# Hive函数

## 1.内置运算符

1.1关系运算符

|  |  |  |
| --- | --- | --- |
| 运算符 | 类型 | 说明 |
| A = B | 所有原始类型 | 如果A与B相等,返回TRUE,否则返回FALSE |
| A == B | 无 | 失败，因为无效的语法。 SQL使用”=”，不使用”==”。 |
| A <> B | 所有原始类型 | 如果A不等于B返回TRUE,否则返回FALSE。如果A或B值为”NULL”，结果返回”NULL”。 |
| A < B | 所有原始类型 | 如果A小于B返回TRUE,否则返回FALSE。如果A或B值为”NULL”，结果返回”NULL”。 |
| A <= B | 所有原始类型 | 如果A小于等于B返回TRUE,否则返回FALSE。如果A或B值为”NULL”，结果返回”NULL”。 |
| A > B | 所有原始类型 | 如果A大于B返回TRUE,否则返回FALSE。如果A或B值为”NULL”，结果返回”NULL”。 |
| A >= B | 所有原始类型 | 如果A大于等于B返回TRUE,否则返回FALSE。如果A或B值为”NULL”，结果返回”NULL”。 |
| A IS NULL | 所有类型 | 如果A值为”NULL”，返回TRUE,否则返回FALSE |
| A IS NOT NULL | 所有类型 | 如果A值不为”NULL”，返回TRUE,否则返回FALSE |
| A LIKE B | 字符串 | 如 果A或B值为”NULL”，结果返回”NULL”。字符串A与B通过sql进行匹配，如果相符返回TRUE，不符返回FALSE。B字符串中 的”\_”代表任一字符，”%”则代表多个任意字符。例如： (‘foobar’ like ‘foo’)返回FALSE，（ ‘foobar’ like ‘foo\_ \_ \_’或者 ‘foobar’ like ‘foo%’)则返回TURE |
| A RLIKE B | 字符串 | 如 果A或B值为”NULL”，结果返回”NULL”。字符串A与B通过java进行匹配，如果相符返回TRUE，不符返回FALSE。例如：（ ‘foobar’ rlike ‘foo’）返回FALSE，（’foobar’ rlike ‘^f.\*r$’ ）返回TRUE。 |
| A REGEXP B | 字符串 | 与RLIKE相同。 |

1.2算术运算符

|  |  |  |
| --- | --- | --- |
| 运算符 | 类型 | 说明 |
| A + B | 所有数字类型 | A和B相加。结果的与操作数值有共同类型。例如每一个整数是一个浮点数，浮点数包含整数。所以，一个浮点数和一个整数相加结果也是一个浮点数。 |
| A – B | 所有数字类型 | A和B相减。结果的与操作数值有共同类型。 |
| A \* B | 所有数字类型 | A和B相乘，结果的与操作数值有共同类型。需要说明的是，如果乘法造成溢出，将选择更高的类型。 |
| A / B | 所有数字类型 | A和B相除，结果是一个double（双精度）类型的结果。 |
| A % B | 所有数字类型 | A除以B余数与操作数值有共同类型。 |
| A & B | 所有数字类型 | 运算符查看两个参数的二进制表示法的值，并执行按位”与”操作。两个表达式的一位均为1时，则结果的该位为 1。否则，结果的该位为 0。 |
| A|B | 所有数字类型 | 运算符查看两个参数的二进制表示法的值，并执行按位”或”操作。只要任一表达式的一位为 1，则结果的该位为 1。否则，结果的该位为 0。 |
| A ^ B | 所有数字类型 | 运算符查看两个参数的二进制表示法的值，并执行按位”异或”操作。当且仅当只有一个表达式的某位上为 1 时，结果的该位才为 1。否则结果的该位为 0。 |
| ~A | 所有数字类型 | 对一个表达式执行按位”非”（取反）。 |

1.3逻辑运算符

|  |  |  |
| --- | --- | --- |
| 运算符 | 类型 | 说明 |
| A AND B | 布尔值 | A和B同时正确时,返回TRUE,否则FALSE。如果A或B值为NULL，返回NULL。 |
| A && B | 布尔值 | 与”A AND B”相同 |
| A OR B | 布尔值 | A或B正确,或两者同时正确返返回TRUE,否则FALSE。如果A和B值同时为NULL，返回NULL。 |
| A | B | 布尔值 | 与”A OR B”相同 |
| NOT A | 布尔值 | 如果A为NULL或错误的时候返回TURE，否则返回FALSE。 |
| ! A | 布尔值 | 与”NOT A”相同 |

1.4复杂类型函数

|  |  |  |
| --- | --- | --- |
| 函数 | 类型 | 说明 |
| map | (key1, value1, key2, value2, …) | 通过指定的键/值对，创建一个map。 |
| struct | (val1, val2, val3, …) | 通过指定的字段值，创建一个结构。结构字段名称将COL1，COL2，… |
| array | (val1, val2, …) | 通过指定的元素，创建一个数组。 |

1.5对复杂类型函数操作

|  |  |  |
| --- | --- | --- |
| 函数 | 类型 | 说明 |
| A[n] | A是一个数组，n为int型 | 返回数组A的第n个元素，第一个元素的索引为0。如果A数组为['foo','bar']，则A[0]返回’foo’和A[1]返回”bar”。 |
| M[key] | M是Map<K, V>，关键K型 | 返回关键值对应的值，例如mapM为 \{‘f’ -> ‘foo’, ‘b’ -> ‘bar’, ‘all’ -> ‘foobar’\}，则M['all'] 返回’foobar’。 |
| S.x | S为struct | 返回结构x字符串在结构S中的存储位置。如 foobar \{int foo, int bar\} foobar.foo的领域中存储的整数。 |

## 2.内置函数

2.1数学函数

|  |  |  |
| --- | --- | --- |
| 返回类型 | 函数 | 说明 |
| BIGINT | round(double a) | 四舍五入 |
| DOUBLE | round(double a, int d) | 小数部分d位之后数字四舍五入，例如round(21.263,2),返回21.26 |
| BIGINT | floor(double a) | 对给定数据进行向下舍入最接近的整数。例如floor(21.2),返回21。 |
| BIGINT | ceil(double a), ceiling(double a) | 将参数向上舍入为最接近的整数。例如ceil(21.2),返回23. |
| double | rand(), rand(int seed) | 返回大于或等于0且小于1的平均分布随机数（依重新计算而变） |
| double | exp(double a) | 返回e的n次方 |
| double | ln(double a) | 返回给定数值的自然对数 |
| double | log10(double a) | 返回给定数值的以10为底自然对数 |
| double | log2(double a) | 返回给定数值的以2为底自然对数 |
| double | log(double base, double a) | 返回给定底数及指数返回自然对数 |
| double | pow(double a, double p) power(double a, double p) | 返回某数的乘幂 |
| double | sqrt(double a) | 返回数值的平方根 |
| string | bin(BIGINT a) | 返回二进制格式 |
| string | hex(BIGINT a) hex(string a) | 将整数或字符转换为十六进制格式 |
| string | unhex(string a) | 十六进制字符转换由数字表示的字符。 |
| string | conv(BIGINT num, int from\_base, int to\_base) | 将 指定数值，由原来的度量体系转换为指定的试题体系。例如CONV(‘a’,16,2),返回。参考：’1010′ http://dev.mysql.com/doc/refman/5.0/en/mathematical-functions.html#function\_conv |
| double | abs(double a) | 取绝对值 |
| int double | pmod(int a, int b) pmod(double a, double b) | 返回a除b的余数的绝对值 |
| double | sin(double a) | 返回给定角度的正弦值 |
| double | asin(double a) | 返回x的反正弦，即是X。如果X是在-1到1的正弦值，返回NULL。 |
| double | cos(double a) | 返回余弦 |
| double | acos(double a) | 返回X的反余弦，即余弦是X，，如果-1<= A <= 1，否则返回null. |
| int double | positive(int a) positive(double a) | 返回A的值，例如positive(2)，返回2。 |
| int double | negative(int a) negative(double a) | 返回A的相反数，例如negative(2),返回-2。 |

2.2收集函数

|  |  |  |
| --- | --- | --- |
| 返回类型 | 函数 | 说明 |
| int | size(Map<K.V>) | 返回的map类型的元素的数量 |
| int | size(Array<T>) | 返回数组类型的元素数量 |

2.3类型转换函数

|  |  |  |
| --- | --- | --- |
| 返回类型 | 函数 | 说明 |
| 指定 “type” | cast(expr as <type>) | 类型转换。例如将字符”1″转换为整数:cast(’1′ as bigint)，如果转换失败返回NULL。 |

2.4日期函数

|  |  |  |
| --- | --- | --- |
| 返回类型 | 函数 | 说明 |
| string | from\_unixtime(bigint unixtime[, string format]) | UNIX\_TIMESTAMP参数表示返回一个值’YYYY- MM – DD HH：MM：SS’或YYYYMMDDHHMMSS.uuuuuu格式，这取决于是否是在一个字符串或数字语境中使用的功能。该值表示在当前的时区。 |
| bigint | unix\_timestamp() | 如果不带参数的调用，返回一个Unix时间戳（从’1970- 01 – 0100:00:00′到现在的UTC秒数）为无符号整数。 |
| bigint | unix\_timestamp(string date) | 指定日期参数调用UNIX\_TIMESTAMP（），它返回参数值’1970- 01 – 0100:00:00′到指定日期的秒数。 |
| bigint | unix\_timestamp(string date, string pattern) | 指定时间输入格式，返回到1970年秒数：unix\_timestamp(’2009-03-20′, ‘yyyy-MM-dd’) = 1237532400 |
| string | to\_date(string timestamp) | 返回时间中的年月日： to\_date(“1970-01-01 00:00:00″) = “1970-01-01″ |
| string | to\_dates(string date) | 给定一个日期date，返回一个天数（0年以来的天数） |
| int | year(string date) | 返回指定时间的年份，范围在1000到9999，或为”零”日期的0。 |
| int | month(string date) | 返回指定时间的月份，范围为1至12月，或0一个月的一部分，如’0000-00-00′或’2008-00-00′的日期。 |
| int | day(string date) dayofmonth(date) | 返回指定时间的日期 |
| int | hour(string date) | 返回指定时间的小时，范围为0到23。 |
| int | minute(string date) | 返回指定时间的分钟，范围为0到59。 |
| int | second(string date) | 返回指定时间的秒，范围为0到59。 |
| int | weekofyear(string date) | 返回指定日期所在一年中的星期号，范围为0到53。 |
| int | datediff(string enddate, string startdate) | 两个时间参数的日期之差。 |
| int | date\_add(string startdate, int days) | 给定时间，在此基础上加上指定的时间段。 |
| int | date\_sub(string startdate, int days) | 给定时间，在此基础上减去指定的时间段。 |

2.5条件函数

|  |  |  |
| --- | --- | --- |
| 返回类型 | 函数 | 说明 |
| T | if(boolean testCondition, T valueTrue, T valueFalseOrNull) | 判断是否满足条件，如果满足返回一个值，如果不满足则返回另一个值。 |
| T | COALESCE(T v1, T v2, …) | 返回一组数据中，第一个不为NULL的值，如果均为NULL,返回NULL。 |
| T | CASE a WHEN b THEN c [WHEN d THEN e]\* [ELSE f] END | 当a=b时,返回c；当a=d时，返回e，否则返回f。 |
| T | CASE WHEN a THEN b [WHEN c THEN d]\* [ELSE e] END | 当值为a时返回b,当值为c时返回d。否则返回e。 |

2.6字符函数

|  |  |  |
| --- | --- | --- |
| 返回类型 | 函数 | 说明 |
| int | length(string A) | 返回字符串的长度 |
| string | reverse(string A) | 返回倒序字符串 |
| string | concat(string A, string B…) | 连接多个字符串，合并为一个字符串，可以接受任意数量的输入字符串 |
| string | concat\_ws(string SEP, string A, string B…) | 链接多个字符串，字符串之间以指定的分隔符分开。 |
| string | substr(string A, int start) substring(string A, int start) | 从文本字符串中指定的起始位置后的字符。 |
| string | substr(string A, int start, int len) substring(string A, int start, int len) | 从文本字符串中指定的位置指定长度的字符。 |
| string | upper(string A) ucase(string A) | 将文本字符串转换成字母全部大写形式 |
| string | lower(string A) lcase(string A) | 将文本字符串转换成字母全部小写形式 |
| string | trim(string A) | 删除字符串两端的空格，字符之间的空格保留 |
| string | ltrim(string A) | 删除字符串左边的空格，其他的空格保留 |
| string | rtrim(string A) | 删除字符串右边的空格，其他的空格保留 |
| string | regexp\_replace(string A, string B, string C) | 字符串A中的B字符被C字符替代 |
| string | regexp\_extract(string subject, string pattern, int index) | 通过下标返回正则表达式指定的部分。regexp\_extract(‘foothebar’, ‘foo(.\*?)(bar)’, 2) returns ‘bar.’ |
| string | parse\_url(string urlString, string partToExtract [, string keyToExtract]) | 返回URL指定的部分。parse\_url(‘http://facebook.com/path1/p.php?k1=v1&k2=v2#Ref1′, ‘HOST’) 返回：’facebook.com’ |
| string | get\_json\_object(string json\_string, string path) | select a.timestamp, get\_json\_object(a.appevents, ‘$.eventid’), get\_json\_object(a.appenvets, ‘$.eventname’) from log a; |
| string | space(int n) | 返回指定数量的空格 |
| string | repeat(string str, int n) | 重复N次字符串 |
| int | ascii(string str) | 返回字符串中首字符的数字值 |
| string | lpad(string str, int len, string pad) | 返回指定长度的字符串，给定字符串长度小于指定长度时，由指定字符从左侧填补。 |
| string | rpad(string str, int len, string pad) | 返回指定长度的字符串，给定字符串长度小于指定长度时，由指定字符从右侧填补。 |
| array | split(string str, string pat) | 将字符串转换为数组。 |
| int | find\_in\_set(string str, string strList) | 返回字符串str第一次在strlist出现的位置。如果任一参数为NULL,返回NULL；如果第一个参数包含逗号，返回0。 |
| array<array<string>> | sentences(string str, string lang, string locale) | 将字符串中内容按语句分组，每个单词间以逗号分隔，最后返回数组。 例如sentences(‘Hello there! How are you?’) 返回：( (“Hello”, “there”), (“How”, “are”, “you”) ) |
| array<struct<string,double>> | ngrams(array<array<string>>, int N, int K, int pf) | SELECT ngrams(sentences(lower(tweet)), 2, 100 [, 1000]) FROM twitter; |
| array<struct<string,double>> | context\_ngrams(array<array<string>>, array<string>, int K, int pf) | SELECT context\_ngrams(sentences(lower(tweet)), array(null,null), 100, [, 1000]) FROM twitter; |

## 3.内置的聚合函数（UDAF）

|  |  |  |
| --- | --- | --- |
| 返回类型 | 函数 | 说明 |
| bigint | count(\*) , count(expr), count(DISTINCT expr[, expr\_., expr\_.]) | 返回记录条数。 |
| double | sum(col), sum(DISTINCT col) | 求和 |
| double | avg(col), avg(DISTINCT col) | 求平均值 |
| double | min(col) | 返回指定列中最小值 |
| double | max(col) | 返回指定列中最大值 |
| double | var\_pop(col) | 返回指定列的方差 |
| double | var\_samp(col) | 返回指定列的样本方差 |
| double | stddev\_pop(col) | 返回指定列的偏差 |
| double | stddev\_samp(col) | 返回指定列的样本偏差 |
| double | covar\_pop(col1, col2) | 两列数值协方差 |
| double | covar\_samp(col1, col2) | 两列数值样本协方差 |
| double | corr(col1, col2) | 返回两列数值的相关系数 |
| double | percentile(col, p) | 返回数值区域的百分比数值点。0<=P<=1,否则返回NULL,不支持浮点型数值。 |
| array<double> | percentile(col, array(p~1,,\ [, p,,2,,]…)) | 返回数值区域的一组百分比值分别对应的数值点。0<=P<=1,否则返回NULL,不支持浮点型数值。 |
| double | percentile\_approx(col, p[, B]) | Returns an approximate p^th^ percentile of a numeric column (including floating point types) in the group. The B parameter controls approximation accuracy at the cost of memory. Higher values yield better approximations, and the default is 10,000. When the number of distinct values in col is smaller than B, this gives an exact percentile value. |
| array<double> | percentile\_approx(col, array(p~1,, [, p,,2\_]…) [, B]) | Same as above, but accepts and returns an array of percentile values instead of a single one. |
| array<struct\{‘x’,'y’\}> | histogram\_numeric(col, b) | Computes a histogram of a numeric column in the group using b non-uniformly spaced bins. The output is an array of size b of double-valued (x,y) coordinates that represent the bin centers and heights |
| array | collect\_set(col) | 返回无重复记录 |

## 4.内置表生成函数（UDTF）

|  |  |  |
| --- | --- | --- |
| 返回类型 | 函数 | 说明 |
| 数组 | explode(array<TYPE> a) | 数组一条记录中有多个参数，将参数拆分，每个参数生成一列。 |
|  | json\_tuple | get\_json\_object 语句：select a.timestamp, get\_json\_object(a.appevents, ‘$.eventid’), get\_json\_object(a.appenvets, ‘$.eventname’) from log a; json\_tuple语句: select a.timestamp, b.\* from log a lateral view json\_tuple(a.appevent, ‘eventid’, ‘eventname’) b as f1, f2 |

## 5.自定义函数

自定义函数包括三种UDF、UDAF、UDTF

UDF(User-Defined-Function) 一进一出

UDAF(User- Defined Aggregation Funcation) 聚集函数，多进一出。Count/max/min

UDTF(User-Defined Table-Generating Functions)&#160; 一进多出，如lateral view explore()

使用方式 ：在HIVE会话中add 自定义函数的jar文件，然后创建function继而使用函数

### ****5.1 UDF 开发****

1、UDF函数可以直接应用于select语句，对查询结构做格式化处理后，再输出内容。

2、编写UDF函数的时候需要注意一下几点：

a）自定义UDF需要继承org.apache.hadoop.hive.ql.UDF。

b）需要实现evaluate函数，evaluate函数支持重载。

3、步骤

a）把程序打包放到目标机器上去；

b）进入hive客户端，添加jar包：hive>add jar /run/jar/udf\_test.jar;

c）创建临时函数：hive>CREATE TEMPORARY FUNCTION add\_example AS 'hive.udf.Add';

d）查询HQL语句：

    SELECT add\_example(8, 9) FROM scores;

    SELECT add\_example(scores.math, scores.art) FROM scores;

    SELECT add\_example(6, 7, 8, 6.8) FROM scores;

e）销毁临时函数：hive> DROP TEMPORARY FUNCTION add\_example;

### ****5.2 UDAF 自定义集函数****

多行进一行出，如sum()、min()，用在group  by时

1.必须继承

org.apache.hadoop.hive.ql.exec.UDAF(函数类继承)

 org.apache.hadoop.hive.ql.exec.UDAFEvaluator(内部类Evaluator实现UDAFEvaluator接口)

2.Evaluator需要实现 init、iterate、terminatePartial、merge、terminate这几个函数

 init():类似于构造函数，用于UDAF的初始化

 iterate():接收传入的参数，并进行内部的轮转，返回boolean

 terminatePartial():无参数，其为iterate函数轮转结束后，返回轮转数据，类似于hadoop的Combiner

 merge():接收terminatePartial的返回结果，进行数据merge操作，其返回类型为boolean

 terminate():返回最终的聚集函数结果

开发一个功能同：

Oracle的wm\_concat()函数

Mysql的group\_concat()

Hive UDF的数据类型：

