

PRODUCTO 1

CREACIÓN Y MODIFICACIÓN DE UNA BASE DE DATOS

COMPONENTES DEL GRUPO C2047 LAC

M^aCelia Garcia Molina

Liliana Díaz Ibáñez

Agnès Garcia Mateo

CONSULTORA

Rita de la Torre Chirivella

Índice

Índice	1
Descripción	3
Consulta y Discusión en grupo	3
Consultar y discutir con el grupo cuál es la forma más adecuada de abordar el producto. Para ello, se deberán responder las siguientes preguntas:	3
1.2 ¿Cómo se usan los cualificadores en MySQL?	3
1.3 ¿Cuáles son las sentencias (instrucciones) que permiten crear y modificar una tabla?	4
1.4 Tipos de datos MYSQL	6
¿Cuáles son los tipos de datos que admite el MySQL y cuál es más pertinente para cada campo? Recordad que hay que considerar siempre cuáles serán los valores mínimos y máximos que se almacenará dentro de un campo, antes de elegir el tipo de dato más adecuado.	6
1.6 ¿Cómo se definen las claves primarias y las claves candidatas en SQL?	10
1.7 Relaciones entre tablas.	11
1.8 ¿Para qué sirven las cláusulas ON UPDATE... ON DELETE?	12
1.9 ¿Cuál es la sentencia (instrucción) que permite insertar datos en una tabla?	13
1.10. Delimitación de los campos en la inserción.	14
Otras sentencias para trabajar con las tablas.	14
Definir la creación de la base de datos.	14
Modificaciones a la base de datos	19
3.1 Cambia el tipo de datos.	19
3.2 Crea dos nuevos campos en la tabla PRODUCTO:	19
3.2.1 Stock	19
3.2.2 Precio_Compra	20
3.3 Agrega los campos PVP e IVA a la tabla ELABORADO.	20
3.4 Generar una restricción	20
3.5 Inventa una modificación de estructura en la base de datos.	20
3.6 Añade un Campo.	20
Insertar Alérgenos.	21
5. Insertar en las siguientes Tablas:	21
5.1 Insertar Proveedor.	22
insert into PROVEEDOR	22
5.2 Insertar Categoría.	23
5.3 Insertar TIPO.	24
6. Diseñar un Menú	24

7. Insertar Registros	26
Captura Pantalla cuando se está exportando la Base de Datos y sus modificaciones e inserciones.	29

Descripción

En este producto crearemos una base de datos con el lenguaje SQL, es decir, el modelo físico de la misma. Además, trabajaremos con la modificación de la estructura y la inserción de datos. Esta base de datos será el inicio de nuestro proyecto en grupo.

1. Consulta y Discusión en grupo

Consultar y discutir con el grupo cuál es la forma más adecuada de abordar el producto. Para ello, se deberán responder las siguientes preguntas:

1.2 ¿Cómo se usan los cualificadores en MySQL?

El nombre de los objetos pueden ser Cualificados o no cualificados.

Un nombre no cualificado es permitido cuando la interpretación del nombre no es ambiguo. Un nombre cualificado incluye al menos un cualificador que aclara la interpretación del contexto.

por ejemplo:

Un nombre no cualificado es **t4**:

CREATE TABLE t4(i INT);

Un nombre cualificado es **db1.t4**:

CREATE TABLE db1.t4(i INT);

Ahora bien, **el cualificador debe ser especificado** si no hay base de datos por defecto. **El cualificador puede ser modificado** si hay base de datos por defecto para especificar una base de datos diferente desde la que está por defecto ó hacer la base de datos explícita si es igual a la base de datos por defecto.¹

En general los **cualificadores** de los identificadores se usan al aceptar nombres MYSQL que pueden consistir en un sólo identificador o en múltiples identificadores. Los componentes de un nombre múltiple deben separarse con un punto ('.'). Las partes

¹ <https://dev.mysql.com/doc/refman/8.0/en/identifier-qualifiers.html>

iniciales de un identificador múltiple actúan como calificadores que afectan el contexto en el cual se interpreta la parte final.²

1.3 ¿Cuáles son las sentencias (instrucciones) que permiten crear y modificar una tabla?

Para crear tabla:

```
CREATE TABLE <nombre_tabla>  
  
    (<definicion_columna>  
  
    [,definicion_columna>...]  
  
    [,<restricciones_tabla>]  
  
    );
```

Ejemplo:

El nombre de la tabla se puede especificar como **db_name.tb1_name** para crear la tabla en una base de datos específica. Esto funciona independientemente de si existe una base de datos predeterminada, suponiendo que exista la base de datos. Si usa identificadores entre comillas, cite los nombres de la base de datos y la tabla por separado. Por ejemplo, escriba `mydb`.`mytbl`, no `mydb.mytbl`.

Para las reglas permitidas para el nombre de las tablas se pueden encontrar aquí;
<https://dev.mysql.com/doc/refman/8.0/en/identifiers.html>

CREATE TABLE crea una tabla con el nombre dado. Debe tener el permiso CREAR para la tabla.

De forma predeterminada, las tablas se crean en la base de datos predeterminada, utilizando el motor de almacenamiento InnoDB. Se produce

² <http://download.nust.na/pub6/mysql/doc/refman/5.0/es/identifier-qualifiers.html>

un error si la tabla existe, si no hay una base de datos predeterminada o si la base de datos no existe.

MySQL no tiene límite en el número de tablas. El sistema de archivos subyacente puede tener un límite en la cantidad de archivos que representan tablas. Los motores de almacenamiento individuales pueden imponer restricciones específicas del motor. InnoDB permite hasta 4 mil millones de tablas.

Hay varios aspectos de la sentencia CREATE TABLE; Nombre de la tabla, Tablas Temporales, Clonación y copia de tablas, Tipos de datos de columna y atributos, Índices, claves foráneas y restricciones CHECK, Opciones de mesa, Particionamiento de tablas. Que se describen en el siguiente enlace;

<https://dev.mysql.com/doc/refman/8.0/en/create-table.html>

Para modificar una tabla:

```
ALTER TABLE <nombre_tabla> {<accion_modificar_columna> |  
                                <accion_modificar_restriccion_tabla>};
```

La sentencia **ALTER TABLE** cambia la estructura de las tablas. Se puede agregar o borrar columnas, crear o destruir índices, cambiar el tipo de las columnas existentes, o renombrar las columnas o la misma tabla. Además también se puede cambiar características como por ejemplo el motor de almacenamiento usado para la tabla o el del comentario de la tabla.

Para usar **ALTER TABLE** se necesitan permisos para esa tabla de crear (CREATE TABLE), modificar(ALTER) e insertar. Para renombrar una tabla se requieren permisos de ALTER, DROP en la antigua tabla y permisos de ALTER, CREATE e INSERT en la nueva tabla.

Entre las diferentes instrucciones para ALTER TABLE están ;

ADD, DROP, ALTER, ALGORITHM, CHANGE, DEFAULT, CONVERT ,DISABLE, DISCARD, FORCE, LOCK, MODIFY, ORDER BY, RENAME entre otras que se pueden encontrar en el siguiente enlace: <https://dev.mysql.com/doc/refman/8.0/en/alter-table.html>

En cuanto a **las operaciones de Partición con ALTER TABLE**; se pueden usar con tablas particionadas para volver a particionar, para agregar, eliminar, descartar, importar,

fusionar y dividir particiones, y para realizar el mantenimiento del particionamiento. Esto se puede ver en el siguiente enlace

<https://dev.mysql.com/doc/refman/8.0/en/alter-table-partition-operations.html>

De igual manera para las operaciones ALTER TABLE permitidas para las columnas generadas son ADD, MODIFY y CHANGE, las cuales podemos encontrar detalladas en el siguiente enlace;

<https://dev.mysql.com/doc/refman/8.0/en/alter-table-generated-columns.html>

Ejemplos de uso ALTER TABLE;

1- Para cambiar la columna **a** desde INTERGER a TINYINT NOT NULL (dejando el mismo nombre), y para cambiar la columna **b** desde CHAR(10) a CHAR (20) así como renombrarla de b a c.

ALTER TABLE t2 MODIFY a TINYINT NOT NULL, CHANGE b c CHAR(20);

2-Para agregar un índice en la columna d y un índice ÚNICO en la columna a.

ALTER TABLE t2 ADD INDEX (d), ADD UNIQUE (a);³

1.4 Tipos de datos MYSQL

¿Cuáles son los tipos de datos que admite el MySQL y cuál es más pertinente para cada campo? Recordad que hay que considerar siempre cuáles serán los valores mínimos y máximos que se almacenará dentro de un campo, antes de elegir el tipo de dato más adecuado.

Tipos de datos predefinidos:

TIPO DE DATO	NATURAL EZA	TAMAÑO/ FORMATO	RANGO DE VALORES
TINYINT	Entero	1 byte	-128 a 127
SMALLINT	Entero	2 bytes	-32768 a 32767

³ <https://dev.mysql.com/doc/refman/8.0/en/alter-table-examples.html>

MEDIUMINT	Entero	3 bytes	-8.388.608 a 8.388.607
INT/INTEGER	Entero	4 bytes	-2147483648 a 2147483647
BIGINT	Entero	8 bytes	-9.223.372.036.854.775.808 a 9.223.372.036.854.775.807
DOUBLE	Real aprox.	8 bytes	-1.7976931348623157E+308 a -2.2250738585072014E-308
FLOAT	Real aprox.	4 bytes	-3.402823466E+38 a -1.175494351E-38
DECIMAL(longitud, decimales)	Real exacto	Variable	
NUMERIC(longitud, decimales)	Real exacto	Variable	
DATE	Fecha	'Aaaa-mm-dd'	100-01-01 hasta 9999-12-31
TIME	Hora	'Hh:mm:ss'	-839:59:59 hasta 839:59:59
TIMESTAMP	Fecha y hora	'aaaa-mm-dd hh:mm:ss'	1 de enero de 1970 al año 2037
DATETIME	Fecha y hora	'aaaa-mm-dd hh:mm:ss'	1000-01-01 00:00:00 al 9999-12-31 23:59:59
YEAR	Año	'aaaa'	desde 1901 hasta 2155
CHAR(longitud)	Caracte.	Longitud fija	255 caracteres
VARCHAR(longitud)	Caracte.	Longitud var.	65.535 caracteres
BLOB	Objetos binarios	Longitud var.	65.535 bytes
TEXT	Campos memo	Longitud var.	
ENUM(valor1, valor2,...)	Enumera.	Lista de valo.	Hasta 65535 opciones
SET(valor1, valor2,...)	Conjuntos	Conj. de valo	Hasta 64 opciones

Tipo de datos	Descripción
---------------	-------------

CHARACTER - CHAR()	Cadenas de caracteres de longitud fija
CHARACTER VARYING - VARCHAR()	Cadenas de caracteres de longitud variable
BIT	Cadenas de bits de longitud fija

12

BIT VARYING	Cadenas de bits de longitud variable
NUMERIC (precisión, escala)	Números decimales con tantos dígitos como indique la precisión y tantos decimales como indique la escala
DECIMAL (precisión, escala)	Números decimales con tantos dígitos como indique la precisión y tantos decimales como indique la escala
INTEGER - INT	Números enteros
SMALLINT	Números enteros pequeños
REAL	Números con coma flotante con precisión predefinida
FLOAT (precisión)	Números con coma flotante con la precisión especificada
DOUBLE PRECISION	Números con coma flotante con más precisión predefinida que la del tipo REAL
DATE	Fechas. Se componen de: YEAR año, MONTH mes, DAY día
TIME	Horas. Se componen de: : HOUR hora, MINUT minutos, SECOND segundos
TIMESTAMP	Fechas y horas. Se componen de: YEAR año, MONTH mes, DAY día, HOUR hora, MINUT minutos, SECOND segundos

Una instrucción es una descripción completa de un conjunto de datos. Esta instrucción tiene cláusulas y cada cláusula realiza una función de la instrucción.

Para describir un conjunto de datos con SQL, se escribe una instrucción **SELECT**. Una instrucción **SELECT** contiene una descripción completa de un conjunto de datos que quiere obtener de una base de datos. Se incluye lo siguiente:

- Qué tablas contienen los datos.
- Cómo se relacionan los datos de orígenes diferentes.
- Qué campos o cálculos proporcionarán los datos.
- Criterios que los datos deben cumplir para ser incluidos.
- Si se deben ordenar los datos y, en caso de ser así, cómo deben ordenarse.

Como una frase, una instrucción SQL tiene cláusulas. Cada cláusula realiza una función de la instrucción SQL. Algunas cláusulas son necesarias en una instrucción **SELECT**. Las cláusulas SQL más comunes.

Cláusula SQL:

- **SELECT** Muestra una lista de los campos que contienen datos de interés. Es obligatorio.
- **FROM** Muestra las tablas que contienen los campos de la cláusula **SELECT**. Es obligatorio.
- **WHERE** especifica los criterios de campo que cada registro debe cumplir para poder ser incluido en los resultados. No es obligatorio.
- **ORDER BY, GROUP BY, HAVING** entre otras.⁴

4

<https://support.microsoft.com/es-es/office/access-sql-conceptos-b%C3%A1sicos-vocabulario-y-sintaxis-44d0303-cde1-424e-9a74-e8dc3e460671>

1.6 ¿Cómo se definen las claves primarias y las claves candidatas en SQL?

Clave primaria:

```
PRIMARY KEY (cod_producto));
```

Clave candidata/foránea:

```
FOREIGN KEY (codigo_cliente) REFERENCES CLIENTES (codigo_cli);
```

Ejemplo:

En el siguiente código creamos la base de datos "Escuela" y tablas en Sql Server para hacer las relaciones respectivas y agregar las claves foráneas en la Base de Datos. Las tablas a crear son: Alumnos, Asignatura. Que tendrán las llaves primarias. Luego la tabla inscripción que tendrá las llaves foráneas campos Id de las tablas Alumnos, Asignatura.⁵

Create table Alumnos(

Id char(8) primary key,

Nombre varchar(20) not null,

Apellido varchar(20) not null,

Direccion varchar(50),

Fecha_nacimiento char(8)

);

Create table Asignatura(

Id char(8) primary key,

⁵ <https://codigosql.top/sql-server/llaves-primarias-y-foraneas-en-sql-server/>

Nombre varchar(20) not null

);

/*Tabla llaves foráneas*/

Create table Incripcion(

Id char(8) primary key,

IdAsignatura char(8) not null,

IdAlumno char(8) not null,

Fecha char(8),

CONSTRAINT fk_Asignatura FOREIGN KEY (IdAsignatura) REFERENCES Asignatura (Id),

CONSTRAINT fk_Alumno FOREIGN KEY (IdAlumno) REFERENCES Alumnos (Id),

);

1.7 Relaciones entre tablas.

¿Cómo se crean las relaciones entre tablas correctamente (establecer Foreign Keys – Claves Foráneas)?

La forma de crear foreign Keys entre tablas es de la siguiente forma:

FOREIGN KEY (codigo_cliente) REFERENCES CLIENTES (codigo_cli);
--

Declaramos primero el atributo con el nombre de la clave foránea seguido por la cláusula references y el nombre de la tabla, luego la columna a la que hace referencia. Así como podemos ver en el ejemplo del punto anterior.⁶

⁶ <https://codigosql.top/sql-server/llaves-primarias-y-foraneas-en-sql-server/>

1.8 ¿Para qué sirven las cláusulas ON UPDATE... ON DELETE?

- **ON UPDATE:** al crear una clave foránea, con esta función las **filas de referencia se van a actualizar en la tabla secundaria** cuando la fila referenciada se actualiza en la tabla principal que tiene una clave primaria.

ejemplo:

```
UPDATE nombre_tabla SET columna1 = 'nuevo_valor' WHERE columna1 = 'valor1';
```

En esta sentencia todos las filas de la tabla nombre_tabla almacenados en la columna1 que contengan el dato valor1, serán modificadas por el dato nuevo_valor. Se actualizan todas las filas, pero actualizamos únicamente los valores de las columnas seleccionadas en columna1.

Ejemplo de varios valores con doble condición en la cláusula **WHERE**:

```
UPDATE mi_tabla SET ciudad = 'Valencia' WHERE nombre = 'Pepito' AND ciudad = 'Sevilla';
```

En este ejemplo cambiará el nombre de la ciudad por Valencia en todas las filas en las que aparezca el nombre Pepito y la ciudad sea Sevilla.

- **ON DELETE:** su función es **eliminar las filas de referencia en la tabla secundaria cuando la fila referenciada se elimina en la tabla primaria** que tiene una clave primaria, al crear una clave foránea.

La sintaxis de la instrucción DELETE en PostgreSQL es la siguiente:

```
DELETE FROM nombre_tabla WHERE columna1 = 'valor1';
```

ejemplo, para eliminar todos los registros de una tabla:

```
DELETE FROM mi_tabla;7
```

⁷ <https://www.todopostgresql.com/las-instrucciones-update-y-delete/>

1.9 ¿Cuál es la sentencia (instrucción) que permite insertar datos en una tabla?

La instrucción que nos permite insertar datos en una tabla es:

```
INSERT INTO <nombreTabla>[(columnas)] (VALUES (valores));
```

Por ejemplo para introducir los datos de un alumno en la tabla alumnos:

```
INSERT INTO alumnos(nif, nombre, apellidos, telefono, direccion, ciudad)  
VALUES('33.111.444-A', 'Marc', 'Lopez', '111111', 'Gracia 44', 'Barcelona');
```

Otra forma de introducir datos a todas las columnas:

```
INSERT INTO alumnos VALUES ( '116', 'Marc', 'Lopez', '111111', 'Gracia 44',  
'Barcelona');
```

En este último caso se debe tener en cuenta el orden de las columnas en la especificación y se debe enviar todos los datos de igual manera como se ha hecho en las columnas de la tabla, ya que se está indicando que se van agregar registros a todas las columnas y datos de ellas.

Otra opción de insertar registros a todas las columnas es mencionando columnas específicas, para este caso las columnas que se omiten deben tener la propiedad null, es decir que aceptan valores nulos.

```
INSERT INTO Alumnos (Id, Nombre, Apellido) VALUES ('116', 'Marc', 'Lopez');
```

No hemos asignado valores a las columnas teléfono, dirección y ciudad, por tanto tomará automáticamente el valor NULL.⁸

⁸ <https://codigosql.top/sql-server/insertar-datos-en-una-tabla/>

1.10. Delimitación de los campos en la inserción.

¿Qué tipo de carácter se usa para delimitar los campos tipo texto y/o fecha durante la inserción?

Para delimitar los valores tipo texto y fecha se utiliza el carácter comilla simple (') y ('). Tal como se observa en el ejemplo:

INSERT INTO Alumnos (Id, Nombre, Apellido, Direccion, Fecha_nacimiento)

VALUES

('0104', 'Franklin2', 'Garcia', 'avenida 01', '12/01/80'),
('0105', 'Franklin3', 'Garcia', 'avenida 01', '12/01/80')⁹

Otras sentencias para trabajar con las tablas.

¿Con qué otras sentencias contamos para trabajar con las tablas (TRUNCATE TABLE, DROP TABLE, ALTER TABLE, RENAME TABLE)?

- **TRUNCATE TABLE:** se utiliza para vaciar una tabla completamente (tiene variaciones con respecto a la sentencia DELETE).
- **DROP TABLE:** se utiliza para el borrado de una o varias tablas.
- **ALTER TABLE:** se utiliza para la modificación de una o varias tablas.
- **RENAME TABLE:** se utiliza para renombrar una o varias tablas.

Además contamos con sentencias como:

- **CREATE INDEX:** Instrucción que nos permite crear los índices que son una forma más rápida para acceder a los datos.
- **CREATE VIEW:** Instrucción que crea vistas. Las vistas son tablas virtuales en las que se puede ver información de las tablas.

2. Definir la creación de la base de datos.

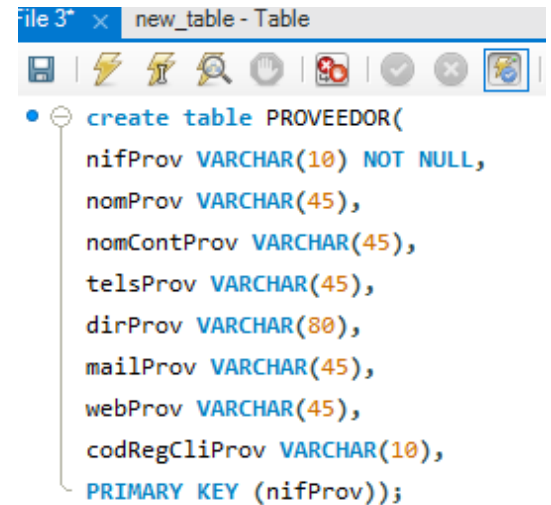
Definir la creación de la base de datos, a partir del modelo relacional y su correspondiente diagrama Entidad-Relación, del caso práctico adjunto. Las sentencias SQL utilizadas para tal fin, deberán apuntarse en el documento a entregar al consultor, y

⁹ <https://codigosql.top/sql-server/insertar-datos-en-una-tabla/>

las propiedades o restricciones indicadas para cada campo (UNIQUE, NOT NULL, etc), así como los tipo de campo utilizados, deberán justificarse.

- Primero se crea la base de datos
create database rosticeria;
- Se comprueba que se ha creado
show database;
- Se Crea las tablas, para crearlas, se debe estar trabajando con la base de datos, para ello se escribe:
use rosticeria;

```
create table PROVEEDOR(  
    nifProv VARCHAR(10) NOT NULL,  
    nomProv VARCHAR(45),  
    nomContProv VARCHAR(45),  
    telsProv VARCHAR(45),  
    dirProv VARCHAR(80),  
    mailProv VARCHAR(45),  
    webProv VARCHAR(45),  
    codRegCliProv VARCHAR(10),  
    idCatProv VARCHAR(5) NOT NULL,  
    PRIMARY KEY (nifProv),  
    FOREIGN KEY (idCatProv) REFERENCES  
    categoria (idcat)  
);
```



- Ahora ya se pueden crear las tablas:

```
create table PROVEEDOR(  
    nifProv VARCHAR(10) NOT NULL,  
    nomProv VARCHAR(45),  
    nomContProv VARCHAR(45),  
    telsProv VARCHAR(45),  
    dirProv VARCHAR(80),  
    mailProv VARCHAR(45),  
    webProv VARCHAR(45),  
    codRegCliProv VARCHAR(10),
```



```

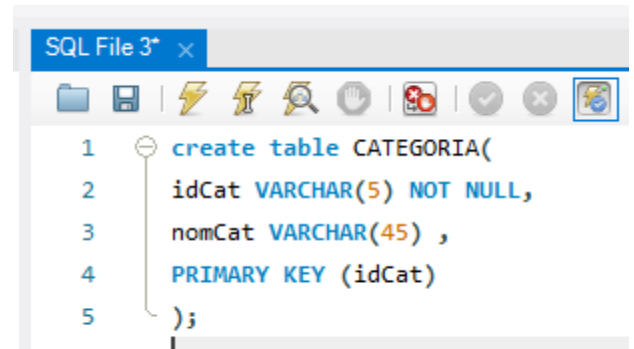
PRIMARY KEY (nifProv)
);

```

```

create table CATEGORIA(
    idCat VARCHAR(5) NOT NULL,
    nomCat VARCHAR(45) ,
    PRIMARY KEY (idCat)
);

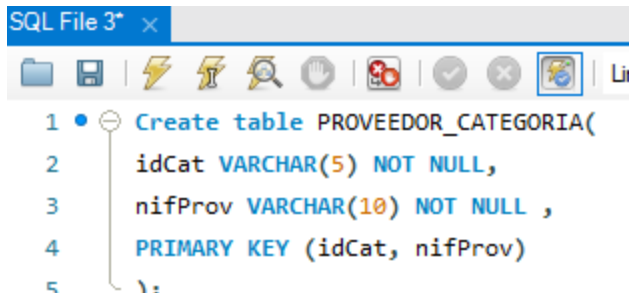
```



```

Create table PROVEEDOR_CATEGORIA(
    idCat VARCHAR(5) NOT NULL,
    nifProv VARCHAR(10) NOT NULL ,
    PRIMARY KEY (idCat, nifProv)
);

```



```

create table PRODUCTO(
    idProd VARCHAR(5) NOT NULL,
    nomProd VARCHAR(45),
    uniProd INT UNSIGNED,
    alertaStockProd INT,
    idCatProd VARCHAR(5) NOT NULL,
    PRIMARY KEY (idProd),
    FOREIGN KEY (idCatProd) REFERENCES CATEGORIA (idCat)
);

```

```

create table COMPRAS_PRODUCTO (
    idProdComProd varchar (5) NOT NULL,
    nifProvComProd varchar(10) NOT NULL,
    fechaComProd date NOT NULL,
    cantComProd int UNSIGNED,
    preComProd decimal (12,2) ,
    ivaComProd decimal (12,2),

```

```

        caducComProd date,
        PRIMARY KEY (idProdComProd, nifProvComProd),
        FOREIGN KEY (idProdComProd) REFERENCES PRODUCTO (idProd),
        FOREIGN KEY (nifProvComProd) REFERENCES PROVEEDOR (nifProv)
    );

create table ALERGENO(
    idAler varchar(5) NOT NULL,
    nomAler varchar(45),
    PRIMARY KEY (idAler)
);

create table ALERGENOS_PRODUCTO
    idProAlPr varchar(5) NOT NULL,(
    alerAlPr varchar(5) NOT NULL,
    PRIMARY KEY (idProAlPr, alerAlPr),
    FOREIGN KEY (idProAlPr) REFERENCES PRODUCTO (idProd),
    FOREIGN KEY (alerAlPr) REFERENCES ALERGENO (idAler)
);

create table DETALLE_DONACION (
    fechaDonaDetDona date NOT NULL,
    lineaDona int UNSIGNED NOT NULL,
    idProdDetDona varchar(5) NOT NULL,
    cantDetDona int UNSIGNED,
    obserDetDona varchar(200),
    PRIMARY KEY (fechaDonaDetDona, lineaDona, idProdDetDona),
    FOREIGN KEY (fechaDonaDetDona) REFERENCES DONACIONES (fechaDona),
    FOREIGN KEY (idProdDetDona) REFERENCES PRODUCTO (idProd)
);

create table DONACIONES(
    fechaDona date NOT NULL,
    PRIMARY KEY (fechaDona)
);

create table INGREDIENTE(
    idProdIngr varchar(5) NOT NULL,
    conservIngr varchar(80),
    PRIMARY KEY (idProdIngr)
);

create table ELABORADOS(
    idProdElab varchar(5) NOT NULL,
    PRIMARY KEY (idProdElab)
);

```

```

create table COMANDA(
    idCo varchar(5) NOT NULL,
    fechaCo date,
    mesaCo int UNSIGNED,
    horaCo time,
    comensCo int UNSIGNED,
    ticketCo int UNSIGNED,
    PRIMARY KEY (idCo)
);

```

```

create table COMANDA_ELABORADOS(
    idCoComElab varchar(5) NOT NULL,
    idProdComElab varchar(5) NOT NULL,
    cantComElab int UNSIGNED,
    pvpComElab decimal (12,2),
    ivaComElab int UNSIGNED,
    PRIMARY KEY (idCoComElab, idProdComElab),
    FOREIGN KEY (idCoComElab) REFERENCES COMANDA (idCo),
    FOREIGN KEY (idProdComElab) REFERENCES PRODUCTO (idProd)
);

```

```

create table PLATO(
    idPlato varchar(5) NOT NULL,
    nomPlato varchar(45),
    idTipoPlatoPlato varchar(5) NOT NULL,
    elaboPlato varchar(200),
    pvpPlato decimal (12,2),
    ivaPlato decimal (12,2),
    enMenuPlato TinyInt,
    PRIMARY KEY (idPlato),
    FOREIGN KEY (idTipoPlatoPlato) REFERENCES TIPO (idTipoPlato)
);

```

```

create table COMANDA_PLATO(
    idCoPla varchar(5) NOT NULL,
    idPIPla varchar(5) NOT NULL,
    cantCoPla int UNSIGNED,
    pvpCoPla decimal (12,2),
    ivaCoPI decimal (12,2),
    PRIMARY KEY (idCoPla, idPIPla),
    FOREIGN KEY (idCoPla) REFERENCES COMANDA(idCo),
    FOREIGN KEY (idPIPla) REFERENCES PLATO (idPlato)
);

```

```

create table TIPO(
    idTipoPlato varchar(5) NOT NULL,
    descTipoDePlato varchar(200),
    PRIMARY KEY (idTipoPlato)
);

create table INGREDIENTE_PLATO(
    idPlatoInPI varchar(5) NOT NULL,
    idProdInPI varchar(5) NOT NULL,
    cantBrutaInPI decimal (12,2),
    cantNetaInPI decimal (12,2),
    uniIngrPlato varchar(20),
    PRIMARY KEY (idPlatoInPI, idProdInPI),
    FOREIGN KEY (idPlatoInPI) REFERENCES PLATO (idPlato),
    FOREIGN KEY (idProdInPI) REFERENCES PRODUCTO (idProd)
);

```

3.Modificaciones a la base de datos

Realizar las siguientes modificaciones a la estructura de la base de datos, anotar la sentencia SQL y su salida en un documento:

3.1 Cambia el tipo de datos.

Cambia el tipo de datos del campo Unidad en la tabla PRODUCTO a Enum. Los valores que podrá recibir el campo son: Kilogramos, Litros, Unidades.

```

ALTER TABLE PRODUCTO
MODIFY COLUMN uniProd enum("Kilogramos","Litros","Unidades");

```

3.2 Crea dos nuevos campos en la tabla PRODUCTO:

3.2.1 Stock

```

ALTER TABLE PRODUCTO
ADD Stock INT;

```

3.2.2 Precio_Compra

donde posteriormente se registrará el último precio de compra de cada producto.

```
ALTER TABLE PRODUCTO  
ADD Precio_Compra decimal(12,2);
```

3.3 Agrega los campos PVP e IVA a la tabla ELABORADO.

```
ALTER TABLE ELABORADOS  
ADD pvp decimal (12,2),  
ADD iva decimal(12,2);
```

3.4 Generar una restricción

Genera una restricción de tipo UNIQUE para la combinación Fecha + Hora + Mesa, en la tabla COMANDA.

```
ALTER TABLE COMANDA  
ADD CONSTRAINT UC_comanda UNIQUE (fechaCo,horaCo,mesaCo);
```

3.5 Inventa una modificación de estructura en la base de datos.

```
ALTER TABLE INGREDIENTE  
ADD caducidadIngrediente date;
```

3.6 Añade un Campo.

Añade el campo Porcentaje_Merma a la tabla INGREDIENTES_PLATO.

```
ALTER TABLE INGREDIENTE_PLATO  
ADD Porcentaje_Merma int;
```

4. Insertar Alérgenos.

Insertar en la tabla ALERGENO los siguientes 14 alérgenos: Altramuces, Apio, Moluscos, Sésamo, Gluten, Pescado, Sulfitos, Mostaza, Crustáceos, Lactosa, Huevo, Soja, Frutos secos, Cacahuets y anotar la sentencia SQL y su salida en un documento.

insert into **ALERGENO**

(idAler, nomAler)

Values

("1", "Altramuces"),

("2", "Apio"),

("3", "Moluscos"),

("4", "Sésamo"),

("5", "Gluten"),

("6", "Pescado"),

("7", "Sulfitos"),

("8", "Mostaza"),

("9", "Crustáceos"),

("10", "Lactosa"),

("11", "Huevo"),

("12", "Soja"),

("13", "Frutos secos"),

("14", "Cacahuets");



	idAler	nomAler
▶	1	Altramuces
	10	Lactosa
	11	Huevo
	12	Soja
	13	Frutos secos
	14	Cacahuets
	2	Apio
	3	Moluscos
	4	Sésamo
	5	Gluten
	6	Pescado
	7	Sulfitos
	8	Mostaza
	9	Crustáceos

Para ver el contenido de la tabla alérgeno:

select * from alergeno;

5. Insertar en las siguientes Tablas:

Insertar en las tablas **PROVEEDORES**, **CATEGORÍAS** y **TIPOS DE PLATO** los datos que el consultor os facilitará en el tablón del consultor.

5.1 Insertar Proveedor.

insert into **PROVEEDOR**

(nifProv, nomProv, nomContProv, telsProv, dirProv, mailProv, webProv, codRegCliProv)

Values

('B65639155', 'BIRRA 365', 'NON', '960714310', 'Jeronimo Monsoriu, 58, Valencia',
'info@birra365.com', 'https://www.birra365.com', '3'),
(A50090349', 'ALBERTO POLO DISTRIBUCIONES', 'A. Polo', '976574909', 'Calle Rao Ara 8,
50014, Zaragoza', 'info@albertopolo.com', 'https://albertopolo.com/', '2'),
(A58085135', 'CARNS B', 'J. Bigorda Alberni', '933364040', 'Longitudinal 10, num. 60
Mercabarna, Barcelona', 'joaquim@carnsb.com', 'www.carnsb.com', '2'),
(B23373624', 'INDUSTRIA AVICOLA SURENA', 'J. Molina', '915077600', 'C/ Toledo 149 E Pa
28005, Madrid', 'info@inasur.es', 'www.inasur.es', '2'),
(B85227635', 'RUBIATO PAREDES SL', 'J. Pedros Riasol', '916415512', 'Calle de los Cerrajeros, 6 y
8, Alcorca, Madrid 28923', 'rubiatoparedes@rubiatoparedes.com',
'https://www.rubiatoparedes.com/', '2'),
(A28647451', 'MAKRO', 'L. Piquer', '933363111', 'Carrer A, 1, Sector C Poligono Industrial Zona
Franca 08040 Barcelona', 'atencionclientes.02@makro.es', 'https://www.makro.es/', '5'),
(B73148793', 'AGRORIZAO', 'M. Agrozao', '968425470', 'Carretera Nacional 340 (KM 614), -
30850 Totana, Murcia 30850', 'ventas@agrorizao.com', 'www.agrorizao.com', '6'),
(B87867834', 'CHEF FRUIT', 'J. Domingo', '910578136', 'Centro de Transportes de Madrid (ctm),
CL EJE 6 2, Madrid, 28053, Madrid', 'pedidos@chef-fruit.com', 'https://www.chef-fruit.com/', '6'),
(B90307034', 'FRUTAS CUEVAS', 'J. Cuevas', '954417158', 'Poligono Pagusa Calle Labrador 47,
41007, Sevilla', 'info@frutascuevas.es', 'https://www.frutascuevas.es/', '6'),
(A58058868', 'HUEVERIAS BONET, S.A.', 'A. Bonet Pedret', '933357212', 'Transversal 8, nun. 48
Multiservei I', 'dbonet@dbonet.es', 'http://dbonet.es/', '13'),
(A58241084', 'MONTSEC', 'J. del Castillo', '938498799', 'Severo Ochoa, 36 - Pol. Ind. Font del
Radium, 08403 Granollers. Barcelona.', 'info@comercialmontsec.com',
'http://www.comercialmontsec.com', '12'),
(A01189364', 'ALAVESA DE PATATAS', 'J. Suarez Tascon', '945400429', 'Poligono industrial
Lurgorri s/n Alegria-Dulantzi 01240 - Alava', 'alavesadepatatas@alavesadepatatas.com',
'http://alavesadepatatas.com/', '7')

```

Query 1 x PROVEEDOR
Limit to 1000 rows
1 • insert into PROVEEDOR
2 (nifProv, nomProv, nomContProv, telsProv, dirProv, mailProv, webProv, codRegCliProv)
3 Values
4 ("B65639155", "BIRRA 365", "NON", "960714310 ", "Jeronimo Monsoriu, 58, Valencia", "info@birra365.com",
5 ("A60177623", "BOU Cafe", "C. Gotor", "902305352 ", "Calle Rao Ara 8, 50014, Zaragoza", "info@albertopol

```

```

SQL File 3* x
Limit to 1000 rows
1 • SELECT * FROM PROVEEDOR
2

```

nifProv	nomProv	nomContProv	telsProv	dirProv	mailProv	webProv	cod
A01189364	ALAVESA DE PATATAS	J. Suarez Tascon	945400429	Poligono industrial Lurgorri s/n Alegria-Dulantzi 0...	alavesadepatas@alaves...	http://alavesadepatas.com/	7
A28647451	MAKRO	L. Piquer	933363111	Carrer A, 1, Sector C Poligono Industrial Zona F...	atencionclientes.02@makr...	https://www.makro.es/	5
A50090349	ALBERTO POLO DIST...	A. Polo	976574909	Calle Rao Ara 8, 50014, Zaragoza	info@albertopolo.com	https://albertopolo.com/	2
A58058868	HUEVERIAS BONET, ...	A. Bonet Pedret	933357212	Transversal 8, nun. 48 Multiservei I	dbonet@dbonet.es	http://dbonet.es/	13
A58085135	CARNS B	J. Bigorda Alberni	933364040	Longitudinal 10, num. 60 Mercabarna, Barcelona	joaquin@carnsb.com	www.carnsb.com	2
A58241084	MONTSEC	J. del Castillo	938498799	Severo Ochoa, 36 - Pol. Ind. Font del Radium, ...	info@comercialmontsec.com	http://www.comercialmontsec.com	12
A60177623	BOU Cafe	C. Gotor	902305352	Calle Rao Ara 8, 50014, Zaragoza	info@albertopolo.com	https://albertopolo.com/	4
B23373624	INDUSTRIA AVICOLA...	J. Molina	915077600	C/ Toledo 149 E Pa 28005, Madrid	info@inasur.es	www.inasur.es	2
B65639155	BIRRA 365	NON	960714310	Jeronimo Monsoriu, 58, Valencia	info@birra365.com	https://www.birra365.com	3
B73148793	AGRORIZAO	M. Agrorizao	968425470	Carretera Nacional 340 (KM 614), - 30850 Tota...	ventas@agrorizao.com	www.agrorizao.com	6
B85227635	RUBIATO PAREDES SL	J. Pedros Riasol	916415512	Calle de los Cerrajeros, 6 y 8, Alcorca, Madrid 2...	rubiatoparedes@rubiatopa...	https://www.rubiatoparedes.com/	2
B87867834	CHEF FRUIT	J. Domingo	910578136	Centro de Transportes de Madrid (ctm), CL EJE ...	pedidos@chef-fruit.com	https://www.chef-fruit.com/	6
B90307034	FRUTAS CUEVAS	J. Cuevas	954417158	Poligono Pagusa Calle Labrador 47, 41007, Sevilla	info@frutascuevas.es	https://www.frutascuevas.es/	6

5.2 Insertar Categoría.

insert into **CATEGORIA**

(idCat, nomCat)

Values

("1", "Aceites y vinagres"),

("2", "Carnes y Aves"),

("3", "Bebidas"),

("4", "Cafés"),

("5", "Elaborados"),

("6", "Fruta y Verdura"),

("7", "Pescados Y Mariscos"),

("8", "Conservas"),

("9", "Elaborados"),

("10", "Embutidos"),

("11", "Especies"),

```

SQL File 3* x
1 • insert into CATEGORIA
2 (idCat, nomCat)
3 Values
4 ("1", "Aceites y vinagres"),
5 ("2", "Carnes y Aves"),
6 ("3", "Bebidas"),
7 ("4", "Cafés"),
8 ("5", "Elaborados"),
9 ("6", "Fruta y Verdura"),
10 ("7", "Pescados Y Mariscos"),
11 ("8", "Conservas"),
12 ("9", "Elaborados"),
13 ("10", "Embutidos"),
14 ("11", "Especies"),
15 ("12", "Productos Lácteos"),
16 ("13", "Ovoproductos"),
17 ("14", "Pan y Bollería"),
18 ("15", ";Postres"),
19 ("16", "Harinas"),
20 ("17", "Salsas"),
21 ("18", "Semi-elaborados"),
22 ("19", "Sin Gluten"),
23 ("20", "Sin lactosa")

```



```

("12", "Productos Lácteos"),
("13", "Ovoproductos"),
("14", "Pan y Bollería"),
("15", "Postres"),
("16", "Harinas"),
("17", "Salsas"),
("18", "Semielaborados"),
("19", "Sin Gluten"),
("20", "Sin lactosa")

```

5.3 Insertar TIPO.

```

insert into TIPO
(idTipoPlato, descTipoDePlato)
Values
("1", "Primero"),
("2", "Segundo"),
("3", "Postre")

```



6. Diseñar un Menú

Diseñar un menú para la Rosticería, (que incluya recetas) para que a partir de allí llenéis las tablas PLATO, INGREDIENTES_PLATO, PRODUCTO, ALERGENOS_PRODUCTO, INGREDIENTE y ELABORADO.

INSERT INTO PLATO

```

(IdPlato, nomPlato, idTipoPlatoPlato, elaboPlato, pvpPlato, ivaPlato, enMenuPlato)
VALUES
("1", "Gambas al ajillo", "2", "Calentar las gambas en sartén al ajillo", 12.5, 2.6, 1),
("2", "Ensalada de tomate", "1", "Cortar tomates y cebolla", 3.5, 0.73, 1)

```



INSERT INTO PRODUCTO

(idProd, nomProd, uniProd, alertaStockProd,
idCatProd)

VALUES

("1", "Gambas", "Kilogramos", 10, "7"),
("2", "Ajo", "Unidades", 10, "6"),
("3", "Tomates", "Unidades", 5, "6"),
("4", "Cebolla", "Unidades", 5, "6"),
("5", "Pan Integral", "Unidades", 20, "14")

```
SQL File 3
1 • INSERT INTO PRODUCTO
2 (idProd, nomProd, uniProd, alertaStockProd, idCatProd)
3 VALUES
4 ("1", "Gambas", "Kilogramos", 10, "7"),
5 ("2", "Ajo", "Unidades", 10, "6"),
6 ("3", "Tomates", "Unidades", 5, "6"),
7 ("4", "Cebolla", "Unidades", 5, "6"),
8 ("5", "Pan Integral", "Unidades", 20, "14")
9
```

INSERT INTO INGREDIENTE_PLATO

(idPlatoInPI, idProdInPI, cantBrutaInPI,
cantNetaInPI, uniIngrPlato)

VALUES

("1", "1", 10, 10, "Unidades"),
("1", "2", 1, 1, "Unidades"),
("2", "3", 1, 1, "Unidades"),
("2", "4", 1, 1, "Unidades"),
("1", "5", 1, 1, "Unidades")

```
1 • SELECT * FROM INGREDIENTE_PLATO
2
```

Result Grid					
Filter Rows:					
	idPlatoInPI	idProdInPI	cantBrutaInPI	cantNetaInPI	uniIngrPlato
1	1	1	10.00	10.00	Unidades
1	2	1	1.00	1.00	Unidades
1	5	1	1.00	1.00	Unidades
2	3	2	1.00	1.00	Unidades
2	4	2	1.00	1.00	Unidades

INSERT INTO ALERGENOS_PRODUCTO

(IdProAlPr, alerAlPr)

VALUES

("1", "9"),
("5", "5")

INSERT INTO INGREDIENTE

(idProdIngr, conservIngr)

VALUES

("1", "Congelador"),
("2", "Temperatura Ambiente"),
("3", "Temperatura Ambiente"),
("4", "Temperatura Ambiente")

INSERT INTO ELABORADOS

(idProdElab)

VALUES

("5")

7. Insertar Registros

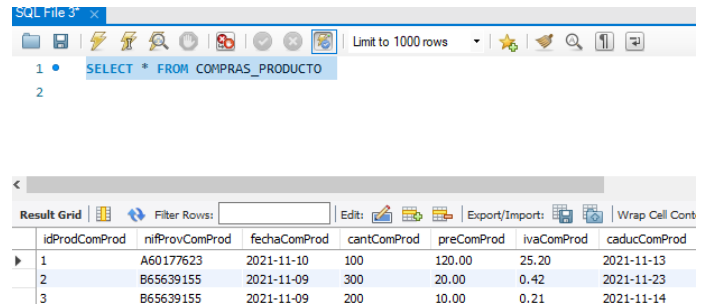
Insertar un mínimo de 10 registros para el resto de las tablas (COMPRAS_PRODUCTO, DONACIONES, DETALLE_DONACIÓN, COMANDA, COMANDA_PLATOS, COMANDA_ELABORADOS).

INSERT INTO **COMPRAS_PRODUCTO**

(IdProdComProd, nifProvComProd, fechaComProd, cantComProd, preComProd, ivaComProd, caducComProd)

VALUES

("1", "A60177623", "2021/11/10", 100, 120, 25.2, "2021/11/13"),
("2", "B65639155", "2021/11/09", 300, 20, 0.42, "2021/11/23"),
("3", "B65639155", "2021/11/09", 200, 10, 0.21, "2021/11/14")



The screenshot shows a SQL interface with a query window and a results grid. The query window contains the following SQL statement:

```
SELECT * FROM COMPRAS_PRODUCTO
```

The results grid displays the following data:

idProdComProd	nifProvComProd	fechaComProd	cantComProd	preComProd	ivaComProd	caducComProd
1	A60177623	2021-11-10	100	120.00	25.20	2021-11-13
2	B65639155	2021-11-09	300	20.00	0.42	2021-11-23
3	B65639155	2021-11-09	200	10.00	0.21	2021-11-14

INSERT INTO **DONACIONES**

(fechaDona)

VALUES

("2021/11/12"),
("2021/11/22")

INSERT INTO **DETALLE_DONACION**

(fechaDonaDetDona, lineaDona, idProdDetDona, cantDetDona, obserDetDona)

VALUES

("2021/11/12", 1, "2", 5, "Vencen mañana"),
("2021/11/22", 1, "3", 7, "N/A"),
("2021/11/22", 2, "4", 1, "No se usaron")

INSERT INTO **COMANDA**

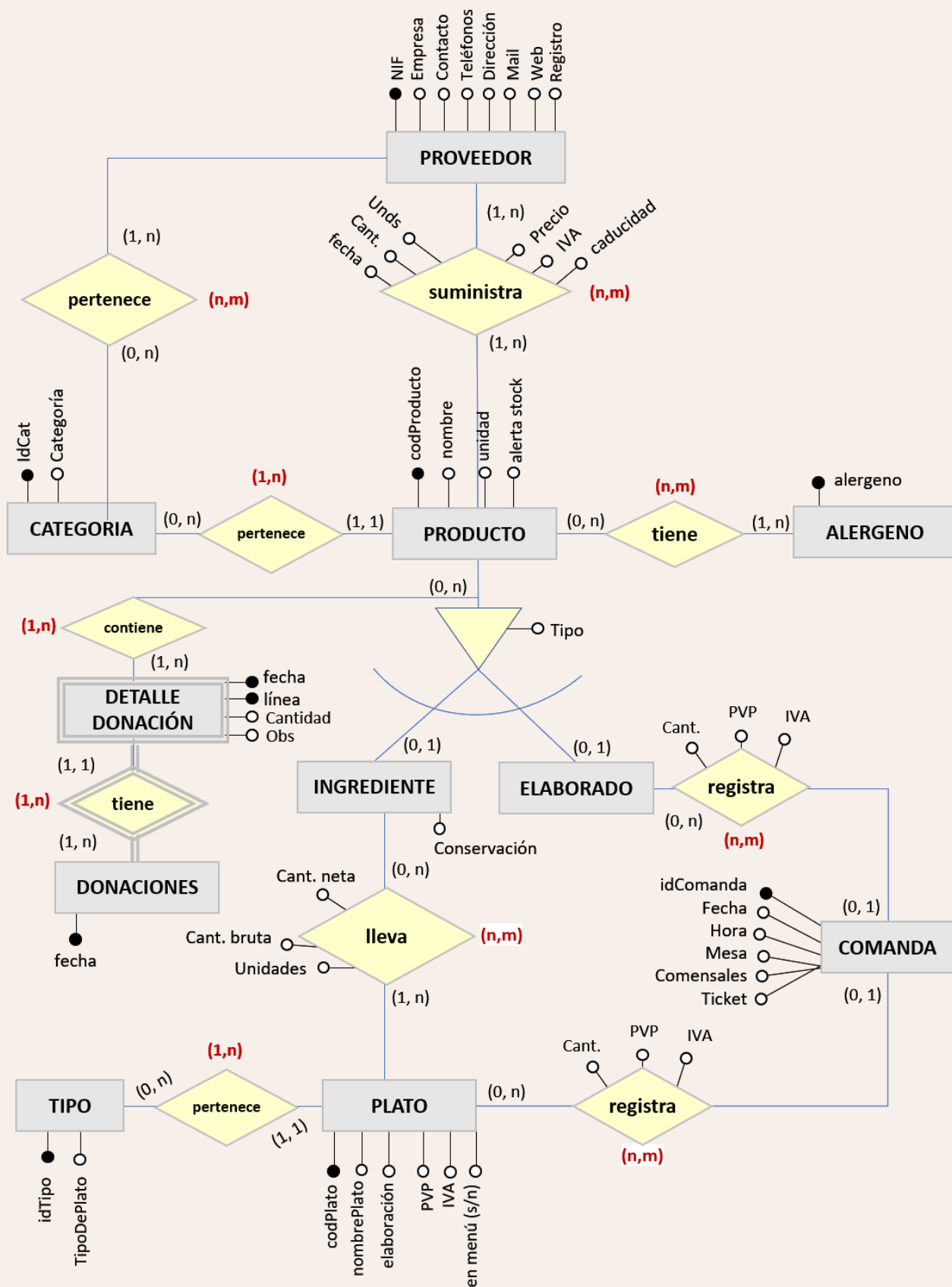
(idCo, fechaCo, mesaCo, horaCo, comensCo, ticketCo)

VALUES

("1", "2021/11/15", "1", "14:05", 2, 1),
("2", "2021/11/15", "2", "14:10", 3, 2),
("3", "2021/11/15", "3", "14:11", 2, 3)

```
INSERT INTO COMANDA_PLATO
(idCoPla, idPIPla, cantCoPla, pvpCoPla, ivaCoPl)
VALUES
("1", "1", 2, 30, 0.63),
("2", "1", 2, 30, 0.63),
("2", "2", 1, 10, 0.21),
("1", "2", 2, 20, 0.42)
```

```
INSERT INTO COMANDA_ELABORADOS
(idCoComElab, idProdComElab, cantComElab, pvpComElab, ivaComElab)
VALUES
("1", "5", 2, 0, 0)
```



Captura Pantalla cuando se está exportando la Base de Datos y sus modificaciones e inserciones.

