Introducción a
Bases de Datos

Módulo 2

Predicado de consulta SQL

Operadores de comparación

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

El siguiente ejemplo selecciona de la tabla **Artículos** la columna **Nombre** del registro cuyo valor de la columna **código** sea igual a 1

Sintaxis

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

El siguiente ejemplo selecciona de la tabla **Artículos** las columnas **Nombre** y **Precio** de los registros cuyo precio tenga un valor mayor a 150.

Sintaxis

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

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 (o bien 1 ó 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 debe cumplir con todas las condiciones
OR	Se debe cumplir al menos una de las condiciones

Operadores lógicos

El siguiente ejemplo selecciona de la tabla **Articulos,** los registros **cuyo Precio** tenga un valor mayor o igual a 500 **ó** su **stock** sea mayor o igual a 100.

Sintaxis

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

El siguiente ejemplo selecciona de la tabla **Articulos,** los registros cuyo **Precio** tenga un valor menor a 20 **y** su **stock** sea mayor o igual a 100.

Sintaxis

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

Cláusulas especiales

Sentencia BETWEEN

Entre los operadores de MySQL, hay uno 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

Sintaxis

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

Sintaxis

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

Cláusulas especiales

Sentencia IN

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

- **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.

Cláusulas especiales

Ejemplos **Sentencia IN**:

Sintaxis

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

Sintaxis

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

Sintaxis

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

Cláusulas especiales

Sentencia 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.

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

Sintaxis

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

Cláusulas especiales

Sentencia LIKE

La comparación es independiente del tipo de los caracteres, es decir, LIKE no distingue mayúsculas de minúsculas, salvo que se indique lo contrario (ver operadores de casting):

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 '_':

Sintaxis

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

Cláusulas especiales

Valores NULL

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

Sintaxis

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

Sintaxis

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

Conceptos avanzados

SQL permite especificar una lista de valores para ser utilizados con la cláusula IN de manera dinámica a través de la utilización de subconsultas.

Subconsultas

El siguiente ejemplo utiliza una subconsulta para conocer todos los datos de los artículos que fueron vendidos.

Sintaxis

```
SELECT * FROM articulos WHERE articuloID IN
(SELECT articuloID FROM facturas);
```

¡Muchas gracias!

¡Sigamos trabajando!