```
-- 1.3.3
-- 1. Lista el nombre de todos los productos que hay en la tabla producto.
SELECT nombre FROM producto;
-- 2  Lista los nombres y los precios de todos los productos de la tabla producto.
SELECT nombre, precio_venta, precio_proveedor FROM producto;
-- 3 Lista todas las columnas de la tabla producto.
SELECT * FROM producto;

-- 1.4.4
-- 1 Devuelve un listado con el código de oficina y la ciudad donde hay oficinas.
SELECT codigo_oficina, ciudad FROM oficina WHERE codigo_oficina IS NOT NULL;

-- 2 Devuelve un listado con la ciudad y el teléfono de las oficinas de España.
SELECT ciudad, telefono FROM oficina WHERE pais = "España";

-- 3 Devuelve un listado con el nombre, apellidos y email de los empleados cuyo jefe tiene
un código de jefe igual a 7.
SELECT nombre,apellido1, apellido2, email FROM empleado WHERE codigo_jefe = "2";

-- 4 Devuelve el nombre del puesto, nombre, apellidos y email del jefe de la empresa.
SELECT puesto, nombre, apellido1, apellido2, email FROM empleado WHERE codigo_jefe
IS NULL;

-- 5 Devuelve un listado con el nombre, apellidos y puesto de aquellos empleados que no
sean representantes de ventas.
SELECT nombre, apellido1, apellido2, puesto FROM empleado WHERE puesto !=
"Representante Ventas";

-- 6 Devuelve un listado con el nombre de los todos los clientes españoles.
SELECT nombre_cliente FROM cliente WHERE pais = "Spain";

-- 7 Devuelve un listado con los distintos estados por los que puede pasar un pedido.
SELECT DISTINCT estado FROM pedido;

-- 8 Devuelve un listado con el código de cliente de aquellos clientes que realizaron algún
pago en 2008. Tenga en cuenta que deberá eliminar aquellos códigos de cliente que
aparezcan repetidos. Resuelva la consulta:
--  Utilizando la función YEAR de MySQL.
SELECT codigo_cliente, fecha_pago FROM pago WHERE year(fecha_pago) = "2008";
--  Utilizando la función DATE_FORMAT de MySQL.
SELECT DISTINCT codigo_cliente, fecha_pago FROM pago WHERE (fecha_pago >=
"2008-01-01" AND fecha_pago < "2009-01-01");

--  Sin utilizar ninguna de las funciones anteriores.
SELECT DISTINCT codigo_cliente, fecha_pago FROM pago WHERE fecha_pago LIKE
"2008%";
```

-- 9 Devuelve un listado con el código de pedido, código de cliente, fecha esperada y fecha de entrega de los pedidos que no han sido entregados a tiempo.
SELECT codigo_pedido, codigo_cliente, fecha_esperada, fecha_entrega FROM pedido WHERE fecha_entrega > fecha_esperada;

-- 10 Devuelve un listado con el código de pedido, código de cliente, fecha esperada y fecha de entrega de los pedidos cuya fecha de entrega ha sido al menos dos días antes de la fecha esperada.
-- Utilizando la función ADDDATE de MySQL.
SELECT codigo_pedido, codigo_cliente, fecha_esperada, fecha_entrega FROM pedido WHERE fecha_entrega <= (ADDDATE (fecha_esperada, INTERVAL -2 DAY));
-- Utilizando la función DATEDIFF de MySQL.
SELECT codigo_pedido, codigo_cliente, fecha_esperada, fecha_entrega FROM pedido WHERE DATEDIFF(fecha_entrega, fecha_esperada) <= "-2";
SELECT codigo_pedido, codigo_cliente, fecha_esperada, fecha_entrega FROM pedido WHERE DATEDIFF(fecha_esperada, fecha_entrega) >= "2";
-- ¿Sería posible resolver esta consulta utilizando el operador de suma + o resta -?
SELECT codigo_pedido, codigo_cliente, fecha_esperada, fecha_entrega FROM pedido WHERE fecha_entrega <= (fecha_esperada - INTERVAL 2 DAY);

-- 11 vuelve un listado de todos los pedidos que fueron rechazados en 2009.
SELECT codigo_pedido, estado FROM pedido WHERE (estado = "rechazado" AND year(fecha_entrega) = '2009');

-- 12 Devuelve un listado de todos los pedidos que han sido entregados en el mes de enero de cualquier año.
SELECT codigo_pedido, fecha_entrega FROM pedido WHERE month(fecha_entrega) = "01";

-- 13 Devuelve un listado con todos los pagos que se realizaron en el año 2008 mediante Paypal. Ordene el resultado de mayor a menor.
SELECT total, fecha_pago, forma_pago FROM pago WHERE forma_pago = "PayPal" ORDER BY total DESC;

-- 14 Devuelve un listado con todas las formas de pago que aparecen en la tabla pago. Tenga en cuenta que no deben aparecer formas de pago repetidas.
SELECT DISTINCT forma_pago FROM pago;

-- 15 Devuelve un listado con todos los productos que pertenecen a la gama Ornamentales y que tienen más de 100 unidades en stock. El listado deberá estar ordenado por su precio de venta, mostrando en primer lugar los de mayor precio.
SELECT nombre, gama, cantidad_en_stock, precio_venta FROM producto WHERE cantidad_en_stock > "100" ORDER BY precio_venta DESC;

-- 16 Devuelve un listado con todos los clientes que sean de la ciudad de Madrid y cuyo representante de ventas tenga el código de empleado 11 o 30.
SELECT nombre_cliente, ciudad, codigo_empleado_rep_ventas AS "representante", codigo_empleado, nombre FROM cliente, empleado WHERE (ciudad = "Madrid" AND

```
codigo_empleado_rep_ventas = codigo_empleado AND (codigo_empleado = "11" OR
codigo_empleado = "30"));

-- 1.4.5

-- Resuelva todas las consultas utilizando la sintaxis de SQL1 y SQL2. Las consultas con
sintaxis de SQL2 se deben resolver con INNER JOIN y NATURAL JOIN.
-- 1 Obtén un listado con el nombre de cada cliente y el nombre y apellido de su
representante de ventas.
SELECT c1.nombre_cliente AS cliente, e2.nombre AS representante_nombre, e2.apellido1
AS representante_apellido FROM cliente c1  INNER JOIN empleado e2 on
c1.codigo_empleado_rep_ventas = e2.codigo_empleado;

-- 2 Muestra el nombre de los clientes que hayan realizado pagos junto con el nombre de
sus representantes de ventas.
SELECT c1.nombre_cliente AS Cliente, p1.total AS Pago, e1.nombre AS Representante
FROM cliente c1 INNER JOIN pago p1 INNER JOIN empleado e1 on p1.total IS NOT NULL;

-- 3 Muestra el nombre de los clientes que no hayan realizado pagos junto con el nombre de
sus representantes de ventas.
SELECT nombre_cliente AS Cliente, total AS Pago, nombre AS Representante FROM
cliente INNER JOIN pago INNER JOIN empleado on total IS NULL;

-- 4 Devuelve el nombre de los clientes que han hecho pagos y el nombre de sus
representantes junto con la ciudad de la oficina a la que pertenece el representante.
select  c1.nombre_cliente AS Cliente,
        p1.total AS Pagos,
        e1.nombre AS Representante,
        o1.ciudad AS Ciudad_Oficina
FROM pago p1 INNER JOIN cliente c1 INNER JOIN empleado e1 INNER JOIN oficina o1
ON p1.total IS NOT NULL AND c1.codigo_empleado_rep_ventas = e1.codigo_empleado
AND e1.codigo_oficina = o1.codigo_oficina;

-- 5 Devuelve el nombre de los clientes que no hayan hecho pagos y el nombre de sus
representantes junto con la ciudad de la oficina a la que pertenece el representante.
SELECT  c1.nombre_cliente AS cliente,
        p1.total AS Pagos,
        e1.nombre AS representante,
        o1.ciudad
FROM pago p1 INNER JOIN cliente c1 INNER JOIN empleado e1 INNER JOIN oficina o1
ON c1.codigo_cliente = p1.codigo_cliente AND p1.total IS NULL AND
c1.codigo_empleado_rep_ventas = e1.codigo_empleado AND e1.codigo_oficina =
o1.codigo_oficina;

-- 6 Lista la dirección de las oficinas que tengan clientes en Fuenlabrada.
SELECT  o1.codigo_oficina AS Oficina,
        o1.ciudad AS Ciudad_Oficina,
        o1.codigo_postal AS CP_oficina,
```

```
        c1.nombre_cliente AS Cliente,
        c1.ciudad AS Ciudad_cliente
FROM oficina o1 INNER JOIN empleado e1 INNER JOIN cliente c1
ON o1.codigo_oficina = e1.codigo_oficina AND e1.codigo_empleado =
c1.codigo_empleado_rep_ventas AND c1.ciudad = "Fuenlabrada";

-- 7 Devuelve el nombre de los clientes y el nombre de sus representantes junto con la
ciudad de la oficina a la que pertenece el representante.
SELECT  c1.nombre_cliente AS Cliente,
        e1.nombre AS Representante,
        o1.ciudad AS Ciudad_oficina
FROM cliente c1 INNER JOIN empleado e1 INNER JOIN oficina o1
ON c1.codigo_empleado_rep_ventas = e1.codigo_empleado AND e1.codigo_oficina =
o1.codigo_oficina;

-- 8 Devuelve un listado con el nombre de los empleados junto con el nombre de sus jefes.
SELECT e1.nombre AS Empleado,
        e2.nombre AS Jefe
FROM empleado e1 INNER JOIN empleado e2
ON e1.codigo_jefe = e2.codigo_empleado;

-- 9 Devuelve un listado que muestre el nombre de cada empleados, el nombre de su jefe y
el nombre del jefe de sus jefe.
SELECT e1.nombre AS Empleado,
        e2.nombre AS Jefe,
        e3.nombre AS Jefe_de_Jefe
FROM empleado e1 INNER JOIN empleado e2 INNER JOIN empleado e3
ON e1.codigo_jefe = e2.codigo_empleado AND e2.codigo_jefe = e3.codigo_empleado;

-- 10 Devuelve el nombre de los clientes a los que no se les ha entregado a tiempo un
pedido.
SELECT c1.nombre_cliente AS Cliente,
        p1.fecha_esperada AS Fecha_esperada,
        p1.fecha_entrega AS Entrega
FROM cliente c1 INNER JOIN pedido p1
ON fecha_entrega > fecha_esperada;

-- 11 Devuelve un listado de las diferentes gamas de producto que ha comprado cada
cliente.
SELECT c1.nombre_cliente AS Cliente,
        g1.gama AS Gamas
FROM cliente c1 INNER JOIN pedido p1 INNER JOIN detalle_pedido d1 INNER JOIN
producto pr1 INNER JOIN gama_producto g1
ON c1.codigo_cliente = p1.codigo_cliente AND p1.codigo_pedido = d1.codigo_pedido AND
d1.codigo_producto = pr1.codigo_producto AND pr1.gama = g1.gama;

-- 1.4.6 Resuelva todas las consultas utilizando las cláusulas LEFT JOIN, RIGHT JOIN,
NATURAL LEFT JOIN y NATURAL RIGHT JOIN.
```

```
-- 1 Devuelve un listado que muestre solamente los clientes que no han realizado ningún pago.
SELECT c1.nombre_cliente AS Cliente, p1.id_transaccion AS Pago
FROM cliente c1 LEFT JOIN pago p1
ON c1.codigo_cliente = p1.codigo_cliente
WHERE p1.id_transaccion IS NULL;

-- 2 Devuelve un listado que muestre solamente los clientes que no han realizado ningún pedido.
SELECT c1.nombre_cliente AS Cliente, p1.codigo_pedido AS Pedido
FROM cliente c1 LEFT JOIN pedido p1
ON c1.codigo_cliente = p1.codigo_pedido
WHERE p1.codigo_pedido IS NULL;

-- 3 Devuelve un listado que muestre los clientes que no han realizado ningún pago y los que no han realizado ningún pedido.
SELECT c.nombre_cliente AS Cliente, pa.id_transaccion AS Pago, pe.codigo_pedido AS Pedido
FROM cliente c LEFT JOIN pago pa NATURAL JOIN pedido pe
ON pa.codigo_cliente = c.codigo_cliente
WHERE pa.id_transaccion IS NULL AND pe.codigo_pedido IS NULL;

-- 4 Devuelve un listado que muestre solamente los empleados que no tienen una oficina asociada.
SELECT e.nombre AS Empleado, o.codigo_oficina AS Oficina
FROM empleado e NATURAL LEFT JOIN oficina o
WHERE o.codigo_oficina IS NULL;

-- 5 Devuelve un listado que muestre solamente los empleados que no tienen un cliente asociado.
SELECT e.nombre AS Empleado, c.nombre_cliente AS Cliente
FROM empleado e LEFT JOIN cliente c
ON e.codigo_empleado = c.codigo_empleado_rep_ventas
WHERE c.codigo_cliente IS NULL;

-- 6 Devuelve un listado que muestre solamente los empleados que no tienen un cliente asociado junto con los datos de la oficina donde trabajan.
SELECT e.nombre AS Empleado, c.nombre_cliente AS Cliente, o.*
FROM empleado e LEFT JOIN cliente c NATURAL LEFT JOIN oficina o
ON e.codigo_empleado = c.codigo_empleado_rep_ventas
WHERE c.codigo_cliente IS NULL;

-- 7 Devuelve un listado que muestre los empleados que no tienen una oficina asociada y los que no tienen un cliente asociado.
SELECT e.nombre AS Empleado, c.nombre_cliente AS Cliente, o.codigo_oficina AS Oficina
FROM empleado e
LEFT JOIN cliente c ON e.codigo_empleado = c.codigo_empleado_rep_ventas
```

```
LEFT JOIN oficina o ON o.codigo_oficina = e.codigo_oficina
WHERE o.ciudad IS NULL OR c.codigo_cliente IS NULL;

-- 8 Devuelve un listado de los productos que nunca han aparecido en un pedido.
SELECT pr.nombre AS Producto, pe.codigo_pedido
FROM producto pr
NATURAL LEFT JOIN detalle_pedido
NATURAL LEFT JOIN pedido pe
WHERE pe.fecha_pedido IS NULL;

-- 9 Devuelve un listado de los productos que nunca han aparecido en un pedido. El resultado debe mostrar el nombre, la descripción y la imagen del producto.
SELECT pr.nombre AS Producto, g.descripcion_texto AS Descripcion, g.imagen AS Imagen
FROM producto pr
NATURAL LEFT JOIN gama_producto g
NATURAL LEFT JOIN detalle_pedido
NATURAL LEFT JOIN pedido pe
WHERE pe.fecha_pedido IS NULL;

-- 10 Devuelve las oficinas donde no trabajan ninguno de los empleados que hayan sido los representantes de ventas de algún cliente que haya realizado la compra de algún producto de la gama Frutales.
SELECT o.codigo_oficina AS Oficina, e.codigo_empleado AS Empleado, c.nombre_cliente AS Cliente, pr.nombre AS Producto, g.gama AS Gama
FROM oficina o
LEFT JOIN empleado e ON o.codigo_oficina = e.codigo_oficina
LEFT JOIN cliente c ON e.codigo_empleado = c.codigo_empleado_rep_ventas
LEFT JOIN pedido pe ON pe.codigo_cliente = c.codigo_cliente
LEFT JOIN detalle_pedido dp ON pe.codigo_pedido = dp.codigo_pedido
LEFT JOIN producto pr ON dp.codigo_producto = pr.codigo_producto
LEFT JOIN gama_producto g ON pr.gama = g.gama
WHERE NOT (g.gama = "Frutales" AND e.puesto = "Representante Ventas")
GROUP BY o.codigo_oficina;

-- 11 Devuelve un listado con los clientes que han realizado algún pedido pero no han realizado ningún pago.
SELECT c.nombre_cliente AS Cliente, pe.codigo_pedido AS Pedido, pa.id_transaccion AS Pago
FROM cliente c
NATURAL LEFT JOIN pedido pe
NATURAL LEFT JOIN pago pa
WHERE pe.estado IS NOT NULL AND pa.fecha_pago IS NULL
GROUP BY c.nombre_cliente;

-- 12 Devuelve un listado con los datos de los empleados que no tienen clientes asociados y el nombre de su jefe asociado.
SELECT e1.*, c.telefono AS Cliente, e2.nombre AS Jefe
FROM empleado e1
```

```
LEFT JOIN cliente c ON e1.codigo_empleado = c.codigo_empleado_rep_ventas
LEFT JOIN empleado e2 ON e1.codigo_jefe = e2.codigo_empleado
WHERE c.telefono IS NULL;

-- 1.4.7 Consultas resumen
-- 1 ¿Cuántos empleados hay en la compañía?
SELECT MAX(codigo_empleado) FROM empleado;
SELECT COUNT(codigo_empleado) FROM empleado;
-- 2 ¿Cuántos clientes tiene cada país?
SELECT DISTINCT pais FROM cliente; -- ver que paises hay
SELECT COUNT(pais) AS USA FROM cliente WHERE pais = "USA"; -- ver cuantos clientes
hay en usa
SELECT COUNT(pais) AS Spain FROM cliente WHERE pais = "Spain"; -- ver cuantos
clientes hay en españa
SELECT COUNT(pais) AS France FROM cliente WHERE pais = "France"; -- ver cuantos
clientes hay en francia
SELECT COUNT(pais) AS Australia FROM cliente WHERE pais = "Australia"; -- ver cuantos
clientes hay en australia
SELECT COUNT(pais) AS United_Kingdom FROM cliente WHERE pais = "United
Kingdom"; -- ver cuantos clientes hay en uk

SELECT DISTINCT pais AS Pais, COUNT (pais) AS Cantidad -- ver cuantos clientes hay en
cada pais
FROM cliente
GROUP BY pais;

-- 3 ¿Cuál fue el pago medio en 2009?
SELECT AVG(total)
FROM pago
WHERE year(fecha_pago) = "2009";

-- 4 ¿Cuántos pedidos hay en cada estado? Ordena el resultado de forma descendente por
el número de pedidos.
SELECT DISTINCT estado as Estado, COUNT (estado)
FROM pedido
GROUP BY (estado);

-- 5 Calcula el precio de venta del producto más caro y más barato en una misma consulta.
SELECT DISTINCT MAX(total) AS Caro, MIN(total) as Barato
FROM pago;

-- 6 Calcula el número de clientes que tiene la empresa.
SELECT COUNT(codigo_cliente) AS cantidad_clientes
FROM cliente;

-- 7 ¿Cuántos clientes existen con domicilio en la ciudad de Madrid?
SELECT COUNT(ciudad) AS Clientes, ciudad AS Ciudad
FROM cliente
```

```
WHERE ciudad = "Madrid";

-- 8 ¿Calcula cuántos clientes tiene cada una de las ciudades que empiezan por M?
SELECT DISTINCT ciudad AS Ciudad, COUNT(ciudad) AS Clientes
FROM cliente
WHERE ciudad LIKE "M%"
GROUP BY (ciudad);

-- 9 Devuelve el nombre de los representantes de ventas y el número de clientes al que atiende cada uno.
SELECT DISTINCT e.nombre AS Representante, COUNT(c.codigo_cliente) AS Clientes
FROM empleado e
INNER JOIN cliente c
ON e.codigo_empleado = codigo_empleado_rep_ventas
GROUP BY (e.nombre);

-- 10 Calcula el número de clientes que no tiene asignado representante de ventas.
SELECT COUNT(codigo_empleado_rep_ventas) AS Clientes_sin_representante
FROM cliente
WHERE codigo_empleado_rep_ventas IS NULL;

-- 11 Calcula la fecha del primer y último pago realizado por cada uno de los clientes. El listado deberá mostrar el nombre y los apellidos de cada cliente.
SELECT nombre_cliente AS Cliente, MAX(fecha_pago) AS Ultimo_pago, MIN(fecha_pago) AS primer_pago
FROM cliente
NATURAL JOIN pago
GROUP BY codigo_cliente;

-- 12 Calcula el número de productos diferentes que hay en cada uno de los pedidos.
SELECT pe.codigo_pedido AS Pedido, COUNT (DISTINCT pr.nombre) AS Productos
FROM pedido pe
NATURAL JOIN detalle_pedido dp
NATURAL JOIN producto pr
GROUP BY pe.codigo_pedido;

-- 13 Calcula la suma de la cantidad total de todos los productos que aparecen en cada uno de los pedidos.
  SELECT pe.codigo_pedido AS Pedido, COUNT (pr.nombre) AS Productos
FROM pedido pe
NATURAL JOIN detalle_pedido dp
NATURAL JOIN producto pr
GROUP BY pe.codigo_pedido;

-- 14 Devuelve un listado de los 20 productos más vendidos y el número total de unidades que se han vendido de cada uno. El listado deberá estar ordenado por el número total de unidades vendidas.
SELECT DISTINCT pr.nombre AS Nombre, dp.cantidad AS Unidades
```

```
FROM producto pr
NATURAL JOIN detalle_pedido dp
ORDER BY dp.cantidad DESC LIMIT 0,20;

-- 15 La facturación que ha tenido la empresa en toda la historia, indicando la base imponible, el IVA y el total facturado. La base imponible se calcula sumando el coste del producto por el número de unidades vendidas de la tabla detalle_pedido. El IVA es el 21 % de la base imponible, y el total la suma de los dos campos anteriores.
SELECT codigo_pedido AS Pedido ,dp.codigo_producto  AS Producto, dp.cantidad * dp.precio_unidad AS 'Base imponible',dp.cantidad * dp.precio_unidad * (21 / 100) AS 'IVA', dp.cantidad * dp.precio_unidad + (dp.cantidad * dp.precio_unidad * (21 / 100)) AS 'Total Facturado'
FROM detalle_pedido dp;

-- 16 La misma información que en la pregunta anterior, pero agrupada por código de producto.
SELECT dp.codigo_producto  AS Producto, SUM(dp.cantidad * dp.precio_unidad) AS 'Base imponible',SUM(dp.cantidad * dp.precio_unidad * (21 / 100)) AS 'IVA', SUM(dp.cantidad * dp.precio_unidad + (dp.cantidad * dp.precio_unidad * (21 / 100))) AS 'Total Facturado'
FROM detalle_pedido dp
GROUP BY dp.codigo_producto;

-- 17 La misma información que en la pregunta anterior, pero agrupada por código de producto filtrada por los códigos que empiecen por OR.
SELECT dp.codigo_producto  AS Producto, SUM(dp.cantidad * dp.precio_unidad) AS 'Base imponible',SUM(dp.cantidad * dp.precio_unidad * (21 / 100)) AS 'IVA', SUM(dp.cantidad * dp.precio_unidad + (dp.cantidad * dp.precio_unidad * (21 / 100))) AS 'Total Facturado'
FROM detalle_pedido dp
WHERE dp.codigo_producto LIKE "OR%"
GROUP BY dp.codigo_producto;

-- 18 Lista las ventas totales de los productos que hayan facturado más de 3000 euros. Se mostrará el nombre, unidades vendidas, total facturado y total facturado con impuestos (21% IVA).


SELECT p.nombre AS 'Nombre', COUNT(dp.cantidad) AS 'Cantidad', SUM(dp.cantidad * dp.precio_unidad) AS 'Total Facturado' , SUM(dp.cantidad * dp.precio_unidad + (dp.cantidad * dp.precio_unidad * (21 / 100))) AS 'Total Facturado con IVA'
FROM detalle_pedido dp
INNER JOIN producto p ON dp.codigo_producto = p.codigo_producto
WHERE 'Total Facturado' > "3000";


-- 19 Muestre la suma total de todos los pagos que se realizaron para cada uno de los años que aparecen en la tabla pagos.

-- 1.4.8.1 Con operadores básicos de comparación
```

```
-- 1 Devuelve el nombre del cliente con mayor límite de crédito.
SELECT nombre_cliente AS Cliente, limite_credito AS 'Limite de crédito'
FROM cliente
WHERE limite_credito = (SELECT MAX(DISTINCT limite_credito) FROM cliente);

-- 2 Devuelve el nombre del producto que tenga el precio de venta más caro.
SELECT nombre AS Producto, precio_venta AS 'Precio venta'
FROM producto
WHERE precio_venta = (SELECT MAX(DISTINCT precio_venta) FROM producto);

-- 3 Devuelve el nombre del producto del que se han vendido más unidades. (Tenga en
cuenta que tendrá que calcular cuál es el número total de unidades que se han vendido de
cada producto a partir de los datos de la tabla detalle_pedido)
-- DA ERROR
SELECT p.codigo_producto AS Producto, SUM(dp.cantidad) AS 'Unidades vendidas totales'
FROM detalle_pedido dp
INNER JOIN producto p ON p.codigo_producto = dp.codigo_producto
GROUP BY p.codigo_producto;
WHERE (SUM(dp.cantidad) GROUP BY p.codigo_producto)= (SELECT
MAX(SUM(dp.cantidad) GROUP BY p.codigo_producto) FROM producto p);

-- 4 Los clientes cuyo límite de crédito sea mayor que los pagos que haya realizado. (Sin
utilizar INNER JOIN).

SELECT cli.nombre_cliente,cli.codigo_cliente,cli.limite_credito
from cliente cli
where cli.limite_credito >
(SELECT sum(p.total)from pago p where cli.codigo_cliente =p.codigo_cliente GROUP by
p.codigo_cliente);

-- 5 Devuelve el producto que más unidades tiene en stock.
SELECT nombre AS Producto, cantidad_en_stock AS 'Cantidad en stock'
FROM producto
WHERE cantidad_en_stock = (SELECT (MAX(cantidad_en_stock)) FROM producto);

-- 6 Devuelve el producto que menos unidades tiene en stock.
SELECT nombre AS Producto, cantidad_en_stock AS 'Cantidad en stock'
FROM producto
WHERE cantidad_en_stock = (SELECT (MIN(cantidad_en_stock)) FROM producto);

-- 7 Devuelve el nombre, los apellidos y el email de los empleados que están a cargo de
Alberto Soria.
SELECT e1.nombre, e1.apellido1, e1.apellido2, e1.email, e2.nombre, e2.apellido1
FROM empleado e1
INNER JOIN empleado e2 ON e1.codigo_jefe = e2.codigo_empleado
WHERE e2.nombre = (SELECT e2.nombre WHERE e2.nombre LIKE "Alberto Soria");
```