第五章 MySQL函数

一、数学函数

(1) 绝对值函数：ABS(x)

ABS(x) 用于返回 x 的绝对值

mysql> SELECT ABS(2), ABS(-2.3), ABS(-33);

+--------+-----------+----------+

| ABS(2) | ABS(-2.3) | ABS(-33) |

+--------+-----------+----------+

| 2 | 2.3 | 33 |

+--------+-----------+----------+

(2) 返回圆周率的函数：PI()

PI() 返回圆周率值

mysql> SELECT PI();

+----------+

| PI() |

+----------+

| 3.141593 |

+----------+

(3) 平方根函数：SQRT(x)

SQRT(x) 返回非负数 x 的二次方根

mysql> SELECT SQRT(9), SQRT(40), SQRT(-49);

+---------+-------------------+-----------+

| SQRT(9) | SQRT(40) | SQRT(-49) |

+---------+-------------------+-----------+

| 3 | 6.324555320336759 | NULL |

+---------+-------------------+-----------+

(4) 求余函数：MOD(x,y)

MOD(x,y) 返回 x 被 y 除后的余数，MOD(x,y) 对于带有小数部分的数值也起作用，它返回除法运算后的精确余数

mysql> SELECT MOD(31,8), MOD(234,10), MOD(45.5,6);

+-----------+-------------+-------------+

| MOD(31,8) | MOD(234,10) | MOD(45.5,6) |

+-----------+-------------+-------------+

| 7 | 4 | 3.5 |

+-----------+-------------+-------------+

(5) 获取整数的函数：CEIL(x) 、CEILING(x) 、FLOOR(x)

CEIL(x) 用于返回不小于 x 的最小整数值

mysql> SELECT CEIL(-3.35), CEIL(3.35);

+-------------+------------+

| CEIL(-3.35) | CEIL(3.35) |

+-------------+------------+

| -3 | 4 |

+-------------+------------+

CEILING(x) 用于返回不小于 x 的最小整数

mysql> SELECT CEILING(-3.35), CEILING(3.35);

+----------------+---------------+

| CEILING(-3.35) | CEILING(3.35) |

+----------------+---------------+

| -3

| 4 |

+----------------+---------------+

FLOOR(x) 返回不大于 x 的最大整数值

mysql> SELECT FLOOR(-3.35), FLOOR(3.35);

+--------------+-------------+

| FLOOR(-3.35) | FLOOR(3.35) |

+--------------+-------------+

| -4 | 3 |

+--------------+-------------+

(6) 获取随机数的函数：RAND() 、RAND(x)

RAND() 用于返回一个随机浮点值，范围在 0 ~ 1 之间

mysql> SELECT RAND(), RAND(), RAND();

+--------------------+--------------------+---------------------+

| RAND() | RAND() | RAND() |

+--------------------+--------------------+---------------------+

| 0.7393965169222994 | 0.5185907432051289 | 0.37476419599239175 |

+--------------------+--------------------+---------------------+

RAND(x) 用于返回一个随机浮点值，范围在 0~1 之间，x 被用作种子值，用来产生重复序列

mysql> SELECT RAND(5), RAND(5), RAND(8);

+---------------------+---------------------+---------------------+

| RAND(5) | RAND(5) | RAND(8) |

+---------------------+---------------------+---------------------+

| 0.40613597483014313 | 0.40613597483014313 | 0.15668530311126755 |

+---------------------+---------------------+---------------------+

(7) 四舍五入的函数：ROUND(x) 、ROUND(x,y)

ROUND(x) 用于对 x 进行四舍五入

mysql> SELECT ROUND(-1.14), ROUND(1.14);

+--------------+-------------+

| ROUND(-1.14) | ROUND(1.14) |

+--------------+-------------+

| -1 | 1 |

+--------------+-------------+

ROUND(x,y) 用于对 x 进行四舍五入，并保留小数点后 y 位

mysql> SELECT ROUND(1.38,1), ROUND(1.38,0), ROUND(232.38,-1), ROUND(232.38,-2);

+---------------+---------------+------------------+------------------+

| ROUND(1.38,1) | ROUND(1.38,0) | ROUND(232.38,-1) | ROUND(232.38,-2) |

+---------------+---------------+------------------+------------------+

| 1.4 | 1 | 230 | 200 |

+---------------+---------------+------------------+------------------+

(8) 截取数值的函数：TRUNCATE(x,y)

TRUNCATE(x,y) 用于对 x 进行截取，结果保留小数点后 y 位

mysql> SELECT TRUNCATE(1.31,1), TRUNCATE(1.99,1), TRUNCATE(1.99,0);

+------------------+------------------+------------------+

| TRUNCATE(1.31,1) | TRUNCATE(1.99,1) | TRUNCATE(1.99,0) |

+------------------+------------------+------------------+

| 1.3 | 1.9 | 1 |

+------------------+------------------+------------------+

(9) 符号函数：SIGN(x)

SIGN(x) 用于返回参数 x 的符号，当 x 为负数时返回 -1 ，当 x 为正数时返回 1 ， 当 x 为零时返回 0

mysql> SELECT SIGN(-21), SIGN(0), SIGN(21);

+-----------+---------+----------+

| SIGN(-21) | SIGN(0) | SIGN(21) |

+-----------+---------+----------+

| -1 | 0 | 1 |

+-----------+---------+----------+

(10) 幂运算函数：POW(x,y) 、POWER(x,y) 、EXP(x)

POW(x,y) 用于返回 x 的 y 次方的结果

mysql> SELECT POW(2,4), POW(2,-4);

+----------+-----------+

| POW(2,4) | POW(2,-4) |

+----------+-----------+

| 16 | 0.0625 |

+----------+-----------+

(11) 对数运算函数：LOG(x) 、LOG10(x)

LOG(x) 返回 x 的自然对数

mysql> SELECT LOG(3), LOG(-3);

+--------------------+---------+

| LOG(3) | LOG(-3) |

+--------------------+---------+

| 1.0986122886681098 | NULL |

+--------------------+---------+

使用 LOG10(x) 计算以 10 为基数的对数

mysql> SELECT LOG10(2), LOG10(100), LOG10(-100);

+--------------------+------------+-------------+

| LOG10(2) | LOG10(100) | LOG10(-100) |

+--------------------+------------+-------------+

| 0.3010299956639812 | 2 | NULL |

+--------------------+------------+-------------+

二、 字符串函数

(1) 计算字符串长度的函数：CHAR\_LENGTH(str) 、LENGTH(str)

CHAR\_LENGTH(str) 用于统计 str 的字符个数

mysql> SELECT CHAR\_LENGTH('date'), CHAR\_LENGTH('egg');

+---------------------+--------------------+

| CHAR\_LENGTH('date') | CHAR\_LENGTH('egg') |

+---------------------+--------------------+

| 4 | 3 |

+---------------------+--------------------+

LENGTH(str) 用于统计 str 的字节长度，使用 utf8 编码字符集时，一个汉字是3个字节，一个数字或字母是1个字节

mysql> SELECT LENGTH('date'), LENGTH('egg');

+----------------+---------------+

| LENGTH('date') | LENGTH('egg') |

+----------------+---------------+

| 4 | 3 |

+----------------+---------------+

(2) 合并字符串的函数：CONCAT(s1,s2,...) 、CONCAT\_WS(x,s1,s2,...)

CONCAT(s1,s2,...) 用于合并字符串 s1, s2, ..... ，如果有一个参数为 NULL，则结果为 NULL

CONCAT\_WS(x,s1,s2,...) 以 x 作为分隔符，将 s1, s2, .... 合并，如果分隔符为 NULL ，则结果为 NULL

mysql> SELECT CONCAT\_WS('-','1st','2nd','3rd'), CONCAT\_WS(NULL,'1st','2nd','3rd');

+----------------------------------+-----------------------------------+

| CONCAT\_WS('-','1st','2nd','3rd') | CONCAT\_WS(NULL,'1st','2nd','3rd') |

+----------------------------------+-----------------------------------+

| 1st-2nd-3rd | NULL |

+----------------------------------+-----------------------------------+

mysql> SELECT CONCAT('My','SQL','5.7'), CONCAT('My',NULL,'SQL');

+--------------------------+-------------------------+

| CONCAT('My','SQL','5.7') | CONCAT('My',NULL,'SQL') |

+--------------------------+-------------------------+

| MySQL5.7 | NULL |

+--------------------------+-------------------------+

(3) 替换字符串的函数：INSERT(s1,x,len,s2)

(1) INSERT(s1,x,len,s2) 用于返回字符串 s1，其子字符串起始于 x 位置和被字符串 s2 取代的 len 字符

(2) 如果 x 超过字符串长度，则返回值为原始字符串，如果 len 的长度大于其他字符串的长度则从位置 x 开始替换

(3) 如果任何一个参数为 NULL ，则返回值为 NULL

mysql> SELECT INSERT('Quest', 2, 4, 'What') AS col1,

-> INSERT('Quest', -1, 4, 'What') AS col2,

-> INSERT('Quest', 3, 100, 'Wh') AS col3;

+-------+-------+------+

| col1 | col2 | col3 |

+-------+-------+------+

| QWhat | Quest | QuWh |

+-------+-------+------+

(4) 转换大小写的函数：LOWER(str) 、LCASE(str) 、UPPER(str) 、UCASE(str)

LOWER(str) 用于将字符串 str 中的字母字符全部转换成小写字母

mysql> SELECT LOWER('BEAUTIFUL'), LOWER('Well');

+--------------------+---------------+

| LOWER('BEAUTIFUL') | LOWER('Well') |

+--------------------+---------------+

| beautiful | well |

+--------------------+---------------+

LCASE(str) 用于将字符串 str 中的字母字符全部转换成小写字母

mysql> SELECT LCASE('BEAUTIFUL'), LCASE('Well');

+--------------------+---------------+

| LCASE('BEAUTIFUL') | LCASE('Well') |

+--------------------+---------------+

| beautiful | well |

+--------------------+---------------+

UPPER(str) 用于将 str 中的小写字母转换成大写字母

mysql> SELECT UPPER('black'), UPPER('BLacK');

+----------------+----------------+

| UPPER('black') | UPPER('BLacK') |

+----------------+----------------+

| BLACK | BLACK |

+----------------+----------------+

UCASE(str) 用于将 str 中的小写字母转换成大写字母

mysql> SELECT UCASE('black'), UCASE('BLacK');

+----------------+----------------+

| UCASE('black') | UCASE('BLacK') |

+----------------+----------------+

| BLACK | BLACK |

+----------------+----------------+

(5) 获取指定长度的字符串的函数：LEFT(s,n) 、RIGHT(s,n)

LEFT(s,n) 用于返回字符串 s 开始的最左边 n 个字符

mysql> SELECT LEFT('football', 5);

+---------------------+

| LEFT('football', 5) |

+---------------------+

| footb |

+---------------------+

RIGHT(s,n) 用于返回字符串 s 中右边 n 个字符

mysql> SELECT RIGHT('football',4);

+---------------------+

| RIGHT('football',4) |

+---------------------+

| ball |

+---------------------+

(6) 填充字符串的函数：LPAD(s1,len,s2) 、RPAD(s1,len,s2)

LPAD(s1,len,s2) 返回字符串 s1 ，其左边由字符串 s2 填充到 len 字符长度，如果 s1 的长度大于 len ，则返回值被缩短至 len 长度

mysql> SELECT LPAD('hello',4,'??'), LPAD('hello',10,'??');

+----------------------+-----------------------+

| LPAD('hello',4,'??') | LPAD('hello',10,'??') | # 字符串'hello'长度大于4,不需要填充，只被缩短

+----------------------+-----------------------+

| hell | ?????hello | # 字符串'hello'长度小于10,因此被填充

+----------------------+-----------------------+

RPAD(s1,len,s2) 返回字符串 s1 ，其右边由字符串 s2 填充到 len 字符长度，如果 s1 的长度大于 len ，则返回值被缩短至 len 长度

mysql> SELECT RPAD('hello',4,'?'), RPAD('hello',10,'?');

+---------------------+----------------------+

| RPAD('hello',4,'?') | RPAD('hello',10,'?') | # 字符串'hello'长度大于4,不需要被填充，因此被缩短至4

+---------------------+----------------------+

| hell | hello????? | # 字符串'hello'长度小于10,因此被填充

+---------------------+----------------------+

(7) 删除空格的函数：LTRIM(s) 、RTRIM(s) 、TRIM(s)

LTRIM(s) 用于删除字符串 s 左侧的空格

mysql> SELECT LTRIM(' book ');

+-----------------+

| LTRIM(' book ') |

+-----------------+

| book |

+-----------------+

RTRIM(s) 用于删除字符串 s 右侧的空格

mysql> SELECT RTRIM(' book ');

+-----------------+

| RTRIM(' book ') |

+-----------------+

| book |

+-----------------+

TRIM(s) 用于删除字符串 s 两侧的空格

mysql> SELECT TRIM(' book ');

+----------------+

| TRIM(' book ') |

+----------------+

| book |

+----------------+

(8) 删除指定字符串的函数：TRIM(s1 FROM s)

TRIM(s1 FROM s) 用于删除字符串 s 中两端所有的子字符串 s1 ，如果没有指定 s1 ，则默认删除字符串 s 两侧的空格

mysql> SELECT TRIM('xy' FROM 'xyxyabcxy');

+-----------------------------+

| TRIM('xy' FROM 'xyxyabcxy') |

+-----------------------------+

| abc |

+-----------------------------+

(9) 重复生成字符串的函数：REPEAT(s,n)

REPEAT(s,n) 用于重复字符串 s ，n 表示重复多少次

mysql> SELECT REPEAT('mysql',3);

+-------------------+

| REPEAT('mysql',3) |

+-------------------+

| mysqlmysqlmysql |

+-------------------+

(10) 空格函数：SPACE(n)

SPACE(n) 用于返回 n 个空格

mysql> SELECT SPACE(20);

+----------------------+

| SPACE(20) |

+----------------------+

| |

+----------------------+

(11) 替换函数：REPLACE(s,s1,s2)

REPLACE(s,s1,s2) 表示使用字符串 s2 替换字符串 s 中所有的字符串 s1

mysql> SELECT REPLACE('xxx.mysql.com', 'x', 'w');

+------------------------------------+

| REPLACE('xxx.mysql.com', 'x', 'w') |

+------------------------------------+

| www.mysql.com |

+------------------------------------+

(12) 比较字符串大小的函数：STRCMP(s1,s2)

STRCMP(s1,s2) 用于比较字符串 s1 和 s2 的大小，若所有字符串相同则返回 0 ，若第一个字符串大于第二个字符串则返回 1 ，若第一个字符串小于第二个字符串则返回 -1

mysql> SELECT STRCMP('txt', 'txt2'), STRCMP('txt', 'txt');

+-----------------------+----------------------+

| STRCMP('txt', 'txt2') | STRCMP('txt', 'txt') |

+-----------------------+----------------------+

| -1 | 0 |

+-----------------------+----------------------+

(13) 获取子字符串的函数：SUBSTRING(s,n,len) 、MID(s,n,len)

SUBSTRING(s,n,len) 用于获取指定位置的子字符串

mysql> SELECT SUBSTRING('breakfast',5) AS col1, # 从第5个字符串开始获取

-> SUBSTRING('breakfast',5,3) AS col2, # 从第5个字符串开始，获取3个

-> SUBSTRING('breakfast',-5) AS col3, # (倒向)从第5个字符串开始获取

-> SUBSTRING('breakfast',-5,3) AS col4; # (倒向)从第5个字符串开始获取，获取3个

+-------+------+-------+------+

| col1 | col2 | col3 | col4 |

+-------+------+-------+------+

| kfast | kfa | kfast | kfa |

+-------+------+-------+------+

MID(s,n,len) 用于获取指定位置的子字符串

mysql> SELECT MID('breakfast',5) AS col1, # 从第5个字符串开始获取

-> MID('breakfast',5,3) AS col2, # 从第5个字符串开始，获取3个

-> MID('breakfast',-5) AS col3, # (倒向)从第5个字符串开始获取

-> MID('breakfast',-5,3) AS col4; # (倒向)从第5个字符串开始获取，获取3个

+-------+------+-------+------+

| col1 | col2 | col3 | col4 |

+-------+------+-------+------+

| kfast | kfa | kfast | kfa |

+-------+------+-------+------+

(14) 匹配子字符串开始位置的函数：LOCATE(str1,str) 、POSITION(str1 IN str) 、INSTR(str, str1)

LOCATE(str1,str) 用于返回字符串 str1 在字符串 str 中的开始位置

mysql> SELECT LOCATE('ball', 'football');

+----------------------------+

| LOCATE('ball', 'football') |

+----------------------------+

| 5 |

+----------------------------+

POSITION(str1 IN str) 用于返回字符串 str1 在字符串 str 中的开始位置

mysql> SELECT POSITION('ball' IN 'football');

+--------------------------------+

| POSITION('ball' IN 'football') |

+--------------------------------+

| 5 |

+--------------------------------+

INSTR(str, str1) 用于返回子字符串 str1 在字符串 str 中的开始位置

mysql> SELECT INSTR('football', 'ball');

+---------------------------+

| INSTR('football', 'ball') |

+---------------------------+

| 5 |

+---------------------------+

(15) 反转字符串的函数：REVERSE(s)

REVERSE(s) 用于将字符串 s 反转

mysql> SELECT REVERSE('abcd');

+-----------------+

| REVERSE('abcd') |

+-----------------+

| dcba |

+-----------------+

(16) 返回指定位置的字符串的函数：ELT(n, s1, s2, s3, .....)

ELT(n, s1, s2, s3, .....) 用于返回第 n 个字符串，如果 n 超出范围则返回 NULL

mysql> SELECT ELT(3, 'a', 'b', 'c', 'd'), ELT(5, 'a', 'b', 'c', 'd');

+----------------------------+----------------------------+

| ELT(3, 'a', 'b', 'c', 'd') | ELT(5, 'a', 'b', 'c', 'd') |

+----------------------------+----------------------------+

| c | NULL |

+----------------------------+----------------------------+

(17) 返回指定字符串位置的函数：FIELD(s, s1, s2, .....)

FIELD(s, s1, s2, .....) 用于返回字符串 s 在列表 s1, s2, .... 中的位置，如果不存在字符串 s 则返回 0 ，如果字符串 s 是 NULL 也返回 0

mysql> SELECT FIELD('hi', 'hihi', 'hey', 'hi', 'bas');

+-----------------------------------------+

| FIELD('hi', 'hihi', 'hey', 'hi', 'bas') |

+-----------------------------------------+

| 3 |

+-----------------------------------------+

(18) 返回子字符串位置的函数：FIND\_IN\_SET(s1, s2)

FIND\_IN\_SET(s1, s2) 用于返回字符串 s1 在字符串列表 s2 中的位置

mysql> SELECT FIND\_IN\_SET('hi', 'hihi,hey,hi,bas'); # 注意s2是一个列表

+--------------------------------------+

| FIND\_IN\_SET('hi', 'hihi,hey,hi,bas') |

+--------------------------------------+

| 3 |

+--------------------------------------+

三、日期和时间函数

(1) 获取当前日期的函数：CURDATE() 、CURRENT\_DATE()

CURDATE() 用于获取系统当前日期

mysql> SELECT CURDATE();

+------------+

| CURDATE() |

+------------+

| 2017-05-23 |

+------------+

CURRENT\_DATE() 用于系统获取当前日期

mysql> SELECT CURRENT\_DATE();

+----------------+

| CURRENT\_DATE() |

+----------------+

| 2017-05-23 |

+----------------+

(2) 获取当前时间的函数：CURTIME() 、CURRENT\_TIME()

CURTIME() 用于获取系统当前时间

mysql> SELECT CURTIME();

+-----------+

| CURTIME() |

+-----------+

| 20:25:23 |

+-----------+

(3) 获取当前日期和时间的函数：CURRENT\_TIMESTAMP() 、LOCALTIME() 、NOW() 、SYSDATE()

CURRENT\_TIMESTAMP() 用于获取系统当前日期和时间

mysql> SELECT CURRENT\_TIMESTAMP();

+---------------------+

| CURRENT\_TIMESTAMP() |

+---------------------+

| 2017-05-23 20:26:48 |

+---------------------+

LOCALTIME() 用于获取系统当前日期和时间

mysql> SELECT LOCALTIME();

+---------------------+

| LOCALTIME() |

+---------------------+

| 2017-05-23 20:27:29 |

+---------------------+

NOW() 用于获取系统当前日期和时间

mysql> SELECT NOW();

+---------------------+

| NOW() |

+---------------------+

| 2017-05-23 20:28:10 |

+---------------------+

SYSDATE() 用于获取系统当前日期和时间

mysql> SELECT SYSDATE();

+---------------------+

| SYSDATE() |

+---------------------+

| 2017-05-23 20:29:02 |

+---------------------+

(4) 获取时间戳的函数：UNIX\_TIMESTAMP()

UNIX\_TIMESTAMP() 用于获取 UNIX 格式的时间戳

mysql> SELECT UNIX\_TIMESTAMP();

+------------------+

| UNIX\_TIMESTAMP() |

+------------------+

| 1495542689 |

+------------------+

(5) 转换时间戳的函数：FROM\_UNIXTIME()

FROM\_UNIXTIME() 用于将 UNIX 格式的时间戳转换为普通格式的时间

mysql> SELECT FROM\_UNIXTIME('1495542689');

+-----------------------------+

| FROM\_UNIXTIME('1495542689') |

+-----------------------------+

| 2017-05-23 20:31:29 |

+-----------------------------+

(6) 获取 UTC 日期的函数：UTC\_DATE()

UTC\_DATE() 用于获取当前 UTC (世界标准时间) 日期值

mysql> SELECT UTC\_DATE();

+------------+

| UTC\_DATE() |

+------------+

| 2017-05-23 |

+------------+

(7) 获取 UTC 时间的函数：UTC\_TIME()

UTC\_TIME() 用于获取当前 UTC (世界标准时间) 时间值

mysql> SELECT UTC\_TIME();

+------------+

| UTC\_TIME() |

+------------+

| 12:36:29 |

+------------+

(8) 获取月份的函数：MONTH(date) 、MONTHNAME(date)

MONTH(date) 用于返回 date 对应的月份

mysql> SELECT MONTH('2017-02-23');

+---------------------+

| MONTH('2017-02-23') |

+---------------------+

| 2 |

+---------------------+

MONTHNAME(date) 用于返回 date 对应月份的英文全名

mysql> SELECT MONTHNAME('2017-02-23');

+-------------------------+

| MONTHNAME('2017-02-23') |

+-------------------------+

| February |

+-------------------------+

(9) 获取星期的函数：DAYNAME(date) 、DAYOFWEEK(date) 、WEEKDAY(date) 、WEEK(date) 、WEEKOFYEAR(date)

DAYNAME(date) 用于返回 date 对应的工作日的英文名称

mysql> SELECT DAYNAME('2017-02-23');

+-----------------------+

| DAYNAME('2017-02-23') |

+-----------------------+

| Thursday |

+-----------------------+

DAYOFWEEK(date) 用于返回 date 对应的一周中的索引，1 表示周日，2 表示周一，...... ，7 表示周六

mysql> SELECT DAYOFWEEK('2017-02-23');

+-------------------------+

| DAYOFWEEK('2017-02-23') |

+-------------------------+

| 5 |

+-------------------------+

WEEKDAY(date) 用于返回日期对应的工作日索引，0 表示周一，1 表示周二，...... ，6 表示周日

mysql> SELECT WEEKDAY('2017-05-23');

+-----------------------+

| WEEKDAY('2017-05-23') |

+-----------------------+

| 1 |

+-----------------------+

WEEK(date) 用于计算 date 是一年中的第几周，一年有 53 周

mysql> SELECT WEEK('2017-05-23');

+--------------------+

| WEEK('2017-05-23') |

+--------------------+

| 21 |

+--------------------+

WEEKOFYEAR(date) 用于计算日期 date 是一年中的第几周，一年有 53 周

mysql> SELECT WEEKOFYEAR('2017-05-23');

+--------------------------+

| WEEKOFYEAR('2017-05-23') |

+--------------------------+

| 21 |

+--------------------------+

(10) 获取天数的函数：DAYOFYEAR(date) 、DAYOFMONTH(date)

DAYOFYEAR(date) 用于返回 date 是一年中的第几天，一年有 365 天

mysql> SELECT DAYOFYEAR('2017-05-23');

+-------------------------+

| DAYOFYEAR('2017-05-23') |

+-------------------------+

| 143 |

+-------------------------+

DAYOFMONTH(date) 用于计算 date 是一个月中的第几天

mysql> SELECT DAYOFMONTH('2017-05-23');

+--------------------------+

| DAYOFMONTH('2017-05-23') |

+--------------------------+

| 23 |

+--------------------------+

(11) 获取年份的函数：YEAR(date)

YEAR(date) 返回 date 对应的年份

mysql> SELECT YEAR('11-02-03'), YEAR('98-02-03');

+------------------+------------------+

| YEAR('11-02-03') | YEAR('98-02-03') |

+------------------+------------------+

| 2011 | 1998 |

+------------------+------------------+

(12) 获取季度的函数：QUARTER(date)

QUARTER(date) 返回 date 对应的一年中的季度值

mysql> SELECT QUARTER('17-05-23');

+---------------------+

| QUARTER('17-05-23') |

+---------------------+

| 2 |

+---------------------+

(13) 获取分钟的函数：MINUTE(time)

MINUTE(time) 返回 time 对应的分钟值

mysql> SELECT MINUTE('17-02-03 10:10:03');

+-----------------------------+

| MINUTE('17-02-03 10:10:03') |

+-----------------------------+

| 10 |

+-----------------------------+

(14) 获取秒钟的函数：SECOND(time)

SECOND(time) 返回 time 对应的秒数

mysql> SELECT SECOND('10:05:03');

+--------------------+

| SECOND('10:05:03') |

+--------------------+

| 3 |

+--------------------+

(15) 获取日期的指定值的函数：EXTRACT(type FROM date)

EXTRACT(type FROM date) 用于获取指定的日期值

mysql> SELECT EXTRACT(YEAR FROM '2016-07-02') AS col1, # 当type为YEAR时，只返回年值

-> EXTRACT(YEAR\_MONTH FROM '2016-07-02 01:02:03') AS col2, # 当type为YEAR\_MONTH时，返回年与月

-> EXTRACT(DAY\_MINUTE FROM '2016-07-02 01:02:03') AS col3; # 当type为DAY\_MINUTE时，返回日、小时、分钟

+------+--------+-------+

| col1 | col2 | col3 |

+------+--------+-------+

| 2016 | 201607 | 20102 |

+------+--------+-------+

(16) 时间和秒钟转换的函数：TIME\_TO\_SEC(time) 、SEC\_TO\_TIME(time)

TIME\_TO\_SEC(time) 用于将 time 转换为秒钟，公式为 " 小时\*3600 + 分钟\*60 + 秒 "

mysql> SELECT TIME\_TO\_SEC('23:23:00');

+-------------------------+

| TIME\_TO\_SEC('23:23:00') |

+-------------------------+

| 84180 |

+-------------------------+

SEC\_TO\_TIME(time) 用于将秒值转换为时间格式

mysql> SELECT SEC\_TO\_TIME('84180');

+----------------------+

| SEC\_TO\_TIME('84180') |

+----------------------+

| 23:23:00 |

+----------------------+

(17) 计算日期和时间的函数：DATE\_ADD() 、ADDDATE() 、DATE\_SUB() 、SUBDATE() 、ADDTIME() 、SUBTIME() 、DATE\_DIFF()

DATE\_ADD() 用于对日期进行加运算，格式为 DATE\_ADD(date, INTERVAL expr type) ，expr 与 type 的关系

mysql> SELECT DATE\_ADD('2010-12-31 23:59:59', INTERVAL 1 SECOND); # 对指定的日期增加1秒

+----------------------------------------------------+

| DATE\_ADD('2010-12-31 23:59:59', INTERVAL 1 SECOND) |

+----------------------------------------------------+

| 2011-01-01 00:00:00 |

+----------------------------------------------------+

mysql> SELECT DATE\_ADD('2010-12-31 23:59:59', INTERVAL '1:1' MINUTE\_SECOND); # 对指定的日期增加1分1秒

+---------------------------------------------------------------+

| DATE\_ADD('2010-12-31 23:59:59', INTERVAL '1:1' MINUTE\_SECOND) |

+---------------------------------------------------------------+

| 2011-01-01 00:01:00 |

+---------------------------------------------------------------+

DATE\_SUB() 用于对日期进行减运算，格式为 DATE\_SUB(date, INTERVAL expr type) ，expr 与 type 的关系

mysql> SELECT DATE\_SUB('2011-01-02', INTERVAL 31 DAY); # 给指定的日期减去31天

+-----------------------------------------+

| DATE\_SUB('2011-01-02', INTERVAL 31 DAY) |

+-----------------------------------------+

| 2010-12-02 |

+-----------------------------------------+

SUBDATE() 用于对日期进行减运算，格式为 SUBDATE(date, INTERVAL expr type) ，expr 与 type 的关系

mysql> SELECT SUBDATE('2011-01-02', INTERVAL 31 DAY); # 对指定的日期减去31天

+----------------------------------------+

| SUBDATE('2011-01-02', INTERVAL 31 DAY) |

+----------------------------------------+

| 2010-12-02 |

+----------------------------------------+

ADDTIME() 用于对日期进行加运算，格式为 ADDTIME(date, expr)

mysql> SELECT ADDTIME('2000-12-31 23:59:59', '1:1:1'); # 给指定的日期增加1时1分1秒

+-----------------------------------------+

| ADDTIME('2000-12-31 23:59:59', '1:1:1') |

+-----------------------------------------+

| 2001-01-01 01:01:00 |

+-----------------------------------------+

SUBTIME() 用于对日期进行减运算，格式为 SUBTIME(date, expr)

mysql> SELECT SUBTIME('2000-12-31 23:59:59', '1:1:1'); # 给指定的日期减去1时1分1秒

+-----------------------------------------+

| SUBTIME('2000-12-31 23:59:59', '1:1:1') |

+-----------------------------------------+

| 2000-12-31 22:58:58 |

+-----------------------------------------+

DATE\_DIFF() 用于计算两个日期之间的间隔天数

mysql> SELECT DATEDIFF('2017-12-31', '2010-12-31');

+--------------------------------------+

| DATEDIFF('2017-12-31', '2010-12-31') |

+--------------------------------------+

| 2557 |

+--------------------------------------+

(18) 将日期和时间格式化的函数：DATE\_FORMAT(date, format) 、TIME\_FORMAT(time, format) 、GET\_FORMAT(val\_type, format\_type)

DATE\_FORMAT(date, format) 用于格式化日期，即根据 format 指定的格式显示 date 值，format 格式

mysql> SELECT DATE\_FORMAT('1997-10-04 22:23:00', '%W %M %Y');

+------------------------------------------------+

| DATE\_FORMAT('1997-10-04 22:23:00', '%W %M %Y') |

+------------------------------------------------+

| Saturday October 1997 |

+------------------------------------------------+

TIME\_FORMAT(time, format) 用于格式化时间，即根据 format 指定的格式显示 time 值，format 格式

mysql> SELECT TIME\_FORMAT('16:00:00', '%H %k %I');

+-------------------------------------+

| TIME\_FORMAT('16:00:00', '%H %k %I') |

+-------------------------------------+

| 16 16 04 |

+-------------------------------------+

GET\_FORMAT() ，我们指定值类型和格式化类型，然后会显示成格式字符串

mysql> SELECT DATE\_FORMAT('2000-10-05 22:23:00', GET\_FORMAT(DATE,'USA'));

+------------------------------------------------------------+

| DATE\_FORMAT('2000-10-05 22:23:00', GET\_FORMAT(DATE,'USA')) |

+------------------------------------------------------------+

| 10.05.2000 |

+------------------------------------------------------------+

%d该月日期，数字形式（00..31）

%e该月日期，数字形式（0..31）

%H以2位数表示24小时（00..23）

%h,%I 以2位数表示12小时（01..12）

%i分钟，数字形式（00-59）

%j一年中的天数（001-366）

%k以24小时（0-23）

%l以12小时（0..12）

%M月份名称（january..December）

%m月份数字形式（00..12）

%p上午（AM）或下午（PM）

%r时间，12小时制（小时hh：分钟mm：秒钟ss后面加AM或PM）

%S,%s以2位数形式表示秒（00..59）

%T时间，24小时制（小时hh：分钟mm：秒数ss）

%W工作日名称（周日..周六）

%w一周中的每日（0=周日..6=周六）

%Y4位数形式表示年份

%y2位数形式表示年份

四、条件判断函数

(1) IF()

IF(expr, v1, v2) 如果表达式 expr 为 TRUE ，则返回值为 v1 ，否则返回 v2

mysql> SELECT IF(1>2, 2, 3);

+---------------+

| IF(1>2, 2, 3) |

+---------------+

| 3 |

+---------------+

(2) IFNULL()

IFNULL(v1, v2) ，如果 v1 不为 NULL ，则返回值为 v1 ；如果 v1 为 NULL ，则返回值为 v2

mysql> SELECT IFNULL(1,2), IFNULL(NULL,10);

+-------------+-----------------+

| IFNULL(1,2) | IFNULL(NULL,10) |

+-------------+-----------------+

| 1 | 10 |

+-------------+-----------------+

(3) CASE

语法：CASE expr WHEN v1 THEN r1 [WHEN v2 THEN r2] [ELSE rn] END

含义：如果 expr 等于某个 vn ，则返回对应位置 THEN 后面的结果，如果与所有值都不相等，则返回 ELSE 后面的 rn

mysql> SELECT CASE 2 WHEN 1 THEN 'one' WHEN 2 THEN 'two' ELSE 'more' END;

+------------------------------------------------------------+

| CASE 2 WHEN 1 THEN 'one' WHEN 2 THEN 'two' ELSE 'more' END |

+------------------------------------------------------------+

| two |

+------------------------------------------------------------+

五、系统信息函数

(1) 获取 MySQL 版本号的函数：VERSION()

VERSION() 用于获取 MySQL 版本号

mysql> SELECT VERSION();

+------------+

| VERSION() |

+------------+

| 5.7.18-log |

+------------+

(2) 查看当前用户的连接数的ID函数：CONNECTION\_ID()

CONNECTION\_ID() 用于查看当前用户的连接数

mysql> SELECT CONNECTION\_ID();

+-----------------+

| CONNECTION\_ID() |

+-----------------+

| 10 |

+-----------------+

mysql> SHOW PROCESSLIST; # 查看当前用户的连接信息

+----+------+-----------------+------+---------+------+----------+------------------+

| Id | User | Host | db | Command | Time | State | Info |

+----+------+-----------------+------+---------+------+----------+------------------+

| 10 | root | localhost:52421 | NULL | Query | 0 | starting | SHOW PROCESSLIST |

+----+------+-----------------+------+---------+------+----------+------------------+

(1) Id ：用户登录 MySQL 时，系统分配的连接 id

(2) User ：当前连接的用户

(3) Host ：显示这个语句是从哪个 IP 的哪个端口上发出的，可以用来追踪出现问题语句的用户

(4) db ：显示这个进程目前连接的是哪个数据库

(5) Command ：显示当前连接执行的命令，一般取值为休眠(Sleep)、查询(Query)、连接(Connect)

(6) Time ：显示这个状态持续的时间，单位是秒

(7) State ：显示使用当前连接的 SQL 语句的状态

(8) Info ：显示这个 SQL 语句

(3) 查看当前使用的数据库的函数：DATABASE() 、SCHEMA()

DATABASE() 用于查看当前使用的数据库

mysql> SELECT DATABASE();

+------------+

| DATABASE() |

+------------+

| test\_db |

+------------+

SCHEMA() 用于查看当前使用的数据库

mysql> SELECT SCHEMA();

+----------+

| SCHEMA() |

+----------+

| test\_db |

+----------+

(4) 查看当前登录的用户名的函数：USER() 、CURRENT\_USER() 、SYSTEM\_USER()

USER() 返回当前登录的用户及主机名

mysql> SELECT USER();

+----------------+

| USER() |

+----------------+

| root@localhost |

+----------------+

CURRENT\_USER() 用于返回当前登录的用户及主机名

mysql> SELECT CURRENT\_USER();

+----------------+

| CURRENT\_USER() |

+----------------+

| root@localhost |

+----------------+

SYSTEM\_USER() 用于返回当前登录的用户及主机名

mysql> SELECT SYSTEM\_USER();

+----------------+

| SYSTEM\_USER() |

+----------------+

| root@localhost |

+----------------+

(5) 查看指定字符串的字符集的函数：CHARSET(str)

CHARSET(str) 用于查看字符串 str 的字符集

mysql> SELECT CHARSET('abc');

+----------------+

| CHARSET('abc') |

+----------------+

| utf8 |

+----------------+

(6) 查看指定字符串的排列方式的函数：COLLATION(str)

COLLATION(str) 用于查看字符串 str 的字符排列方式

mysql> SELECT COLLATION('abc');

+------------------+

| COLLATION('abc') |

+------------------+

| utf8\_general\_ci |

+------------------+

(7) 获取最后一个自动生成的 ID 值得函数：LAST\_INSERT\_ID()

LAST\_INSERT\_ID() 用于获取最后一个自动生成的 ID 值

mysql> CREATE TABLE worker

-> (

-> id INT AUTO\_INCREMENT PRIMARY KEY, # 先创建一个表，其id字段带有AUTO\_INCREMENT约束

-> name VARCHAR(30)

-> );

mysql> INSERT INTO worker VALUES (NULL,'jimy'); # 插入一条数据，这时id没有指定，则自动生成，id为1

mysql> INSERT INTO worker VALUES (NULL,'Tom'); # 插入一条数据，这时id没有指定，则自动生成，id为2

mysql> SELECT \* FROM worker; # 查看表的信息

+----+------+

| id | name |

+----+------+

| 1 | jimy |

| 2 | Tom |

+----+------+

六、加/解密函数

(1) 加密函数：PASSWORD(str) 、MD5(str) 、ENCODE(str, pswd\_str)

PASSWORD(str) 从明文密码 str 计算并返回加密后的密码字符串，当参数为 NULL 时，返回 NULL

mysql> SELECT PASSWORD('newpwd');

+-------------------------------------------+

| PASSWORD('newpwd') |

+-------------------------------------------+

| \*1FA85AA204CC12B39B20E8F1E839D11B3F9E6AA4 |

+-------------------------------------------+

MD5(str) 为字符串 str 算出一个 MD5 128 比特校验和

mysql> SELECT MD5('newpwd');

+----------------------------------+

| MD5('newpwd') |

+----------------------------------+

| a5e3094ce553e08de5ba237525b106d5 |

+----------------------------------+

ENCODE(str, pswd\_str) 使用 pswd\_str 作为密码，加密 str

mysql> SELECT ENCODE('secret', 'newpwd');

+----------------------------+

| ENCODE('secret', 'newpwd') |

+----------------------------+

| ü­EE |

+----------------------------+

(2) 解密函数：DECODE(crypt\_str, pswd\_str)

DECODE(crypt\_str, pswd\_str) 使用 pswd\_str 作为密码，解密加密字符串 crypt\_str

mysql> SELECT DECODE(ENCODE('secret','cry'), 'cry');

+---------------------------------------+

| DECODE(ENCODE('secret','cry'), 'cry') |

+---------------------------------------+

| secret |

+---------------------------------------+

七、其他函数

(1) 格式化函数：FORMAT(x, n)

FORMAT(x, n) 将数字 x 格式化，并以四舍五入的方式保留小数点后 n 位，结果以字符串的形式返回

mysql> SELECT FORMAT(1.23456, 4), FORMAT(1.2, 4), FORMAT(1.234, 0);

+--------------------+----------------+------------------+

| FORMAT(1.23456, 4) | FORMAT(1.2, 4) | FORMAT(1.234, 0) |

+--------------------+----------------+------------------+

| 1.2346 | 1.2000 | 1 |

+--------------------+----------------+------------------+

(2) 不同进制的数字进行转换的函数：CONV()

CONV() 用于不同进制数之间的转换

mysql> SELECT CONV('a',16,2), # 将16进制的a转换为2进制

-> CONV(15,10,2), # 将10进制的15转换为2进制

-> CONV(15,10,8), # 将10进制的15转换为8进制

-> CONV(15,10,16); # 将10进制的15转换为16进制

+----------------+---------------+---------------+----------------+

| CONV('a',16,2) | CONV(15,10,2) | CONV(15,10,8) | CONV(15,10,16) |

+----------------+---------------+---------------+----------------+

| 1010 | 1111 | 17 | F |

+----------------+---------------+---------------+----------------+

(3) IP 地址与数字互相转换的函数：INET\_ATON(expr) 、INET\_NTOA(expr)

INET\_ATON(expr) 用于将网络地址转换为一个代表该地址数值的整数

mysql> SELECT INET\_ATON('192.168.1.1');

+--------------------------+

| INET\_ATON('192.168.1.1') |

+--------------------------+

| 3232235777 |

+--------------------------+

(4) 加锁函数和解锁函数：GET\_LOCK(str, timeout) 、RELEASE\_LOCAK(str) 、

IS\_FREE\_LOCK(str) 、IS\_USED\_LOCK(str)

GET\_LOCK(str, timeout) 使用字符串 str 来得到一个锁，持续时间 timeout 秒

(1) 若成功得到锁，则返回 1

(2) 若操作超时，则返回 0

(3) 若发生错误，则返回 NULL

mysql> SELECT GET\_LOCK('lock1', 10);

+-----------------------+

| GET\_LOCK('lock1', 10) |

+-----------------------+

| 1 | # 返回结果为1，说明成功得到了一个名称为'lock1'的锁，持续时间为10秒

+-----------------------+

RELEASE\_LOCAK(str) 用于解开被 GET\_LOCK() 获取的，用字符串 str 所命名的锁

(1) 若锁被解开，则返回 1

(2) 若该线程尚未创建锁，则返回 0

(3) 若命名的锁不存在，则返回 NULL

(4) 若该锁从未被 GET\_LOCK() 的调用获取，或锁已经被提前解开，则该锁不存在

mysql> SELECT RELEASE\_LOCK('lock1');

+-----------------------+

| RELEASE\_LOCK('lock1') |

+-----------------------+

| 1 | # 返回值为1说明解锁成功

+-----------------------+

IS\_FREE\_LOCK(str) 检查名为 str 的锁是否可以使用

(1) 若锁可以使用，则返回 1

(2) 若锁正在被使用，则返回 0

(3) 若出现错误，则返回 NULL

mysql> SELECT IS\_FREE\_LOCK('lock1');

+-----------------------+

| IS\_FREE\_LOCK('lock1') |

+-----------------------+

| 1 | # 返回值为1说明锁可以使用

+-----------------------+

IS\_USED\_LOCK(str) 用于检查名为 str 的锁是否正在被使用，若被封锁，则返回使用该锁的客户端的连接标识符，否则返回 NULL

mysql> SELECT IS\_USED\_LOCK('lock1');

+-----------------------+

| IS\_USED\_LOCK('lock1') |

+-----------------------+

| 10 | # 返回结果为当前连接ID，表示名称为'lock1'的锁正在被使用

+-----------------------+-

(5) 重复执行指定操作的函数：BENCHMARK(count, expr)

BENCHMARK(count, expr) 用于重复 count 次执行表达式 expr

(1) 可以用于计算 MySQL 处理表达式的速度

(2) 可以在 MySQL 客户端内部报告语句执行的时间

mysql> SELECT PASSWORD('newpwd');

+-------------------------------------------+

| PASSWORD('newpwd') |

+-------------------------------------------+

| \*1FA85AA204CC12B39B20E8F1E839D11B3F9E6AA4 |

+-------------------------------------------+

1 row in set, 1 warning (0.00 sec) # 执行1次加密操作花费了0.00秒

mysql> SELECT BENCHMARK( 500000, PASSWORD('newpwd') );

+-----------------------------------------+

| BENCHMARK( 500000, PASSWORD('newpwd') ) |

+-----------------------------------------+

| 0 |

+-----------------------------------------+

1 row in set, 1 warning (0.16 sec) # 执行500000次加密操作花费了0.16秒

(6) 改变字符集的函数：CONVERT(... USING ...)

CONVERT(... USING ...) 用于改变字符串的默认字符集

mysql> SELECT CHARSET('abc'); # 默认是utf8字符集

+----------------+

| CHARSET('abc') |

+----------------+

| utf8 |

+----------------+

mysql> SELECT CHARSET(CONVERT('abc' USING latin1)); # 转换成latin1字符集

+--------------------------------------+

| CHARSET(CONVERT('abc' USING latin1)) |

+--------------------------------------+

| latin1 |

+--------------------------------------+

(7) 改变数据类型的函数：CAST(x, AS type) 、CONVERT(x, type)

CAST(x, AS type) 用于将一个数据类型的值转换为另一个数据类型的值

CONVERT(x, type) 用于将一个数据类型的值转换为另一个数据类型的值

复制代码

mysql> SELECT CAST(100 AS CHAR(2)); # 将整数类型100转换为带有两个显示宽度的字符串类型，结果为'10'

+----------------------+

| CAST(100 AS CHAR(2)) |

+----------------------+

| 10 |

+----------------------+

CONVERT(x, type) 用于将一个数据类型的值转换为另一个数据类型的值

mysql> SELECT CONVERT(100, CHAR(2)); # 将整数类型的100转换为带有两个显示宽度的字符串类型，结果为'10'

+-----------------------+

| CONVERT(100, CHAR(2)) |

+-----------------------+

| 10 |

+-----------------------+