* Operadores Lógicos.
* <https://www.youtube.com/watch?v=XkPLx96er5U>
* Funciones Fecha-Hora
* <https://www.youtube.com/watch?v=jZR0d_QvgKA>
* Funciones cadenas de caracteres
* <https://www.youtube.com/watch?v=vlSWqnXFQrA>

Los operadores lógicos comprueban la veracidad de alguna condición. (representan un valor de verdad, es decir, TRUE o FALSE,  puede producir un valor desconocido,  el valor TRUE, FALSE o UNKNOWN.

Operator             Significado

ALL                         TRUE si el conjunto completo de comparaciones es TRUE.

AND                      TRUE si ambas expresiones booleanas son TRUE.

ANY                       TRUE si cualquier miembro del conjunto de comparaciones es TRUE.

BETWEEN            TRUE si el operando está dentro de un intervalo.

EXISTS                  TRUE si una subconsulta contiene cualquiera de las filas.

IN                           TRUE si el operando es igual a uno de la lista de expresiones.

LIKE                       TRUE si el operando coincide con un patrón.

NOT                       Invierte el valor de cualquier otro operador booleano.

OR                          TRUE si cualquiera de las dos expresiones booleanas es TRUE.

SOME                   TRUE si alguna de las comparaciones de un conjunto es TRUE.

(Los operadores marcados con color rojo veremos en el siguiente módulo)

Los registros recuperados con una sentencia que une 2 condiciones con el operador "or", cumplen 1 de las condiciones o ambas.

Queremos recuperar los libros que NO cumplan la condición dada, por ejemplo, aquellos cuya editorial NO sea "SANREMO":

select \* from libros

 where not editorial='SANREMO';

El operador "not" invierte el resultado de la condición a la cual antecede.

Los registros recuperados en una sentencia en la cual aparece el operador "not", no cumplen con la condición a la cual afecta el "NOT".

**Los paréntesis se usan para encerrar condiciones, para que se evalúen como una sola expresión.**

Cuando explicitamos varias condiciones con diferentes operadores lógicos (combinamos "and", "or") permite establecer el orden de prioridad de la evaluación; además permite diferenciar las expresiones más claramente.

Por ejemplo, las siguientes expresiones devuelven un resultado diferente:

select\* from libros

 where (autor='Gabriela Mistral') or

 (editorial='CLARA' and precio<20);

select \* from libros

 where (autor='MISTRA' or editorial='CLARA') and

 (precio<20);

Si bien los paréntesis no son obligatorios en todos los casos, se recomienda utilizarlos para evitar confusiones.

El orden de prioridad de los operadores lógicos es el siguiente: "not" se aplica antes que "and" y "and" antes que "or", si no se especifica un orden de evaluación mediante el uso de paréntesis.

El orden en el que se evalúan los operadores con igual nivel de precedencia es indefinido, por ello se recomienda usar los paréntesis.

Entonces, para establecer más de una condición en un "where" es necesario emplear operadores lógicos. "and" significa "y", indica que se cumplan ambas condiciones; "or" significa "y/o", indica que se cumpla una u otra condición (o ambas); "not" significa "no", indica que no se cumpla la condición especificada.

SQL Server, se procede a crear la tabla.

Ingresemos el siguiente lote de comandos en el SQL Server Management Studio:

create table libros(

 codigo int identity,

 titulo varchar(40) not null,

 autor varchar(20) default 'Desconocido',

 editorial varchar(20),

 precio decimal(6,2),

)

insert into libros

values('Pasión de Enseñar','Gabriela Mistral','Emece',15.90);

insert into libros

values('Antología poética','Borges','Sanremo',39.50);

insert into libros

values('Java en 10 minutos','Mario Molina','Sanremo',50.50);

insert into libros

 values('Alicia en el pais de las maravillas','Lewis Carroll','Emece',19.90);

insert into libros

 values('Martin Fierro','Jose Hernandez','Emece',25.90);

insert into libros

 values('Martin Fierro','Jose Hernandez','Clara',16.80);

insert into libros

 values('Aprenda PHP','Mario Molina','Emece',19.50);

insert into libros

- Recuperamos los libros cuyo autor sea igual a "Gabriela Mistral" y cuyo precio no supere los 20 pesos:

select \* from libros

 where (autor='MISTRAL') and

 (precio<=20);

-Seleccionamos los libros cuyo autor es "Gabriela Mistral" y/o cuya editorial es "Sanremo"

select \* from libros

 where autor='Gabriela Mistral' or

 editorial='Sanremo';

-Recuperamos los libros cuya editorial NO es "Sanremo"

select \* from libros

 where not editorial='Sanremo';

- Veamos cómo el uso de paréntesis hace que SQL Server evalúe en forma diferente

ciertas consultas aparentemente iguales

select \* from libros

 where (autor='Borges') or

 (editorial='Emece' and precio<20);

select \* from libros

 where (autor='Borges' or editorial='Emece') and

 (precio<20);

BETWEEN: Recupera Rango de Valores (entre)

El operador SQL BETWEEN se utiliza para comparar un rango de valores y es equivalente a usar la combinación AND  de los operadores de comparación >= y <=, pero hace que la comparación de rango sea más compacta.

Este operador se usa más comúnmente con el comando SELECT. La sintaxis de esto es la siguiente:

SELECT nombre\_columna(s)

FROM nombre\_tabla

WHERE nombre\_columna BETWEEN valor1 AND valor2;

SELECT CODIGO

FROM LIBROS

WHERE CODIGO BETWEEN 1 AND 7;

Las columnas que se recuperarán se especifican en la instrucción SELECT.

La tabla de la que se recuperan las columnas se especifica en la instrucción FROM.

El operador BETWEEN se usa en la cláusula WHERE. La columna a la que queremos aplicar la condición de rango  “range\_start” y el valor final en el parámetro “range\_end”.

- Funciones Fecha-Hora

GETDATE OBTENER FECHA

DATEADD AGREGAR FECHA

DATENAME NOMBRE DE FECHA

DAY - MONTH - YEAR

- OBTENER LA FECHA Y LA HORA DEL SISTEMA

SELECT GETDATE ()

- OBTENER UNA PARTE DE LA FECHA Y HORA

SELECT DATEPART (DAY, GETDATE ())

- OBTIENE EL NOMBRE DE UNA PARTE DE LA FECHA

SELECT DATENAME (MONTH, GETDATE())

- CALCULAR EL INTERVALO DE TIEMPO

SELECT DATEDIFF (YEAR,'2020/01/15', '2021/01/15')

SELECT DATEDIFF (DAY,'2020/01/15', '2021/01/15')

- OBTENER EL AÑO

SELECT YEAR (GETDATE ())

- VISUALIZAR FECHA ACTUAL

SELECT GETDATE()

- AGREGAR 10 DIAS A LA FECHA ACTUAL

SELECT DATEADD (DAY,10, GETDATE())

- CONVERTIR LA FECHA A CADENA

SELECT CONVERT (VARCHAR, GETDATE (), 107)

-MOSTRAR EL NOMBRE DEL MES DE UNA FECHA

DECLARE @FECHA DATETIME

SET @FECHA='20220415'

SELECT DATENAME (MM,@FECHA)

-FUNCIÓN SQL LEFT-RIGHT

Estas funciones devuelven la cantidad especificada de caracteres más a la izquierda o más a la derecha de una cadena de caracteres.

- La cantidad se basa en la cantidad de caracteres, no bytes, por lo que los caracteres multibyte se cuentan  como caracteres simples.

SELECT LEFT ( string,  integer )

SELECT LEFT ( nombre\_campo, integer) from nombre\_tabla

SELECT RIGHT ( string,  integer )

SELECT RIGHT ( nombre\_campo, integer) from nombre\_tabla

select\* from libros

SELECT TITULO, LEFT (AUTOR,10) FROM LIBROS

SELECT TITULO, RIGHT (AUTOR,10) FROM LIBROS

- FUNCIÓN LEN()  Devuelve la longitud de una cadena

Sin embargo, los espacios iniciales al comienzo de la cadena se incluyen al calcular la longitud.

SELECT LEN('caracteres');

SELECT LEN (nombre\_campo) from nombre\_tabla

select\* from libros

SELECT LEN (TITULO)FROM LIBROS

- FUNCIÓN LOWER

La función LOWER () retorna una expresión de caracteres convertida de mayúsculas a minúsculas.

SELECT LOWER ('caracteres')

SELECT LOWER ( nombre\_campo) from nombre\_tabla}

select\* from libros

SELECT AUTOR, LOWER (AUTOR)

FROM LIBROS;

- FUNCIÓN SQL UPPER

La función UPPER() convierte el valor del campo a mayúscula

SELECT UPPER ( caracteres)

SELECT UPPER ( nombre\_campo) from nombre\_tabla

select\* from libros

SELECT AUTOR, UPPER (AUTOR)

FROM LIBROS;

- FUNCIÓN SQL SUBSTRING

- La función SUBSTRING() retorna el substring (cadena) especificado, recorte de cadena.

SELECT SUBSTRING (COLUMNA, POSICION INICIAL, POSICION FINAL)

select\*

FROM LIBROS

SELECT SUBSTRING (AUTOR,1,3),editorial, precio

SELECT SUBSTRING (AUTOR,1,3), substring(editorial,1,3), precio

from libros

- FUNCIÓN SQL CONCAT

CONCAT toma un número variable de argumentos de cadena y los concatena (o combina) en una sola cadena.

Necesita un mínimo de dos valores de entrada; de lo contrario, se produce un error en CONCAT .

CONCAT convierte implícitamente todos los argumentos en tipos de cadena antes de la concatenación Retorna un string que es el resultado de concatenar dos o más valores (strings).

DECLARE @NRO INT = 10

DECLARE @STRING VARCHAR(255) = 'SQL'

DECLARE @STRING2 VARCHAR(255) = NULL

DECLARE @FECHA DATE = GETDATE()

SELECT CONCAT(@NRO,'-',@STRING,'-',@STRING2,'-',@FECHA)