# Obtener nombre de las columnas

Para obtener los nombres de las columnas de una tabla en MySQL, se puede utilizar la siguiente consulta SQL:

mysqlCopy code

SHOW COLUMNS FROM nombre\_de\_la\_tabla;

Reemplaza "nombre\_de\_la\_tabla" con el nombre de la tabla que desees consultar.

Si deseas mostrar los nombres de las columnas junto con otra información sobre las columnas, como el tipo de datos y si permiten valores nulos o no, también puedes utilizar la siguiente consulta:

mysqlCopy code

DESCRIBE nombre\_de\_la\_tabla;

De nuevo, reemplaza "nombre\_de\_la\_tabla" con el nombre de la tabla que desees consultar.

Estas consultas devolverán una tabla con información sobre las columnas de la tabla especificada, incluyendo el nombre de la columna.

# Tipos de datos en MySQL.

En MySQL, existen varios tipos de datos que se pueden utilizar para almacenar diferentes tipos de información. A continuación, se muestran algunos de los tipos de datos más comunes:

* **INTEGER**: Este tipo de dato se utiliza para almacenar números enteros. Hay varios subtipos de INTEGER, que varían en tamaño y capacidad de almacenamiento, como TINYINT, SMALLINT, MEDIUMINT, INT y BIGINT.
* **DECIMAL**: Este tipo de dato se utiliza para almacenar números decimales. Se puede especificar el número total de dígitos y la cantidad de dígitos a la derecha del punto decimal.
* **FLOAT y DOUBLE**: Estos tipos de datos se utilizan para almacenar números con decimales flotantes. FLOAT se utiliza para números más pequeños y DOUBLE para números más grandes.
* **CHAR y VARCHAR**: Estos tipos de datos se utilizan para almacenar cadenas de caracteres. CHAR se utiliza para cadenas de longitud fija, mientras que VARCHAR se utiliza para cadenas de longitud variable.
* **DATE, TIME y DATETIME**: Estos tipos de datos se utilizan para almacenar fechas y horas. DATE se utiliza para almacenar fechas, TIME se utiliza para almacenar horas y DATETIME se utiliza para almacenar fechas y horas juntas.
* **BOOLEAN**: Este tipo de dato se utiliza para almacenar valores booleanos, es decir, verdadero o falso.

Estos son solo algunos de los tipos de datos disponibles en MySQL. Existen otros tipos de datos, como BLOB, ENUM y SET, que se utilizan para casos específicos.

* **BLOB**: BLOB significa Binary Large Object y se utiliza para almacenar datos binarios grandes, como imágenes, archivos de audio y video, en la base de datos. Los datos BLOB se almacenan en un formato binario y se pueden recuperar y mostrar en su formato original.
* **ENUM**: ENUM es un tipo de datos que se utiliza para definir una lista de valores permitidos para una columna. Solo se pueden insertar valores que estén en la lista ENUM. Por ejemplo, si tienes una tabla de usuarios, puedes utilizar un campo ENUM para especificar el género como "Masculino", "Femenino" o "Otro". Esto garantiza que solo se puedan insertar valores válidos en el campo y facilita la validación de datos.
* **SET**: SET es similar a ENUM en que se utiliza para definir una lista de valores permitidos para una columna. Sin embargo, a diferencia de ENUM, SET permite múltiples valores seleccionados de la lista. Por ejemplo, si tienes una tabla de tareas, puedes utilizar un campo SET para especificar las categorías de las tareas, como "Trabajo", "Familia" y "Personal". Esto permite que una tarea se asigne a múltiples categorías al mismo tiempo.

En resumen, BLOB se utiliza para almacenar datos binarios grandes, ENUM se utiliza para definir una lista de valores permitidos para una columna y SET se utiliza para definir una lista de valores permitidos para una columna con múltiples selecciones posibles.

## Diferencia entre DECIMAL, FLOAT Y DOUBLE.

La principal diferencia entre los tipos de columnas DECIMAL, FLOAT y DOUBLE en MySQL es la forma en que se almacenan y representan los números en la base de datos.

* **DECIMAL**: El tipo DECIMAL se utiliza para almacenar números decimales de precisión fija. La precisión se define por el número total de dígitos que se pueden almacenar (precisión) y la cantidad de dígitos que se pueden almacenar a la derecha del punto decimal (escala). DECIMAL se almacena como una cadena de caracteres y se utiliza para valores que requieren una precisión exacta, como valores monetarios.
* **FLOAT**: El tipo FLOAT se utiliza para almacenar números con decimales flotantes. FLOAT es un tipo de punto flotante que se almacena en 4 bytes y tiene una precisión de aproximadamente 7 dígitos. Debido a que FLOAT utiliza una representación binaria, puede haber problemas de precisión al almacenar valores decimales precisos.
* **DOUBLE**: El tipo DOUBLE se utiliza para almacenar números con decimales flotantes con mayor precisión que FLOAT. DOUBLE es un tipo de punto flotante que se almacena en 8 bytes y tiene una precisión de aproximadamente 15 dígitos. Al igual que FLOAT, DOUBLE utiliza una representación binaria, por lo que puede haber problemas de precisión al almacenar valores decimales precisos.

En resumen, la principal diferencia entre los tipos DECIMAL, FLOAT y DOUBLE es la forma en que se almacenan y representan los números en la base de datos. DECIMAL se utiliza para valores que requieren una precisión exacta, mientras que FLOAT y DOUBLE se utilizan para valores que no requieren una precisión exacta pero requieren una representación más compacta. En general, DECIMAL es más adecuado para valores monetarios o científicos precisos, mientras que FLOAT y DOUBLE son más adecuados para valores de ingeniería o científicos que no requieren precisión exacta.

## Ejemplo de ENUM.

Supongamos que tienes una tabla "usuarios" en tu base de datos que almacena información de los usuarios y deseas incluir una columna que indique el nivel de acceso de cada usuario. Puedes utilizar el tipo de datos ENUM para definir una lista de valores permitidos para esta columna. Por ejemplo:

CREATE TABLE usuarios

( id INT PRIMARY KEY,

nombre VARCHAR(50),

email VARCHAR(100),

nivel\_acceso ENUM('usuario', 'editor', 'administrador')

);

En este ejemplo, se crea una tabla "usuarios" con una columna "nivel\_acceso" que solo permite los valores "usuario", "editor" o "administrador". Si intentas insertar un valor que no esté en esta lista, se producirá un error.

Puedes insertar datos en la tabla utilizando los valores permitidos en la lista ENUM, por ejemplo:

INSERT INTO usuarios (id, nombre, email, nivel\_acceso) VALUES (1, 'Juan', 'juan@example.com', 'usuario');

En este caso, se inserta un usuario con nivel de acceso "usuario". También puedes actualizar los datos de la tabla utilizando los valores permitidos en la lista ENUM, por ejemplo:

UPDATE usuarios SET nivel\_acceso = 'editor' WHERE id = 1;

En este caso, se actualiza el nivel de acceso del usuario con ID 1 a "editor".

## Tipos de dato fecha (DATE, TIME, DATETIME).

Los tipos de fechas en MySQL, DATE, TIME y DATETIME, se utilizan para almacenar información de fecha y hora en la base de datos.

* **DATE**: El tipo DATE se utiliza para almacenar valores de fecha en formato 'YYYY-MM-DD'. Por ejemplo, '2023-03-11' representa el 11 de marzo de 2023. El tipo DATE se utiliza para almacenar fechas sin la hora del día.
* **TIME**: El tipo TIME se utiliza para almacenar valores de hora en formato 'HH:MM:SS'. Por ejemplo, '09:30:00' representa las 9:30 AM. El tipo TIME se utiliza para almacenar horas del día sin la fecha.
* **DATETIME**: El tipo DATETIME se utiliza para almacenar valores de fecha y hora en formato 'YYYY-MM-DD HH:MM:SS'. Por ejemplo, '2023-03-11 09:30:00' representa el 11 de marzo de 2023 a las 9:30 AM. El tipo DATETIME se utiliza para almacenar fechas y horas del día.

Para crear una tabla con un campo de fecha, hora o fecha y hora, se utiliza la sintaxis:

mysqlCopy code

CREATE TABLE tabla\_ejemplo (

id INT PRIMARY KEY,

fecha DATE,

hora TIME,

fecha\_hora DATETIME

);

Para insertar datos en la tabla, se utiliza la sintaxis:

mysqlCopy code

INSERT INTO tabla\_ejemplo (id, fecha, hora, fecha\_hora)

VALUES (1, '2023-03-11', '09:30:00', '2023-03-11 09:30:00');

Para recuperar datos de la tabla, se utiliza la sintaxis:

mysqlCopy code

SELECT \* FROM tabla\_ejemplo;

La consulta anterior mostrará todos los registros de la tabla, incluyendo los valores de fecha, hora y fecha y hora.

En resumen, los tipos de fechas en MySQL se utilizan para almacenar información de fecha y hora en la base de datos. DATE se utiliza para almacenar fechas, TIME se utiliza para almacenar horas del día y DATETIME se utiliza para almacenar fechas y horas del día.

### ¿Y si quiero especificar las horas en formato 24Horas?

Sí, se puede especificar en formato de 24 horas en MySQL. En lugar de usar el formato de 12 horas con AM/PM, se puede utilizar el formato de 24 horas que va desde 00:00:00 hasta 23:59:59.

Por ejemplo, si se quiere insertar la hora 14:30:00 en una tabla, se puede hacer de la siguiente manera:

mysqlCopy code

INSERT INTO tabla\_ejemplo (id, fecha, hora, fecha\_hora) VALUES (1, '2023-03-11', '14:30:00', '2023-03-11 14:30:00');

De esta manera, se inserta la hora 14:30:00 en el campo 'hora' y 'fecha\_hora' en formato de 24 horas.

Al recuperar datos de la tabla, se puede utilizar la función **TIME\_FORMAT()** para formatear la hora en el formato deseado, por ejemplo:

mysqlCopy code

SELECT id, DATE\_FORMAT(fecha, '%Y-%m-%d') AS fecha,

TIME\_FORMAT(hora, '%H:%i:%s') AS hora\_24h,

DATE\_FORMAT(fecha\_hora, '%Y-%m-%d %H:%i:%s') AS fecha\_hora\_24h

FROM tabla\_ejemplo;

La consulta anterior muestra los campos de fecha, hora y fecha y hora en formato de 24 horas utilizando la función **TIME\_FORMAT()** y **DATE\_FORMAT()**.

En resumen, se puede especificar horas por encima de las 12 AM en formato de 24 horas en MySQL utilizando el formato 'HH:MM:SS' y se pueden formatear las horas en el formato deseado utilizando las funciones **TIME\_FORMAT()** y **DATE\_FORMAT()**.