# AE 1 Proyecto Grupal Bases de datos relacionales

### Integrantes: Valeria Sanchez, Fernando Quezada, Alejandro Lanas y Carlos Galleguillos

## Respondan las siguientes preguntas en grupo:

#### ¿Qué tipo de datos les permite guardar fechas? ¿Y horas?

presentar el tipo de datos DATE para la introducción de fechas en MySQL.

En ocasiones necesitarás almacenar fechas en tus tablas. La fecha de nacimiento de tus contactos, el vencimiento de una tarea, el día en que adquiriste un determinado libro de tu biblioteca, son algunos ejemplos de usos típicos.

MySQL tiene unos cuantos tipos para manejar fechas y horas. Hoy voy a mostrarte uno especializado en las primeras (es decir, sin incluir información horaria): el tipo DATE.

A la hora de trabajar con fechas debes tener en cuenta el formato que MySQL maneja, tanto para la introducción como para la visualización de datos.

Se indica primero el año, seguido del mes y terminando por el día, como en el siguiente ejemplo:

'2012-10-25'  -->  25 de octubre de 2012

Quédate con la copla: 'AAAA-MM-DD'. Si introduces una fecha errónea, MySQL se la tragará, pero la guardará como '0000-00-00'.

Cuando digo fecha errónea no me refiero únicamente a que cruces los campos. Fíjate en esto:

'2012-02-29'

es perfectamente válido, pues este año es bisiesto. Pero lo siguiente es erróneo,

'2011-02-29'

y MySQL lo almacenaría en la tabla como '0000-00-00'.

Veremos más adelante que podemos configurar MySQL para que no sea tan estricto a la hora de comprobar fechas. Pero no olvides que, por defecto, no te pasará ni una.

Observa también que las fechas las introducimos entre comillas, al igual que hacemos con los datos de tipo VARCHAR (no importa si son simples o dobles). Si se te olvidan, obtendrás un 0000-00-00; advertido quedas.

MySQL te permite cierta flexibilidad a la hora de introducir fechas:

'2012-4-5' es equivalente a '2012-04-05'.

Y para el año podemos registrar tan sólo las dos últimas cifras:

'12-10-5' es equivalente a '2012-10-5'.

Pero claro, esto presenta un problema: ¿cómo sé, en el ejemplo anterior, si me estoy refiendo a 2012 en vez de 1912?

MySQL sigue el siguiente protocolo a la hora de traducir fechas con sólo dos cifras en el año:

Si el año está comprendido entre 00 y 69, ambos incluidos, lo precede con un 20.

En caso contrario, es decir, entre 70 y 99, es precedido por 19.

MySQL también permite flexibilidad a la hora de elegir el separador entre los campos. Puedes usar otros signos de puntuación, aparte del guión. Por ejemplo, las siguientes fechas son equivalentes:

'2012-7-31' '2012/7/31' '2012#7#31' '2012:7:31'

Para finalizar, vamos a practicar creando una tabla, denominada contactos, en la que registraremos el nombre, apellido y la fecha de nacimiento de nuestros conocidos:

mysql> CREATE TABLE contactos

    -> (

    -> nombre VARCHAR(20),

    -> apellidos VARCHAR(30),

    -> f\_nacimiento DATE

    -> );

Query OK, 0 rows affected (0.38 sec)

### ¿Qué utilidad tiene especificar el número de caracteres en SQL?

Ya explicamos que al crear una tabla debemos elegir la estructura adecuada, esto es, definir los campos y sus tipos más precisos, según el caso.

Para almacenar TEXTO usamos cadenas de caracteres.

Las cadenas se colocan entre comillas simples.

Podemos almacenar letras, símbolos y dígitos con los que no se realizan operaciones matemáticas, por ejemplo, códigos de identificación, números de documentos, números telefónicos.

Tenemos los siguientes tipos:

varchar(x): define una cadena de caracteres de longitud variable en la cual determinamos el máximo de caracteres con el argumento "x" que va entre paréntesis.

Si se omite el argumento coloca 1 por defecto. Su rango va de 1 a 8000 caracteres.

char(x): define una cadena de longitud fija determinada por el argumento "x". Si se omite el argumento coloca 1 por defecto. Su rango es de 1 a 8000 caracteres.

Si la longitud es invariable, es conveniente utilizar el tipo char; caso contrario, el tipo varchar.

Ocupa tantos bytes como se definen con el argumento "x".

"char" viene de character, que significa carácter en inglés.

text: guarda datos binarios de longitud variable, puede contener hasta 2000000000 caracteres. No admite argumento para especificar su longitud.

nvarchar(x): es similar a "varchar", excepto que permite almacenar caracteres Unicode, su rango va de 0 a 4000 caracteres porque se emplean 2 bytes por cada carácter.

nchar(x): es similar a "char" excpeto que acepta caracteres Unicode, su rango va de 0 a 4000 caracteres porque se emplean 2 bytes por cada caracter.

ntext: es similar a "text" excepto que permite almacenar caracteres Unicode, puede contener hasta 1000000000 caracteres. No admite argumento para especificar su longitud.

En general se usarán los 3 primeros.

Si intentamos almacenar en un campo una cadena de caracteres de mayor longitud que la definida, aparece un mensaje indicando tal situación y la sentencia no se ejecuta.

Por ejemplo, si definimos un campo de tipo varchar(10) y le asignamos la cadena 'Aprenda PHP' (11 caracteres), aparece un mensaje y la sentencia no se ejecuta.

Si ingresamos un valor numérico (omitiendo las comillas), lo convierte a cadena y lo ingresa como tal.

Por ejemplo, si en un campo definido como varchar(5) ingresamos el valor 12345, lo toma como si hubiésemos tipeado '12345', igualmente, si ingresamos el valor 23.56, lo convierte a '23.56'. Si el valor numérico, al ser convertido a cadena supera la longitud definida, aparece un mensaje de error y la sentencia no se ejecuta.

Es importante elegir el tipo de dato adecuado según el caso, el más preciso.

Para almacenar cadenas que varían en su longitud, es decir, no todos los registros tendrán la misma longitud en un campo determinado, se emplea "varchar" en lugar de "char".

Por ejemplo, en campos que guardamos nombres y apellidos, no todos los nombres y apellidos tienen la misma longitud.

Para almacenar cadenas que no varían en su longitud, es decir, todos los registros tendrán la misma longitud en un campo determinado, se emplea "char".

Por ejemplo, definimos un campo "código" que constará de 5 caracteres, todos los registros tendrán un código de 5 caracteres, ni más ni menos.

Para almacenar valores superiores a 8000 caracteres se debe emplear "text".

Tipo Bytes de almacenamiento

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

varchar(x) 0 a 8K

char(x) 0 a 8K

text 0 a 2GB

nvarchar(x) 0 a 8K

nchar(x) 0 a 8K

ntext 0 a 2GB

### Por último, definan el acrónimo CRUD.

¿Estás utilizando una API o creando una base de datos y aún no sabes qué es CRUD? Este acrónimo es uno de los procesos más importantes en el mundo de la programación.

Si te ha pasado que has realizado algún tipo de desarrollo y no funciona correctamente, a lo mejor el fallo se debe a que no estás teniendo en cuenta varios pasos en la comprobación.

Por esta razón, **vamos a contarte en detalle qué es CRUD**y por qué es necesario conocerlo para gestionar adecuadamente la información que ingresa y sale al ejecutar las líneas de código de tu desarrollo. ¡Sigue leyendo para saber más!

¿Qué significa CRUD en programación?

CRUD es una agrupación de cuatro acciones que sirven para gestionar la información que se almacena en diferentes tipos de desarrollo. Asimismo, se usa para comprobar que los desarrollos sean completamente funcionales.

Si te preguntas qué significa CRUD, cada una de las letras de esta sigla corresponden a una acción en particular: **Create (crear), Read (leer), Update (actualizar) y Delete (eliminar)**.

El objetivo de realizar cada una de estas acciones es la de almacenar, organizar y clasificar los datos. Pero también funcionan para corregir los errores en la manipulación de información.

## ¿Cuál es la importancia de CRUD?

Después de haber visto **qué es CRUD**, vamos a explicarte por qué su uso es tan importante en el desarrollo web. La capacidad de **crear, leer, actualizar y eliminar** elementos en una **aplicación web** es crucial para la mayoría de los proyectos full stack, es decir, toda la tecnología de un sitio web que interpreta los requisitos de los usuarios a la arquitectura del sitio.

Por ejemplo, si estamos **creando una tienda, una página de publicación en un blog, una lista de tareas o**[**clones de redes sociales**](https://www.redeszone.net/tutoriales/seguridad/clones-redes-sociales/) sin **operaciones CRUD** los desarrolladores quedarían atorados muy rápidamente.

El CRUD es demasiado importante para ser ignorado, por lo que sería ideal aprenderlo primero, pues puede realmente mejorar tu confianza a la hora de enfrentarte a **stacks de desarrollo** desconocidos.

Por eso, el objetivo personal de un desarrollador al aprender un nuevo lenguaje de consulta de base de datos o de programación, es cómo realizar operaciones CRUD. Pues el CRUD se utiliza constantemente para cualquier cosa relacionada con la base de datos y el diseño de la misma.

Por ejemplo, el **desarrollo de sitios** web utiliza **REST** (Representational State Transfer), que es un **superconjunto de CRUD** utilizado para los **recursos** **HTTP**.

Por otro lado, el CRUD es igual de importante para los usuarios finales. Sin él, cosas como el **registro en sitios web, la creación de blogs**o**los marcadores** serían imposibles de realizar.

La mayoría de las aplicaciones que utilizamos nos permiten **añadir o crear nuevas entradas, buscar las existentes, realizar cambios**en ellas o**eliminarlas**. Para ello, el CRUD ofrece muchas ventajas, entre ellas:

**Facilita el control de la seguridad** para los distintos requisitos de acceso.

**Simplifica y facilita el diseño** de la aplicación haciéndola más escalable.

**Tiene un mejor rendimiento** en comparación con las **sentencias** **SQL ad-hoc**.

## Qué palabra clave se utiliza para:

## 1.Mostrar bases de datos en SQL

Lo primero que haremos será **realizar la consulta de todas las bases de datos que hay en nuestro servidor**; para ello, se utilizarán dos palabras clave del lenguaje SQL; la primera de ellas, conocida como **"SHOW"** (mostrar), con la cual indicaremos que queremos mostrar y/o listar ciertos elementos; seguidamente, la segunda palabra, **"DATABASES"** (Bases de datos), que lógicamente nos permite hacer referencia de manera general a las bases de datos de nuestro servidor:

SHOW DATABASES; **3.**Una vez tengas el listado de cada una de las bases de datos actuales, procederás a digitar las 2 sentencias SQL que siguen; las cuales, te ayudarán a **obtener las tablas de una base de datos específica**.  
  
 En la siguiente sentencia o línea de código, lo que se busca es seleccionar o indicarle al servidor, que trabajaremos con una base de datos determinada; es por ello, que emplearemos la expresión **"USE"** (usar); seguidamente, deberás colocar el nombre de la base de datos con la que quieres trabajar, en este caso se llama **"BD"**:

USE BD;

Luego de elegir la base de datos, ya se podrá llevar a cabo el proceso para poder consultar las tablas que hacen parte de la misma; en esta ocasión, al igual como se hizo en la primera sentencia (punto 2), se emplearán de forma seguida dos palabras reservadas del lenguaje SQL; la primera **"SHOW"** (mostrar), la cual ya conocemos; y la segunda de ellas, llamada **"TABLES"** (tablas), que nos ayudará a informarle al servidor que nos referimos a las tablas de una base de datos seleccionada anteriormente (punto 4):

## 2.Describir una base de datos en SQL.

## 3.Seleccionar una base de datos:

USE BD;

Luego de elegir la base de datos, ya se podrá llevar a cabo el proceso para poder consultar las tablas que hacen parte de la misma; en esta ocasión, al igual como se hizo en la primera sentencia (punto 2), se emplearán de forma seguida dos palabras reservadas del lenguaje SQL; la primera **"SHOW"** (mostrar), la cual ya conocemos; y la segunda de ellas, llamada **"TABLES"** (tablas), que nos ayudará a informarle al servidor que nos referimos a las tablas de una base

## 4. Crear una tabla.

Create Table “nombre de la tabla a crear “

declaración se utiliza para crear una nueva tabla en una base de datos.