|  |  |
| --- | --- |
| **33 - Contar registros (count)** |  |

Existen en MySQL funciones que nos permiten contar registros, calcular sumas, promedios, obtener valores máximos y mínimos. Veamos algunas de ellas.

Imaginemos que nuestra tabla "libros" contiene muchos registros. Para averiguar la cantidad sin necesidad de contarlos manualmente usamos la función "count()":

select count(\*) from Libro;

La función "count()" cuenta la cantidad de registros de una tabla, incluyendo los que tienen valor nulo.

Para saber la cantidad de libros de la editorial "Planeta" tipeamos:

select count(\*) from Libro

where editorial='Planeta';

También podemos utilizar esta función junto con la clausula "where" para una consulta más específica. Por ejemplo, solicitamos la cantidad de libros que contienen la cadena "Borges":

select count(\*) from Libro

where autor like '%Borges%';

Para contar los registros que tienen precio (sin tener en cuenta los que tienen valor nulo), usamos la función "count()" y en los paréntesis colocamos el nombre del campo que necesitamos contar:

select count(precio) from Libro;

Note que "count(\*)" retorna la cantidad de registros de una tabla (incluyendo los que tienen valor "null") mientras que "count(precio)" retorna la cantidad de registros en los cuales el campo "precio" no es nulo. No es lo mismo. "count(\*)" cuenta registros, si en lugar de un asterisco colocamos como argumento el nombre de un campo, se contabilizan los registros cuyo valor en ese campo no es nulo.

Tenga en cuenta que no debe haber espacio entre el nombre de la función y el paréntesis, porque puede confundirse con una referencia a una tabla o campo. Las siguientes sentencias son distintas:

select count(\*) from Libro;

select count (\*) from Libro;

La primera es correcta, la segunda incorrecta.

**Servidor de MySQL instalado en forma local.**

Ingresemos al programa "Workbench" y ejecutemos el siguiente bloque de instrucciones SQL para probar la función 'count':

drop table if exists Libro;

create table Libro(

codigo int unsigned auto\_increment,

titulo varchar(40) not null,

autor varchar(30),

editorial varchar(15),

precio decimal(5,2) unsigned,

cantidad mediumint unsigned,

primary key(codigo)

);

insert into Libro (titulo,autor,editorial,precio,cantidad)

values('El aleph','Borges','Planeta',15,100);

insert into Libro (titulo,autor,editorial,precio,cantidad)

values('Martin Fierro','Jose Hernandez','Emece',22.20,200);

insert into Libro (titulo,autor,editorial,precio,cantidad)

values('Antologia poetica','J.L. Borges','Planeta',40,150);

insert into Libro (titulo,autor,editorial,precio,cantidad)

values('Aprenda PHP','Mario Molina','Emece',18.20,200);

insert into Libro (titulo,autor,editorial,precio,cantidad)

values('Cervantes y el quijote','Bioy Casares- J.L. Borges','Paidos',36.40,100);

insert into Libro (titulo,autor,editorial,precio,cantidad)

values('Manual de PHP', 'J.C. Paez', 'Paidos',30.80,120);

insert into Libro (titulo,autor,editorial,precio,cantidad)

values('Harry Potter y la piedra filosofal','J.K. Rowling','Paidos',45.00,50);

insert into Libro (titulo,autor,editorial,precio,cantidad)

values('Harry Potter y la camara secreta','J.K. Rowling','Paidos',46.00,100);

insert into Libro (titulo,autor,editorial,precio,cantidad)

values('Alicia en el pais de las maravillas','Lewis Carroll','Paidos',null,200);

select count(\*) from Libro;

select count(\*) from Libro

where editorial='Planeta';

select count(\*) from Libro where autor like '%Borges%';

select count(precio) from Libro;

|  |  |
| --- | --- |
| **34 - Funciones de agrupamiento (count - max - min - sum - avg)** |  |

Existen en MySQL funciones que nos permiten contar registros, calcular sumas, promedios, obtener valores máximos y mínimos. Ya hemos aprendido "count()", veamos otras.

La función "sum()" retorna la suma de los valores que contiene el campo especificado. Por ejemplo, queremos saber la cantidad de Libro que tenemos disponibles para la venta:

select sum(cantidad) from Libro;

También podemos combinarla con "where". Por ejemplo, queremos saber cuántos Libro tenemos de la editorial "Planeta":

select sum(cantidad) from Libro

where editorial ='Planeta';

Para averiguar el valor máximo o mínimo de un campo usamos las funciones "max()" y "min()" respectivamente. Ejemplo, queremos saber cuál es el mayor precio de todos los libros:

select max(precio) from Libro;

Queremos saber cuál es el valor mínimo de los libros de "Rowling":

select min(precio) from Libro

where autor like '%Rowling%';

La función avg() retorna el valor promedio de los valores del campo especificado. Por ejemplo, queremos saber el promedio del precio de los Libro referentes a "PHP":

select avg(precio) from Libro

where titulo like '%PHP%';

Estas funciones se denominan "funciones de agrupamiento" porque operan sobre conjuntos de registros, no con datos individuales.

Tenga en cuenta que no debe haber espacio entre el nombre de la función y el paréntesis, porque puede confundirse con una referencia a una tabla o campo. Las siguientes sentencias son distintas:

select count(\*) from Libro;

select count (\*) from Libro;

La primera es correcta, la segunda incorrecta.

|  |  |
| --- | --- |
| [Listado completo de tutoriales](https://www.tutorialesprogramacionya.com/)  **35 - Agrupar registros (group by)** |  |

Hemos aprendido que las funciones de agrupamiento permiten contar registros, calcular sumas y promedios, obtener valores máximos y mínimos. También dijimos que dichas funciones operan sobre conjuntos de registros, no con datos individuales.

Generalmente estas funciones se combinan con la sentencia "group by", que agrupa registros para consultas detalladas.

Queremos saber la cantidad de visitantes de cada ciudad, podemos tipear la siguiente sentencia:

select count(\*) from Visitante

where ciudad='Cordoba';

y repetirla con cada valor de "ciudad":

select count(\*) from Visitante

where ciudad='Alta Gracia';

select count(\*) from Visitante

where ciudad='Villa Dolores';

...

Pero hay otra manera, utilizando la cláusula "group by":

select ciudad, count(\*)

from Visitante

group by ciudad;

Entonces, para saber la cantidad de visitantes que tenemos en cada ciudad utilizamos la función "count()", agregamos "group by" y el campo por el que deseamos que se realice el agrupamiento, también colocamos el nombre del campo a recuperar.

La instrucción anterior solicita que muestre el nombre de la ciudad y cuente la cantidad agrupando los registros por el campo "ciudad". Como resultado aparecen los nombres de las ciudades y la cantidad de registros para cada valor del campo.

Para obtener la cantidad visitantes con teléfono no nulo, de cada ciudad utilizamos la función "count()" enviándole como argumento el campo "telefono", agregamos "group by" y el campo por el que deseamos que se realice el agrupamiento (ciudad):

select ciudad, count(telefono)

from Visitante

group by ciudad;

Como resultado aparecen los nombres de las ciudades y la cantidad de registros de cada una, sin contar los que tienen teléfono nulo. Recuerde la diferencia de los valores que retorna la función "count()" cuando enviamos como argumento un asterisco o el nombre de un campo: en el primer caso cuenta todos los registros incluyendo los que tienen valor nulo, en el segundo, los registros en los cuales el campo especificado es no nulo.

Para conocer el total de las compras agrupadas por sexo:

select sexo, sum(montocompra)

from Visitante

group by sexo;

Para saber el máximo y mínimo valor de compra agrupados por sexo:

select sexo, max(montocompra) from Visitante

group by sexo;

select sexo, min(montocompra) from Visitante

group by sexo;

Se pueden simplificar las 2 sentencias anteriores en una sola sentencia, ya que usan el mismo "group by":

select sexo, max(montocompra),

min(montocompra)

from Visitante

group by sexo;

Para calcular el promedio del valor de compra agrupados por ciudad:

select ciudad, avg(montocompra) from Visitante

group by ciudad;

Podemos agrupar por más de un campo, por ejemplo, vamos a hacerlo por "ciudad" y "sexo":

select ciudad, sexo, count(\*) from Visitante

group by ciudad,sexo;

También es posible limitar la consulta con "where".

Vamos a contar y agrupar por ciudad sin tener en cuenta "Cordoba":

select ciudad, count(\*) from Visitante

where ciudad<>'Cordoba'

group by ciudad;

Podemos usar las palabras claves "asc" y "desc" para una salida ordenada:

select ciudad, count(\*) from Visitante

group by ciudad desc;

**Servidor de MySQL instalado en forma local.**

Ingresemos al programa "Workbench" y ejecutemos el siguiente bloque de instrucciones SQL para analizar la cláusula 'group by':

drop table if exists visitantes;

create table Visitante(

nombre varchar(30),

edad tinyint unsigned,

sexo char(1),

domicilio varchar(30),

ciudad varchar(20),

telefono varchar(11),

montocompra decimal (6,2) unsigned

);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Susana Molina', 28,'f','Colon 123','Cordoba',null,45.50);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Marcela Mercado',36,'f','Avellaneda 345','Cordoba','4545454',0);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Alberto Garcia',35,'m','Gral. Paz 123','Alta Gracia','03547123456',25);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Teresa Garcia',33,'f','Gral. Paz 123','Alta Gracia','03547123456',0);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Roberto Perez',45,'m','Urquiza 335','Cordoba','4123456',33.20);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Marina Torres',22,'f','Colon 222','Villa Dolores','03544112233',25);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Julieta Gomez',24,'f','San Martin 333','Alta Gracia','03547121212',53.50);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Roxana Lopez',20,'f','Triunvirato 345','Alta Gracia',null,0);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Liliana Garcia',50,'f','Paso 999','Cordoba','4588778',48);

insert into Visitante (nombre,edad, sexo,domicilio,ciudad,telefono,montocompra)

values ('Juan Torres',43,'m','Sarmiento 876','Cordoba','4988778',15.30);

-- Para saber la cantidad de visitantes que tenemos de cada ciudad tipeamos:

select ciudad, count(\*)

from Visitante

group by ciudad;

-- Necesitamos conocer la cantidad visitantes con teléfono no nulo, de cada ciudad:

select ciudad, count(telefono)

from Visitante

group by ciudad;

-- Queremos conocer el total de las compras agrupadas por sexo:

select sexo, sum(montocompra) from Visitante

group by sexo;

-- Para obtener el máximo y mínimo valor de compra agrupados por sexo:

select sexo, max(montocompra) from Visitante

group by sexo;

select sexo, min(montocompra) from Visitante

group by sexo;

-- Se pueden simplificar las 2 sentencias anteriores en una sola sentencia, ya que usan el mismo "group by":

select sexo, max(montocompra),

min(montocompra)

from Visitante

group by sexo;

-- Queremos saber el promedio del valor de compra agrupados por ciudad:

select ciudad, avg(montocompra) from Visitante

group by ciudad;

-- Contamos los registros y agrupamos por 2 campos, "ciudad" y "sexo":

select ciudad, sexo, count(\*) from Visitante

group by ciudad,sexo;

-- Limitamos la consulta, no incluimos los visitantes de "Cordoba", contamos y agrupar por ciudad:

select ciudad, count(\*) from Visitante

where ciudad<>'Cordoba'

group by ciudad;

-- Usando la palabra clave "desc" obtenemos la salida ordenada en forma descendente:

select ciudad, count(\*) from Visitante

group by ciudad desc;

|  |  |
| --- | --- |
| **36 - Selección de un grupo de registros (having)** |  |

Así como la cláusula "where" permite seleccionar (o rechazar) registros individuales; la cláusula "having" permite seleccionar (o rechazar) un grupo de registros.

Si queremos saber la cantidad de libros agrupados por editorial usamos la siguiente instrucción ya aprendida:

select editorial, count(\*) from Libro

group by editorial;

Si queremos saber la cantidad de libros agrupados por editorial pero considerando sólo algunos grupos, por ejemplo, los que devuelvan un valor mayor a 2, usamos la siguiente instrucción:

select editorial, count(\*) from Libro

group by editorial

having count(\*)>2;

Se utiliza "having", seguido de la condición de búsqueda, para seleccionar ciertas filas retornadas por la cláusula "group by".

Veamos otros ejemplos. Queremos el promedio de los precios de los libros agrupados por editorial:

select editorial, avg(precio) from Libro

group by editorial;

Ahora, sólo queremos aquellos cuyo promedio supere los 25 pesos:

select editorial, avg(precio) from Libro

group by editorial

having avg(precio)>25;

En algunos casos es posible confundir las cláusulas "where" y "having". Queremos contar los registros agrupados por editorial sin tener en cuenta a la editorial "Planeta".

Analicemos las siguientes sentencias:

select editorial, count(\*) from Libro

where editorial<>'Planeta'

group by editorial;

select editorial, count(\*) from Libro

group by editorial

having editorial<>'Planeta';

Ambas devuelven el mismo resultado, pero son diferentes.

La primera, selecciona todos los registros rechazando los de editorial "Planeta" y luego los agrupa para contarlos. La segunda, selecciona todos los registros, los agrupa para contarlos y finalmente rechaza la cuenta correspondiente a la editorial "Planeta".

No debemos confundir la cláusula "where" con la cláusula "having"; la primera establece condiciones para la selección de registros de un "select"; la segunda establece condiciones para la selección de registros de una salida "group by".

Veamos otros ejemplos combinando "where" y "having".

Queremos la cantidad de libros, sin considerar los que tienen precio nulo, agrupados por editorial, sin considerar la editorial "Planeta":

select editorial, count(\*) from Libro

where precio is not null

group by editorial

having editorial<>'Planeta';

Aquí, selecciona los registros rechazando los que no cumplan con la condición dada en "where", luego los agrupa por "editorial" y finalmente rechaza los grupos que no cumplan con la condición dada en el "having".

Generalmente se usa la cláusula "having" con funciones de agrupamiento, esto no puede hacerlo la cláusula "where". Por ejemplo queremos el promedio de los precios agrupados por editorial, de aquellas editoriales que tienen más de 2 libros:

select editorial, avg(precio) from Libro

group by editorial

having count(\*) > 2;

Podemos encontrar el mayor valor de los libros agrupados por editorial y luego seleccionar las filas que tengan un valor mayor o igual a 30:

select editorial, max(precio) from Libro

group by editorial

having max(precio)>=30;

Esta misma sentencia puede usarse empleando un "alias", para hacer referencia a la columna de la expresión:

select editorial, max(precio) as 'mayor' from Libro

group by editorial

having mayor>=30;