

Reglas de normalización

La primera regla de normalización se expresa generalmente en forma de dos indicaciones separadas.

1. Todos los atributos, valores almacenados en las columnas, deben ser indivisibles.
2. No deben existir grupos de valores repetidos.

El valor de una columna debe ser una **entidad atómica**, indivisible, excluyendo así las dificultades que podría conllevar el tratamiento de un dato formado de varias partes.

Supongamos que tienes en una tabla una columna Dirección para almacenar la dirección completa, dato que se compondría del nombre de la calle, el número exterior, el número interior (puerta), el código postal, el estado y la capital.

id	Nombre	Dirección	Teléfono	URL
1	Anaya	J:l: Luca	92199932	Anaya.com
2	Pericles	C/Luna # 20-28018 Tlaxcala	99299492	Pericles.com




Calle	Número	Puerta	CP	Población	Provincia
Luna	20		28018	Tlaxcala	Tlaxcala

Una tabla con esta estructura plantea problemas a la hora de recuperar información. Imagina que necesitas conocer todas las entradas correspondientes a una determinada población, o que quieres buscar a partir del código postal. Al ser la dirección completa una secuencia de caracteres de estructura libre no resultaría nada fácil.

Existirán más columnas, pero cada una de ellas contendrá un valor simple e indivisible que facilitará la realización de las operaciones antes mencionadas.

En cuanto a la segunda indicación, se debe evitar la repetición de los datos de la población y provincia en cada una de las filas. Siempre que al muestrear la información de una tabla aparezcan datos repetidos, existe la posibilidad de crear una tabla independiente con ellos.

id	Nombre	calle	Número	Puerta	CP	Estado	Capital	Teléfono	URL
1	Anaya	Jl: Luca	15	2	28917	Tepic	Nayarit	93488345	Anaya.com
2	Pericles	Luna	20		28120	San Blas	Nayarit	88238188	Pericles.com
3	Mieres	Tajin	12	1	28120	San Blas	Nayarit	94989982	Mieres.es



Estado	CP	PK
CP	Estado	Capital
28917	Tepic	Nayarit
28120	San Blas	Nayarit
23009	Jean	Jaen

Listado y descripción de los distintos tipos de datos de MySQL.

Después de la fase de diseño de una base de datos, y una vez se ha realizado el paso a tablas del mismo, es necesario crear las tablas correspondientes dentro de la base de datos. Para cada campo de cada una de las tablas, es necesario determinar el tipo de datos que contiene, para de esa forma ajustar el diseño de la base de datos, y conseguir un almacenamiento óptimo con la menor utilización de espacio. El presente artículo describe cada uno de los tipos de datos que puede tener un campo en Mysql, para la versión 4.xx.xx.

Los tipos de datos que puede haber en un campo, se pueden agrupar en tres grandes grupos:

1. [Tipos numéricos](#)
2. [Tipos de Fecha](#)
3. [Tipos de Cadena](#)

1 Tipos numéricos:

Existen tipos de datos numéricos, que se pueden dividir en dos grandes grupos, los que están en coma flotante (con decimales) y los que no.

TinyInt:

Es un número entero con o sin signo. Con signo el rango de valores válidos va desde -128 a 127. Sin signo, el rango de valores es de 0 a 255

Bit ó Bool:

Un número entero que puede ser 0 ó 1

SmallInt:

Número entero con o sin signo. Con signo el rango de valores va desde -32768 a 32767. Sin signo, el rango de valores es de 0 a 65535.

MediumInt:

Número entero con o sin signo. Con signo el rango de valores va desde -8.388.608 a 8.388.607. Sin signo el rango va desde 0 a 16777215.

Integer, Int:

Número entero con o sin signo. Con signo el rango de valores va desde -2147483648 a 2147483647. Sin signo el rango va desde 0 a 429.4967.295

BigInt:

Número entero con o sin signo. Con signo el rango de valores va desde -9.223.372.036.854.775.808 a 9.223.372.036.854.775.807. Sin signo el rango va desde 0 a 18.446.744.073.709.551.615.

Float:

Número pequeño en coma flotante de precisión simple. Los valores válidos van desde -3.402823466E+38 a -1.175494351E-38, 0 y desde 1.175494351E-38 a 3.402823466E+38.

xReal, Double:

Número en coma flotante de precisión doble. Los valores permitidos van desde -1.7976931348623157E+308 a -2.2250738585072014E-308, 0 y desde 2.2250738585072014E-308 a 1.7976931348623157E+308

Decimal, Dec, Numeric:

Número en coma flotante desempaquetado. El número se almacena como una cadena

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

2 Tipos fecha:

A la hora de almacenar fechas, hay que tener en cuenta que Mysql no comprueba de una manera estricta si una fecha es válida o no. Simplemente comprueba que el mes esta comprendido entre 0 y 12 y que el día esta comprendido entre 0 y 31.

Date:

Tipo fecha, almacena una fecha. El rango de valores va desde el 1 de enero del 1001 al 31 de diciembre de 9999. El formato de almacenamiento es de año-mes-día

DateTime:

Combinación de fecha y hora. El rango de valores va desde el 1 de enero del 1001 a las 0 horas, 0 minutos y 0 segundos al 31 de diciembre del 9999 a las 23 horas, 59 minutos y 59 segundos. El formato de almacenamiento es de año-mes-día horas:minutos:segundos

TimeStamp:

Combinación de fecha y hora. El rango va desde el 1 de enero de 1970 al año 2037. El formato de almacenamiento depende del tamaño del campo:

Tamaño	Formato
14	AñoMesDiaHoraMinutoSegundo aaaammdhmmss
12	AñoMesDiaHoraMinutoSegundo aammdhmmss
8	ñoMesDia aaaamdd
6	AñoMesDia aamdd
4	AñoMes aamm
2	Año aa

Time:

Almacena una hora. El rango de horas va desde -838 horas, 59 minutos y 59 segundos a 838, 59 minutos y 59 segundos. El formato de almacenamiento es de 'HH:MM:SS'

Year:

Almacena un año. El rango de valores permitidos va desde el año 1901 al año 2155. El campo puede tener tamaño dos o tamaño 4 dependiendo de si queremos almacenar el año con dos o cuatro dígitos.

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

3 Tipos de cadena:

Char(n):

Almacena una cadena de longitud fija. La cadena podrá contener desde 0 a 255 caracteres.

VarChar(n):

Almacena una cadena de longitud variable. La cadena podrá contener desde 0 a 255 caracteres.

Dentro de los tipos de cadena se pueden distinguir otros dos subtipos, los tipo Text y los tipo BLOB (Binary large Object)

La diferencia entre un tipo y otro es el tratamiento que reciben a la hora de realizar ordenamientos y comparaciones. Mientras que el tipo test se ordena sin tener en cuenta las Mayúsculas y las minúsculas, el tipo BLOB se ordena teniéndolas en cuenta. Los tipos BLOB se utilizan para almacenar datos binarios como pueden ser ficheros.

TinyText y TinyBlob:

Columna con una longitud máxima de 255 caracteres.

Blob y Text:

Un texto con un máximo de 65535 caracteres.

MediumBlob y MediumText:

Un texto con un máximo de 16.777.215 caracteres.

LongBlob y LongText:

Un texto con un máximo de caracteres 4.294.967.295. Hay que tener en cuenta que debido a los protocolos de comunicación los paquetes pueden tener un máximo de 16 Mb.

Enum:

Campo que puede tener un único valor de una lista que se especifica. El tipo Enum acepta hasta 65535 valores distintos

Set:

Un campo que puede contener ninguno, uno ó varios valores de una lista. La lista puede tener un máximo de 64 valores.

Tipo de campo	Tamaño de Almacenamiento
CHAR(n)	n bytes
VARCHAR(n)	n +1 bytes
TINYBLOB, TINYTEXT	Longitud+1 bytes
BLOB, TEXT	Longitud +2 bytes
MEDIUMBLOB, MEDIUMTEXT	Longitud +3 bytes
LOB, LONGTEXT	Longitud +4 bytes
ENUM('value1','value2',...)	1 ó dos bytes dependiendo del número de valores
SET('value1','value2',...)	1, 2, 3, 4 ó 8 bytes, dependiendo del número de valores

Diferencia de almacenamiento entre los tipos Char y VarChar

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
------------	--------	---------	--------	---------

Referencias:

https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/872/mod_resource/content/1/contenido/index.html

<https://desarrolloweb.com/articulos/1054.php>