Material descargable

egg

Funciones de fecha y hora

Las funciones de fecha y hora en MySQL son herramientas poderosas que permiten a los desarrolladores y administradores de bases de datos trabajar con fechas y horas de manera eficiente y precisa. Estas funciones son esenciales para diversas tareas, como la gestión de registros de eventos temporales, el cálculo de duraciones, el formateo de fechas, agregar o restar intervalos de tiempo a fechas, extraer partes específicas de fechas, convertir entre zonas horarias, calcular diferencias entre fechas y mucho más. Estas funciones son esenciales para garantizar que las aplicaciones basadas en bases de datos puedan gestionar adecuadamente seguimiento de datos históricos y otras operaciones relacionadas con el tiempo.

Aquí tienes una tabla con todas las funciones y una breve descripción:

Nombre	**Descripción**
ADDDATE()	Agregar valores de tiempo (intervalos) a un valor de fecha
ADDTIME()	Agregar tiempo
CONVERT_TZ()	Convertir de una zona horaria a otra
CURDATE()	Devolver la fecha actual
CURRENT_DATE()	Sinónimo de CURDATE()
CURRENT_TIME()	Sinónimo de CURTIME()
CURRENT_TIMESTAMP()	Sinónimo de NOW()
CURTIME()	Devolver la hora actual

DATE()	Extraer la parte de fecha de una expresión de fecha o datetime
DATE_ADD()	Agregar valores de tiempo (intervalos) a un valor de fecha
DATE_FORMAT()	Formatear una fecha según lo especificado
DATE_SUB()	Restar un valor de tiempo (intervalo) de una fecha
DATEDIFF()	Restar dos fechas
DAY()	Sinónimo de DAYOFMONTH()
DAYNAME()	Devolver el nombre del día de la semana
DAYOFMONTH()	Devolver el día del mes (0-31)
DAYOFWEEK()	Devolver el índice del día de la semana
DAYOFYEAR()	Devolver el día del año (1-366)
EXTRACT()	Extraer parte de una fecha
FROM_DAYS()	Convertir un número de día en una fecha
FROM_UNIXTIME()	Formatear una marca de tiempo Unix como fecha
GET_FORMAT()	Devolver una cadena de formato de fecha
HOUR()	Extraer la hora
LAST_DAY()	Devolver el último día del mes para el argumento
LOCALTIME()	Sinónimo de NOW()
LOCALTIMESTAMP()	Sinónimo de NOW()
MAKEDATE()	Crear una fecha a partir del año y el día del año

MAKETIME()	Crear una hora a partir de hora, minuto, segundo
MICROSECOND()	Devolver los microsegundos del argumento
MINUTE()	Devolver el minuto del argumento
MONTH()	Devuelve el mes de la fecha pasada
MONTHNAME()	Devuelve el nombre del mes
NOW()	Devuelve la fecha y hora actual
PERIOD_ADD()	Agrega un período a un año-mes
PERIOD_DIFF()	Devuelve el número de meses entre períodos
QUARTER()	Devuelve el trimestre de un argumento de fecha
SEC_TO_TIME()	Convierte segundos a formato 'hh:mm:ss'
SECOND()	Devuelve los segundos (0-59)
STR_TO_DATE()	Convierte una cadena en una fecha
SUBDATE()	Sinónimo de DATE_SUB() cuando se invoca con tres argumentos
SUBTIME()	Resta horas
SYSDATE()	Devuelve la hora a la que se ejecuta la función
TIME()	Extrae la porción de tiempo de la expresión pasada
TIME_FORMAT()	Formato como hora
TIME_TO_SEC()	Devuelve el argumento convertido a segundos
TIMEDIFF()	Resta tiempo

TIMESTAMP()	Con un solo argumento, esta función devuelve la expresión de fecha o fecha y hora; con dos argumentos, la suma de los argumentos
TIMESTAMPADD()	Agrega un intervalo a una expresión de fecha y hora
TIMESTAMPDIFF()	Devuelve la diferencia entre dos expresiones de fecha y hora, utilizando las unidades especificadas
TO_DAYS()	Devuelve el argumento de fecha convertido en días
TO_SECONDS()	Devuelve el argumento de fecha o fecha y hora convertido en segundos desde el Año 0
UNIX_TIMESTAMP()	Devuelve una marca de tiempo Unix
UTC_DATE()	Devuelve la fecha UTC actual
UTC_TIME()	Devuelve la hora UTC actual
UTC_TIMESTAMP()	Devuelve la fecha y hora UTC actual
WEEK()	Devuelve el número de semana
WEEKDAY()	Devuelve el índice del día de la semana
WEEKOFYEAR()	Devuelve la semana del calendario de la fecha (1-53)
YEAR()	Devuelve el año
YEARWEEK()	Devuelve el año y la semana

📍 *Todas las funciones retornan NULL en caso de error.*

ADDDATE(date, INTERVAL expr unit), ADDDATE(date, days)

Cuando se llama con la forma INTERVAL del segundo argumento, ADDDATE() es un sinónimo de DATE_ADD(). La función relacionada SUBDATE() es un sinónimo de DATE_SUB. Para obtener información sobre el argumento unit del intervalo, consulta Intervalos Temporales.

```
SELECT DATE_ADD('2008-01-02', INTERVAL 31 DAY);
2008-02-02
SELECT ADDDATE('2008-01-02', INTERVAL 31 DAY);
2008-02-02'
```

Cuando se invoca con pasando solo un número en el segundo argumento, MySQL lo trata como un número entero de días a añadir a expr.

```
SELECT ADDDATE('2008-01-02', 31);
2008-02-02
```

Esta función devuelve NULL si date o days es NULL.

ADDTIME(expr1, expr2)

ADDTIME() suma expr2 a expr1 y devuelve el resultado. expr1 es una expresión de tiempo o fecha y expr2 es una expresión de tiempo. Devuelve NULL si expr1 o expr2 es NULL.

El tipo devuelto de la función se deriva del tipo del primer argumento.

```
SELECT ADDTIME('2007-12-31 23:59:59.999999', '1 1:1:1.000002');
2008-01-02 01:01:01.000001
SELECT ADDTIME('01:00:00.999999', '02:00:00.999998');
03:00:01.999997
```

CONVERT_TZ(dt, from_tz, to_tz)

CONVERT_TZ() convierte un valor de fecha y hora “dt” de la zona horaria from_tz a la zona horaria to_tz y devuelve el valor resultante. Esta función devuelve NULL si alguno de los argumentos es inválido o si alguno de ellos es NULL.

Independientemente de la plataforma o la versión de MySQL, si el valor cae fuera del rango admitido al convertirse de from_tz a UTC, no se realiza la conversión.

```
SELECT CONVERT_TZ('2004-01-01 12:00:00', 'GMT', 'MET');
2004-01-01 13:00:00
SELECT CONVERT_TZ('2004-01-01 12:00:00', '+00:00', '+10:00');
2004-01-01 22:00:00
```

💡 zonas horarias

CURDATE()

Devuelve la fecha actual como un valor en formato 'AAAA-MM-DD' o YYYYMMDD, dependiendo de si la función se utiliza en un contexto de cadena o numérico.

```
SELECT CURDATE();
2008-06-13
SELECT CURDATE() + 0;
20080613
```

CURRENT_DATE, CURRENT_DATE()

CURRENT_DATE y CURRENT_DATE() son sinónimos de CURDATE.

CURRENT_TIME, CURRENT_TIME([fsp])

CURRENT_TIME y CURRENT_TIME() son sinónimos de CURTIME.

CURRENT_TIMESTAMP, CURRENT_TIMESTAMP([fsp])

CURRENT_TIMESTAMP y CURRENT_TIMESTAMP() son sinónimos de NOW().

CURTIME([fsp])

Devuelve la hora actual como un valor en formato 'hh:mm:ss' o hhmmss, dependiendo de si la función se utiliza en un contexto de cadena o numérico. El valor se expresa en la zona horaria de la sesión.

Si se proporciona el argumento fsp para especificar una precisión de segundos fraccionales de 0 a 6, el valor devuelto incluye una parte de segundos fraccionales con esa cantidad de dígitos.

```
SELECT CURTIME();
19:25:37
SELECT CURTIME(3);
08:53:39.913
```

DATE(expr)

Extrae la parte de fecha de la expresión de fecha o fecha y hora expr. Devuelve NULL si expr es NULL.

```
SELECT DATE('2003-12-31 01:02:03');
2003-12-31
```

DATEDIFF(expr1, expr2)

DATEDIFF() devuelve expr1 - expr2 expresado como un valor en días desde una fecha a la otra. expr1 y expr2 son expresiones de fecha o fecha y hora. Solo se utilizan las partes de fecha de los valores en el cálculo. Esta función devuelve NULL si expr1 o expr2 es NULL.

```
SELECT DATEDIFF('2007-12-31 23:59:59', '2007-12-30');
1
SELECT DATEDIFF('2010-11-30 23:59:59', '2010-12-31');
-31
```

DATE_ADD(date, INTERVAL expr unit), DATE_SUB(date, INTERVAL expr unit)

Estas funciones realizan cálculos de fecha. El argumento de fecha especifica la fecha de inicio o el valor de fecha y hora. expr es una expresión que especifica el valor del intervalo que se debe agregar o restar a la fecha de inicio. expr se evalúa como una cadena; puede comenzar con un "-" para intervalos negativos. unit es una palabra clave que indica las unidades en las que se debe interpretar la expresión.

El valor de retorno depende de los argumentos:

- Si date es NULL, la función devuelve NULL.
- DATE si el argumento date es un valor DATE y tus cálculos involucran solo las partes de AÑO, MES y DÍA (es decir, ninguna parte de hora).
- TIME si el argumento date es un valor TIME y los cálculos involucran solo las partes de HORAS, MINUTOS y SEGUNDOS (es decir, ninguna parte de fecha).
- DATETIME si el primer argumento es un valor DATETIME (o TIMESTAMP), o si el primer argumento es un DATE y el valor unit utiliza HORAS, MINUTOS o SEGUNDOS, o si el primer argumento es de tipo TIME y el valor unit utiliza AÑOS, MESES o DÍAS.

Lista de unidades: YEAR (Años), QUARTER (Trimestres), MONTH (Meses), WEEK (Semanas), DAY (Días), HOUR (Horas), MINUTE (Minutos), SECOND (Segundos), MICROSECOND (Microsegundos).

```
SELECT DATE_ADD('2018-05-01', INTERVAL 1 DAY);
2018-05-02
SELECT DATE_SUB('2018-05-01', INTERVAL 1 YEAR);
2017-05-01
SELECT DATE_ADD('2020-12-31 23:59:59', INTERVAL 1 SECOND);
2021-01-01 00:00:00
```

DATE_FORMAT(date, format)

Formatea el valor de fecha de acuerdo con la cadena de formato. Si cualquiera de los argumentos es NULL, la función devuelve NULL.

Los especificadores que se muestran en la siguiente tabla se pueden usar en la cadena de formato. El carácter % es requerido antes de los caracteres del especificador de formato. Los especificadores se aplican a otras funciones también: STR_TO_DATE(), TIME_FORMAT(), UNIX_TIMESTAMP().

Especificador	Descripción
%a	Nombre abreviado del día de la semana (Dom..Sáb)
%b	Nombre abreviado del mes (Ene..Dic)
%c	Mes, numérico (0..12)
%D	Día del mes con sufijo en inglés (0th, 1st, 2nd, 3rd, …)
%d	Día del mes, numérico (00..31)
%e	Día del mes, numérico (0..31)
%f	Microsegundos (000000..999999)
%H	Hora (00..23)
%h	Hora (01..12)

%I	Hora (01..12)
%i	Minutos, numéricos (00..59)
%j	Día del año (001..366)
%k	Hora (0..23)
%l	Hora (1..12)
%M	Nombre del mes (Enero..Diciembre)
%m	Mes, numérico (00..12)
%p	AM o PM
%r	Hora, formato de 12 horas (hh:mm:ss seguido de AM o PM)
%S	Segundos (00..59)
%s	Segundos (00..59)
%T	Hora, formato de 24 horas (hh:mm:ss)
%U	Semana (00..53), donde el domingo es el primer día de la semana; modo WEEK() 0
%u	Semana (00..53), donde el lunes es el primer día de la semana; modo WEEK() 1
%V	Semana (01..53), donde el domingo es el primer día de la semana; modo WEEK() 2; se usa con %X
%v	Semana (01..53), donde el lunes es el primer día de la semana; modo WEEK() 3; se usa con %x
%W	Nombre del día de la semana (Domingo..Sábado)
%w	Día de la semana (0=Domingo..6=Sábado)

%X	Año para la semana donde el domingo es el primer día de la semana, numérico, cuatro dígitos; se usa con %V
%x	Año para la semana donde el lunes es el primer día de la semana, numérico, cuatro dígitos; se usa con %v
%Y	Año, numérico, cuatro dígitos
%y	Año, numérico (dos dígitos)
%%	Carácter % literal
%x	x, para cualquier "x" no listado anteriormente

Los rangos para los especificadores de mes y día comienzan con cero debido a que MySQL permite el almacenamiento de fechas incompletas como '2014-00-00'.

El idioma utilizado para los nombres y abreviaturas de los días y meses se controla mediante el valor de la variable de sistema lc_time_names (Sección 10.16, "Soporte de la Localización del Servidor MySQL").

```
SELECT DATE_FORMAT('2009-10-04 22:23:00', '%W %M %Y');
Sunday October 2009
SELECT DATE_FORMAT('2007-10-04 22:23:00', '%H:%i:%s');
22:23:00
SELECT DATE_FORMAT('1900-10-04 22:23:00','%D %y %a %d %m %b %j');
4th 00 Thu 04 10 Oct 277
SELECT DATE_FORMAT('1997-10-04 22:23:00','%H %k %I %r %T %S %w');
22 22 10 10:23:00 PM 22:23:00 00 6
SELECT DATE_FORMAT('1999-01-01', '%X %V');
1998 52
SELECT DATE_FORMAT('2006-06-01', '%d');
01
```

DATE_SUB(date, INTERVAL expr unit)

Consulta la descripción de DATE_ADD().

DAY(date)

DAY() es un sinónimo de DAYOFMONTH().

DAYNAME(date)

Devuelve el nombre del día de la semana para la fecha. El idioma utilizado para el nombre está controlado por el valor de la variable del sistema lc_time_names. Devuelve NULL si la fecha es NULL.

```
SELECT DAYNAME('2007-02-03');
Sábado
```

DAYOFMONTH(date)

Devuelve el día del mes para la fecha, en el rango de 1 a 31, o 0 para fechas como '0000-00-00' o '2008-00-00' que tienen una parte de día igual a cero. Devuelve NULL si la fecha es NULL.

```
SELECT DAYOFMONTH('2007-02-03');
3
```

DAYOFWEEK(date)

Devuelve el índice del día de la semana para la fecha (1 = domingo, 2 = lunes, ..., 7 = sábado). Estos valores de índice corresponden al estándar ODBC. Devuelve NULL si la fecha es NULL.

```
SELECT DAYOFWEEK('2007-02-03');
7
```

DAYOFYEAR(date)

Devuelve el día del año para la fecha, en el rango de 1 a 366. Devuelve NULL si la fecha es NULL.

```
SELECT DAYOFYEAR('2007-02-03');
34
```

EXTRACT(unit FROM date)

La función EXTRACT() utiliza los mismos tipos de especificadores de unidad que DATE_ADD() o DATE_SUB(), pero extrae partes de la fecha en lugar de realizar cálculos de fecha. Para obtener información sobre el argumento de unidad, consulta Intervalos temporales. Devuelve NULL si la fecha es NULL.

```
SELECT EXTRACT(YEAR FROM '2019-07-02');
2019
SELECT EXTRACT(YEAR_MONTH FROM '2019-07-02 01:02:03');
201907
SELECT EXTRACT(DAY_MINUTE FROM '2019-07-02 01:02:03');
20102
SELECT EXTRACT(MICROSECOND FROM '2003-01-02 10:30:00.000123');
123
```

FROM_DAYS(N)

Dado un número de día N, devuelve un valor de tipo DATE. Devuelve NULL si N es NULL.

```
SELECT FROM_DAYS(730669);
2000-07-03
```

Utiliza FROM_DAYS() con precaución en fechas antiguas. No está destinado a su uso con valores que preceden a la introducción del calendario gregoriano (1582).

FROM_UNIXTIME(unix_timestamp[,format])

Devuelve una representación de unix_timestamp como un valor de tipo datetime o cadena de caracteres. El valor devuelto se expresa utilizando la zona horaria de la sesión, unix_timestamp es un valor de marca de tiempo interno que representa segundos desde '1970-01-01 00:00:00' UTC, como el que produce la función UNIX_TIMESTAMP().

Si se omite el formato, esta función devuelve un valor de tipo DATETIME.

Si unix_timestamp o el formato son NULL, esta función devuelve NULL.

Cuando unix_timestamp es un número entero, la precisión de los segundos fraccionales del DATETIME es cero. Cuando unix_timestamp es un valor decimal, la precisión de los segundos fraccionales del DATETIME es la misma que la

precisión del valor decimal, con un máximo de 6. Cuando unix_timestamp es un número de punto flotante, la precisión de los segundos fraccionales del datetime es de 6.

En computadoras de 32 bits, el valor máximo útil para unix_timestamp es 2147483647.999999, lo que devuelve '2038-01-19 03:14:07.999999' UTC. En plataformas de 64 bits que ejecutan MySQL 8.0.28 o posterior, el valor máximo efectivo es 32536771199.999999, que devuelve '3001-01-18 23:59:59.999999' UTC. Independientemente de la plataforma o versión, un valor mayor que el máximo efectivo para unix_timestamp devuelve 0.

El formato se utiliza para dar formato al resultado de la misma manera que la cadena de formato utilizada para la función DATE_FORMAT(). Si se proporciona un formato, el valor devuelto es una cadena de caracteres (VARCHAR).

```
SELECT FROM_UNIXTIME(1447430881);
2015-11-13 10:08:01
SELECT FROM_UNIXTIME(1447430881) + 0;
20151113100801
SELECT FROM_UNIXTIME(1447430881, '%Y %D %M %h:%i:%s %x');
'2015 13th November 10:08:01 2015'
```

GET_FORMAT({DATE|TIME|DATETIME}, {'EUR'|'USA'|'JIS'|'ISO'|'INTERNAL'})

Devuelve una cadena de formato. Esta función es útil en combinación con las funciones DATE_FORMAT() y STR_TO_DATE(). Si el formato es NULL, esta función devuelve NULL.

Los posibles valores para el primer y segundo argumento resultan en varias cadenas de formato posibles. El formato ISO se refiere a ISO 9075, no a ISO 8601.

Llamada de función	Resultado
GET_FORMAT(DATE,'USA')	'%m.%d.%Y'
GET_FORMAT(DATE,'JIS')	'%Y-%m-%d'
GET_FORMAT(DATE,'ISO')	'%Y-%m-%d'

GET_FORMAT(DATE,'EUR')	'%d.%m.%Y'
GET_FORMAT(DATE,'INTERNAL')	'%Y%m%d'
GET_FORMAT(DATETIME,'USA')	'%Y-%m-%d %H.%i.%s'
GET_FORMAT(DATETIME,'JIS')	'%Y-%m-%d %H:%i:%s'
GET_FORMAT(DATETIME,'ISO')	'%Y-%m-%d %H:%i:%s'
GET_FORMAT(DATETIME,'EUR')	'%Y-%m-%d %H.%i.%s'
GET_FORMAT(DATETIME,'INTERNAL')	'%Y%m%d%H%i%s'
GET_FORMAT(TIME,'USA')	'%h:%i:%s %p'
GET_FORMAT(TIME,'JIS')	'%H:%i:%s'
GET_FORMAT(TIME,'ISO')	'%H:%i:%s'
GET_FORMAT(TIME,'EUR')	'%H.%i.%s'
GET_FORMAT(TIME,'INTERNAL')	'%H%i%s'

TIMESTAMP también se puede utilizar como primer argumento en GET_FORMAT(), en cuyo caso la función devuelve los mismos valores que para DATETIME.

```
SELECT DATE_FORMAT('2003-10-03',GET_FORMAT(DATE,'EUR'));
03.10.2003
SELECT STR_TO_DATE('10.31.2003',GET_FORMAT(DATE,'USA'));
2003-10-31
```

HOUR(time)

Devuelve la hora de un valor de tiempo (time). El rango del valor devuelto es de 0 a 23 para valores de hora del día. Sin embargo, el rango de valores de TIME en realidad es mucho más grande, por lo que HOUR puede devolver valores mayores que 23. Devuelve NULL si time es NULL.

```
SELECT HOUR('10:05:03');
10
SELECT HOUR('272:59:59');
272
```

LAST_DAY(date)

Toma un valor de fecha o datetime y devuelve el valor correspondiente para el último día del mes. Devuelve NULL si el argumento es inválido o NULL.

```
SELECT LAST_DAY('2003-02-05');
2003-02-28
SELECT LAST_DAY('2004-02-05');
2004-02-29
SELECT LAST_DAY('2004-01-01 01:01:01');
2004-01-31
SELECT LAST_DAY('2003-03-32');
NULL
```

LOCALTIME, LOCALTIME([fsp])

LOCALTIME y LOCALTIME() son sinónimos de NOW().

LOCALTIMESTAMP, LOCALTIMESTAMP([fsp])

LOCALTIMESTAMP y LOCALTIMESTAMP() son sinónimos de NOW().

MAKEDATE(year, dayofyear)

Devuelve una fecha dados el año y el día del año. dayofyear debe ser mayor que 0 o el resultado será NULL. El resultado también es NULL si alguno de los argumentos es NULL.

```
SELECT MAKEDATE(2011, 31), MAKEDATE(2011, 32);
2011-01-31, 2011-02-01
SELECT MAKEDATE(2011, 365), MAKEDATE(2014, 365);
2011-12-31, 2014-12-31
SELECT MAKEDATE(2011, 0);
NULL
```

MAKETIME(hour, minute, second)

Devuelve un valor de tiempo calculado a partir de los argumentos de hora, minutos y segundos. Devuelve NULL si alguno de sus argumentos es NULL.

```
SELECT MAKETIME(12, 15, 30);
12:15:30
```

MICROSECOND(expr)

Devuelve los microsegundos de la expresión de tiempo o fecha y hora expr como un número en el rango de 0 a 999999. Devuelve NULL si expr es NULL.

```
SELECT MICROSECOND('12:00:00.123456');
123456
SELECT MICROSECOND('2019-12-31 23:59:59.000010');
10
```

MINUTE(time)

Devuelve los minutos para el tiempo, en el rango de 0 a 59, o NULL si el tiempo es NULL.

```
SELECT MINUTE('2008-02-03 10:05:03');
5
```

MONTH(date)

Devuelve el mes para la fecha, en el rango de 1 a 12 para enero a diciembre, o 0 para fechas como '0000-00-00' o '2008-00-00' que tienen un mes igual a cero. Devuelve NULL si la fecha es NULL.

```
SELECT MONTH('2008-02-03');
2
```

MONTHNAME(date)

Devuelve el nombre completo del mes para la fecha. El idioma utilizado para el nombre está controlado por el valor de la variable del sistema lc_time_names. Devuelve NULL si la fecha es NULL.

```
SELECT MONTHNAME('2008-02-03');
Febrero
```

NOW([fsp])

Devuelve la fecha y hora actual como un valor en formato 'YYYY-MM-DD hh:mm:ss' o formato YYYYMMDDhhmmss, dependiendo de si la función se usa en contexto de cadena o numérico. El valor se expresa en la zona horaria de la sesión. Si se proporciona el argumento fsp para especificar una precisión de segundos fraccionarios de 0 a 6, el valor devuelto incluye una parte de segundos fraccionarios con esa cantidad de dígitos.

```
SELECT NOW();
2007-12-15 23:50:26
mysql> SELECT NOW() + 0;
20071215235026.000000
```

NOW() devuelve una hora constante que indica la hora a la que comenzó a ejecutarse la declaración. Esto difiere del comportamiento de SYSDATE(), que devuelve la hora exacta en la que se ejecuta.

PERIOD_ADD(P, N)

Añade N meses a un período P (en el formato YYMM o YYYYMM). Devuelve un valor en el formato YYYYMM. Esta función devuelve NULL si P o N son NULL.

📍 *El argumento de período P no es un valor de fecha.*

```
SELECT PERIOD_ADD(200801, 2);
200803
```

PERIOD_DIFF(P1, P2)

Devuelve el número de meses entre los períodos P1 y P2. P1 y P2 deben estar en el formato YYMM o YYYYMM. Ten en cuenta que los argumentos de período P1 y P2 no son valores de fecha. Esta función devuelve NULL si P1 o P2 son NULL.

```
SELECT PERIOD_DIFF(200802, 200703);
11
```

QUARTER(date)

Devuelve el trimestre del año para la fecha, en el rango de 1 a 4, o NULL si la fecha es NULL.

```
SELECT QUARTER('2008-04-01');
2
```

SECOND(time)

Devuelve los segundos para el tiempo, en el rango de 0 a 59, o NULL si el tiempo es NULL.

```
SELECT SECOND('10:05:03');
3
```

SEC_TO_TIME(seconds)

Devuelve el argumento de segundos convertido en horas, minutos y segundos, como un valor TIME. El rango del resultado está limitado al del tipo de dato TIME. Se genera una advertencia si el argumento corresponde a un valor fuera de ese rango. La función devuelve NULL si seconds es NULL.

```
SELECT SEC_TO_TIME(2378);
00:39:38
SELECT SEC_TO_TIME(2378) + 0;
3938
```

STR_TO_DATE(str, format)

Esto es el inverso de la función DATE_FORMAT(). Toma una cadena str y una cadena de formato format. STR_TO_DATE() devuelve un valor DATETIME si la cadena de formato contiene tanto partes de fecha como de hora, o un valor DATE o TIME si la cadena contiene solo partes de fecha o de hora. Si str o format es NULL, la función devuelve NULL. Si el valor de fecha, hora o fecha y hora extraído de str no se puede analizar de acuerdo con las reglas seguidas por el servidor, STR_TO_DATE() devuelve NULL y produce una advertencia.

El servidor examina str intentando que coincida con el formato. La cadena de formato puede contener caracteres literales y especificadores de formato que

comienzan con %. Los caracteres literales en el formato deben coincidir literalmente en la cadena str. Los especificadores de formato en el formato deben coincidir con una parte de fecha u hora en la cadena str. Para conocer los especificadores que se pueden usar en el formato, consulta la descripción de la función DATE_FORMAT.

```
SELECT STR_TO_DATE('01,5,2013', '%d,%m,%Y');
2013-05-01
SELECT STR_TO_DATE('May 1, 2013', '%M %d,%Y');
2013-05-01
SELECT STR_TO_DATE('04/31/2004', '%m/%d/%Y');
2004-04-31
```

La búsqueda comienza al principio de la cadena str y falla si no se encuentra una coincidencia con el formato. Los caracteres adicionales al final de str se ignoran.

```
SELECT STR_TO_DATE('a09:30:17', 'a%h:%i:%s');
09:30:17
SELECT STR_TO_DATE('a09:30:17', '%h:%i:%s');
NULL
SELECT STR_TO_DATE('09:30:17a', '%h:%i:%s');
09:30:17
```

Las partes de fecha u hora no especificadas tienen un valor de 0, por lo que los valores incompletamente especificados en str producen un resultado con algunas o todas las partes configuradas en 0:

```
SELECT STR_TO_DATE('abc', 'abc');
0000-00-00
SELECT STR_TO_DATE('9', '%m');
0000-09-00
SELECT STR_TO_DATE('9', '%s');
00:00:09
```

En MySQL 8.0.35 y versiones posteriores, STR_TO_DATE() realiza una comprobación de rango completa y genera un error si la fecha después de la conversión sería inválida.

SUBDATE(date, INTERVAL expr unit), SUBDATE(expr, days)

Cuando se invoca con la forma de INTERVAL del segundo argumento, SUBDATE() es un sinónimo de DATE_SUB().

```
SELECT DATE_SUB('2008-01-02', INTERVAL 31 DAY);
2007-12-02
SELECT SUBDATE('2008-01-02', INTERVAL 31 DAY);
2007-12-02
```

La segunda forma permite el uso de un valor entero para los días. En tales casos, se interpreta como el número de días que se restarán de la expresión de fecha o fecha y hora expr.

```
SELECT SUBDATE('2008-01-02 12:00:00', 31);
2007-12-02 12:00:00
```

Esta función devuelve NULL si alguno de sus argumentos es NULL.

SUBTIME(expr1, expr2)

La función SUBTIME() devuelve expr1 - expr2 expresado como un valor en el mismo formato que expr1. expr1 es una expresión de hora o fecha y hora, y expr2 es una expresión de tiempo.

La resolución del tipo de retorno de esta función se realiza de la misma manera que en la función ADDTIME. Esta función devuelve NULL si expr1 o expr2 es NULL.

```
SELECT SUBTIME('2007-12-31 23:59:59.999999', '1 1:1:1.000002');
2007-12-30 22:58:58.999997
SELECT SUBTIME('01:00:00.999999', '02:00:00.999998');
-00:59:59.999999
```

SYSDATE([fsp])

La función SYSDATE() devuelve la fecha y hora actual como un valor en formato 'YYYY-MM-DD hh:mm:ss' o formato YYYYMMDDhhmmss, dependiendo de si se utiliza la función en contexto de cadena o numérico.

Si se proporciona el argumento fsp para especificar una precisión de segundos fraccionarios de 0 a 6, el valor devuelto incluirá una parte de segundos fraccionarios con esa cantidad de dígitos.

SYSDATE() devuelve la hora en la que se ejecuta. Esto difiere del comportamiento de NOW(), que devuelve una hora constante que indica la hora a la que comenzó a ejecutarse la declaración.

📍*La función SYSDATE() puede devolver diferentes valores incluso dentro de la misma declaración.*

```
SELECT NOW(), SLEEP(2), NOW();
+---------------------+----------+---------------------+
| NOW()               | SLEEP(2) | NOW()               |
+---------------------+----------+---------------------+
| 2006-04-12 13:47:36 |        0 | 2006-04-12 13:47:36 |
+---------------------+----------+---------------------+

SELECT SYSDATE(), SLEEP(2), SYSDATE();
+---------------------+----------+---------------------+
| SYSDATE()           | SLEEP(2) | SYSDATE()           |
+---------------------+----------+---------------------+
| 2006-04-12 13:47:44 |        0 | 2006-04-12 13:47:46 |
+---------------------+----------+---------------------+
```

TIME(expr)

La función TIME(expr) extrae la parte de tiempo de la expresión de tiempo o fecha y hora expr y la devuelve como una cadena. Retorna NULL si expr es NULL.

```
SELECT TIME('2003-12-31 01:02:03');
'01:02:03'
SELECT TIME('2003-12-31 01:02:03.000123');
'01:02:03.000123'
```

TIME_FORMAT(time, format)

Esta función se utiliza de manera similar a la función DATE_FORMAT(), pero la cadena de formato puede contener especificadores de formato solo para horas, minutos, segundos y microsegundos. Otros especificadores producirán un valor NULL o 0. TIME_FORMAT() devuelve NULL si time o format es NULL.

Si el valor de tiempo contiene una parte de hora mayor que 23, los especificadores de formato de hora %H y %k producirán un valor mayor que el rango habitual de 0 a 23. Los demás especificadores de formato de hora producirán el valor de la hora módulo 12.

```
SELECT TIME_FORMAT('100:00:00', '%H %k %h %I %l');
100 100 04 04 4
```

TIME_TO_SEC(time)

Devuelve el argumento de tiempo convertido a segundos. Devuelve NULL si time es NULL.

```
SELECT TIME_TO_SEC('22:23:00');
80580
SELECT TIME_TO_SEC('00:39:38');
2378
```

TIMEDIFF(expr1, expr2)

La función TIMEDIFF() devuelve expr1 - expr2 expresado como un valor de tiempo. expr1 y expr2 son cadenas que se convierten en expresiones de TIEMPO o FECHA Y HORA; estas deben ser del mismo tipo después de la conversión. Retorna NULL si expr1 o expr2 es NULL.

El resultado devuelto por TIMEDIFF() está limitado al rango permitido para los valores de TIEMPO. Alternativamente, puedes usar las funciones TIMESTAMPDIFF() y UNIX_TIMESTAMP(), ambas de las cuales devuelven enteros.

```
SELECT TIMEDIFF('2000-01-01 00:00:00', '2000-01-01 00:00:00.000001');
-00:00:00.000001
SELECT      TIMEDIFF('2008-12-31      23:59:59.000001',      '2008-12-30
01:01:01.000002');
46:58:57.999999
```

TIMESTAMP(expr), TIMESTAMP(expr1, expr2)

Con un solo argumento, esta función devuelve la expresión de fecha o fecha y hora expr como un valor de fecha y hora. Con dos argumentos, agrega la expresión de tiempo expr2 a la expresión de fecha o fecha y hora expr1 y devuelve el resultado como un valor de fecha y hora. Retorna NULL si expr, expr1, o expr2 es NULL.

```
SELECT TIMESTAMP('2003-12-31');
2003-12-31 00:00:00
SELECT TIMESTAMP('2003-12-31 12:00:00', '12:00:00');
2004-01-01 00:00:00
```

TIMESTAMPADD(unit, interval, datetime_expr)

Añade el valor entero del intervalo expresado a la expresión de fecha o fecha y hora datetime_expr. La unidad del intervalo se especifica mediante el argumento de unidad, que debe ser uno de los siguientes valores: MICROSECOND (microsegundos), SECOND (segundos), MINUTE (minutos), HOUR (horas), DAY (días), WEEK (semanas), MONTH (meses), QUARTER (trimestres) o YEAR (años).

Esta función devuelve NULL si interval o datetime_expr es NULL.

```
SELECT TIMESTAMPADD(MINUTE, 1, '2003-01-02');
2003-01-02 00:01:00
SELECT TIMESTAMPADD(WEEK, 1, '2003-01-02');
2003-01-09
```

Cuando se agrega un intervalo de MONTH a un valor DATE o DATETIME, y la fecha resultante incluye un día que no existe en el mes dado, el día se ajusta al último día del mes, como se muestra aquí:

```
SELECT TIMESTAMPADD(MONTH, 1, DATE '2024-03-30'), TIMESTAMPADD(MONTH, 1,
DATE '2024-03-31');
 2024-04-30 | 2024-04-30
```

TIMESTAMPDIFF(unit, datetime_expr1, datetime_expr2)

Devuelve datetime_expr2 - datetime_expr1, donde datetime_expr1 y datetime_expr2 son expresiones de fecha o fecha y hora. Una expresión puede ser una fecha y la otra una fecha y hora; un valor de fecha se trata como una fecha y hora con la parte de hora '00:00:00' cuando es necesario. La unidad del resultado (un número entero) se especifica mediante el argumento de unidad. Los valores legales para la unidad son los mismos que se enumeran en la descripción de la función TIMESTAMPADD(). Esta función devuelve NULL si datetime_expr1 o datetime_expr2 son NULL.

```
SELECT TIMESTAMPDIFF(MONTH, '2003-02-01', '2003-05-01');
3
SELECT TIMESTAMPDIFF(YEAR, '2002-05-01', '2001-01-01');
-1
SELECT TIMESTAMPDIFF(MINUTE, '2003-02-01', '2003-05-01 12:05:55');
128885
```

TO_DAYS(date)

Dada una fecha date, devuelve un número de día (el número de días desde el año 0). Devuelve NULL si date es NULL.

```
SELECT TO_DAYS(950501);
728779
SELECT TO_DAYS('2007-10-07');
733321
```

📍*Como TO_SECONDS(), TO_DAYS() no está destinada a usarse con valores que preceden la adopción del calendario gregoriano (1582).*

TO_SECONDS(expr)

Dada una fecha o fecha y hora expr, esta función devuelve el número de segundos desde el año 0. Si expr no es un valor de fecha o fecha y hora válido (incluido NULL), devuelve NULL.

```
SELECT TO_SECONDS(950501);
        -> 62966505600
mysql> SELECT TO_SECONDS('2009-11-29');
        -> 63426672000
mysql> SELECT TO_SECONDS('2009-11-29 13:43:32');
        -> 63426721412
mysql> SELECT TO_SECONDS( NOW() );
        -> 63426721458
```

📍*Como TO_DAYS(), TO_SECONDS() no está destinada a usarse con valores que preceden la adopción del calendario gregoriano (1582).*

UNIX_TIMESTAMP([date])

Si se llama a UNIX_TIMESTAMP() sin argumento de fecha, devuelve una marca de tiempo de Unix que representa los segundos desde '1970-01-01 00:00:00' UTC.

Si se llama a UNIX_TIMESTAMP() con un argumento de fecha, devuelve el valor del argumento como segundos desde '1970-01-01 00:00:00' UTC. El servidor interpreta la fecha como un valor en la zona horaria de la sesión y la convierte en un valor interno de marca de tiempo de Unix en UTC.

El argumento de fecha puede ser una cadena de DATE, DATETIME o TIMESTAMP, o un número en formato YYMMDD, YYMMDDhhmmss, YYYYMMDD o

YYYYMMDDhhmmss. Si el argumento incluye una parte de tiempo, puede incluir opcionalmente una parte de segundos fraccionarios.

El valor devuelto es un número entero si no se proporciona ningún argumento o el argumento no incluye una parte de segundos fraccionarios, o DECIMAL si se proporciona un argumento que incluye una parte de segundos fraccionarios.

Para plataformas de 32 bits y antes de MySQL 8.0.28, el rango válido de valores de argumento es el mismo que para el tipo de datos TIMESTAMP: '1970-01-01 00:00:01.000000' UTC a '2038-01-19 03:14:07.999999' UTC. Para MySQL 8.0.28 y posteriores que se ejecutan en plataformas de 64 bits, el rango válido de valores de argumento para UNIX_TIMESTAMP() es '1970-01-01 00:00:01.000000' UTC a '3001-01-19 03:14:07.999999' UTC (correspondiente a 32536771199.999999 segundos).

Independientemente de la versión de MySQL o la arquitectura de la computadora, si se pasa una fecha fuera de rango a UNIX_TIMESTAMP(), devuelve 0. Si la fecha es NULL, devuelve NULL.

```
SELECT UNIX_TIMESTAMP();
1447431666
SELECT UNIX_TIMESTAMP('2015-11-13 10:20:19');
1447431619
SELECT UNIX_TIMESTAMP('2015-11-13 10:20:19.012');
1447431619.012
```

UTC_DATE, UTC_DATE()

Devuelve la fecha UTC actual como un valor en formato 'AAAA-MM-DD' o formato AAAAMMDD, dependiendo de si la función se utiliza en un contexto de cadena o numérico.

```
SELECT UTC_DATE(), UTC_DATE() + 0;
'2003-08-14', 20030814
```

UTC_TIME, UTC_TIME([fsp])

Devuelve la hora UTC actual como un valor en formato 'hh:mm:ss' o formato hhmmss, dependiendo de si la función se utiliza en un contexto de cadena o numérico.

Si se proporciona el argumento fsp para especificar una precisión de segundos fraccionarios de 0 a 6, el valor devuelto incluirá una parte de segundos fraccionarios con esa cantidad de dígitos.

```
SELECT UTC_TIME(), UTC_TIME() + 0;
'18:07:53', 180753.000000
```

UTC_TIMESTAMP, UTC_TIMESTAMP([fsp])

Devuelve la fecha y hora UTC actual como un valor en formato 'AAAA-MM-DD hh:mm:ss' o formato AAAAMMDDhhmmss, dependiendo de si la función se utiliza en un contexto de cadena o numérico.

Si se proporciona el argumento fsp para especificar una precisión de segundos fraccionarios de 0 a 6, el valor de retorno incluirá una parte de segundos fraccionarios con esa cantidad de dígitos.

```
SELECT UTC_TIMESTAMP(), UTC_TIMESTAMP() + 0;
'2003-08-14 18:08:04', 20030814180804.000000
```

WEEK(date[,mode])

Esta función devuelve el número de semana para la fecha. La forma de dos argumentos de WEEK() te permite especificar si la semana comienza el domingo o el lunes y si el valor de retorno debe estar en el rango de 0 a 53 o de 1 a 53. Si se omite el argumento de modo, se utiliza el valor de la variable de sistema default_week_format. Para un valor de fecha NULL, la función devuelve NULL.

La siguiente tabla describe cómo funciona el argumento de modo.

Modo	Primer día de la semana	Rango	Semana 1 es la primera semana...
0	Domingo	0-53	con un domingo en este año
1	Lunes	0-53	con 4 o más días en este año
2	Domingo	1-53	con un domingo en este año
3	Lunes	1-53	con 4 o más días en este año

4	Domingo	0-53	con 4 o más días en este año
5	Lunes	0-53	con un lunes en este año
6	Domingo	1-53	con 4 o más días en este año
7	Lunes	1-53	con un lunes en este año

Para valores de modo con un significado de "con 4 o más días este año", las semanas se numeran según ISO 8601:1988:

- Si la semana que contiene el 1 de enero tiene 4 o más días en el nuevo año, es la semana 1.
- De lo contrario, es la última semana del año anterior, y la próxima semana es la semana 1.

```
SELECT WEEK('2008-02-20');
7
SELECT WEEK('2008-02-20',0);
7
SELECT WEEK('2008-02-20',1);
8
SELECT WEEK('2008-12-31',1);
53
```

Si una fecha cae en la última semana del año anterior, MySQL devuelve 0 si no utilizas 2, 3, 6 o 7 como argumento de modo opcional:

```
SELECT YEAR('2000-01-01'), WEEK('2000-01-01',0);
2000, 0
```

Si prefieres un resultado evaluado con respecto al año que contiene el primer día de la semana para la fecha dada, utiliza 0, 2, 5 o 7 como argumento de modo opcional.

```
SELECT WEEK('2000-01-01',2);
52
```

Alternativamente, utiliza la función YEARWEEK:

```
SELECT YEARWEEK('2000-01-01');
199952
```

```
SELECT SUBSTRING(YEARWEEK('2000-01-01'),5,2);
52
```

WEEKDAY(date)

Devuelve el índice del día de la semana para una fecha (0 = lunes, 1 = martes, ... 6 = domingo). Devuelve NULL si la fecha es NULL.

```
SELECT WEEKDAY('2008-02-03 22:23:00');
6
SELECT WEEKDAY('2007-11-06');
1
```

WEEKOFYEAR(date)

Devuelve la semana del calendario de la fecha como un número en el rango de 1 a 53. Devuelve NULL si la fecha es NULL.

```
SELECT WEEKOFYEAR('2008-02-20');
8
```

YEAR(date)

Devuelve el año de la fecha, en el rango de 1000 a 9999, o 0 para la "fecha cero". Devuelve NULL si la fecha es NULL.

```
SELECT YEAR('1987-01-01');
1987
```

YEARWEEK(date), YEARWEEK(date,mode)

Devuelve el año y la semana de una fecha. El año en el resultado puede ser diferente del año en el argumento de fecha para la primera y la última semana del año. Devuelve NULL si la fecha es NULL.

```
SELECT YEARWEEK('1987-01-01');
198652
```

Zonas horarias

Aquí tienes una lista de ejemplos de valores de zona horaria válidos en MySQL:

- 'SYSTEM': Esto indica que se usara la zona horaria del servidor MySQL.
- Formato de desplazamiento desde UTC, como '+10:00', '-6:00' o '+05:30'. Puede incluir un signo más o menos y, opcionalmente, un cero inicial para las horas menores a 10. MySQL convierte '-00:00' o '-0:00' a '+00:00'. Antes de MySQL 8.0.19, el rango permitido era '-12:59' a '+13:00', pero a partir de MySQL 8.0.19, el rango se amplió a '-13:59' a '+14:00'.
- Nombres de zonas horarias, como 'Europe/Helsinki', 'US/Eastern', 'MET' o 'UTC'. Estos nombres de zona horaria son válidos siempre y cuando las tablas de información de zona horaria en la base de datos mysql estén creadas.

lc_time_names

Aqui te dejamos la forma en la que puedes cambiar el valor de esta variable.

```
SELECT @@lc_time_names;
en_US

SELECT DAYNAME('2020-01-01'), MONTHNAME('2020-01-01');
Wednesday | January

SELECT DATE_FORMAT('2020-01-01','%W %a %M %b');
Wednesday Wed January Jan

SET lc_time_names = 'es_MX';

SELECT @@lc_time_names;
es_MX

SELECT DAYNAME('2020-01-01'), MONTHNAME('2020-01-01');
miércoles | enero
```
