

INSTITUTO TECNOLÓGICO
SUPERIOR DE COMALCALCO

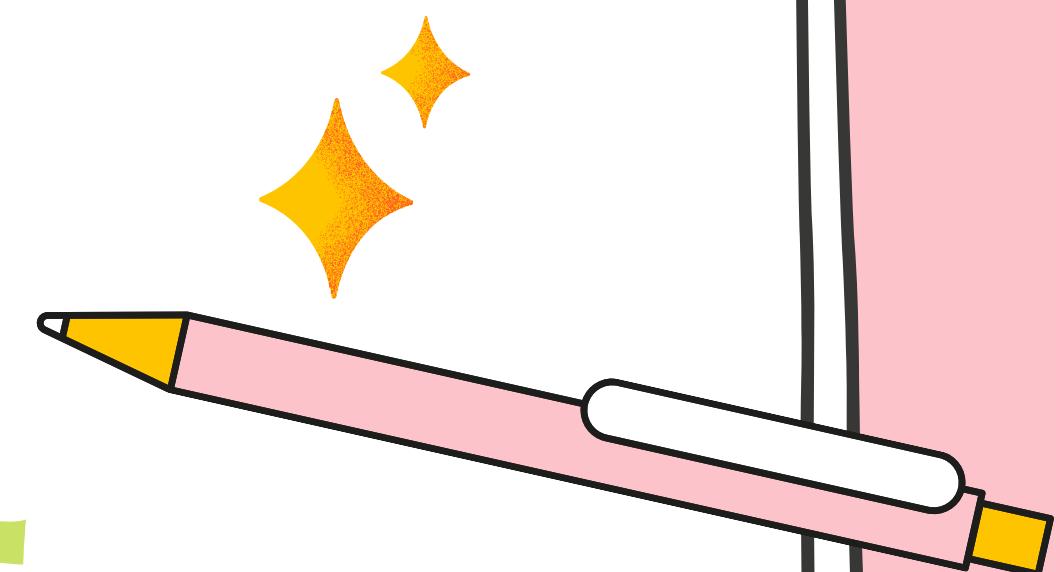
TALLER DE BASE DE DATOS

JOSE ARMANDO ARIAS

ARIAS

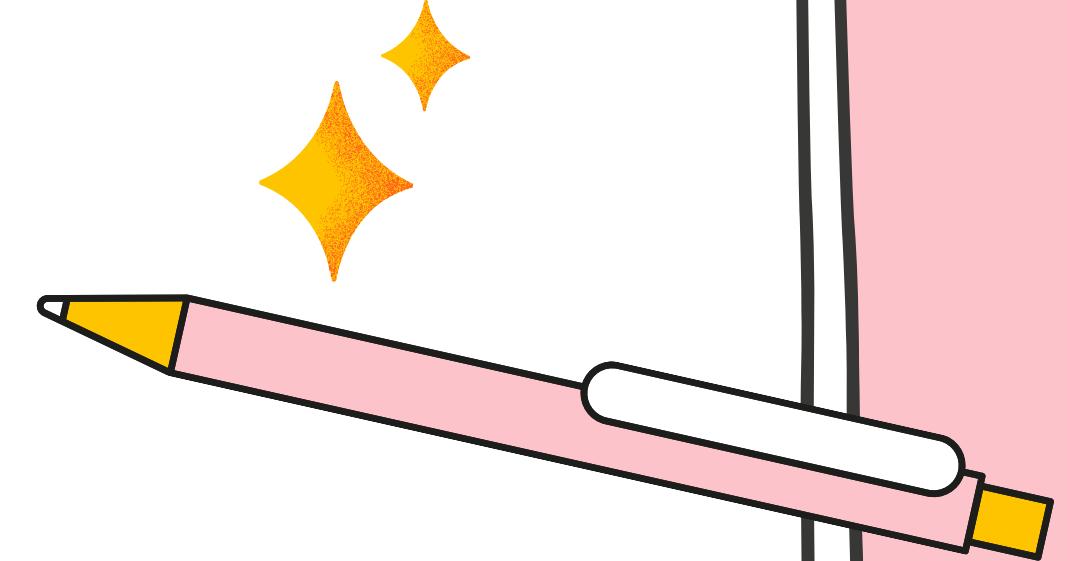
18 / 10 / 2022

SENTENCIAS SQL



SENTENCIAS DE DEFINICION Y MANIPULACION DE DATOS

JOSE ARMANDO ARIAS ARIAS



INTRODUCCIÓN

En la siguiente presentación se mostraran las sentencias de definición y manipulación de datos, estas vistas en clase por la profesora. estas sentencias tienen un fin, que es el ayudarnos y facilitarnos el trabajo cuando trabajamos en una base de datos.

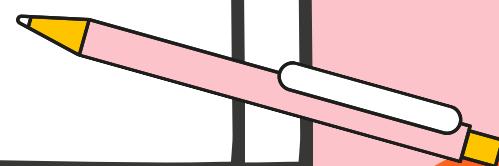
También nos ayudaran a manipular los datos ingresados dependiendo de cada necesidad que se nos presente, puede ser desde insertar datos en las tablas, mostrar los datos contenidos, tambien puede ser el actualizarlos, modificarlos y hasta la eliminación de los mismos.

Todo esto dentro de WampServer

SENTENCIAS DE DEFINICIÓN DE DATOS(DDL)



Las sentencias DDL se utilizan para describir una base de datos, para definir su estructura, para crear sus objetos y para crear los subobjetos de la tabla. Puede realizar cambios en un conjunto de reglas después de crearlo. es un lenguaje proporcionado por el sistema de gestión de base de datos que permite a los usuarios de la misma llevar a cabo las tareas de definición de las estructuras que almacenarán los datos así como de los procedimientos o funciones que permitan consultarlos.



CREATE TABLE

```
CREATE TABLE Libro[  
isbn INT NOT NULL,  
titulo VARCHAR(40),  
PRIMARY KEY(isbn)];
```

```
DESCRIBE Libro;
```

Field	Type	Null	Key	Default	Extra
isbn	int(11)	NO	PRI	NULL	
titulo	varchar(40)	YES		NULL	

1.

Se utiliza para crear una nueva tabla, donde la información se almacena realmente.

2.

SINTAXIS

```
CREATE TABLE table_name (  
column1_name datatype, column2_name  
datatype, column3_name, PRIMARY KEY,  
FOREIGN KEY);
```

ALTER TABLE

```
ALTER TABLE zona ADD control INT;
```

```
DESCRIBE zona;
```

Field	Type	Null	Key	Default
zona	int(11)	NO	PRI	NULL
descripcion	varchar(30)	YES		NULL
control	int(11)	YES		NULL

Le permite cambiar la estructura de una tabla existente. Por ejemplo, puede añadir o borrar columnas, crear o destruir índices, cambiar el tipo de columnas existentes

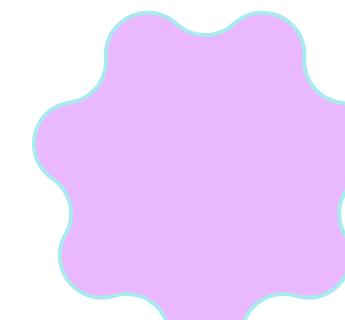
1.

2.

SINTAXIS

**ALTER TABLE name_table ADD,DROP
column datatype;**

CONSTRAINT



1. *Definir un campo como llave primaria*
2. *Para definir un campo como llave foránea,*
3. *Para definir una restricción de contenido de campo*

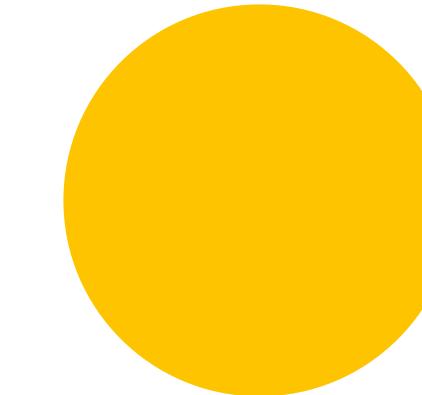
SINTAXIS

2. *ALTER TABLE name_table ADD CONSTRAINT
name_CT PRIMARY KEY(name_PK);*
2. *ALTER TABLE name_table ADD CONSTRAINT
name_CT FOREIGN KEY(name_FK) REFERENCES
name_table(name_PK);*
3. *ALTER TABLE name_table ADD CONSTRAINT
nam_CT CHECK(colum_name IN('content',
'content',));*

```
ALTER TABLE Cuentas ADD CONSTRAINT fk_cuentas  
FOREIGN KEY(clave_sucursal) REFERENCES Sucursal(clave_sucursal);
```

```
ALTER TABLE zona ADD CONSTRAINT ck_zona CHECK(descripcion IN  
('Centro', 'Sur', 'Este', 'Oeste'));
```

DROP



```
DROP TABLE nombredecolumna;
```



La sentencia para eliminar cualquier objetos de la base de datos, por ejemplo tablas, vistas, disparadores, o incluso la propia base de datos



SINTAXIS

DROP TABLE name_table;

AL IGUAL EXISTEN

1.

El comando SHOW DATABASES permite visualizar las bases de datos actualmente activas.

El comando USE nos permite utilizar una base de datos. Es (junto con quit) el único comando que no requiere punto y coma.

El comando DESCRIBE seguido del nombre de una tabla, nos permite ver la estructura completa de una tabla.

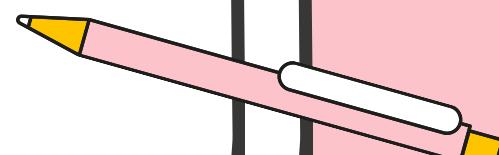
El comando SHOW TABLES muestra las tablas de la base de datos actual.

mysql> describe personas;					
Field	Type	Null	Key	Default	Extra
nombre	varchar(30)	YES		NULL	
apellido1	varchar(30)	YES		NULL	
apellido2	varchar(30)	YES		NULL	
telefono	varchar(9)	YES		NULL	

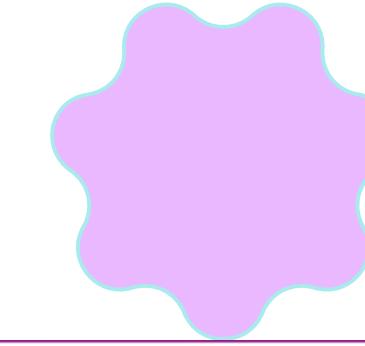
LENGUAJE DE MANIPULACIÓN DE DATOS (DML)



(Lenguaje de Modificación de Datos) es una de las partes fundamentales del lenguaje SQL. Lo forman las instrucciones capaces de modificar (añadir, cambiar o eliminar) los datos de las tablas. Al conjunto de instrucciones DML que se ejecutan consecutivamente, se le llama transacción. a llevar a cabo las tareas de consulta o modificación de los datos contenidos en las Bases de Datos del Sistema Gestor de Bases de Datos.



INSERT



```
INSERT INTO articulos VALUES(1, 'Disco duro Sata3 1TB', 86, 5),  
(2, 'Memoria RAM DDR4 8GB', 120, 6),  
(3, 'Disco SSD 1TB', 151, 4),  
(4, 'Gforce GTX 1050Ti', 185, 7),  
(5, 'Gforce GTX 1080 Xtreme', 755, 6),  
(6, 'Monitor 24 LED Full HD', 202, 1),  
(7, 'Monitor 27 LED Full HD', 246, 1),  
(8, 'Portátil Yoga 520', 559, 2),  
(9, 'Portátil Ideapd 320', 444, 2),  
(10, 'Impresora HP Deskjet 3720', 60, 3),  
(11, 'Impresora HP Laserjet Pro M26nw', 180, 3);
```

1.

Esta sentencia se utiliza para insertar nuevas filas de datos en una tabla de la base de datos

2.

SINTAXIS

INSERT INTO nombre_tabla (campo1, campo2,...,campoN) VALUES (valor1,valor,2..., valorN)

SELECT

```
SELECT nombre FROM articulos;
+-----+
| nombre
+-----+
| Disco duro Sata3 1TB
| Memoria RAM DDR4 8GB
| Disco SSD 1TB
| Gforce GTX 1050Ti
| Gforce GTX 1080 Xtreme
| Monitor 24 LED Full HD
| Monitor 27 LED Full HD
| Portátil Yoga 520
| Portátil Ideapd 320
| Impresora HP Deskjet 3720
| Impresora HP Laserjet Pro M26n
+-----+
```

1.

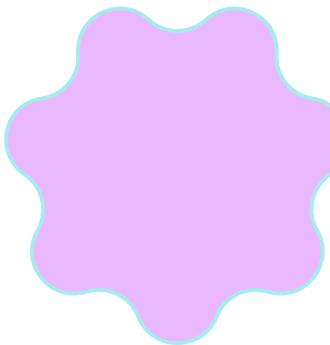
Sentencia que recupera datos de una o más tablas.

2.

SELECT nombre_campo(s) FROM nombre_tabla;

SINTAXIS

UPDATE



1.

Sentencia que se utiliza cuando se desea cambiar un gran número de registros o cuando estos se encuentran en múltiples tablas.

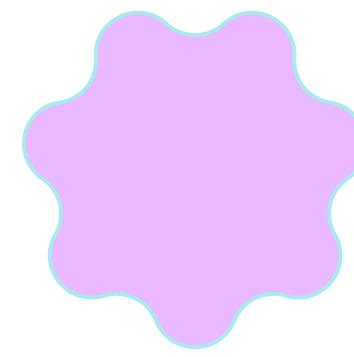
SINTAXIS

2.

*UPDATE nombre_tabla SET
nombre_campo=valor que se desea
WHERE (condición);*

UPDATE articulos SET precio=(precio+20) WHERE Clave_fabricante=1;			
Clave_articulo	Nombre	Precio	Clave_fabricante
1	Disco duro SATA3 1TB	96	5
2	Memoria RAM DDR4 8GB	130	6
3	Disco SSD 1TB	161	4
4	Gforce GTX 1050Ti	195	7
5	Gforce GTX 1080 Xtreme	765	6
6	Monitor 24 LED Full HD	232	1
7	Monitor 27 LED Full HD	276	1

DELETE



```
DELETE FROM articulos WHERE Clave_fabricante=3;
+-----+-----+-----+
| Clave_articulo | Nombre           | Precio | Clave_fabricante |
+-----+-----+-----+
| 1 | Disco duro SATA3 1TB | 96 | 5 |
| 2 | Memoria RAM DDR4 8GB | 130 | 6 |
| 3 | Disco SSD 1TB       | 161 | 4 |
| 4 | Gforce GTX 1050Ti   | 195 | 7 |
| 5 | Gforce GTX 1080 Xtreme | 765 | 6 |
| 6 | Monitor 24 LED Full HD | 232 | 1 |
| 7 | Monitor 27 LED Full HD | 276 | 1 |
| 8 | Portátil Yoga 520     | 569 | 2 |
| 9 | Portátil Ideapd 320     | 454 | 2 |
```



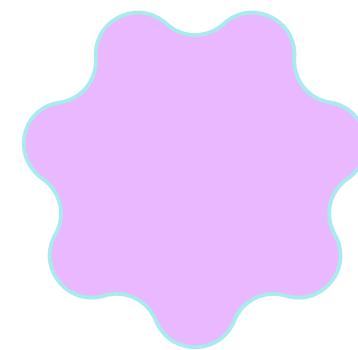
Una sentencia DELETE de SQL borra uno o más registros existentes en una tabla.

SINTAXIS



*DELETE FROM name_table WHERE
(condición);*

DELETE



```
DELETE FROM articulos WHERE Clave_fabricante=3;
+-----+-----+-----+
| Clave_articulo | Nombre           | Precio | Clave_fabricante |
+-----+-----+-----+
| 1 | Disco duro SATA3 1TB | 96 | 5 |
| 2 | Memoria RAM DDR4 8GB | 130 | 6 |
| 3 | Disco SSD 1TB       | 161 | 4 |
| 4 | Gforce GTX 1050Ti   | 195 | 7 |
| 5 | Gforce GTX 1080 Xtreme | 765 | 6 |
| 6 | Monitor 24 LED Full HD | 232 | 1 |
| 7 | Monitor 27 LED Full HD | 276 | 1 |
| 8 | Portátil Yoga 520    | 569 | 2 |
| 9 | Portátil Ideapd 320   | 454 | 2 |
```



Una sentencia DELETE de SQL borra uno o más registros existentes en una tabla.

SINTAXIS

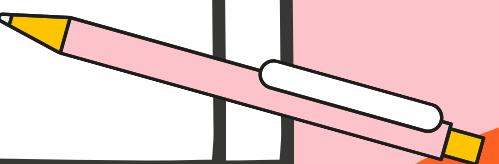


*DELETE FROM name_table WHERE
(condición);*



CLAUSULAS

Son condiciones de modificación que se utilizan para definir datos que se desean seleccionar o manipular.



FROM

```
SELECT * FROM fabricantes;
```

	Clave_fabricante	Nombre
1	Asus	
2	Lenovo	
3	Hewlett-Packard	
4	Samsung	
5	Seagate	
6	Crucial	
7	Gigabyte	
8	Huawei	
9	Xiaomi	

1.

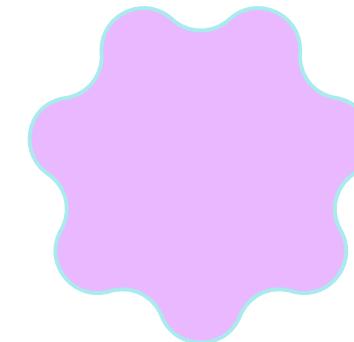
*Sentencia que se utiliza para especificar de que tabla se trabajarán los datos. Si se quiere trabajar con todos los campos de la tabla utilizarán el símbolo * después del SELECT*

SINTAXIS

2.

*SELECT *, campo(s) FROM name_table;*

WHERE



```
SELECT nombre, Clave_fabricante FROM articulos  
WHERE Clave_fabricante=2;  
+-----+-----+  
| nombre | Clave_fabricante |  
+-----+-----+  
| Portátil Yoga 520 | 2 |  
| Portátil Ideapd 320 | 2 |  
+-----+-----+
```



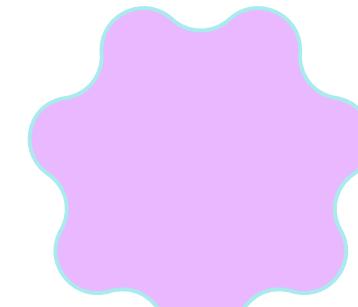
Especifica criterios que tienen que cumplir los valores de campo para que los registros que contienen los valores se incluyan en los resultados de la consulta

SINTAXIS



*SELECT nombre_campo(s) FROM
nombre_tabla WHERE campo = criterio;*

GROUP BY



1.

La función GROUP BY se utiliza para juntar filas de resultados que coincidan en el valor de alguna columna seleccionada.

```
SELECT nombre,telefono,ciudad from cliente  
GROUP BY ciudad,nombre;
```

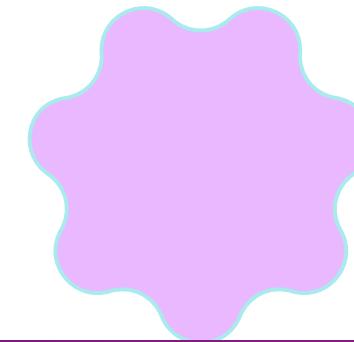
nombre	telefono	ciudad
Alicia Rodriguez Salas	5412142246	Mendoza
Martha Ramirez Luna	5468903524	Mendoza
Pablo Mirando Romero	5411132145	Mendoza
Ricardo Mendoza Jimenez	5367899099	Mendoza
Ana Gutierrez Palacio	5500687855	Valladolid
Cecilia Lopez Hernandez	5567893423	Valladolid
Esteban Morales Sanchez	5510143298	Valladolid
Jesus Hernandez Gomez	5599677754	Valladolid

SINTAXIS

2.

```
SELECT *, campo(s) FROM name_table  
GROUP BY campo(s);
```

HAVING



1. La cláusula **HAVING** filtra los resultados obtenidos por la cláusula **GROUP BY**

```
SELECT nombre,telefono,ciudad from cliente  
-> GROUP BY ciudad,nombre  
-> HAVING nombre LIKE('A%');  
+-----+-----+-----+  
| nombre | telefono | ciudad |  
+-----+-----+-----+  
| Alicia Rodriguez Salas | 5412142246 | Mendoza |  
| Andy Ramirez Luna | 54689403524 | Mendoza |  
| Ana Gutierrez Palacio | 5500687855 | Valladolid |  
+-----+-----+-----+
```

SINTAXIS

2.

*SELECT *, campo(s) FROM name_table
GROUP BY campo(s) HAVING campo
condición;*

ORDER BY

1.

La cláusula ORDER BY se utiliza para ordenar los registros seleccionados por una consulta SQL.

```
SELECT nombre FROM fabricantes ORDER BY nombre DESC;  
+-----+  
| nombre |  
+-----+  
| Xiaomi |  
| Seagate |  
| Samsung |  
| LENOVO |  
| Huawei |  
| Hewlett-Packard |  
| Gigabyte |  
| Crucial |  
| ASUS |  
+-----+
```

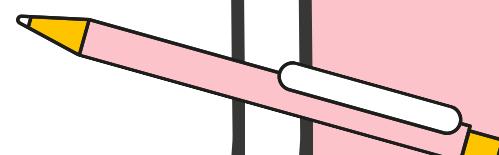
SINTAXIS

2.

ORDER BY campo(s) ASC/DESC;

FUNCIONES DE AGREGADO

Se usa dentro de una cláusula SELECT para devolver un único valor que se aplica a un grupo de registros.



AVG

1. *La función AVG devuelve el valor medio de una columna de tipo numérico*

```
SELECT AVG(cantidad) Promedio FROM pedido;
```

```
+-----+  
| Promedio |  
+-----+  
| 1312.051874999999 |  
+-----+
```

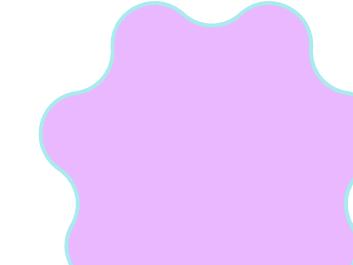
SINTAXIS

2.
SELECT AVG(columna) FROM table;

COUNT

1.

La función COUNT devuelve el número de filas de la consulta, es decir, el número de registros que cumplen una determinada condición.



```
SELECT COUNT(nombre) Total_Clientes FROM cliente;  
+-----+  
| Total_Clientes |  
+-----+  
|          10 |  
+-----+
```

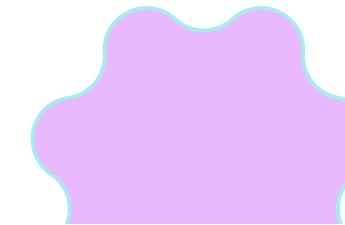
2.

SINTAXIS

SELECT COUNT(columna) FROM tabla;

SUM

1. *La función SUM permite obtener la suma total de los valores de una columna de tipo numérico.*



```
SELECT SUM(cantidad) TotalPedidos  
FROM pedido;  
+-----+  
| TotalPedidos |  
+-----+  
| 20992.829999999998 |  
+-----+
```

SINTAXIS

2.
SELECT SUM(columna) FROM table;

MAX

1.

La función MAX sirve para obtener el mayor valor para una columna determinada.



```
SELECT MAX(cantidad)  
FROM pedido;
```

MAX(cantidad)
5760

SINTAXIS

2.

SELECT MAX(columna) FROM tabla

MIN

1.

La función MIN sirve para obtener el valor más pequeño para una columna determinada.

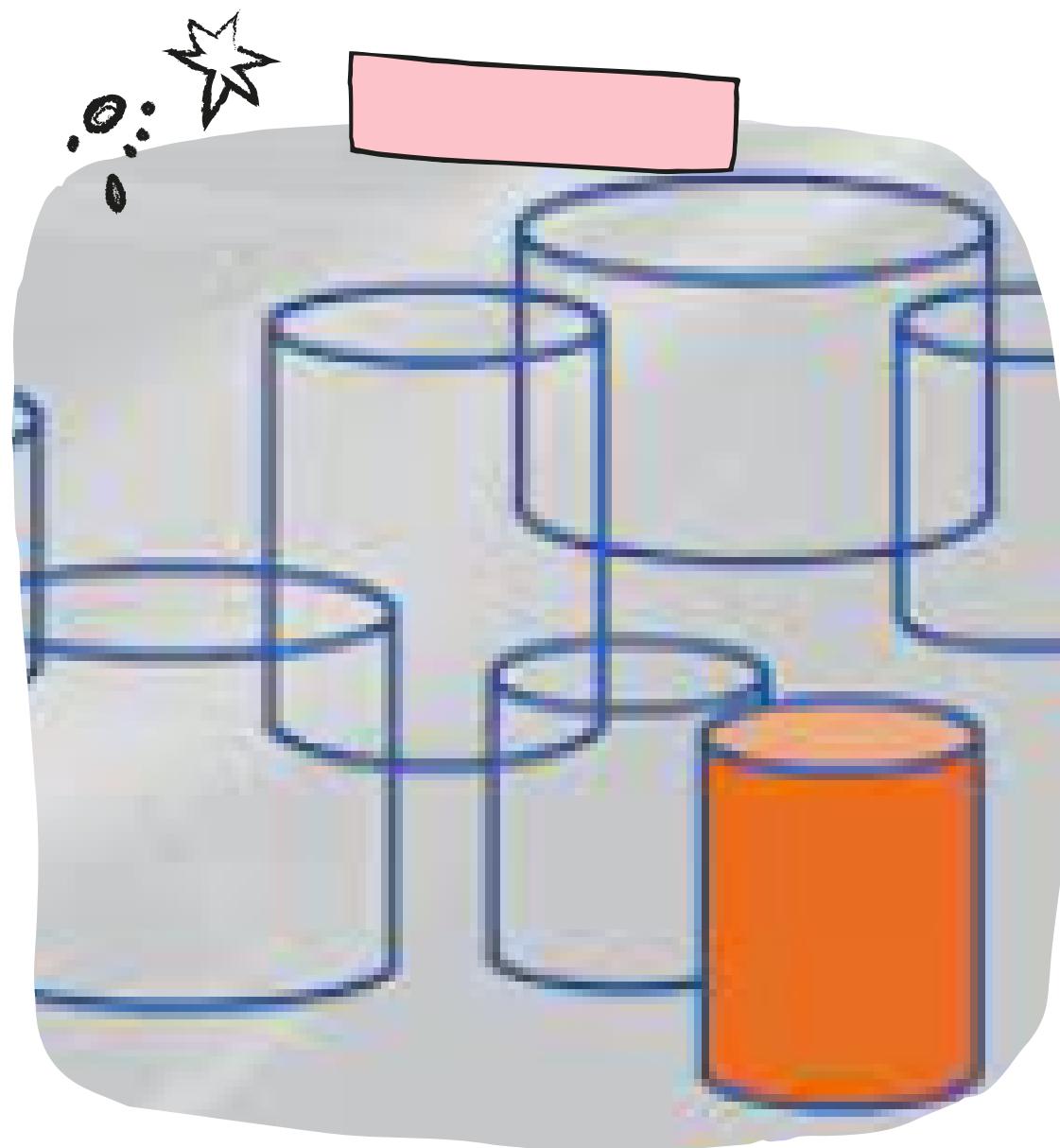
```
SELECT MIN(cantidad)  
FROM pedido;
```

MIN(cantidad)
65.26

SINTAXIS

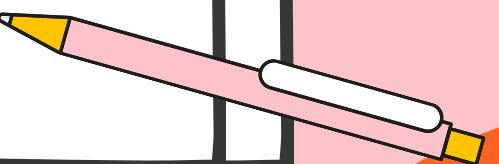
2.

```
SELECT MIN(columna) FROM tabla
```



CONSULTAS DE PREDICADO

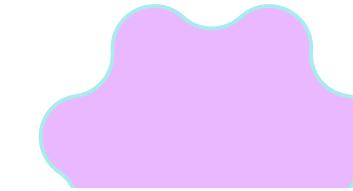
El predicado se incluye entre la clausula y el primer nombre del campo a recuperar



ALL(*)



Indica al SELECT que se deben seleccionar todos los campos de la tabla en cuestión



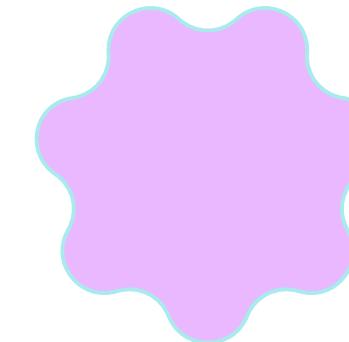
SELECT * FROM fabricantes;	
Clave_fabricante	Nombre
1	Asus
2	Lenovo
3	Hewlett-Packard
4	Samsung
5	Seagate
6	Crucial
7	Gigabyte
8	Huawei
9	Xiaomi

SINTAXIS



*SELECT * FROM table;*

TOP(LIM IT)



1.

su funcionalidad es la de limitar el número de filas (registros/resultados) devueltas en las consultas SELECT.

SINTAXIS

2.

*SELECT campo1, campo2... FROM tabla1,
tabla2... ORDER BY campo ASC/DESC
[LIMIT Longitud];*

```
SELECT * FROM pedido ORDER BY cantidad DESC LIMIT 2;  
+-----+-----+-----+  
| id3 | Cantidad | Fecha      | id_cliente | id_comercial |  
+-----+-----+-----+  
| 7   |    5760  | 2015-09-10 |          2 |           11 |  
| 12  |  3045.6  | 2017-04-25 |          2 |           11 |  
+-----+-----+-----+
```

DISTINC

1.

La cláusula MySQL DISTINCT se utiliza para eliminar duplicados del conjunto de resultados. La cláusula DISTINCT sólo se puede utilizar con sentencias SELECT.

```
SELECT DISTINCT nombre FROM comercial
WHERE nombre LIKE('%el')
OR nombre LIKE('%o');
+-----+
| nombre |
+-----+
| Daniel |
| Diego  |
| Antonio |
| Manuel |
| Alfredo|
+-----+
```

2.

*SELECT DISTINCT campo FROM tablas
[WHERE condiciones];*

SINTAXIS

LIKE

1.

Se utiliza para comparar una expresión de cadena de caracteres con un modelo



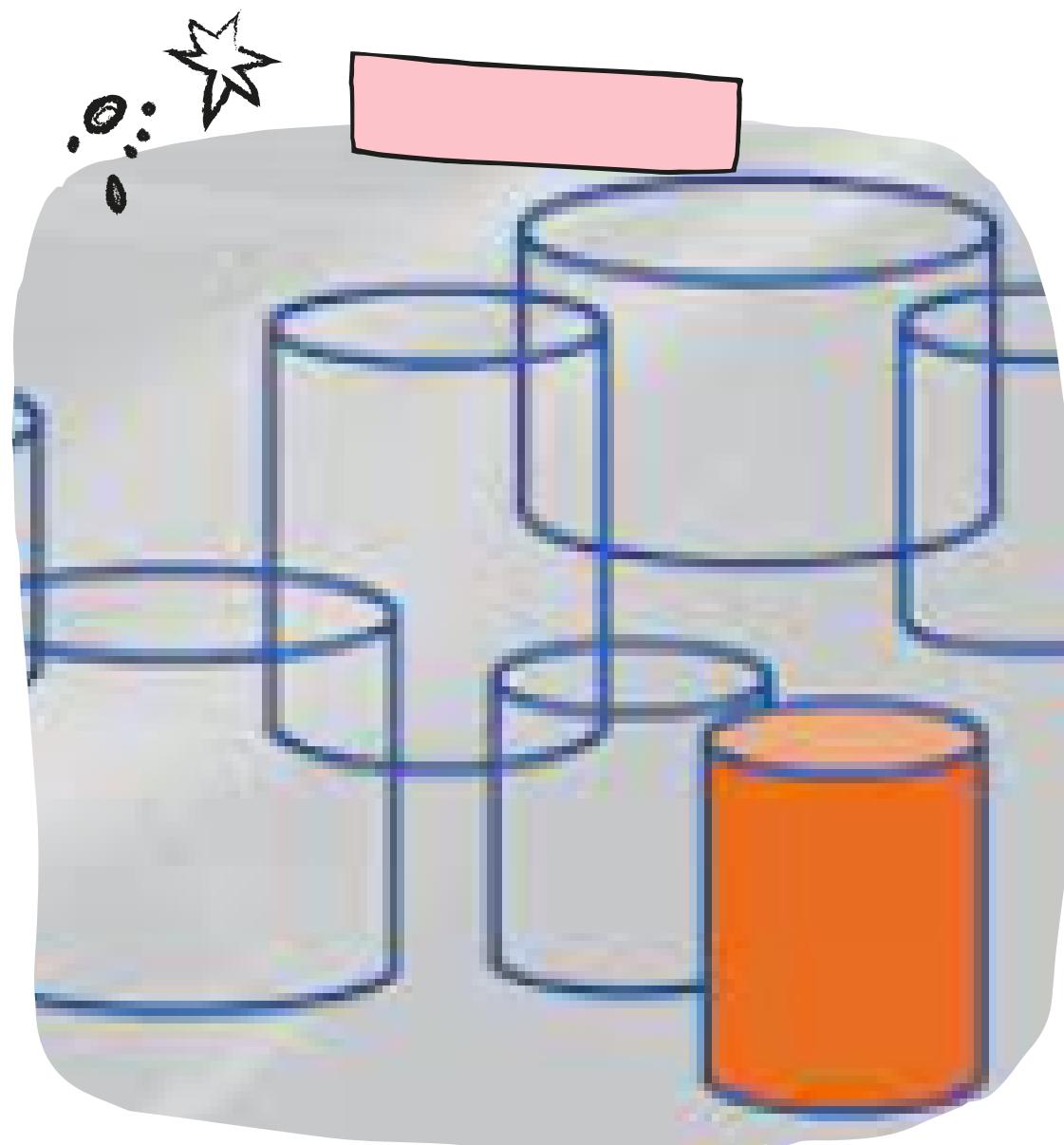
```
SELECT DISTINCT nombre FROM comercial  
WHERE nombre LIKE('%el')  
OR nombre LIKE('%o');  
+-----+  
| nombre |  
+-----+  
| Daniel |  
| Diego |  
| Antonio |  
| Manuel |  
| Alfredo |  
+-----+
```

SINTAXIS

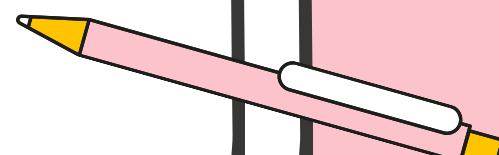
2.

SELECT campo(s) FROM tabla(s) WHERE campo LIKE('%condicion')

COMPOSICIÓN INTERNA: NATURAL JOIN



Esta cláusula busca coincidencia entre 2 tablas, en función a una columna que tienen en común. De tal modo que sólo la intersección se mostrará en los resultados.



INNER JOIN

1.

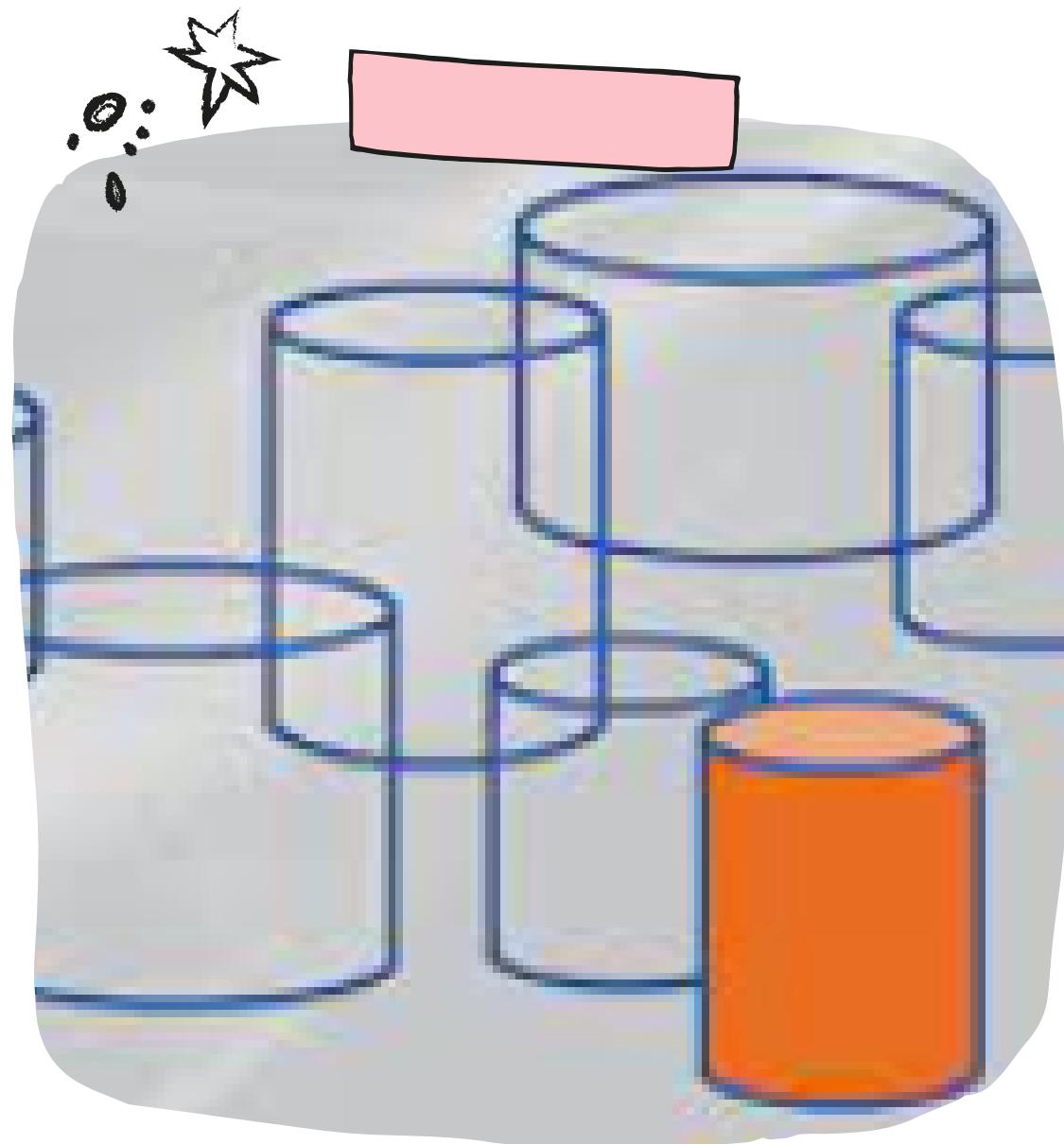
Esta cláusula busca coincidencia entre 2 tablas, en función a una columna que tienen en común. De tal modo que sólo la intersección se mostrará en los resultados.

Nombre	DepartmentId	Id	Nombre
Rafferty	31	31	Sales
Jones	33	33	Engineering
Heisenberg	33	33	Engineering
Robinson	34	34	Clerical
Smith	34	34	Clerical

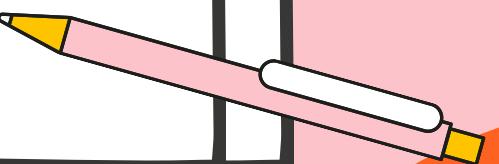
SINTAXIS

2.

*SELECT * FROM table_a JOIN table_b ON
table_b.FK = table_a.PK;*



**COMPOSICIÓN
EXTERNA: LEFT
JOIN, RIGHT JOIN**



LEFT JOIN

Empleado	Departamento
Rafferty	Sales
Jones	Engineering
Heisenberg	Engineering
Robinson	Clerical
Smith	Clerical
Williams	NULL

1. A diferencia de un *INNER JOIN*, donde se busca una intersección respetada por ambas tablas, con *LEFT JOIN* damos prioridad a la tabla de la izquierda, y buscamos en la tabla derecha.

SINTAXIS

2. *SELECT campo(s) FROM table_a LEFT JOIN table_b ON table_b.FK = table_a.PK;*

RIGHT JOIN

Empleado	Departamento
Rafferty	Sales
Jones	Engineering
Heisenberg	Engineering
Robinson	Clerical
Smith	Clerical
Williams	NULL

1. En el caso de *RIGHT JOIN* la situación es muy similar, pero aquí se da prioridad a la tabla de la derecha. De tal modo que si usamos la siguiente consulta estaremos mostrando todas las filas de la tabla de la derecha

SINTAXIS

2. *SELECT campo(s) FROM table_a RIGHT JOIN table_b ON table_b.FK = table_a.PK;*

FULL JOIN

Empleado	Departamento
Rafferty	Sales
Jones	Engineering
Heisenberg	Engineering
Robinson	Clerical
Smith	Clerical
Williams	NULL
NULL	Marketing

1.

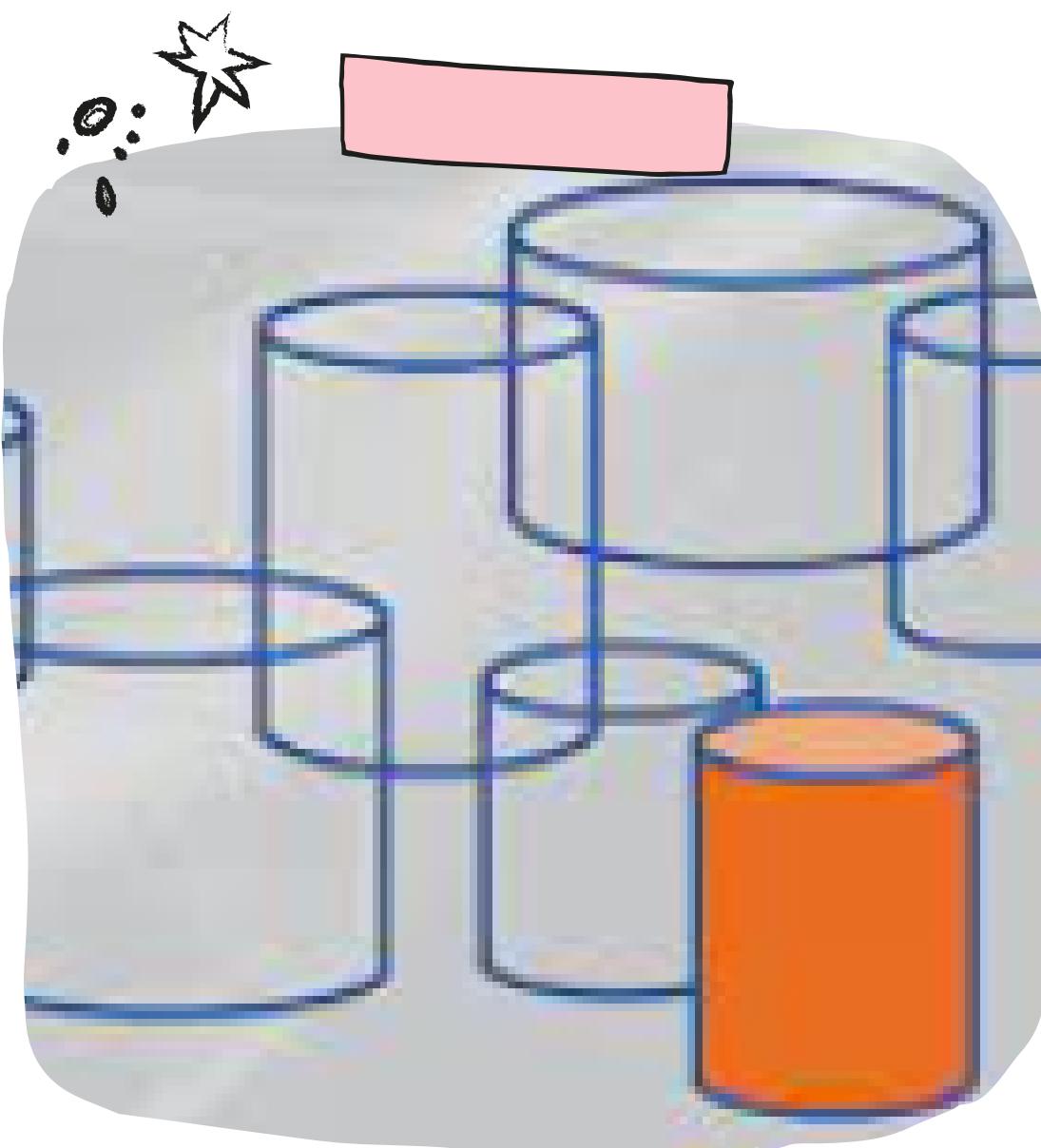
***FULL OUTER JOIN** (o simple **FULL JOIN**) se encarga de mostrar todas las filas de ambas tablas, sin importar que no existan coincidencias (usará **NULL** como un valor por defecto para dichos casos).*

2.

SELECT campo(s) FROM table_a FULL JOIN table_b ON table_b.FK = table_a.PK;

SINTAXIS

BIBLIOGRAFIAS



Sentencias de manipulación de datos. (2022). Retrieved 18 October 2022, from https://www.tetrainfo.com/bd4/lenguaje_st_manipulacion.html

Cláusulas SQL. (2022). Retrieved 18 October 2022, from https://wiki-code.net/20513488-sql-clauses_*-*

Clausulas SQL mas usadas ¿Cuáles son las clausula?. (2022). Retrieved 18 October 2022, from <https://thedevelopmentstages.com/clausulas-sql-mas-usadas/>

Tutorial de Funciones Agregadas de MySQL: SUMA, AVG, MAX, MIN, COUNT, DISTINCT. (2022). Retrieved 18 October 2022, from <https://guru99.es/aggregate-functions/>

