目录

[一、关系运算： 4](#_Toc323327845)

[1. 等值比较: = 4](#_Toc323327846)

[2. 不等值比较: <> 4](#_Toc323327847)

[3. 小于比较: < 4](#_Toc323327848)

[4. 小于等于比较: <= 4](#_Toc323327849)

[5. 大于比较: > 5](#_Toc323327850)

[6. 大于等于比较: >= 5](#_Toc323327851)

[7. 空值判断: IS NULL 5](#_Toc323327852)

[8. 非空判断: IS NOT NULL 6](#_Toc323327853)

[9. LIKE比较: LIKE 6](#_Toc323327854)

[10. JAVA的LIKE操作: RLIKE 6](#_Toc323327855)

[11. REGEXP操作: REGEXP 7](#_Toc323327856)

[二、数学运算： 7](#_Toc323327857)

[1. 加法操作: + 7](#_Toc323327858)

[2. 减法操作: - 7](#_Toc323327859)

[3. 乘法操作: \* 8](#_Toc323327860)

[4. 除法操作: / 8](#_Toc323327861)

[5. 取余操作: % 8](#_Toc323327862)

[6. 位与操作: & 9](#_Toc323327863)

[7. 位或操作: | 9](#_Toc323327864)

[8. 位异或操作: ^ 9](#_Toc323327865)

[9．位取反操作: ~ 10](#_Toc323327866)

[三、逻辑运算： 10](#_Toc323327867)

[1. 逻辑与操作: AND 10](#_Toc323327868)

[2. 逻辑或操作: OR 10](#_Toc323327869)

[3. 逻辑非操作: NOT 10](#_Toc323327870)

[四、数值计算 11](#_Toc323327871)

[1. 取整函数: round 11](#_Toc323327872)

[2. 指定精度取整函数: round 11](#_Toc323327873)

[3. 向下取整函数: floor 11](#_Toc323327874)

[4. 向上取整函数: ceil 12](#_Toc323327875)

[5. 向上取整函数: ceiling 12](#_Toc323327876)

[6. 取随机数函数: rand 12](#_Toc323327877)

[7. 自然指数函数: exp 13](#_Toc323327878)

[8. 以10为底对数函数: log10 13](#_Toc323327879)

[9. 以2为底对数函数: log2 13](#_Toc323327880)

[10. 对数函数: log 13](#_Toc323327881)

[11. 幂运算函数: pow 14](#_Toc323327882)

[12. 幂运算函数: power 14](#_Toc323327883)

[13. 开平方函数: sqrt 14](#_Toc323327884)

[14. 二进制函数: bin 14](#_Toc323327885)

[15. 十六进制函数: hex 15](#_Toc323327886)

[16. 反转十六进制函数: unhex 15](#_Toc323327887)

[17. 进制转换函数: conv 15](#_Toc323327888)

[18. 绝对值函数: abs 16](#_Toc323327889)

[19. 正取余函数: pmod 16](#_Toc323327890)

[20. 正弦函数: sin 16](#_Toc323327891)

[21. 反正弦函数: asin 16](#_Toc323327892)

[22. 余弦函数: cos 17](#_Toc323327893)

[23. 反余弦函数: acos 17](#_Toc323327894)

[24. positive函数: positive 17](#_Toc323327895)

[25. negative函数: negative 17](#_Toc323327896)

[五、日期函数 18](#_Toc323327897)

[1. UNIX时间戳转日期函数: from\_unixtime 18](#_Toc323327898)

[2. 获取当前UNIX时间戳函数: unix\_timestamp 18](#_Toc323327899)

[3. 日期转UNIX时间戳函数: unix\_timestamp 18](#_Toc323327900)

[4. 指定格式日期转UNIX时间戳函数: unix\_timestamp 18](#_Toc323327901)

[5. 日期时间转日期函数: to\_date 19](#_Toc323327902)

[6. 日期转年函数: year 19](#_Toc323327903)

[7. 日期转月函数: month 19](#_Toc323327904)

[8. 日期转天函数: day 19](#_Toc323327905)

[9. 日期转小时函数: hour 20](#_Toc323327906)

[10. 日期转分钟函数: minute 20](#_Toc323327907)

[11. 日期转秒函数: second 20](#_Toc323327908)

[12. 日期转周函数: weekofyear 20](#_Toc323327909)

[13. 日期比较函数: datediff 21](#_Toc323327910)

[14. 日期增加函数: date\_add 21](#_Toc323327911)

[15. 日期减少函数: date\_sub 21](#_Toc323327912)

[六、条件函数 21](#_Toc323327913)

[1. If函数: if 21](#_Toc323327914)

[2. 非空查找函数: COALESCE 22](#_Toc323327915)

[3. 条件判断函数：CASE 22](#_Toc323327916)

[4. 条件判断函数：CASE 22](#_Toc323327917)

[七、字符串函数 23](#_Toc323327918)

[1. 字符串长度函数：length 23](#_Toc323327919)

[2. 字符串反转函数：reverse 23](#_Toc323327920)

[3. 字符串连接函数：concat 23](#_Toc323327921)

[4. 带分隔符字符串连接函数：concat\_ws 23](#_Toc323327922)

[5. 字符串截取函数：substr,substring 24](#_Toc323327923)

[6. 字符串截取函数：substr,substring 24](#_Toc323327924)

[7. 字符串转大写函数：upper,ucase 24](#_Toc323327925)

[8. 字符串转小写函数：lower,lcase 25](#_Toc323327926)

[9. 去空格函数：trim 25](#_Toc323327927)

[10. 左边去空格函数：ltrim 25](#_Toc323327928)

[11. 右边去空格函数：rtrim 25](#_Toc323327929)

[12. 正则表达式替换函数：regexp\_replace 26](#_Toc323327930)

[13. 正则表达式解析函数：regexp\_extract 26](#_Toc323327931)

[14. URL解析函数：parse\_url 26](#_Toc323327932)

[15. json解析函数：get\_json\_object 27](#_Toc323327933)

[16. 空格字符串函数：space 27](#_Toc323327934)

[17. 重复字符串函数：repeat 27](#_Toc323327935)

[18. 首字符ascii函数：ascii 28](#_Toc323327936)

[19. 左补足函数：lpad 28](#_Toc323327937)

[20. 右补足函数：rpad 28](#_Toc323327938)

[21. 分割字符串函数: split 28](#_Toc323327939)

[22. 集合查找函数: find\_in\_set 29](#_Toc323327940)

[八、集合统计函数 29](#_Toc323327941)

[1. 个数统计函数: count 29](#_Toc323327942)

[2. 总和统计函数: sum 29](#_Toc323327943)

[3. 平均值统计函数: avg 30](#_Toc323327944)

[4. 最小值统计函数: min 30](#_Toc323327945)

[5. 最大值统计函数: max 30](#_Toc323327946)

[6. 非空集合总体变量函数: var\_pop 30](#_Toc323327947)

[7. 非空集合样本变量函数: var\_samp 31](#_Toc323327948)

[8. 总体标准偏离函数: stddev\_pop 31](#_Toc323327949)

[9. 样本标准偏离函数: stddev\_samp 31](#_Toc323327950)

[10．中位数函数: percentile 31](#_Toc323327951)

[11. 中位数函数: percentile 31](#_Toc323327952)

[12. 近似中位数函数: percentile\_approx 32](#_Toc323327953)

[13. 近似中位数函数: percentile\_approx 32](#_Toc323327954)

[14. 直方图: histogram\_numeric 32](#_Toc323327955)

[九、复合类型构建操作 32](#_Toc323327956)

[1. Map类型构建: map 32](#_Toc323327957)

[2. Struct类型构建: struct 33](#_Toc323327958)

[3. array类型构建: array 33](#_Toc323327959)

[十、复杂类型访问操作 33](#_Toc323327960)

[1. array类型访问: A[n] 33](#_Toc323327961)

[2. map类型访问: M[key] 34](#_Toc323327962)

[3. struct类型访问: S.x 34](#_Toc323327963)

[十一、复杂类型长度统计函数 34](#_Toc323327964)

[1. Map类型长度函数: size(Map<K.V>) 34](#_Toc323327965)

[2. array类型长度函数: size(Array<T>) 34](#_Toc323327966)

[3. 类型转换函数 35](#_Toc323327967)

十二、join,group,order,distribute,常用函数...........................................................................35

## 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

举例：

hive> select 1 from lxw\_dual where 1=1;

1

### 2. 不等值比较: <>

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 lxw\_dual where 1 <> 2;

1

### 3. 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 lxw\_dual where 1 < 2;

1

### 4. 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 lxw\_dual where 1 <= 1;

1

### 5. 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 lxw\_dual where 2 > 1;

1

### 6. 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 lxw\_dual 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 lxw\_dual;

OK

2011111209 00:00:00 2011111209

hive> select a,b,a<b,a>b,a=b from lxw\_dual;

2011111209 00:00:00 2011111209 false true false

### 7. 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 lxw\_dual where null is null;

1

### 8. 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 lxw\_dual where 1 is not null;

1

### 9. 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 lxw\_dual where 'football' like 'foot%';

1

hive> select 1 from lxw\_dual where 'football' like 'foot\_\_\_\_';

1

**注意：否定比较时候用NOT A LIKE B**

hive> select 1 from lxw\_dual where NOT 'football' like 'fff%';

1

### 10. 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 lxw\_dual 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 lxw\_dual where '123456' rlike '^\\d+$';

1

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

### 11. 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 lxw\_dual where 'footbar' REGEXP '^f.\*r$';

1

## 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，int + int 一般结果为int类型，而int + double 一般结果为double类型

举例：

hive> select 1 + 9 from lxw\_dual;

10

hive> create table lxw\_dual as select 1 + 1.2 from lxw\_dual;

hive> describe lxw\_dual;

\_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，int – int 一般结果为int类型，而int – double 一般结果为double类型

举例：

hive> select 10 – 5 from lxw\_dual;

5

hive> create table lxw\_dual as select 5.6 – 4 from lxw\_dual;

hive> describe lxw\_dual;

\_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 lxw\_dual;

200

### 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 lxw\_dual;

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(28.0/6.999999999999999999999) from lxw\_dual limit 1;

结果为4

hive>select ceil(28.0/6.99999999999999) from lxw\_dual 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 lxw\_dual;

1

hive> select 8.4 % 4 from lxw\_dual;

0.40000000000000036

**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 lxw\_dual;

0.4

### 6. 位与操作: &

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 lxw\_dual;

0

hive> select 6 & 4 from lxw\_dual;

4

### 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 lxw\_dual;

12

hive> select 6 | 8 from lxw\_dual;

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 lxw\_dual;

12

hive> select 6 ^ 4 from lxw\_dual;

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 lxw\_dual;

-7

hive> select ~4 from lxw\_dual;

-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 lxw\_dual 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 lxw\_dual where 1=2 or 2=2;

1

### 3. 逻辑非操作: NOT

语法: NOT 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 lxw\_dual where not 1=2;

1

## 四、数值计算

### 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 lxw\_dual;

3

hive> select round(3.5) from lxw\_dual;

4

hive> create table lxw\_dual as select round(9542.158) from lxw\_dual;

hive> describe lxw\_dual;

\_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 lxw\_dual;

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 lxw\_dual;

3

hive> select floor(25) from lxw\_dual;

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 lxw\_dual;

4

hive> select ceil(46) from lxw\_dual;

46

### 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 lxw\_dual;

4

hive> select ceiling(46) from lxw\_dual;

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 lxw\_dual;

0.5577432776034763

hive> select rand() from lxw\_dual;

0.6638336467363424

hive> select rand(100) from lxw\_dual;

0.7220096548596434

hive> select rand(100) from lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

2.0

### 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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

16.0

### 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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

11

hive> select hex(‘abc’) from lxw\_dual;

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 lxw\_dual;

abc

hive> select unhex(‘11’) from lxw\_dual;

-

hive> select unhex(616263) from lxw\_dual;

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 lxw\_dual;

11

hive> select conv(17,10,2) from lxw\_dual;

10001

### 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 lxw\_dual;

3.9

hive> select abs(10.9) from lxw\_dual;

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 lxw\_dual;

1

hive> select pmod(-9,4) from lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

-10

hive> select positive(12) from lxw\_dual;

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 lxw\_dual;

5

hive> select negative(8) from lxw\_dual;

-8

## 五、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 lxw\_dual;

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 lxw\_dual;

1323309615

### 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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

2011

hive> select year('2012-12-08') from lxw\_dual;

2012

### 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 lxw\_dual;

12

hive> select month('2011-08-08') from lxw\_dual;

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 lxw\_dual;

8

hive> select day('2011-12-24') from lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

3

### 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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

2012-11-28

## 六、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 lxw\_dual;

200

hive> select if(1=1,100,200) from lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

mary

hive> Select case 200 when 50 then 'tom' when 100 then 'mary' else 'tim' end from lxw\_dual;

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 lxw\_dual;

mary

hive> select case when 1=1 then 'tom' when 2=2 then 'mary' else 'tim' end from lxw\_dual;

tom

## 七、字符串函数

### 1. 字符串长度函数：length

语法: length(string A)

返回值: int

说明：返回字符串A的长度

举例：

hive> select length('abcedfg') from lxw\_dual;

7

### 2. 字符串反转函数：reverse

语法: reverse(string A)

返回值: string

说明：返回字符串A的反转结果

举例：

hive> select reverse(abcedfg’) from lxw\_dual;

gfdecba

### 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 lxw\_dual;

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 lxw\_dual;

abc,def,gh

### 5. 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 lxw\_dual;

cde

hive> select substring('abcde',3) from lxw\_dual;

cde

hive> select substr('abcde',-1) from lxw\_dual; （和ORACLE相同）

e

### 6. 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 lxw\_dual;

cd

hive> select substring('abcde',3,2) from lxw\_dual;

cd

hive>select substring('abcde',-2,2) from lxw\_dual;

de

### 7. 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 lxw\_dual;

ABSED

hive> select ucase('abSEd') from lxw\_dual;

ABSED

### 8. 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 lxw\_dual;

absed

hive> select lcase('abSEd') from lxw\_dual;

absed

### 9. 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 lxw\_dual;

abc

### 10. 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 lxw\_dual;

abc

### 11. 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 lxw\_dual;

abc

### 12. 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 lxw\_dual;

fb

### 13. 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 lxw\_dual;

the

hive> select regexp\_extract('foothebar', 'foo(.\*?)(bar)', 2) from lxw\_dual;

bar

hive> select regexp\_extract('foothebar', 'foo(.\*?)(bar)', 0) from lxw\_dual;

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;

### 14. 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 lxw\_dual;

facebook.com

hive> select parse\_url('http://facebook.com/path1/p.php?k1=v1&k2=v2#Ref1', 'QUERY', 'k1') from lxw\_dual;

v1

### 15. 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 lxw\_dual;

amy

### 16. 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 lxw\_dual;

hive> select length(space(10)) from lxw\_dual;

10

### 17. 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 lxw\_dual;

abcabcabcabcabc

### 18. 首字符ascii函数：ascii

语法: ascii(string str)

返回值: int

说明：返回字符串str第一个字符的ascii码

举例：

hive> select ascii('abcde') from lxw\_dual;

97

### 19. 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 lxw\_dual;

tdtdtdtabc

**注意：与GP，ORACLE不同，pad 不能默认**

### 20. 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 lxw\_dual;

abctdtdtdt

### 21. 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('abtcdtef','t') from lxw\_dual;

["ab","cd","ef"]

### 22. 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 lxw\_dual;

2

hive> select find\_in\_set('at','ef,ab,de') from lxw\_dual;

0

// select split('12304560789','0') as a from hmbbs limit 1 返回数组：["123","456","789"]

// concat\_ws(',',e.a[0],e.a[1],e.a[2])返回：123,456,789转换string

hive> select find\_in\_set('456',concat\_ws(',',e.a[0],e.a[1],e.a[2])) from (select split('12304560789','0') as a from hmbbs limit 1)e;

## 八、集合统计函数

### 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 lxw\_dual;

20

hive> select count(distinct t) from lxw\_dual;

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 lxw\_dual;

100

hive> select sum(distinct t) from lxw\_dual;

70

### 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 lxw\_dual;

50

hive> select avg (distinct t) from lxw\_dual;

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 lxw\_dual;

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 lxw\_dual;

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：

### 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 lxw\_dual； 取0.2，0.4位置的数据

### 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pth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 lxw\_dual;

[{"x":100.0,"y":1.0}]

## 九、复合类型构建操作

### 1. Map类型构建: map

语法: map (key1, value1, key2, value2, …)

说明：根据输入的key和value对构建map类型

举例：

hive> Create table lxw\_test as select map('100','tom','200','mary') as t from lxw\_dual;

hive> describe lxw\_test;

t map<string,string>

hive> select t from lxw\_test;

{"100":"tom","200":"mary"}

### 2. Struct类型构建: struct

语法: struct(val1, val2, val3, …)

说明：根据输入的参数构建结构体struct类型

举例：

hive> create table lxw\_test as select struct('tom','mary','tim') as t from lxw\_dual;

hive> describe lxw\_test;

t struct<col1:string,col2:string,col3:string>

hive> select t from lxw\_test;

{"col1":"tom","col2":"mary","col3":"tim"}

### 3. array类型构建: array

语法: array(val1, val2, …)

说明：根据输入的参数构建数组array类型

举例：

hive> create table lxw\_test as select array("tom","mary","tim") as t from lxw\_dual;

hive> describe lxw\_test;

t array<string>

hive> select t from lxw\_test;

["tom","mary","tim"]

## 十、复杂类型访问操作

### 1. array类型访问: A[n]

语法: A[n]

操作类型: A为array类型，n为int类型

说明：返回数组A中的第n个变量值。数组的起始下标为0。比如，A是个值为['foo', 'bar']的数组类型，那么A[0]将返回'foo',而A[1]将返回'bar'

举例：

hive> create table lxw\_test as select array("tom","mary","tim") as t from lxw\_dual;

hive> select t[0],t[1],t[2] from lxw\_test;

tom mary tim

### 2. map类型访问: M[key]

语法: M[key]

操作类型: M为map类型，key为map中的key值

说明：返回map类型M中，key值为指定值的value值。比如，M是值为{'f' -> 'foo', 'b' -> 'bar', 'all' -> 'foobar'}的map类型，那么M['all']将会返回'foobar'

举例：

hive> Create table lxw\_test as select map('100','tom','200','mary') as t from lxw\_dual;

hive> select t['200'],t['100'] from lxw\_test;

mary tom

### 3. struct类型访问: S.x

语法: S.x

操作类型: S为struct类型

说明：返回结构体S中的x字段。比如，对于结构体struct foobar {int foo, int bar}，foobar.foo返回结构体中的foo字段

举例：

hive> create table lxw\_test as select struct('tom','mary','tim') as t from lxw\_dual;

hive> describe lxw\_test;

t struct<col1:string,col2:string,col3:string>

hive> select t.col1,t.col3 from lxw\_test;

tom tim

## 十一、复杂类型长度统计函数

### Map类型长度函数: size(Map<K.V>)

语法: size(Map<K.V>)

返回值: int

说明: 返回map类型的长度

举例：

hive> select size(map('100','tom','101','mary')) from lxw\_dual;

2

### array类型长度函数: size(Array<T>)

语法: size(Array<T>)

返回值: int

说明: 返回array类型的长度

举例：

hive> select size(array('100','101','102','103')) from lxw\_dual;

4

### 类型转换函数

类型转换函数: cast

语法: cast(expr as <type>)

返回值: Expected "=" to follow "type"

说明: 返回array类型的长度

举例：

hive> select cast(1 as bigint) from lxw\_dual;

1

## 十二、join,group,order,distribute,常用函数

//查看表的详细属性

desc formatted TableName;

//创建常用的内部表(列以‘\t’分割)

create table TableName (ip string,time string,url string) row format delimited fields terminated by '\t';

//创建常用的外部表(列以‘\t’分割)

create external table TableName (ip string,time string,url string) row format delimited fields terminated by '\t';

//创建常用的外部分区表(列以‘\t’分割)

create external table TableName (ip string,time string,url string) partitioned by (dt string) row format delimited fields terminated by '\t' location '/hmbbs\_cleaned';

//为分区加载数据

alter table TableName add partition (dt='2014\_12\_29') location '/hmbbs\_cleaned/2014\_12\_28';

CREATE TABLE IF NOT EXISTS test01

(

id int ,

name STRING

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY "\t"

//内连接

select t1.id,t2.name

from

(select id,name from test01) t1

join

(select id,name from test02) t2

on t1.id=t2.id

//对多表连接的优化查询（不执行reduce操作）

select /\*+mapjoin(t1)\*/ t1.id,t2.name

from

(select id,name from test01) t1

join

(select id,name from test02) t2

on t1.id=t2.id

//左外链接

select t1.id,t2.name

from

(select id,name from test01) t1

left outer join

(select id,name from test02) t2

on t1.id=t2.id

//右外连接

select t1.id,t2.name

from

(select id,name from test01) t1

right outer join

(select id,name from test02) t2

on t1.id=t2.id

//分组

select name,count(1) from test01 group by name;

//排序

select \* from test01 order by name;

//将大文件打散成5个小文件(散列操作)

CREATE TABLE IF NOT EXISTS info

(

ip string,

atime string,

url string,

logdate string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY "\t"

//设置reduce的个数

set mapred.reduce.tasks = 5;

insert overwrite table info select ip,atime,url,logdate from hmbbs distribute by atime;

//将5个小文件组合成一个大文件(散列操作)

CREATE TABLE IF NOT EXISTS info2

(

ip string,

atime string,

url string,

logdate string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY "\t"

//设置reduce的个数

set mapred.reduce.tasks = 1;

insert overwrite table info2 select ip,atime,url,logdate from info distribute by atime;

//json

select get\_json\_object('{"name":"zhangsan","age":"18"}','$.name') from test02 limit 1;

执行结果：zhangsan

//获取url中的信息

select parse\_url('http://www.aboutyun.com/thread-7826-1-1.html','HOST') from test02 limit 1;

执行结果：www.aboutyun.com

//拼接函数

select concat(name,'123') from test02;

执行结果：XXX123

//通过反射机制调用类中的方法

select java\_method("java.lang.Math","sqrt",cast(id as double)) from test02;

//一行记录转多行，explode(split(name,'a'))：name字段根据a来拆分成数组

select id,name,sname from test02 lateral view explode(split(name,'a')) t1 as sname;

原纪录：

id name

1 zhan

2 c

3 d

执行结果：

id name sname

1 zhan zh

1 zhan n

2 c c

3 d d

//用户自定义函数（UDF）

1.编写相关的类

2.继承UDF类

3.重写evaluate方法（返回值，参数可以自定义）

例如：public boolean evaluate(Text t1, Text t2){

if (t1 == null || t2 == null) {

return false;

}

double d1 = Double.parseDouble(t1.toString());

double d2 = Double.parseDouble(t2.toString());

if (d1 > d2) {

return true;

} else {

return false;

}

}

4.打包jar

5.添加到hive; 命令：add jar /XX/XX/XXXXX.jar;

6.建立自定义的函数;命令：create temporary function FunctionName as 'com.udf.ClassName.FunctionName';

7.使用；例如：select id,name,testfunction01(id,3) from test01;

原始数据：

1 zhan

2 c

3 d

4 wangwu

5 h

6 lisi

8.效果：

1 zhan false

2 c false

3 d false

4 wangwu true

5 h true

6 lisi true

//初始化UDF方法

1.执行echo $HOME查看

2.在对应目录下创建.hiverc文件（不存在是要创建）；命令：touch .hiverc

3.使用vi编辑器编辑.hiverc；命令：vi .hiverc

4.加入添加jar命令；例如：add jar /XX/XX/XXXXX.jar;

5.加入建立自定义的函数;命令：create temporary function FunctionName as 'com.udf.ClassName.FunctionName';

6.保存退出；这样hive在开启是先会读取.hiverc文件，将jar与自定义函数添加到hive中

7.还有一种方法：修改hive的jar包后重新编译；位置在hive-0.9.0\src\ql\src\java\org\apache\hadoop\hive\ql\exec\ FunctionRegistry.java中注册

注：hive -i filename hive的初始化操作，

\*\*\*深入学习UDAF（自定义聚合函数）\*\*\*

//内部表转外部表

alter table TableName set tblproperties ('EXTERNAL'='TRUE');

//外部表转内部表

alter table TableName set tblproperties ('EXTERNAL'=FALSE');