Tema 4. Las consultas de resumen (I)

Introducción

En SQL de Microsoft Jet 4.x y de la mayoría de los motores de bases de datos relacionales, podemos definir un **tipo de consultas** cuyas filas resultantes **son un resumen de las filas de la tabla origen**, por eso las denominamos **consultas de resumen**, también se conocen como consultas sumarias.

Es importante entender que las filas del resultado de una consulta de resumen tienen una **naturaleza distinta** a las filas de las demás tablas resultantes de consultas, ya que corresponden a varias filas de la tabla orgen. Para simplificar, veamos el caso de una consulta basada en una sola tabla, una fila de una consulta 'no resumen' corresponde a una fila de la tabla origen, contiene datos que se encuentran en una sola fila del origen, mientras que **una fila de una consulta de resumen corresponde** a un **resumen de varias filas** de la **tabla origen**, esta diferencia es lo que va a originar una serie de restricciones que sufren las consultas de resumen y que veremos a lo largo del tema.

En el ejemplo que viene a continuación tienes un ejemplo de consulta normal en la que se visualizan las filas de la tabla oficinas ordenadas por region, en este caso cada fila del resultado se corresponde con una sola fila de la tabla oficinas, mientras que la segunda consulta es una consulta resumen, cada fila del resultado se corresponde con una o varias filas de la tabla oficinas.

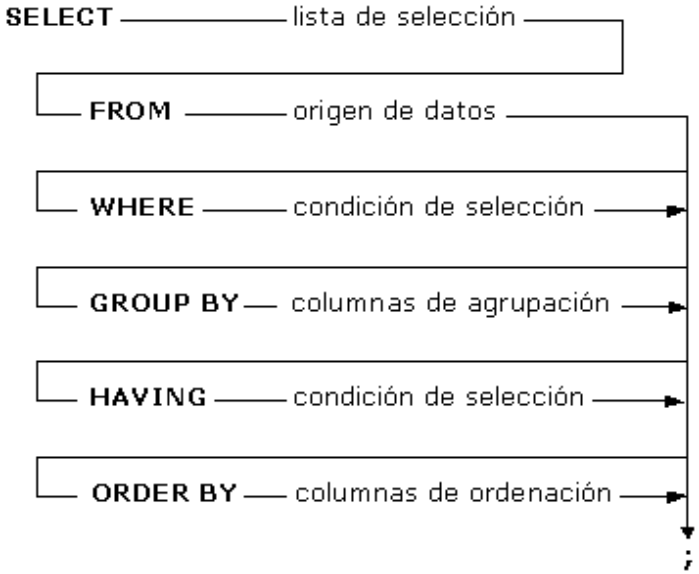
```
SELECT oficina,region,ventas
FROM oficinas
ORDER BY region
```

oficina	region	ventas
24	centro	150.000 Pts
23	centro	
28	este	0 Pts
13	este	368.000 Pts
12	este	735.000 Pts
11	este	693.000 Pts
26	norte	
22	oeste	186.000 Pts
21	oeste	836.000 Pts

```
SELECT region,SUM(ventas)
FROM oficinas
GROUP BY region
```

region	SumaDeventas
centro	150000
este	1796000
norte	
oeste	1022000

Las consultas de resumen introducen dos **nuevas cláusulas** a la sentencia SELECT, la **cláusula GROUP BY** y la **cláusula HAVING**, son cláusulas que **sólo** se pueden utilizar en una consulta de resumen, se tienen que escribir **entre** la cláusula **WHERE** y la cláusula **ORDER BY** y tienen la siguiente sintaxis:



Las detallaremos en la página siguiente del tema, primero vamos a introducir otro concepto relacionado con las consultas de resumen, las funciones de columna.

Funciones de columna

En la lista de selección de una consulta de resumen aparecen **funciones de columna** también denominadas funciones de dominio agregadas. Una función de columna **se aplica a una columna** y obtiene un **valor que resume el contenido de la columna**.

Tenemos las siguientes funciones de columna:

SUM (expresión)	MIN (expresión)
AVG (expresión)	MAX (expresión)
STDEV (expresión)	COUNT (nbcolumna)
STDEVP (expresión)	COUNT (*)

El **argumento** de la función indica **con qué valores** se tiene que operar, por eso ***expresión*** suele ser un **nombre de columna**, columna que contiene los valores a resumir, pero también puede ser cualquier expresión válida que devuelva una lista de valores.

La función **SUM()** calcula la **suma** de los valores indicados en el argumento. Los datos que se suman deben ser de **tipo numérico** (entero, decimal, coma flotante o monetario...). El resultado será del mismo tipo aunque puede tener una precisión mayor.

Ejemplo:

```
SELECT SUM(ventas)
FROM oficinas
```

Obtiene una sola fila con el resultado de sumar todos los valores de la columna ventas de la tabla oficinas.

La función **AVG()** calcula el **promedio** (la media arimética) de los valores indicados en el argumento, también se aplica a **datos numéricos**, y en este caso el tipo de dato del resultado puede cambiar según las necesidades del sistema para representar el valor del resultado.

StDev() y **StDevP()** calculan la **desviación estándar** de una población o de una muestra de la población representada por los valores contenidos en la columna indicada en el argumento. Si la consulta base (el origen) tiene menos de dos registros, el resultado es nulo.

Es interesante destacar que el **valor nulo no equivale al valor 0**, las **funciones de columna no consideran** los **valores nulos** mientras que consideran el valor 0 como un valor, por lo tanto en las funciones AVG(), STDEV(), STDEVP() los resultados no serán los mismos con valores 0 que con valores nulos. Veámoslo con un ejemplo:

Si tenemos esta tabla:

Tabla1 : Tabla

col1
10
5
0
3
6
0

La consulta

```
SELECT AVG(col1) AS media
FROM tabla1
```

devuelve:

media
4

En este caso los ceros entran en la media por lo que sale igual a 4
(10+5+0+3+6+0)/6 = 4

Con esta otra tabla:

Tabla2 : Tabla

col1
10
5
3
6

```
SELECT AVG(col1) AS media
FROM tabla2
```

devuelve:

media
6

En este caso los ceros se han sustituido por valores nulos y no entran en el cálculo por lo que la media sale igual a 6
(10+5+3+6)/4 = 4

Las funciones **MIN()** y **MAX()** determinan los **valores menores** y **mayores** respectivamente. Los valores de la columna pueden ser de **tipo numérico**, **texto** o **fecha**. El resultado de la función tendrá el mismo tipo de dato que la columna. Si la columna es de **tipo numérico MIN()** devuelve el **valor menor** contenido en la columna, si la columna es de **tipo texto MIN()** devuelve el **primer valor** en **orden alfabético**, y si la columna es de **tipo fecha**, **MIN()** devuelve la **fecha más antigua** y **MAX()** la **fecha más reciente**.

La función **COUNT(nb columna)** cuenta el **número de valores** que hay **en la columna**, los datos de la columna pueden ser de **cualquier tipo**, y la función siempre devuelve un número entero. Si la columna contiene **valores nulos** esos valores **no se cuentan**, si en la columna aparece un **valor repetido**, lo **cuenta varias veces**.

COUNT(*) permite **contar filas** en vez de valores. Si la columna no contiene ningún valor nulo, **COUNT**(nbcolumna) y **COUNT(*)** devuelven el mismo resultado, mientras que si hay valores nulos en la columna, **COUNT(*)** cuenta también esos valores mientras que **COUNT**(nb columna) no los cuenta.

Ejemplo:
¿Cuántos empleados tenemos?

```
SELECT COUNT(numemp)
FROM empleados
```

o bien

```
SELECT COUNT(*)
FROM empleados
```

En este caso las dos sentencias devuelen el mismo resultado ya que la columna numemp no contiene valores nulos (es la clave principal de la tabla empleados).

¿Cuántos empleados tienen una oficina asignada?

```
SELECT COUNT(oficina)
FROM empleados
```

Esta sentencia por el contrario, nos devuelve el número de valores no nulos que se encuentran en la columna oficina de la tabla empleados, por lo tanto nos dice cuántos empleados tienen una oficina asignada.

Se pueden combinar varias funciones de columna en una expresión pero no se pueden anidar funciones de columna, es decir:

```
SELECT (AVG(ventas) * 3) + SUM(cuota)
FROM ...
```

es correcto

```
SELECT AVG(SUM(ventas))
FROM ...
```

NO es correcto, no se puede incluir una función de columna dentro de una función de columna

Selección en el origen de datos.

Si queremos **eliminar** del origen de datos algunas **filas**, basta incluir la cláusula **WHERE** que ya conocemos después de la cláusula **FROM**.

Ejemplo: Queremos saber el acumulado de ventas de los empleados de la oficina 12.

```
SELECT SUM(ventas)
FROM empleados
WHERE oficina = 12
```

Origen múltiple.

Si los **datos** que necesitamos utilizar para obtener nuestro resumen **se encuentran** en **varias tablas**, formamos el origen de datos adecuado en la cláusula **FROM** como si fuera una consulta **multitabla** normal.

Ejemplo: Queremos obtener el importe total de ventas de todos los empleados y el mayor objetivo de las oficinas asignadas a los empleados:

```
SELECT SUM(empleados.ventas), MAX(objetivo)
FROM empleados LEFT JOIN oficinas ON empleados.oficina=oficinas.oficina
```

NOTA: combinamos empleados con oficinas por un **LEFT JOIN** para que aparezcan en el origen de datos todos los empleados incluso los que no tengan una oficina asignada, así el origen de datos estará formado por una tabla con tantas filas como empleados hayan en la tabla empleados, con los datos de cada empleado y de la oficina a la que está asignado. De esta tabla sacamos la suma del campo ventas (importe total de ventas de todos los empleados) y el objetivo máximo. Observar que el origen de datos no incluye las oficinas que no tienen empleados asignados, por lo que esas oficinas no entran a la hora de calcular el valor máximo del objetivo.

<< Índice >> Pag. 4.1