Parte 2

- Esta es la recuperación del segundo trabajo, que trata sobre modelado E/R y consultas SQL avanzadas.
- En la guía docente se indica que este trabajo serán individuales o por parejas, pero se admitirán entregas de hasta 3 alumnos, para los que lo prefieran. En principio, los grupos serán los mismos que para trabajos anterores, pues hay apartados del trabajo que son continuación del trabajo 1. Sin embargo, en caso de que no podáis o no queráis seguir en el mismo grupo, podéis separaros del compañero, pero no iros a otro grupo. Tenedme informado por mail lo antes posible en ese caso.

Parte E/R

Crear el diagrama E/R correspondiente al siguiente enunciado. En el resultado final deberá de aparecer:

1. Una relación ternaria (y sólo una)
2. Una agregación (y sólo una)
3. Una relación ISA (y sólo una)
4. Una relación 1:1 (y sólo una)
5. Exactamente tres entidades débiles
6. Varias relaciones binarias 1:N, N:M

Enunciado:

En una empresa de carpintería se ha de registrar toda esta información:

1. **clientes**, tienen un CIF, un nombre o razón social, dirección, teléfono, nro cuenta bancaria, fax y email. los clientes:
    1. Tienen una forma de pago por defecto de entre las definidas en la tabla de formas de pago
    2. Son de una ciudad de entre las definidas en la tabla de ciudades
2. **facturas**, tienen un número secuencial único que las identifica, y tienen una fecha, además
    1. Están asociadas a un cliente de los de la tabla de clientes
    2. Tienen una forma de pago de las de la tabla de formas de pago
3. **lineas de factura** (una factura esta compuesta por varias lineas de factura, cada línea representa una compra de una cantidad de un determinado artículo).

    Las líneas de factura contienen un campo cantidad, indicando cuantas unidades de ese artículo se han comprado y un campo precio unitario. Además:

    1. El artículo será uno de los de la tabla productos
    2. Y la factura a la que pertenece esa línea de factura será una factura existente en la tabla de facturas.

    Cuando se borra una factura queremos que se borren en cascada todas sus líenas de factura.
4. **formas de pago**, es una tabla que contiene un campo con un acrónimo que representa una forma de pago y otro campo que describe ese acrónimo. Por ejemplo: el acrónimo puede ser CONTA y la descripción "*pago al contado*", el acrónimo puede ser LETRA_90 y la

descripción *"letra a 90 días"*. Utilizaremos el acrónimo como clave primaria.

Cuando una forma de pago cambie de acrónimo queremos que las facturas y clientes que tengan asociada esa forma de pago también cambien la referencia a la misma.

Un cliente puesta tener **autorizadas** varias formas de pago, pero solo tiene una **forma de pago por defecto**.

> **Nota**: Desde la base de datos, en principio, no controlaremos que la forma de pago por defecto de un cliente sea una de las formas de pago autorizadas. Ese control lo delegaremos en la lógica de la aplicación que trabaja contra esa base de datos. No obstante, puedes anexar una propuesta de cómo controlar esto desde la base de datos, y se valorará como nota extra.

5. **tipos de clientes**, es una tabla que contiene un acrónimo que representa el tipo de cliente, y otro campo descripción que lo explica. Por ejemplo: el acrónimo POT representaría que el cliente es potencial, MED que es mediano, MUL que es multinacional etc ... Utilizaremos el acrónimo como clave primaria.

   Cuando un tipo cambie de acrónimo queremos que los clientes que tengan asociados ese tipo también cambien la referencia al mismo.

   Un cliente pueda estar asociado a varios tipos de cliente simultáneamente, y naturalmente, varios clientes podrían ser del mismo tipo.

6. **provincias**, es una tabla que contiene un acrónimo de la provincia (p.e. BU, VA, MAD) y otro campo con el nombre completo de la provincia (p.e. Burgos, Valladolid, Madrid). Intenta hacer un check que compruebe que metemos los nombres de las provincias en mayúsculas.

   Cuando una provincia cambie de acrónimo queremos que las ciudades de esa provincia también cambien la referencia a la misma.

7. **ciudades**, es una tabla que contiene 2 campos: el nombre de la ciudad y el acrónimo de la provincia. El acrónimo de la provincia será uno de los de la tabla de provincias. Las ciudades se identifican por la combinación de ambos campos.

8. **color** tiene 4 campos, Los campos R, G y B representan las coordenadas RGB que van de 0 a 255 del color, y el campo nombre representa el nombre del color. No puede 2 colores con las mismas coordenadas RGB, ni puede haber tampoco 2 colores con el mismo nombre. Puedes elegir como clave cualquiera de esas 2 claves candidatas, a tu gusto...

   Cuando un color cambie de valor en su clave primeria queremos que los productos que tengan asociados ese color también cambien la referencia al mismo.

9. **productos**, es una tabla que contiene un campo familia que describe una familia de productos (por ejemplo la familia puertas, o la familia camas), una referencia que es única para cada familia, además los productos tienen un campo color que será uno de los colores de la tabla colores. Cada producto se identifica por estos 3 campos, es decir puede haber una puerta con una referencia 5, y una cama también con referencia 5, incluso puede haber 2 camas con referencia 5, pero lo que no puede haber son 2 camas con referencia 5 del mismo color en la misma familia).

   La tabla de productos tiene un campo existencias con el número de unidades de ese producto que quedan en el almacén.

10. Habrá de mantenerse una tabla de **comerciales** con los campos: DNI, nombre, ape1, ape2, tfno, e_mail

11. Asimismo, habrá de crear una tabla de **contactos** que relacione cada comercial, con cada cliente, en las distintas fechas en la que lo visitó (un comercial puede visitar al mismo

cliente varias veces, pero en fechas distintas, no obstante el mismo cliente no puede recibir más de una visita el mismo día).

12. Haz los ajustes necesarios para poder representar en las tablas que para cada tipo de cliente **siempre** hay un comercial (y sólo uno) responsable de ese tipo. Por ejemplo, Pepe es el responsable de los clientes de tipo grande y Juan de los de tipo pequeño. Representar también que cada comercial puede ser responsable de **uno ó cero** tipos de clientes.
13. Además se registrarán los **pedidos**. Un pedido tiene un número de pedido secuencial y una fecha. Cada pedido se corresponde siempre con un cliente, pero en ese sentido distinguiremos entre dos tipos de pedidos*:*
    1. Pedidos por cantacto: que han tenido lugar como consecuencia de un contacto (una visita que realizó el comercial al cliente) de la tabla de contactos. En estos pedidos se debe poder deducir el cliente a través del contacto.
    2. Pedidos sin contacto: que no han tenido lugar como consecuencia de un contacto, en los que el cliente se conoce directamente a través de una relación con el correspondiente cliente.
14. Cada línea de factua habrá de referenciar a un pedido, de manera que una línea de factura pueda pertenecer a la vez a un pedido (que indica cuando se pide y por quien) y a una factura (que indica cuando se abona la cantidad y qué cantidad). Toda línea de factura se corresponde con un pedido, pero puede que esté dada de alta en la base de datos durante un tiempo sin que aún se corresponda con una factura, a la espera de que dicha línea se facture (se asocie a una factura).

    Así puede hacer un pedido de 20 líneas y luego fraccionarlo más tarde en 5 facturas de 4 líneas, por ejemplo; o al revés: se puede hacer una factura que englobe/mezcle líneas de varios pedidos hechos con anterioridad.

    > **Nota**: Desde la base de datos, en principio, no controlaremos que el CIF de la factura y del pedido sean el mismo para cada línea. Ese control lo delegaremos en la lógica de la aplicación que trabaja contra esa base de datos. No obstante, puedes anexar una propuesta de cómo controlar esto desde la base de datos, y se valorará como nota extra.

1. **ENTREGABLES**
    1. **Deberás de entregar un word o PDF con el diagrama**. Conteniendo:
        1. Razonamiento de las cardinalidades en la ternaria
        2. Opcionalmente, comentarios explicativos de las decisiones de diseño tomadas.
    2. **Entregar un script con los create tables correspondientes para representar ese modelo E/R**. Para la relación ISA, utiliza la solución de una sola tabla.
    3. No hay vídeos porque en el examen oral, en la parte de E/R te preguntaré preferentemente cosas de este trabajo. Si no entregas el trabajo, o lo entregas muy mal, tendré que buscar otras preguntas.

Parte Consultas SQL

Para esta parte utiliza las tablas de la parte 1 del trabajo (normalización+alg+SQL)

- Si necesitas en algún caso añadir alguna tabla más o algún campo, utiliza los CREATE/ALTER tables oportunos, y justifícalos con comentarios en el script.
- En particular necesitarás alguna relación reflexiva. Intenta no usar las de siempre (jefes, partes, parientes ...). ¡Qué tengan que ver algo con tu trabajo 1! ¡sé un poco original! Te

podemos ayudar a refinar una idea, si nos la comentas en el foro

Para que te sea más fácil explicar cada SELECT, idealmente cada una llevará sus propias filas de ejemplo:

- Utiliza ROLLBACK o DELETE o TRUNCATE (lo que prefieras) después de cada SELECT para limpiar las tablas antes de insertar las filas de ejemplo de la consulta siguiente.
- Si una consulta solo necesita filas de algunas tablas, sus filas de ejemplo solo harán INSERT en las tablas estrictamente necesarias.

Se pide, por cada miembro del equipo:

Nota: Si hay un solo script en el que no se identifica de quién es cada consulta, os pongo a todos lo miembros del grupo la misma nota haciendo la media de cada uno de los siguientes puntos.

1. Una consulta que utilice la nueva relación reflexiva en el cálculo de una función de agregación. Naturalmente, la consulta tendrá agrupamiento.

   Nota: Esta consulta no puede tener subconsultas

2. Una consulta que represente un anidamiento de funciones de agregación. Haz 2 versiones, una con subconsulta en el FROM y otra con subconsulta en el WITH

   Nota: Antes de hacerla mírate la sección 6.1.2.1 de los apuntes del tema 5

3. Modifica la versión con WITH haciendo que el resultado de la consulta anterior sea otra subconsulta que puedas utilizar en una nueva consulta principal.

   Nota: Antes de hacerla mírate las transparencias “Mezcla de las 2 anteriores usando WITH” en Vídeos sesión práctica 11 > Transparencias del Vídeo (Consultas tipo examen o trabajo), así como el correspondiente vídeo

4. Dos cocientes relacionales con subconsulta en HAVING. En lugar de pedir elementos del dividendo que se relacionan con todos los elementos del divisor, puedes relajarlo a elementos del dividendo que se relacionan al menos con un *porcentaje* de elementos del divisor, seguramente así te sea más fácil reutilizar algunas filas de ejemplo

   1. Uno de los 2 cocientes ha de necesitar algún COUNT(DISTINCT) en la subconsulta, y el otro lo ha de necesitar en la consulta principal. Tendrás que justificar en el script esa necesidad a través de la información de los create tables (i.e., las claves). Por ejemplo, como la clave es X, y agrupamos por Y, se pueden repetir las Zs dentro de cada grupo.
   2. Al menos uno de los dos cocientes deberá de tener un WHERE en la subconsulta

   Nota 1: La subconsulta no podrá ser correlacionada:

   Nota 2: Antes de hacerlo mírate todos los casos que aparecen en la sección 6.3 de los apuntes del tema 5

5. Una consulta con subconsulta en el WHERE, que se pueda hacer tanto con MAX/MIN en la subconsulta, o como con ALL en la subconsulta. Presentar una versión con MAX/MIN, otra con ALL y otra con NOT EXISTS

6. Una consulta con subconsulta correlacionada en la SELECT. La subconsulta calculará una función de agregación.

   1. Explica en el script el resultado de la misma deshaciendo la correlación para el caso de una fila que salga en el resultado.
   2. Haz una consulta equivalente que no necesite subconsulta. Pista: utiliza GROUP BY.

7. Haz una consulta que tenga una intersección y una resta. Rehazla sustituyendo ambas operaciones de conjunto con subconsultas EXISTS/NOT EXISTS.

   Nota: Antes de hacerlo, estúdiate el tema del álgebra a fondo. No hagas intersecciones para consultas que se puedan hacer con WHERE+AND ni restas para consultas que se puedan hacer con WHERE+AND NOT.

   Mírate las transparencias "de las operaciones de conjunto" en Vídeos sesión práctica 11 > Transparencias del Vídeo (Consultas tipo examen o trabajo), así como el correspondiente vídeo

   La resta habrá de hacerse con EXCEPT (ni con NOT IN, ni utilizando un join externo)

8. Una consulta con subconsulta correlacionada en el HAVING. Explica el resultado de la misma deshaciendo la correlación para una fila (un grupo) que salga en el resultado, y para otra que no salga.

Notas:

1. **Todas las consultas tendrán al menos un JOIN que sea necesario** (Cuidado: Si una tabla es innecesaria para resolver una consulta, resta puntos)
2. **Al menos habrá de haber 2 joins externos necesarios** (Cuidado: Si un JOIN externo no es necesario, también quita puntos)

ENTREGABLES

Entregar **un único script SQL** con:

1. DROPs y CREATEs iniciales
2. Por cada consulta
   1. Los INSERTs con los que vas a ilustrarlo.
   2. La consulta (o consultas, si se piden varias versiones)
   3. Las explicaciones en un comentario, cuando se pidan explicaciones
   4. ROLLBACK/DELETE/TRUNCATE para descartar los INSERTs de esa consulta
   5. Esta vez no hay vídeos porque en el examen oral, en la parte de SQL te preguntaré preferentemente cosas de este trabajo. Si no entregas el trabajo, o lo entregas muy mal, tendré que buscar otras preguntas.