Sintaxis y ejemplos con SQL

Dónde se escribe SQL en phpMyAdmin

1. Entra a la BD (barra izquierda).
2. Pestaña **SQL**.
3. Escribes consultas y ejecutas.

Los comentarios pueden ser:
-- comentario (con espacio después)
comentario
/* comentario largo */

Crear base de datos y usarla

```
CREATE DATABASE tienda;
USE tienda;
```

Tipos de datos más usados

- INT / BIGINT: números enteros
- DECIMAL(10,2): dinero/precios (mejor que FLOAT)
- VARCHAR(50): texto corto
- TEXT: texto largo
- DATE, DATETIME, TIMESTAMP: fechas
- BOOLEAN: en MariaDB suele ser alias de TINYINT(1)
- ENUM('A','B'): valores cerrados

CREATE TABLE (crear tablas) + claves

Vamos a crear un ejemplo típico: **clientes**, **productos**, **pedidos**.

Clientes

```
CREATE TABLE clientes (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(60) NOT NULL,
  email VARCHAR(120) UNIQUE,
  edad INT,
  creado_en TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Qué significa:

- AUTO_INCREMENT: el id se genera solo.
- PRIMARY KEY: clave primaria.
- NOT NULL: obligatorio.
- UNIQUE: no se repite (por ejemplo email).
- DEFAULT CURRENT_TIMESTAMP: fecha/hora automática.

Productos

```
CREATE TABLE productos (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(80) NOT NULL,
  precio DECIMAL(10,2) NOT NULL,
  stock INT NOT NULL DEFAULT 0
);
```

Pedidos con clave foránea (relaciones)

```
CREATE TABLE pedidos (
  id INT AUTO_INCREMENT PRIMARY KEY,
  cliente_id INT NOT NULL,
  fecha DATETIME DEFAULT CURRENT_TIMESTAMP,
  total DECIMAL(10,2) NOT NULL DEFAULT 0,

  CONSTRAINT fk_pedidos_clientes
    FOREIGN KEY (cliente_id) REFERENCES clientes(id)
);
```

Importante: si al borrar un cliente quieres que se borren sus pedidos:

```
FOREIGN KEY (cliente_id) REFERENCES clientes(id) ON DELETE CASCADE
```

INSERT (insertar datos)

Insert normal

```
INSERT INTO clientes (nombre, email, edad)
VALUES ('Ana Pérez', 'ana@correo.com', 29);
```

Insert múltiple

```
INSERT INTO productos (nombre, precio, stock)
VALUES
  ('Teclado', 19.99, 50),
  ('Ratón', 9.99, 120),
  ('Monitor', 149.90, 15);
```

Insert sin indicar columnas (solo si metes TODAS en orden)

No recomendado, pero existe:

```
INSERT INTO clientes
VALUES (NULL, 'Luis Gómez', 'luis@correo.com', 31, DEFAULT);
```

SELECT (consultar datos)

Seleccionar todo

```
SELECT * FROM clientes;
```

Seleccionar columnas

```
SELECT nombre, email FROM clientes;
```

Alias (renombrar columnas en la salida)

```
SELECT nombre AS cliente, creado_en AS alta FROM clientes;
```

ORDER BY (orden)

```
SELECT * FROM productos ORDER BY precio DESC;
```

LIMIT (top N)

SELECT * FROM productos ORDER BY precio DESC LIMIT 2;

WHERE (filtrar) + operadores

Igualdad y comparaciones

SELECT * FROM productos WHERE precio > 20;
SELECT * FROM clientes WHERE edad >= 30;
SELECT * FROM clientes WHERE email = 'ana@correo.com';

AND / OR / NOT

SELECT * FROM productos
WHERE precio > 10 AND stock > 0;

SELECT * FROM clientes
WHERE edad < 18 OR edad IS NULL;

SELECT * FROM productos
WHERE NOT (stock = 0);

IS NULL (nulos)

SELECT * FROM clientes WHERE edad IS NULL;

LIKE (búsquedas por patrón)

- % = cualquier cantidad de caracteres
- _ = un carácter

SELECT * FROM clientes WHERE nombre LIKE 'Ana%'; -- empieza por Ana

SELECT * FROM clientes WHERE nombre LIKE '%Pérez%'; -- contiene Pérez

SELECT * FROM clientes WHERE email LIKE '%@correo.com';

SELECT * FROM clientes WHERE nombre LIKE 'L_is'; -- L + cualquier letra + is

UPDATE (actualizar)

Actualizar filas concretas

```
UPDATE productos
SET precio = 17.99, stock = stock - 1
WHERE id = 1;
```

Importante: sin WHERE actualizas TODO.

```
-- PELIGRO:
UPDATE productos SET stock = 0;
```

DELETE (borrar filas)

```
DELETE FROM clientes WHERE id = 3;
```

Sin WHERE borra todo:

```
-- PELIGRO:
DELETE FROM clientes;
```

DROP (borrar estructura)

Borrar tabla

```
DROP TABLE productos;
```

Borrar base de datos

```
DROP DATABASE tienda;
```

ALTER TABLE (cambiar estructura)

Añadir columna

```
ALTER TABLE clientes ADD telefono VARCHAR(20);
```

Modificar tipo

```
ALTER TABLE clientes MODIFY telefono VARCHAR(30);
```

Renombrar columna

```
ALTER TABLE clientes CHANGE telefono movil VARCHAR(30);
```

Borrar columna

```
ALTER TABLE clientes DROP COLUMN movil;
```

JOIN (unir tablas)

Creamos una tabla de líneas de pedido:

```
CREATE TABLE pedido_detalle (
  id INT AUTO_INCREMENT PRIMARY KEY,
  pedido_id INT NOT NULL,
  producto_id INT NOT NULL,
  cantidad INT NOT NULL DEFAULT 1,
  precio_unitario DECIMAL(10,2) NOT NULL,

  FOREIGN KEY (pedido_id) REFERENCES pedidos(id),
  FOREIGN KEY (producto_id) REFERENCES productos(id)
);
```

INNER JOIN (solo coincidencias)

```
SELECT p.id, c.nombre, p.fecha, p.total
FROM pedidos p
INNER JOIN clientes c ON p.cliente_id = c.id;
```

LEFT JOIN (aunque no haya coincidencia)

Clientes aunque no tengan pedidos:

```
SELECT c.nombre, p.id AS pedido
FROM clientes c
LEFT JOIN pedidos p ON p.cliente_id = c.id;
```

Funciones agregadas + GROUP BY

COUNT, SUM, AVG, MIN, MAX

```
SELECT COUNT(*) AS num_clientes FROM clientes;
```

```
SELECT SUM(stock) AS stock_total FROM productos;

SELECT AVG(precio) AS precio_medio FROM productos;
```

GROUP BY (agrupar)
Cuántos pedidos tiene cada cliente:

```
SELECT c.nombre, COUNT(p.id) AS pedidos
FROM clientes c
LEFT JOIN pedidos p ON p.cliente_id = c.id
GROUP BY c.id, c.nombre;
```

HAVING (filtrar grupos)
Solo los que tienen 2 o más pedidos:

```
SELECT c.nombre, COUNT(p.id) AS pedidos
FROM clientes c
JOIN pedidos p ON p.cliente_id = c.id
GROUP BY c.id, c.nombre
HAVING COUNT(p.id) >= 2;
```

IN, BETWEEN, DISTINCT

```
SELECT * FROM clientes WHERE edad IN (18, 21, 30);

SELECT * FROM productos WHERE precio BETWEEN 10 AND 50;

SELECT DISTINCT email FROM clientes;
```

Subconsultas (nivel medio)

Productos más caros que el precio medio:

```
SELECT *
FROM productos
WHERE precio > (SELECT AVG(precio) FROM productos);
```

Índices (para acelerar búsquedas)

```
CREATE INDEX idx_clientes_email ON clientes(email);
```

Buenas prácticas

- Siempre pon WHERE en UPDATE/DELETE (y prueba antes con SELECT).
- Usa DECIMAL para dinero.
- Pon claves: PRIMARY KEY, FOREIGN KEY, UNIQUE.
- Pon NOT NULL cuando sea obligatorio.
- Nombres claros: tablas en plural o singular, pero consistente.