

# Hive 函数大全

(基于 Hive0.13)

By lxw1234

<http://lxw1234.com>

[lxw 的大数据田地](#)

## 目录

一、关系运算:	6
1. 等值比较: =	6
2. 等值比较: <=>	6
3. 不等值比较: <>和!=	6
4. 小于比较: <	6
5. 小于等于比较: <=	7
6. 大于比较: >	7
7. 大于等于比较: >=	7
8. 区间比较	8
9. 空值判断: IS NULL	8
10. 非空判断: IS NOT NULL	8
10. LIKE 比较: LIKE	8
11. JAVA 的 LIKE 操作: RLIKE	9
12. REGEXP 操作: REGEXP	9
二、数学运算:	10
1. 加法操作: +	10
2. 减法操作: -	10
3. 乘法操作: *	10
4. 除法操作: /	11
5. 取余操作: %	11
6. 位与操作: &	11
7. 位或操作:	12
8. 位异或操作: ^	12
9. 位取反操作: ~	12
三、逻辑运算:	13
1. 逻辑与操作: AND 、 &&	13
2. 逻辑或操作: OR 、	13
3. 逻辑非操作: NOT、!	13
四、复合类型构造函数	14
1. map 结构	14
2. struct 结构	14
3. named_struct 结构	14
4. array 结构	14
5. create_union	15
五、复合类型操作符	15
1. 获取 array 中的元素	15
2. 获取 map 中的元素	15
3. 获取 struct 中的元素	16
六、数值计算函数	16
1. 取整函数: round	16
2. 指定精度取整函数: round	16
3. 向下取整函数: floor	17

4. 向上取整函数: ceil .....	17
5. 向上取整函数: ceiling .....	17
6. 取随机数函数: rand .....	17
7. 自然指数函数: exp .....	18
8. 以 10 为底对数函数: log10 .....	18
9. 以 2 为底对数函数: log2 .....	19
10. 对数函数: log .....	19
11. 幂运算函数: pow .....	19
12. 幂运算函数: power .....	19
13. 开平方函数: sqrt .....	20
14. 二进制函数: bin .....	20
15. 十六进制函数: hex .....	20
16. 反转十六进制函数: unhex .....	20
17. 进制转换函数: conv .....	21
18. 绝对值函数: abs .....	21
19. 正取余函数: pmod .....	21
20. 正弦函数: sin .....	22
21. 反正弦函数: asin .....	22
22. 余弦函数: cos .....	22
23. 反余弦函数: acos .....	22
24. positive 函数: positive .....	23
25. negative 函数: negative .....	23
七、集合操作函数 .....	23
1. map 类型大小: size .....	23
2. array 类型大小: size .....	23
3. 判断元素数组是否包含元素: array_contains .....	24
4. 获取 map 中所有 value 集合 .....	24
5. 获取 map 中所有 key 集合 .....	24
6. 数组排序 .....	24
八、类型转换函数 .....	25
1. 二进制转换: binary .....	25
2. 基础类型之间强制转换: cast .....	25
九、日期函数 .....	25
1. UNIX 时间戳转日期函数: from_unixtime .....	25
2. 获取当前 UNIX 时间戳函数: unix_timestamp .....	26
3. 日期转 UNIX 时间戳函数: unix_timestamp .....	26
4. 指定格式日期转 UNIX 时间戳函数: unix_timestamp .....	26
5. 日期时间转日期函数: to_date .....	26
6. 日期转年函数: year .....	27
7. 日期转月函数: month .....	27
8. 日期转天函数: day .....	27
9. 日期转小时函数: hour .....	28
10. 日期转分钟函数: minute .....	28
11. 日期转秒函数: second .....	28

12. 日期转周函数: weekofyear .....	28
13. 日期比较函数: datediff .....	29
14. 日期增加函数: date_add .....	29
15. 日期减少函数: date_sub .....	29
十、条件函数 .....	29
1. If 函数: if .....	29
2. 非空查找函数: COALESCE .....	30
3. 条件判断函数: CASE .....	30
4. 条件判断函数: CASE .....	30
十一、字符串函数 .....	31
1. 字符 ascii 码函数: ascii .....	31
2. base64 字符串 .....	31
3. 字符串连接函数: concat .....	31
4. 带分隔符字符串连接函数: concat_ws .....	31
5. 数组转换成字符串的函数: concat_ws .....	32
6. 小数位格式化字符串函数: format_number .....	32
7. 字符串截取函数: substr,substring .....	32
8. 字符串截取函数: substr,substring .....	33
9. 字符串查找函数: instr .....	33
10. 字符串长度函数: length .....	33
11. 字符串查找函数: locate .....	33
12. 字符串格式化函数: printf .....	34
13. 字符串转换成 map 函数: str_to_map .....	34
14. base64 解码函数: unbase64(string str) .....	34
15. 字符串转大写函数: upper,ucase .....	35
16. 字符串转小写函数: lower,lcase .....	35
17. 去空格函数: trim .....	35
18. 左边去空格函数: ltrim .....	36
19. 右边去空格函数: rtrim .....	36
20. 正则表达式替换函数: regexp_replace .....	36
21. 正则表达式解析函数: regexp_extract .....	36
22. URL 解析函数: parse_url .....	37
23. json 解析函数: get_json_object .....	37
24. 空格字符串函数: space .....	38
25. 重复字符串函数: repeat .....	38
26. 左补足函数: lpad .....	38
27. 右补足函数: rpad .....	39
28. 分割字符串函数: split .....	39
29. 集合查找函数: find_in_set .....	39
30. 分词函数: sentences .....	39
31. 分词后统计一起出现频次最高的 TOP-K .....	40
32. 分词后统计与指定单词一起出现频次最高的 TOP-K .....	40
十二、混合函数 .....	41
1. 调用 Java 函数: java_method .....	41

---

2. 调用 Java 函数: reflect .....	41
3. 字符串的 hash 值: hash .....	41
十三、XPath 解析 XML 函数 .....	42
1. xpath.....	42
2. xpath_string .....	42
3. xpath_boolean .....	42
4. xpath_short, xpath_int, xpath_long .....	43
5. xpath_float, xpath_double, xpath_number .....	43
十四、汇总统计函数 (UDAF) .....	44
1. 个数统计函数: count .....	44
2. 总和统计函数: sum .....	44
3. 平均值统计函数: avg .....	45
4. 最小值统计函数: min.....	45
5. 最大值统计函数: max.....	45
6. 非空集合总体变量函数: var_pop .....	45
7. 非空集合样本变量函数: var_samp .....	46
8. 总体标准偏离函数: stddev_pop.....	46
9. 样本标准偏离函数: stddev_samp .....	46
10. 中位数函数: percentile .....	46
11. 中位数函数: percentile .....	46
12. 近似中位数函数: percentile_approx .....	47
13. 近似中位数函数: percentile_approx .....	47
14. 直方图: histogram_numeric .....	47
15. 集合去重数: collect_set .....	47
16. 集合不去重函数: collect_list .....	48
十五、表格生成函数 Table-Generating Functions (UDTF) .....	48
1. 数组拆分成多行: explode.....	48
2. Map 拆分成多行: explode.....	49

## 一、关系运算：

### 1. 等值比较: =

语法: A=B

操作类型: 所有基本类型

描述: 如果表达式 A 与表达式 B 相等, 则为 TRUE; 否则为 FALSE; 只要有任意比较项为 NULL, 均返回 FALSE;

举例:

```
hive> select 1 from lxw1234 where 1=1;
```

```
1
```

```
hive> select 1 from lxw1234 where NULL = NULL;
```

```
OK
```

### 2. 等值比较:<=>

语法: A <=> B

操作类型: 所有基本类型

描述: 如果 A 和 B 都是非 NULL 值, 则返回结果和=一样, 如果两者都为 NULL, 返回 TRUE, 如果有一个为 NULL, 则返回 FALSE。

举例:

```
hive> select 1 from lxw1234 where NULL <=> NULL;
```

```
OK
```

```
1
```

### 3. 不等值比较: <>和!=

语法: A <> B    A != B

操作类型: 所有基本类型

描述: 如果表达式 A 为 NULL, 或者表达式 B 为 NULL, 返回 NULL; 如果表达式 A 与表达式 B 不相等, 则为 TRUE; 否则为 FALSE

举例:

```
hive> select 1 from lxw1234 where 1 <> 2;
```

```
1
```

### 4. 小于比较: <

语法: A < B

操作类型: 所有基本类型

描述: 如果表达式 A 为 NULL, 或者表达式 B 为 NULL, 返回 NULL; 如果表达式 A 小于

表达式 B，则为 TRUE；否则为 FALSE

举例：

```
hive> select 1 from lxw1234 where 1 < 2;  
1
```

## 5. 小于等于比较: <=

语法: A <= B

操作类型: 所有基本类型

描述: 如果表达式 A 为 NULL，或者表达式 B 为 NULL，返回 NULL；如果表达式 A 小于或者等于表达式 B，则为 TRUE；否则为 FALSE

举例：

```
hive> select 1 from lxw1234 where 1 <= 1;  
1
```

## 6. 大于比较: >

语法: A > B

操作类型: 所有基本类型

描述: 如果表达式 A 为 NULL，或者表达式 B 为 NULL，返回 NULL；如果表达式 A 大于表达式 B，则为 TRUE；否则为 FALSE

举例：

```
hive> select 1 from lxw1234 where 2 > 1;  
1
```

版权所有: <http://lxw1234.com>

## 7. 大于等于比较: >=

语法: A >= B

操作类型: 所有基本类型

描述: 如果表达式 A 为 NULL，或者表达式 B 为 NULL，返回 NULL；如果表达式 A 大于或者等于表达式 B，则为 TRUE；否则为 FALSE

举例：

```
hive> select 1 from lxw1234 where 1 >= 1;  
1
```

**注意: String 的比较要注意(常用的时间比较可以先 to\_date 之后再比较)**

```
hive> select * from lxw1234;  
OK
```

---

```
2011111209 00:00:00    2011111209

hive> select a,b,a<b,a>b,a=b from lxw1234;
2011111209 00:00:00    2011111209    false    true    false
```

## 8. 区间比较

语法: A [NOT] BETWEEN B AND C

操作类型: 所有类型

描述: 如果 A、B、C 有任一个为 NULL,则返回 FALSE. 等价于  $B \leq A < C$ .

举例:

```
hive> select 1 from lxw1234 where 1 between 1 and 2;
```

OK

1

## 9. 空值判断: IS NULL

语法: A IS NULL

操作类型: 所有类型

描述: 如果表达式 A 的值为 NULL, 则为 TRUE; 否则为 FALSE

举例:

```
hive> select 1 from lxw1234 where null is null;
```

1

版权所有: <http://lxw1234.com>

## 10. 非空判断: IS NOT NULL

语法: A IS NOT NULL

操作类型: 所有类型

描述: 如果表达式 A 的值为 NULL, 则为 FALSE; 否则为 TRUE

举例:

```
hive> select 1 from lxw1234 where 1 is not null;
```

1

## 10. LIKE 比较: LIKE

语法: A LIKE B

操作类型: strings



描述: 如果字符串 A 或者字符串 B 为 NULL, 则返回 NULL; 如果字符串 A 符合表达式 B 的正则语法, 则为 TRUE; 否则为 FALSE。B 中字符”\_”表示任意单个字符, 而字符”%”表示任意数量的字符。

举例:

```
hive> select 1 from lxw1234 where 'football' like 'foot%';
```

```
1
```

```
hive> select 1 from lxw1234 where 'football' like 'foot_____';
```

```
1
```

**注意: 否定比较时候用 NOT A LIKE B**

```
hive> select 1 from lxw1234 where NOT 'football' like 'fff%';
```

```
1
```

## 11. JAVA 的 LIKE 操作: RLIKE

语法: A RLIKE B

操作类型: strings

描述: 如果字符串 A 或者字符串 B 为 NULL, 则返回 NULL; 如果字符串 A 符合 JAVA 正则表达式 B 的正则语法, 则为 TRUE; 否则为 FALSE。

举例:

```
hive> select 1 from lxw1234 where 'footbar' rlike '^f.*r$';
```

```
1
```

**注意: 判断一个字符串是否全为数字:**

```
hive>select 1 from lxw1234 where '123456' rlike '^\\d+$';
```

```
1
```

```
hive> select 1 from lxw1234 where '123456aa' rlike '^\\d+$';
```

## 12. REGEXP 操作: REGEXP

语法: A REGEXP B

操作类型: strings

描述: 功能与 RLIKE 相同

举例:

```
hive> select 1 from lxw1234 where 'footbar' REGEXP '^f.*r$';
```

```
1
```

版权所有: <http://lxw1234.com>

## 二、数学运算：

### 1. 加法操作：+

语法:  $A + B$

操作类型：所有数值类型

说明：返回  $A$  与  $B$  相加的结果。结果的数值类型等于  $A$  的类型和  $B$  的类型的最小父类型（详见数据类型的继承关系）。比如， $\text{int} + \text{int}$  一般结果为  $\text{int}$  类型，而  $\text{int} + \text{double}$  一般结果为  $\text{double}$  类型

举例：

```
hive> select 1 + 9 from lxw1234;
```

```
10
```

```
hive> create table lxw1234 as select 1 + 1.2 from lxw1234;
```

```
hive> describe lxw1234;
```

```
_c0      double
```

### 2. 减法操作：-

语法:  $A - B$

操作类型：所有数值类型

说明：返回  $A$  与  $B$  相减的结果。结果的数值类型等于  $A$  的类型和  $B$  的类型的最小父类型（详见数据类型的继承关系）。比如， $\text{int} - \text{int}$  一般结果为  $\text{int}$  类型，而  $\text{int} - \text{double}$  一般结果为  $\text{double}$  类型

举例：

```
hive> select 10 - 5 from lxw1234;
```

```
5
```

```
hive> create table lxw1234 as select 5.6 - 4 from lxw1234;
```

```
hive> describe lxw1234;
```

```
_c0      double
```

### 3. 乘法操作：\*

语法:  $A * B$

操作类型：所有数值类型

说明：返回  $A$  与  $B$  相乘的结果。结果的数值类型等于  $A$  的类型和  $B$  的类型的最小父类型（详见数据类型的继承关系）。注意，如果  $A$  乘以  $B$  的结果超过默认结果类型的数值范围，则需要通过 `cast` 将结果转换成范围更大的数值类型

举例：

```
hive> select 40 * 5 from lxw1234;
```

```
10
```

200

版权所有: <http://lxw1234.com>

## 4. 除法操作: /

语法: A / B

操作类型: 所有数值类型

说明: 返回 A 除以 B 的结果。结果的数值类型为 double

举例:

```
hive> select 40 / 5 from lxw1234;
```

8.0

**注意: hive 中最高精度的数据类型是 double,只精确到小数点后 16 位,在做除法运算的时候要特别注意**

```
hive>select ceil(28.0/6.999999999999999999) from lxw1234 limit 1;
```

结果为 4

```
hive>select ceil(28.0/6.999999999999999) from lxw1234 limit 1;
```

结果为 5

## 5. 取余操作: %

语法: A % B

操作类型: 所有数值类型

说明: 返回 A 除以 B 的余数。结果的数值类型等于 A 的类型和 B 的类型的最小父类型 (详见数据类型的继承关系)。

举例:

```
hive> select 41 % 5 from lxw1234;
```

1

```
hive> select 8.4 % 4 from lxw1234;
```

0.400000000000000036

**注意: 精度在 hive 中是个很大的问题,类似这样的操作最好通过 round 指定精度**

```
hive> select round(8.4 % 4 , 2) from lxw1234;
```

0.4

## 6. 位与操作: &

语法: A &amp; B

操作类型: 所有数值类型

说明: 返回 A 和 B 按位进行与操作的结果。结果的数值类型等于 A 的类型和 B 的类型的最小父类型 (详见数据类型的继承关系)。

举例:

```
hive> select 4 & 8 from lxw1234;
```

0

```
hive> select 6 & 4 from lxw1234;
```

4

版权所有: <http://lxw1234.com>

## 7. 位或操作: |

语法: A | B

操作类型: 所有数值类型

说明: 返回 A 和 B 按位进行或操作的结果。结果的数值类型等于 A 的类型和 B 的类型的最小父类型 (详见数据类型的继承关系)。

举例:

```
hive> select 4 | 8 from lxw1234;
```

12

```
hive> select 6 | 8 from lxw1234;
```

14

## 8. 位异或操作: ^

语法: A ^ B

操作类型: 所有数值类型

说明: 返回 A 和 B 按位进行异或操作的结果。结果的数值类型等于 A 的类型和 B 的类型的最小父类型 (详见数据类型的继承关系)。

举例:

```
hive> select 4 ^ 8 from lxw1234;
```

12

```
hive> select 6 ^ 4 from lxw1234;
```

2

## 9. 位取反操作: ~

语法: ~A

操作类型: 所有数值类型

说明: 返回 A 按位取反操作的结果。结果的数值类型等于 A 的类型。

举例:

```
hive> select ~6 from lxw1234;
```

-7

```
hive> select ~4 from lxw1234;  
-5
```

### 三、逻辑运算：

#### 1. 逻辑与操作: AND 、 &&

语法: A AND B

操作类型: boolean

说明: 如果 A 和 B 均为 TRUE, 则为 TRUE; 否则为 FALSE。如果 A 为 NULL 或 B 为 NULL, 则为 NULL

举例:

```
hive> select 1 from lxw1234 where 1=1 and 2=2;  
1
```

#### 2. 逻辑或操作: OR 、 ||

语法: A OR B

操作类型: boolean

说明: 如果 A 为 TRUE, 或者 B 为 TRUE, 或者 A 和 B 均为 TRUE, 则为 TRUE; 否则为 FALSE

举例:

```
hive> select 1 from lxw1234 where 1=2 or 2=2;  
1
```

版权所有: <http://lxw1234.com>

#### 3. 逻辑非操作: NOT、 !

语法: NOT A、 !A

操作类型: boolean

说明: 如果 A 为 FALSE, 或者 A 为 NULL, 则为 TRUE; 否则为 FALSE

举例:

```
hive> select 1 from lxw1234 where not 1=2;  
1
```

```
hive> select 1 from lxw1234 where !1=2;
```

OK

1

## 四、复合类型构造函数

### 1. map 结构

语法: `map(k1,v1,k2,v2,...)`

操作类型: `map`

说明: 使用给定的 key-value 对, 构造一个 map 数据结构

举例:

```
hive> select map('k1','v1','k2','v2') from lxw1234;
```

OK

```
{"k2":"v2","k1":"v1"}
```

### 2. struct 结构

语法: `struct(val1,val2,val3,...)`

操作类型: `struct`

说明: 使用给定的表达式, 构造一个 struct 数据结构

举例:

```
hive> select struct(1,'aaa',FALSE) from lxw1234;
```

OK

```
{"col1":1,"col2":"aaa","col3":false}
```

版权所有: <http://lxw1234.com>

### 3. named\_struct 结构

语法: `named_struct(name1,val1,name2,val2,name3,val3,...)`

操作类型: `struct`

说明: 使用给定的表达式, 构造一个指定列名的 struct 数据结构

举例:

```
hive> select named_struct('a',1,'b','aaa','c',FALSE) from lxw1234;
```

OK

```
{"a":1,"b":"aaa","c":false}
```

### 4. array 结构

语法: `array(val1,val2,val3,...)`

操作类型: `array`

说明: 使用给定的表达式, 构造一个 array 数据结构

举例:

```
hive> select array(1,2,3) from lxw1234;
OK
[1,2,3]
```

## 5. create\_union

语法: `create_union (tag, val1, val2, ...)`

操作类型: `uniontype`

说明: 使用给定的 `tag` 和表达式, 构造一个 `uniontype` 数据结构。 `tag` 表示使用第 `tag` 个表达式作为 `uniontype` 的 `value`

举例:

```
hive> select create_union(0,'ss',array(1,2,3)) from lxw1234;
```

OK

```
{0:"ss"}
```

```
hive> select create_union(1,'ss',array(1,2,3)) from lxw1234;
```

OK

```
{1:[1,2,3]}
```

版权所有: <http://lxw1234.com>

## 五、复合类型操作符

### 1. 获取 array 中的元素

语法: `A[n]`

操作类型: 所有基础类型

说明: 返回数组 `A` 中第 `n` 个索引的元素值。

举例:

```
hive> select array('a','b','c')[1] from lxw1234;
```

OK

```
b
```

### 2. 获取 map 中的元素

语法: `M[key]`

操作类型: 所有基础类型

说明: 返回 `map` 结构 `M` 中 `key` 对应的 `value`。

举例:

```
hive> select map('k1','v1')['k1'] from lxw1234;
```

OK

v1

版权所有: <http://lxw1234.com>

### 3. 获取 struct 中的元素

语法: S.x

操作类型: 所有类型

说明: 返回 struct 结构 S 中名为 x 的元素。

举例:

```
hive> select named_struct('a',1,'b','aaa','c',FALSE).c from lxw1234;
```

OK

false

## 六、数值计算函数

### 1. 取整函数: round

语法: round(double a)

返回值: BIGINT

说明: 返回 double 类型的整数值部分 (遵循四舍五入)

举例:

```
hive> select round(3.1415926) from lxw1234;
```

3

```
hive> select round(3.5) from lxw1234;
```

4

```
hive> create table lxw1234 as select round(9542.158) from lxw1234;
```

```
hive> describe lxw1234;
```

```
_c0      bigint
```

### 2. 指定精度取整函数: round

语法: round(double a, int d)

返回值: DOUBLE

说明: 返回指定精度 d 的 double 类型

举例:

```
hive> select round(3.1415926,4) from lxw1234;
```

3.1416



### 3. 向下取整函数: floor

语法: floor(double a)

返回值: BIGINT

说明: 返回等于或者小于该 double 变量的最大的整数

举例:

```
hive> select floor(3.1415926) from lxw1234;
```

3

```
hive> select floor(25) from lxw1234;
```

25

### 4. 向上取整函数: ceil

语法: ceil(double a)

返回值: BIGINT

说明: 返回等于或者大于该 double 变量的最小的整数

举例:

```
hive> select ceil(3.1415926) from lxw1234;
```

4

```
hive> select ceil(46) from lxw1234;
```

46

版权所有: <http://lxw1234.com>

### 5. 向上取整函数: ceiling

语法: ceiling(double a)

返回值: BIGINT

说明: 与 ceil 功能相同

举例:

```
hive> select ceiling(3.1415926) from lxw1234;
```

4

```
hive> select ceiling(46) from lxw1234;
```

46

### 6. 取随机数函数: rand

语法: rand(),rand(int seed)

返回值: double

说明: 返回一个 0 到 1 范围内的随机数。如果指定种子 `seed`, 则会等到一个稳定的随机数序列

举例:

```
hive> select rand() from lxw1234;
```

```
0.5577432776034763
```

```
hive> select rand() from lxw1234;
```

```
0.6638336467363424
```

```
hive> select rand(100) from lxw1234;
```

```
0.7220096548596434
```

```
hive> select rand(100) from lxw1234;
```

```
0.7220096548596434
```

## 7. 自然指数函数: `exp`

语法: `exp(double a)`

返回值: `double`

说明: 返回自然对数  $e$  的  $a$  次方

举例:

```
hive> select exp(2) from lxw1234;
```

```
7.38905609893065
```

自然对数函数: `ln`

语法: `ln(double a)`

返回值: `double`

说明: 返回  $a$  的自然对数

举例:

```
hive> select ln(7.38905609893065) from lxw1234;
```

```
2.0
```

版权所有: <http://lxw1234.com>

## 8. 以 10 为底对数函数: `log10`

语法: `log10(double a)`

返回值: `double`

说明: 返回以 10 为底的  $a$  的对数

举例:

```
hive> select log10(100) from lxw1234;
```

```
2.0
```

## 9. 以 2 为底对数函数: log2

语法: log2(double a)

返回值: double

说明: 返回以 2 为底的 a 的对数

举例:

```
hive> select log2(8) from lxw1234;  
3.0
```

## 10. 对数函数: log

语法: log(double base, double a)

返回值: double

说明: 返回以 base 为底的 a 的对数

举例:

```
hive> select log(4,256) from lxw1234;  
4.0
```

## 11. 幂运算函数: pow

语法: pow(double a, double p)

返回值: double

说明: 返回 a 的 p 次幂

举例:

```
hive> select pow(2,4) from lxw1234;  
16.0
```

## 12. 幂运算函数: power

语法: power(double a, double p)

返回值: double

说明: 返回 a 的 p 次幂,与 pow 功能相同

举例:

```
hive> select power(2,4) from lxw1234;  
16.0
```

版权所有: <http://lxw1234.com>

## 13. 开平方函数: sqrt

语法: `sqrt(double a)`

返回值: `double`

说明: 返回 a 的平方根

举例:

```
hive> select sqrt(16) from lxw1234;  
4.0
```

## 14. 二进制函数: bin

语法: `bin(BIGINT a)`

返回值: `string`

说明: 返回 a 的二进制代码表示

举例:

```
hive> select bin(7) from lxw1234;  
111
```

## 15. 十六进制函数: hex

语法: `hex(BIGINT a)`

返回值: `string`

说明: 如果变量是 `int` 类型, 那么返回 a 的十六进制表示; 如果变量是 `string` 类型, 则返回该字符串的十六进制表示

举例:

```
hive> select hex(17) from lxw1234;  
11  
hive> select hex('abc') from lxw1234;  
616263
```

## 16. 反转十六进制函数: unhex

语法: `unhex(string a)`

返回值: `string`

说明: 返回该十六进制字符串所代码的字符串

举例:

```
hive> select unhex('616263') from lxw1234;  
abc
```

```
hive> select unhex('11') from lxw1234;
-
hive> select unhex(616263) from lxw1234;
abc
```

## 17. 进制转换函数: conv

语法: conv(BIGINT num, int from\_base, int to\_base)  
返回值: string  
说明: 将数值 num 从 from\_base 进制转化到 to\_base 进制  
举例:  
hive> select conv(17,10,16) from lxw1234;  
11  
hive> select conv(17,10,2) from lxw1234;  
10001

版权所有: <http://lxw1234.com>

## 18. 绝对值函数: abs

语法: abs(double a) abs(int a)  
返回值: double int  
说明: 返回数值 a 的绝对值  
举例:  
hive> select abs(-3.9) from lxw1234;  
3.9  
hive> select abs(10.9) from lxw1234;  
10.9

## 19. 正取余函数: pmod

语法: pmod(int a, int b), pmod(double a, double b)  
返回值: int double  
说明: 返回正的 a 除以 b 的余数  
举例:  
hive> select pmod(9,4) from lxw1234;  
1  
hive> select pmod(-9,4) from lxw1234;  
3

## 20. 正弦函数: sin

语法: `sin(double a)`

返回值: `double`

说明: 返回 a 的正弦值

举例:

```
hive> select sin(0.8) from lxw1234;  
0.7173560908995228
```

## 21. 反正弦函数: asin

语法: `asin(double a)`

返回值: `double`

说明: 返回 a 的反正弦值

举例:

```
hive> select asin(0.7173560908995228) from lxw1234;  
0.8
```

## 22. 余弦函数: cos

语法: `cos(double a)`

返回值: `double`

说明: 返回 a 的余弦值

举例:

```
hive> select cos(0.9) from lxw1234;  
0.6216099682706644
```

## 23. 反余弦函数: acos

语法: `acos(double a)`

返回值: `double`

说明: 返回 a 的反余弦值

举例:

```
hive> select acos(0.6216099682706644) from lxw1234;  
0.9
```

## 24. positive 函数: positive

语法: positive(int a), positive(double a)

返回值: int double

说明: 返回 a

举例:

```
hive> select positive(-10) from lxw1234;
```

```
-10
```

```
hive> select positive(12) from lxw1234;
```

```
12
```

## 25. negative 函数: negative

语法: negative(int a), negative(double a)

返回值: int double

说明: 返回-a

举例:

```
hive> select negative(-5) from lxw1234;
```

```
5
```

```
hive> select negative(8) from lxw1234;
```

```
-8
```

## 七、集合操作函数

### 1. map 类型大小: size

语法: size(Map<K,V>)

返回值: int

说明: 返回 map 类型的 size

举例:

```
hive> select size(map('k1','v1','k2','v2')) from lxw1234;
```

```
OK
```

```
2
```

版权所有: <http://lxw1234.com>

### 2. array 类型大小: size

语法: size(Array<T>)

返回值: int  
说明: 返回 array 类型的 size  
举例:  
hive> select size(array(1,2,3,4,5)) from lxw1234;  
OK  
5

### 3. 判断元素数组是否包含元素: array\_contains

语法: array\_contains(Array<T>, value)  
返回值: boolean  
说明: 返回 Array<T>中是否包含元素 value  
举例:  
hive> select array\_contains(array(1,2,3,4,5),3) from lxw1234;  
OK  
true

### 4. 获取 map 中所有 value 集合

语法: map\_values(Map<K,V>)  
返回值: array<V>  
说明: 返回 Map<K,V>中所有 value 的集合  
举例:  
hive> select map\_values(map('k1','v1','k2','v2')) from lxw1234;  
OK  
["v2","v1"]

### 5. 获取 map 中所有 key 集合

语法: map\_keys(Map<K,V>)  
返回值: array<K>  
说明: 返回 Map<K,V>中所有 key 的集合  
举例:  
hive> select map\_keys(map('k1','v1','k2','v2')) from lxw1234;  
OK  
["k2","k1"]

### 6. 数组排序

语法: sort\_array(Array<T>)  
返回值: array<t>



说明: 对 `Array<T>` 进行升序排序

举例:

```
hive> select sort_array(array(5,7,3,6,9)) from lxw1234;
```

OK

[3,5,6,7,9]

版权所有: <http://lxw1234.com>

## 八、类型转换函数

### 1. 二进制转换: `binary`

语法: `binary(string|binary)`

返回值: `binary`

说明: 将 `string` 类型转换为二进制

举例:

```
hive> select binary('lxw1234') from lxw1234;
```

OK

lxw1234

### 2. 基础类型之间强制转换: `cast`

语法: `cast(expr as <type>)`

返回值: `<type>`

说明: 将 `expr` 转换成 `<type>`

举例:

```
hive> select cast('1' as DOUBLE) from lxw1234;
```

OK

1.0

## 九、日期函数

### 1. UNIX 时间戳转日期函数: `from_unixtime`

语法: `from_unixtime(bigint unixtime[, string format])`

返回值: `string`

说明: 转化 UNIX 时间戳 (从 1970-01-01 00:00:00 UTC 到指定时间的秒数) 到当前时区的时间格式

举例:

```
hive> select from_unixtime(1323308943,'yyyyMMdd') from lxw1234;  
20111208
```

## 2. 获取当前 UNIX 时间戳函数: `unix_timestamp`

语法: `unix_timestamp()`

返回值: `bigint`

说明: 获得当前时区的 UNIX 时间戳

举例:

```
hive> select unix_timestamp() from lxw1234;  
1323309615
```

版权所有: <http://lxw1234.com>

## 3. 日期转 UNIX 时间戳函数: `unix_timestamp`

语法: `unix_timestamp(string date)`

返回值: `bigint`

说明: 转换格式为"yyyy-MM-dd HH:mm:ss"的日期到 UNIX 时间戳。如果转化失败, 则返回 0。

举例:

```
hive> select unix_timestamp('2011-12-07 13:01:03') from lxw1234;  
1323234063
```

## 4. 指定格式日期转 UNIX 时间戳函数: `unix_timestamp`

语法: `unix_timestamp(string date, string pattern)`

返回值: `bigint`

说明: 转换 `pattern` 格式的日期到 UNIX 时间戳。如果转化失败, 则返回 0。

举例:

```
hive> select unix_timestamp('20111207 13:01:03','yyyyMMdd HH:mm:ss') from lxw1234;  
1323234063
```

## 5. 日期时间转日期函数: `to_date`

语法: `to_date(string timestamp)`

返回值: `string`

说明: 返回日期时间字段中的日期部分。

举例:

```
hive> select to_date('2011-12-08 10:03:01') from lxw1234;  
2011-12-08
```

## 6. 日期转年函数: year

语法: year(string date)

返回值: int

说明: 返回日期中的年。

举例:

```
hive> select year('2011-12-08 10:03:01') from lxw1234;  
2011  
hive> select year('2012-12-08') from lxw1234;  
2012
```

版权所有: <http://lxw1234.com>

## 7. 日期转月函数: month

语法: month (string date)

返回值: int

说明: 返回日期中的月份。

举例:

```
hive> select month('2011-12-08 10:03:01') from lxw1234;  
12  
hive> select month('2011-08-08') from lxw1234;  
8
```

## 8. 日期转天函数: day

语法: day (string date)

返回值: int

说明: 返回日期中的天。

举例:

```
hive> select day('2011-12-08 10:03:01') from lxw1234;  
8  
hive> select day('2011-12-24') from lxw1234;  
24
```

## 9. 日期转小时函数: hour

语法: hour (string date)

返回值: int

说明: 返回日期中的小时。

举例:

```
hive> select hour('2011-12-08 10:03:01') from lxw1234;  
10
```

## 10. 日期转分钟函数: minute

语法: minute (string date)

返回值: int

说明: 返回日期中的分钟。

举例:

```
hive> select minute('2011-12-08 10:03:01') from lxw1234;  
3
```

版权所有: <http://lxw1234.com>

## 11. 日期转秒函数: second

语法: second (string date)

返回值: int

说明: 返回日期中的秒。

举例:

```
hive> select second('2011-12-08 10:03:01') from lxw1234;  
1
```

## 12. 日期转周函数: weekofyear

语法: weekofyear (string date)

返回值: int

说明: 返回日期在当前的周数。

举例:

```
hive> select weekofyear('2011-12-08 10:03:01') from lxw1234;  
49
```

### 13. 日期比较函数: datediff

语法: datediff(string enddate, string startdate)

返回值: int

说明: 返回结束日期减去开始日期的天数。

举例:

```
hive> select datediff('2012-12-08','2012-05-09') from lxw1234;  
213
```

### 14. 日期增加函数: date\_add

语法: date\_add(string startdate, int days)

返回值: string

说明: 返回开始日期 startdate 增加 days 天后的日期。

举例:

```
hive> select date_add('2012-12-08',10) from lxw1234;  
2012-12-18
```

### 15. 日期减少函数: date\_sub

语法: date\_sub (string startdate, int days)

返回值: string

说明: 返回开始日期 startdate 减少 days 天后的日期。

举例:

```
hive> select date_sub('2012-12-08',10) from lxw1234;  
2012-11-28
```

版权所有: <http://lxw1234.com>

## 十、条件函数

### 1. If 函数: if

语法: if(boolean testCondition, T valueTrue, T valueFalseOrNull)

返回值: T

说明: 当条件 testCondition 为 TRUE 时, 返回 valueTrue; 否则返回 valueFalseOrNull

举例:

```
hive> select if(1=2,100,200) from lxw1234;
```

```
200
```

```
hive> select if(1=1,100,200) from lxw1234;
```

```
100
```

## 2. 非空查找函数: COALESCE

语法: COALESCE(T v1, T v2, ...)

返回值: T

说明: 返回参数中的第一个非空值; 如果所有值都为 NULL, 那么返回 NULL

举例:

```
hive> select COALESCE(null,'100','50' ) from lxw1234;
```

```
100
```

## 3. 条件判断函数: CASE

语法: CASE a WHEN b THEN c [WHEN d THEN e]\* [ELSE f] END

返回值: T

说明: 如果 a 等于 b, 那么返回 c; 如果 a 等于 d, 那么返回 e; 否则返回 f

举例:

```
hive> Select case 100 when 50 then 'tom' when 100 then 'mary' else 'tim' end from  
lxw1234;
```

```
mary
```

```
hive> Select case 200 when 50 then 'tom' when 100 then 'mary' else 'tim' end from  
lxw1234;
```

```
tim
```

## 4. 条件判断函数: CASE

语法: CASE WHEN a THEN b [WHEN c THEN d]\* [ELSE e] END

返回值: T

说明: 如果 a 为 TRUE,则返回 b; 如果 c 为 TRUE, 则返回 d; 否则返回 e

举例:

```
hive> select case when 1=2 then 'tom' when 2=2 then 'mary' else 'tim' end from lxw1234;
```

```
mary
```

```
hive> select case when 1=1 then 'tom' when 2=2 then 'mary' else 'tim' end from lxw1234;
```

```
tom
```

## 十一、字符串函数

### 1. 字符 `ascii` 码函数: `ascii`

语法: `ascii(string str)`

返回值: `int`

说明: 返回字符串 `str` 中第一个字符的 `ascii` 码

举例:

```
hive> select ascii('ba') from lxw1234;
```

```
OK
```

```
98
```

### 2. `base64` 字符串

语法: `base64(binary bin)`

返回值: `string`

说明: 返回二进制 `bin` 的 `base` 编码字符串

举例:

```
hive> select base64(binary('lxw1234')) from lxw1234;
```

```
OK
```

```
bHh3MTIzNA==
```

### 3. 字符串连接函数: `concat`

语法: `concat(string A, string B...)`

返回值: `string`

说明: 返回输入字符串连接后的结果, 支持任意个输入字符串

举例:

```
hive> select concat('abc','def','gh') from lxw1234;
```

```
abcdefgh
```

### 4. 带分隔符字符串连接函数: `concat_ws`

语法: `concat_ws(string SEP, string A, string B...)`

返回值: `string`

说明: 返回输入字符串连接后的结果, `SEP` 表示各个字符串间的分隔符

举例:

```
hive> select concat_ws(',', 'abc', 'def', 'gh') from lxw1234;  
abc,def,gh
```

## 5. 数组转换成字符串的函数: `concat_ws`

语法: `concat_ws(string SEP, array<string>)`

返回值: string

说明: 返回将数组链接成字符串后的结果, SEP 表示各个字符串间的分隔符

举例:

```
hive> select concat_ws('|', array('a', 'b', 'c')) from lxw1234;  
OK  
a|b|c
```

版权所有: <http://lxw1234.com>

## 6. 小数位格式化成字符串函数: `format_number`

语法: `format_number(number x, int d)`

返回值: string

说明: 将数值 x 的小数位格式化成 d 位, 四舍五入

举例:

```
hive> select format_number(5.23456, 3) from lxw1234;  
OK  
5.235
```

## 7. 字符串截取函数: `substr, substring`

语法: `substr(string A, int start), substring(string A, int start)`

返回值: string

说明: 返回字符串 A 从 start 位置到结尾的字符串

举例:

```
hive> select substr('abcde', 3) from lxw1234;  
cde  
hive> select substring('abcde', 3) from lxw1234;  
cde  
hive> select substr('abcde', -1) from lxw1234; (和 ORACLE 相同)  
e
```



## 8. 字符串截取函数: substr, substring

语法: substr(string A, int start, int len), substring(string A, int start, int len)

返回值: string

说明: 返回字符串 A 从 start 位置开始, 长度为 len 的字符串

举例:

```
hive> select substr('abcde',3,2) from lxw1234;
```

```
cd
```

```
hive> select substring('abcde',3,2) from lxw1234;
```

```
cd
```

```
hive> select substring('abcde',-2,2) from lxw1234;
```

```
de
```

版权所有: <http://lxw1234.com>

## 9. 字符串查找函数: instr

语法: instr(string str, string substr)

返回值: int

说明: 返回字符串 substr 在 str 中首次出现的位置

举例:

```
hive> select instr('abcdef', 'df') from lxw1234;
```

```
OK
```

```
4
```

## 10. 字符串长度函数: length

语法: length(string A)

返回值: int

说明: 返回字符串的长度

举例:

```
hive> select length('abc') from lxw1234;
```

```
OK
```

```
3
```

## 11. 字符串查找函数: locate

语法: locate(string substr, string str[, int pos])

返回值: int

说明: 返回字符串 substr 在 str 中从 pos 后查找, 首次出现的位置

举例:

```
hive> select locate('a','abcda',1) from lxw1234;
```

OK

1

```
hive> select locate('a','abcda',2) from lxw1234;
```

OK

5

版权所有: <http://lxw1234.com>

## 12. 字符串格式化函数: printf

语法: printf(String format, Obj... args)

返回值: string

说明: 将指定对象用 format 格式进行格式化.

举例:

```
hive> select printf("%08X",123) from lxw1234;
```

OK

0000007B

## 13. 字符串转换成 map 函数: str\_to\_map

语法: str\_to\_map(text[, delimiter1, delimiter2])

返回值: map<string,string>

说明: 将字符串按照给定的分隔符转换成 map 结构.

举例:

```
hive> select str_to_map('k1:v1,k2:v2') from lxw1234;
```

OK

```
{"k2":"v2","k1":"v1"}
```

```
hive> select str_to_map('k1=v1,k2=v2',','=' ') from lxw1234;
```

OK

```
{"k2":"v2","k1":"v1"}
```

## 14. base64 解码函数: unbase64(string str)

语法: unbase64(string str)

返回值: binary

说明: 将给定的 base64 字符串解码成二进制.

举例:

```
hive> select unbase64('bHh3MTIzNA==') from lxw1234;  
OK  
lxw1234
```

## 15. 字符串转大写函数: upper,ucase

语法: upper(string A) ucase(string A)

返回值: string

说明: 返回字符串 A 的大写格式

举例:

```
hive> select upper('abSEd') from lxw1234;  
ABSED  
hive> select ucase('abSEd') from lxw1234;  
ABSED
```

版权所有: <http://lxw1234.com>

## 16. 字符串转小写函数: lower,lcase

语法: lower(string A) lcase(string A)

返回值: string

说明: 返回字符串 A 的小写格式

举例:

```
hive> select lower('abSEd') from lxw1234;  
absed  
hive> select lcase('abSEd') from lxw1234;  
absed
```

## 17. 去空格函数: trim

语法: trim(string A)

返回值: string

说明: 去除字符串两边的空格

举例:

```
hive> select trim(' abc ') from lxw1234;  
abc
```

## 18. 左边去空格函数: ltrim

语法: ltrim(string A)

返回值: string

说明: 去除字符串左边的空格

举例:

```
hive> select ltrim(' abc ') from lxw1234;  
abc
```

版权所有: <http://lxw1234.com>

## 19. 右边去空格函数: rtrim

语法: rtrim(string A)

返回值: string

说明: 去除字符串右边的空格

举例:

```
hive> select rtrim(' abc ') from lxw1234;  
abc
```

## 20. 正则表达式替换函数: regexp\_replace

语法: regexp\_replace(string A, string B, string C)

返回值: string

说明: 将字符串 A 中的符合 java 正则表达式 B 的部分替换为 C。注意, 在有些情况下要使用转义字符, 类似 oracle 中的 regexp\_replace 函数。

举例:

```
hive> select regexp_replace('foobar', 'oo|ar', '') from lxw1234;  
fb
```

## 21. 正则表达式解析函数: regexp\_extract

语法: regexp\_extract(string subject, string pattern, int index)

返回值: string

说明: 将字符串 subject 按照 pattern 正则表达式的规则拆分, 返回 index 指定的字符。

举例:

```
hive> select regexp_extract('foothebar', 'foo(.*?)(bar)', 1) from lxw1234;  
the
```

```
hive> select regexp_extract('foothebar', 'foo(.?)(bar)', 2) from lxw1234;
bar
hive> select regexp_extract('foothebar', 'foo(.?)(bar)', 0) from lxw1234;
foothebar
```

注意，在有些情况下要使用转义字符，下面的等号要用双竖线转义，这是 java 正则表达式的规则。

```
select data_field,
regexp_extract(data_field,'.*?bgStart\\=[^&]+',1) as aaa,
regexp_extract(data_field,'.*?contentLoaded_headStart\\=[^&]+',1) as bbb,
regexp_extract(data_field,'.*?AppLoad2Req\\=[^&]+',1) as ccc
from pt_nginx_loginlog_st
where pt = '2012-03-26' limit 2;
```

版权所有：<http://lxw1234.com>

## 22. URL 解析函数：parse\_url

语法：parse\_url(string urlString, string partToExtract [, string keyToExtract])

返回值：string

说明：返回 URL 中指定的部分。partToExtract 的有效值为：HOST, PATH, QUERY, REF, PROTOCOL, AUTHORITY, FILE, and USERINFO.

举例：

```
hive> select parse_url('http://facebook.com/path1/p.php?k1=v1&k2=v2#Ref1', 'HOST') from
lxw1234;
facebook.com
hive> select parse_url('http://facebook.com/path1/p.php?k1=v1&k2=v2#Ref1', 'QUERY', 'k1')
from lxw1234;
v1
```

## 23. json 解析函数：get\_json\_object

语法：get\_json\_object(string json\_string, string path)

返回值：string

说明：解析 json 的字符串 json\_string, 返回 path 指定的内容。如果输入的 json 字符串无效，那么返回 NULL。

举例：

```
hive> select  get_json_object('{"store":
> {"fruit":\ [{"weight":8,"type":"apple"}, {"weight":9,"type":"pear"}],
> "bicycle":{"price":19.95,"color":"red"}
> },
> "email":"amy@only_for_json_udf_test.net",
```

```
> "owner":"amy"  
> }  
> ',${.owner}') from lxw1234;  
amy
```

## 24. 空格字符串函数: space

语法: space(int n)  
返回值: string  
说明: 返回长度为 n 的字符串  
举例:  
hive> select space(10) from lxw1234;  
hive> select length(space(10)) from lxw1234;  
10

版权所有: <http://lxw1234.com>

## 25. 重复字符串函数: repeat

语法: repeat(string str, int n)  
返回值: string  
说明: 返回重复 n 次后的 str 字符串  
举例:  
hive> select repeat('abc',5) from lxw1234;  
abcabcabcabcabc

## 26. 左补足函数: lpad

语法: lpad(string str, int len, string pad)  
返回值: string  
说明: 将 str 进行用 pad 进行左补足到 len 位  
举例:  
hive> select lpad('abc',10,'td') from lxw1234;  
tdtdtdtabc

**注意: 与 GP, ORACLE 不同, pad 不能默认**

## 27. 右补足函数: rpad

语法: rpad(string str, int len, string pad)

返回值: string

说明: 将 str 进行用 pad 进行右补足到 len 位

举例:

```
hive> select rpad('abc',10,'td') from lxw1234;  
abctdtdtdt
```

## 28. 分割字符串函数: split

语法: split(string str, string pat)

返回值: array

说明: 按照 pat 字符串分割 str, 会返回分割后的字符串数组

举例:

```
hive> select split('abctdef','t') from lxw1234;  
["ab","cd","ef"]
```

版权所有: <http://lxw1234.com>

## 29. 集合查找函数: find\_in\_set

语法: find\_in\_set(string str, string strList)

返回值: int

说明: 返回 str 在 strlist 第一次出现的位置, strlist 是用逗号分割的字符串。如果没有找该 str 字符, 则返回 0

举例:

```
hive> select find_in_set('ab','ef,ab,de') from lxw1234;  
2  
hive> select find_in_set('at','ef,ab,de') from lxw1234;  
0
```

## 30. 分词函数: sentences

语法: sentences(string str, string lang, string locale)

返回值: array<array<string>>  
说明: 返回输入 str 分词后的单词数组  
举例:  
hive> select sentences('hello word!hello hive,hi hive,hello hive') from lxw1234;  
OK  
[["hello","word"],["hello","hive","hi","hive","hello","hive"]]

## 31. 分词后统计一起出现频次最高的 TOP-K

语法: ngrams(array<array<string>>, int N, int K, int pf)  
返回值: array<struct<string,double>>  
说明: 与 sentences()函数一起使用, 分词后, 统计分词结果中一起出现频次最高的 TOP-K 结果  
举例:  
hive> SELECT ngrams(sentences('hello word!hello hive,hi hive,hello hive'),2,2) FROM lxw1234;  
[{"ngram":["hello","hive"],"estfrequency":2.0},{ "ngram":["hive","hello"],"estfrequency":1.0}]  
该查询中, 统计的是两个词在一起出现频次最高的 TOP-2 结果中, hello 与 hive 同时出现 2 次

版权所有: <http://lxw1234.com>

## 32. 分词后统计与指定单词一起出现频次最高的 TOP-K

语法: context\_ngrams(array<array<string>>, array<string>, int K, int pf)  
返回值: array<struct<string,double>>  
说明: 与 sentences()函数一起使用, 分词后, 统计分词结果中与数组中指定的单词一起出现(包括顺序)频次最高的 TOP-K 结果  
举例:  
hive> SELECT context\_ngrams(sentences('hello word!hello hive,hi hive,hello hive'),array('hello',null),3) FROM lxw1234;  
[{"ngram":["hive"],"estfrequency":2.0},{ "ngram":["word"],"estfrequency":1.0}]  
该查询中, 统计的是与'hello'一起出现, 并且在 hello 后面的频次最高的 TOP-3 结果中, hello 与 hive 同时出现 2 次, hello 与 word 同时出现 1 次。

hive> SELECT context\_ngrams(sentences('hello word!hello hive,hi hive,hello hive'),array(null,'hive'),3) FROM lxw1234;  
[{"ngram":["hello"],"estfrequency":2.0},{ "ngram":["hi"],"estfrequency":1.0}]  
该查询中, 统计的是与'hive'一起出现, 并且在 hive 之前的频次最高的 TOP-3



## 十二、混合函数

### 1. 调用 Java 函数: java\_method

语法: java\_method(class, method[, arg1[, arg2..]])

返回值: varies

说明: 调用 Java 中的方法处理数据。

举例:

```
hive> select reflect("java.net.URLEncoder", "encode", 'http://lxw1234.com',"UTF-8") from lxw1234;
```

OK

[http%3A%2F%2Flxw1234.com](http://lxw1234.com)

该查询中调用 java.net.URLEncoder 中的 encode 方法，给该方法传的参数为 'http://lxw1234.com',"UTF-8"

### 2. 调用 Java 函数: reflect

语法: reflect(class, method[, arg1[, arg2..]])

返回值: varies

说明: 调用 Java 中的方法处理数据。

举例:

```
hive> select reflect("java.net.URLDecoder", "decode", 'http%3A%2F%2Flxw1234.com',"UTF-8") from lxw1234;
```

OK

<http://lxw1234.com>

### 3. 字符串的 hash 值: hash

语法: hash(a1[, a2...])

返回值: int

说明: 返回字符串的 hash 值。

举例:

```
hive> select hash('lxw1234.com') from lxw1234;
```

OK

-809268416

版权所有: <http://lxw1234.com>

## 十三、XPath 解析 XML 函数

### 1. xpath

语法: `xpath(string xmlstr,string xpath_expression)`

返回值: `array<string>`

说明: 从 xml 字符串中返回匹配到表达式的结果数组。

举例:

//获取 xml 字符串中 a/b/节点的值

```
hive> select xpath('<a><b>b1</b><b>b2</b><c>c1</c></a>', 'a/b/text()') from lxw1234;
```

OK

```
["b1","b2"]
```

//获取 xml 字符串中所有名为 id 的属性值

```
hive> select xpath('<a><b id="foo">b1</b><b id="bar">b2</b></a>', '//*[@id]') from lxw1234;
```

OK

```
["foo","bar"]
```

### 2. xpath\_string

语法: `xpath_string(string xmlstr,string xpath_expression)`

返回值: `string`

说明: 默认情况下, 从 xml 字符串中返回第一个匹配到表达式的节点的值。

举例:

```
hive> SELECT xpath_string ('<a><b>b1</b><b>b2</b></a>', '//b') FROM lxw1234;
```

OK

```
b1
```

3.

//指定返回匹配到哪一个节点

```
hive> SELECT xpath_string ('<a><b>b1</b><b>b2</b></a>', '//b[2]') FROM lxw1234;
```

OK

```
b2
```

版权所有: <http://lxw1234.com>

### 3. xpath\_boolean

语法: `xpath_boolean (string xmlstr,string xpath_expression)`

返回值: boolean

说明: 返回 xml 字符串中是否匹配 xml 表达式。

举例:

```
hive> SELECT xpath_boolean ('<a><b>b</b></a>', 'a/b') FROM lxw1234;
```

OK

true

```
hive> SELECT xpath_boolean ('<a><b>10</b></a>', 'a/b < 10') FROM lxw1234;
```

OK

false

## 4. xpath\_short, xpath\_int, xpath\_long

语法: xpath\_short (string xmlstr,string xpath\_expression)

        xpath\_int (string xmlstr,string xpath\_expression)

        xpath\_long (string xmlstr,string xpath\_expression)

返回值: int

说明: 返回 xml 字符串中经过 xml 表达式计算后的值, 如果不匹配, 则返回 0。

举例:

```
hive> SELECT xpath_int ('<a>this is not a number</a>', 'a') FROM lxw1234;
```

OK

0

```
hive> SELECT xpath_int ('<a><b class="odd">1</b><b class="even">2</b><b class="odd">4</b><c>8</c></a>', 'sum(a/*)') FROM lxw1234;
```

OK

15

```
hive> select xpath_long('<a><b>10.5</b><c>11.2</c></a>', 'sum(a/*)') from lxw1234;
```

OK

21

## 5. xpath\_float, xpath\_double, xpath\_number

语法: xpath\_float (string xmlstr,string xpath\_expression)

        xpath\_double (string xmlstr,string xpath\_expression)

        xpath\_number (string xmlstr,string xpath\_expression)

返回值: number

说明: 返回 xml 字符串中经过 xml 表达式计算后的值, 如果不匹配, 则返回 0。

举例:

```
hive> select xpath_double('<a><b>10.5</b><c>11.2</c></a>', 'sum(a/*)') from lxw1234;
```

OK

21.7

版权所有： <http://lxw1234.com>

XPath 参考资料：

<http://baike.baidu.com/view/307399.htm>

xpath 函数 Hive 官方介绍文档：

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual+XPathUDF>

## 十四、汇总统计函数（UDAF）

### 1. 个数统计函数: count

语法: count(\*), count(expr), count(DISTINCT expr[, expr\_.])

返回值: int

说明: count(\*)统计检索出的行的个数，包括 NULL 值的行；count(expr)返回指定字段的非空值的个数；count(DISTINCT expr[, expr\_.])返回指定字段的不同的非空值的个数

举例：

```
hive> select count(*) from lxw1234;
```

```
20
```

```
hive> select count(distinct t) from lxw1234;
```

```
10
```

### 2. 总和统计函数: sum

语法: sum(col), sum(DISTINCT col)

返回值: double

说明: sum(col)统计结果集中 col 的相加的结果；sum(DISTINCT col)统计结果中 col 不同值相加的结果

举例：

```
hive> select sum(t) from lxw1234;
```

```
100
```

```
hive> select sum(distinct t) from lxw1234;
```

```
70
```

版权所有： <http://lxw1234.com>

### 3. 平均值统计函数: avg

语法: avg(col), avg(DISTINCT col)

返回值: double

说明: avg(col)统计结果集中 col 的平均值; avg(DISTINCT col)统计结果中 col 不同值相加的平均值

举例:

```
hive> select avg(t) from lxw1234;
```

```
50
```

```
hive> select avg (distinct t) from lxw1234;
```

```
30
```

### 4. 最小值统计函数: min

语法: min(col)

返回值: double

说明: 统计结果集中 col 字段的最小值

举例:

```
hive> select min(t) from lxw1234;
```

```
20
```

### 5. 最大值统计函数: max

语法: maxcol)

返回值: double

说明: 统计结果集中 col 字段的最大值

举例:

```
hive> select max(t) from lxw1234;
```

```
120
```

### 6. 非空集合总体变量函数: var\_pop

语法: var\_pop(col)

返回值: double

说明: 统计结果集中 col 非空集合的总体变量 (忽略 null)

举例:

版权所有: <http://lxw1234.com>

## 7. 非空集合样本变量函数: var\_samp

语法: var\_samp (col)

返回值: double

说明: 统计结果集中 col 非空集合的样本变量 (忽略 null)

举例:

## 8. 总体标准偏离函数: stddev\_pop

语法: stddev\_pop(col)

返回值: double

说明: 该函数计算总体标准偏离, 并返回总体变量的平方根, 其返回值与 VAR\_POP 函数的平方根相同

举例:

## 9. 样本标准偏离函数: stddev\_samp

语法: stddev\_samp (col)

返回值: double

说明: 该函数计算样本标准偏离

举例:

## 10. 中位数函数: percentile

语法: percentile(BIGINT col, p)

返回值: double

说明: 求准确的第 pth 个百分位数, p 必须介于 0 和 1 之间, 但是 col 字段目前只支持整数, 不支持浮点数类型

举例:

## 11. 中位数函数: percentile

语法: percentile(BIGINT col, array(p1 [, p2]...))

返回值: array<double>

说明: 功能和上述类似, 之后后面可以输入多个百分位数, 返回类型也为 `array<double>`, 其中为对应的百分位数。

举例:

```
select percentile(score,<0.2,0.4>) from lxw1234; 取 0.2, 0.4 位置的数据
```

版权所有: <http://lxw1234.com>

## 12. 近似中位数函数: percentile\_approx

语法: `percentile_approx(DOUBLE col, p [, B])`

返回值: `double`

说明: 求近似的第 `p`th 个百分位数, `p` 必须介于 0 和 1 之间, 返回类型为 `double`, 但是 `col` 字段支持浮点类型。参数 `B` 控制内存消耗的近似精度, `B` 越大, 结果的准确度越高。

默认为 10,000。当 `col` 字段中的 `distinct` 值的个数小于 `B` 时, 结果为准确的百分位数

举例:

## 13. 近似中位数函数: percentile\_approx

语法: `percentile_approx(DOUBLE col, array(p1 [, p2]...) [, B])`

返回值: `array<double>`

说明: 功能和上述类似, 之后后面可以输入多个百分位数, 返回类型也为 `array<double>`, 其中为对应的百分位数。

举例:

## 14. 直方图: histogram\_numeric

语法: `histogram_numeric(col, b)`

返回值: `array<struct {'x','y'}>`

说明: 以 `b` 为基准计算 `col` 的直方图信息。

举例:

```
hive> select histogram_numeric(100,5) from lxw1234;
[{"x":100.0,"y":1.0}]
```

版权所有: <http://lxw1234.com>

## 15. 集合去重数: collect\_set

语法: `collect_set (col)`

返回值: array

说明: 将 col 字段进行去重, 合并成一个数组。

举例:

```
hive> select cookie,ip from lxw1234;
```

```
cookie1 127.0.0.1
```

```
cookie1 127.0.0.1
```

```
cookie1 127.0.0.2
```

```
cookie1 127.0.0.3
```

```
hive> select cookie,collect_set(ip) from lxw1234 group by cookie;
```

```
cookie1 ["127.0.0.1","127.0.0.2","127.0.0.3"]
```

## 16. 集合不去重函数: collect\_list

语法: collect\_list (col)

返回值: array

说明: 将 col 字段合并成一个数组,不去重

举例:

```
hive> select cookie,ip from lxw1234;
```

```
cookie1 127.0.0.1
```

```
cookie1 127.0.0.1
```

```
cookie1 127.0.0.2
```

```
cookie1 127.0.0.3
```

```
hive>select cookie,collect_list(ip) from lxw1234 group by cookie;
```

```
cookie1 ["127.0.0.1","127.0.0.1","127.0.0.2","127.0.0.3"]
```

## 十五、表格生成函数 Table-Generating Functions (UDTF)

### 1. 数组拆分成多行: explode

语法: explode (ARRAY)

返回值: 多行

说明: 将数组中的元素拆分成多行显示

举例:

```
hive> select explode(array(1,2,3)) from lxw1234;
```

```
OK
```

```
1
```

```
2
```

```
3
```

版权所有: <http://lxw1234.com>



## 2. Map 拆分成多行: explode

语法: explode(Map)

返回值: 多行

说明: 将 Map 中的元素拆分成多行显示

举例:

```
hive> select explode(map('k1','v1','k2','v2')) from lxw1234;
```

OK

```
k2      v2
```

```
k1      v1
```