

Introducción a Bases de Datos y SQL

Módulo 2

Predicado de consulta SQL

Cláusula WHERE

La cláusula **WHERE** permite especificar las condiciones o criterios que deben cumplir los registros a buscar dentro de una tabla. **Recuerda siempre el orden en que se deben especificar las cláusulas:**

FROM

WHERE

GROUP BY

HAVING

ORDER BY

Anteriormente, sólo se utilizaron las cláusulas **FROM** y **ORDER BY** para especificar la tabla en la que se realizaría la búsqueda y el orden en que se debían mostrar los registros de dicha tabla en el resultado de la consulta. Al especificar condiciones de búsqueda dentro de una tabla, se utilizará la cláusula **WHERE**, la cual se debe colocar **obligatoriamente** después de la cláusula **FROM**.

Operadores de comparación

Operador	Descripción
=	Igual a
<	Menor que
>	Mayor que
>=	Mayor o igual que
<=	Menor o igual que
<>	No es igual a

Operadores de comparación

El siguiente ejemplo selecciona de la tabla **ARTICULOS** la columna **NOMBRE** y muestra todos aquellos registros cuyo valor en la columna **CODIGO** sea igual a 1.

```
SELECT Nombre FROM Articulos WHERE codigo = 1;
```

El siguiente ejemplo selecciona de la tabla **ARTICULOS** las columnas **NOMBRE** y **PRECIO** y muestra todos aquellos registros cuyo **PRECIO** sea superior a 150.

```
SELECT Nombre, Precio FROM Articulos WHERE Precio > 150;
```

Cláusula WHERE

Operadores lógicos

Para crear expresiones lógicas disponemos de varios operadores de comparación. Estos operadores se aplican a cualquier tipo de columna: fechas, cadenas, números, etc, y devuelven valores lógicos: verdadero o falso (1 o 0).

Si uno o los dos valores a comparar son **NULL**, el resultado es **NULL**, excepto con el operador **<=>** que es usado para una comparación con **NULL** segura.

El operador **<=>** funciona igual que el operador **=**, salvo que si en la comparación una o ambas de las expresiones es nula el resultado no es **NULL**. Si se comparan dos expresiones nulas, el resultado es verdadero.

Operador	Descripción
AND	Se deben cumplir todas las condiciones especificadas
OR	Se debe cumplir al menos una de las condiciones especificadas
NOT	No debe cumplir las condiciones especificadas

Operadores lógicos

Cláusula WHERE

Operadores lógicos

El siguiente ejemplo selecciona de la tabla **ARTICULOS** todos aquellos registros cuyo **PRECIO** tenga un valor mayor o igual a 500 o su **STOCK** sea mayor o igual a 100.

```
SELECT * FROM Articulos WHERE precio >= 500 OR stock >= 100;
```

El siguiente ejemplo selecciona de la tabla **ARTICULOS** todos aquellos registros cuyo **PRECIO** tenga un valor menor a 20 y su **STOCK** sea mayor o igual a 100.

```
SELECT * FROM Articulos WHERE Precio < 20 AND stock >= 100;
```

Operadores BETWEEN / NOT BETWEEN

Entre los operadores de MySQL, existe uno denominado **BETWEEN** (entre), el cual se utiliza para comprobar si una expresión está comprendida en un determinado rango de valores. La sintaxis es:

- **BETWEEN** mínimo **AND** máximo
- **NOT BETWEEN** mínimo **AND** máximo

```
SELECT * FROM Articulos WHERE precio BETWEEN 100 AND 200;
```

```
SELECT * FROM Articulos WHERE precio NOT BETWEEN 100 AND 200;
```



Operadores IN / NOT IN

Los operadores **IN** y **NOT IN** sirven para averiguar si el valor de una expresión determinada está dentro de un conjunto indicado. Su sintaxis es:

- **IN** (<expr1>, <expr2>, <expr3>,...)
- **NOT IN** (<expr1>, <expr2>, <expr3>,...)

El operador **IN** devuelve un valor verdadero si el valor de la expresión es igual a alguno de los valores especificados en la lista. El operador **NOT IN** devuelve un valor falso en el caso contrario.

Cláusula WHERE

Operadores IN / NOT IN

Ejemplo N° 1:

```
SELECT * FROM Articulos WHERE codigo IN (1,2,3);
```

Ejemplo N° 2:

```
SELECT * FROM Articulos WHERE nombre IN ('Pala', 'Maza');
```

Ejemplo N° 3:

```
SELECT * FROM Articulos WHERE nombre NOT IN ('Pala', 'Maza');
```



Cláusula WHERE

Operadores LIKE / NOT LIKE

El operador **LIKE** se usa para hacer comparaciones entre cadenas y patrones. El resultado es verdadero (1) si la cadena se ajusta al patrón, y falso (0) en caso contrario. Tanto si la cadena como el patrón son **NULL**, el resultado es **NULL**.

Para definir estos patrones, se hace uso de **comodines**:

Carácter	Descripción
%	Coincidencia con cualquier número de caracteres, incluso ninguno.
_	Coincidencia con un único carácter.

El siguiente ejemplo muestra, de la tabla **ARTICULOS**, todos aquellos registros que en el campo **NOMBRE** figure la palabra **PALA**.

```
SELECT * FROM Articulos WHERE nombre LIKE '%Pala%';
```

Cláusula WHERE

Operadores LIKE / NOT LIKE

La comparación es independiente del tipo de los caracteres, es decir, **LIKE** no distingue mayúsculas de minúsculas, salvo que se indique lo contrario.

Como siempre que se usan caracteres concretos para crear patrones, se presenta la dificultad de hacer comparaciones cuando se deben buscar precisamente esos caracteres concretos.

Esta dificultad se suele superar mediante secuencias de escape. Si no se especifica nada en contra, el carácter que se usa para escapar es '\'. De este modo, si queremos que nuestro patrón contenga los caracteres '%' o '_', los escaparemos de este modo: '\%' y '_':

Ejemplo:

```
SELECT * FROM clientes WHERE mail LIKE '%\_%';
```

Cláusula WHERE

Operadores IS NULL / IS NOT NULL

Los operadores **IS NULL** e **IS NOT NULL** sirven para verificar si una expresión determinada es o no nula.

El siguiente ejemplo mostrará todos aquellos registros de la tabla **CLIENTES** que no tengan cargado ningún valor en el campo **COMENTARIOS**:

```
SELECT * FROM clientes WHERE comentarios IS NULL;
```

El siguiente ejemplo mostrará todos aquellos registros de la tabla **CLIENTES** que tengan algún valor cargado en el campo **COMENTARIOS**:

```
SELECT * FROM clientes WHERE comentarios IS NOT NULL;
```



¡Muchas gracias!

¡Sigamos trabajando!