# Consultas multitabla

* 1. Devuelve una lista de todos los productos del fabricante Lenovo.

SELECT \*

FROM producto

WHERE id\_fabricante = (SELECT id FROM fabricante WHERE nombre = 'Įenovo');

* 1. Devuelve una lista de todos los productos del fabricante Crucial que tengan un precio mayor que 200.

SELECT \*

FROM producto

WHERE id\_fabricante = (SELECT id FROM fabricante WHERE nombre = 'Crucial') AND precio > 200;

* 1. Devuelve un listado con todos los productos de los fabricantes Asus, Hewlett- Packardy Seagate. Sin utilizar el operador IN.

SELECT\*

FROM producto

WHERE id\_fabricante = (SELECTid FROM fabricante WHERE nombre = 'Asus')

OR id\_fabricante = (SELECTid FROM fabricante WHERE nombre

= 'Hewlett-Packard')

OR id\_fabricante = (SELECTid FROM fabricante WHERE nombre

= 'Seagate');

* 1. Devuelve un listado con todos los productos de los fabricantes Asus, Hewlett- Packard y Seagate. Utilizando el operador IN.

SELECT\*

FROM producto

WHERE id\_fabricante IN (SELECTid FROM fabricante WHERE nombre IN ('Asus', 'Hewlett-Packard', 'Seagate'));

* 1. Devuelve un listado con el nombre y el precio de todos los productos de los fabricantes cuyo nombre termine por la vocal e.

SELECTp.nombre, p.precio FROM producto p, fabricante f

WHERE p.id\_fabricante = f.id AND f.nombre ĮIKE '%e';

* 1. Devuelve un listado con el nombre y el precio de todos los productos cuyo nombre de fabricante contenga el carácter w en su nombre.

SELECTp.nombre, p.precio FROM producto p, fabricante f

WHERE p.id\_fabricante = f.id AND f.nombre ĮIKE '%w%';

* 1. Devuelve un listado con el nombre de producto, precio y nombre de fabricante, de todos los productos que tengan un precio mayor o igual a 180. Ordene el resultado en primer lugar por el precio (en orden descendente) y en segundo lugar por el nombre (en orden ascendente)

SELECTp.nombre, p.precio, f.nombre FROM producto p, fabricante f

WHERE p.id\_fabricante = f.id AND p.precio >= 180 ORDER BY p.precio DESC, p.nombre ASC;

* 1. Devuelve un listado con el identificador y el nombre de fabricante, solamente de aquellos fabricantes que tienen productos asociados en la base de datos.

SELECTf.id, f.nombre FROM fabricante f

WHERE EXISTS (SELECT 1 FROM producto p WHERE

p.id\_fabricante = f.id);

Resuelva las siguientes consultas utilizando las cláusulas ***LEFT JOIN*** y ***RIGHT JOIN***.

1. Devuelve un listado de **todos los fabricantes** que existen en la base de datos, junto con los productos que tiene cada uno de ellos. El listado deberá mostrar también aquellos fabricantes que no tienen productos asociados.
2. Devuelve un listado donde sólo aparezcan aquellos fabricantes que no tienen ningún producto asociado.
3. ¿Pueden existir productos que no estén relacionados con un fabricante? Justifique su respuesta.

# Consultas resumen

1. Calcula el número total de productos que hay en la tabla productos.
2. Calcula el número total de fabricantes que hay en la tabla fabricante.
3. Calcula el número de valores distintos de identificador de fabricante aparecen en la tabla productos.
4. Calcula la media del precio de todos los productos.
5. Calcula el precio más barato de todos los productos.
6. Calcula el precio más caro de todos los productos.
7. Lista el nombre y el precio del producto más barato.
8. Lista el nombre y el precio del producto más caro.
9. Calcula la suma de los precios de todos los productos.
10. Calcula el número de productos que tiene el fabricante Asus.
11. Calcula la media del precio de todos los productos del fabricante Asus.
12. Calcula el precio más barato de todos los productos del fabricante Asus.
13. Calcula el precio más caro de todos los productos del fabricante Asus.
14. Calcula la suma de todos los productos del fabricante Asus.
15. Muestra el precio máximo, precio mínimo, precio medio y el número total de productos que tiene el fabricante Crucial.
16. Muestra el número total de productos que tiene cada uno de los fabricantes. El listado también debe incluir los fabricantes que no tienen ningún producto. El resultado mostrará dos columnas, una con el nombre del fabricante y otra con el número de productos que tiene. Ordene el resultado descendentemente por el número de productos.
17. Muestra el precio máximo, precio mínimo y precio medio de los productos de cada uno de los fabricantes. El resultado mostrará el nombre del fabricante junto con los datos que se solicitan.
18. Muestra el precio máximo, precio mínimo, precio medio y el número total de productos de los fabricantes que tienen un precio medio superior a 200. No es necesario mostrar el nombre del fabricante, con el identificador del fabricante es suficiente.
19. Muestra el nombre de cada fabricante, junto con el precio máximo, precio mínimo, precio medio y el número total de productos de los fabricantes que tienen un precio medio superior a 200. Es necesario mostrar el nombre del fabricante.
20. Calcula el número de productos que tienen un precio mayor o igual a 180.
21. Calcula el número de productos que tiene cada fabricante con un precio mayor o igual a 180.
22. Lista el precio medio los productos de cada fabricante, mostrando solamente el identificador del fabricante.
23. Lista el precio medio los productos de cada fabricante, mostrando solamente el nombre del fabricante.
24. Lista los nombres de los fabricantes cuyos productos tienen un precio medio mayor o igual a 150.
25. Devuelve un listado con los nombres de los fabricantes que tienen 2 o más productos.

-----------------------------------------------------------------------------

SELECT f.nombre, COUNT(p.codigo) AS num\_productos

FROM fabricante f

LEFT JOIN producto p ON f.codigo = p.codigo\_fabricante

GROUP BY f.nombre;

SELECT f.nombre

FROM fabricante f

LEFT JOIN producto p ON f.codigo = p.codigo\_fabricante

WHERE p.codigo IS NULL;

SELECT COUNT(\*) AS total\_productos

FROM producto;

SELECT COUNT(\*) AS total\_fabricantes

FROM fabricante;

SELECT COUNT(DISTINCT codigo\_fabricante) AS total\_valores\_distintos

FROM producto;

SELECT AVG(precio) AS media\_precio

FROM producto;

SELECT MIN(precio) AS precio\_mas\_barato

FROM producto;

SELECT MAX(precio) AS precio\_mas\_caro

FROM producto;

SELECT nombre, precio

FROM producto

WHERE precio = (SELECT MIN(precio) FROM producto);

SELECT nombre, precio

FROM producto

WHERE precio = (SELECT MAX(precio) FROM producto);

SELECT SUM(precio) AS suma\_precios

FROM producto;

SELECT COUNT(\*) AS cantidad\_productos

FROM producto

WHERE codigo\_fabricante = (SELECT codigo FROM fabricante WHERE nombre = 'Asus');

SELECT AVG(precio) AS media\_precios

FROM producto

WHERE codigo\_fabricante = (SELECT codigo FROM fabricante WHERE nombre = 'Asus');

SELECT MIN(precio) AS precio\_mas\_barato

FROM producto

WHERE codigo\_fabricante = (SELECT codigo FROM fabricante WHERE nombre = 'Asus');

SELECT MAX(precio) AS precio\_mas\_caro

FROM producto

WHERE codigo\_fabricante = (SELECT codigo FROM fabricante WHERE nombre = 'Asus');

SELECT SUM(precio) AS suma\_precios

FROM producto

WHERE codigo\_fabricante = (SELECT codigo FROM fabricante WHERE nombre = 'Asus');

SELECT MAX(precio) AS precio\_maximo, MIN(precio) AS precio\_minimo, AVG(precio) AS

precio\_medio, COUNT(\*) AS cantidad\_productos

FROM producto

WHERE codigo\_fabricante = (SELECT codigo FROM fabricante WHERE nombre = 'Crucial');

SELECT f.nombre AS fabricante, COUNT(p.codigo) AS numero\_productos

FROM fabricante f

LEFT JOIN producto p ON f.codigo = p.codigo\_fabricante

GROUP BY f.nombre

ORDER BY numero\_productos DESC;

SELECT f.nombre AS fabricante, MAX(p.precio) AS precio\_maximo,

MIN(p.precio) AS precio\_minimo, AVG(p.precio) AS precio\_medio

FROM fabricante f

LEFT JOIN producto p ON f.codigo = p.codigo\_fabricante

GROUP BY f.nombre;

SELECT codigo\_fabricante, MAX(precio) AS precio\_maximo,

MIN(precio) AS precio\_minimo, AVG(precio) AS precio\_medio,

COUNT(codigo) AS numero\_productos

FROM producto

GROUP BY codigo\_fabricante

HAVING precio\_medio > 200;

select \* from producto;

SELECT f.nombre AS fabricante, MAX(p.precio) AS precio\_maximo,

MIN(p.precio) AS precio\_minimo, AVG(p.precio) AS precio\_medio,

COUNT(p.codigo) AS numero\_productos

FROM fabricante f

LEFT JOIN producto p ON f.codigo = p.codigo\_fabricante

GROUP BY f.nombre

HAVING precio\_medio > 200;

SELECT COUNT(\*) AS numero\_productos

FROM producto

WHERE precio >= 180;

SELECT f.nombre AS fabricante, COUNT(p.codigo) AS numero\_productos

FROM fabricante f

LEFT JOIN producto p ON f.codigo = p.codigo\_fabricante

WHERE p.precio >= 180

GROUP BY f.nombre;

SELECT codigo\_fabricante, AVG(precio) AS precio\_medio

FROM producto

GROUP BY codigo\_fabricante;

SELECT f.nombre AS fabricante, AVG(p.precio) AS precio\_medio

FROM fabricante f

LEFT JOIN producto p ON f.codigo = p.codigo\_fabricante

GROUP BY f.nombre;

SELECT f.nombre AS fabricante

FROM fabricante f

LEFT JOIN producto p ON f.codigo = p.codigo\_fabricante

GROUP BY f.nombre

HAVING AVG(p.precio) >= 150;

SELECT f.nombre AS fabricante

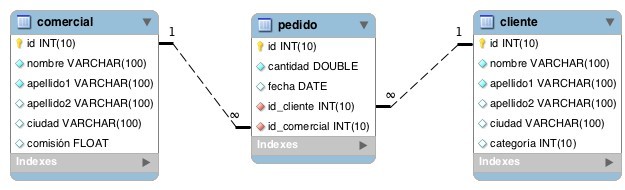
FROM fabricante f

LEFT JOIN producto p ON f.codigo = p.codigo\_fabricante

GROUP BY f.nombre

HAVING COUNT(p.codigo) >= 2;

Cree la siguiente Base de datos “Ventas” con sus tres tablas, e introduzca los datos indicados:



CREATE DATABASE ventas;

USE ventas;

CREATE TABĮE cliente (

id INT AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(100), apellido1 VARCHAR(100), apellido2 VARCHAR(100), ciudad VARCHAR(100), categoria INT

);

CREATE TABĮE comercial (

id INT AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(100), apellido1 VARCHAR(100), apellido2 VARCHAR(100), comision FĮOAT

);

CREATE TABĮE pedido (

id INT AUTO\_INCREMENT PRIMARY KEY, total DOUBĮE NOT NUĮĮ,

fecha DATE, id\_cliente INT, id\_comercial INT,

FOREIGN KEY (id\_cliente) REFERENCES cliente(id), FOREIGN KEY (id\_comercial) REFERENCES comercial(id)

);

INSERT INTO cliente (nombre, apellido1, apellido2, ciudad, categoria) VAĮUES('Aarón', 'Rivero', 'Gómez', 'Almería', 100),

INSERT INTO cliente VAĮUES('Adela', 'Salas', 'Díaz', 'Granada', 200), INSERT INTO cliente VAĮUES('Adolfo', 'Rubio', 'Flores', 'Sevilla', NUĮĮ), INSERT INTO cliente VAĮUES('Adrián', 'Suárez', NUĮĮ, 'Jaén', 300), INSERT INTO cliente VAĮUES('Marcos', 'Įoyola', 'Méndez', 'Almería', 200), INSERT INTO cliente VAĮUES('María', 'Santana', 'Moreno', 'Cádiz', 100), INSERT INTO cliente VAĮUES('Pilar', 'Ruiz', NUĮĮ, 'Sevilla', 300),

INSERT INTO cliente VAĮUES('Pepe', 'Ruiz', 'Santana', 'Huelva', 200), INSERT INTO cliente VAĮUES('Guillermo', 'Įópez', 'Gómez', 'Granada', 225), INSERT INTO cliente VAĮUES('Daniel', 'Santana', 'Įoyola', 'Sevilla', 125);

INSERT INTO comercial(nombre, apellido1, apellido2, comision) VAĮUES('Daniel', 'Sáez', 'Vega', 0.15),

('Juan', 'Gómez', 'Įópez', 0.13),

('Diego','Flores', 'Salas', 0.11),

('Marta','Herrera', 'Gil', 0.14),

('Antonio','Carretero', 'Ortega', 0.12),

('Manuel','Domínguez', 'Hernández', 0.13),

('Antonio','Vega', 'Hernández', 0.11),

('Alfredo','Ruiz', 'Flores', 0.05);

INSERT INTO pedido(total, fecha, id\_cliente, id\_comercial) VAĮUES(150.5, '2017-10-05', 5, 2),

(270.65, '2016-09-10', 1, 5),

(65.26, '2017-10-05', 2, 1),

(110.5, '2016-08-17', 8, 3),

(948.5, '2017-09-10', 5, 2),

(2400.6, '2016-07-27', 7, 1),

(5760, '2015-09-10', 2, 1),

(1983.43, '2017-10-10', 4, 6),

(2480.4, '2016-10-10', 8, 3),

(250.45, '2015-06-27', 8, 2),

(75.29, '2016-08-17', 3, 7),

(3045.6, '2017-04-25', 2, 1),

(545.75, '2019-01-25', 6, 1),

(145.82, '2017-02-02', 6, 1),

(370.85, '2019-03-11', 1, 5),

(2389.23, '2019-03-11', 1, 5);

* **Realice las siguientes consultas:**

# Consultas sobre una tabla

* 1. Devuelve un listado con todos los pedidos que se han realizado. Los pedidos deben estar ordenados por la fecha de realización, mostrando en primer lugar los pedidos más recientes.
  2. Devuelve todos los datos de los dos pedidos de mayor valor.
  3. Devuelve un listado con los identificadores de los clientes que han realizado algún pedido. Tenga en cuenta que no debe mostrar identificadores que estén repetidos.
  4. Devuelve un listado de todos los pedidos que se realizaron durante el año 2017, cuya cantidad total sea superior a 500.
  5. Devuelve un listado con el nombre y los apellidos de los comerciales que tienen una comisión entre 0.05 y 0.11.
  6. Devuelve el valor de la comisión de mayor valor que existe en la tabla comercial.
  7. Devuelve el identificador, nombre y primer apellido de aquellos clientes cuyo segundo apellido no es NULL. El listado deberá estar ordenado alfabéticamente por apellidos y nombre.
  8. Devuelve un listado de los nombres de los clientes que empiezan por A y terminan por n y también los nombres que empiezan por P. El listado deberá estar ordenado alfabéticamente.
  9. Devuelve un listado de los nombres de los clientes que no empiezan por A. El listado deberá estar ordenado alfabéticamente.
  10. Devuelve un listado con los nombres de los comerciales que terminan por el o o. Tenga en cuenta que se deberán eliminar los nombres repetidos.

# Consultas multitabla (Composición interna)

1. Devuelve un listado con el identificador, nombre y los apellidos de todos los clientes que han realizado algún pedido. El listado debe estar ordenado alfabéticamente y se deben eliminar los elementos repetidos.
2. Devuelve un listado que muestre todos los pedidos que ha realizado cada cliente. El resultado debe mostrar todos los datos de los pedidos y del cliente. El listado debe mostrar los datos de los clientes ordenados alfabéticamente.
3. Devuelve un listado que muestre todos los pedidos en los que ha participado un comercial. El resultado debe mostrar todos los datos de los pedidos y de los comerciales. El listado debe mostrar los datos de los comerciales ordenados alfabéticamente.
4. Devuelve un listado que muestre todos los clientes, con todos los pedidos que han realizado y con los datos de los comerciales asociados a cada pedido.
5. Devuelve un listado de todos los clientes que realizaron un pedido durante el año 2017, cuya cantidad esté entre 300 € y 1000 €.
6. Devuelve el nombre y los apellidos de todos los comerciales que ha participado en algún pedido realizado por María Santana Moreno.
7. Devuelve el nombre de todos los clientes que han realizado algún pedido con el comercial Daniel Sáez Vega.

# Consultas multitabla (Composición externa)

1. Devuelve un listado con todos los clientes junto con los datos de los pedidos que han realizado. Este listado también debe incluir los clientes que no han realizado ningún pedido. El listado debe estar ordenado alfabéticamente por el primer apellido, segundo apellido y nombre de los clientes.
2. Devuelve un listado con todos los comerciales junto con los datos de los pedidos que han realizado. Este listado también debe incluir los comerciales que no han realizado ningún pedido. El listado debe estar ordenado alfabéticamente por el primer apellido, segundo apellido y nombre de los comerciales.
3. Devuelve un listado que solamente muestre los clientes que no han realizado ningún pedido.
4. Devuelve un listado que solamente muestre los comerciales que no han realizado ningún pedido.
5. Devuelve un listado con los clientes que no han realizado ningún pedido y de los comerciales que no han participado en ningún pedido. Ordene el listado alfabéticamente por los apellidos y el nombre. En en listado deberá diferenciar de algún modo los clientes y los comerciales.

# Consultas resumen

1. Calcula la cantidad total que suman todos los pedidos que aparecen en la tabla pedido.
2. Calcula la cantidad media de todos los pedidos que aparecen en la tabla pedido.
3. Calcula el número total de comerciales distintos que aparecen en la tabla pedido.
4. Calcula el número total de clientes que aparecen en la tabla cliente.
5. Calcula cuál es la mayor cantidad que aparece en la tabla pedido.
6. Calcula cuál es la menor cantidad que aparece en la tabla pedido.
7. Calcula cuál es el valor máximo de categoría para cada una de las ciudades que aparece en la tabla cliente.
8. Calcula cuál es el máximo valor de los pedidos realizados durante el mismo día para cada uno de los clientes. Es decir, el mismo cliente puede haber realizado varios pedidos de diferentes cantidades el mismo día. Se pide que se calcule cuál es el pedido de máximo valor para cada uno de los días en los que un cliente ha realizado un pedido. Muestra el identificador del cliente, nombre, apellidos, la fecha y el valor de la cantidad.
9. Calcula cuál es el máximo valor de los pedidos realizados durante el mismo día para cada uno de los clientes, teniendo en cuenta que sólo queremos mostrar aquellos pedidos que superen la cantidad de 2000 €.
10. Calcula el máximo valor de los pedidos realizados para cada uno de los comerciales durante la fecha 2016-08-17. Muestra el identificador del comercial, nombre, apellidos y total.
11. Devuelve un listado con el identificador de cliente, nombre y apellidos y el número total de pedidos que ha realizado cada uno de clientes. Tenga en cuenta que pueden existir clientes que no han realizado ningún pedido. Estos clientes también deben aparecer en el listado indicando que el número de pedidos realizados es 0.
12. Devuelve un listado con el identificador de cliente, nombre y apellidos y el número total de pedidos que ha realizado cada uno de clientes durante el año 2017.
13. Devuelve un listado que muestre el identificador de cliente, nombre, primer apellido y el valor de la máxima cantidad del pedido realizado por cada uno de los clientes. El resultado debe mostrar aquellos clientes que no han realizado ningún pedido indicando que la máxima cantidad de sus pedidos realizados es 0. Puede hacer uso de la función IFNULL.
14. Devuelve cuál ha sido el pedido de máximo valor que se ha realizado cada año.
15. Devuelve el número total de pedidos que se han realizado cada año.