual. En caso de que no sea así, pues realizamos la inserción. En el caso de que sea "I" de inventario, pues realizamos lo mismo, pero en la tabla de inventario, en vez de en la de Stock. Probamos las dos consultas. Primero comenzamos viendo nuestro Stock:



Recordemos que el Item enlace con el ID de Stock o Inventario, por lo tanto, vamos a crear un Item que no esté, y luego otro que este. Primero, escogeremos el Item 1, que en Stock tiene 21000 Unidades:



Ahora introduciremos una compra de 4000 unidades en el Stock de ID_Ingrediente_stock = 1, y en el establecimiento de La Laguna:

Vemos como los ingredientes han pasado de 21000 a 25000:



Lo mismo ocurre si lo realizamos con la tabla de inventario.

2.3. comprobar_alergeno_ingrediente

Comprueba los alérgenos al introducir un alimento, por si hay alguno que sea incorrecto. Los alérgenos que acepta la aplicación son:

Pescado	Frutos Secos	Lácteos	Moluscos	Gluten
Crustaceos	Huevos	Cacahuetes	Soja	Apio
Mostaza	Sésamo	Altramuz	Sulfitos	

Vamos a probar con uno que no esté en la lista y luego con otro que sí lo esté. Por ejemplo, uno que no esté en la lista: "*Agua*", (El cual no es un alérgeno), y uno que esté: "*Sulfitos*".

```
Error

consulta SQL: Copiar

INSERT INTO ingredientes (nombre, Alergeno, Tipo) VALUES ("Agua", "Agua", "Bebina")

MySQL ha dicho: 

#1644 - Error al introducir el Alergeno de los ingredientes
```

Como podemos ver, no podemos llamar al alergeno de un alimento "Agua". Sin embargo, si introducimos la mostaza con sulfito, si nos dejará introducirlo:



2.4. fecha_cliente

Comprobamos que la fecha de nacimiento de un cliente es superior a 1900, y que la edad de la persona es mayor a 14 años:

```
Error

consulta SQL: Copiar

INSERT INTO cliente (DNI, Nombre, Telefono, Direccion, Fecha_de_nacimiento, ID_Tarjeta) VALUES ("837583498", "Juan", 849302421, "Calle X", '1820-10-20', 5)

MySQL ha dicho: 

#1644 - Una persona no puede tener ese año de nacimiento
```

Como podemos ver, una persona no puede haber nacido en 1820 y poder registrarse en nuestra plataforma. A continuación, una fecha que sea futura, es decir, mayor que la fecha actual en la que estamos:

```
Error

consulta SQL: Copiar

INSERT INTO cliente (DNI, Nombre, Telefono, Direccion, Fecha_de_nacimiento, ID_Tarjeta) VALUES ("83758349B", "Juan", 849302421, "Calle X", '2077-10-20', 5)

MySQL ha dicho: 

#1644 - Una persona no puede tener ese año de nacimiento
```

Como podemos ver, una persona no puede nacer en el año 2077, y registrarse en nuestra plataforma a tiempo de hoy. A continuación, una persona con doce años, por lo tanto, el trigger no debería dejar que esta persona lograra registrarse:

```
Error

consulta SQL: Copiar

INSERT INTO cliente (DNI, Nombre, Telefono, Direccion, Fecha_de_nacimiento, ID_Tarjeta) VALUES ("83758349B", "Juan", 849302421, "Calle X", '2009-10-20', S)

MySQL ha dicho: 

#1644 - Una persona no puede tener ese año de nacimiento
```

Por último, una fecha que SI es válida. Una persona que nació, por ejemplo, en 1980:

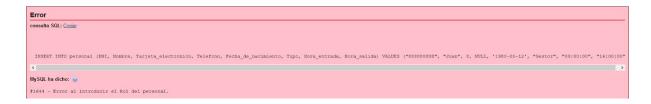
```
✓ 1 fila insertada. (La consulta tardó 0,0622 segundos.)

INSERT INTO cliente (DNI, Nombre, Telefono, Direccion, Fecha_de_nacimiento, ID_Tarjeta) 
VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)

| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("83758349B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("8375849B", "Juan", 849302421, "Calle X", '1980-10-20', 5)
| VALUES ("8375849B", 8495849B", 84958
```

2.5. comprobar_puestos

En este Trigger, comprobamos que el puesto de la persona a introducir es el correcto. Por ejemplo, si introducimos "Gerente", el cual NO está aceptado, nos saldrá el siguiente error:



Sin embargo, si introducimos Camarero, el cual está aceptado:



Los puestos que son aceptados son: Camarero, Cocinero, Gerente.

2.6. comprobar_tipo_ingrediente

Este Trigger es muy similar al de comprobar los alérgenos, pero en este caso, compara los tipos de alimentos. Los tipos de alimentos disponibles son:

Bebida	Constructor	Harinas	Verdura
Lácteo	Carnes	Salsa	Pasta
Tubérculo	Cafe	Te	Cereal
Frutas	Frutos Secos	Aceites	Hortalizas

Vamos a probar a introducir un tipo de ingrediente que esté en la lista, y luego otro que no esté en la lista. Primero, uno que **NO** esté en la lista de ingredientes, por ejemplo: "Condimento":

```
Error

consulta SQL: Copiar

INSERT INTO ingredientes (Nombre, Alergeno, Tipo) VALUES ("Canela", NULL, "Condimento")

MySQL ha dicho: 

#1644 - Error al introducir el tipo de los ingredientes
```

Y ahora, uno que sí esté en la lista de ingredientes (Harinas):

```
✓ 1 fila insertada.

La Id de la fila insertada es: 13 (La consulta tardó 0,0577 segundos.)

INSERT INTO ingredientes (Nombre, Alergeno, Tipo) VALUES ("Canela", NULL, "Harinas")

Output

Description:

VALUES ("Canela", NULL, "Harinas")

Output

Description:

Output

Description:
```

2.7. precio_plato

Comprueba el precio del plato. Es decir, que no sea 0 o inferior:

Primero probamos con -1, luego con 0, y luego con 2:

```
Error

consulta SQL: Copiar

INSERT INTO plato (Nombre, Precio, Alergenos) VALUES ("Ensalada", -1, "Gluten")

MySQL ha dicho: 

#1644 - Error, el precio de un plato no puede ser igual o menor a 0
```

Ahora probamos con 0:

```
Error

consulta SQL: Copiar

INSERT INTO plato (Nombre, Precio, Alergenos) VALUES ("Ensalada", 0, "Gluten")

MySQL ha dicho: 

#1644 - Error, el precio de un plato no puede ser igual o menor a 0
```

Ahora con 2, el cual es un número positivo:

```
✓ 1 fila insertada.
La Id de la fila insertada es: 5 (La consulta tardó 0,0543 segundos.)
INSERT INTO plato (Nombre, Precio, Alergenos) VALUES ("Ensalada", 2, "Gluten")
```

2.8. comprobacion_valoracion

Comprueba que las estrellas de valoración de un usuario se encuentren entre 1 o 5. Probamos con -1, 12 y 4. Primero con -1:

```
Error

consulta SQL: Copiar

INSERT INTO opinion (Commentario_opinion, Nombre_Establecimiento, ID_Cliente_o, Valoracion) VALUES ("Buen establecimiento", "C.A.F.E La Laguna", "12345678A", -1)

MySQL ha dicho: 

#1644 - Una valoracion tiene que estar entre 1 y 5 puntos
```

A continuación, probamos con 12:

```
Error

consulta SQL: Copiar

INSERT INTO opinion (Comentario_opinion, Nombre_Establecimiento, ID_Cliente_o, Valoracion) VALUES ("Buen establecimiento", "C.A.F.E La Laguna", "12345678A", 12)

MySQL ha dicho: 

#1644 - Una valoracion tiene que estar entre 1 y 5 puntos
```

Por último, probamos con 4 estrellas:

Vemos como este último si es aceptado por el Trigger.

2.10.verificarTipoCompra

Comprobamos que al realizar la compra sea de tipo Stock (S) o Inventario (I). Probamos a insertar una S, una I, y luego otra letra como una F:

```
✓ 1 fila insertada.

La ld de la fila insertada es: 10 (La consulta tardó 0,0873 segundos.)

INSERT INTO compras (Item, Tipo, Cantidad, Nombre_est) VALUES (9, "S", 2000, "C.A.F.E Santa Cruz")
```

Vemos que hemos insertado "S", y se ha realizado correctamente. Probamos con la "I":

Vemos que también se ha insertado correctamente. Ahora probamos con la F:

```
Error

consulta SQL: Copiar

INSERT INTO compras (Item, Tipo, Cantidad, Nombre_est) VALUES (9, "F", 5000, "C.A.F.E Santa Cruz")

MySQL ha dicho: 
#1644 - Error, el tipo de compra solo puede ser "S" (Stock) o "I" (Inventario)
```

Vemos que la F nos muestra un error, y es que no es ni Stock ni Inventario.