MySQL操作符和常用函数

# 操作符

## 操作符的优先级

|  |
| --- |
| INTERVARL  BINARY COLLATE  !  -(负) ~(按位取反)  ^  \* / DIV % MOD  + -  << >>  &  |  < <= = <=> <> != > IN IS LIKE REGEXP RLIKE  BETWEEN CASE WHEN THEN ELSE  NOT  AND &&  XOR  OR ||  := |

注意：SQL模式为PIPES\_AS\_CONCAT则||变成一个字符串连接操作符，NOT的优先级比！低，如果想让NOT和！一样高使用SQL模式HIGT\_NOT\_PRECEDENCE

## 分组操作符

([expr],[expr])：可以改变操作符顺序。

## 算数操作符

算数运算符的规则：

* 在数字上下文里面出现的字符串会被转化为双精度数
* 如果两个操作符都是整数，那么+，-，\*运算符使用64为整数运算。如果其中至少有一个无符号数，那么结果为无符号数。如果其中至少有一个实数，那么+，-，、，\*，%的精度由精度最大的那个操作符决定。
* 使用/对两个精度值进行除法运算的时候，所得结果小数位数等于除数的小数位加上系统变量div\_precision\_increment的值（默认4）

+（加），-（减），\*（乘），/（除），DIV （整除），%或者mod（取余）

## 比较操作符

比较操作符操作结果取值为真，返回1；比较操作符操作结果取值为假，返回0。比较操作符的规则：

* 除了<=>以外，所有涉及NULL值的比较结果都为NULL。NULL<=>NULL的结果为真。
* 字符串的比较。对于二进制串，按字节逐个比较对应字节的数字。对于非二进制串，按字符串字符集对应的排序规则进行比较。对于二进制串和非二进制串按二进制进行比较
* 如果两个操作符都是整数，那么他们将按整数方式进行数字比较。
* 如果十六进制常量是与数字进行比较，那么会被视为二进制串。
* 除IN外，如果一个操作符是TIMESTAMP或者DATETIME，另一个为常量，那么这些操作数会被当做TIMESTAMP值进行比较。
* 如果一个操作数为小数，而另一个操作符为小数或者整数，那么按照小数进行比较，否则按照双精度进行比较
* 除上述情况，比较操作符都将被视为双精度浮点数。字符串和数字进行比较，字符串会被转化为双精度数（如果字符串看起来不像数字，那么结果转化为0）

=，<=>,<>或！=，<,<=,>,>=

expr BETWEEN min AND max(包含min,max)

CASE [expr] WHEN expr1 THEN result1 … [ELSE default] END

expr [NOT] IN (value1,value2,…)

expr IS {FLASE|TRUE|UNKNOWN} 0为假，null为未知，非0非null为真

expr IS [NOT] NULL

|  |
| --- |
| SELECT CASE "中文" REGEXP "[u0391-uFFE5]" WHEN 1 THEN "不是中文字符" ELSE "是中文字符" END AS "判断是否是中文字符" |

## 位操作符

位操作必须用BITINT值（64位整数）

&，|，^，<<，>>，~

## 逻辑操作符

逻辑操作符用来测试表达式是否成立。NOT(!),AND(&&),OR(||),XOR(异或)

注意：||不像标准SQL表示字符串连接，可以使用SQL模式PIPES\_AS\_CONCAT来启用。

## 类型转换操作符

\_charset str：使用指定字符集来解释字符

BINARY str：把str转化为一个二进制串。

str COLLATE collation：str拥有给定的排序规则

## 模式匹配操作符

### LIKE的SQL模式匹配

格式：str [NOT] LIKE pattern [ESCAPE 'c' ]

* %(任意字符出现任意次(0到n))
* \_(匹配单个字符)

注意：like允许对数值表达式进行运算，escape重新定义转义字符

### REGEXP的正则表达式匹配

格式：str [NOT] REGEXP pattern

. ：表示任意字符（不包含回车）

| ：为正则表达式or匹配，匹配其中的一个

[…]：匹配几个字符中的一个字符，其中可以使用^表示非的意思

- ：范围匹配，[1-9]

匹配特殊字符：

\\f(换页) \\n(换行) \\r(回车) \\t(制表)

\\v(纵向制表) \\\(\) 以\\开始，如\\(,\\),\\.等

匹配字符类：

* [:alnum:] 任意字母和数字 同[a-zA-Z0-9]
* [:alpha:] 任意字符[a-zA-Z]
* [:lower:] 任意小写字母[a-z]
* [:upper:] 任意大写字母[A-Z]
* [:digit:] 任意数字[0-9]
* [:xdigit:] 任意十六进制数字[a-fA-F0-9]
* [:blank:] 空格和制表 [ \\t]
* [:space:] 包括空格在内的任意空白字符
* [:cntrl:] ASCII控制字符0-31和127
* [:graph:] 除空格外的所有打印字符
* [:print:] 任意可打印字符
* [:punct:] 不在[:alnum:]和[:cntrl:]的所有字符

匹配多个实例：

* \*：(0个或者多个)
* +：(1个或者多个)
* ?：(0个或者一个)
* {n}：指定数目匹配
* {n, }：不少于指定数目的匹配
* {n,m}：匹配数目的范围（m不超过255）

定位符：

* ^：文本开始
* $：文本结束
* [[:<:]]：单词开始
* [[:>:]]：单词结尾

# 常用函数的使用

## 数值函数

BIN(x)：返回x的二进制（OCT返回八进制，HEX返回十六进制）

CEILING(x)：返回大于x的最小整数值

FLOOR(x)：返回小于x的最大整数值

RAND()：返回０到１内的随机值,通过提供一个参数(种子)RAND()随机数生成器生成一个指定的值。

ROUND(x,y)：返回参数x的四舍五入的有y位小数的值

TRUNCATE(x,y)：返回数字x截短为y位小数的结果

|  |
| --- |
| 示例代码如下:  **select** **ceiling**(1.2),**floor**(1.2),**round**(1.2,2),**rand**();  2;1;1.20;0.4503040438986421 |

## 聚合函数

(常用于GROUP BY从句的SELECT查询中)

AVG(col)：返回指定列的平均值

COUNT(col)：返回指定列中非NULL值的个数

MIN(col)：返回指定列的最小值

MAX(col)：返回指定列的最大值

SUM(col)：返回指定列的所有值之和

GROUP\_CONCAT(col)： 返回由属于一组的列值连接组合而成的结果

## 字符串函数

ASCII(char)：返回字符的ASCII码值。

CONCAT(s1,s2...,sn)：将s1,s2...,sn连接成字符串。CONCAT\_WS(sep,s1,s2...,sn)：将s1,s2...,sn连接成字符串，并用sep字符间隔。

LCASE(str)或LOWER(str) ：返回字符串str中所有字符为小写后的结果。

UCASE(str)或UPPER(str) ：返回字符串str中所有字符为大写后 的结果。

REVERSE(str) ：返回字符串str倒叙的结果。

LEFT(str,len)：返回字符串str中最左边的len个字符。

RIGHT(str,len)： 返回字符串str中最右边的len个字符。

SUBSTRING(str,position,length)：返回str的从position开始length个字符的子字符串。位置从1开始。

SUBSTRING\_INDEX(str,delim,n)：如果n为正值，则str从左到右的顺序找到字符串delim的第n次出现，并返回该位置左侧的全部内容。如果n为负值，则str从右到左的顺序找到字符串delim的第n次出现，并返回该位置右侧的全部内容。如果没有找到delim则返回str。

INSERT(str,pos,len,instr) ：将字符串str从第pos位置开始，len个字符长度的子串替换为字符串instr，返回结果。（位置从1开始。）

REPEAT(str,srchstr,rplcstr)：返回字符串str重复x次的结果。

LENGTH(s)：返回字符串str中的字节数。

CHAR\_LENGTH：返回字符串字符个数

LTRIM(str) ：从字符串str中切掉开头的空格（只是空格）。

RTRIM(str) ：返回字符串str尾部的空格。

TRIM([{LEADING | TRAILING | BOTH } [trim\_str] from str)：trim\_str默认是空格，BOTH是也是默认选项。LEADING首部去除，TRAILING尾部清除

FORMAT(x,d[,locale]) ：使用一个类似'nn,nnn,nnn',把数值x格式化为小数点后带n位小数的数字，返回的值是字符串。

|  |  |
| --- | --- |
| 示例代码如下:  **select** **insert**('hello',2,3,'中国人')  **select** **substr**('h中国人',1,3)  **select** **substring\_index**('zhongguo','gg',-1) |  |

## 日期和时间函数

CURDATE()或CURRENT\_DATE() ：返回当前的日期。

CURTIME([fsp])或CURRENT\_TIME() ：返回当前的时间。

NOW() ：返回当前的日期和时间。

DATE\_ADD(datetime,INTERVAL int interval)：返回日期date加上间隔时间int的结果(int必须按照关键字进行格式化),如：

DATE\_SUB(datetime,INTERVAL int interval)：返回日期date加上间隔时间int的结果(int必须按照关键字进行格式化),

DATE\_DIFF(expr1,expr2)：两个值都应该是日期值或者日期时间值。返回前一个值减去后一个值的天数，时间部分会被忽略。

EXTRACT(interval FROM datetime)

|  |
| --- |
| Interval常见值有：   * MICROSECOND（微秒） * SECOND（秒） * MINUTE（分钟） * HOUR（小时） * DAY（日） * WEEK（周） * MONTH（月） * YEAR（年）   注意：也可以组合使用（second,minute,hour,day）(year,month) |

DATE\_FORMAT(date,fmt) ：依照指定的fmt格式格式化日期date值。

根据format字符串格式化date值:

*  %S, %s ：两位数字形式的秒（ 00,01, ..., 59）
*  %I, %i ：两位数字形式的分（ 00,01, ..., 59）
*  %H :两位数字形式的小时，24 小时（00,01, ..., 23）
*  %h :两位数字形式的小时，12 小时（01,02, ..., 12）
*  %k :数字形式的小时，24 小时（0,1, ..., 23）
*  %l :数字形式的小时，12 小时（1, 2, ..., 12）
*  %T :24 小时的时间形式（hh:mm:ss）
*  %r :12 小时的时间形式（hh:mm:ss AM 或hh:mm:ss PM）
*  %p :AM或PM
*  %W :一周中每一天的名称（Sunday, Monday, ..., Saturday）
*  %a :一周中每一天名称的缩写（Sun, Mon, ..., Sat）
*  %d :两位数字表示月中的天数（00, 01,..., 31）
*  %e :数字形式表示月中的天数（1, 2， ..., 31）
*  %D :英文后缀表示月中的天数（1st, 2nd, 3rd,...）
*  %w :以数字形式表示周中的天数（ 0 = Sunday, 1=Monday, ..., 6=Saturday）
*  %j :以三位数字表示年中的天数（ 001, 002, ..., 366）
*  %U :周（0, 1, 52），其中Sunday 为周中的第一天
*  %u :周（0, 1, 52），其中Monday 为周中的第一天
*  %M :月名（January, February, ..., December）
*  %b :缩写的月名（ January, February,...., December）
*  %m :两位数字表示的月份（01, 02, ..., 12）
*  %c :数字表示的月份（1, 2, ...., 12）
*  %Y :四位数字表示的年份
*  %y :两位数字表示的年份
*  %% :直接值“%”

|  |
| --- |
| 示例代码如下:  **select** **curdate**(),**curtime**(),**now**()  **SELECT** **DATE\_FORMAT**(**NOW**(),'%Y-%m-%d %T');  **select** **date\_add**(**now**(),**INTERVAL** -1 **MONTH**);  **SELECT** **EXTRACT**(**YEAR** **FROM** **NOW**());  **SELECT** **EXTRACT**(**MONTH** **FROM** **NOW**());  **SELECT** **EXTRACT**(**DAY** **FROM** **NOW**());  **SELECT** **EXTRACT**(**HOUR** **FROM** **NOW**());  **SELECT** **EXTRACT**(**MINUTE** **FROM** **NOW**());  **SELECT** **EXTRACT**(**SECOND** **FROM** **NOW**()); |

## 加密函数

AES\_ENCRYPT(str,key) ：返回用密钥key对字符串str利用高级加密准算法加密后的结果，调用AES\_ENCRYPT的结果是一个二进制字符串，以BLOB类型存储。

AES\_DECRYPT(str,key) ：返回用密钥key对字符串str利用高级加密标准算法解密后的结果。

ENCODE(str,key) ：使用key作为密钥加密字符串str，调用ENCODE()的结果是一个二进制字符串，它以BLOB类型存储。

DECODE(str,key) ：使用key作为密钥解密加密字符串str。

ENCRYPT(str,[salt]) ：使用UNIXcrypt()函数，用关键词salt(一个可以惟一确定口令的字符串，就像钥匙一样)加密字符串str，不可逆算法。

MD5(str) ：计算字符串str的MD5校验和。

PASSWORD(str) ：返回字符串str的加密版本，这个加密过程是不可 逆转的，和UNIX密码加密过程使用不同的算法。

SHA(str) ：计算字符串str的安全散列算法(SHA)校验和。

|  |
| --- |
| 示例代码如下:  **select** **ENCRYPT**('secret','AB');  ABS5SGh1EL6bk |

## 比较函数

ISNULL (expr)： expr为null，isnull()的返回值为1，否则返回值为0。

IF(expr,t,f) 如果expr非0且非NULL，返回t；否则返回f

IFNULL(expr1,expr2) ：expr1为NULL，返回expr2，否则返回expr1

NULLIF(expr1,expr2)：expr1= expr2返回NULL；否则返回expr1

CASE [expression to be evaluated]

WHEN [val 1] THEN [result 1]

WHEN [val 2] THEN [result 2]

WHEN [val 3] THEN [result 3]

......

WHEN [val n] THEN [result n]

ELSE [default result]

END

第一个参数是要被判断的值或表达式，接下来的是一系列的WHEN- THEN块，每一块的第一个参数指定要比较的值，如果为真，就返回结 果。所有的WHEN-THEN块将以ELSE块结束，当END结束了所有外 部的CASE块时，如果前面的每一个块都不匹配就会返回ELSE块指定 的默认结果。如果没有指定ELSE块，而且所有的WHEN-THEN比较都 不是真，MySQL将会返回NULL。

|  |
| --- |
| 示例代码如下:  **select** **if**(1,'true','false');  **select** **ifnull**(**null**,1);  **select** **isnull**(**null**);  **select** **nullif**(3,4);  **SELECT** **CASE**'green'  **WHEN** 'red' **THEN** 'stop'  **WHEN** 'green' **THEN** 'go'  **END**; |

## ip函数

INET\_ATON(str) ：返回IP地址的数字表示。

INET\_NTOA(num) ：返回数字所代表的IP地址。  
INET6\_ATON(expr) ：以字符串形式给定ipv4/ipv6地址，返回二进制串

INET6\_NTOA(expr) ：给定二进串返回相应ip地址字符串。

IS\_IPV4(expr)：如果参数是有效的iPv4地址字符串则返回1，否则0

IS\_IPV6(expr)：如果参数是有效的iPv6地址字符串则返回1，否则0

|  |
| --- |
| 示例代码如下:  **select** **inet\_aton**('172.16.129.0');  **select** **inet\_ntoa**(2886762752); |

## 类型转化函数

CAST(expr as type)

CONVERT(expr, type) ,CONVERT(expr using charset)

注意：CAST和CONVERT函数类似，前者遵循标准SQL

可以转换的类型是有限制的。这个类型可以是以下值其中的一个：

*  二进制，同带binary前缀的效果 : BINARY
*  字符型，可带参数 : CHAR()
*  日期 : DATE
*  时间: TIME
*  日期时间型 : DATETIME
*  浮点数 : DECIMAL
*  整数 : SIGNED
*  无符号整数 : UNSIGNED

示例代码如下:

|  |
| --- |
| **select** **cast**(20131011 **as** **datetime**)  **select** **cast**(13.2 **as** **decimal**(5,3)) |

## 其他函数

DATABASE() ：返回当前数据库名

BENCHMARK(count,expr) ：将表达式expr重复运行count次

CONNECTION\_ID() ：返回当前客户的连接ID

USER()或SYSTEM\_USER() ：返回当前登陆用户名

FOUND\_ROWS() ：返回最后一个SELECT查询进行检索的总行数

VERSION() ：返回MySQL服务器的版本

UUID()：返回一个全局的唯一标识符

|  |
| --- |
| 示例代码如下:  SELECT DATABASE(),VERSION(),USER();  #该例中,MySQL计算LOG(RAND()\*PI())表达式9999999次。  SELECT BENCHMARK(9999999,LOG(RAND()\*PI())); |

# 强制类型转换

|  |
| --- |
| #转换成数值类型：往表达式中添加+0或者+0.0  0x65 +0  #去掉数值的小数部分：floor()，ceilling()，round()，cast()  cast(13.3 as signed)  #字符转换成十六进制字符串，字符串转换成数字字符串  hex(255) #'FF'  hex('abc') #'616263'  #单字节字符串转化为ASCII码值  ASCII('A') #65  #数字转化为字符char()  char(47) # /  #字符串，数字转化为日期  date\_add(20130101,interval 0 day)  date\_add('20130101',interval 0 day)  #日期转化为数字  curdate()+0  #字符集转化为另一个  convert('20130101' using ucs2)  #二进制串转化为非二进制串  convert（0x616263 using latin1）  #非二进制串转化为二进制串  binary 'abc' |