Fundamentos de las bases de datos

Ejercicios de práctica SQL resueltos

T02.001- Obtén toda la información de los usuarios

```
select * from usuario;
```

T02.002- Lista los email y nombre y apellidos de los usuarios

```
select email,nombre,apellidos from usuario
```

T02.003- Lista los email y nombre y apellidos de los usuarios ordenados por email

```
select email,nombre,apellidos
from usuario
order by email;
```

T02.004- Lista los email y nombre y apellidos de los usuarios ordenados por apellidos y nombre

```
select email,nombre,apellidos
from usuario
order by apellidos,nombre;
```

T02.005- Lista los email y nombre y apellidos de los usuarios ordenados ascendentemente por apellidos y descendentemente por nombre

```
select email,nombre,apellidos
from usuario
order by apellidos,nombre desc;
```

T02.006- Lista los email y nombre y apellidos de los usuarios en orden descendente de apellidos y nombre

```
select email,nombre,apellidos
from usuario
order by apellidos desc, nombre desc;
```

T02.007- DNI,email,nombre y apellidos de los usuarios de la provincia de Asturias (código 33).

```
select dni,email,nombre,apellidos
```

```
from usuario
where provincia='33';
```

Solución alternativa:

```
select dni,email,u.nombre,apellidos
from usuario u, provincia pv
where u.provincia=codp and pv.nombre='Asturias';
```

El código de las localidades y de las provincias es texto, no número.

T02.008- Toda la información (código y nombre) de las provincias de las que se tienen usuarios

```
select pv.*
from usuario u, provincia pv
where u.provincia=codp;
```

T02.009- Toda la información (código y nombre) de las provincias de las que se tienen usuarios, eliminando duplicados y ordenando por nombre

```
select distinct pv.*
from usuario u, provincia pv
where u.provincia=codp
order by pv.nombre;
```

Solución alternativa:

```
select distinct pv.*
from usuario u, provincia pv
where u.provincia=codp
order by 2;
```

T02.010- Email de los usuarios de la provincia de Murcia que no tienen teléfono, acompañado en la salida por un mensaje que diga "No tiene teléfono"

```
select email,'No tiene teléfono'
from usuario u, provincia pv
where u.provincia=codp and pv.nombre = 'Murcia'
and telefono is null;
```

T02.011- Marcas

```
select * from marca;
```

T02.012- Artículos que no tienen marca

```
select * from articulo where marca is null
```

T02.013- Código de los artículos que pertenecen a algún pack.

```
select articulo
from ptienea;
```

T02.014- Número de pack, nombre y precio del mismo.

```
select p.cod,nombre,pvp
from articulo a, pack p
where a.cod = p.cod
```

T02.015- Código, nombre y marca de los articulos que pertenecen a algún pack.

```
select articulo,nombre,marca
from articulo, ptienea
where cod = articulo
```

T02.016- Código y precio de venta de los artículos solicitados en el pedido número 1.

```
select articulo,precio
from linped
where numpedido=1
```

T02.017- Código, nombre, marca, pvp y precio de venta de los artículos solicitados en el pedido número 1.

```
select articulo,nombre,marca,pvp,precio
from linped l, articulo a
where numpedido=1
and a.cod=l.articulo
```

T02.018- Código, nombre, marca, pvp y precio de venta de los artículos solicitados en el pedido número 1 que sean televisores.

```
select articulo,nombre,marca,pvp,precio
from linped l, articulo a, tv t
where numpedido=1
and a.cod=l.articulo
and a.cod=t.cod
```

T02.019- Fecha y usuario del pedido, código, nombre, marca, pvp y precio de venta de los artículos solicitados en el pedido número 1 que sean televisores.

```
select fecha,usuario,articulo,nombre,marca,pvp,precio
```

```
from linped l, articulo a, tv t, pedido p
where l.numpedido=1 and l.numpedido = p.numpedido
and a.cod=l.articulo
and a.cod=t.cod
```

T02.021- Código,nombre y precio de venta al público de los artículos de menos de 100€; la salida ha de ser código, nombre, "tiene el precio de", pvp.

```
select cod,nombre,'tiene el precio de',pvp from articulo where pvp < 100
```

T02.022- Código, sensor y pantalla de las cámaras, si es que "pantalla" tiene valor, ordenado por código descendentemente;

```
select cod,sensor,pantalla
from camara
where pantalla is not null order by cod desc;
```

T02.023- Panel de los televisores de 21 pulgadas o menos de pantalla, eliminando duplicados.

```
select distinct panel
from tv
where pantalla <= 21;
```

T02.024- Código, nombre, marca y precio de venta al público de los artículos que tienen ese precio entre 350 y 450.

```
select cod, nombre, marca, pvp
from articulo
where pvp >=350 and pvp <= 450;
```

T02.025- Número de pack, nombre y precio del mismo, y código, nombre y pvp de los artículos que pertenezcan a ellos.

```
select p.cod, a1.nombre, a1.pvp, a2.cod, a2.nombre, a2.pvp
from articulo a1, pack p, ptienea pp, articulo a2
where a1.cod = p.cod
and pp.pack = p.cod
and pp.articulo = a2.cod
```

T03.001- Código y nombre de los articulos con un precio entre 400 y 500 euros.

```
select cod,nombre from articulo where pvp between 400 and 500;
```

Solución alternativa:

```
select cod,nombre from articulo where pvp >= 400 and pvp <= 500;
```

T03.002- Código y nombre de los articulos con precio 415, 129, 1259 o 3995.

```
select cod,nombre from articulo
where pvp in (415, 129, 1259, 3995);
```

Solución alternativa:

```
select cod,nombre from articulo
where pvp = 415 or pvp = 129 or pvp = 1259 or pvp = 3995;
```

T03.003- Código y nombre de las provincias que no son Huelva, Sevilla, Asturias ni Barcelona.

```
select codp,nombre from provincia
where nombre not in ('huelva', 'sevilla', 'asturias', 'barcelona');
```

Solución alternativa:

```
select codp,nombre from provincia
where nombre != 'huelva' and nombre != 'sevilla'
  and nombre != 'asturias'and nombre != 'barcelona';
```

T03.004- Código de la provincia Alicante.

```
select codp from provincia
where nombre like 'Alicante%';
```

T03.005- Obtener el código, nombre y pvp de los articulos cuya marca comience por S.

```
select cod, nombre, pvp from articulo where marca like 'S%'
```

T03.006- Información sobre los usuarios cuyo email es de la eps.

```
select * from usuario where email like '%@eps.%'
```

T03.007- Código, nombre y resolución de los televisores cuya pantalla no esté entre 22 y 42.

```
select a.cod, nombre, resolucion
from articulo a, tv
where a.cod=tv.cod and pantalla not between 22 and 42;
```

T03.008- Código y nombre de los televisores cuyo panel sea tipo LED y su precio no supere los 1000 euros.

```
select t.cod, nombre from tv t, articulo a where t.cod=a.cod and panel like '%LED%' and
pvp<=1000;
```

T03.009- Email de los usuarios cuyo código postal no sea 02012, 02018 o 02032.

```
select email from usuario
where codpos not in ('02012','02018','02032');
```

T03.010- Código y nombre de los packs de los que se conoce qué articulos los componen.

```
select distinct cod, nombre
from  articulo, ptienea
where pack=cod;
```

T03.011- ¿Hay algún artículo en cesta que esté descatalogado?

```
select *
from cesta c, stock s
where c.articulo=s.articulo
  and entrega='descatalogado';
```

T03.012- Código, nombre y pvp de las cámaras de tipo compacta.

```
select a.cod, nombre, pvp
from articulo a, camara c
where a.cod=c.cod and tipo like'%compacta%';
```

T03.013- Código, nombre y diferencia entre pvp y precio de los articulos que hayan sido solicitados en algún pedido a un precio distinto de su precio de venta.

```
select cod, nombre, pvp-precio
from articulo, linped
where cod=articulo and pvp<>precio;
```

T03.014- Número de pedido,fecha y nombre y apellidos del usuario que solicita el pedido, para aquellos pedidos solicitados por algún usuario de apellido MARTINEZ.

```
select numpedido, fecha, nombre, apellidos
from pedido, usuario
where usuario=email and apellidos like'%MARTINEZ%'
```

T03.015- Código, nombre y marca del artículo más caro.

```
select cod, nombre, marca, pvp
from articulo
where pvp = (select max(pvp) from articulo);
```

Solución alternativa:

```
select cod, nombre, marca, pvp
from articulo
where pvp >= all (select pvp from articulo);
```

La solución alternativa no funciona por un "bug" de la versión instalada del motor MySQL.

T03.016- Nombre, marca y resolucion de las cámaras que nunca se han solicitado.

```
select nombre, marca, resolucion
from articulo a, camara c
where a.cod=c.cod and
c.cod not in (select articulo from linped);
```

T03.017- Código, nombre, tipo y marca de las cámaras de marca Nikon, LG o Sigma.

```
select a.cod, nombre, tipo, marca
from articulo a, camara c
where a.cod=c.cod and marca in ('NIKON','LG','SIGMA');
```

T03.018- Código, nombre y pvp de la cámara más cara de entre las de tipo réflex.

```
select a.cod, nombre, pvp
from articulo a, camara c
where a.cod=c.cod
and tipo like '%reflex%'
and pvp=(
    select max(pvp)
    from articulo a, camara c
    where a.cod=c.cod and tipo like '%reflex%');
```

Solución alternativa:

```
select a.cod, nombre, pvp
from articulo a, camara c
where a.cod=c.cod and tipo like'%réflex%'
and pvp >= all (select pvp from articulo a, camara c
                where a.cod=c.cod and tipo like'%réflex%');
```

T03.019- Marcas de las que no existe ningún televisor en nuestra base de datos.

```
select marca from marca
where marca not in (select marca
                    from articulo a, tv t
                    where a.cod=t.cod);
```

Solución alternativa:

```
SELECT marca
FROM marca m
WHERE NOT EXISTS
  (SELECT 1 FROM articulo a, tv t
   where a.cod=t.cod and a.marca=m.marca);
```

T03.020- Código, nombre y disponibilidad de los artículos con menor disponibilidad de entre los que pueden estar disponibles en 24 horas.

```
select cod,nombre,disponible
from stock, articulo
where cod=articulo
  and entrega='24 horas'
  and disponible=(select min(disponible)
                  from stock
                  where entrega='24 horas');
```

Solución alternativa:

```
select cod, nombre, disponible
from stock, articulo
where cod=articulo
and entrega ='24 horas'
and disponible <=all (select disponible
                      from stock
                      where entrega='24 horas');
```

T03.021- Nombre de los artículos cuyo nombre contenga la palabra EOS.

```
select nombre from articulo where nombre like '%EOS%';
```

T03.022- Tipo y focal de los objetivos que se monten en una cámara Canon sea cual sea el modelo.

```
select tipo, focaL from objetivo where montura like 'Canon%';
```

T03.023- Nombre de los artículos cuyo precio sea mayor de 100 pero menor o igual que 200.

```
select nombre from articulo where pvp>100 and pvp<=200;
```

T03.024- Nombre de los artículos cuyo precio sea mayor o igual que 100 pero menor o igual que 300.

```
select nombre from articulo where pvp between 100 and 300;
```

T03.025- Nombre de las cámaras cuya marca no comience por la letra S.

```
select nombre from articulo a, camara c where a.cod=c.cod and marca not like 'S%';;
```

T03.026- Dirección de correo de los usuarios cuyo dni termine en B, L o P.

```
select email from usuario where dni like '%B' or dni like '%L' or dni like '%P';
```

T03.027- Código de los televisores que tengan un panel LCD o LED.

```
select cod from tv where panel like '%LCD%' or panel like '%LED%';
```

T03.028- Número de pedido y artículo con la línea de pedido de menor precio.

```
select numpedido, articulo
from linped
where precio = (select min(precio) from linped);
```

T03.029- Nombre de los televisores que tengan una pantalla mayor que el televisor de código A0686.

```
select nombre
from articulo a, tv t
where a.cod=t.cod
  and pantalla > (select pantalla from tv where cod ='A0686');
```

T03.030- Líneas de pedido y número de pedido al que correspondan dichas líneas, y que incluyan más cantidad de artículos que las demás.

```
select linea, numpedido
from linped
where cantidad = (select max(cantidad) from linped);
```

Solución alternativa:

```
select linea, numpedido
from linped
where cantidad >= all (select cantidad from linped);
```

T03.031- Líneas de pedido y nombre de los artículos que aparecen en esas líneas, si el precio de esas líneas no es el menor de todas las líneas conocidas.

```
select distinct linea, nombre
from articulo a, linped l
where a.cod=l.articulo
   and precio > any (select precio from linped);
```

Solución alternativa:

```
select distinct linea, nombre
from articulo a, linped l
where a.cod=l.articulo
   and precio <> (select min(precio) from linped);
```

T03.032- Nombre, precio y marca de los artículos con mayor disponibilidad de stock.

```
select nombre, pvp, marca
from articulo a, stock s
where a.cod=s.articulo
   and s.disponible = (select max(disponible) from stock);
```

Solución alternativa:

```
select nombre, pvp, marca
from articulo a, stock s
where a.cod=s.articulo
  and s.disponible >=all (select disponible from stock);
```

T03.033- Nombre, precio y marca de los artículos que no tengan la mayor disponibilidad de stock.

```
select nombre, pvp, marca
from articulo a, stock s
where a.cod=s.articulo
   and s.disponible <> (select max(disponible) from stock);
```

Solución alternativa:

```
select nombre, pvp, marca
from articulo a, stock s
where a.cod=s.articulo
   and s.disponible < any (select disponible from stock);
```

T03.034- Nombre de las provincias en las que viven usuarios que hayan realizado algún pedido, eliminando duplicados.

```
select distinct p.nombre
from usuario u, provincia p, pedido pe
where u.provincia=p.codp and u.email=pe.usuario;
```

T03.035- Nombre de los artículos que hayan sido seleccionados en alguna cesta con fecha entre 01.11.2010 y 31.12.2010

```
select distinct nombre
```

```
from articulo a, cesta c
where a.cod=c.articulo
   and c.fecha between '2010-11-01' and '2010-12-31';
```

T03.036- Nombre de los artículos que hayan sido seleccionados en alguna cesta por usuarios de las provincias de Valencia o Alicante.

```
select distinct a.nombre
from articulo a, cesta c, usuario u
where a.cod=c.articulo
  and c.usuario=u.email
  and u.provincia in
     (select codp
      from provincia
      where nombre like 'Alicante%' or nombre like 'Valencia%');
```

T03.037- Número de los pedidos en los que se han incluido artículos a un precio menor que su pvp.

```
select distinct numpedido
from linped l, articulo a
where l.articulo=a.cod and l.precio < a.pvp;
```

T04.001- Toda la información de los pedidos anteriores a octubre de 2010.

```
select * from pedido where fecha < '2010-10-01';
```

T04.002- Toda la información de los pedidos posteriores a agosto de 2010.

```
select * from pedido where fecha > '2010-08-31';
```

T04.003- Toda la información de los pedidos realizados entre agosto y octubre de 2010.

```
select * from pedido where fecha > '2010-07-31' and fecha < '2010-11-01';
```

Solución alternativa:

```
select * from pedido where fecha between '2010-08-01' and '2010-10-30';
```

T04.004- Toda la información de los pedidos realizados el 3 de marzo o el 27 de octubre de 2010.

```
select * from pedido where fecha = '2010-03-03' or fecha = '2010-10-27';
```

T04.005- Toda la información de los pedidos realizados el 3 de marzo o el 27 de octubre de 2010, y que han sido realizados por usuarios del dominio "cazurren"

```
select * from pedido
where (fecha = '2010-03-03' or fecha = '2010-10-27')
       and usuario like '%@cazurren.%';
```

T04.006- ¿En qué día y hora vivimos?

```
select now();
```

T04.007- 21 de febrero de 2011 en formato dd/mm/aaaa

```
select date_format('2011-02-21','%d/%m/%Y') lafecha;
```

T04.008- 31 de febrero de 2011 en formato dd/mm/aaaa

```
select date_format('2011-02-31','%d/%m/%Y') lafecha;
```

T04.009- Pedidos realizados el 13.9.2010 (este formato, obligatorio en la comparación).

```
select * from pedido
```

```
where fecha = str_to_date('13.9.2010','%d.%c.%Y');
```

¿Por qué no obtiene el mismo resultado la orden siguiente?

*select * from pedido*

where '13.9.2010' = date_format(fecha,'%d.%m.%Y');

Pista: ejecuta "select date_format('2010-09-13','%d.%m.%Y'), date_format('2010-09-13','%e.%c.%Y'), str_to_date('13.9.2010','%d.%m.%Y')"

T04.010- Numero y fecha de los pedidos realizados el 13.9.2010 (este formato, obligatorio tanto en la comparación como en la salida).

```
select numpedido, date_format(fecha,'%d.%c.%Y') fecha
from pedido
where fecha = str_to_date('13.9.2010','%d.%c.%Y');
```

El formato '%c' es para los meses con uno o dos dígitos (no es lo mismo '09' que '9') aunque MySQL es bastante flexible y permite ciertas "licencias": str_to_date('13.9.2010','%d.%m.%Y') también funcionaría.

Igualmente, podríamos haber usado el formato '%e' para los días con uno o dos dígitos '1.3.2012'

T04.011- Numero, fecha, y email de cliente de los pedidos (formato dd.mm.aa) ordenado descendentemente por fecha y ascendentemente por cliente.

```
select numpedido, date_format(fecha,'%d.%m.%y') lafecha, usuario
from pedido
order by fecha desc, usuario;
```

¿Por qué parece que no funciona correctamente la ordenación si cambiamos el nombre de la columna de salidad de la fecha (hemos cambiado la etiqueta de la columna de la fecha)?

select numpedido, date_format(fecha,'%d.%m.%y') fecha, usuario

from pedido

order by fecha desc, usuario;

T04.012- Códigos de articulos solicitados en 2010, eliminando duplicados y ordenado ascendentemente.

```
select distinct articulo
from linped l, pedido p
where l.numpedido=p.numpedido
  and year(fecha)=2010
order by articulo;
```

Solución alternativa:

```
select distinct articulo
from linped l, pedido p
where l.numpedido=p.numpedido
  and fecha between '2010-01-01' and '2010-12-31'
```

```
order by articulo;
```

T04.013- Códigos de articulos solicitados en pedidos de marzo de 2010, eliminando duplicados y ordenado ascendentemente.

```
select articulo
from linped l, pedido p
where l.numpedido=p.numpedido
  and month(fecha)=3 and year(fecha)=2010
order by articulo;
```

Solución alternativa:

```
select articulo
from linped l, pedido p
where l.numpedido=p.numpedido
  and fecha between '2010-03-01' and '2010-03-31'
order by articulo;
```

T04.014- Códigos de articulos solicitados en pedidos de septiembre de 2010, y semana del año (la semana comienza en lunes) y año del pedido, ordenado por semana.

```
select articulo, date_format(fecha,'%u') semana, year(fecha) año
from linped l, pedido p
where l.numpedido=p.numpedido
  and month(fecha)=9 and year(fecha)=2010
order by semana;
```

T04.015- Nombre, apellidos y edad (aproximada) de los usuarios del dominio "dlsi.ua.es", ordenado descendentemente por edad.

```
select nombre, apellidos, year(now())-year(nacido) edad
from usuario
where email like '%@dlsi.ua.es'
order by edad desc;
```

Para para obtener la edad exacta: http://dev.mysql.com/doc/refman/5.0/es/date-calculations.html

O también:

select email, year(now())-year(nacido) + IF(date_format(now(),'%m-%d') > date_format(nacido,'%m-%d'), 0, -1) edad FROM usuario...

T04.016- Email y cantidad de días que han pasado desde los pedidos realizados por cada usuario hasta la fecha de cada cesta que también sea suya. Eliminad duplicados.

```
select distinct c.usuario, datediff(c.fecha,p.fecha) dias
from cesta c, pedido p
where c.usuario=p.usuario;
```

T04.017- Información sobre los usuarios menores de 25 años.

```
select * from usuario
where year(now())-year(nacido)<25;
```

T04.018- Número de pedido, usuario y fecha (dd/mm/aaaa) al que se le solicitó para los pedidos que se realizaron durante la semana del 7 de noviembre de 2010.

```
select numpedido, usuario, date_format(fecha,'%d/%m/%Y') cuando
from pedido
where date_format(fecha,'%u')=date_format('2010-11-07','%u');
```

Solución alternativa:

```
select numpedido, usuario, date_format(fecha,'%d/%m/%Y') cuando
from pedido
where date_format(fecha,'%u')=date_format(str_to_date('10/11/07','%y/%m/%d'),'%u');
```

La solución alternativa es un ejemplo de tratamiento más complejo de una fecha: se supone que nos llega en ese formato (la del where) y tenemos que transformarla a algo que MySQL entienda.

Consejo: consultad sobre las diferencias entre '%u', '%U' para date_format y la función weekofyear().

T04.019- Código, nombre, panel y pantalla de los televisores que no se hayan solicitado ni en lo que va de año, ni en los últimos seis meses del año pasado.

```
select a.cod, nombre, panel, pantalla
from articulo a, tv
where a.cod=tv.cod
   and a.cod not in (select articulo
                     from linped l, pedido p
                     where l.numpedido=p.numpedido
                       and (
                            (year(fecha)=year(now())-1
                             and month(fecha) between 7 and 12)
                           ) or year(fecha)=year(now())
                    );
```

Este ejercicio es demasiado lioso incluso para nuestra "dificultad alta", pero lo dejamos como "curiosidad".

T04.020- Email y cantidad de días que han pasado desde los pedidos realizados por cada usuario hasta la fecha de cada artículo que ahora mismo hay en su cesta. Eliminad duplicados.

```
select distinct c.usuario, datediff(c.fecha,p.fecha) dias
from cesta c, pedido p, linped l
where c.usuario=p.usuario
and l.numpedido=p.numpedido
and l.articulo = c.articulo;
```

T05.001- Número de pedido e identificador, apellidos y nombre del usuario que realiza el pedido (usando join).

```
select numpedido,usuario,apellidos,nombre
from pedido join usuario on (usuario=email)
order by apellidos,nombre;
```

Solución alternativa:

```
select numpedido,usuario,apellidos,nombre
from pedido, usuario
where usuario=email
order by 3,4;
```

T05.002- Número de pedido e identificador, apellidos y nombre del usuario que realiza el pedido, y nombre de la localidad del usuario (usando join).

```
select numpedido,usuario,apellidos,u.nombre,l.pueblo
from pedido
join usuario u on (usuario=email)
join localidad l on (l.codm=u.pueblo and l.provincia=u.provincia)
order by apellidos,u.nombre;
```

T05.003- Número de pedido e identificador, apellidos y nombre del usuario que realiza el pedido, nombre de la localidad y nombre de la provincia del usuario (usando join).

```
select numpedido,usuario,apellidos,u.nombre,l.pueblo,p.nombre
from pedido join usuario u on (usuario=email)
join localidad l on (l.codm=u.pueblo and l.provincia=u.provincia)
join provincia p on (l.provincia=codp)
order by apellidos,u.nombre;
```

T05.004- Nombre de provincia y nombre de localidad ordenados por provincia y localidad (usando join) de las provincias de Aragón y de localidades cuyo nombre comience por "B".

```
select nombre,pueblo from provincia join localidad on (provincia=codp)
where nombre in ('huesca','zaragoza','teruel')
  and pueblo like 'B%'  order by nombre,pueblo;
```

T05.005- Apellidos y nombre de los usuarios y, si tienen, pedido que han realizado.

```
select apellidos,nombre,numpedido
from usuario left join pedido on (email=usuario);
```

T05.006- Código y nombre de los artículos, si además es una cámara, mostrar también la resolución y el sensor.

```
select a.cod, nombre, resolucion, sensor
from articulo a
```

```
left join camara c on (a.cod = c.cod);
```

T05.007- Código, nombre y precio de venta al público de los artículos, si además se trata de un objetivo mostrar todos sus datos.

```
select a.cod, nombre, pvp, o.*
from articulo a
left join objetivo o on (a.cod = o.cod);
```

T05.008- Muestra las cestas del año 2010 junto con el nombre del artículo al que referencia y su precio de venta al público.

```
select c.*, nombre, pvp
from cesta c
join articulo on (cod=articulo)
where year(fecha) = 2010;
```

Solución alternativa:

```
select c.*, nombre, pvp
from cesta c
join articulo on (cod=articulo and year(fecha) = 2010);
```

T05.009- Muestra toda la información de los artículos. Si alguno aparece en una cesta del año 2010 muestra esta información.

```
select a.*, c.*
from articulo a
left join cesta c on (articulo=cod and year(fecha) = 2010);
```

A diferencia del ejercicio anterior, es obligatorio incluir la condición de la fecha en la condición del join puesto que queremos la información de todos los artículos y solo las cestas de 2010 en las que estén aquellos. Prueba lo siguiente:

select a., c.**
from articulo a
left join cesta c on (articulo=cod)
where year(fecha) = 2010;

T05.010- Disponibilidad en el stock de cada cámara junto con la resolución de todas las cámaras.

```
select s.*, resolucion
from stock s
right join camara on (cod=articulo);
```

T05.011- Código y nombre de los artículos que no tienen marca.

```
select cod, nombre
from articulo
where marca is null;
```

T05.012- Código, nombre y marca de todos los artículos, tengan o no marca.

```
select cod, nombre, marca
from articulo;
```

T05.013- Código, nombre, marca y empresa responsable de la misma de todos los artículos. Si algún artículo no tiene marca debe aparecer en el listado con esta información vacía.

```
select cod, nombre, a.marca, empresa
from articulo a
left join marca m on (a.marca = m.marca);
```

T05.014- Información de todos los usuarios de la comunidad valenciana cuyo nombre empiece por 'P' incluyendo la dirección de envío en caso de que la tenga.

```
select u.*, d.*
from usuario u
left join direnvio d on (d.email=u.email)
where u.provincia in ('46','03','12')
  and u.nombre like 'P%';
```

Solución alternativa:

```
select u.*, d.*
from usuario u
join provincia p on (u.provincia = codp)
left join direnvio d on (d.email=u.email)
where (p.nombre like '%Alicante%'
    or p.nombre like '%Valencia%'
    or p.nombre like '%Castell%')
and u.nombre like 'P%';
```

La forma más sencilla necesita conocer los códigos de provincia y, además, tener en cuenta que estos códigos son cadenas de caracteres.

La forma alternativa utiliza el like para evitar problemas de comparación.

T05.015- Código y nombre de los artículos, y código de pack en el caso de que pertenezca a alguno.

```
select cod, nombre, pack
from articulo
left join ptienea on (cod=articulo);
```

T05.016- Usuarios y pedidos que han realizado.

```
select u.*, p.*
from usuario u, pedido p
where u.email=p.usuario;
```

Solución alternativa:

```
select *
from usuario u
join pedido p on (u.email=p.usuario);
```

Nótese que no se ha dicho "...y, si tienen, pedidos que..."

T05.017- Información de aquellos usuarios de la comunidad valenciana (códigos 03, 12 y 46) cuyo nombre empiece por 'P' que tienen dirección de envío pero mostrando, a la derecha, todas las direcciones de envío de la base de datos.

```
select u.*, d.*
from usuario u
right join direnvio d
   on (d.email=u.email
   and u.provincia in ('46','03','12')
   and u.nombre like 'P%');
```

Nótese, respecto de otro ejercicio similar, que las condiciones del usuario se han introducido dentro de la condición del right join.

T06.001- Crea una tabla de nombre XX con 2 columnas, col1 de tipo integer, y col2 de tipo char(3), con col1 como clave primaria.

```
create table XX (col1 integer, col2 char(3), primary key (col1))
```

T06.002- Consulta la tabla

```
select * from xx
```

Obviamente, la tabla está vacía y muestra 0 filas.

T06.003- Inserta en la tabla la fila (1,'AA')

```
insert into xx values (1,'AA')
```

Ningún problema, fila insertada.

T06.004- inserta en la tabla la fila ('BB',2)

```
insert into xx values ('BB',2)
```

Esta orden falla porque los valores no están en el orden adecuado. En este caso deberíamos escribir:

insert into xx (col2,col1) values ('BB',2)

NOTA: en nuestro servidor MySQL, el sistema lanza un warning pero ejecuta la sentencia realizando una conversión de tipos de varchar a int, pero como 'BB' no sabe interpretarlo, lo almacena como 0.

T06.005- Inserta en la tabla la fila (2,'BB')

```
insert into xx values (2,'BB')
```

Hemos cambiado el orden de los valores y coinciden con la estructura de la tabla.

T06.006- Consulta la tabla XX

```
select * from xx
```

Deberías ver 2 filas.

T06.007- Cierra la sesión e identifícate de nuevo ("salte y vuelve a entrar" o "desconecta" y "conecta")

A continuación consulta de nuevo XX

En realidad, si lo tienes claro, no hace falta que lo hagas. Lo que se pretende es comprobar la persistencia de la tabla creada y los datos que pueda contener: el cierre de sesión no borra lo almacenado. Sólo drop table puede eliminar la tabla del catálogo.

```
select * from xx
```

Al volver a entrar, identificarte correctamente y seleccionar tu base de datos, la consulta devuelve otra vez las mismas 2 filas, la información se ha mantenido intacta.

T06.008- Borra la tabla XX

```
drop table xx
```

*Borrar una tabla elimina todo del sistema, sus filas y la propia tabla. Si intentas ahora select * from xx obtendrás un mensaje de error.*

T06.009- Crea una tabla YY con 3 columnas

col1(integer),
col2(char(2)) y
col3(varchar(10)),

y con clave primaria (col1, col2)

```
create table yy (
col1 integer,
col2 char(2),
col3 varchar(10),
primary key (col1, col2))
```

Ahora la clave primaria está compuesta por 2 columnas.

T06.010- Inserta los siguientes datos y consulta la tabla para ver los datos almacenados

(1,'AA','primera')
(2,'AA','segunda')
(2,'BB','tercera')
(1,'AA','cuarta')
(NULL,NULL,'quinta')
(NULL,'CC','sexta')
(3,NULL,'séptima')
(0,'','octava') --0, cadena vacía, 'octava'
(3,'AA',NULL)

```
insert into yy values (1,'AA','primera');
insert into yy values (2,'AA','segunda');
insert into yy values (2,'BB','tercera');
insert into yy values (1,'AA','cuarta');
insert into yy values (NULL,NULL,'quinta');
insert into yy values (NULL,'CC','sexta');
insert into yy values (3,NULL,'séptima');
insert into yy values (0,'','octava');
insert into yy values (3,'AA',NULL);
```

** insert into yy values (1,'AA','cuarta') falla por que estamos introduciendo un duplicado de clave primaria.*

** insert into yy values (NULL,NULL,'quinta') falla por que la clave primaria no admite nulos.*

** insert into yy values (NULL,'CC','sexta') falla por que tampoco los admite parcialmente.*

** insert into yy values (3,NULL,'séptima') falla por la misma razón.*

** insert into yy values (0,'','octava') NO falla porque cadena vacía es un valor, NO es NULL.*

T06.011- Ejecuta lo siguiente:

```
create table T1(a int,b int,c int,
primary key(a)) engine=innodb;
create table T2(a int,d int,e int,
primary key(d),foreign key(a) references T1(a)) engine=innodb;
```

y comprueba, buscando el porqué en caso de fallo, el resultado de cada una de las órdenes de la siguiente secuencia:

a) insertar en T1(1,10,100)
b) insertar en T1(NULO,20,NULO)
c) insertar en T1(2,20,NULO)
d) insertar en T1(3,NULO,300)
e) insertar en T2(2,NULO,NULO)
f) insertar en T2(2,20,NULO)
g) insertar en T1(1,20,200)
h) insertar en T2(4,10,100)
i) insertar en T2(2,30,230)

```
insert into T1 values (1,10,100); -- a
insert into T1 values (NULL,20,NULL); -- b falla
insert into T1 values (2,20,NULL); -- c
```

```
insert into T1 values (3,NULL,300); -- d
insert into T2 values (2,NULL,NULL); -- e falla
insert into T2 values (2,20,NULL); -- f
insert into T1 values (1,20,200); -- g falla
insert into T2 values (4,10,100); -- h falla
insert into T2 values (2,30,230);-- i
```

T06.012- Continúa el anterior

j) modificar T1(1,10,100) a (2,10,100)
k) modificar T1(1,10,100) a (5,10,100)
l) modificar T2(2,20,NULO) a (2,20,220)
m) modificar T2(2,20,220) a (5,20,220)
n) modificar T2(5,20,220) a (2,10,100)
o) modificar T1(2,20,200) a (6,60,600)
p) modificar T1(3,NULO,300) a (7,70,700)
q) modificar T2(2,10,100) a (7,10,100)
r) modificar T2(2,30,230) a (7,30,230)
s) modificar T1(2,20,NULO) a (6,60,600)

```
update T1 set a=2 where a=1; -- j falla
update T1 set a=5 where a=1; -- k
update T2 set e=220 where d=20; -- l
update T2 set a=5 where d=20; -- m
update T2 set a=2,d=10,e=100 where d=20; -- n
update T1 set a=6,b=60,c=600 where a=2; -- o falla
update T1 set a=7,b=70,c=700 where a=3; -- p
update T2 set a=7 where d=10; -- q
update T2 set a=7 where d=30; -- r
update T1 set a=6,b=60,c=600 where a=2; -- s
```

T06.013- Continúa el anterior

t) borrar T2(7,30,230)
u) borrar T1(7,70,700)
v) borrar T1(5,10,100)
w) borrar T2(7,10,100)
x) borrar T1(7,70,700)
y) borrar T1(6,60,600)

```
delete from T2 where d=30; -- t
delete from T1 where a=7; -- u falla
delete from T1 where a=5; -- v
delete from T2 where d=10; -- w
delete from T1 where a=7; -- x ahora sí
delete from T1 where a=6; -- y
```

El resultado final debe ser las 2 tablas vacías de filas.

T07.001- Crea las siguientes tablas:

TA (a int, b int) CP(a)

TB (c int, d int) CP(c) CAj(d) >> TA (borrados: propagar, modificaciones: propagar)

TC (e int, f int) CP(e) CAj(f) >> TB (borrados: propagar, modificaciones: propagar)

```
create table TA(a int, b int, primary key(a)) engine=innodb;

create table TB(c int, d int, primary key(c),
  foreign key (d) references TA(a) on delete cascade on update cascade)
  engine=innodb;

create table TC(e int, f int, primary key(e),
  foreign key (f) references TB(c) on delete cascade on update cascade)
  engine=innodb;
```

T07.002- Inserta los siguientes datos

TA(1,10)
TA(2,20)
TA(3,30)
TB(100,1)
TB(200,1)
TB(300,2)
TB(400,NULL)
TC(1000,100)
TC(2000,100)
TC(3000,NULL)

```
insert into TA values (1,10);
insert into TA values (2,20);
insert into TA values (3,30);
insert into TB values (100,1);
insert into TB values (200,1);
insert into TB values (300,2);
insert into TB values (400,NULL);
insert into TC values (1000,100);
insert into TC values (2000,100);
insert into TC values (3000,NULL);
```

T07.003- Borra TA(2,20) y comprueba los cambios que se han producido en las 3 tablas

```
delete from TA where a = 2;
```

T07.004- Modifica TA(1,10) a TA(15,10) y comprueba los cambios que se han producido en las 3 tablas.

```
update TA set a=15 where a = 1;
```

T07.005- Borra TC(2000,100) y comprueba los cambios que se han producido en las 3 tablas.

```
delete from TC where e=2000;
```

T07.006- Borra TA(3,30) y comprueba los cambios que se han producido en las 3 tablas.

```
delete from TA where a=3;
```

T07.007- Borra TB(100,15) y comprueba los cambios que se han producido en las 3 tablas.

```
delete from TB where c=100;
```

T07.008- Borra TC(3000,NULL) y comprueba los cambios que se han producido en las 3 tablas.

```
delete from TC where e=3000;
```

T07.009- Borra TB(400,NULL) y comprueba los cambios que se han producido en las 3 tablas.

```
delete from TB where c=400;
```

T07.010- Borra TA(15,10) y comprueba los cambios que se han producido en las 3 tablas: ¿ESTÁN LAS 3 TABLAS VACÍAS?

```
delete from TA where a=15;
```

T07.011- Vuelve a crear las tablas:

TA (a int, b int) CP(a)
TB (c int, d int) CP(c) CAj(d) >> TA
(borrados: anular, modificaciones: anular)
TC (e int, f int) CP(e) CAj(f) >> TB
(borrados: anular, modificaciones: anular)

```
drop table TC;
drop table TB;
drop table TA;

create table TA(a int, b int, primary key(a)) engine=innodb;

create table TB(c int, d int, primary key(c),
  foreign key (d) references TA(a)
    on delete set null on update set null)
```

```
  engine=innodb;

create table TC(e int, f int, primary key(e),
  foreign key (f) references TB(c)
    on delete set null on update set null)
  engine=innodb;
```

T07.012- Vuelve a rellenar las tablas:
TA(1,10)
TA(2,20)
TA(3,30)
TB(100,1)
TB(200,1)
TB(300,2)
TB(400,NULL)
TC(1000,100)
TC(2000,100)
TC(3000,NULL)

```
insert into TA values (1,10);
insert into TA values (2,20);
insert into TA values (3,30);
insert into TB values (100,1);
insert into TB values (200,1);
insert into TB values (300,2);
insert into TB values (400,NULL);
insert into TC values (1000,100);
insert into TC values (2000,100);
insert into TC values (3000,NULL);
```

T07.013- Ejecuta las siguientes órdenes:

Borra TA(2,20)
Modifica TA(1,10) a TA(15,10)
Modifica TB(100,NULL) a TB(150,NULL)

¿Queda algún valor de clave ajena distinto de NULL?

```
delete from TA where a = 2;
update TA set a=15 where a = 1;
update TB set c=150 where c = 100;
```

T07.014- Vuelve a crear las tablas:

TA (a int, b int) CP(a)
TB (c int, d int) CP(c) CAj(d) >> TA (borrados: propagar)
TC (e int, f int) CP(e) CAj(f) >> TB (modificaciones: anular)

TA(1,10)
TA(2,20)
TA(3,30)
TB(100,1)
TB(200,1)
TB(300,2)
TB(400,NULL)
TC(1000,100)
TC(2000,100)
TC(3000,NULL)

```
drop table TC;
drop table TB;
drop table TA;

create table TA(a int, b int, primary key(a)) engine=innodb;

create table TB(c int, d int, primary key(c),
  foreign key (d) references TA(a)
    on delete cascade)
  engine=innodb;

create table TC(e int, f int, primary key(e),
  foreign key (f) references TB(c)
    on update set null)
  engine=innodb;
insert into TA values (1,10);
insert into TA values (2,20);
insert into TA values (3,30);
insert into TB values (100,1);
insert into TB values (200,1);
insert into TB values (300,2);
insert into TB values (400,NULL);
insert into TC values (1000,100);
insert into TC values (2000,100);
insert into TC values (3000,NULL);
```

T07.015- Borra TA(1,10): ¿qué ha pasado?

```
delete from TA where a=1;
```

T07.016- Borra TA(2,20): ¿qué ha pasado?

```
delete from TA where a=2;
```

T07.017- Modifica TB(100,1) a TB(170,1): ¿qué ha pasado?

```
update TB set c=170 where c=100;
```

T07.018- Vuelve a intentar borrar TA(1,10): ¿por qué ahora sí?

```
delete from TA where a=1;
```

T08.001- Obtener el precio total por línea para el pedido 1, en la salida aparecerá los campos numlinea, articulo y el campo calculado total.

```
select linea, articulo, (precio*cantidad) total
from linped
where numPedido=1;
```

T08.002- Obtener la cantidad de provincias distintas de las que tenemos conocimiento de algún usuario.

```
select count(distinct provincia) provincias from usuario;
```

T08.003- Cantidad de usuarios de nuestra BD.

```
select count(*) usuarios from usuario;
```

T08.004- Número de articulos con precio de venta mayor de 200 euros.

```
select count(*) from articulo where pvp>200;
```

T08.005- Total en euros de la cesta del usuario "bmm@agwab.com".

```
select sum(pvp) total
from cesta, articulo
where usuario='bmm@agwab.com' and articulo=cod;
```

T08.006- Tamaño máximo de pantalla para las televisiones.

```
select MAX(pantalla) maxPantalla from tv;
```

T08.007- Media de precios de venta al público distintos de los articulos, redondeada a dos decimales.

```
select round(avg(distinct pvp),2) mediaventa from articulo;
```

T08.008- Nombre y precio de los articulos con el mínimo stock disponible.

```
select nombre,pvp
from articulo,stock
where cod=articulo
  and disponible = (select min(disponible) from stock);
```

T08.009- Número de pedido, fecha y nombre y apellidos del usuario de las lineas de pedido cuyo total en euros es el más alto.

```
select p.numPedido,fecha,nombre,apellidos
from pedido p,linped l,usuario u
where p.usuario=email
  and p.numPedido=l.numPedido
  and (cantidad*precio)=(select max(cantidad*precio) from linped);
```

T08.010- Máximo, mínimo y media de precio de venta de los artículos.

```
select MAX(pvp) maxPvp, MIN(pvp) minPvp, AVG(pvp) mediaPvp from articulo;
```

T08.011- Código, nombre, pvp y fecha de incorporación del artículo a la cesta más reciente.

```
select cod, nombre, pvp
from cesta, articulo
where articulo=cod
  and fecha=(select MAX(fecha) from cesta);
```

T08.012- Cantidad de artículos que están descatalogados.

```
select count(*) NumArtDescatalogados
from stock
where entrega='Descatalogado'
```

T08.013- Precio máximo del artículo en stock que será entregado próximamente.

```
select max(pvp)
from articulo, stock
where cod=articulo
  and entrega='Próximamente';
```

T08.014- Nombre, código y disponible en stock para todos los artículos cuyo código acabe en 3, siendo ese disponible el mínimo de toda la tabla.

```
select nombre, cod, disponible MinDisponible
from stock, articulo
where articulo=cod
  and cod like '%3'
  and disponible=(select MIN(disponible) from stock);
```

T08.015- Precio máximo, mínimo y medio de las líneas de pedido que incluyen el artículo “Bravia KDL-32EX402”

```
select MAX(precio) MaxPrecio, MIN(precio) MinPrecio, AVG(precio) PrecioMedio
```

```
from linped l, articulo a
where a.cod=l.articulo and a.nombre='Bravia KDL-32EX402';
```

T08.016- Cantidad total que se ha pedido de los artículos cuyo nombre empieza por "UE22".

```
select SUM(cantidad) cantidadTotal
from linped, articulo
where articulo=cod and nombre like 'UE22%';
```

T08.017- Precio medio de los artículos incluidos en la línea de pedido número 4, redondeado a 3 decimales.

```
select round(avg(precio),3) PrecioMedio
from linped
where linea=4;
```

T08.018- Número de pedido, nombre, teléfono y email de usuario del pedido (o los pedidos) que contiene líneas de pedido cuyo precio sea igual al precio más alto de entre todas las segundas líneas de todos los pedidos.

```
select l.numPedido, nombre, telefono, email
from usuario u, pedido p, linped l
where l.numPedido=p.numPedido
  and p.usuario=u.email
  and precio=(select MAX(precio) from linped where linea=2);
```

Primero se ha de calcular el precio máximo de entre todas las segundas líneas (la subconsulta) y, entonces, ya se pueden comparar TODAS las líneas de pedido: saldrá, seguro, el pedido de esa segunda línea que da el máximo, pero ese precio máximo se puede repetir en cualquier otra línea.

T08.019- Diferencia entre el precio máximo y el precio mínimo del pedido número 30.

```
select (MAX(precio)-MIN(precio)) DiferenciaPrecios
from linped
where numPedido=30;
```

T08.020- Código, nombre, precio de venta del artículo que más hay en stock.

```
select cod,nombre,pvp
from articulo,stock
where cod=articulo
  and disponible = (select MAX(disponible) from stock);
```

T08.021- Fecha de nacimiento del usuario más viejo.

```
select MIN(nacido) from usuario;
```

T08.022- Obtener en una única consulta, cuántas filas tiene la tabla artículo, cuántas de ellas tienen valor en la columna marca y cuántas marcas distintas hay almacenadas en la tabla.

```
select count(*) filas, count(marca) conmarca, count(distinct marca) marcas from
articulo;
```

T09.001- ¿Cuántos artículos de cada marca hay?

```
select marca,count(*)
from articulo
group by marca;
```

T09.002- ¿Cuáles son las marcas que tienen menos de 150 artículos?

```
select marca,count(*) from articulo
group by marca
having count(*)<150;
```

T09.003- ¿Cuáles son las marcas que tienen menos de 150 artículos (eliminar las marcas que sean null)?

```
select marca,count(*) from articulo
where marca is not null
group by marca
having count(*)<150
```

T09.004- Número de cámaras que tienen sensor CMOS

```
select count(*) from camara where sensor like 'CMOS%'
```

T09.005- Dni, nombre, apellidos y email de los usuarios que han realizado más de un pedido.

```
select dni,nombre,apellidos,email
from usuario U, pedido P
where U.email=P.Usuario
group by dni,nombre,apellidos,email
having count(*)>1;
```

Solución alternativa:

```
select max(dni),max(nombre),max(apellidos),email
from usuario U, pedido P
where U.email=P.Usuario
group by email
having count(*)>1;
```

T09.006- Pedidos (número de pedido y usuario) de importe mayor a 4000 euros.

```
select p.numpedido,p.usuario
from pedido p, linped l
where p.numpedido=l.numpedido
group by p.numpedido,p.usuario
having sum(cantidad*precio)>4000;
```

Cuando nos piden el importe (total) de un pedido, se entiende que es el precio unitario de cada línea de ese pedido multiplicado por la cantidad pedida de ese artículo.

T09.007- Pedidos (número de pedido y usuario) con más de 10 artículos, mostrando esta cantidad.

```
select P.numPedido,P.usuario, sum(cantidad)
from pedido P, linped L
where P.numPedido=L.numPedido
group by P.numPedido,P.usuario
having sum(cantidad)>10;
```

T09.008- Pedidos (número de pedido y usuario) que contengan más de cuatro artículos distintos.

```
select P.numPedido,P.usuario, count(distinct articulo)
from pedido P, linped L
where P.numPedido=L.numPedido
group by P.numPedido,P.usuario
having count(distinct articulo)>4;
```

T09.009- ¿Hay dos provincias que se llamen igual (con nombre repetido)?

```
select nombre,count(*) from provincia group by nombre having count(*)>1;
```

T09.010- ¿Hay algún pueblo con nombre repetido?

```
select pueblo,count(*) from localidad group by pueblo having count(*)>1;
```

T09.011- Obtener el código y nombre de las provincias que tengan más de 100 pueblos.

```
select P.codp,P.nombre,count(*)
FROM provincia P, localidad L
WHERE P.codp=L.provincia
group by P.codp,P.nombre
having count(*)>100;
```

T09.012- Ha habido un error en Tiendaonline y se han colado varios artículos sin stock en la cesta. Averigua el código de esos artículos y las veces que aparecen en la cesta.

```
select c.articulo, count(c.articulo)
from cesta c, stock s
where c.articulo=s.articulo
  and disponible=0
group by c.articulo;
```

T09.013- Clientes que hayan adquirido (pedido) más de 2 tv

```
select p.usuario, sum(cantidad)
from pedido p, linped l, tv
where p.numpedido=l.numpedido
  and l.articulo=tv.cod
group by p.usuario
having sum(cantidad)>2;
```

T09.014- ¿Cuántas veces se ha pedido cada artículo? Si hubiese artículos que no se han incluido en pedido alguno también se mostrarán. Mostrar el código y nombre del artículo junto con las veces que ha sido incluido en un pedido (solo si ha sido incluido, no se trata de la "cantidad").

```
select cod, nombre, count(numpedido)
from articulo a
left join linped l on (a.cod=l.articulo)
group by cod, nombre;
```

El "truco" consiste en contar los números de pedido asociados a algunos artículos. El left join hace que ciertos artículos aparezcan una vez junto con NULL como numpedido. El resto de artículos aparecen tantas veces como líneas tengan en LINPED. Como count(numpedido) solo cuenta los valores distintos de NULL, por eso conseguimos que la cuenta sea 0 en los artículos no pedidos nunca.

T09.015- Código y nombre de las provincias que tienen más de 50 usuarios (provincia del usuario, no de la dirección de envío).

```
select p.codp, p.nombre
from provincia p, usuario u
where p.codp=u.provincia
group by p.codp, p.nombre
having count(email)>50;
```

T09.016- Cantidad de artículos con stock 0

```
select count(*) from stock where disponible=0;
```

T09.017- Cantidad de artículos que no son ni memoria, ni tv, ni objetivo, ni cámara ni pack.

```
select  count(*)
from articulo
where cod not in (select cod from camara)
  and cod not in (select cod from tv)
  and cod not in (select cod from memoria)
  and cod not in (select cod from objetivo)
  and cod not in (select cod from pack);
```

Solución alternativa:

```
select  count(*)
from articulo
where cod not in (select cod from camara
```

```
union select cod from tv
union select cod from memoria
union select cod from objetivo
union select cod from pack);
```

T09.018- Número de artículos pedidos por provincia (provincia del usuario no de la dirección de envío). Mostrar el código de la provincia, su nombre y la cantidad de veces que se ha pedido el artículo; si la provincia no tiene asociada esta cantidad, mostrar "0" en esa columna.

```
select pr.codp, pr.nombre, IFNULL(sum(cantidad),0)
from provincia pr
  left join usuario u on (pr.codp=u.provincia)
  left join pedido p on (p.usuario=u.email)
  left join linped l on (p.numpedido=l.numpedido)
group by pr.codp,pr.nombre;
```

T11.001- Listado de los códigos de los artículos Samsung que han sido pedidos.

```
SELECT a.cod
FROM articulo a
WHERE marca = 'Samsung' AND EXISTS
        (SELECT * FROM linped l WHERE a.cod = l.articulo);
```

Solución alternativa:

```
SELECT DISTINCT a.cod
FROM articulo a, linped l
WHERE a.cod = articulo AND marca = 'Samsung';
```

T11.002- Utilizando operadores de conjuntos obtener los nombres de los artículos que sean cámaras compactas con visor electrónico o televisores CRT.

```
SELECT nombre FROM camara c
        JOIN articulo a ON (c.cod = a.cod)
        WHERE tipo LIKE '%compacta%visor%electrónico%'
UNION
SELECT nombre FROM tv t
        JOIN articulo a ON (t.cod = a.cod)
        WHERE panel LIKE '%televisor%CRT%';
```

T11.003- Utilizando operadores de conjuntos obtener el nombre de los usuarios, la localidad y la provincia de los usuarios que sean de un pueblo que contenga 'San Vicente' o que sean de la provincia de 'Valencia'.

```
SELECT u.nombre, p.nombre, l.pueblo FROM usuario u
        JOIN localidad l ON (u.pueblo = l.codm AND u.provincia = l.provincia)
        JOIN provincia p ON l.provincia = p.codp
        WHERE l.pueblo LIKE '%San Vicente%'
UNION
SELECT u.nombre, p.nombre, l.pueblo FROM usuario u
        JOIN localidad l ON (u.pueblo = l.codm AND u.provincia = l.provincia)
        JOIN provincia p ON l.provincia = p.codp
        WHERE p.nombre LIKE '%Valencia%'
```

T11.004- Nombre y email de los usuarios de Asturias que tengan la misma dirección de envió que de residencia (por defecto es la misma dirección si no se especifica una dirección de envío).

```
SELECT u.nombre, email FROM usuario u
        JOIN provincia p ON (u.provincia = codp)
        WHERE p.nombre = 'Asturias' AND email NOT IN
                (SELECT email FROM direnvio)
```

T11.005- Código, nombre y marca de los objetivos con focales de 500 o 600 mm para las marcas de las que no se ha solicitó ningún artículo en el mes de noviembre de 2010.

```
SELECT a.cod, nombre, marca
```

```
FROM objetivo o, articulo a
WHERE a.cod = o.cod AND (focal = '500 mm' OR focal = '600 mm')
        AND marca NOT IN
                (SELECT marca
                FROM pedido p, linped l, articulo
                where p.numpedido=l.numpedido and articulo= cod
                      and year(fecha)='2010' and month(fecha)='11')
```

T11.006- Código y pvp de los artículos 'Samsung' que tengan pvp y que no tengan pedidos.

```
SELECT distinct a.cod, pvp
        FROM articulo a
        WHERE marca = 'Samsung' AND pvp IS NOT NULL AND a.cod NOT IN
                (select articulo from linped);
```

T11.007- Utilizando operadores de conjuntos, muestra los nombres de los artículos que estén en un pack.

```
SELECT nombre
FROM articulo
WHERE EXISTS
        (SELECT * FROM ptienea WHERE cod = articulo);
```

T11.008- Utilizando el producto cartesiano, obtener los nombres de las localidades con 2 o más usuarios (sin usar group by).

```
SELECT distinct l.pueblo
FROM usuario u1, usuario u2, localidad l
WHERE u1.email != u2.email
  AND u1.pueblo = u2.pueblo AND u1.provincia = u2.provincia
  AND u1.pueblo=l.codm AND u1.provincia=l.provincia;
```

Solución alternativa:

```
Compáralo con la solución usando group by-having.

SELECT l.pueblo
FROM localidad l, usuario u
WHERE u.pueblo = l.codm AND u.provincia=l.provincia
GROUP BY l.codm, l.provincia, l.pueblo
HAVING COUNT(*) >= 2;
```

T11.009- Los códigos de los artículos que están en stock, en la cesta y han sido pedidos.

```
SELECT DISTINCT s.articulo
FROM stock s, cesta c, linped l
WHERE s.articulo = c.articulo AND s.articulo = l.articulo;
```

T11.010- Código y nombre de los artículos, aunque estén repetidos, que aparezcan en un pack o en una cesta.

```
SELECT p.articulo, a.nombre
FROM ptienea p
        JOIN articulo a ON (p.articulo = a.cod)
UNION ALL
SELECT c.articulo, a.nombre
FROM cesta c
        JOIN articulo a ON (c.articulo = a.cod);
```

T11.011- Códigos de artículos que están en alguna cesta o en alguna línea de pedido.

```
select  articulo from cesta
union
select  articulo from linped;
```

Solución alternativa:

```
select cod
from articulo
where cod in (select articulo from cesta)
   or cod in (select articulo from linped);

select cod
from articulo
where exists (select 1 from cesta where articulo=cod)
   or exists (select 1 from linped where articulo=cod);
```

T11.012- Email y nombre de los usuarios que no han hecho ningún pedido o que han hecho sólo uno.

```
select email, nombre
from usuario
where email not in (select usuario from pedido)
union
select email, nombre
from usuario, pedido
where email=usuario
group by email, nombre
having count(*)=1;
```

T11.013- Apellidos que se repitan en más de un usuario (sin utilizar group by).

```
select distinct u1.apellidos
from usuario u1, usuario u2
where u1.email <> u2.email and u1.apellidos= u2.apellidos;
```

Solución alternativa:

```
Compáralo con la solución usando group by-having.

select apellidos
from usuario
group by apellidos
having count(*) > 1;
```

T11.014- Parejas de nombres de provincia que tienen algún pueblo que se llama igual, junto con el nombre del pueblo.

```
select p1.nombre, p2.nombre, l1.pueblo
from provincia p1, provincia p2, localidad l1, localidad l2
where p1.codp<>p2.codp
  and p1.codp=l1.provincia
  and p2.codp=l2.provincia
  and l1.pueblo=l2.pueblo;
```

Solución alternativa:

Observad lo que tarda en realizar la consulta comparado con otras.

T11.015- Código y nombre de los artículos que en stock están "Descatalogado" o que no se han solicitado en ningún pedido.

```
select cod, nombre
from articulo
where cod in(
  select articulo from stock where entrega='Descatalogado'
  union
  select cod
  from articulo where cod not in (select articulo from linped))
```

T11.016- Email, nombre y apellidos de los usuarios que han solicitado televisores pero nunca han solicitado cámaras.

```
select email, nombre, apellidos
from usuario
where email in
  (select usuario from pedido p, linped l, tv
   where p.numpedido=l.numpedido and articulo=cod)
 and email not in
  (select usuario from pedido p, linped l, camara
   where p.numpedido=l.numpedido and articulo=cod);
```

T11.017- Usuarios que han solicitado pedidos de importe superior a 10000 (por pedido) o que han solicitado más de 5 artículos distintos entre todos sus pedidos.

```
select usuario
from linped l, pedido p where l.numpedido=p.numpedido
group by p.numpedido, usuario
having sum(cantidad*precio)>10000
union
```

```
select usuario
from linped l, pedido p where l.numpedido=p.numpedido
group by usuario
having count(distinct articulo)>5;
```

T11.018- Obtener un listado en el que figuren para todos los usuarios: su email, su nombre y sus apellidos junto con una frase en la que se muestre lo que a continuación se indica:
- para los usuarios con un importe total entre todos sus pedidos superior a 10000 mostraremos GRAN CLIENTE
- para los que el importe está entre 6000 y 10000 mostraremos CLIENTE MEDIO
- para los que el importe es inferior a 6000 mostraremos COMPRA POCO
- para los que no han hecho ningún pedido mostraremos ** NO HA COMPRADO NUNCA.

El listado se ordenará por apellidos.

```
select email, nombre, apellidos, '  GRAN CLIENTE'
from usuario, linped l, pedido p where l.numpedido=p.numpedido and p.usuario=email
group by email, nombre, apellidos
having sum(cantidad*precio)>10000
UNION
select email, nombre, apellidos, '  CLIENTE MEDIO'
from usuario, linped l, pedido p where l.numpedido=p.numpedido and p.usuario=email
group by email, nombre, apellidos
having sum(cantidad*precio) between 6000 and 10000
UNION
select email, nombre, apellidos, '  COMPRA POCO'
from usuario, linped l, pedido p where l.numpedido=p.numpedido and p.usuario=email
group by email, nombre, apellidos
having sum(cantidad*precio)<6000
UNION
select email, nombre, apellidos, '  ** NO HA COMPRADO NUNCA'
from usuario
where email not in (select usuario from pedido)
order by 3;
```

T11.019- ¿Hay alguna fila en la tabla marca?

```
select exists (select * from marca)
```

T11.020- Email y nombre de los usuarios que no han pedido ninguna cámara.

```
select email, nombre
from usuario u
where not exists
 (select 1
  from linped l, pedido p, camara c
  where email=p.usuario
    and l.numpedido=p.numpedido and c.cod=l.articulo);
```

Solución alternativa:

```
select email, nombre
```

```
from usuario u
where email not in
  (select usuario
   from linped l, pedido p, camara c
   where l.numpedido=p.numpedido and l.articulo = c.cod);
```

T11.021- Email y nombre de los usuarios que, habiendo realizado algún pedido, no han pedido ninguna cámara.

```
select email, nombre
from usuario u
where email in (select usuario from pedido)
and not exists
 (select 1
  from linped l, pedido p, camara c
  where email=p.usuario
    and l.numpedido=p.numpedido and c.cod=l.articulo);
```

Solución alternativa:

```
select email, nombre
from usuario u
where email in (select usuario from pedido)
and email not in
  (select usuario
   from linped l, pedido p, camara c
   where l.numpedido=p.numpedido and l.articulo = c.cod)
```

Obsérvese el uso de usuario.email en la subconsulta del exists.

T11.022- Código y nombre del artículo que ha sido incluído en todos los pedidos.

```
select cod,nombre
from articulo a
where not exists
  (select 1
   from pedido p
   where not exists
      (select 1
       from linped l
       where l.numpedido=p.numpedido
         and l.articulo=a.cod))
```

Solución alternativa:

```
select articulo
from linped l
group by articulo
having count(distinct numpedido) =
  (select count(*) from pedido);
```

"Artículo tal que NO existe pedido que NO lo incluya".

T11.023- Código y nombre de los artículos que han sido solicitados en todos los pedidos del usuario acm@colegas.com.

```
select cod,nombre
from articulo a
where not exists
  (select 1
   from pedido p
   where usuario='acm@colegas.com'
     and not exists
       (select 1
        from linped l
        where l.numpedido=p.numpedido
          and l.articulo=a.cod)
  )
```

Solución alternativa:

```
select articulo,nombre
from articulo a, pedido p,linped l
where usuario='acm@colegas.com'
  and p.numpedido=l.numpedido
  and l.articulo=a.cod
group by articulo,nombre
having count(distinct p.numpedido) =
            (select count(*) from pedido where usuario='acm@colegas.com');

También:

select cod,nombre
from articulo a
where cod in
   (select articulo
    from pedido p,linped l
    where usuario='acm@colegas.com' and p.numpedido=l.numpedido
    group by articulo
    having count(distinct p.numpedido) =
             (select count(*) from pedido where usuario='acm@colegas.com')
   );
```

T11.024- ¿Hay alguna fila en la tabla marca? Si la respuesta es positiva, que muestre la palabra "sí".

```
select 'sí' respuesta
from dual
where exists (select 1 from marca);
```

"Dual" es una tabla ficticia del sistema que sirve precisamente para esto, para consultas que no utilizan ninguna tabla.

T11.025- ¿Hay alguna fila en la tabla memoria? Si la respuesta es negativa, que muestre la palabra "no".

```
select 'no' respuesta
from dual
where not exists (select 1 from memoria);
```

"Dual" es una tabla ficticia del sistema que sirve precisamente para esto, para consultas que no utilizan ninguna tabla.

T11.026- Pedidos que incluyen cámaras y televisiones.

```
select * from pedido p
where exists (select 1 from linped l where l.numpedido=p.numpedido
                                       and articulo in (select cod from tv))
  and exists (select 1 from linped l where l.numpedido=p.numpedido
                                       and articulo in (select cod from camara));
```

T11.027- Pedidos que incluyen cámaras y objetivos.

```
select * from pedido p
where exists (select 1 from linped l where l.numpedido=p.numpedido
                                       and articulo in (select cod from camara))
  and exists (select 1 from linped l where l.numpedido=p.numpedido
                                       and articulo in (select cod from objetivo));
```

T11.028- Concatenación natural de artículos y memorias.

```
select * from articulo natural join memoria;
```

T11.029- Código de artículo, nombre, pvp, marca y tipo de la concatenación natural de artículos y memorias.

```
select cod,nombre,pvp,marca,tipo from articulo natural join memoria;
```

T11.030- Código de artículo, nombre, pvp, marca y tipo de la concatenación natural de artículos y memorias, si el tipo es "Compact Flash".

```
select cod,nombre,pvp,marca,tipo
from articulo natural join memoria
where tipo='Compact Flash';
```

Solución alternativa:

```
select a.cod,a.nombre,a.pvp,a.marca,m.tipo
from articulo a, memoria m
where a.cod=m.cod and tipo='Compact Flash';
```

T11.031- Concatenación natural de pedido y linped, ordenado por fecha de pedido.

```
select *
from pedido natural join linped
order by fecha;
```

Solución alternativa:

```
select p.*, linea,articulo,precio,cantidad
from pedido p, linped l
where p.numpedido=l.numpedido
order by fecha;
```

T11.032- Comprueba que la concatenación natural de cesta y pack produce un producto cartesiano.

```
select *
from cesta natural join pack;
```

Solución alternativa:

```
select *
from cesta, pack;
```

En CESTA se utiliza el nombre de columna "articulo" mientras que en pack se utiliza "cod": no hay columnas comunes.

T11.034- ¿Por qué la concatenación natural de usuario y direnvio resulta en una tabla vacía?

```
select *
from usuario natural join direnvio;
```

Entre USUARIO y DIRENVIO hay varias columnas comunes: "email", "calle", "calle2", "codpos", "pueblo" y "provincia". Más útil sería concatenar únicamente por "email"; si nos empeñamos en utilizar la concatenación natural la única solución es renombrar las columnas:

*select **

from usuario

natural join

(select email,calle dcalle,calle2 dcalle2,codpos dcodpos,pueblo dpueblo,provincia dprovincia from direnvio) as d

---

T12.001- Días que han pasado entre el primer y último pedido.

```
select datediff(
   (select max(fecha) from pedido),
   (select min(fecha) from pedido)
) dias;
```

Solución alternativa:

```
select datediff(max(fecha), min(fecha)) dias from pedido;
```

T12.002- Calcula y muestra la cantidad de televisores, cámaras y objetivos almacenados en la base de datos.

```
select
(select count(*) from tv) tv,
(select count(*) from camara) camaras,
(select count(*) from objetivo) objetivos;
```

T12.003- Calcula y muestra el porcentaje de televisores, cámaras y objetivos sobre el total de artículos almacenados en la base de datos.

```
select
(select count(*) from tv)/(select count(*) from articulo)*100 tvPC,
(select count(*) from camara)/(select count(*) from articulo)*100 camarasPC,
(select count(*) from objetivo)/(select count(*) from articulo)*100 objetivosPC;
```

T12.004- Email, nombre y apellidos de los usuarios de la provincia 03, y si tienen un pedido cuyo importe total sea mayor que 10000€, mostrar también el número de pedido y ese importe; ordena la salida descendentemente por el valor del pedido.

Comienza resolviendo número de pedido, usuario e importe total de los pedidos valorados en más de 10000€ y utilíza el resultado como tabla temporal.

```
select email,nombre, apellidos,numpedido, valor
from usuario u left join
  (select p.numpedido,usuario, sum(cantidad*precio) valor
   from pedido p, linped l
   where p.numpedido=l.numpedido
   group by numpedido,usuario
   having sum(cantidad*precio)>10000) calculo
on (email=usuario)
where provincia='03'
order by valor desc
```

T12.005- De los usuarios que tengan algún pedido sin líneas de pedido y artículos pendientes de solicitud en alguna cesta, mostrar su email, nombre, apellidos, número del pedido sin líneas, y valor total de su cesta.

Comienza resolviendo pedidos sin líneas y valor de la cesta por usuario y utilíza los resultados como tablas temporales.

```
select email,nombre, apellidos,numpedido, pendiente
from usuario u,
  (select numpedido,usuario
   from pedido
   where numpedido not in (select numpedido from linped)
  ) pedidos,
  (select usuario,sum(pvp) pendiente
   from cesta c, articulo a
   where c.articulo=a.cod
   group by usuario
  ) cestas
where email=pedidos.usuario and email=cestas.usuario;
```

Date cuenta que el último where utiliza columnas de las tablas temporales.

Para entender mejor qué hace esta consulta, ejecuta las subconsultas por separado:

select numpedido,usuario
from pedido
where numpedido not in (select numpedido from linped)

select usuario,sum(pvp) pendiente
from cesta c, articulo a
where c.articulo=a.cod
group by usuario

T12.006- Para aquellos usuarios que tengan más de un pedido en 2010, obtener una tabla donde cada columna se corresponda con un mes del año y muestre la cantidad de pedidos realizada por ese usuario en ese mes. Cada fila empieza por el email, nombre y apellidos del usuario.

```
select *
from
  (select email,nombre,apellidos,
   (select count(*) from pedido where month(fecha)=1 and year(fecha)=2010 and
usuario=email) enero,
   (select count(*) from pedido where month(fecha)=2 and year(fecha)=2010 and
usuario=email) febrero,
   (select count(*) from pedido where month(fecha)=3 and year(fecha)=2010 and
usuario=email) marzo,
   (select count(*) from pedido where month(fecha)=4 and year(fecha)=2010 and
usuario=email) abril,
   (select count(*) from pedido where month(fecha)=5 and year(fecha)=2010 and
usuario=email) mayo,
   (select count(*) from pedido where month(fecha)=6 and year(fecha)=2010 and
usuario=email) junio,
   (select count(*) from pedido where month(fecha)=7 and year(fecha)=2010 and
usuario=email) julio,
```

```
   (select count(*) from pedido where month(fecha)=8 and year(fecha)=2010 and
usuario=email) agosto,
   (select count(*) from pedido where month(fecha)=9 and year(fecha)=2010 and
usuario=email) septiembre,
   (select count(*) from pedido where month(fecha)=10 and year(fecha)=2010 and
usuario=email) octubre,
   (select count(*) from pedido where month(fecha)=11 and year(fecha)=2010 and
usuario=email) noviembre,
   (select count(*) from pedido where month(fecha)=12 and year(fecha)=2010 and
usuario=email) diciembre
   from usuario) pormeses
where
enero+febrero+marzo+abril+mayo+junio+julio+agosto+septiembre+octubre+noviembre+diciembr
e > 1;
```

Solución alternativa:

```
select email,nombre,apellidos,
   (select count(*) from pedido where month(fecha)=1 and year(fecha)=2010 and
usuario=email) enero,
   (select count(*) from pedido where month(fecha)=2 and year(fecha)=2010 and
usuario=email) febrero,
   (select count(*) from pedido where month(fecha)=3 and year(fecha)=2010 and
usuario=email) marzo,
   (select count(*) from pedido where month(fecha)=4 and year(fecha)=2010 and
usuario=email) abril,
   (select count(*) from pedido where month(fecha)=5 and year(fecha)=2010 and
usuario=email) mayo,
   (select count(*) from pedido where month(fecha)=6 and year(fecha)=2010 and
usuario=email) junio,
   (select count(*) from pedido where month(fecha)=7 and year(fecha)=2010 and
usuario=email) julio,
   (select count(*) from pedido where month(fecha)=8 and year(fecha)=2010 and
usuario=email) agosto,
   (select count(*) from pedido where month(fecha)=9 and year(fecha)=2010 and
usuario=email) septiembre,
   (select count(*) from pedido where month(fecha)=10 and year(fecha)=2010 and
usuario=email) octubre,
   (select count(*) from pedido where month(fecha)=11 and year(fecha)=2010 and
usuario=email) noviembre,
   (select count(*) from pedido where month(fecha)=12 and year(fecha)=2010 and
usuario=email) diciembre
from usuario
where email in (select usuario from pedido group by usuario having count(*) > 1);
```

BDgite (GITE-11014-UA)

Departamento de Lenguajes y Sistemas Informáticos

Universitat d'Alacant
Universidad de Alicante