

2017

ALMACÉN DE PINTURA

BASES DE DATOS

PROFESORA: PIEDAD GARRIDO PICAZO

ALBERTO ROBLES DE LA FUENTE | MIGUEL YANES FERNÁNDEZ

ESCUELA UNIVERSITARIA POLITÉCNICA TERUEL

ÍNDICE

DEFINICIÓN DEL PROBLEMA.....	2
1.DISEÑO CONCEPTUAL.....	3
CASUÍSTICA	3
ESQUEMA E/R	4
2.DISEÑO LÓGICO	5
ENTIDADES.....	5
CÁLCULO RELACIONAL DE TUPLAS	6
3.DISEÑO FÍSICO	7
INSTALACIÓN Y PUESTA A PUNTO DEL SERVIDOR	7
CREACIÓN DE LA BASE DE DATOS.....	8
ESQUEMA FINAL DE LA BASE DE DATOS.....	11
APLICACIÓN WEB	12
AUTOMATIZACIÓN DE LA BASE DE DATOS - TRIGGERS.....	12
4.RECURSOS.....	13
5.BIBLIOGRAFÍA	14

ALMACÉN DE PINTURA

Se desea informatizar una empresa que es almacenista de artículos de pintura. La gestión fundamental de esta empresa es el control de stock de almacén, la petición de artículos a los proveedores que le suministran sus artículos y la venta de artículos a clientes (generalmente empresas del ramo).

La información relevante de los artículos almacenados en la empresa es la siguiente: código (único para cada artículo), descripción, características técnicas, color, tipo, stock y el stock mínimo en almacén. Los artículos son suministrados por los proveedores de la empresa a un cierto precio por kilo y con un plazo de entrega.

La información asociada a los proveedores es el NIF, nombre, dirección, teléfono y fax. A estos proveedores se les solicitan artículos mediante la confección de un pedido. Cada pedido de la empresa tiene un número único, va dirigido a un único proveedor y en él se incluye la fecha de confección del mismo. Además, un pedido está compuesto por un conjunto de líneas que contienen la información de qué artículos se solicitan, a qué precios y en qué cantidad. Claramente, el proveedor que suministra los artículos solicitados a un pedido ha de ser único y corresponder con el proveedor del pedido.

La venta de artículos a los clientes de la empresa se realiza a su vez mediante los pedidos que estos clientes efectúan. De estos clientes se conoce su NIF, nombre, dirección, teléfono y fax. Un pedido para la venta de artículos siempre lo realiza un único cliente y tiene un número único y una fecha. Los pedidos de los clientes están compuestos por un conjunto de líneas que contienen la información de los artículos solicitados por éstos, y la cantidad solicitada. Las líneas de los pedidos de los clientes se distinguen entre pendientes y entregadas. Se entiende que cuando la cantidad de un artículo solicitada por un cliente en un pedido en una cierta línea del mismo es superior al stock existente en el almacén de ese artículo, esa línea queda pendiente de entrega. Las líneas pendientes de entrega provocan que se tengan que confeccionar pedidos a los proveedores solicitando más artículos de los que existen carencias. En la empresa se desea saber qué líneas de pedidos a proveedores han aparecido por esta causa.

De los pedidos realizados a los clientes se emiten albaranes de forma que un mismo pedido se puede fraccionar en diferentes albaranes. De todo albarán se conoce su número que es único y la fecha de emisión. Finalmente, para el cobro de los pedidos realizados por clientes se emiten facturas que pueden incluir a un conjunto de albaranes. Toda factura tiene un número de factura (único), una fecha y un cliente asociado. Evidentemente el conjunto de albaranes asociados a una cierta factura han de corresponder a un mismo cliente que es al que se le factura.

1. DISEÑO CONCEPTUAL

1.1 CASUÍSTICA

El diseño elegido para la base de datos se compone de un total de 7 entidades, con 7 relaciones y un objeto agregado.

Las entidades son:

Cliente: se compone de los datos de cada cliente y se relaciona con *Pedido* mediante la relación *hace*.

Pedido: la entidad pedido gestiona únicamente los pedidos realizados por los clientes al almacén, no los que hace el propio almacén a sus proveedores. Cada pedido tiene restricción de existencia de algún artículo. Cada pedido tendrá un atributo que indicará si el pedido se ha entregado o no (dicho atributo será de tipo no nulo). Es una entidad débil de Cliente, por lo que cada pedido se identificará también en función del cliente que lo ha realizado.

Almacén: entidad principal que almacena registros de las facturas de los pedidos realizados por los clientes, además de gestionar los artículos y su stock. El almacén tendrá obligatoriamente algún artículo almacenado.

Artículo: almacena todos los datos necesarios de los artículos, incluyendo el stock actual y mínimo, y se relaciona tanto con *Pedido*, como con *Almacén*, como también con *Proveedor*, ya que los proveedores son los que suministran dichos artículos (aun así, un artículo podrá no tener un proveedor directamente relacionado). Tiene dos atributos compuestos, uno para los datos y otro para el stock. El atributo compuesto del stock incluye un atributo llamado *max_stock*, y otro llamado *min_stock*. Estos atributos servirán para indicar la cantidad de stock máxima y mínima de cada artículo, para su posterior gestión automática. Estos atributos relacionados con el stock serán no nulos.

Proveedor: similar a *Cliente*, con la diferencia de que en la relación con *Artículo* se almacenará el precio de venta.

Albarán: se relaciona con *Pedido* mediante el objeto agregado *entrega*. A su vez, este objeto agregado se relaciona con *Factura*.

Factura: se relaciona con el objeto agregado *genera*. Por último, se relaciona también con la entidad *Almacén*.

-
1. *Cuando la cantidad de un artículo está bajo mínimos, se mandará automáticamente un aviso al proveedor para reestablecer el stock.*
 2. *Cuando la cantidad de un artículo solicitada por un cliente en un pedido en una cierta línea del mismo es superior al stock existente, esa línea quedará pendiente de entrega.*
-

1.2 DIAGRAMA ENTIDAD/RELACIÓN

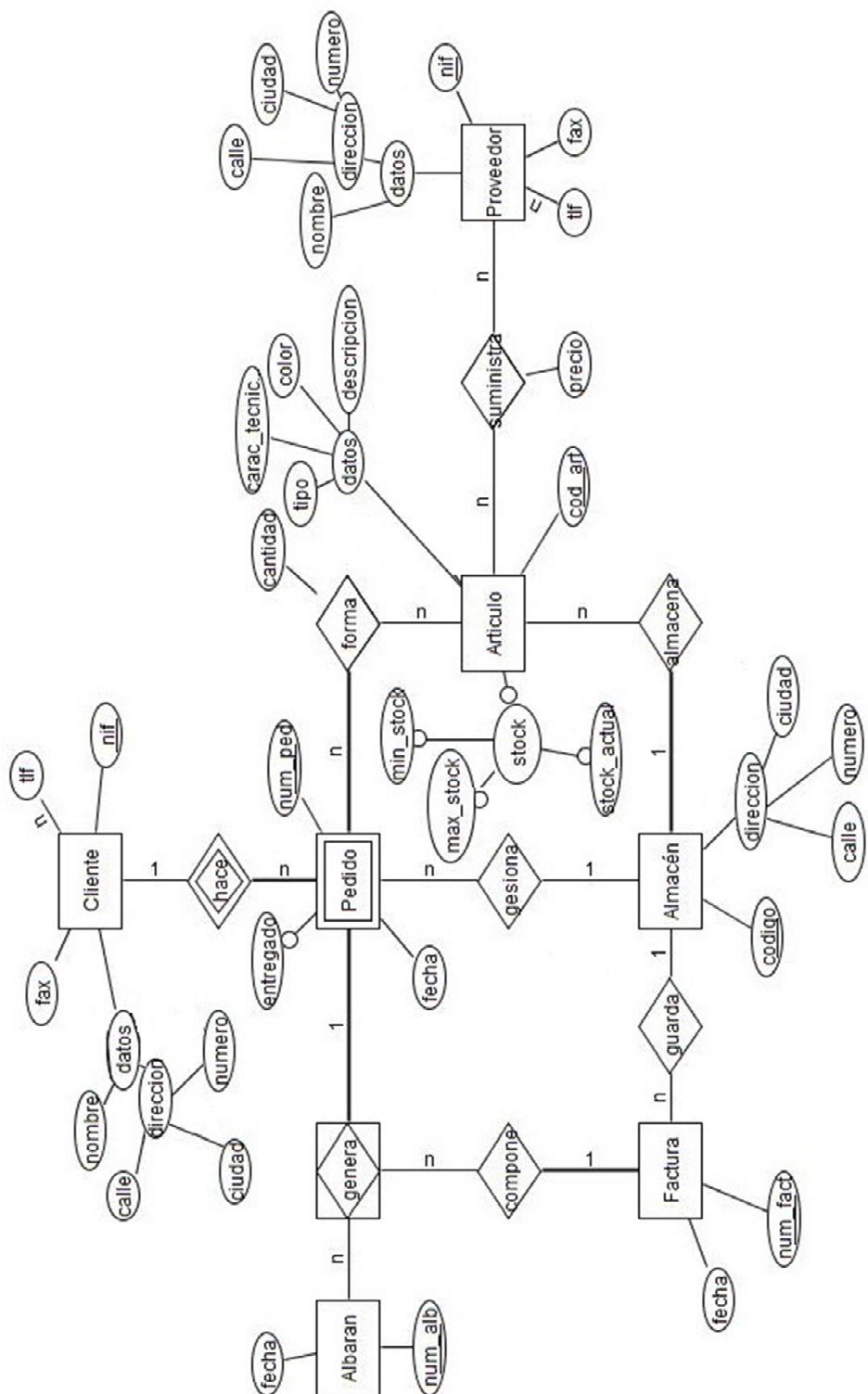


Figura 1. Diagrama Entidad/Relación - <http://dereditor.sourceforge.net/>

2. DISEÑO LÓGICO

CLIENTE (nif: dom_nif, datos.direccion.calle: dom_calle, datos.direccion.ciudad: dom_ciudad, datos.direccion.numero: dom_numero, datos.nombre: dom_nombre, fax: dom_fax, num: dom_num)

CP {nif}

TLF_CLIENTE (nif: dom_nif, tlf: dom_tlf)

CP {nif, tlf}

CAj {nif} hace referencia a **CLIENTE**

PEDIDO (num_ped: dom_num, fecha: dom_fecha, entregado: dom_entregado, nif: dom_nif, cod_alm: dom_cod_alm)

CP {numero, nif}

CAj {nif} hace referencia a **CLIENTE**

CAj {cod_alm} hace referencia a **ALMACEN**

VNN {entregado}

ARTICULO (cod_art: dom_cod, stock.stock_actual: dom_stock_actual, stock.min_stock: dom_min_stock, stock.stock_alerta: dom_stock_alerta, datos.color: dom_color, datos.tipo: dom_tipo, datos.descripcion: dom_descripcion, datos.carac_tec: dom_carac_tec, num_ped: dom_num, cantidad: dom_cantidad, cod_alm: dom_cod_alm, nif: dom_nif)

CP {cod_art}

CAj {num_ped, nif} hace referencia a **PEDIDO**

CAj {cod_alm} hace referencia a **ALMACEN**

VNN {stock.stock_actual}

VNN {cantidad}

ALMACEN (cod_alm: dom_cod_alm, direccion.ciudad: dom_ciudad, direccion.calle: dom_calle, direccion.numero: dom_numero)

CP {cod_alm}

PROVEEDOR (nif: dom_nif, fax: dom_fax, datos.nombre: dom_nombre, datos.direccion.ciudad: dom_ciudad, datos.direccion.numero: dom_numero)

CP {nif}

TLF_PROV (nif: dom_nif, tlf: dom_tlf)

CP {nif, tlf}

CAj {nif} hace referencia a **PROVEEDOR**

SUMINISTRA (precio: dom_precio, nif: dom_nif, cod_art: dom_cod)

CP {nif, codigo}

CAj {nif} hace referencia a **PROVEEDOR**

CAj {codigo} hace referencia a **ARTICULO**

Único {nif, precio}

ALBARAN (num_alb: dom_num_alb, fecha: dom_fecha, num_ped: dom_num_ped, num_alb: cod_num_alb, nif: dom_nif)

CP {num_alb}

CAj {num_ped, nif} hace referencia a **PEDIDO**

CAj {num_fact} hace referencia a **FACTURA**

FACTURA (num_fact: dom_num_fact, fecha: dom_fecha, cod_alm: dom_cod_alm)

CP {num_fact}

CAj {cod_alm} hace referencia a **ALMACEN**

CÁLCULO RELACIONAL DE TUPLAS

Restricción Pedido-Artículo

Px: P, Ax: A

$\forall Px(P(Px) \rightarrow \exists Ax(A(Ax) \wedge Ax.\text{num_ped} = Px.\text{num_ped}))$

Restricción Almacén-Artículo

Alx: Al, Ax: A

$\forall Alx(Al(Alx) \rightarrow \exists Ax(A(Ax) \wedge Ax.cod_alm = Alx.cod_alm))$

Restricción Pedido-Albarán

Px: P, Ax: A

$\forall Px(P(Px) \rightarrow \exists Ax(A(Ax) \wedge Ax.num_ped = Px.num_ped))$

Restricción Albarán-Factura

Fx: F, Alx: Al

$\forall Fx(F(Fx) \rightarrow \exists Alx(Al(Alx) \wedge Alx.\text{num_fact} = Fx.\text{num_fact}))$

3. DISEÑO FÍSICO

3.1 INSTALACIÓN Y PUESTA A PUNTO DEL SERVIDOR

Para llevar a cabo el diseño físico del proyecto, hemos decidido usar una máquina virtual sobre un servidor de la universidad basado en el SO Ubuntu Server 14.04 LTS. Dicha máquina virtual ya nos ha sido proporcionado con el SO instalado y con la IP externa configurada. Dicha IP es 155.210.68.183. La gestión remota de la máquina virtual se ha realizado mediante el software de conexión SSH PuTTY.

- Instalación de DBMS MySQL.

1. Instalar servidor.

```
| apt-get update  
| apt-get install mysql-server
```

2. Securizar.

```
| /usr/bin/mysql_secure_installation
```

3. Configuración de conexiones externas a la base de datos.

```
| vi /etc/mysql/my.cnf  
| Comentar la linea:  
| bind-address = 127.0.0.1
```

- Instalación de servidor web Apache.

```
| apt-get install apache2
```

- Instalación módulo PHP para Apache y conector ODBC para PHP.

1. Instalación modulo PHP para APACHE2.

```
| apt-get install php5-common php5
```

2. Instalación módulo MySQL para PHP.

```
| apt-get install php5-mysql  
| añadir a /etc/php5/apache2/php.ini:  
| extension=mysql.so
```

3. Instalación SGDBD (phpmyadmin) mediante página PHP.

```
| apt-get install phpmyadmin  
| crear enlace simbólico al phpadmin:  
| ln -s /usr/share/phpmyadmin phpmyadmin
```

En lo relativo a la aplicación web y el código php:

1. Conectarse a MySQL creando instancia de la clase base PDO.

```
try{  
    $conexion = new PDO('mysql:host=155.210.68.183;dbname=almacen_pintura', '  
        root', 'bd2');  
} catch (PDOException $e){  
    echo "Error: " . $e->getMessage();  
}
```

Figura 2 – Conexión a la base de datos por código php

2. Gestión de formularios mediante método POST.

3. Prevención de SQL Injection

```
$nombre = filter_var(strtolower($_POST['nombre']), FILTER_SANITIZE_STRING);
```

Figura 3 – Prevención de SQL Injection y ejemplo de método POST

4. Algoritmo de encriptación hash para contraseñas

```
$password_hash = hash('sha512', $password);
```

Figura 4 – Ejemplo cifrado de contraseñas

5. Control de sesión de usuario.

```
$_SESSION['nombre'] = $nombre;
```

Figura 5 - Sesión

6. Framework Bootstrap.

3.2 CREACIÓN DE LA BASE DE DATOS

El código de la creación de las distintas tablas (en orden alfabético) es el siguiente:

Tabla *albaran*

```
CREATE TABLE `albaran` (  
    `num_alb` INT(15) NOT NULL AUTO_INCREMENT,  
    `num_ped` INT(50) NOT NULL,  
    `nif` VARCHAR(10) NOT NULL,  
    `num_fact` INT(15) NOT NULL,  
    `fecha` DATETIME NOT NULL,  
    PRIMARY KEY (`num_alb`),  
    INDEX `num_ped` (`num_ped`, `nif`),  
    INDEX `num_fact` (`num_fact`),  
    CONSTRAINT `FK_albaran_factura` FOREIGN KEY (`num_fact`)  
        REFERENCES `factura` (`num_fact`),  
    CONSTRAINT `FK_albaran_pedido` FOREIGN KEY (`num_ped`, `nif`)  
        REFERENCES `pedido` (`num_ped`, `nif`)  
)
```

GII – Bases de Datos 1

Almacén de Pintura – Alberto Robles y Miguel Yanes

Tabla *almacen*

```
CREATE TABLE `almacen` (
  `cod_alm` VARCHAR(10) NOT NULL,
  `ciudad` VARCHAR(50) NOT NULL,
  `calle` VARCHAR(50) NOT NULL,
  `numero` INT(10) NOT NULL,
  PRIMARY KEY (`cod_alm`)
)
```

Tabla *articulo*

```
CREATE TABLE `articulo` (
  `cod_art` VARCHAR(10) NOT NULL,
  `carac` VARCHAR(500) NULL DEFAULT NULL,
  `descripcion` VARCHAR(100) NULL DEFAULT NULL,
  `tipo` ENUM('temple','esmalte','lacado','vinilo','plastica','decorativa') NULL DEFAULT NULL,
  `color` VARCHAR(10) NULL DEFAULT NULL,
  `stock` INT(10) NOT NULL,
  `cod_alm` VARCHAR(10) NOT NULL,
  `cantidad` INT(11) NULL DEFAULT NULL,
  PRIMARY KEY (`cod_art`),
  INDEX `cod_alm` (`cod_alm`),
  CONSTRAINT `FK_articulo_almacen` FOREIGN KEY (`cod_alm`)
    REFERENCES `almacen` (`cod_alm`)
)
```

Tabla *cliente*

```
CREATE TABLE `cliente` (
  `nif` VARCHAR(10) NOT NULL DEFAULT '',
  `nombre` VARCHAR(10) NOT NULL DEFAULT '',
  `ciudad` VARCHAR(50) NOT NULL DEFAULT '',
  `calle` VARCHAR(50) NOT NULL DEFAULT '',
  `numero` INT(10) NOT NULL DEFAULT '0',
  `fax` INT(15) NULL DEFAULT NULL,
  `direccion` VARCHAR(150) NULL DEFAULT NULL,
  `password` VARCHAR(200) NOT NULL DEFAULT '',
  PRIMARY KEY (`nif`)
)
```

Tabla *factura*

```
CREATE TABLE `factura` (
  `num_fact` INT(15) NOT NULL AUTO_INCREMENT,
  `fecha` DATETIME NOT NULL,
  `cod_alm` VARCHAR(10) NOT NULL,
  PRIMARY KEY (`num_fact`),
  INDEX `cod_alm` (`cod_alm`),
  CONSTRAINT `FK_factura_almacen` FOREIGN KEY (`cod_alm`)
    REFERENCES `almacen` (`cod_alm`)
)
```

Tabla forma

```
CREATE TABLE `forma` (
  `cod_art` VARCHAR(10) NOT NULL,
  `num_ped` INT(50) NOT NULL,
  `nif` VARCHAR(10) NOT NULL,
  `cantidad` INT(11) NULL DEFAULT NULL,
  PRIMARY KEY (`num_ped`, `nif`, `cod_art`),
  INDEX `num_ped` (`num_ped`, `nif`),
  INDEX `cod_art` (`cod_art`),
  CONSTRAINT `FK_forma_articulo` FOREIGN KEY (`cod_art`)
    REFERENCES `articulo` (`cod_art`),
  CONSTRAINT `FK_forma_pedido` FOREIGN KEY (`num_ped`, `nif`)
    REFERENCES `pedido` (`num_ped`, `nif`)
)
```

Tabla pedido

```
CREATE TABLE `pedido` (
  `num_ped` INT(50) NOT NULL AUTO_INCREMENT,
  `nif` VARCHAR(10) NOT NULL,
  `fecha` DATETIME NULL DEFAULT NULL,
  `entregado` VARCHAR(50) NULL DEFAULT 'pendiente',
  `cod_alm` VARCHAR(10) NULL DEFAULT NULL,
  PRIMARY KEY (`num_ped`, `nif`),
  INDEX `cod_alm` (`cod_alm`),
  INDEX `nif` (`nif`),
  CONSTRAINT `FK_pedido_almacen` FOREIGN KEY (`cod_alm`)
    REFERENCES `almacen` (`cod_alm`),
  CONSTRAINT `FK_pedido_cliente` FOREIGN KEY (`nif`) REFERENCES
    `cliente` (`nif`)
)
```

Tabla proveedor

```
CREATE TABLE `proveedor` (
  `nif` VARCHAR(10) NOT NULL,
  `nombre` VARCHAR(10) NOT NULL,
  `ciudad` VARCHAR(50) NOT NULL,
  `calle` VARCHAR(50) NOT NULL,
  `numero` INT(10) NOT NULL,
  `fax` INT(15) NOT NULL,
  PRIMARY KEY (`nif`)
)
```

Tabla suministra

```
CREATE TABLE `suministra` (
  `nif` VARCHAR(10) NOT NULL,
  `cod_art` VARCHAR(10) NOT NULL,
  `precio` DOUBLE NOT NULL,
  PRIMARY KEY (`nif`, `cod_art`),
  INDEX `nif` (`nif`),
  INDEX `cod_art` (`cod_art`),
  CONSTRAINT `FK_suministra_articulo` FOREIGN KEY (`cod_art`)
    REFERENCES `articulo` (`cod_art`),
  CONSTRAINT `FK_suministra_proveedor` FOREIGN KEY (`nif`)
    REFERENCES `proveedor` (`nif`)
)
```

Tabla tlf_cliente

```
CREATE TABLE `tlf_cliente` (
  `nif` VARCHAR(10) NOT NULL,
  `tlf` INT(15) NOT NULL,
  PRIMARY KEY (`nif`, `tlf`),
  INDEX `nif` (`nif`),
  CONSTRAINT `FK_tlf_cliente_cliente` FOREIGN KEY (`nif`)
  REFERENCES `cliente` (`nif`) ON DELETE CASCADE
)
)
```

Tabla tlf_proveedor

```
CREATE TABLE `tlf_proveedor` (
  `nif` VARCHAR(10) NOT NULL,
  `tlf` INT(15) NOT NULL,
  PRIMARY KEY (`nif`, `tlf`),
  INDEX `nif` (`nif`),
  CONSTRAINT `FK_tlf_proveedor_proveedor` FOREIGN KEY (`nif`)
  REFERENCES `proveedor` (`nif`)
)
)
```

3.3 ESQUEMA FINAL DE LA BASE DE DATOS

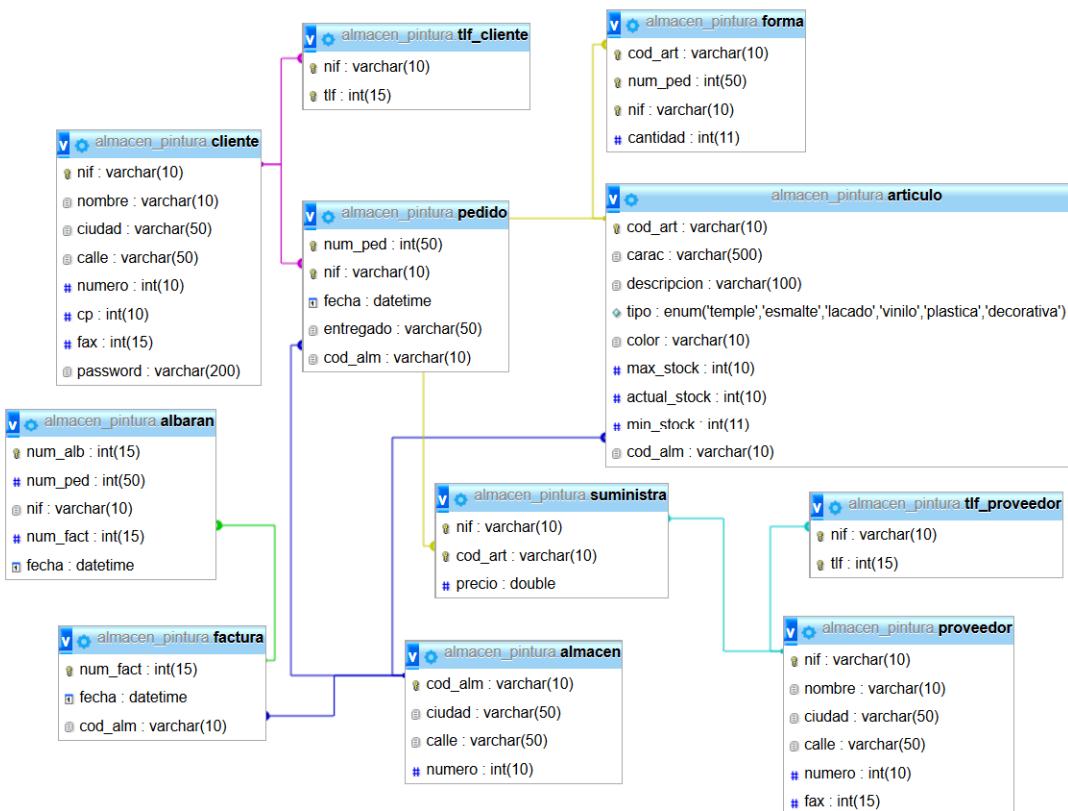


Figura 6. Diagrama E/R o esquema conceptual del Sistema de Información (SI) por <https://www.phpmyadmin.net/>

En el esquema generado por phpmyadmin no se representan gráficamente las relaciones dobles, que corresponderían a las claves foráneas de albarán-pedido (nif, num_ped) y a pedido-forma (nif, num_ped).

3.4 APPLICACIÓN WEB

La aplicación web que gestionará los distintos pedidos relativos al almacén de pintura se gestionará mediante sesiones dedicadas a los distintos clientes, de tal forma que sea necesario iniciar sesión previamente para poder realizar, ver, modificar o eliminar algún pedido. Por simplicidad en la solución, la modificación de los pedidos no se puede realizar desde la propia aplicación web sino tan solo desde el lado de la base de datos. De todas formas, la modificación de pedidos tan solo afecta al momento de añadir un albarán a un pedido (un único albarán, también por simplificar la aplicación) y por tanto marcarlo como entregado, por lo que lo más lógico es que sea el propio almacén quien se encargue de dichos albaranes y no el propio cliente. Además, todos los nuevos clientes tendrán que registrarse previamente.

El inicio de sesión se realiza usando las contraseñas cifradas por un método hash según lo comentado en el apartado 3.1.

Toda la programación relacionada con la base de datos y con la aplicación web ha sido realizada en php.

3.5 AUTOMATIZACIÓN DE LA BASE DE DATOS – TRIGGERS

Implementamos 2 triggers para automatizar la gestión de los datos en la base de datos.

albaran_pedido

Este trigger se dispara al insertar una fila en la tabla albarán y su función consiste en actualizar el pedido correspondiente para modificar el estado de dicho pedido a ‘entregado’.

```
CREATE TRIGGER `albaran_pedido`  
AFTER INSERT ON `albaran`  
FOR EACH ROW UPDATE pedido  
    SET entregado = 'entregado'  
    WHERE num_ped = NEW.num_ped AND nif = NEW.nif
```

restar_stock

Para un nuevo pedido relacionado con forma, se actualiza el artículo correspondiente para restar el stock solicitado.

```
CREATE TRIGGER `restar_stock`  
AFTER INSERT ON `forma`  
FOR EACH ROW UPDATE articulo a  
    SET a.actual_stock = a.actual_stock - NEW.cantidad  
    WHERE NEW.cod_art = a.cod_art
```

4. RECURSOS

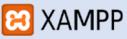
DEREDITOR	Software de diseño Entidad-Relación http://dereditor.sourceforge.net/
 XAMPP	XAMPP es un servidor independiente de plataforma de código libre. Lo hemos empleado para pruebas en local. - https://sourceforge.net/projects/xampp/
 HeidiSQL	Administrador de base de datos (software libre). Lo hemos empleado principalmente para pruebas en local. - https://www.heidisql.com/download.php
 phpMyAdmin	Administrador de base de datos. https://www.phpmyadmin.net/downloads/
 PHP	Lenguaje de programación de uso general de código del lado del servidor. https://secure.php.net/downloads.php
 MySQL	Sistema gestor de bases de datos relacionales. https://www.mysql.com/
 VMware	Software de virtualización. Ubuntu Server 14.04 LTS. https://www.vmware.com/
 Bootstrap	Bootstrap - https://getbootstrap.com/
 Moodle	Recursos del ADD: Bases de Datos - https://moodle2.unizar.es/add/course/view.php?id=13993 Sistemas de Información - https://moodle2.unizar.es/add/course/view.php?id=16396
 PuTTY	PuTTY – Conexiones SSH al servidor - http://www.putty.org/
 WinSCP	WinSCP – Subida de archivos al servidor - https://winscp.net/eng/download.php

Figura 7 – Tabla de recursos empleados en el mini proyecto

5. BIBLIOGRAFÍA:

<http://www.slideshare.net/josecuartas/transformar-modelo-entidad-relacion-a-modelo-logico> - (diapositiva nº 6)

<https://www.w3schools.com/sql/>

<https://dev.mysql.com/doc/refman/5.7/en/trigger-syntax.html>

<https://stackoverflow.com/questions/43726354/mysql-after-update-trigger>



Este obra está bajo una [licencia de Creative Commons Reconocimiento-CompartirIgual 4.0 Internacional](#).

