Programación Web con PHP y MySQL

Unidad 2: Introducción a MYSQL

Contacto: consultas@elearning-total.com
Web: www.elearning-total.com

Indice

Unidad 2: Introducción a MySQL

Objetivos

Que el alumno logre:

- Aprender los conceptos de tabla, columna, fila o registro y valor de un dato dentro de la tabla.
- Aprender a definir una llave o clave primaria y cuál es el propósito de la misma.

MySQL

TIPOS DE CAMPOS EN MYSQL.

Retomemos y profundicemos un tema muy importante al momento de crear nuestras bases de datos, ni más ni menos que la selección de los tipos de campos que albergaran los datos en nuestras tablas, MySQL dijimos, soporta un número de tipos de campos (columnas) divididos en varias categorías. Una vez que hemos decidido qué información debemos almacenar, el siguiente paso consiste en elegir el tipo adecuado para cada atributo.

Podemos agrupar los tipos de datos en las siguientes categorías: de **caracteres**, **enteros**, de **coma flotante**, **tiempos**, **bloques**, **enumerados** y **conjuntos**.

Parámetros

Se define entonces brevemente a una base de datos como una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular.

M. Este parámetro se utiliza para indicar el número máximo de caracteres que pueden tener los valores que incluyamos en esta columna. M puede ser cualquier número entero entre 1 y 255. El almacenamiento de un número de caracteres mayor, indefectiblemente, nos acabará causando problemas, sobre todo cuando las relaciones entre las distintas bases de datos sean más complejas.

D. Este parámetro nos permite especificar cuántos números decimales pueden ser almacenados en valores de punto flotante. El máximo valor de este es 30, siempre y cuando el parámetro M nos lo permita.

Atributos

ZEROFILL. Esta opción consigue que la columna siempre ocupe su máxima longitud, dado que en el caso de que no le asignemos ningún valor, el sistema automáticamente lo completará con ceros. De esta manera nos aseguramos de que en una búsqueda siempre encontramos un valor, aunque sea cero. Al activar esta opción, automáticamente se activa la opción UNSIGNED.

UNSIGNED. Esta opción consigue que la columna sólo acepte valores positivos, o cero. Hayque tener precaución y, antes de activarla, tener completamente seguro que no queremos aceptar valores negativos.

BINARY. Por defecto, los caracteres de comparación en MySQL no distinguen entre mayúsculas y minúsculas. Sin embargo, los caracteres de comparación en columnas BINARY sí admiten mayúsculas y minúsculas.

Los corchetes indican información adicional. En terminología ODBC, a M se le denomina precisión y a D escala.

TIPOS DE DATOS DE CADENAS DE CARACTERES

CHAR

CHAR

Es un sinónimo de CHAR(1), y puede contener un único carácter.

CHAR()

CHAR(M) [BINARY | ASCII | UNICODE]

Contiene una cadena de longitud constante. Para mantener la longitud de la cadena, se rellena a la derecha con espacios. Estos espacios se eliminan al recuperar el valor. Los valores válidos para M son de 0 a 255, y de 1 a 255 para versiones de MySQL previas a 3.23. Si no se especifica la palabra clave BINARY estos valores se ordenan y comparan sin distinguir mayúsculas y minúsculas. CHAR es un alias para CHARACTER.

VARCHAR()

VARCHAR(M) [BINARY]

Contiene una cadena de longitud variable. Los valores válidos para M son de 0 a 255, y de 1 a 255 en versiones de MySQL anteriores a 4.0.2. Los espacios al final se eliminan.

Si no se especifica la palabra clave BINARY estos valores se ordenan y comparan sin distinguir mayúsculas y minúsculas. VARCHAR es un alias para CHARACTER

TIPOS DE DATOS ENTEROS

TINYINT

TINYINT[(M)] [UNSIGNED] [ZEROFILL]

Contiene un valor entero muy pequeño. El rango con signo es entre -128 y 127. El rango sin singo, de 0 a 255.

BIT, BOOL y BOOLEAN, todos son sinónimos de TINYINT(1).

SMALLINT

SMALLINT[(M)] [UNSIGNED] [ZEROFILL]

Contiene un entero corto. El rango con signo es de -32768 a 32767. El rango sin singo, de 0 a 65535.

MEDIUMINT

MEDIUMINT[(M)] [UNSIGNED] [ZEROFILL]

Contiene un entero de tamaño medio, el rango con signo está entre -8388608 y 8388607. El rango sin signo, entre 0 y 16777215.

INT

INT[(M)] [UNSIGNED] [ZEROFILL]

Contiene un entero de tamaño normal. El rango con signo está entre -2147483648 y 2147483647. El rango sin singo, entre 0 y 4294967295.

INTEGER

INTEGER[(M)] [UNSIGNED] [ZEROFILL]

Es sinónimo de INT.

BIGINT

BIGINT[(M)] [UNSIGNED] [ZEROFILL]

Contiene un entero grande. El rango con signo es de -9223372036854775808 a 9223372036854775807. El rango sin signo, de 0 a 18446744073709551615.

TIPOS DE DATOS EN COMA FLOTANTE

FLOAT

FLOAT(precision) [UNSIGNED] [ZEROFILL]

Contiene un número en coma flotante, precisión puede ser menor o igual que 24 para números de precisión sencilla y entre 25 y 53 para números en coma flotante de doble precisión. Estos tipos son idénticos que los tipos FLOAT y DOUBLE descritos a continuación. FLOAT(X) tiene el mismo rango que los tipos FLOAT y DOUBLE correspondientes, pero el tamaño mostrado y el número de decimales quedan indefinidos.

FLOAT()

FLOAT[(M,D)] [UNSIGNED] [ZEROFILL]

Contiene un número en coma flotante pequeño (de precisión sencilla). Los valores permitidos son entre -3.402823466E+38 y -1.175494351E-38, 0, y entre 1.175494351E- 38 y 3.402823466E+38. Si se especifica el modificador UNSIGNED, los valores negativos no se permiten. El valor M es la anchura a mostrar y D es el número de decimales. Si se usa sin argumentos o si se usa FLOAT(X), donde X sea menor o igual que 24, se sigue definiendo un valor en coma flotante de precisión sencilla.

DOUBLE

DOUBLE[(M,D)] [UNSIGNED] [ZEROFILL]

Contiene un número en coma flotante de tamaño normal (precisión doble). Los valores permitidos están entre -1.7976931348623157E+308 y -2.2250738585072014E-308, 0, y entre 2.2250738585072014E-308 y 1.7976931348623157E+308. Si se especifica el modificador UNSIGNED, no se permiten los valores negativos. El valor M es la anchura a mostrar y D es el número de decimales. Si se usa sin argumentos o si se usa FLOAT(X), donde X esté entre 25 y 53, se sigue definiendo un valor en coma flotante de doble precisión.

DOUBLE PRECISION

REAL

DOUBLE PRECISION[(M,D)] [UNSIGNED] [ZEROFILL]

REAL[(M,D)] [UNSIGNED] [ZEROFILL]

Ambos son sinónimos de DOUBLE.

DECIMAL

DECIMAL[(M[,D])] [UNSIGNED] [ZEROFILL]

Contiene un número en coma flotante sin empaquetar. Se comporta igual que una columna CHAR: "sin empaquetar" significa que se almacena como una cadena, usando un carácter para cada dígito del valor. El punto decimal y el signo '-' para valores negativos, no se cuentan en M (pero el espacio para estos se reserva). Si D es 0, los valores no tendrán punto decimal ni decimales. El rango de los valores DECIMAL es el mismo que para DOUBLE, pero el rango actual para una columna DECIMAL dada está restringido por la elección de los valores M y D. Si se especifica el modificador UNSIGNED, los valores negativos no están permitidos. Si se omite D, el valor por defecto es 0. Si se omite M, el valor por defecto es 10.

DEC

NUMERIC

FIXED

DEC[(M[,D])] [UNSIGNED] [ZEROFILL]

NUMERIC[(M[,D])] [UNSIGNED] [ZEROFILL]

FIXED[(M[,D])] [UNSIGNED] [ZEROFILL]

Todos ellos son sinónimos de DECIMAL.

TIPOS DE DATOS PARA TIEMPOS

DATE

DATE

Contiene una fecha. El rango soportado está entre '1000-01-01' y '9999-12-31'. MySQL muestra los valores DATE con el formato 'AAAA-MM-DD', pero es posible asignar valores a columnas de este tipo usando tanto números como cadenas.

DATETIME

DATETIME

Contiene una combinación de fecha y hora. El rango soportado está entre '1000-01-01 00:00:00' y '9999-12-31 23:59:59'. MySQL muestra los valores DATETIME con el formato 'AAAA-MM-DD HH:MM:SS', pero es posible asignar valores a columnas de este tipo usando tanto cadenas como números.

TIMESTAMP

TIMESTAMP[(M)]

Contiene un valor del tipo timestamp. El rango está entre '1970-01-01 00:00:00' y algún momento del año 2037. Hasta MySQL 4.0 los valores TIMESTAMP se mostraban como AAAAMMDDHHMMSS, AAMMDDHHMMSS, AAAAMMDD o AAMMDD, dependiendo del si el valor de M es 14 (o se omite), 12, 8 o 6, pero está permitido asignar valores a columnas TIMESTAMP usando tanto cadenas como números.

Desde MySQL 4.1, TIMESTAMP se devuelve como una cadena con el formato 'AAAA-MM-DD HH:MM:SS'. Para convertir este valor a un número, bastará con sumar el valor 0. Ya no se soportan distintas longitudes para estas columnas.

Se puede asignar fácilmente la fecha y hora actual a uno de estas columnas asignando el valor NULL. El argumento M afecta sólo al modo en que se visualiza la columna TIMESTAMP. Los valores siempre se almacenan usando cuatro bytes. Además, los valores de columnas TIMESTAMP(M), cuando M es 8 ó 14 se devuelven como números, mientras que para el resto de valores se devuelven como cadenas.

TIME

TIME

Una hora. El rango está entre '-838:59:59' y '838:59:59'. MySQL muestra los valores TIME en el formato 'HH:MM:SS', pero permite asignar valores a columnas TIME usando tanto cadenas como números.

YEAR

YEAR[(2|4)]

Contiene un año en formato de 2 ó 4 dígitos (por defecto es 4). Los valores válidos son entre 1901 y 2155, y 0000 en el formato de 4 dígitos. Y entre 1970-2069 si se usa el formato de 3 dígitos (70-69). MySQL muestra los valores YEAR usando el formato AAAA, pero permite asignar valores a una columna YEAR usando tanto cadenas como números.

TIPOS DE DATOS PARA DATOS SIN TIPO O GRANDES BLOQUES DE DATOS

TINYBLOB

TINYTEXT

TINYBLOB

TINYTEXT

Contiene una columna BLOB o TEXT con una longitud máxima de 255 caracteres.

BLOB

TEXT

BLOB

TEXT

Contiene una columna BLOB o TEXT con una longitud máxima de 65535 caracteres.

MEDIUMBLOB

MEDIUMTEXT

MEDIUMBLOB

MEDIUMTEXT

Contiene una columna BLOB o TEXT con una longitud máxima de 16777215 caracteres.

LONGBLOB

LONGTEXT

LONGBLOB

LONGTEXT

Contiene una columna BLOB o TEXT con una longitud máxima de 4294967298 caracteres.

TIPOS ENUMERADOS Y CONJUNTOS

ENUM

ENUM('valor1','valor2',...)

Contiene un enumerado. Un objeto de tipo cadena que puede tener un único valor, entre una lista de valores 'valor1', 'valor2', ..., NULL o el valor especial de error "". Un ENUM puede tener un máximo de 65535 valores diferentes.

SET

SET('valor1','valor2',...)

Contiene un conjunto. Un objeto de tipo cadena que puede tener cero o más valores, cada uno de los cuales debe estar entre una lista de valores 'valor1', 'valor2', ... Un conjunto puede tener un máximo de 64 miembros.

TAMAÑOS DE ALMACENAMIENTO

CAMPOS NUMÉRICOS:

Tipo de Campo	Tamaño de Almacenamiento
TINYINT	1 byte
SMALLINT	2 bytes
MEDIUMINT	3 bytes
INT	4 bytes
INTEGER	4 bytes
BIGINT	8 bytes
FLOAT(X)	4 ú 8 bytes
FLOAT	4 bytes
DOUBLE	8 bytes
DOUBLE PRECISION	8 bytes
REAL	8 bytes
DECIMAL(M,D	M+2 bytes sí D > 0, M+1 bytes sí D = 0
NUMERIC(M,D)	M+2 bytes if D > 0, M+1 bytes if D = 0

CAMPOS DE FECHA:

Tipo de Campo	Tamaño de Almacenamiento
DATE	3 bytes
DATETIME	8 bytes
TIMESTAMP	4 bytes
TIME	3 bytes
YEAR	1 byte

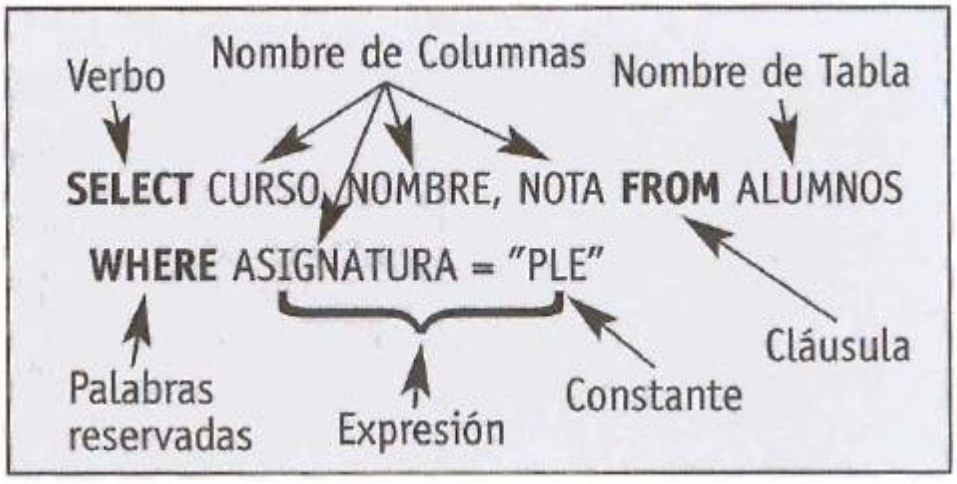
CAMPOS DE TEXTO:

Valor	CHAR(4)	Almacenamiento	VARCHAR(4)	Almacenamiento
''	''	4 bytes	''	1 byte
'ab'	'ab '	4 bytes	'ab'	3 bytes
'abcd'	'abcd'	4 bytes	'abcd'	
'abcdefgh'	'abcd'	4 bytes	'abcd'	5 bytes

Componentes sintácticos

La mayoría de sentencias SQL tienen la misma estructura.

Todas comienzan por un verbo (select, insert, update, create), a continuación le sigue una o más clausulas que nos dicen los datos con los que vamos a operar (from, where), algunas de estas son opcionales y otras obligatorias como es el caso del from.



Los mandatos de SQL se dividen en tres grandes grupos diferenciados:

- **DDL** (Data Definition Language)
- **DML** (Data Manipulation Language)
- **DCL** (Data Control Language)

Resumen

En esta Unidad…

En la presente unidad trabajamos con los parámetros de SQL necesarios para comenzar a interactuar con nuestras bases de datos, para prepararnos para incorporar contenido dinámico a nuestros sitios.

En la próxima Unidad…

En la próxima unidad seguiremos trabajando con MySQL.