

Principios básicos para realizar consultas

Hemos aprendido como crear una tabla con sus respectivos atributos, además comprendimos cual es la mejor forma de agregar registros (INSERT). Ahora, nos adentraremos en el mundo de las consultas para entender qué debemos tener en cuenta al momento de crearlas.

Leer – Conociendo al SELECT:

El SELECT es la sentencia que nos permite consultar registros de nuestras tablas. Con las diferentes palabras reservadas y condiciones podremos obtener o, mejor dicho, recuperar nuestra información para que podamos mostrarla o procesarla.

Arranquemos por conocer todo lo que puede tener como reservado la sentencia y como venimos afirmando en varios de los módulos previos, siempre es conveniente revisar el manual del motor de base de datos que estemos utilizando por alguna variante adicional.

La sentencia:

```
SELECT nombre_columna_1, nombre_columna_2 .....  
FROM [tablas]                (dominio de nuestra consulta)  
WHERE [condiciones]          (a nivel de registro)  
GROUP BY nombre_columna_1, nombre_columna_2 ... (Agrupaciones por  
columnas)  
HAVING [condiciones]         (a nivel de grupos)  
ORDER BY nombre_columna_1, nombre_columna_2 ...
```

Arranquemos por entender como está conformada la sentencia, y como la valida el motor de base de datos.

En principio, el motor de base de datos valida que el dominio que elegimos exista. Es decir, las tablas que están luego del *FROM*. Luego, validará que las condiciones, establecidas en el *WHERE*, sean correctas. A continuación, ordenará todos los registros que cumplen con las

condiciones basado en lo declarado en *ORDER BY*. Por último, valida que las columnas (y operaciones) luego del *SELECT* existan y procederá a mostrarlas

Ahora bien, cuando ponemos a continuación del *SELECT* nombre_columna_1, no necesariamente significa que solo pueda traer información de columnas (campos de nuestras tablas), también puede realizar operaciones aritméticas y demás.

Acá les dejamos algunos ejemplos simples:

```
[mysql> SELECT 1320+17;
```

Ejecutamos la siguiente sentencia y este es el resultado obtenido

```
[mysql> SELECT 1320+17;
+-----+
| 1320+17 |
+-----+
|      1337 |
+-----+
1 row in set (0.00 sec)
```

De esta misma forma puedo combinar diferentes operaciones, y como vemos, no estamos haciendo uso del *FROM*. Solo lo vamos a usar cuando requiramos traer algún dato de nuestras tablas.

Sigamos probando algunas variantes con operaciones antes de pasar a agregar las tablas:

```
[mysql> SELECT 1320+17, 1340-3, 7*191, 8022/6;
+-----+-----+-----+-----+
| 1320+17 | 1340-3 | 7*191 | 8022/6 |
+-----+-----+-----+-----+
|      1337 |      1337 |      1337 | 1337.0000 |
+-----+-----+-----+-----+
1 row in set (0.02 sec)
```

En este ejemplo ejecutamos más de una operación y las separamos por comas, de esta forma nos separa el resultado en diferentes columnas.

Aprovechamos para poner algo más ordenado y prolijo, ya que los nombres de las columnas, en el ejemplo anterior, no quedaron muy bien, aunque muestran qué operación realizamos.

En este ejemplo renombramos las columnas, utilizando la palabra reservada AS luego de definir lo que voy a mostrar y a continuación poniendo una palabra. Si queremos agregar como nombre de columna una frase (conviene que sea corta) debemos encerrarla entre comillas simples.

```
[mysql> SELECT 1+2 as "es una SUMA", 3*2 as Multiplicación;
+-----+-----+
| es una SUMA | Multiplicación |
+-----+-----+
|          3 |             6 |
+-----+-----+
1 row in set (0.00 sec)
```

Agregando tablas y campos al SELECT

Vamos a arrancar con lo más sencillo, listar todos los registros que tengo en una tabla.

```
SELECT *
FROM nombre_tabla;
```

El * lo utilizamos para decir que queremos ver todas las columnas que estén incluidas en nuestro dominio de la consulta (lo que está en el FROM).

Dada mi tabla clientes (Customer)

Customer
Address
City
Company
Country
CustomerId
Email
Fax
FirstName
LastName
Phone
PostalCode
State
SupportRepld

Por ejemplo, si quisiéramos listar todos los registros de mi tabla clientes (Customer) sería algo así:

```

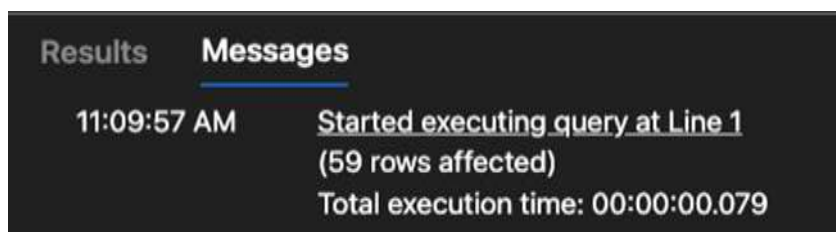
1  SELECT *
2  FROM CUSTOMER ;

```

Y como resultado obtendríamos algo similar a esto (en este caso, tenemos varios registros cargados).

Results	Messages	CustomerId	FirstName	LastName	Company	Address	City	State	Country
1		1	Luis	Gonçalves	Embraer - Empresa Brasile...	Av. Brigadeiro Faria Lima	São José dos Campos	SP	Braz
2		2	Leonie	Köhler	NULL	Theodor-Neuss-Straße 34	Stuttgart	NULL	Germ
3		3	François	Tremblay	NULL	1498 rue Bélanger	Montréal	QC	Can
4		4	Bjorn	Hansen	NULL	Ullevålsveien 14	Oslo	NULL	Norw
5		5	František	Wichterlová	JetBrains s.r.o.	Klanova 9/506	Prague	NULL	Czec
6		6	Helena	Holý	NULL	Rilská 3174/6	Prague	NULL	Czec
7		7	Astrid	Gruber	NULL	Rotenturmstraße 4, 1010 L	Vienne	NULL	Aust
8		8	Daan	Peeters	NULL	Grêtrystraat 63	Brussels	NULL	Belg
9		9	Kara	Nielsen	NULL	Sender Boulevard 51	Copenhagen	NULL	Denn
10		10	Eduardo	Martins	Woodstock Discos	Rua Dr. Faício Filho, 155	São Paulo	SP	Braz
11		11	Alexandre	Rocha	Banco do Brasil S.A.	Av. Paulista, 2022	São Paulo	SP	Braz
12		12	Roberto	Almeida	Riotur	Praça Pio X, 119	Rio de Janeiro	RJ	Braz
13		13	Fernanda	Ramos	NULL	Qe 7 Bloco G	Brasília	DF	Braz
14		14	Mark	Philips	Telus	8230 111 ST NW	Edmonton	AB	Can
15		15	Jennifer	Peterson	Rogers Canada	788 W Pender Street	Vancouver	BC	Can
16		16	Frank	Harris	Google Inc.	1600 Amphitheatre Parkway	Mountain View	CA	USA
17		17	Jack	Smith	Microsoft Corporation	1 Microsoft Way	Redmond	WA	USA

Y en la parte de mensajes vería información que puede ser muy útil.



Donde nos informa la cantidad de columnas que encontró (con las condiciones solicitadas) y el tiempo total de ejecución.

Consulta, pero con límite de registros:

Si queremos que nos traiga un límite de registros nos vamos a encontrar que dependiendo del motor puede variar:

A continuación, realizamos la consulta del ejemplo anterior referido a clientes, pero solo retornando los primeros 10 primeros registros.

En MySQL:

- SELECT *
- FROM Customer
- LIMIT 10

En SQL Server:

- SELECT TOP 10 *
- FROM Customer;

Así que en algunos casos nos conviene consultar en el manual de motor, o basta con googlear: “Cómo consulto una tabla con un límite de registros en [motor de base de datos]”.

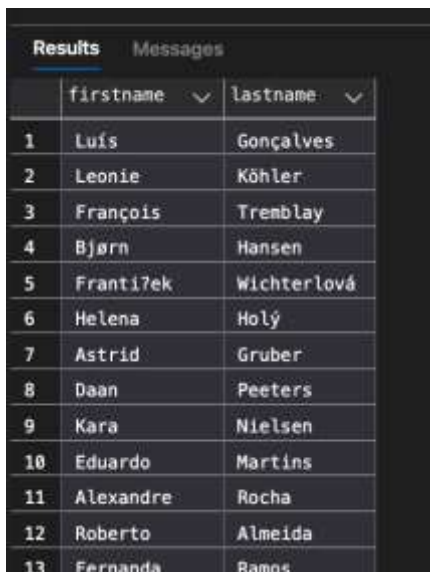
Avanzamos incorporando a los resultados de nuestra consulta, la posibilidad de mostrar columnas de las tablas.

Imaginemos que vamos a necesitar el nombre y apellido de nuestros clientes.

Bastaría con reemplazar el * de nuestra consulta inicial e incorporar separado por comas los campos que requerimos.

```
SELECT firstname, lastname  
FROM CUSTOMER;
```

Y el resultado sería el siguiente:



The screenshot shows a database interface with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with two columns: 'firstname' and 'lastname'. The table contains 13 rows of data, numbered 1 through 13 in the first column. The names are as follows:

	firstname	lastname
1	Luis	Gonçalves
2	Leonie	Köhler
3	François	Trenblay
4	Bjørn	Hansen
5	František	Wichterlová
6	Helena	Holý
7	Astrid	Gruber
8	Daan	Peeters
9	Kara	Nielsen
10	Eduardo	Martins
11	Alexandre	Rocha
12	Roberto	Almeida
13	Fernanda	Ramos

Como vimos antes, podemos cambiar el nombre a las columnas utilizando el AS en caso que fuera necesario.

Ahora hagamos lo mismo, pero ordenando la salida, vamos a hacer uso del ORDER BY, hay dos combinaciones posibles por cada columna que incorporemos:

- Ascendente.
- Descendente.

Cuando no lo aclaramos, el motor va a ordenar de manera ascendente, y si quisiéramos de manera descendente debemos incorporar la palabra reservada desc luego de la columna de esta forma

ORDER BY nombre_columna_1 DESC

Vamos a un ejemplo, consultar los nombres y apellidos de mis clientes, pero ordenado ascendente “Alfabéticamente” por apellido.

```
SELECT firstname, lastname
FROM CUSTOMER
ORDER BY lastname ASC;
```

El ASC no es necesario incorporarlo, ya que el motor lo toma por defecto.

```
SELECT firstname, lastname
FROM CUSTOMER
ORDER BY lastname;
```

Si comparamos con lo anterior obtenido podremos ver la diferencia:

Con el orden

Results Messages		
	firstname	lastname
1	Roberto	Almeida
2	Julia	Barnett
3	Camille	Bernard
4	Michelle	Brooks
5	Robert	Brown
6	Kathy	Chase
7	Richard	Cunningham
8	Marc	Dubois
9	João	Fernandes
10	Edward	Francis
11	Wyatt	Girard
12	Luís	Gonçalves
13	John	Gordon
14	Tim	Goyer
15	Patrick	Gray

Sin el orden

Results Messages		
	firstname	lastname
1	Luís	Gonçalves
2	Leonie	Köhler
3	François	Tremblay
4	Bjørn	Hansen
5	František	Wichterlová
6	Helena	Holý
7	Astrid	Gruber
8	Daan	Peeters
9	Kara	Nielsen
10	Eduardo	Martins
11	Alexandre	Rocha
12	Roberto	Almeida
13	Fernanda	Ramos

Comentando nuestras consultas:

Antes de seguir avanzando en aprender aún más sobre consultas a la base, es conveniente ver cómo realizar comentarios en mis consultas.

Siempre es recomendable dejar algún tipo de comentario para poder ver qué se buscaba con la consulta o el propósito de algunas partes de esta.

Hay dos tipos de comentarios:

- Por línea.
- Por bloque.

Por línea:

En el caso de línea nos es útil cuando queremos dejar en el código algún tipo de comentario corto y que solo afecte a una línea. Para este tipo de comentario vamos usar un doble menos "--".

Veamos un ejemplo anterior donde le agregamos un comentario.

```
SELECT  firstname, lastname
FROM CUSTOMER
ORDER BY lastname; -- Ordeno por apellido
```

El motor no va a interpretar cualquier texto que se encuentre luego del doble menos, es conveniente ponerlo al final de alguna sentencia. O incluso nos puede servir por si estamos realizando una prueba de una consulta y queremos comentar una línea y validar su resultado, por ejemplo.

```
SELECT  firstname, lastname
FROM CUSTOMER
-- ORDER BY lastname; -- Ordeno por apellido
```

En este último ejemplo el motor va a ignorar el orden.

Por bloque:

El modo de comentario por bloque nos va a servir para comentar varias líneas al mismo tiempo y que el motor lo ignore. Es ideal para cuando queremos describir lo que realiza

nuestra consulta o incluso poner que tablas involucra o datos adicionales. Para realizar este bloque comentado vamos a utilizar la barra de división seguido de un asterisco /* y para cerrar el comentario utilizaremos un asterisco y la barra de división */.

```
/* Creador: Rubén  
Fecha: Agosto/2021  
Propósito: Traer todos los clientes (Customer), solo nombre (firstname)  
y apellido (lastname) ordenado por apellido de forma ascendente.  
*/  
  
SELECT firstname, lastname  
FROM CUSTOMER  
ORDER BY lastname; -- Ordeno por apellido
```

En el ejemplo agregamos comentarios que podrían ser útiles para guardar un histórico de las consultas realizadas, permitiendo conocer el propósito de la consulta, creador y la fecha.

Incorporando algunas cosas adicionales al SELECT:

Ahora sí, avancemos con algunos agregados que podemos realizar con las columnas a mostrar. Vimos que podemos mostrar operaciones aritméticas como una columna adicional, esto mismo lo vamos a poder hacer con alguna columna.

Imaginemos que necesitamos un listado de las canciones que incluya el nombre, el valor actual y una columna con el valor aumentado en un 10%.

Esta es mi tabla de canciones (Track)

Track
AlbumId
Bytes
Composer
GenreId
MediaTypeId
Milliseconds
Name
TrackId
UnitPrice

Y para sacar el porcentaje de incremento puedo multiplicar por 1,10 y la consulta me quedaría así:

```
SELECT t.Name as Nombre, t.UnitPrice as PrecioActual, t.UnitPrice * 1.10 as Incremento
FROM Track t
```

Y el resultado se vería algo así:

Results		Messages		
	Nombre	PrecioActual	Incremento	
1	For Those About To Rock (...	0.99	1.0890	
2	Balls to the Wall	0.99	1.0890	
3	Fast As a Shark	0.99	1.0890	
4	Restless and Wild	0.99	1.0890	
5	Princess of the Dawn	0.99	1.0890	
6	Put The Finger On You	0.99	1.0890	
7	Let's Get It Up	0.99	1.0890	
8	Inject The Venom	0.99	1.0890	
9	Snowballed	0.99	1.0890	
10	Evil Walks	0.99	1.0890	
11	C.O.D.	0.99	1.0890	
12	Breaking The Rules	0.99	1.0890	
13	Night Of The Long Knives	0.99	1.0890	
14	Spellbound	0.99	1.0890	
15	Go Down	0.99	1.0890	
16	Dog Eat Dog	0.99	1.0890	
17	Let There Be Rock	0.99	1.0890	
18	Bad Boy Boogie	0.99	1.0890	

Es decir que podemos “operar” con las columnas de las tablas de la forma que necesitemos. Esto no significa que la información en nuestra tabla se modificó. Es solo para poder realizar algún tipo de informe.

Algo que también podemos hacer es concatenar caracteres o cadena de caracteres, puede resultar muy práctico cuando requerimos combinar columnas o agregar algún mensaje adicional en la salida.

Imaginemos el siguiente ejemplo: vimos que la tabla Clientes tiene separado el nombre del apellido, pero para realizar un envío masivo de correo necesitamos tener “Apellidos, Nombres” de cada registro. Deberíamos concatenar en una misma columna el campo LastName luego una coma y seguido de un espacio el campo FirstName.

Esto que vamos a realizar puede variar dependiendo el motor.

En SQL Server:

```
SELECT LastName + ', ' + Firstname as "Apellidos, Nombres"
FROM Customer
```

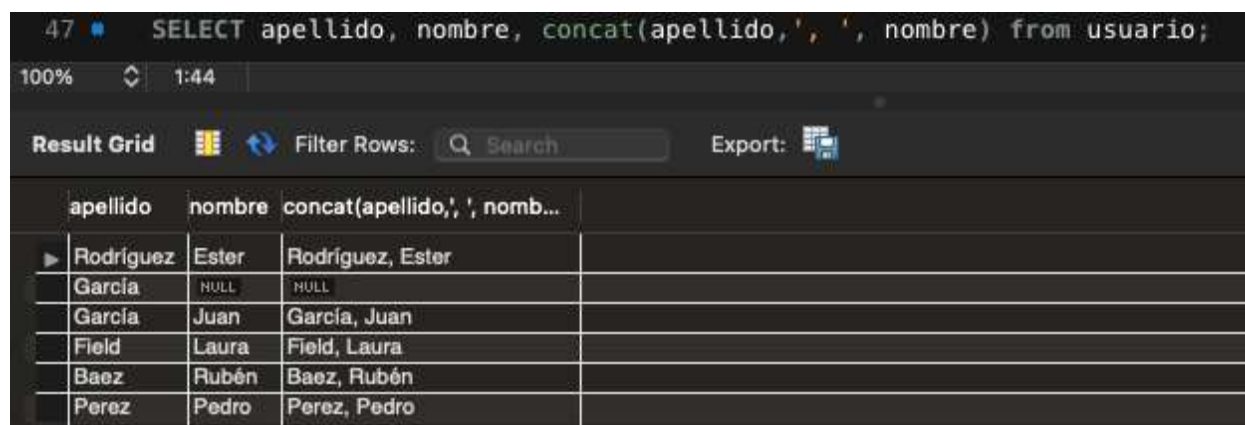
En MySQL:

Usando la función CONCAT:

```
SELECT CONCAT(LastName, ', ', Firstname) as "Apellidos, Nombres"
FROM Customer
```

Concat es una función que incorpora MySQL que nos permite concatenar caracteres, pero tiene un inconveniente si uno de los campos es NULL regresará algo así.

Dada mi tabla usuario:

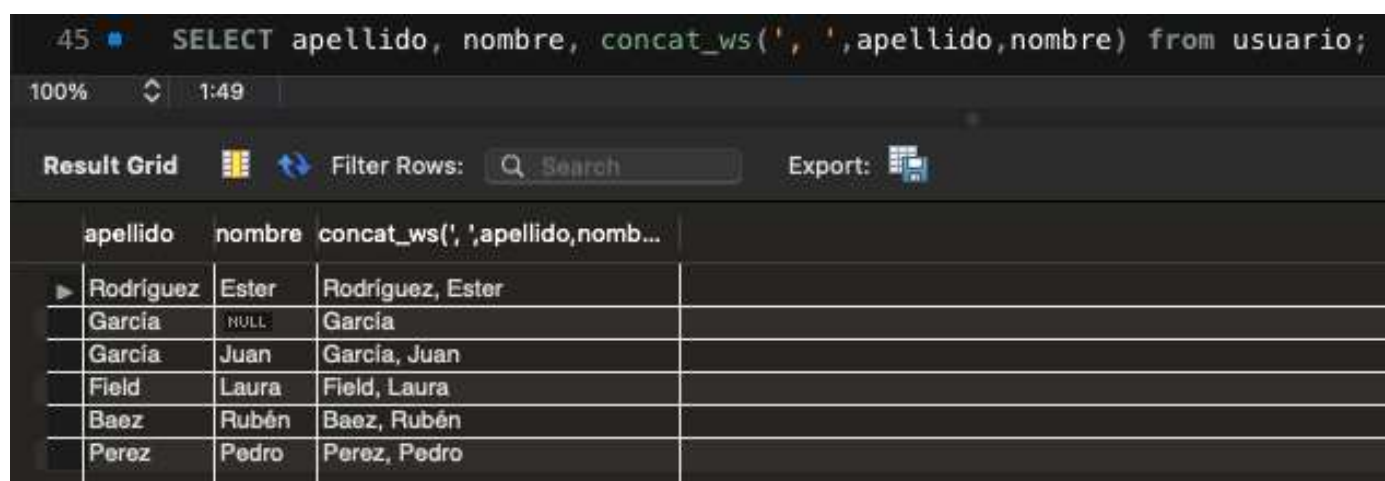


apellido	nombre	concat(apellido, ', ', nomb...
Rodríguez	Ester	Rodríguez, Ester
García	NULL	NULL
García	Juan	García, Juan
Field	Laura	Field, Laura
Baez	Rubén	Baez, Rubén
Perez	Pedro	Perez, Pedro

Como se puede ver en el segundo resultado no tengo guardado el nombre, y como consecuencia me regresa NULL.

Para salvar este error existe otra función que permite concatenar campos utilizando un separador, y el resultado es distinto.

Usando la función CONCAT_WS (With Separator - Concatenar con un separado).



```
45 SELECT apellido, nombre, concat_ws(',', apellido, nombre) from usuario;
```

100% 1:49

Result Grid Filter Rows: Search Export:

	apellido	nombre	concat_ws(',', apellido, nomb...
▶	Rodríguez	Ester	Rodríguez, Ester
	García	NULL	García
	García	Juan	García, Juan
	Field	Laura	Field, Laura
	Baez	Rubén	Baez, Rubén
	Perez	Pedro	Perez, Pedro

Como se puede ver, el resultado aún con alguno de los campos nulos es salvado y queda prolijo. Como estas, los motores suelen tener varias funciones que nos van a facilitar algunas cuestiones.

Hasta acá vimos algo de consultas básicas. En los próximos módulos vamos a avanzar incorporando condiciones y algunas otras cuestiones que van a colaborar para que nuestros desarrollos web tenga lo necesario.