Atitit 查询语句运算符 参考mongodb in http param

运算符是保留字或主要用于SQL语句的WHERE子句中的字符，用于执行操作，例如：比较和算术运算。 这些运算符用于指定SQL语句中的条件，并用作语句中多个条件的连词。常见运算符有以下几种 -

* 算术运算符
* 比较运算符
* 逻辑运算符
* 否定条件运算符

## 1. SQL算术运算符

假设变量a的值是：10，变量b的值是：20，则 -

| **操作符** | **描述** | **示例** |
| --- | --- | --- |
| + | 加法，执行加法运算。 | a + b = 30 |
| - | 减法，执行减法运算。 | a + b = -10 |
| \* | 除法，执行除法运算。 | a \* b = 200 |
| / | 用左操作数除右手操作数。 | b / a = 2 |
| % | 用左手操作数除左手操作数并返回余数。 | b % a = 0 |

[SQL算术运算符示例](https://www.yiibai.com/sql/sql-arithmetic-operators.html" \o "SQL算术运算符示例)

## 2. SQL比较运算符

变量a的值是：10，变量b的值是：20，则 -

=等于运算符

" username=joe&age= 27

大于小于

: {"$gte" : 18, "$lte" : 30}}) select \* from users where age >=18 and age <= 30

不等于 "$ne"

| **操作符** | **描述** | **示例** |
| --- | --- | --- |
| = | 检查两个操作数的值是否相等，如果是，则条件为真(true)。 | (a = b)结果为false。 |
| != | 检查两个操作数的值是否相等，如果值不相等则条件为真(true)。 | (a != b)结果为：true |
| <> | 检查两个操作数的值是否相等，如果值不相等则条件为真(true)。 | (a <> b)结果为：true |
| > | 检查左操作数的值是否大于右操作数的值，如果是，则条件为真(true)。 | (a > b)结果为：false |
| < | 检查左操作数的值是否小于右操作数的值，如果是，则条件为真(true)。 | (a < b)结果为：true |
| >= | 检查左操作数的值是否大于或等于右操作数的值，如果是，则条件为真(true)。 | (a >= b)结果为：false |
| <= | 检查左操作数的值是否小于或等于右操作数的值，如果是，则条件为真(true)。 | (a <= b)结果为：true |  |
| !< | 检查左操作数的值是否不小于右操作数的值，如果是，则条件变为真(true)。 | (a !< b)结果为：false |  |
| !> | 检查左操作数的值是否不大于右操作数的值，如果是，则条件变为真(true)。 | (a !> b)结果为：true |  |

[SQL比较运算符示例](https://www.yiibai.com/sql/sql-comparison-operators.html" \o "SQL比较运算符示例)

## 3. SQL逻辑运算符

以下是SQL中可用的所有逻辑运算符的列表。

In notion $nin" $in

And or "$or" : [{"ticket\_no" : 725}, {"winner" : true}]

{$all : ["apple", "banana"]}}) // 对数组的查询, 字段fruit中，既包含"apple",又包含"banana"的纪录 any 多行比较符(IN、ALL、ANY)

d({"$where" : "this.x + this.y == 10"}) //

****模糊查询简介****

MongoDB查询条件可以使用正则表达式，从而实现模糊查询的功能。模糊查询可以使用$regex操作符或直接使用正则表达式对象。

| **序号** | **操作符** | **描述** |
| --- | --- | --- |
| 1 | ALL | ALL运算符用于将值与另一个值集中的所有值进行比较。 |
| 2 | AND | AND运算符允许在SQL语句的WHERE子句中指定多个条件。 |
| 3 | ANY | ANY运算符用于根据条件将值与列表中的任何适用值进行比较。 |
| 4 | BETWEEN | BETWEEN运算符用于搜索在给定最小值和最大值内的值。 |
| 5 | EXISTS | EXISTS运算符用于搜索指定表中是否存在满足特定条件的行。 |
| 6 | IN | IN运算符用于将值与已指定的文字值列表进行比较。 |
| 7 | LIKE | LIKE运算符用于使用通配符运算符将值与类似值进行比较。 |
| 8 | NOT | NOT运算符反转使用它的逻辑运算符的含义。 例如：NOT EXISTS, NOT BETWEEN, NOT IN等等，这是一个否定运算符。 |
| 9 | OR | OR运算符用于组合SQL语句的WHERE子句中的多个条件。 |
| 10 | IS NULL | NULL运算符用于将值与NULL值进行比较。 |
| 11 | UNIQUE | UNIQUE运算符搜索指定表的每一行的唯一性(无重复项)。 |

//原文出自【易百教程】，商业转载请联系作者获得授权，非商业转载请保留原文链接：https://www.yiibai.com/sql/sql-operators.html

左边是mongodb查询语句，右边是sql语句。对照着用，挺方便。

db.users.find() select \* from users

db.users.find({"age" : 27}) select \* from users where age = 27

db.users.find({"username" : "joe", "age" : 27}) select \* from users where "username" = "joe" and age = 27

db.users.find({}, {"username" : 1, "email" : 1}) select username, email from users

db.users.find({}, {"username" : 1, "\_id" : 0}) // no case // 即时加上了列筛选，\_id也会返回；必须显式的阻止\_id返回

db.users.find({"age" : {"$gte" : 18, "$lte" : 30}}) select \* from users where age >=18 and age <= 30 // $lt(<) $lte(<=) $gt(>) $gte(>=)

db.users.find({"username" : {"$ne" : "joe"}}) select \* from users where username <> "joe"

db.users.find({"ticket\_no" : {"$in" : [725, 542, 390]}}) select \* from users where ticket\_no in (725, 542, 390)

db.users.find({"ticket\_no" : {"$nin" : [725, 542, 390]}}) select \* from users where ticket\_no not in (725, 542, 390)

db.users.find({"$or" : [{"ticket\_no" : 725}, {"winner" : true}]}) select \* form users where ticket\_no = 725 or winner = true

db.users.find({"id\_num" : {"$mod" : [5, 1]}}) select \* from users where (id\_num mod 5) = 1

db.users.find({"$not": {"age" : 27}}) select \* from users where not (age = 27)

db.users.find({"username" : {"$in" : [null], "$exists" : true}}) select \* from users where username is null // 如果直接通过find({"username" : null})进行查询，那么连带"没有username"的纪录一并筛选出来

db.users.find({"name" : /joey?/i}) // 正则查询，value是符合PCRE的表达式

db.food.find({fruit : {$all : ["apple", "banana"]}}) // 对数组的查询, 字段fruit中，既包含"apple",又包含"banana"的纪录

db.food.find({"fruit.2" : "peach"}) // 对数组的查询, 字段fruit中，第3个(从0开始)元素是peach的纪录

db.food.find({"fruit" : {"$size" : 3}}) // 对数组的查询, 查询数组元素个数是3的记录，$size前面无法和其他的操作符复合使用

db.users.findOne(criteria, {"comments" : {"$slice" : 10}}) // 对数组的查询，只返回数组comments中的前十条，还可以{"$slice" : -10}， {"$slice" : [23, 10]}; 分别返回最后10条，和中间10条

db.people.find({"name.first" : "Joe", "name.last" : "Schmoe"}) // 嵌套查询

db.blog.find({"comments" : {"$elemMatch" : {"author" : "joe", "score" : {"$gte" : 5}}}}) // 嵌套查询，仅当嵌套的元素是数组时使用,

db.foo.find({"$where" : "this.x + this.y == 10"}) // 复杂的查询，$where当然是非常方便的，但效率低下。对于复杂查询，考虑的顺序应当是 正则 -> MapReduce -> $where

db.foo.find({"$where" : "function() { return this.x + this.y == 10; }"}) // $where可以支持javascript函数作为查询条件

db.foo.find().sort({"x" : 1}).limit(1).skip(10); // 返回第(10, 11]条，按"x"进行排序; 三个limit的顺序是任意的，应该尽量避免skip中使用large-number

​