Material imprimible

Curso de SQL Server

Módulo 2: Consultas

Cláusulas

Para realizar consultas sobre las tablas de las bases de datos disponemos de la instrucción **SELECT**. Con ella podemos consultar una o varias tablas. Es sin duda **el comando más versátil del lenguaje SQL**.

Existen muchas cláusulas asociadas a la sentencia SELECT (GROUP BY, ORDER, HAVING, UNION). También es una de las instrucciones en la que con más frecuencia los motores de bases de datos incorporan cláusulas adicionales al estándar.

El resultado de una consulta SELECT nos devuelve **una tabla lógica**. Es decir, los resultados son una relación de datos, que tiene filas/registros, con una serie de campos/columnas. Igual que cualquier tabla de la base de datos. Sin embargo, esta tabla está en memoria mientras la utilicemos, y luego se descarta. Cada vez que ejecutamos la consulta se vuelve a calcular el resultado.

La sintaxis básica de una consulta SELECT es la siguiente (los valores opcionales van entre corchetes):

```
SELECT [ ALL / DISTINC ] [ * ] / [ListaColumnas_Expresiones] AS [Expresion]
FROM Nombre_Tabla_Vista
WHERE Condiciones
ORDER BY ListaColumnas [ ASC / DESC ]
```

SELECT

Permite seleccionar las columnas que se van a mostrar y en el orden en que lo van a hacer. Simplemente es la instrucción que la base de datos interpreta como que vamos a solicitar información.

FROM

Esta cláusula permite indicar las tablas o vistas de las cuales vamos a obtener la información. De momento veremos ejemplos para obtener información de una sola tabla.

WHERE

Especifica la condición de filtro de las filas devueltas. Se utiliza cuando no se desea que se devuelvan todas las filas de una tabla, sino sólo las que cumplen ciertas condiciones. Lo habitual es utilizar esta cláusula en la mayoría de las consultas.

Condiciones

Son expresiones lógicas a comprobar para la condición de filtro, que tras su resolución devuelven para cada fila TRUE o FALSE, en función de que se cumplan o no. Se puede utilizar cualquier expresión lógica y en ella utilizar diversos operadores como:

- > (Mayor)
- >= (Mayor o igual)
- < (Menor)
- <= (Menor o igual)
- = (Igual)
- <>o != (Distinto)
- IS [NOT] NULL (para comprobar si el valor de una columna es o no es nula, es decir, si contiene o no contiene algún valor)

Se dice que una columna de una fila es NULL si está completamente vacía. Hay que tener en cuenta que, si se ha introducido cualquier dato, incluso en un campo alfanumérico si se introduce una cadena en blanco o un cero en un campo numérico, deja de ser NULL.

LIKE

Se utiliza para la comparación de un modelo. Para ello utiliza los caracteres comodín especiales: “%” y “_”. Con el primero indicamos que en su lugar puede ir cualquier cadena de caracteres, y con el segundo que puede ir cualquier carácter individual (un solo carácter). Con la combinación de estos caracteres podremos obtener múltiples patrones de búsqueda. Por ejemplo:

- El nombre empieza por A: Nombre LIKE 'A%'
- El nombre acaba por A: Nombre LIKE '%A'
- El nombre contiene la letra A: Nombre LIKE '%A%'
- El nombre empieza por A y después contiene un solo carácter cualquiera: Nombre LIKE 'A_'
- El nombre empieza una A, después cualquier carácter, luego una E y al final cualquier cadena de caracteres: Nombre LIKE 'A_E%'

BETWEEN

Se utiliza para un intervalo de valores. Por ejemplo:

- Clientes entre el 30 y el 100: CodCliente BETWEEN 30 AND 100
- Clientes nacidos entre 1970 y 1979: FechaNac BETWEEN '19700101' AND '19791231'

IN

Se utiliza para especificar una relación de valores concretos. Por ejemplo: Ventas de los Clientes 10, 15, 30 y 75: CodCliente IN(10, 15, 30, 75)

Por supuesto es posible combinar varias condiciones simples de los operadores anteriores utilizando los operadores lógicos **OR**, **AND** y **NOT**, así como el uso de paréntesis para controlar la prioridad de los operadores (como en matemáticas). Por ejemplo: … (Cliente = 100 AND Provincia = 30) OR Ventas > 1000 … que sería para los clientes de las provincias 100 y 30 o cualquier cliente cuyas ventas superen 1000.

ORDER BY

Define el orden de las filas del conjunto de resultados. Se especifica el campo o campos (separados por comas) por los cuales queremos ordenar los resultados.

ASC / DESC

ASC es el valor predeterminado, especifica que la columna indicad en la cláusula ORDER BY se ordenará de forma ascendente, o sea, de menor a mayor. Si por el contrario se especifica DESC se ordenará de forma descendente (de mayor a menor). Por ejemplo, para ordenar los resultados de forma ascendente por ciudad, y los que sean de la misma ciudad de forma descendente por nombre, utilizaríamos esta cláusula de ordenación:

```
… ORDER BY Ciudad, Nombre DESC …
```

Como a la columna *Ciudad* no le hemos puesto ASC o DESC se usará para la misma Como a la columna *Ciudad* no le hemos puesto ASC o DESC se usará para la misma el valor predeterminado (que es ASC)

Cuidado: Aunque al principio si aún no se está habituado, pueda dar la impresión de que se ordena por ambas columnas en orden descendente. Si es eso lo que queremos deberemos escribir ... ORDER BY Ciudad DESC, Nombre DESC ...

Algunos ejemplos

Veamos algunos ejemplos con **la base de datos Northwind** en SQL Server:

- Mostrar todos los datos de los Clientes de nuestra empresa:

```
SELECT * FROM Customers
```

- Mostrar apellido, ciudad y región (LastName, city, region) de los empleados de USA (nótese el uso de AS para darle el nombre en español a los campos devueltos):

```
SELECT E.LastName AS Apellido, City AS Ciudad, Region
FROM Employees AS E
WHERE Country = 'USA'
```

- Mostrar los clientes que no sabemos a qué región pertenecen (o sea, que no tienen asociada ninguna región) :

```
SELECT * FROM Customers WHERE Region IS NULL
```

- Mostrar las distintas regiones de las que tenemos algún cliente, accediendo sólo a la tabla de clientes:

```
SELECT DISTINCT Region FROM Customers WHERE Region IS NOT NULL
```

- Mostrar los clientes que pertenecen a las regiones CA, MT o WA, ordenados por región ascendentemente y por nombre descendentemente.

```
CODE SELECT * FROM Customers WHERE Region IN('CA', 'MT', 'WA')
ORDER BY Region, CompanyName DESC
```

- Mostrar los clientes cuyo nombre empieza por la letra “W”:

```
SELECT * FROM Customers WHERE CompanyName LIKE ‘W%’
```

- Mostrar los empleados cuyo código está entre el 2 y el 9:

```
SELECT * FROM Employees WHERE EmployeeID BETWEEN 2 AND 9
```

- Mostrar los clientes cuya dirección contenga “ki”:

```
SELECT * FROM Customers WHERE Address LIKE ‘%ki%’
```

- Mostrar las Ventas del producto 65 con cantidades entre 5 y 10, o que no tengan descuento:

```
SELECT * FROM [Order Details] WHERE (ProductID = 65 AND Quantity BETWEEN 5
AND 10) OR Discount = 0
```

Nota: En SQL Server, para utilizar nombres de objetos con caracteres especiales se deben poner entre corchetes. Por ejemplo, en la consulta anterior [Order Details] se escribe entre corchetes porque lleva un espacio en blanco en su nombre. En otros SGBDR se utilizan comillas dobles (Oracle, por ejemplo: “Order Details”) y en otros se usan comillas simples (por ejemplo, en MySQL).

Fuente: https://www.campusmvp.es/recursos/post/Fundamentos-de-SQL-Como-realizar-consultas-simples-con-SELECT.aspx