CONSULTAS BÁSICAS MySQL

David Rodera

ÍNDICE

INTRODUCCIÓN

La **estructura básica** de una consulta en MySQL usa la sentencia SELECT seguida de la lista de columnas y la tabla de la que se obtienen los datos. Esta es así:

```
SELECT columnas FROM tabla;
```

O así:

```
SELECT
    columnas
FROM
    tabla;
```

COLUMNAS

Son los **nombres** de los **campos** que va a devolver la consulta, separados por comas:

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

Aunque no siempre es lo más eficiente, para indicar que se devuelvan **todas las columnas** de la tabla, se usa ***** :

```
SELECT * FROM clientes;
```

Además, se puede poner un alias a cada columna usando un ***as*** y después el nuevo nombre.

```
SELECT detalle_pedido AS detalles FROM clientes;
```

TABLAS

Son el sitio de dónde salen los datos, es decir, qué tabla o tablas se van a leer en la base de datos.

```
SELECT * FROM clientes;
```

```
SELECT * FROM detalle_pedido;
```

WHERE

WHERE es la cláusula que se usa para **filtrar filas** en una **consulta**, indicando qué condición deben cumplir los registros para aparecer en el resultado.

```
SELECT * FROM detalle_pedido WHERE precio_unitario >= 9.99;
```

IMPORTANTE, si la cláusula es un **texto** debe ir entre **comillas**.

```
SELECT * FROM clientes WHERE nombre = ‘Ana’;
```

Si queremos **varias cláusulas**, debemos poner un ***and*** o un ***or*** después de la primera claúsula.

```
SELECT
    *
FROM
    clientes
WHERE
    nombre = ‘Ana’ AND pais = ‘España’;
```

Dentro de las clausulas se pueden introducir más funciones que se verán más adelante.

```
SELECT * FROM clientes WHERE nombre LIKE ('Ana%');
```

Cuando buscamos NULL, se escribe **WHERE columna IS NULL** y cuando no lo buscamos, se escribe **WHERE columna IS NOT NULL**.

ORDER BY y LIMIT

ORDER BY sirve para **ordenar** las filas que devuelve el SELECT, por una o varias columnas, en orden **ascendente** ↑ (ASC) o **descendente** ↓ (DESC). Hay que tener en cuenta que las letras no son un valor, pero podemos interpretarlas para guiarnos, siendo la letra a como un 1 y así hasta la z que será un 27.

```
SELECT * FROM clientes ORDER BY nombre ASC;
```

```
SELECT * FROM clientes ORDER BY nombre ASC, pais DESC;
```

LIMIT se usa para **recortar** el **número** de **filas** del **resultado**, por ejemplo LIMIT 10 devuelve solo las 10 **primeras** filas después de aplicar el ORDER BY si lo hay.

```
SELECT * FROM clientes LIMIT 5;
```

Obviamente estas dos se pueden **combinar entre si** y con más **funciones**.

```
SELECT
    nombre, email, pais
FROM
    clientes
WHERE
    nombre = 'Ana' AND pais = 'España'
ORDER BY email ASC
LIMIT 10;
```

FUNCIONES

FUNCIONES DE FILTRADO

IN comprueba si un valor está dentro de una lista de valores.

```
SELECT * FROM clientes WHERE nombre IN ('España');
```

LIKE busca patrones en cadenas de texto usando comodines (**%** para **cualquier cantidad** de caracteres, _ para **un** solo **carácter**).

```
SELECT * FROM clientes WHERE nombre LIKE ('Ana%');
```

```
SELECT * FROM clientes WHERE nombre LIKE ('Ana%Torre_');
```

BETWEEN filtra valores dentro de un rango inclusivo, por ejemplo entre dos números o fechas.

```
SELECT
    *
FROM
    clientes
WHERE
    precio_unitario BETWEEN 10 AND 100;
```

FUNCIONES DE AGREGACIÓN

Las funciones **YEAR(), MONTH(), DAY(), HOUR()** y **MINUTE()** extraen partes específicas de una fecha/hora en MySQL.

```
SELECT * FROM clientes WHERE YEAR(fecha_registro) = 2023;
```

```
SELECT HOUR(fecha_pedido) FROM pedidos;
```

DAYOFWEEK() devuelve el día de la semana como un valor y **DAYNAME()** devuelve el nombre del día de la semana.

Nombre del día:	Domingo	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
Valor del día:	1	2	3	4	5	6	7

```
SELECT DAYOFWEEK(fecha_registro) FROM clientes;
```

```
SELECT DAYNAME(fecha_registro) FROM clientes;
```

MIN() devuelve el valor más pequeño de una columna y **MAX()** el más grande.

```
SELECT MAX(coste_total), MIN(coste_total) FROM pedidos;
```

COUNT() devuelve el número de filas o valores no nulos en una columna. Si se cogen dos columnas debe ir acompañado de un ***group by***.

```
SELECT COUNT(nombre) FROM clientes;
```

SUM() calcula la suma total de los valores numéricos de una columna, ignorando los valores NULL. Si se cogen dos columnas debe ir acompañado de un ***group by***.

```
SELECT SUM(coste_total) FROM pedidos;
```

AVG() calcula el promedio aritmético (suma dividida entre cantidad) de los valores numéricos de una columna, ignorando los NULL. Si se cogen dos columnas debe ir acompañado de un ***group by***.

```
SELECT AVG(coste_total) FROM pedidos;
```

FUNCIONES NUMÉRICAS

ROUND(numero, decimales) redondea a los decimales indicados.

FLOOR(número) devuelve el entero menor o igual (siempre hacia abajo).

SQRT(número) hace la raíz cuadrada.

FUNCIONES DE TEXTO

LOWER(texto) convierte todo a minúsculas.

UPPER(texto) convierte todo a MAYÚSCULAS.

SUBSTRING(texto, inicio, longitud) extrae parte de una cadena.

LENGTH() devuelve el número de caracteres de una cadena de texto.

Todas estas funciones se pueden **combinar entre sí** y con la clausula ***where***.

GROUP BY

GROUP BY agrupa filas con valores iguales en una columna para aplicar **funciones** de **agregación** (COUNT, SUM, AVG) a cada grupo.

```
SELECT pais, COUNT(nombre) FROM clientes GROUP BY pais;
```

```
SELECT
    categoria, COUNT(nombre)
FROM
    productos
GROUP BY categoria;
```

HAVING

HAVING filtra grupos después de que **GROUP BY** los ha creado y las funciones de agregación se han calculado, en cambio **WHERE**, filtra columnas que eran de la tabla primeramente.

```
SELECT
    categoria, COUNT(nombre)
FROM
    productos
GROUP BY categoria
HAVING COUNT(nombre) > 2;
```

ORDEN DE EJECUCIÓN

FROM → WHERE → GROUP BY → HAVING → SELECT → ORDER BY → LIMIT

PARA PRACTICAR

1. Lista todos los clientes
2. Muestra todos los productos
3. Encuentra clientes de España
4. Lista productos con precio menor a 50
5. Muestra clientes cuyo nombre empiece con "A"
6. Cuenta cuántos clientes hay
7. Encuentra el precio máximo de los productos
8. Lista los 5 productos más caros
9. Muestra clientes registrados después de 2025-01-01
10. Cuenta productos por categoría que contenga "Electr"
11. Lista productos entre 20 y 100 euros
12. Encuentra clientes cuyo email termine en "gmail.com"
13. Muestra el nombre del producto más barato
14. Cuenta pedidos con total mayor a 200
15. Lista productos ordenados por precio descendente
16. Encuentra clientes cuyo nombre tenga más de 8 caracteres
17. Muestra el año de los pagos más recientes
18. Cuenta productos cuyo precio no sea NULL
19. Lista detalles de pedido con cantidad mayor a 2
20. Encuentra el día de la semana del pedido más antiguo
21. Muestra el promedio de pagos
22. Lista categorías en mayúsculas
23. Cuenta productos con un stock único
24. Encuentra productos cuyo nombre contenga "phone" o "Phone"
25. Redondea a 2 decimales el precio mínimo de productos
26. Muestra el nombre del cliente con ID 5 en minúsculas
27. Cuenta detalles de pedido por producto que superen 100 unidades totales
28. Encuentra meses con más de 3 pagos realizados
29. Lista productos cuya descripción tenga exactamente 10 caracteres
30. Muestra categorías con más de 5 productos, ordenadas por cantidad descendente

SOLUCIONES

1.
```sql
SELECT * FROM clientes;
```
2.
```sql
SELECT * FROM productos;
```
3.
```sql
SELECT * FROM clientes WHERE pais = 'España';
```
4.
```sql
SELECT * FROM productos WHERE precio > 50;
```
5.
```sql
SELECT * FROM clientes WHERE nombre LIKE ('A%');
```
6.
```sql
SELECT * FROM productos WHERE precio > 50;
```
7.
```sql
SELECT MAX(precio) FROM productos;
```
8.
```sql
SELECT * FROM productos ORDER BY precio DESC LIMIT 5;
```
9.
```sql
SELECT * FROM clientes WHERE fecha_registro > '2025-01-01';
```
10.
```sql
SELECT
    COUNT(*)
FROM
    productos
WHERE
    categoria LIKE ('Electr%');
```
11.
```sql
SELECT * FROM productos WHERE precio BETWEEN 20 AND 100;
```
12.
```sql
SELECT * FROM clientes WHERE email LIKE ('%gmail.com');
```
13.
```sql
SELECT * FROM productos ORDER BY precio ASC LIMIT 1;
```

14.

```
SELECT * FROM pedidos WHERE precio > 1;
```

15.

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

16.

```
SELECT * FROM clientes WHERE LENGTH(nombre) > 8;
```

17.

```
SELECT YEAR(fecha_pago) FROM pagos ORDER BY fecha_pago DESC;
```

18.

```
SELECT COUNT(*) FROM productos WHERE precio IS NOT NULL;
```

19.

```
SELECT
   id_pedido, count(id_pedido)
FROM
    detalle_pedido
GROUP BY id_pedido
HAVING count(id_pedido) > 2;
```

20.

```
SELECT
   DAYNAME(fecha_pedido)
FROM
    pedidos
ORDER BY fecha_pedido ASC
LIMIT 1;
```

21.

```
SELECT ROUND(AVG(total_pagado) 2)  FROM pagos;
```

22.

```
SELECT UPPER(categoria) FROM productos GROUP BY categoria;
```

23.

```
SELECT COUNT(DISTINCT(stock)) FROM productos;
```

24.

```
SELECT * FROM productos WHERE nombre LIKE ('%phone%');
```

25.

```
SELECT MIN(precio) FROM productos ORDER BY precio ASC LIMIT
```

```
1;
```

26.

```
SELECT LOWER(nombre) FROM clientes WHERE id_cliente = 5;
```

27.

```
SELECT
    id_producto, COUNT(id_producto)
FROM
    detalle_pedido
GROUP BY id_producto
HAVING COUNT(id_producto) > 100;
```

28.

```
SELECT
   MONTHNAME(fecha_pago), COUNT(id_pago)
FROM
   pagos
GROUP  BY MONTHNAME(fecha_pago)
HAVING COUNT(id_pago) > 3;
```

29.

```
SELECT * FROM productos WHERE LENGTH(descripcion) = 10;
```

30.

```
SELECT
    categoria, COUNT(id_producto)
FROM
    productos
GROUP BY categoria
HAVING COUNT(id_producto) > 5
ORDER BY COUNT(id_producto) DESC;
```