

目录

一、关系运算:	4
1. 等值比较: =	4
2. 不等值比较: <>	4
3. 小于比较: <	4
4. 小于等于比较: <=	4
5. 大于比较: >	5
6. 大于等于比较: >=	5
7. 空值判断: IS NULL	5
8. 非空判断: IS NOT NULL	6
9. LIKE 比较: LIKE	6
10. JAVA 的 LIKE 操作: RLIKE	6
11. REGEXP 操作: REGEXP	7
二、数学运算:	7
1. 加法操作: +	7
2. 减法操作: -	7
3. 乘法操作: *	8
4. 除法操作: /	8
5. 取余操作: %	8
6. 位与操作: &	9
7. 位或操作:	9
8. 位异或操作: ^	9
9. 位取反操作: ~	10
三、逻辑运算:	10
1. 逻辑与操作: AND	10
2. 逻辑或操作: OR	10
3. 逻辑非操作: NOT	10
四、数值计算	11
1. 取整函数: round	11
2. 指定精度取整函数: round	11
3. 向下取整函数: floor	11
4. 向上取整函数: ceil	12
5. 向上取整函数: ceiling	12
6. 取随机数函数: rand	12
7. 自然指数函数: exp	13
8. 以 10 为底对数函数: log10	13
9. 以 2 为底对数函数: log2	13
10. 对数函数: log	13
11. 幂运算函数: pow	14
12. 幂运算函数: power	14
13. 开平方函数: sqrt	14
14. 二进制函数: bin	14

15. 十六进制函数: hex.....	15
16. 反转十六进制函数: unhex.....	15
17. 进制转换函数: conv.....	15
18. 绝对值函数: abs.....	16
19. 正取余函数: pmod.....	16
20. 正弦函数: sin.....	16
21. 反正弦函数: asin.....	16
22. 余弦函数: cos.....	17
23. 反余弦函数: acos.....	17
24. positive 函数: positive.....	17
25. negative 函数: negative.....	17
五、日期函数.....	18
1. UNIX 时间戳转日期函数: from_unixtime.....	18
2. 获取当前 UNIX 时间戳函数: unix_timestamp.....	18
3. 日期转 UNIX 时间戳函数: unix_timestamp.....	18
4. 指定格式日期转 UNIX 时间戳函数: unix_timestamp.....	18
5. 日期时间转日期函数: to_date.....	19
6. 日期转年函数: year.....	19
7. 日期转月函数: month.....	19
8. 日期转天函数: day.....	19
9. 日期转小时函数: hour.....	20
10. 日期转分钟函数: minute.....	20
11. 日期转秒函数: second.....	20
12. 日期转周函数: weekofyear.....	20
13. 日期比较函数: datediff.....	21
14. 日期增加函数: date_add.....	21
15. 日期减少函数: date_sub.....	21
六、条件函数.....	21
1. If 函数: if.....	21
2. 非空查找函数: COALESCE.....	22
3. 条件判断函数: CASE.....	22
4. 条件判断函数: CASE.....	22
七、字符串函数.....	23
1. 字符串长度函数: length.....	23
2. 字符串反转函数: reverse.....	23
3. 字符串连接函数: concat.....	23
4. 带分隔符字符串连接函数: concat_ws.....	23
5. 字符串截取函数: substr,substring.....	24
6. 字符串截取函数: substr,substring.....	24
7. 字符串转大写函数: upper,ucase.....	24
8. 字符串转小写函数: lower,lcase.....	25
9. 去空格函数: trim.....	25
10. 左边去空格函数: ltrim.....	25
11. 右边去空格函数: rtrim.....	25

12. 正则表达式替换函数: <code>regexp_replace</code>	26
13. 正则表达式解析函数: <code>regexp_extract</code>	26
14. URL 解析函数: <code>parse_url</code>	26
15. json 解析函数: <code>get_json_object</code>	27
16. 空格字符串函数: <code>space</code>	27
17. 重复字符串函数: <code>repeat</code>	27
18. 首字符 <code>ascii</code> 函数: <code>ascii</code>	28
19. 左补足函数: <code>lpad</code>	28
20. 右补足函数: <code>rpadd</code>	28
21. 分割字符串函数: <code>split</code>	28
22. 集合查找函数: <code>find_in_set</code>	29
八、集合统计函数	29
1. 个数统计函数: <code>count</code>	29
2. 总和统计函数: <code>sum</code>	29
3. 平均值统计函数: <code>avg</code>	30
4. 最小值统计函数: <code>min</code>	30
5. 最大值统计函数: <code>max</code>	30
6. 非空集合总体变量函数: <code>var_pop</code>	30
7. 非空集合样本变量函数: <code>var_samp</code>	31
8. 总体标准偏离函数: <code>stddev_pop</code>	31
9. 样本标准偏离函数: <code>stddev_samp</code>	31
10. 中位数函数: <code>percentile</code>	31
11. 中位数函数: <code>percentile</code>	31
12. 近似中位数函数: <code>percentile_approx</code>	32
13. 近似中位数函数: <code>percentile_approx</code>	32
14. 直方图: <code>histogram_numeric</code>	32
九、复合类型构建操作	32
1. Map 类型构建: <code>map</code>	32
2. Struct 类型构建: <code>struct</code>	33
3. array 类型构建: <code>array</code>	33
十、复杂类型访问操作	33
1. array 类型访问: <code>A[n]</code>	33
2. map 类型访问: <code>M[key]</code>	34
3. struct 类型访问: <code>S.x</code>	34
十一、复杂类型长度统计函数	34
1. Map 类型长度函数: <code>size(Map<K,V>)</code>	34
2. array 类型长度函数: <code>size(Array<T>)</code>	34
3. 类型转换函数	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;
```

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(28.0/6.99999999999999999999) from lxw_dual limit 1;
```

结果为 4

```
hive>select ceil(28.0/6.999999999999999) 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;
```

0.400000000000000036

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 lxw dual;
```


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;
```

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;
```

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;

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('abcdefg') 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(abcdefg') 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;
abcbcabcbcabcb
```

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: `rpadd`

语法: `rpadd(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 rpadd('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('abtcdef','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。如果没有找到该 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

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: max(col)

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 `p` 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

1. 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
```

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

3. 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