Consultas resumen

Vamos a recordar la sintaxis para realizar una consulta con la sentencia SELECTen MySQL:

```
SELECT [DISTINCT] select_expr [, select_expr ...]
[FROM table_references]
[WHERE where_condition]
[GROUP BY {col_name | expr | position} [ASC | DESC], ... [WITH ROLLUP]]
[HAVING where_condition]
[ORDER BY {col_name | expr | position} [ASC | DESC], ...]
[LIMIT {[offset,] row_COUNT | row_COUNT OFFSET offset}]
```

Es muy importante conocer **en qué orden se ejecuta cada una de las cláusulas** que forman la sentencia SELECT. El orden de ejecución es el siguiente:

- Cláusula FROM.
- Cláusula WHERE(Es opcional, puede ser que no aparezca).
- Cláusula GROUP BY(Es opcional, puede ser que no aparezca).
- Cláusula HAVING(Es opcional, puede ser que no aparezca).
- Cláusula SELECT.
- Cláusula ORDER BY(Es opcional, puede ser que no aparezca).
- Cláusula LIMIT(Es opcional, puede ser que no aparezca).

En esta unidad vamos a trabajar con dos nuevas cláusulas GROUP BYy HAVING.

1.1 Funciones de agregación

Estas funciones realizan una operación específica sobre todas las filas de un grupo.Las funciones de agregación más comunes son:

Función	Descripción
`MAX(expr)`	Valor máximo del grupo
`MIN(expr)`	Valor mínimo del grupo
`AVG(expr)`	Valor medio del grupo
`SUM(expr)`	Suma de todos los valores del grupo
`COUNT(*)`	Número de filas que tiene el resultado de la consulta
`COUNT(columna)`	Número de valores no nulos que hay en esa columna

En la documentación oficial de MySQL puede encontrar una lista completa de todas las funciones de agregaciónque se pueden usar.

Importante: Las funciones de agregación sólo se pueden usar en las cláusulas SELECTY HAVING.

1.1.1 Diferencia entre COUNT(*)y COUNT(columna)

- COUNT(*): Calcula el número de filas que tiene el resultado de la consulta.
- COUNT(columna): Cuenta el número de valores no nulos que hay en esa columna.

Importante: Tenga en cuenta la diferencia que existe entre las funciones COUNT(*)y COUNT(columna), ya quedevolverán resultados diferentes cuando haya valores nulos en la columna que estamos usando en la función.

Ejemplos:

Supongamos que tenemos los siguientes valores en la tabla alumno:

id	nombre	apellido1	apellido2	fecha_nacimiento	es_repetidor	teléfono
1	María	Sánchez	Pérez	1990/12/01	no	NULL
2	Juan	Sáez	Vega	1998/04/02	no	618253876
3	Pepe	Ramírez	Gea	1988/01/03	no	NULL
4	Lucía	López	Ruiz	1993/06/13	sí	678516294

La consulta:

```
SELECT COUNT(teléfono)
FROM alumno;
```

devolverá:

COUNT(teléfono)
2

mientras que la consulta:

```
SELECT COUNT(*)
FROM alumno;
```

COUNT(*)
4

1.1.2 Contar valores distintos COUNT (DISTINCT columna)

Supongamos que tenemos los siguientes valores en la tabla producto:

id	nombre	precio	código_fabricante
1	Disco duro SATA3 1TB	86	5
2	Memoria RAM DDR4 8GB	120	4
3	Disco SSD 1 TB	150	5
4	GeForce GTX 1050Ti	185	5

Y nos piden calcular el número de valores distintos de código de fabricante que aparecen en la tabla producto.

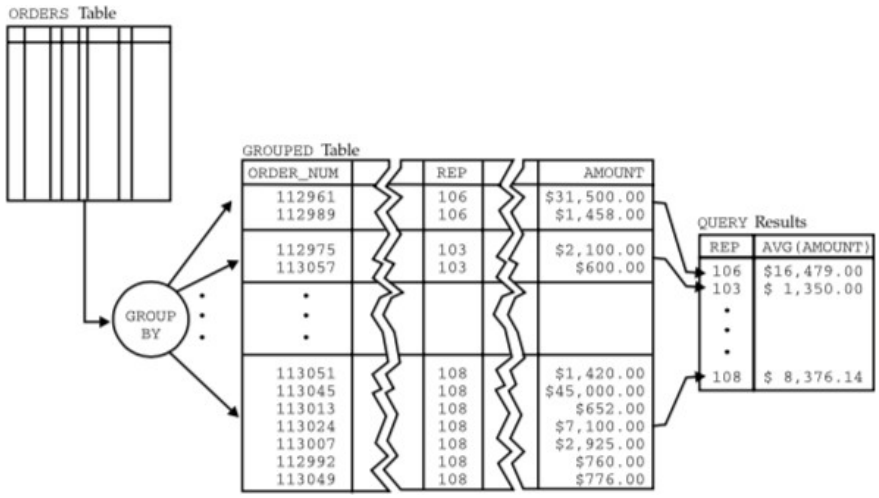
```
SELECT COUNT(DISTINCT código_fabricante)
FROM producto;
```

Esta consulta devolverá:

COUNT(DISTINCT código_fabricante)
2

1.2 Agrupamiento de filas (GROUP BY)

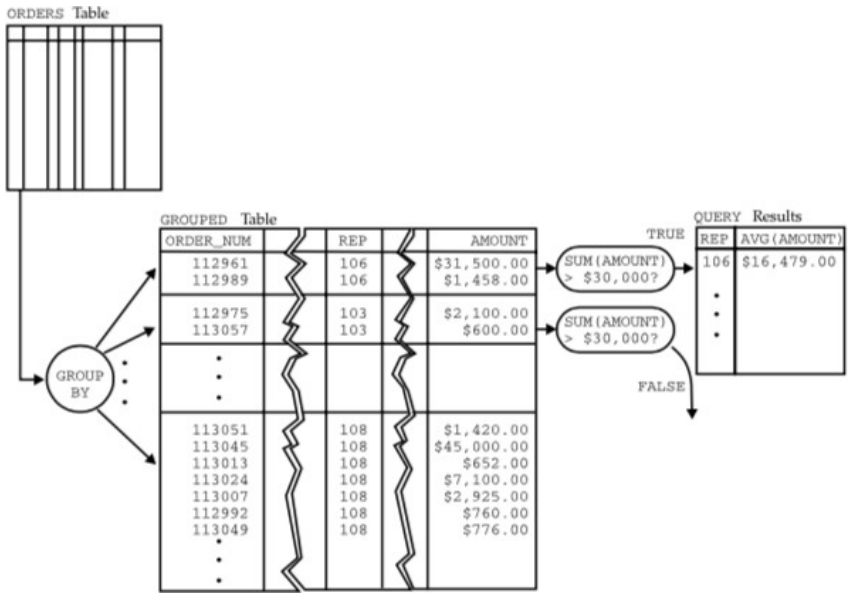
La cláusula GROUP BYnos permite crear **grupos de filas** que tienen los mismos valores en las columnas por lasque se desea agrupar.



En la documentación oficial de MySQL puede consultar **la lista de modificadores que puede utilizar con GROUP BY**.

1.3 Condición de agrupamiento (HAVING)

La cláusula HAVINGnos permite crear **filtros** sobre los grupos de filas que tienen los mismos valores en



1.4 Ejemplo de agrupamiento de filas (GROUP BY) con condición deagrupamiento (HAVING)

Consultas resumen

1

```
SELECT fabricante.nombre, AVG(producto.precio)
FROM producto INNER JOIN fabricante
ON producto.codigo_fabricante = fabricante.codigo
WHERE fabricante.nombre != 'Seagate'
GROUP BY fabricante.codigo
HAVING AVG(producto.precio) >= 150
```

Tabla: producto

codigo	nombre	precio	codigo_fabricante
1	Portátil A		1
2	Monitor 24		2
3	Disco SSD 1 TB		3
4	Impresora		4
5	Monitor 27		2
6	Portátil B		1

Tabla: fabricante

codigo	nombre
1	Lenovo
2	Asus
3	Seagate
4	HP

El **resultado de la operación INNER JOIN** es:

producto. codigo	producto. nombre	producto. precio	producto. codigo_fabricante	fabricante. codigo	fabricante. nombre
1	Portátil A	599	1	1	Lenovo
2	Monitor 24	203	2	2	Asus
3	Disco SSD 1 TB	115	3	3	Seagate
4	Impresora	49	4	4	HP
5	Monitor 27	242	2	2	Asus
6	Portátil B	320	1	1	Lenovo

2

```
SELECT fabricante.nombre, AVG(producto.precio)
FROM producto INNER JOIN fabricante
ON producto.codigo_fabricante = fabricante.codigo
WHERE fabricante.nombre != 'Seagate'
GROUP BY fabricante.codigo
HAVING AVG(producto.precio) >= 150
```

producto. codigo	producto. nombre	producto. precio	producto. codigo_fabricante	fabricante. codigo	fabricante. nombre	
1	Portátil A	599	1	1	Lenovo	✓
2	Monitor 24	203	2	2	Asus	✓
~~3~~	~~Disco SSD 1 TB~~	~~115~~	~~3~~	~~3~~	~~Seagate~~	X
4	Impresora	49	4	4	HP	✓
5	Monitor 27	242	2	2	Asus	✓
6	Portátil B	320	1	1	Lenovo	✓

El **resultado después de aplicar el filtro del WHERE** es:

producto. codigo	producto. nombre	producto. precio	producto. codigo_fabricante	fabricante. codigo	fabricante. nombre
1	Portátil A	599	1	1	Lenovo
2	Monitor 24	203	2	2	Asus
4	Impresora	49	4	4	HP
5	Monitor 27	242	2	2	Asus
6	Portátil B	320	1	1	Lenovo

3

```
SELECT fabricante.nombre, AVG(producto.precio)
FROM producto INNER JOIN fabricante
ON producto.codigo_fabricante = fabricante.codigo
WHERE fabricante.nombre != 'Seagate'
GROUP BY fabricante.codigo
HAVING AVG(producto.precio) >= 150
```

producto.codigo	producto.nombre	producto.precio	producto.codigo_fabricante	fabricante.codigo	fabricante.nombre
1	Portátil A	599	**1**	1	Lenovo
6	Portátil B	320	**1**	1	Lenovo
2	Monitor 24	203	**2**	2	Asus
5	Monitor 27	242	**2**	2	Asus
4	Impresora	49	**4**	4	HP

4

```
SELECT fabricante.nombre, AVG(producto.precio)
FROM producto INNER JOIN fabricante
ON producto.codigo_fabricante = fabricante.codigo
WHERE fabricante.nombre != 'Seagate'
GROUP BY fabricante.codigo
HAVING AVG(producto.precio) >= 150
```

producto.codigo	producto.nombre	producto.precio	producto.codigo_fabricante	fabricante.codigo	fabricante.nombre
1	Portátil A	599	**1**	1	Lenovo
6	Portátil B	320	**1**	1	Lenovo

459.5

AVG(producto.precio) >= 150 ✓

producto.codigo	producto.nombre	producto.precio	producto.codigo_fabricante	fabricante.codigo	fabricante.nombre
2	Monitor 24	203	**2**	2	Asus
5	Monitor 27	242	**2**	2	Asus

222.5

AVG(producto.precio) >= 150 ✓

producto.codigo	producto.nombre	producto.precio	producto.codigo_fabricante	fabricante.codigo	fabricante.nombre
4	Impresora	49	**4**	4	HP

49

AVG(producto.precio) >= 150 X

El **resultado después de aplicar el filtro del HAVING** es:

producto.codigo	producto.nombre	producto.precio	producto.codigo_fabricante	fabricante.codigo	fabricante.nombre
1	Portátil A	599	**1**	1	Lenovo
6	Portátil B	320	**1**	1	Lenovo
2	Monitor 24	203	**2**	2	Asus
5	Monitor 27	242	**2**	2	Asus

5

```
SELECT fabricante.nombre, AVG(producto.precio)
FROM producto INNER JOIN fabricante
ON producto.codigo_fabricante = fabricante.codigo
WHERE fabricante.nombre != 'Seagate'
GROUP BY fabricante.codigo
HAVING AVG(producto.precio) >= 150
```

fabricante.nombre	AVG(producto.precio)
Lenovo	459.5
Asus	222.5

Errores comunes

2.1 Error al contar el número de filas distintas

Ejemplo: Cuenta el número fabricantes distintos que aparecen en la tabla producto.

Consulta incorrecta.

```
SELECT DISTINCT COUNT(codigo_fabricante)
FROM producto;
```

Consulta correcta.

```
SELECT COUNT(DISTINCT codigo_fabricante)
FROM producto;
```

2.2 Error al intentar utilizar una función de agregación en la cláusula WHERE

Las funciones de agregación sólo se pueden usar en las cláusulas SELECTY HAVING, por lo tanto si intenta haceruso de una función de agregación en la cláusula WHEREobtendrá un error.

Ejemplo: Devuelve un listado con los productos que tienen un precio superior al precio medio de todos losartículos que existen en la tabla productos.

Consulta incorrecta.

```
SELECT *
FROM producto
WHERE precio > AVG(precio);
```

Consulta correcta.

```
SELECT *
FROM producto
WHERE precio > (SELECT AVG(precio) FROM producto);
```

Error al usar COUNT(*)y COUNT(columna)al hacer un LEFTJOIN

Ejemplo: Devuelve un listado con el número de productos que tiene cada fabricante. El listado debe incluiraquellos fabricantes que no tienen productos asociados indicando que tienen 0productos.

En este caso es necesario realizar un LEFT JOINcon las tablas fabricantey producto. Por ejemplo, al ejecutar la siguiente consulta podemos ver que los fabricantes Huaweiy Xiaomino tienen productos asociados.

```
SELECT fabricante.nombre, producto.codigo
FROM fabricante LEFT JOIN producto
ON producto.codigo_fabricante = fabricante.codigo
ORDER BY fabricante.codigo;
```

nombre	codigo
...	...
Crucial	2
Crucial	5
Gigabyte	4
Huawei	NULL
Xiaomi	NULL

Para contar el número de productos que tiene cada fabricante es necesario realizar un GROUP BYpor el códigodel fabricante y contar el número de filas que tiene cada uno de los grupos que hemos creado. Pero debemostener cuidado porque obtendremos resultados diferentes al contar el número de filas de cada grupo con COUNT (*) y COUNT(producto.codigo). La opción correcta es hacer uso de COUNT(producto.codigo) porque contará aquellas filas que tienen un valor distinto de NULL en la columna producto.codigo.

Consulta incorrecta.

```
SELECT fabricante.nombre, COUNT(*)
FROM fabricante LEFT JOIN producto
ON producto.codigo_fabricante = fabricante.codigo
GROUP BY fabricante.codigo
ORDER BY 2 DESC;
```

nombre	COUNT(*)
Lenovo	2
Asus	2
Crucial	2
Hewlett-Packard	2
Seagate	1
Gigabyte	1
Samsung	1
Xiaomi	1
Huawei	1

Consulta correcta.

```
SELECT fabricante.nombre, COUNT(producto.codigo)
FROM fabricante LEFT JOIN producto
ON producto.codigo_fabricante = fabricante.codigo
GROUP BY fabricante.codigo
ORDER BY 2 DESC;
```

nombre	COUNT(*)
Lenovo	2
Asus	2
Crucial	2
Hewlett-Packard	2
Seagate	1
Gigabyte	1
Samsung	1

nombre	COUNT(*)
Xiaomi	0
Huawei	0