知数堂-MySQL-内置函数手册

写在前面本，这个手册太长，可以保存搜索着看，或是累了，休息时看一下。

版本信息

时间	作者	版本
2016-11-19	知数堂 Wing	init

关注我了解更多：

常用的MySQL内置函数

MySQL存在很多内置的函数，往往会给我们的工作带来很多方便，接下来就介绍下常用的MySQL内置函数。

比较函数

COALESCE():

返回参数中第一个不是NULL的值。

```
root@localhost : wing 11:23:22> select coalesce(null,null,1,2,3,'test')
;
+---------------------------------+
| coalesce(null,null,1,2,3,'test') |
+---------------------------------+
| 1                               |
+---------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 11:39:14> select coalesce(null,'test',null,1,2);
+-------------------------------+
| coalesce(null,'test',null,1,2) |
+-------------------------------+
| test                          |
+-------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 11:42:54> select coalesce('test',null,1,2);
+---------------------------+
| coalesce('test',null,1,2) |
+---------------------------+
| test                      |
+---------------------------+
1 row in set (0.00 sec)
```

GREATEST():

返回最大的参数,若参数中存在NULL值,则返回NULL值。

```
root@localhost : wing 11:45:24> select greatest(1,2,null,3,4);
+-----------------------+
| greatest(1,2,null,3,4) |
+-----------------------+
|                  NULL |
+-----------------------+
1 row in set (0.00 sec)

root@localhost : wing 11:45:29> select greatest(1,2,3,4);
+-------------------+
| greatest(1,2,3,4) |
+-------------------+
|                 4 |
+-------------------+
1 row in set (0.00 sec)
```

```
root@localhost : wing 11:45:33> select greatest(1,2,3,4,'test');
+-------------------------+
| greatest(1,2,3,4,'test') |
+-------------------------+
| 4                       |
+-------------------------+
1 row in set, 1 warning (0.00 sec)

Warning (Code 1292): Truncated incorrect DOUBLE value: 'test'
```

IN():

检验某个值是否存在于IN的集合中。

INTERVAL():

返回第一个比第一个参数小的索引值,其内部参数必须为数值型,如果第一个参数为NULL值,则返回-1。

```
root@localhost : wing 11:47:18> select interval(null,1,2,3,4);
+-----------------------+
| interval(null,1,2,3,4) |
+-----------------------+
|                    -1 |
+-----------------------+
1 row in set (0.00 sec)

root@localhost : wing 11:54:34> select interval(2,1,2,3,4);
+--------------------+
| interval(2,1,2,3,4) |
+--------------------+
|                  2 |
+--------------------+
1 row in set (0.00 sec)
```

ISNULL():

判断其参数是否为NULL值。

```
root@localhost : wing 11:54:39> select isnull(null);
+--------------+
| isnull(null) |
+--------------+
|            1 |
+--------------+
```

```
1 row in set (0.00 sec)

root@localhost : wing 11:57:34> select isnull(0);
+-----------+
| isnull(0) |
+-----------+
|         0 |
+-----------+
1 row in set (0.00 sec)

root@localhost : wing 11:57:37> select isnull('test');
+----------------+
| isnull('test') |
+----------------+
|              0 |
+----------------+
1 row in set (0.00 sec)
```

LEAST():

返回参数中最小的值,如果参数中存在NULL值,则返回NULL。

```
root@localhost : wing 11:57:44> select least(null,1,2,3);
+-------------------+
| least(null,1,2,3) |
+-------------------+
|              NULL |
+-------------------+
1 row in set (0.00 sec)

root@localhost : wing 12:00:05> select least(3,2,7);
+--------------+
| least(3,2,7) |
+--------------+
|            2 |
+--------------+
1 row in set (0.00 sec)

root@localhost : wing 12:00:13> select least(3,2,7,'test');
+---------------------+
| least(3,2,7,'test') |
+---------------------+
| 0                   |
+---------------------+
1 row in set, 1 warning (0.00 sec)

Warning (Code 1292): Truncated incorrect DOUBLE value: 'test'
root@localhost : wing 12:00:22> select least(3,2,7,null,'test');
+--------------------------+
| least(3,2,7,null,'test') |
```

```
+--------------------------+
| NULL                     |
+--------------------------+
1 row in set (0.00 sec)
```

STRCMP(expr1,expr2)

字符串比较函数

expr1 expr1=expr2返回 0
expr1>expr2返回 1

```
root@localhost : wing 12:08:48> select strcmp('tast','test');
+-----------------------+
| strcmp('tast','test') |
+-----------------------+
|                    -1 |
+-----------------------+
1 row in set (0.00 sec)

root@localhost : wing 12:08:52> select strcmp('test','test');
+-----------------------+
| strcmp('test','test') |
+-----------------------+
|                     0 |
+-----------------------+
1 row in set (0.00 sec)

root@localhost : wing 12:08:54> select strcmp('test','tast');
+-----------------------+
| strcmp('test','tast') |
+-----------------------+
|                     1 |
+-----------------------+
1 row in set (0.00 sec)
```

赋值操作

1. ':='不同于=之处在于,':='操作从来不解释为一个比较操作符,所以可以使用':='操作符将任何一个有效的SQL语句赋值到变量中。
2. '='只能用于两个场景下:
   SET语句中'='相当于':='
   EXAMPLE

```
# 未赋值成功,=相当于比较操作
root@localhost : wing 01:10:02> select @a=1;
+------+
| @a=1 |
+------+
| NULL |
+------+
1 row in set (0.00 sec)

# 赋值成功
root@localhost : wing 01:25:05> select @a:=1;
+-------+
| @a:=1 |
+-------+
|     1 |
+-------+
1 row in set (0.00 sec)
```

控制流函数

CASE

语法

```
sql
CASE value WHEN [compare_value] THEN result [WHEN [compare_value] THEN
result ...] [ELSE result] END
```

```
sql
CASE WHEN [condition] THEN result [WHEN [condition] THEN result ...] [ELSE
result] END
```

实战演练

```
root@localhost : wing 02:07:15> select * from fun_data;
+------+--------+
| id   | name   |
+------+--------+
|    1 | MySQL  |
|    2 | Oracle |
|    3 | Python |
| NULL | SHELL  |
|    5 | NULL   |
+------+--------+
5 rows in set (0.00 sec)

# CASE WHEN ...语句实战演练
root@localhost : wing 02:07:14> select case when id is null then 0 else
```

```
 id end from fun_data;
+-----------------------------------------+
| case when id is null then 0 else id end |
+-----------------------------------------+
|                                       1 |
|                                       2 |
|                                       3 |
|                                       0 |
|                                       5 |
+-----------------------------------------+
5 rows in set (0.00 sec)

# CASE value WHEN ...语句实战演练
root@localhost : wing 02:08:46> select case name when 'SHELL' then 'LIN
UX' else name end from fun_data;
+-----------------------------------------------------+
| case name when 'SHELL' then 'LINUX' else name end |
+-----------------------------------------------------+
| MySQL                                               |
| Oracle                                              |
| Python                                              |
| LINUX                                               |
| NULL                                                |
+-----------------------------------------------------+
5 rows in set (0.00 sec)
```

IF(expr1,expr2,expr3)

如果expr1为TRUE,即expr1<>0以及expr1<>NULL,此时返回expr2;

如果expr1为FALSE,即exor1 is null或expr1=0,此时返回expr3。

IFNULL(expr1,expr2)

如果expr1为NULL值,此时返回expr2;

如果expr1不为NULL值,此时返回expr1。

```
root@localhost : wing 04:15:44> select ifnull(null,0);
+----------------+
| ifnull(null,0) |
+----------------+
|              0 |
+----------------+
1 row in set (0.00 sec)

root@localhost : wing 04:34:18> select ifnull(3,0);
+-------------+
```

```
| ifnull(3,0) |
+-------------+
|           3 |
+-------------+
1 row in set (0.00 sec)

root@localhost : wing 04:34:23> select ifnull('test',0);
+------------------+
| ifnull('test',0) |
+------------------+
| test             |
+------------------+
1 row in set (0.00 sec)

root@localhost : wing 04:34:30> select ifnull(0,1);
+-------------+
| ifnull(0,1) |
+-------------+
|           0 |
+-------------+
1 row in set (0.00 sec)
```

NULLIF(expr1,expr2)

如果expr1=exor2,则返回NULL;

如果expr1<>expr2,则返回expr1。

```
root@localhost : wing 04:37:56> select nullif(null,null);
+-------------------+
| nullif(null,null) |
+-------------------+
| NULL              |
+-------------------+
1 row in set (0.00 sec)

root@localhost : wing 04:39:06> select nullif(1024,1024);
+-------------------+
| nullif(1024,1024) |
+-------------------+
|              NULL |
+-------------------+
1 row in set (0.00 sec)

root@localhost : wing 04:39:18> select nullif(1024,512);
+-----------------+
| nullif(1024,512) |
+-----------------+
|            1024 |
```

```
+------------------+
1 row in set (0.00 sec)

root@localhost : wing 04:39:24> select nullif(512,1024);
+------------------+
| nullif(512,1024) |
+------------------+
|              512 |
+------------------+
1 row in set (0.00 sec)
```

字符串函数

LENGTH()

返回字符串的字节数

CHAR_LENGTH()

返回字符串的字符数

BIT_LENGTH()

返回字符串的位数

```
root@localhost : wing 04:48:06> select char_length('MySQL数据库');
+-------------------------------+
| char_length('MySQL数据库')    |
+-------------------------------+
|                             8 |
+-------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 04:48:21> select length('MySQL数据库');
+--------------------------+
| length('MySQL数据库')    |
+--------------------------+
|                       14 |
+--------------------------+
1 row in set (0.00 sec)

root@localhost : wing 04:48:25> select bit_length('MySQL数据库');
+-------------------------------+
| bit_length('MySQL数据库')    |
+-------------------------------+
|                           112 |
+-------------------------------+
```

```
1 row in set (0.00 sec)
```

CONCAT()

对字符串进行合并

```
root@localhost : wing 04:50:15> select concat('MySQL','数据库');
+-----------------------------+
| concat('MySQL','数据库')    |
+-----------------------------+
| MySQL数据库                 |
+-----------------------------+
1 row in set (0.00 sec)
```

ELT(expr1,expr2....)

根据expr1选择expr2....对应的数据

```
root@localhost : wing 05:23:19> select elt(1,'MySQL','Oracle','Python')
;
+---------------------------------+
| elt(1,'MySQL','Oracle','Python') |
+---------------------------------+
| MySQL                           |
+---------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 05:23:41> select elt(2,'MySQL','Oracle','Python')
;
+---------------------------------+
| elt(2,'MySQL','Oracle','Python') |
+---------------------------------+
| Oracle                          |
+---------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 05:23:45> select elt(0,'MySQL','Oracle','Python')
;
+---------------------------------+
| elt(0,'MySQL','Oracle','Python') |
+---------------------------------+
| NULL                            |
+---------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 05:23:49> select elt(5,'MySQL','Oracle','Python')
;
```

```
+-------------------------------+
| elt(5,'MySQL','Oracle','Python') |
+-------------------------------+
| NULL                          |
+-------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 05:23:54> select elt(,'MySQL','Oracle','Python');
ERROR 1064 (42000): You have an error in your SQL syntax; check the man
ual that corresponds to your MySQL server version for the right syntax
to use near ''MySQL','Oracle','Python')' at line 1
root@localhost : wing 05:23:59> select elt(NULL,'MySQL','Oracle','Pytho
n');
+-----------------------------------+
| elt(NULL,'MySQL','Oracle','Python') |
+-----------------------------------+
| NULL                              |
+-----------------------------------+
1 row in set (0.00 sec)
```

FORMAT(X,D)

数值的输出结果格式,X类似于'#,###,###.##',D为小数点输出的精度。

```
root@localhost : wing 05:37:31> select format(837485938.3,4);
+----------------------+
| format(837485938.3,4) |
+----------------------+
| 837,485,938.3000     |
+----------------------+
1 row in set (0.00 sec)

root@localhost : wing 05:37:48> select format(837485938.3);
ERROR 1064 (42000): You have an error in your SQL syntax; check the man
ual that corresponds to your MySQL server version for the right syntax
to use near ')' at line 1
root@localhost : wing 05:37:54> select format(837485938.3,0);
+----------------------+
| format(837485938.3,0) |
+----------------------+
| 837,485,938          |
+----------------------+
1 row in set (0.00 sec)
```

INSERT(str,x,y,instr)

将字符串str从第x位置开始，y个字符长的子串替换为instr

```sql
mysql> select * from t;
+------------+
| vc |
+------------+
| vcvcvcvcvc |
| wingwing |
+------------+
2 rows in set (0.00 sec)

mysql> select insert(vc,3,5,'fianna') from t;
+------------------------+
| insert(vc,3,5,'fianna') |
+------------------------+
| vcfiannacvc |
| wifiannag |
+------------------------+
2 rows in set (0.04 sec)
```

LCASE() && LOWER()

LCASE()与LOWER()功能相同,将字符串所有字母小写

```
root@localhost : wing 05:37:58> select lower('MySQL');
+----------------+
| lower('MySQL') |
+----------------+
| mysql          |
+----------------+
1 row in set (0.00 sec)
```

LEFT(str,len) && RIGHT(str,len)

分别为返回从左边开始返回str第len个字母,从右边开始返回str第len个字母

```
sql
root@localhost : wing 05:41:08> select left('MySQL',1),RIGHT('MySQL',1);
+-----------------+------------------+
| left('MySQL',1) | RIGHT('MySQL',1) |
+-----------------+------------------+
| M | L |
+-----------------+------------------+
1 row in set (0.00 sec)
```

LOCATE(substr,str),LOCATE(substr,str,pos)

LOCATE(substr,str)返回substr在str中第一次出现的位置

LOCATE(substr,str,pos)返回sunstr在str中pos位置后第一次出现的位置

类似函数有POSITION()

```sql
root@localhost : wing 05:55:13> select locate('sql','MySQLSQL');
+--------------------------+
| locate('sql','MySQLSQL') |
+--------------------------+
| 3 |
+--------------------------+
1 row in set (0.00 sec)

root@localhost : wing 05:55:18> select locate('SQL','MySQLSQL');
+--------------------------+
| locate('SQL','MySQLSQL') |
+--------------------------+
| 3 |
+--------------------------+
1 row in set (0.00 sec)

root@localhost : wing 05:55:26> select locate('SQL','MySQLSQL',4);
+----------------------------+
| locate('SQL','MySQLSQL',4) |
+----------------------------+
| 6 |
+----------------------------+
1 row in set (0.00 sec)
```

LPAD(str,n,pad)

用字符串pad对str最左边进行填充，直到长度为n个字符长度

```sql
mysql> select * from t;
+------+
| vc |
+------+
| vc |
| wing |
+------+
2 rows in set (0.01 sec)

mysql> select lpad(vc,5,'pad') from t;
+------------------+
| lpad(vc,5,'pad') |
+------------------+
| padvc |
| pwing |
+------------------+
2 rows in set (0.00 sec)
```

RPAD(str,n,pad)

用字符串pad对str最右边进行填充，直到长度为n个字符长度

```sql
mysql> select * from t;
+------+
| vc |
+------+
| vc |
| wing |
+------+
2 rows in set (0.01 sec)

mysql> select rpad(vc,5,'mysql') from t;
+--------------------+
| rpad(vc,5,'mysql') |
```

+------------------+
| vcmys |
| wingm |
+------------------+
2 rows in set (0.00 sec)

```
##### LTRIM() && RTRIM() && TRIM()
LTRIM()返回删除左边空格的字符串
RTRIM()返回删除右边空格的字符串
TRIM()返回删除左右空格的字符串
```sql
root@localhost : wing 06:12:24> select ltrim('   MySQL   ');
+----------------------+
| ltrim('   MySQL   ') |
+----------------------+
| MySQL                |
+----------------------+
1 row in set (0.00 sec)

root@localhost : wing 06:12:31> select rtrim('   MySQL   ');
+----------------------+
| rtrim('   MySQL   ') |
+----------------------+
|    MySQL             |
+----------------------+
1 row in set (0.00 sec)

root@localhost : wing 06:12:34> select trim('   MySQL   ');
+--------------------+
| trim('   MySQL   ') |
+--------------------+
| MySQL              |
+--------------------+
1 row in set (0.00 sec)
```

repeat(str,count)

重复count次str

```sql
root@localhost : wing 06:04:49> select repeat('MySQL',2);
+------------------+
| repeat('MySQL',2) |
+------------------+
| MySQLMySQL |
```
```

+------------------+
1 row in set (0.00 sec)

root@localhost : wing 06:04:56> select repeat(null,2);
+----------------+
| repeat(null,2) |
+----------------+
| NULL |
+----------------+
1 row in set (0.00 sec
```

**replace(str,from_str,to_str)**

将str中的from_str替换为to_str

```
root@localhost : wing 06:07:13> Select replace('www.mysql.com','mysql',
'MySQL');
+-----------------------------------------+
| replace('www.mysql.com','mysql','MySQL') |
+-----------------------------------------+
| www.MySQL.com                           |
+-----------------------------------------+
1 row in set (0.00 sec)
```

reverse(str)

将str逆序输出

```
root@localhost : wing 06:08:26> select reverse('MySQL');
+------------------+
| reverse('MySQL') |
+------------------+
| LQSyM            |
+------------------+
1 row in set (0.00 sec)
```

SPACE(n)

返回n个空格

```
root@localhost : wing 06:12:38> select space(6);
+----------+
```
```

```
| space(6) |
+----------+
|          |
+----------+
1 row in set (0.00 sec)

root@localhost : wing 06:15:52> select space(100);
+------------------------------------------------------------------------------------------------------------------------------------+
| space(100)                                                                                                                         |
+------------------------------------------------------------------------------------------------------------------------------------+
|                                                                                                                                    |
+------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

SUBSTRING(str,m,n)

返回从字符串str的m位置起的n个字符长度的字符串

```sql
mysql> select * from t;
+------------+
| vc |
+------------+
| vcvcvcvcvc |
| wingwing |
+------------+
2 rows in set (0.00 sec)

mysql> select substring(vc,3,5) from t;
+------------------+
| substring(vc,3,5) |
+------------------+
| vcvcv |
| ngwin |
+------------------+
2 rows in set (0.00 sec)
```

UCASE() && UPPER()

将str大写

```
root@localhost : wing 06:16:11> select UCASE('python'),UPPER('python');
+-----------------+-----------------+
| UCASE('python') | UPPER('python') |
+-----------------+-----------------+
| PYTHON          | PYTHON          |
+-----------------+-----------------+
1 row in set (0.00 sec)
```

数值函数

CEIL() & CEILING()

两者功能相同,返回比给出的参数大的最小的整数。

```
root@localhost : wing 04:18:06> select ceil(9.99);
+------------+
| ceil(9.99) |
+------------+
|         10 |
+------------+
1 row in set (0.00 sec)

root@localhost : wing 04:18:20> select ceiling(9.99);
+---------------+
| ceiling(9.99) |
+---------------+
|            10 |
+---------------+
1 row in set (0.00 sec)

root@localhost : wing 04:18:24> select ceil(null);
+------------+
| ceil(null) |
+------------+
|       NULL |
+------------+
1 row in set (0.00 sec)

root@localhost : wing 04:18:29> select ceiling(null);
+---------------+
| ceiling(null) |
+---------------+
|          NULL |
+---------------+
1 row in set (0.00 sec)
```

```
root@localhost : wing 04:18:32> select ceil(0);
+---------+
| ceil(0) |
+---------+
|       0 |
+---------+
1 row in set (0.00 sec)

root@localhost : wing 04:18:36> select ceiling(-9.99);
+-----------------+
| ceiling(-9.99) |
+-----------------+
|              -9 |
+-----------------+
1 row in set (0.00 sec)
```

CRC32()

用于循环冗余校验,返回一个32位无符号数值,是对一个传送的数据块进行校验,是一种高效的差错控制方法。

```
root@localhost : wing 04:21:43> select crc32('MySQL');
+-----------------+
| crc32('MySQL') |
+-----------------+
|     3259397556 |
+-----------------+
1 row in set (0.00 sec)

root@localhost : wing 04:23:41> select crc32('mysql');
+-----------------+
| crc32('mysql') |
+-----------------+
|     2501908538 |
+-----------------+
1 row in set (0.00 sec)
```

FLOOR()

返回比参数小的最大的整数。

```
root@localhost : wing 04:41:32> select floor(1.23);
+-------------+
| floor(1.23) |
+-------------+
|           1 |
```

```
+-------------+
1 row in set (0.00 sec)

root@localhost : wing 04:41:39> select floor(0);
+----------+
| floor(0) |
+----------+
|        0 |
+----------+
1 row in set (0.00 sec)

root@localhost : wing 04:41:42> select floor(null);
+-------------+
| floor(null) |
+-------------+
|        NULL |
+-------------+
1 row in set (0.00 sec)

root@localhost : wing 04:41:47> select floor(-1.23);
+--------------+
| floor(-1.23) |
+--------------+
|           -2 |
+--------------+
1 row in set (0.00 sec)
```

POW(x,y) && POWER(x,y)

两者功能相同,求x的y次方

```
root@localhost : wing 04:49:54> select power(2,4);
+------------+
| power(2,4) |
+------------+
|         16 |
+------------+
1 row in set (0.00 sec)
```

RAND() && RAND(N)

1. 不论是同一个事务中还是非同一个事务中,包含RAND()函数的SQL语句之前都会记录两个会话级的参数RAND_SEED1和RAND_SEED2,由这两个参数根据RAND()产生随机数的算法便可得到一个确定的数值,所以即使在binlog_format=STATEMENT模式下,主从复制之间使用RAND()函数也可以确保数据一致;
2. RAND()函数为产生随机数在[0,1)之间;

3. 对于RAND(N),参数N用作种子值,种子值即binlog日志中RAND_SEED1和RAND_SEED2,产生相同的随机数;

```
root@localhost : wing 05:08:51> select * from rand_data;
+------+
| id   |
+------+
|    1 |
|    2 |
|    3 |
+------+
3 rows in set (0.00 sec)

# 在如下示例中可观察到,rand()的函数产生的随机数随机改变,rand(n)的函数产生的随机数是确定值
root@localhost : wing 05:08:58> select id , rand() from rand_data;
+------+---------------------+
| id   | rand()              |
+------+---------------------+
|    1 | 0.07249362959758716 |
|    2 |  0.3962498189846518 |
|    3 |  0.7637682368641423 |
+------+---------------------+
3 rows in set (0.00 sec)

root@localhost : wing 05:09:08> select id , rand(1) from rand_data;
+------+---------------------+
| id   | rand(1)             |
+------+---------------------+
|    1 | 0.40540353712197724 |
|    2 |  0.8716141803857071 |
|    3 |  0.1418603212962489 |
+------+---------------------+
3 rows in set (0.00 sec)

root@localhost : wing 05:09:14> select id , rand() from rand_data;
+------+--------------------+
| id   | rand()             |
+------+--------------------+
|    1 | 0.6300917581004014 |
|    2 | 0.8591541106432249 |
|    3 | 0.4054953096485653 |
+------+--------------------+
3 rows in set (0.00 sec)

root@localhost : wing 05:09:21> select id , rand(1) from rand_data;
+------+---------------------+
| id   | rand(1)             |
+------+---------------------+
|    1 | 0.40540353712197724 |
|    2 |  0.8716141803857071 |
```

```
|    3 |  0.1418603212962489 |
+------+---------------------+
3 rows in set (0.00 sec)
```

1. 对于想获得在指定范围内的随机数,可以通过不同的函数构造得到。

```
# 获得7到12之间的随机整数
root@localhost : wing 05:43:44> select floor(7+rand()*5);
+------------------+
| floor(7+rand()*5) |
+------------------+
|                 7 |
+------------------+
1 row in set (0.00 sec)

root@localhost : wing 05:43:50> select floor(7+rand()*5);
+------------------+
| floor(7+rand()*5) |
+------------------+
|                 9 |
+------------------+
1 row in set (0.00 sec)
```

ROUND(X),ROUND(X,D)

将参数X四舍五入到指定的对应的D精度的值,如果D没有指定,则默认为0。

```
root@localhost : wing 05:50:05> select round(2.34);
+-------------+
| round(2.34) |
+-------------+
|           2 |
+-------------+
1 row in set (0.00 sec)

root@localhost : wing 05:50:08> select round(2.78);
+-------------+
| round(2.78) |
+-------------+
|           3 |
+-------------+
1 row in set (0.00 sec)

root@localhost : wing 05:50:11> select round(2.34,1);
+---------------+
| round(2.34,1) |
```

```
+---------------+
|           2.3 |
+---------------+
1 row in set (0.00 sec)

root@localhost : wing 05:50:17> select round(2.78,1);
+----------------+
| round(2.78,1) |
+----------------+
|           2.8 |
+----------------+
1 row in set (0.00 sec)

root@localhost : wing 05:50:24> select round(2.5,0);
+---------------+
| round(2.5,0) |
+---------------+
|            3 |
+---------------+
1 row in set (0.00 sec)
```

SIGN()

返回参数的符号,-1代表负数,0代表0,1代表正数。

```
root@localhost : wing 05:52:04> select sign(-123);
+-------------+
| sign(-123) |
+-------------+
|         -1 |
+-------------+
1 row in set (0.00 sec)

root@localhost : wing 05:57:51> select sign(0);
+----------+
| sign(0) |
+----------+
|       0 |
+----------+
1 row in set (0.00 sec)

root@localhost : wing 05:57:54> select sign(123);
+-----------+
| sign(123) |
+-----------+
|         1 |
+-----------+
1 row in set (0.00 sec)
```

SQRT()

平方根函数

```
root@localhost : wing 09:18:43> select sqrt(16);
+----------+
| sqrt(16) |
+----------+
|        4 |
+----------+
1 row in set (0.00 sec)

root@localhost : wing 09:18:49> select sqrt(-16);
+-----------+
| sqrt(-16) |
+-----------+
|      NULL |
+-----------+
1 row in set (0.00 sec)

root@localhost : wing 09:18:51> select sqrt(null);
+------------+
| sqrt(null) |
+------------+
|       NULL |
+------------+
1 row in set (0.00 sec)
```

TRUNCATE(X,D)

对参数X取D精度的值,其后的值将都被截取掉。

```
root@localhost : wing 09:20:48> select truncate(1.7699,2);
+--------------------+
| truncate(1.7699,2) |
+--------------------+
|               1.76 |
+--------------------+
1 row in set (0.00 sec)
```

时间类型函数

ADDDATE()

在某一个日期之后增加一定的时间间隔并输出。

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

```
root@localhost : wing 10:16:38> select adddate(now(),interval 31 day);
+-------------------------------+
| adddate(now(),interval 31 day) |
+-------------------------------+
| 2015-09-27 10:16:44           |
+-------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 10:16:44> select adddate(now(),31);
+---------------------+
| adddate(now(),31)   |
+---------------------+
| 2015-09-27 10:16:50 |
+---------------------+
1 row in set (0.00 sec)
```

ADDTIME(expr1,expr2)

在expr1的基础上加上expr2,形成新的时间值。

```
root@localhost : wing 10:40:21> select now(),addtime(now(),'1 1:1:1');
+---------------------+--------------------------+
| now()               | addtime(now(),'1 1:1:1') |
+---------------------+--------------------------+
| 2015-08-27 10:40:33 | 2015-08-28 11:41:34      |
+---------------------+--------------------------+
1 row in set (0.00 sec)
```

CONVERT_TZ(dt,from_tz,to_tz)

将dt从from_tz的时区转换到to_tz的时区,在该函数中,dt会被认为from_tz时区的时间值。

```
root@localhost : wing 10:47:37> select now(),convert_tz(now(),'+08:00',
'+00:00'),convert_tz(now(),'+02:00','+00:00'),convert_tz(now(),'+02:00'
,'-02:00');
+---------------------+-------------------------------------+----------
--------------------------+-----------------------------------------+
| now()               | convert_tz(now(),'+08:00','+00:00') | convert_t
z(now(),'+02:00','+00:00') | convert_tz(now(),'+02:00','-02:00') |
+---------------------+-------------------------------------+----------
--------------------------+-----------------------------------------+
| 2015-08-27 10:49:44 | 2015-08-27 02:49:44                 | 2015-08-2
7 08:49:44                 | 2015-08-27 06:49:44                 |
```

```
+--------------------+-------------------------------------+----------
-------------------------+-------------------------------------+
```

CURDATE() & CURRENT_DATE & CURRENT_DATE()

返回当前日期

```
root@localhost : wing 10:56:47> select curdate();
+------------+
| curdate()  |
+------------+
| 2015-08-27 |
+------------+
1 row in set (0.00 sec)
```

CURRENT_TIME([fsp]) & CURTIMR()

返回当前时间

```
root@localhost : wing 10:56:51> select current_time(6);
+-----------------+
| current_time(6) |
+-----------------+
| 10:57:23.324593 |
+-----------------+
1 row in set (0.00 sec)
```

CURRENT_TIMESTAMP([fsp]) & CURRENT_TIMESTAMP & NOW()

返回当前日期+时间

```
root@localhost : wing 10:57:23> select current_timestamp(6);
+----------------------------+
| current_timestamp(6)       |
+----------------------------+
| 2015-08-27 10:57:43.637662 |
+----------------------------+
1 row in set (0.00 sec)
```

DATE(expr)

返回expr的日期部分。

```
root@localhost : wing 10:59:53> select date(now());
+-------------+
| date(now()) |
+-------------+
| 2015-08-27  |
+-------------+
1 row in set (0.00 sec)
```

DATEDIFF(expr1,expr2)

返回expr1-expr2得到的天数

```
root@localhost : wing 10:59:58> select datediff(now(),'2015-01-25');
+-----------------------------+
| datediff(now(),'2015-01-25') |
+-----------------------------+
|                         214 |
+-----------------------------+
1 row in set (0.00 sec)
```

DATE_ADD() & DATE_SUB()

在date后增加时间间隔：DATE_ADD(date,INTERVAL expr unit)

在date后减少时间间隔：DATE_SUB(date,INTERVAL expr unit)

```
root@localhost : wing 11:08:07> select date_add('2015-01-25 00:00:00',
interval 5 day),date_sub('2015-01-25 00:00:00',interval 5 day);
+-------------------------------------------------+-------------------
--------------------------+
| date_add('2015-01-25 00:00:00', interval 5 day) | date_sub('2015-01-2
5 00:00:00',interval 5 day) |
+-------------------------------------------------+-------------------
--------------------------+
| 2015-01-30 00:00:00                             | 2015-01-20 00:00:00
                           |
+-------------------------------------------------+-------------------
--------------------------+
1 row in set (0.00 sec)

root@localhost : wing 11:11:32> select date_add('2015-01-25 00:00:00',
interval '5 2' day_hour),date_sub('2015-01-25 00:00:00',interval '5 2'
```

```
day_hour);
+-------------------------------------------------------------+-----------
-----------------------------------------------+
| date_add('2015-01-25 00:00:00', interval '5 2' day_hour) | date_sub('
2015-01-25 00:00:00',interval '5 2' day_hour) |
+-------------------------------------------------------------+-----------
-----------------------------------------------+
| 2015-01-30 02:00:00                                         | 2015-01-19
 22:00:00                                    |
+-------------------------------------------------------------+-----------
-----------------------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 11:11:40> select date_add('2015-01-25 00:00:00',
interval '5 2:2:2' day_second),date_sub('2015-01-25 00:00:00',interval
'5 2:2:2' day_second);
+-------------------------------------------------------------------+-----
-------------------------------------------------------+
| date_add('2015-01-25 00:00:00', interval '5 2:2:2' day_second) | date
_sub('2015-01-25 00:00:00',interval '5 2:2:2' day_second) |
+-------------------------------------------------------------------+-----
-------------------------------------------------------+
| 2015-01-30 02:02:02                                               | 2015
-01-19 21:57:58                                          |
+-------------------------------------------------------------------+-----
-------------------------------------------------------+
1 row in set (0.00 sec)
```

DATE_FORMAT(date,format)

将日期以指定的format返回。

关于format的形式,详情见：http://dev.mysql.com/doc/refman/5.6/en/date-and-time-functions.html#function_date-format

```
root@localhost : wing 11:22:35> select date_format('2015-01-25','%W %M
%Y');
+--------------------------------------+
| date_format('2015-01-25','%W %M %Y') |
+--------------------------------------+
| Sunday January 2015                  |
+--------------------------------------+
1 row in set (0.00 sec)
```

DAYNAME(date)

返回星期几。

```
root@localhost : wing 11:26:16> select dayname('2015-01-25');
+-----------------------+
| dayname('2015-01-25') |
+-----------------------+
| Sunday                |
+-----------------------+
1 row in set (0.00 sec)
```

DAYOFMONTH() & DAYOFWEEK & DAYOFYEAR()

DAYOFMONTH()：返回日期在这个月的第多少天

DAYOFWEEK()：返回日期在这个星期的第多少天,星期天为第一天

DAYOFYEAR()：返回日期在这一年的第多少天

```
root@localhost : wing 11:33:22> select dayofmonth('2015-02-14');
+--------------------------+
| dayofmonth('2015-02-14') |
+--------------------------+
|                       14 |
+--------------------------+
1 row in set (0.00 sec)

root@localhost : wing 11:34:30> select dayofweek('2015-01-25');
+-------------------------+
| dayofweek('2015-01-25') |
+-------------------------+
|                       1 |
+-------------------------+
1 row in set (0.00 sec)

root@localhost : wing 11:35:41> select dayofyear('2015-02-14');
+-------------------------+
| dayofyear('2015-02-14') |
+-------------------------+
|                      45 |
+-------------------------+
1 row in set (0.00 sec)
```

from_days(N)

根据数值N换算出日期,开始日期为：0001-01-01

```
root@localhost : wing 12:01:34> SELECT FROM_DAYS(365);
```

```
+----------------+
| FROM_DAYS(365) |
+----------------+
| 0000-00-00     |
+----------------+
1 row in set (0.00 sec)

# from_days从这个日期开始记起,从N=366开始有效
root@localhost : wing 12:01:36> SELECT FROM_DAYS(366);
+----------------+
| FROM_DAYS(366) |
+----------------+
| 0001-01-01     |
+----------------+
1 row in set (0.00 sec)

root@localhost : wing 12:01:39> SELECT FROM_DAYS(367);
+----------------+
| FROM_DAYS(367) |
+----------------+
| 0001-01-02     |
+----------------+
1 row in set (0.00 sec)

root@localhost : wing 12:02:35> SELECT FROM_DAYS(3670);
+-----------------+
| FROM_DAYS(3670) |
+-----------------+
| 0010-01-18      |
+-----------------+
1 row in set (0.00 sec)
```

from_unixtime()

FROM_UNIXTIME(unix_timestamp)

FROM_UNIXTIME(unix_timestamp,format)

将数值转换为时间

```
root@localhost : wing 12:05:32> select from_unixtime(0);
+---------------------+
| from_unixtime(0)    |
+---------------------+
| 1970-01-01 08:00:00 |
+---------------------+
1 row in set (0.00 sec)
```

```
root@localhost : wing 12:05:37> select from_unixtime(125);
+---------------------+
| from_unixtime(125)  |
+---------------------+
| 1970-01-01 08:02:05 |
+---------------------+
1 row in set (0.00 sec)

root@localhost : wing 12:05:48> select from_unixtime(125,'%W');
+-------------------------+
| from_unixtime(125,'%W') |
+-------------------------+
| Thursday                |
+-------------------------+
1 row in set (0.00 sec)
```

get_format()

以指定的格式输出时间

关于格式请见：http://dev.mysql.com/doc/refman/5.6/en/date-and-time-functions.html#function_get-format

```
root@localhost : wing 01:17:58> SELECT DATE_FORMAT('2015-01-25',GET_FOR
MAT(DATE,'EUR'));
+-------------------------------------------------+
| DATE_FORMAT('2015-01-25',GET_FORMAT(DATE,'EUR')) |
+-------------------------------------------------+
| 25.01.2015                                      |
+-------------------------------------------------+
1 row in set (0.00 sec)

# get_format()函数不可单独使用
root@localhost : wing 01:18:20> select GET_format('2015-01-25','EUR');
ERROR 1064 (42000): You have an error in your SQL syntax; check the man
ual that corresponds to your MySQL server version for the right syntax
to use near ''2015-01-25','EUR')' at line 1

root@localhost : wing 01:19:41>  SELECT DATE_FORMAT('2015-01-25 00:00:0
0',GET_FORMAT(DATE,'EUR'));
+-------------------------------------------------------------+
| DATE_FORMAT('2015-01-25 00:00:00',GET_FORMAT(DATE,'EUR')) |
+-------------------------------------------------------------+
| 25.01.2015                                                  |
+-------------------------------------------------------------+
1 row in set (0.00 sec)
```

HOUR(time)

返回时间的小时数,它可以超过23。

```
root@localhost : wing 01:23:51> select now();
+---------------------+
| now()               |
+---------------------+
| 2015-08-27 13:24:00 |
+---------------------+
1 row in set (0.00 sec)

root@localhost : wing 01:24:00> select hour(now());
+-------------+
| hour(now()) |
+-------------+
|          13 |
+-------------+
1 row in set (0.00 sec)

root@localhost : wing 01:24:27> select hour('356:24:24');
+-------------------+
| hour('356:24:24') |
+-------------------+
|               356 |
+-------------------+
1 row in set (0.00 sec)
```

LAST_DAY(date)

返回date所在月份的最后一天日期。

```
root@localhost : wing 01:48:06> select last_day('2015-01-25');
+------------------------+
| last_day('2015-01-25') |
+------------------------+
| 2015-01-31             |
+------------------------+
1 row in set (0.00 sec)

root@localhost : wing 01:48:23> select last_day('2015-02-02');
+------------------------+
| last_day('2015-02-02') |
+------------------------+
| 2015-02-28             |
+------------------------+
1 row in set (0.00 sec)
```

MAKEDATE() & MAKETIME()

MAKEDATE(year,dayofyear)：year为年份,dayofyear为一年天数

MAKETIME(hour,minute,second)

```
root@localhost : wing 01:48:29> select makedate(2015,35);
+------------------+
| makedate(2015,35) |
+------------------+
| 2015-02-04       |
+------------------+
1 row in set (0.00 sec)

root@localhost : wing 01:52:37> select maketime(13,25,25);
+-------------------+
| maketime(13,25,25) |
+-------------------+
| 13:25:25          |
+-------------------+
1 row in set (0.00 sec)
```

MICROSECOND() & MINUTE() & MONTH() & SECOND() & TIME()

分别为返回时间的微秒,分钟,月份,秒数,时间。

MONTHNAME(date)

返回date的月份值

NOW()

返回语句执行时的日期+时间。

now()在同一个语句中时间是一致的。

```
root@localhost : wing 01:56:49> select now(),sleep(2),now();
+---------------------+----------+---------------------+
| now()               | sleep(2) | now()               |
+---------------------+----------+---------------------+
| 2015-08-27 13:57:00 |        0 | 2015-08-27 13:57:00 |
+---------------------+----------+---------------------+
1 row in set (2.00 sec)
```

SEC_TO_TIME(seconds)

将seconds计算为时分秒返回。

```
root@localhost : wing 01:57:02> select sec_to_time(125);
+------------------+
| sec_to_time(125) |
+------------------+
| 00:02:05         |
+------------------+
1 row in set (0.00 sec)
```

STR_TO_DATE(str,format)

将str以format的形式返回一个时间值。

```
root@localhost : wing 02:01:40> select str_to_date('25,01,2015','%Y,%m,
%d');
+-------------------------------------+
| str_to_date('25,01,2015','%Y,%m,%d') |
+-------------------------------------+
| 2025-01-20                          |
+-------------------------------------+
1 row in set, 1 warning (0.00 sec)

Warning (Code 1292): Truncated incorrect date value: '25,01,2015'
root@localhost : wing 02:01:49> select str_to_date('2015,01,25','%Y,%m,
%d');
+-------------------------------------+
| str_to_date('2015,01,25','%Y,%m,%d') |
+-------------------------------------+
| 2015-01-25                          |
+-------------------------------------+
1 row in set (0.00 sec)

root@localhost : wing 02:02:04> select str_to_date('2015,1,2.5','%Y,%m,
%d');
+-------------------------------------+
| str_to_date('2015,1,2.5','%Y,%m,%d') |
+-------------------------------------+
| 2015-01-02                          |
+-------------------------------------+
1 row in set, 1 warning (0.00 sec)
```

SUBDATE() & SUBTIME()

SUBDATE()：将日期减少时间间隔后,返回时间值

SUBDATE(date,INTERVAL expr unit)

SUBDATE(expr,days)

SUBTIME()：将时间减少时间间隔后,返回时间值

SUBTIME(expr1,expr2)

```
root@localhost : wing 02:08:20> select subdate(now(),5),subdate(now(),i
nterval 5 hour),subtime(now(),'2015-01-25 00:00:00');
+---------------------+---------------------------------+---------------
-----------------------+
| subdate(now(),5)    | subdate(now(),interval 5 hour) | subtime(now(),
'2015-01-25 00:00:00') |
+---------------------+---------------------------------+---------------
-----------------------+
| 2015-08-22 14:08:33 | 2015-08-27 09:08:33            | NULL
                       |
+---------------------+---------------------------------+---------------
-----------------------+
1 row in set (0.00 sec)
```

SYSDATE([fsp])

返回SYSDATE()执行时的时间值。

当使用--sysdate-is-now时,sysdate()与now()一样,否则存在不同,不同之处见举例。

```
# 以下均在未使用--sysdate-is-now时
root@localhost : wing 02:13:49> select now(),sysdate(),sleep(2),now(),s
ysdate();
+---------------------+---------------------+----------+---------------
------+---------------------+
| now()               | sysdate()           | sleep(2) | now()
      | sysdate()           |
+---------------------+---------------------+----------+---------------
------+---------------------+
| 2015-08-27 14:14:44 | 2015-08-27 14:14:44 |        0 | 2015-08-27 14:
14:44 | 2015-08-27 14:14:46 |
+---------------------+---------------------+----------+---------------
------+---------------------+
1 row in set (2.00 sec)

# 如上可见,now()值是相同的,因为now()是指语句开始执行时的时间值,sysdate()是不同的,
因为sysdate()指的是执行sysdate时的时间值。
```

TIMEDIFF(expr1,expr2)

返回expr1-expr2之间的差值时间。

```
root@localhost : wing 02:20:02> select timediff(time(now()),'08:55:00')
;
+----------------------------------+
| timediff(time(now()),'08:55:00') |
+----------------------------------+
| 05:25:18                         |
+----------------------------------+
1 row in set (0.00 sec)
```

TIMESTAMP() & TIMESTAMPADD() & TIMESTAMPDIFF()

TO_DAYS() & TO_SECONDS()

TO_DAYS()：返回日期的天数,开始日期为0000-00-00

TO_SECONDS()：返回日期的秒数,开始日期为0000-00-00

```
root@localhost : wing 02:25:47> select to_days('0000-01-01');
+-----------------------+
| to_days('0000-01-01') |
+-----------------------+
|                     1 |
+-----------------------+
1 row in set (0.00 sec)

root@localhost : wing 02:25:53> select to_days('2015-01-25');
+-----------------------+
| to_days('2015-01-25') |
+-----------------------+
|                735988 |
+-----------------------+
1 row in set (0.00 sec)

root@localhost : wing 02:26:02> select to_seconds('2015-01-25');
+--------------------------+
| to_seconds('2015-01-25') |
+--------------------------+
|              63589363200 |
+--------------------------+
1 row in set (0.00 sec)
```

UNIX_TIMESTAMP([date])

返回unix timestamp值，从'1970-01-01 00:00:00'UTC时区的时间值开始。

```
root@localhost : wing 02:28:22> select unix_timestamp('2015-01-25 00:00
:00');
+---------------------------------------+
| unix_timestamp('2015-01-25 00:00:00') |
+---------------------------------------+
|                            1422115200 |
+---------------------------------------+
1 row in set (0.00 sec)
```

UTC_DATE & UTC_DATE() & UTC_TIME & UTC_TIME() & UTC_TIMESTAMP & UTC_TIMESTAMP()

将日期或时间以UTC时区返回。

```
root@localhost : wing 02:31:31> select utc_date();
+------------+
| utc_date() |
+------------+
| 2015-08-27 |
+------------+
1 row in set (0.00 sec)

root@localhost : wing 02:32:51> select utc_timestamp();
+---------------------+
| utc_timestamp()     |
+---------------------+
| 2015-08-27 06:32:57 |
+---------------------+
1 row in set (0.00 sec)

root@localhost : wing 02:32:57> select now(),utc_timestamp();
+---------------------+---------------------+
| now()               | utc_timestamp()     |
+---------------------+---------------------+
| 2015-08-27 14:33:10 | 2015-08-27 06:33:10 |
+---------------------+---------------------+
1 row in set (0.00 sec)

root@localhost : wing 02:33:10> select utc_timestamp;
+---------------------+
| utc_timestamp       |
+---------------------+
| 2015-08-27 06:34:10 |
+---------------------+
1 row in set (0.00 sec)

root@localhost : wing 02:34:10> select time(now()),utc_time;
+-------------+----------+
```

```
| time(now()) | utc_time |
+-------------+----------+
| 14:35:58    | 06:35:58 |
+-------------+----------+
1 row in set (0.00 sec)
```

WEEK(date[,mode])

返回date在这一年的第几周。

关于mode,详见：http://dev.mysql.com/doc/refman/5.6/en/date-and-time-functions.html#function_week

```
root@localhost : wing 02:35:58> select week(date(now()));
+------------------+
| week(date(now())) |
+------------------+
|               34 |
+------------------+
1 row in set (0.00 sec)
```

WEEKDAY(date)

查看date是星期几。(0表示星期一，1表示星期二....)

```
root@localhost : wing 02:38:15> select weekday(date(now()));
+--------------------+
| weekday(date(now())) |
+--------------------+
|                  3 |
+--------------------+
1 row in set (0.00 sec)
```