Tipos de datos en MySQL y sus diferencias:

Tipos de enteros (valor exacto): INTEGER, INT, SMALLINT, TINYINT, MEDIUMINT, BIGINT

Enteros:

- **TINYINT:** Almacena valores enteros pequeños (-128 a 127) con 1 byte.
- **SMALLINT:** Almacena valores enteros (-32768 a 32767) con 2 bytes.
- **MEDIUMINT:** Almacena valores enteros (-8388608 a 8388607) con 3 bytes.
- **INTEGER:** Almacena valores enteros (-2147483648 a 2147483647) con 4 bytes.
- **BIGINT:** Almacena valores enteros grandes (-9223372036854775808 a 9223372036854775807) con 8 bytes.

Table 13.1 Required Storage and Range for Integer Types Supported by MySQL

Type	Storage (Bytes)	Minimum Value Signed	Minimum Value Unsigned	Maximum Value Signed	Maximum Value Unsigned
TINYINT	1	-128	0	127	255
SMALLINT	2	-32768	0	32767	65535
MEDIUMINT	3	-8388608	0	8388607	16777215
INT	4	-2147483648	0	2147483647	4294967295
BIGINT	8	-2^{63}	0	$2^{63}-1$	$2^{64}-1$

Tipos de punto fijo (valor exacto): DECIMAL, NUMERIC

Estos tipos se utilizan cuando es importante preservar la precisión exacta, por ejemplo, con datos monetarios. En MySQL, NUMERIC se implementa como DECIMAL, por lo que las siguientes observaciones DECIMAL se aplican igualmente a NUMERIC.

En una DECIMAL declaración de columna, la precisión y la escala se pueden especificar (y normalmente se especifican). Por ejemplo:

```
salary DECIMAL(5,2)
```

En este ejemplo, 5 es la precisión y 2 es la escala. La precisión representa la cantidad de dígitos significativos que se almacenan para los valores y la escala representa la cantidad de dígitos que se pueden almacenar después del punto decimal.

Por lo tanto, los valores que se pueden almacenar en la salary columna van desde -999.99 hasta 999.99.

El valor predeterminado de es 10. DECIMAL(***M***) DECIMAL(***M***,0) DECIMAL DECIMAL(***M***,0) ***M*** DECIMAL ***M.*** Si la escala es 0, DECIMAL los valores no contienen punto decimal ni parte fraccionaria.

El número máximo de dígitos DECIMAL es 65

Decimales:

- **DECIMAL:** Almacena valores decimales precisos (hasta 65,535 dígitos) con longitud y escala especificadas.
- **NUMERIC:** Similar a DECIMAL, pero con mayor precisión (hasta 127 dígitos) y control de redondeo.

Tipos de coma flotante (valor aproximado): FLOAT, DOUBLE

MySQL permite una sintaxis no estándar: FLOAT(***M***,***D***) o REAL(***M***,***D***) o DOUBLE PRECISION(***M***,***D***).

FLOAT(***M***,***D***) y DOUBLE(***M***,***D)*** son extensiones de MySQL no estándar; y están en desuso. Debería esperar que se elimine la compatibilidad con estas variantes en una versión futura de MySQL.

Debido a que los valores de punto flotante son aproximados y no se almacenan como valores exactos, los intentos de tratarlos como exactos en las comparaciones pueden generar problemas

Flotantes:

- **FLOAT:** Almacena valores de punto flotante aproximados (hasta 4 dígitos decimales) con 4 bytes.
- **DOUBLE:** Almacena valores de punto flotante aproximados (hasta 17 dígitos decimales) con 8 bytes.

Diferencias:

- **Tamaño de almacenamiento:** Los tipos enteros varían en tamaño (1 a 8 bytes) según el rango de valores. Los decimales y flotantes tienen un tamaño fijo.
- **Precisión:** Los decimales permiten mayor precisión que los flotantes.
- **Escala:** Los decimales permiten controlar la escala de los dígitos decimales.
- **Usos:** Los enteros se usan para identificadores, contadores, etc. Los decimales para valores monetarios, porcentajes, etc. Los flotantes para valores aproximados, como medidas físicas.

Tipos de datos de cadena

- **Cadenas:** CHAR, VARCHAR, TEXT, BLOB para almacenar texto de diferentes longitudes y formatos.

CHAR

Una cadena de longitud fija que siempre se rellena a la derecha con espacios hasta la longitud especificada cuando se almacena. ***M*** representa la longitud de la columna en caracteres. El rango de ***M*** es de 0 a 255. Si ***M*** se omite, la longitud es 1.

VARCHAR

Una cadena de longitud variable. ***M*** representa la longitud máxima de la columna en caracteres. El rango de ***M*** es de 0 a 65.535.

Los tipos CHAR y VARCHAR son similares, pero difieren en la forma en que se almacenan y recuperan. También difieren en la longitud máxima y en si se conservan los espacios finales.

Los tipos CHAR y VARCHAR se declaran con una longitud que indica la cantidad máxima de caracteres que desea almacenar. Por ejemplo, CHAR(30)puede contener hasta 30 caracteres.

A diferencia de CHAR, VARCHAR los valores se almacenan como un prefijo de longitud de 1 o 2 bytes más datos. El prefijo de longitud indica el número de bytes del valor. Una columna utiliza un byte de longitud si los valores no requieren más de 255 bytes, dos bytes de longitud si los valores pueden requerir más de 255 bytes.

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

Los valores que se muestran almacenados en la última fila de la tabla se aplican *sólo cuando no se utiliza el modo SQL estricto*; Si el modo estricto está habilitado, los valores que exceden la longitud de la columna *no se almacenan* y se produce un error.

Tipos de datos de fecha y hora

- **Fecha y hora:** DATE, TIME, DATETIME, TIMESTAMP para almacenar fechas y horas con diferentes precisiones.

Tipo de datos	Valor " cero "
DATE	'0000-00-00'
TIME	'00:00:00'
DATETIME	'0000-00-00 00:00:00'
TIMESTAMP	'0000-00-00 00:00:00'
YEAR	0000

DATE

El rango admitido es '1000-01-01'de '9999-12-31'. El formato es '***YYYY-MM-DD***'

DATETIME

Una combinación de fecha y hora. El rango admitido es '1000-01-01 00:00:00.000000'de '9999-12-31 23:59:59.499999'. MySQL muestra DATETIME valores en formato '***YYYY-MM-DD hh:mm:ss***

TIMESTAMP

Una marca de tiempo. El rango es '1970-01-01 00:00:01.000000'de UTC a '2038-01-19 03:14:07.499999'UTC. Los valores se almacenan como la cantidad de segundos desde la época ('1970-01-01 00:00:00' UTC).

TIME

El rango es '-838:59:59.000000' de '838:59:59.000000'. MySQL muestra valores en formato, '***hh:mm:ss***[.***fraction***]'

YEAR

Un año en formato de 4 dígitos. MySQL muestra valores en ***YYYY*** formato. Los valores se muestran como 1901 hasta 2155, o 0000.

Booleanos

- en MySQL es un alias de TINYINT(1).
- Los valores lógicos TRUE y FALSE se almacenan como 1 y 0, respectivamente.
- Esto se debe a razones de compatibilidad, eficiencia y simplicidad.
- Se debe considerar este comportamiento al trabajar con valores booleanos en MySQL para almacenar valores true o false.