Tipos de dato numéricos

Listado de cada uno de los **tipos de dato numéricos en MySQL**, su ocupación en disco y valores.

- **INT (INTEGER)**: Ocupación de 4 bytes con valores entre -2147483648 y 2147483647 o entre 0 y 4294967295.
- **SMALLINT**: Ocupación de 2 bytes con valores entre -32768 y 32767 o entre 0 y 65535.
- **TINYINT**: Ocupación de 1 bytes con valores entre -128 y 127 o entre 0 y 255.
- **MEDIUMINT**: Ocupación de 3 bytes con valores entre -8388608 y 8388607 o entre 0 y 16777215.
- **BIGINT**: Ocupación de 8 bytes con valores entre -8388608 y 8388607 o entre 0 y 16777215.
- **DECIMAL (NUMERIC)**: Almacena los números de coma flotante como cadenas o string.
- **FLOAT (m,d)**: Almacena números de **coma flotante**, donde 'm' es el número de dígitos de la parte entera y 'd' el número de decimales.
- **DOUBLE (REAL)**: Almacena número de coma flotante con precisión doble. Igual que FLOAT, la diferencia es el rango de valores posibles.
- **BIT (BOOL, BOOLEAN)**: Número entero con valor 0 o 1.

Tipos de datos de MySQL Server

A continuación os mostramos los tipos de datos (data types) que se pueden definir para el motor de base de datos MySQL:

Grupo	Tipo de dato	Intervalo	Almacenamiento
Numéricos	TINYINT	De -128 a 127 (signed) De 0 a 255 (unsigned)	1 byte
	SMALLINT	De -32768 a 32767 (signed) De 0 a 65535 (unsigned)	2 bytes
	MEDIUMINT	De -8388608 a 8388607 (signed)	3 bytes

		De 0 a 16777215 (unsigned)	
	INT INTEGER	De -2147483648 a 2147483647 (signed) De 0 a 4294967295 (unsigned)	4 bytes
	BIGINT	De -9223372036854775808 a 9223372036854775807 (signed) De 0 a 18446744073709551615 (unsigned)	8 bytes
	BIT	Equivalente a TINYINT(1)	1 byte
	BOOL BOOLEAN	Equivalente a TINYINT(1) Valor 0 = False Valor 1 = True	1 byte
	FLOAT [(*M*,*D*)]	De -3.402823466E+38 a -1.175494351E-38, 0, y desde 1.175494351E-38 a 3.402823466E+38 *M* es el número total de dígitos y *D* es el número de dígitos después del punto decimal. Si se omite *M* y *D*, los valores se almacenan en los límites permitidos por el hardware (unas 7 posiciones decimales)	4 bytes
	FLOAT (*p*)	*p* representa la precisión en bits, MySQL usa este valor sólo para determinar si se debe usar FLOAT o DOUBLE para el tipo de datos resultante. Si *p* está entre 0 a 24, el tipo de datos se convierte en FLOAT (sin M ó D). Si *p* está entre 25 a 53, el tipo de datos se convierte a DOUBLE (sin M ó D).	4 bytes si 0 <= *p* <= 24, 8 bytes si 25 <= *p* <= 53

		En realidad este tipo de datos es proporcionado por MySQL por compatibilidad con ODBC	
	DOUBLE [(*M*,*D*)]	De -1.7976931348623157E+308 a -2.2250738585072014E-308, 0, y desde 2.2250738585072014E-308 a 1.7976931348623157E+308 *M* es el número total de dígitos y *D* es el número de dígitos después del punto decimal. Si se omite *M* y *D*, los valores se almacenan en los límites permitidos por el hardware (unas 15 posiciones decimales)	8 bytes
	REAL[(*M*,*D*)] DOUBLE PRECISION	Equivalente a DOUBLE, con la excepción de que si está activado el modo REAL_AS_FLOAT, REAL es un sinónimo de FLOAT en lugar de DOUBLE	4 Bytes ó 8 bytes
	DECIMAL [(*M*[,*D*])] DEC [(*M*[,*D*])] NUMERIC [(*M*[,*D*])] FIXED[(M[,D])]	Número en coma flotante sin empaquetar. Se comporta como una columna CHAR. El número se almacena como una cadena, usando un carácter para cada dígito del valor. El rango máximo es el mismo que para el tipo DOUBLE	*M*+2 bytes sí *D* > 0 *M*+1 bytes sí *D* = 0 *D*+2, si *M* < *D*
Fecha y hora	DATE	Fecha, con rango desde '1000-01-01' a '9999-12-31' con formato 'YYYY-MM-DD'	3 bytes

	DATETIME	Fecha y hora, con rango desde '1000-01-01 00:00:00' a '9999-12-31 23:59:59' con formato 'YYYY-MM-DD HH:MM:SS'	8 bytes
	TIMESTAMP[(*M*)]	Fecha y hora, el rango va desde '1970-01-01 00:00:01' UTC a '2038-01-19 03:14:07' UTC. El formato de almacenamiento depende del tamaño del campo	4 bytes
	TIME	Hora, con rango desde '-838:59:59' a '838:59:59', con el formato 'HH:MM:SS'	3 bytes
	YEAR[(2|4)]	Año en dos o cuatro dígitos, para cuatro dígitos, el rango es de 1901 a 2155, para dos dígitos es de 70 a 69 (representando desde 1070 a 2069)	1 byte
Cadenas de caracteres	CHAR (*M*)	Almacena una cadena de longitud fija. La cadena podrá contener desde 0 a 255 caracteres	*M* bytes (tanto si se ocupan como si no)
	VARCHAR (*M*)	Almacena una cadena de longitud variable. La cadena podrá contener desde 0 a 255 caracteres	Tamaño contenido del campo más 1 byte
	BINARY (*M*)	Similar a CHAR, excepto que contiene cadenas de caracteres binarias en lugar de cadenas no binarias. Es decir, que contienen cadenas de bytes en lugar de cadenas de caracteres. Esto significa que no tienen conjunto de caracteres, y la comparación y ordenación se basa en los valores numéricos de los bytes en los valores	*M* bytes, 0 <= *M* <= 255

	VARBINARY (*M*)	Similar a VARCHAR, excepto que contiene cadenas de caracteres binarias en lugar de cadenas no binarias. Es decir, que contienen cadenas de bytes en lugar de cadenas de caracteres. Esto significa que no tienen conjunto de caracteres, y la comparación y ordenación se basa en los valores numéricos de los bytes en los valores	Tamaño contenido del campo más 1 byte
	TEXT	Tipo de datos no binario que puede contener una cantidad variable de datos. Sirve para almacenar texto (gran cantidad). Hasta 65535 caracteres	*Longitud* + 2 bytes, mientras $L < 2^{16}$
	TINYTEX	Tipo de datos no binario que puede contener una cantidad variable de datos. Sirve para almacenar texto (gran cantidad). Hasta 255 caracteres	*Longitud* + 1 bytes, mientras $L < 2^{8}$
	MEDIUMTEXT	Tipo de datos no binario que puede contener una cantidad variable de datos. Sirve para almacenar texto (gran cantidad). Hasta 16.777.215 caracteres	*Longitud* + 3 bytes, mientras $L < 2^{24}$
	LONGTEXT	Tipo de datos no binario que puede contener una cantidad variable de datos. Sirve para almacenar texto (gran cantidad). Hasta 4.294.967.295 caracteres	*Longitud* + 4 bytes, mientras $L < 2^{32}$
	BLOB	Tipo de datos binario que puede contener una cantidad variable de datos. Permite almacenar ficheros (de cualquier tipo). Hasta 65535 bytes	*Longitud* + 2 bytes, mientras $L < 2^{16}$

	TINYBLOB	Tipo de datos binario que puede contener una cantidad variable de datos. Permite almacenar ficheros (de cualquier tipo). Hasta 255 bytes	*Longitud* + 1 bytes, mientras $L < 2^8$
	MEDIUMBLOB	Tipo de datos binario que puede contener una cantidad variable de datos. Permite almacenar ficheros (de cualquier tipo). Hasta 16.777.215 bytes	*Longitud* + 3 bytes, mientras $L < 2^{24}$
	LONGBLOB	Tipo de datos binario que puede contener una cantidad variable de datos. Permite almacenar ficheros (de cualquier tipo). Hasta 4.294.967.295 bytes	*Longitud* + 4 bytes, mientras $L < 2^{32}$
	ENUM (valor1, valor2, ...)	Es un tipo de datos de cadena con un valor elegido de una lista de valores permitidos que se enumeran explícitamente en la especificación de la columna al crear la tabla. Acepta hasta 65535 valores distintos	1 ó 2 bytes, dependiendo del número de valores de ENUM
	SET (valor1, valor2, ...)	Es un tipo de datos de cadena que puede contener ninguno, uno ó varios valores de una lista previamente establecida (al crear la tabla). La lista puede tener un máximo de 64 valores	1, 2, 3, 4, ó 8 bytes, dependiendo del número de miembros del conjunto

BLOB	Tipo de datos binario que puede contener una cantidad variable de datos. Permite almacenar ficheros (de cualquier tipo). Hasta 65535 bytes	*Longitud* + 2 bytes, mientras $L < 2^{16}$	