GROUP BY

SQL ofrece la posibilidad de calcular funciones en grupos de tuplas usando la cláusula GROUP BY.

FORMATO

```
SELECT campo1
FROM tabla

WHERE condición

GROUP BY campo
ORDER BY campo
HAVING condición
```

GROUP BY

Agrupar por un campo o campos, cuando se ha puesto una función de agrupamiento en el SELECT todos los campos sin función deben estar agrupados. Si hay varios se separan por comas.

HAVING

Opcional. Condición tras agrupar

CARACTERÍSTICAS

1. Se evalúa la lista de selección para cada uno de los grupos, produciendo una sola fila en el resultado.

2. Si se incluye GROUP BY, en la lista de selección sólo puede haber:

- Constantes
- Funciones de grupo
- Expresiones idénticas a las del GROUP BY
- Expresiones que usan las expresiones anteriores y obtienen una sola fila por grupo.

3. ¡NO se pueden especificar columnas en la lista de select que no estén en las columnas de agrupamiento!

4. Si se utiliza WHERE con el GROUP BY, WHERE se aplica primero, después los grupos se forman de las filas restantes que satisfacen el predicado.

VISTAS

Una vista es la implementación de un esquema externo, según la Arquitectura ANSI/SPARC, en el Modelo Relacional.

Una vista
- es una tabla *virtual*,
- no hay ninguna tupla que pertenezca a ella,
- el SGBD sólo conoce su definición.

Una vista se define utilizando al instrucción create view, se le da un nombre a la vista y se indica la consulta que la va a calcular.

FORMATO

CREATE VIEW *NombreVista* [(*Nombre de Atributos*)] AS
<*Sentencia de Consulta*>;

CARCTERÍSTICAS

1. Cuando se utiliza la vista (quizás como parte de alguna sentencia SELECT) es cuando el SGBD realiza la consulta que define la vista y presenta aquellas tuplas que forman parte de la vista.

2. Las actualizaciones en las vistas están muy restringidas por los problemas que pueden originar en la consistencia de la información de la base de datos.

3. Los alias nos permiten renombrar todas las columnas de la vista. La consulta nos permite construir una visión de usuario tan compleja como queramos.

4. Si la o las tablas originales desde donde se creo la vista es o son actualizada(s), la vista automáticamente se actualiza.

Borrado de vistas

Formato

```
DROP VIEW NombreVista [RESTRICT | CASCADE]
```

Causa que la definición de la vista sea eliminada de la base de datos.

Ejemplo:

drop view nom_autores

Opciones de Borrado

Con CASCADE, se suprimen todos los objetos subordinados relacionados; es decir, cualquiera vista definida sobre vistas siendo eliminada.

Con RESTRICT (default), si algunos otros objetos dependen para su existencia de la existencia de la vista siendo eliminada, se rechaza el comando.

Ver todas las vistas definidas:

```
SELECT VIEW_NAME FROM USER_VIEWS;
```

Descripción de una Vista.

De forma parecida a como se obtiene la estructura de una tabla se puede obtener la de una vista:

DESC nombre_vista

EJEMPLOS

Dadas las siguientes tablas

AUTOR

COD_AUTOR	NOMBRE_AUTOR	FECHA_NACIMIENTO	NACIONALIDAD
10	Alejandro Dumas	10-10-1802	Francesa
12	Ruben Dario	05-12-1867	Nicaraguense
11	Mark Twain	11-06-1835	Estadounidense
13	Victor Hugo	04-10-1802	Francesa

LIBRO

COD_LIBRO	TITULO	ID_AUTOR	AÑO	NOM_EDITORIAL	PRECIO
102	Los tres mosqueteros	10	2003	Altazor	15700
101	El conde de Montecristo	10	2004	Antares	12500
103	Las Aventuras de Tom Sawyer	11	2002	Contrapunto	17800
104	cantos de Vida y Esperanza	12	2002	Antares	19800
105	Los Miserables	13	2003	Contrapunto	16700

EJEMPLOS DE GROUP BY

1. Determinar la cantidad de libros de cada editorial.

```
SELECT nom_editorial, count (cod_libro) as Cantidad_libros
FROM libro
GROUP BY nom_editorial
```

2. Determinar la suma de los precios de los libros de cada editorial

```
SELECT nom_editorial, sum(precio) AS Suma_Total_de_Precios
FROM libro
GROUP BY  nom_editorial
```

3. Determinar la cantidad de libros de cada editorial que no superen el precio de 18000.

```
SELECT nom_editorial, count(cod_libro) as Cantidad_libros
FROM libro
WHERE precio < 18000
GROUP BY nom_editorial
```

4. Determinar cuantas nacionalidades hay de cada autor ordenadas descendentemente.

```
SELECT nacionalidad, count(cod_autor) AS Cantidad_de_Autores
FROM Autor
GROUP BY nacionalidad
Order by nacionalidad desc
```

5. Determinar la suma de los precios de los libros de cada editorial, donde la suma no supere los 34000

```
SELECT nom_editorial, sum(precio) as suma_libros
FROM libro
GROUP BY nom_editorial
HAVING sum(precio) < 34000
```

EJEMPLOS DE VISTAS

1. Crear una vista que almacene la cantidad de libros de cada editorial que no superen el precio de 18000.

```
Create view cant_libros as
SELECT nom_editorial, count(cod_libro) as Cantidad_libros
FROM libro
WHERE precio < 18000
GROUP BY nom_editorial
```

Para verificar la vista

```
Select * from cant_libros
```

2. Crear una vista que almacene la suma de los precios de los libros de cada editorial que no superen los 34000

```
Create view  suma_precios(editorial, suma) as
SELECT nom_editorial, sum(precio)
FROM libro
GROUP BY nom_editorial
HAVING sum(precio) < 34000
```

Observaciones

1. Se Puede asignar un nombre propio (Alias) a cada columna en la vista.

2. Si se especifica la lista de los nombres de la columna, debe tener mismo número de items que el número de las columnas producidas por el *select*.

3. Si es omitida, cada columna toma el nombre de la columna correspondiente en el *select*

3. Crear una vista que almacene los nombres de los autores cuyos libros fueron editados por Antares

```
Create view  nom_autores(nombre) as
SELECT nombre_autor
FROM libro, autor
Where cod_autor =id_autor and nom_editorial = 'antares'
```

4. Crear una vista que almacene los nombres de los libros que fueron escritos por Alejandro Dumas

```
Create or replace view  nom_libro(nombre_libro) as
SELECT titulo
FROM libro, autor
Where cod_autor =id_autor and nombre_autor = 'Alejandro Dumas'
With read only
```

Observaciones

1. WITH READ ONLY. Asegura que no podrán ejecutarse operaciones DML sobre la vista.

2. OR REPLACE. Se utiliza por si la vista ya estuviera creada anteriormente. De esta forma una vista podrá ser modificada sin ser borrada.