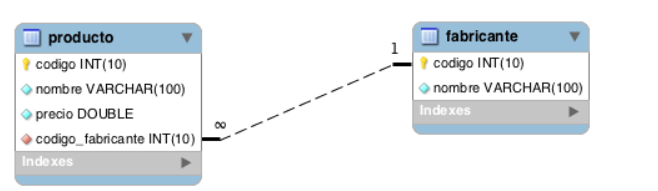
**Ejercicios de SQL**

****

**Base de datos para MySQL**

**DROP** **DATABASE** **IF** **EXISTS** tienda;

**CREATE** **DATABASE** tienda CHARACTER **SET** utf8mb4;

**USE** tienda;

**CREATE** **TABLE** fabricante (

**id** INT UNSIGNED AUTO\_INCREMENT **PRIMARY** **KEY**,

nombre VARCHAR(100) **NOT** **NULL**

);

**CREATE** **TABLE** producto (

**id** INT UNSIGNED AUTO\_INCREMENT **PRIMARY** **KEY**,

nombre VARCHAR(100) **NOT** **NULL**,

precio DOUBLE **NOT** **NULL**,

id\_fabricante INT UNSIGNED **NOT** **NULL**,

**FOREIGN** **KEY** (id\_fabricante) **REFERENCES** fabricante(**id**)

);

**INSERT** **INTO** fabricante **VALUES**(1, 'Asus');

**INSERT** **INTO** fabricante **VALUES**(2, 'Lenovo');

**INSERT** **INTO** fabricante **VALUES**(3, 'Hewlett-Packard');

**INSERT** **INTO** fabricante **VALUES**(4, 'Samsung');

**INSERT** **INTO** fabricante **VALUES**(5, 'Seagate');

**INSERT** **INTO** fabricante **VALUES**(6, 'Crucial');

**INSERT** **INTO** fabricante **VALUES**(7, 'Gigabyte');

**INSERT** **INTO** fabricante **VALUES**(8, 'Huawei');

**INSERT** **INTO** fabricante **VALUES**(9, 'Xiaomi');

**INSERT** **INTO** producto **VALUES**(1, 'Disco duro SATA3 1TB', 86.99, 5);

**INSERT** **INTO** producto **VALUES**(2, 'Memoria RAM DDR4 8GB', 120, 6);

**INSERT** **INTO** producto **VALUES**(3, 'Disco SSD 1 TB', 150.99, 4);

**INSERT** **INTO** producto **VALUES**(4, 'GeForce GTX 1050Ti', 185, 7);

**INSERT** **INTO** producto **VALUES**(5, 'GeForce GTX 1080 Xtreme', 755, 6);

**INSERT** **INTO** producto **VALUES**(6, 'Monitor 24 LED Full HD', 202, 1);

**INSERT** **INTO** producto **VALUES**(7, 'Monitor 27 LED Full HD', 245.99, 1);

**INSERT** **INTO** producto **VALUES**(8, 'Portátil Yoga 520', 559, 2);

**INSERT** **INTO** producto **VALUES**(9, 'Portátil Ideapd 320', 444, 2);

**INSERT** **INTO** producto **VALUES**(10, 'Impresora HP Deskjet 3720', 59.99, 3);

**INSERT** **INTO** producto **VALUES**(11, 'Impresora HP Laserjet Pro M26nw', 180, 3);

db.fabricante.insertMany([

{ \_id: 1, nombre: 'Asus' },

{ \_id: 2, nombre: 'Lenovo' },

{ \_id: 3, nombre: 'Hewlett-Packard' },

{ \_id: 4, nombre: 'Samsung' },

{ \_id: 5, nombre: 'Seagate' },

{ \_id: 6, nombre: 'Crucial' },

{ \_id: 7, nombre: 'Gigabyte' },

{ \_id: 8, nombre: 'Huawei' },

{ \_id: 9, nombre: 'Xiaomi' }

]);

db.producto.insertMany([

{ \_id: 1, nombre: 'Disco duro SATA3 1TB', precio: 86.99, fabricante\_id: 5 },

{ \_id: 2, nombre: 'Memoria RAM DDR4 8GB', precio: 120, fabricante\_id: 6 },

{ \_id: 3, nombre: 'Disco SSD 1 TB', precio: 150.99, fabricante\_id: 4 },

{ \_id: 4, nombre: 'GeForce GTX 1050Ti', precio: 185, fabricante\_id: 7 },

{ \_id: 5, nombre: 'GeForce GTX 1080 Xtreme', precio: 755, fabricante\_id: 6 },

{ \_id: 6, nombre: 'Monitor 24 LED Full HD', precio: 202, fabricante\_id: 1 },

{ \_id: 7, nombre: 'Monitor 27 LED Full HD', precio: 245.99, fabricante\_id: 1 },

{ \_id: 8, nombre: 'Portátil Yoga 520', precio: 559, fabricante\_id: 2 },

{ \_id: 9, nombre: 'Portátil Ideapd 320', precio: 444, fabricante\_id: 2 },

{ \_id: 10, nombre: 'Impresora HP Deskjet 3720', precio: 59.99, fabricante\_id: 3 },

{ \_id: 11, nombre: 'Impresora HP Laserjet Pro M26nw', precio: 180, fabricante\_id: 3 }

]);

**Consultas**

1. Lista el nombre de todos los productos que hay en la tabla producto.

**db.producto.find({}, {nombre:1, \_id:0})**

1. Lista los nombres y los precios de todos los productos de la tabla producto.

**db.producto.find({}, {nombre:1, precio:1, \_id:0})**

1. Lista todas las columnas de la tabla producto.

**db.producto.find()**

1. Lista el nombre de los productos, el precio en euros y el precio en dólares estadounidenses (USD).

**db.producto.find({}, {nombre:1, precio:1, \_id:0, precio\_dolar:{ $multiply:["$precio",1.08]}})**

**db.producto.find({}, {nombre:1, precio:1, \_id:0, precio\_dolar:{ $round: [{ $multiply:["$precio",1.08]}, 2]}})**

1. Lista el nombre de los productos, el precio en euros y el precio en dólares estadounidenses (USD). Utiliza los siguientes alias para las columnas: nombre de producto, euros, dólares. ).

**db.producto.find({}, {"Nombre de Producto":"$nombre", euro:"$precio", \_id:0, dolar:{ $multiply:["$precio",1.08]}})**

1. Lista los nombres y los precios de todos los productos de la tabla producto, convirtiendo los nombres a mayúscula.

**db.producto.find({}, {nombre: {$toUpper: "$nombre"}, precio:1, \_id:0})**

1. Lista los nombres y los precios de todos los productos de la tabla producto, convirtiendo los nombres a minúscula.

**db.producto.find({}, {nombre: {$toLower: "$nombre"}, precio:1, \_id:0})**

1. Lista el nombre de todos los fabricantes en una columna, y en otra columna obtenga en mayúsculas los dos primeros caracteres del nombre del fabricante.

**db.fabricante.find({},{\_id:0, nombre:1, primeroDosCaracteres:{ $toUpper: {$substr:["$nombre",0,2]}}})**

1. Lista los nombres y los precios de todos los productos de la tabla producto, redondeando el valor del precio.

**db.producto.find({}, {\_id:0, nombre:1, precio:{ $round: ["$precio"]}})**

1. Lista los nombres y los precios de todos los productos de la tabla producto, truncando el valor del precio para mostrarlo sin ninguna cifra decimal.

**db.producto.find({}, {\_id:0, nombre:1, precio:{ $trunc: ["$precio"]}})**

1. Lista el identificador de los fabricantes que tienen productos en la tabla producto.

**db.producto.find({},{\_id:0, fabricante\_id:1})**

1. Lista el identificador de los fabricantes que tienen productos en la tabla producto, eliminando los identificadores que aparecen repetidos.

**db.producto.distinct("fabricante\_id")**

1. Lista los nombres de los fabricantes ordenados de forma ascendente.

**db.fabricante.find({},{\_id:0, nombre:1}).sort({nombre:1})**

1. Lista los nombres de los fabricantes ordenados de forma descendente.

**db.fabricante.find({},{\_id:0, nombre:1}).sort({nombre:-1})**

1. Lista los nombres de los productos ordenados en primer lugar por el nombre de forma ascendente y en segundo lugar por el precio de forma descendente.

**db.producto.find({}, {\_id:0, nombre:1, precio:1}).sort({nombre: 1, precio: -1})**

1. Devuelve una lista con las 5 primeras filas de la tabla fabricante.

**db.fabricante.find({}).limit(5)**

1. Devuelve una lista con 2 filas a partir de la cuarta fila de la tabla fabricante. La cuarta fila también se debe incluir en la respuesta.

**db.fabricante.find({}).skip(3).limit(2)**

1. Lista el nombre y el precio del producto más barato. (Utilice solamente las cláusulas ORDER BY y LIMIT)

**db.producto.find({}, {\_id:0, nombre:1, precio:1}).sort({precio: 1}).limit(1)**

1. Lista el nombre y el precio del producto más caro. (Utilice solamente las cláusulas ORDER BY y LIMIT)

**db.producto.find({}, {\_id:0, nombre:1, precio:1}).sort({precio: -1}).limit(1)**

1. Lista el nombre de todos los productos del fabricante cuyo identificador de fabricante es igual a 2.

**db.producto.find({fabricante\_id:2})**

1. Lista el nombre de los productos que tienen un precio menor o igual a 120€.

**db.producto.find({precio:{ "$lte":120}},{nombre:1})**

1. Lista el nombre de los productos que tienen un precio mayor o igual a 400€.

**db.producto.find({precio:{ "$gte":400}},{nombre:1})**

1. Lista el nombre de los productos que **no tienen** un precio mayor o igual a 400€.

**db.producto.find({precio:{ "$lt":400}},{nombre:1})**

**db.producto.find({precio:{$not:{ "$gte":400}}},{nombre:1})**

1. Lista todos los productos que tengan un precio entre 80€ y 300€. Sin utilizar el operador BETWEEN.

**db.producto.find({precio:{ "$gte":80, "$lte":300}})**

1. Lista todos los productos que tengan un precio entre 60€ y 200€. Utilizando el operador BETWEEN.

**db.producto.find({precio:{ "$gte":60, "$lte":200}})**

1. Lista todos los productos que tengan un precio mayor que 200€ y que el identificador de fabricante sea igual a 6.

**db.producto.find({precio:{ "$gt":200}, fabricante\_id:6})**

1. Lista todos los productos donde el identificador de fabricante sea 1, 3 o 5. Sin utilizar el operador IN.

**db.producto.find({"$or":[{fabricante\_id:1},{fabricante\_id:3},{fabricante\_id:5}]})**

1. Lista todos los productos donde el identificador de fabricante sea 1, 3 o 5. Utilizando el operador IN.

**db.producto.find({fabricante\_id:{"$in":[1,3,5]}})**

1. Lista el nombre y el precio de los productos en céntimos (Habrá que multiplicar por 100 el valor del precio). Cree un alias para la columna que contiene el precio que se llame céntimos.

**db.producto.find({}, {\_id:0, nombre:1, precio:1, centimos:{ $multiply:["$precio",100]}})**

1. Lista los nombres de los fabricantes cuyo nombre empiece por la letra S.

**db.fabricante.find({nombre:/^s/i})**

1. Lista los nombres de los fabricantes cuyo nombre termine por la vocal e.

**db.fabricante.find({nombre:/e$/i})**

1. Lista los nombres de los fabricantes cuyo nombre contenga el carácter w.

**db.fabricante.find({nombre:/w/i})**

1. Lista los nombres de los fabricantes cuyo nombre sea de 4 caracteres.

**db.fabricante.find({nombre:/^.{4}$/})**

1. Devuelve una lista con el nombre de todos los productos que contienen la cadena Portátil en el nombre.

**db.producto.find({nombre:/Portátil/})**

1. Devuelve una lista con el nombre de todos los productos que contienen la cadena Monitor en el nombre y tienen un precio inferior a 215 €.

**db.producto.find({nombre:/Monitor/, precio:{$lt: 215}}, {\_id:0, nombre:1})**

1. Lista el nombre y el precio 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).

**db.producto.find( { precio: { $gte: 180 } }, { nombre: 1, precio: 1, \_id: 0 } ).sort({ precio: -1, nombre: 1 })**

**Agregación**

1. Devuelve una lista con el nombre del producto, precio y nombre de fabricante de todos los productos de la base de datos.

**Referencia a sql select \*from productos infofab  
left join fabricante f on infofab.fabricante\_id = f.\_id**

|  |
| --- |
| db.getCollection(**'producto'**).aggregate( |
| [ |
| { |
| $lookup: { |
| as: **'infofab'**, |
| from: **'fabricante'**, |
| foreignField: **'\_id'**, |
| localField: **'fabricante\_id'** |
| } |
| }, |
| { |
| $project: { |
| \_id: 0, |
| nombre: 1, |
| precio: 1, |
| **'Nombre de Fabricante'**: { |
| $arrayElemAt: [**'$infofab.nombre'**, 0] |
| } |
| } |
| } |
| ], |
| ); |

1. Devuelve una lista con el nombre del producto, precio y nombre de fabricante de todos los productos de la base de datos. Ordene el resultado por el nombre del fabricante, por orden alfabético.

|  |
| --- |
| db.getCollection(**'producto'**).aggregate( |
| [ |
| { |
| $lookup: { |
| as: **'infofab'**, |
| from: **'fabricante'**, |
| foreignField: **'\_id'**, |
| localField: **'fabricante\_id'** |
| } |
| }, |
| { |
| $project: { |
| \_id: 0, |
| nombre: 1, |
| precio: 1, |
| **'Nombre de Fabricante'**: { |
| $arrayElemAt: [**'$infofab.nombre'**, 0] |
| } |
| } |
| }, |
| { $sort: { **'Nombre de Fabricante'**: 1 } } |
| ], |
| { maxTimeMS: 60000, allowDiskUse: true } |
| ); |

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

|  |
| --- |
| db.getCollection(**'producto'**).aggregate( |
| [ |
| { |
| $lookup: { |
| as: **'infofab'**, |
| from: **'fabricante'**, |
| foreignField: **'\_id'**, |
| localField: **'fabricante\_id'** |
| } |
| }, |
| { |
| $match: { |
| **'infofab.nombre'**: RegExp(**'Lenovo'**, **'i'**) |
| } |
| }, |
| { |
| $project: { |
| \_id: 0, |
| nombre: 1, |
| precio: 1, |
| **'Nombre de Fabricante'**: { |
| $arrayElemAt: [**'$infofab.nombre'**, 0] |
| } |
| } |
| } |
| ], |
|  |
| ); |

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

|  |
| --- |
| db.getCollection(**'producto'**).aggregate( |
| [ |
| { |
| $lookup: { |
| as: **'infofab'**, |
| from: **'fabricante'**, |
| foreignField: **'\_id'**, |
| localField: **'fabricante\_id'** |
| } |
| }, |
| { |
| $match: { |
| **'infofab.nombre'**: RegExp(**'e$'**, **'i'**) |
| } |
| }, |
| { |
| $project: { |
| \_id: 0, |
| nombre: 1, |
| precio: 1, |
| **'Nombre de Fabricante'**: { |
| $arrayElemAt: [**'$infofab.nombre'**, 0] |
| } |
| } |
| } |
| ], |
|  |
| ); |

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.

|  |
| --- |
| db.getCollection(**'producto'**).aggregate( |
| [ |
| { |
| $lookup: { |
| as: **'infofab'**, |
| from: **'fabricante'**, |
| foreignField: **'\_id'**, |
| localField: **'fabricante\_id'** |
| } |
| }, |
| { |
| $match: { |
| **'infofab.nombre'**: RegExp(**'w'**, **'i'**) |
| } |
| }, |
| { |
| $project: { |
| \_id: 0, |
| nombre: 1, |
| precio: 1, |
| **'Nombre de Fabricante'**: { |
| $arrayElemAt: [**'$infofab.nombre'**, 0] |
| } |
| } |
| } |
| ], |
|  |
| ); |

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)

|  |
| --- |
| db.getCollection(**'producto'**).aggregate( |
| [ |
| { |
| $lookup: { |
| as: **'infofab'**, |
| from: **'fabricante'**, |
| foreignField: **'\_id'**, |
| localField: **'fabricante\_id'** |
| } |
| }, |
| { $match: { precio: { $gte: 180 } } }, |
| { $sort: { precio: -1, nombre: 1 } }, |
| { |
| $project: { |
| \_id: 0, |
| nombre: 1, |
| precio: 1, |
| **'Nombre de Fabricante'**: { |
| $arrayElemAt: [**'$infofab.nombre'**, 0] |
| } |
| } |
| } |
| ], |
| { maxTimeMS: 60000, allowDiskUse: true } |
| ); |

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.

|  |
| --- |
| db.getCollection(**'producto'**).aggregate( |
| [ |
| { $group: { \_id: **'$fabricante\_id'** } }, |
| { |
| $lookup: { |
| as: **'infofab'**, |
| from: **'fabricante'**, |
| foreignField: **'\_id'**, |
| localField: **'\_id'** |
| } |
| }, |
| { |
| $project: { |
|  |
| **'Nombre de Fabricante'**: { |
| $arrayElemAt: [**'$infofab.nombre'**, 0] |
| } |
| } |
| } |
| ], |
| { maxTimeMS: 60000, allowDiskUse: true } |
| ); |