06丨数据过滤：SQL数据过滤都有哪些方法？

2019-06-24 陈旸

我在上篇文章中讲到过，提升查询效率的一个很重要的方式，就是约束返回结果的数量，还有一个很有效的方式，就是指定筛选条件，进行过滤。过滤可以筛选符合条件的结果，并进行返回，减少不必要的数据行。

那么在今天的内容里，我们来学习如何对SQL数据进行过滤，这里主要使用的就是WHERE子句。

你可能已经使用过WHERE子句，说起来SQL其实很简单，只要能把满足条件的内容筛选出来即可，但在实际使用过程中，不同人写出来的WHERE子句存在很大差别，比如执行效率的高低，有没有遇到莫名的报错等。

在今天的学习中，你重点需要掌握以下几方面的内容：

1. 学会使用WHERE子句，如何使用比较运算符对字段的数值进行比较筛选；
2. 如何使用逻辑运算符，进行多条件的过滤；
3. 学会使用通配符对数据条件进行复杂过滤。

比较运算符

在SQL中，我们可以使用WHERE子句对条件进行筛选，在此之前，你需要了解WHERE子句中的比较运算符。这些比较运算符的含义你可以参见下面这张表格：

含义	运算符
等于	=
不等于	<>或!=
小于	<
小于等于（大不于）	<=或!>
大于	>
大于等于（不小于）	>=或!<
不小于	!<
在指定的两个数值之间	BETWEEN
为空值	IS NULL

实际上你能看到，同样的含义可能会有多种表达方式，比如小于等于，可以是（<=），也可以是不大于（!>）。同样不等于，可以用（<>），也可以用（!=），它们的含义都是相同的，但这些符号的顺序都不能颠倒，比如你不能写（=<）。需要注意的是，你需要查看使用的DBMS是否支持，不同的DBMS支持的运算符可能是不同的，比如Access不支持（!=），不等于应该使用（<>）。在MySQL中，不支持（!>）（!<）等。

我在上一篇文章中使用了heros数据表，今天还是以这张表格做练习。下面我们通过比较运算符对王者荣耀的英雄属性进行条件筛选。

WHERE子句的基本格式是：SELECT ……(列名) FROM ……(表名) WHERE ……(子句条件)

比如我们想要查询所有最大生命值大于6000的英雄：

```
SQL：SELECT name, hp_max FROM heros WHERE hp_max > 6000
```

运行结果（41条记录）：

name	hp_max
夏侯惇	7350
钟无艳	7000
张飞	8341
.....	.....
凯	6700

想要查询所有最大生命值在5399到6811之间的英雄：

```
SQL：SELECT name, hp_max FROM heros WHERE hp_max BETWEEN 5399 AND 6811
```

运行结果：（41条记录）

name	hp_max
芈月	6164
雅典娜	6264
后裔	5986
.....	.....
百里守约	5611

需要注意的是hp_max可以取值到最小值和最大值，即5399和6811。

我们也可以对heros表中的hp_max字段进行空值检查。

```
SQL：SELECT name, hp_max FROM heros WHERE hp_max IS NULL
```

运行结果为空，说明heros表中的hp_max字段没有存在空值的数据行。

逻辑运算符

我刚才介绍了比较运算符，如果我们存在多个WHERE条件子句，可以使用逻辑运算符：

含义	逻辑运算符
并且	AND
或者	OR
在指定条件范围内	IN
非（否定）	NOT

我们还是通过例子来看下这些逻辑运算符的使用，同样采用heros这张表的数据查询。

假设想要筛选最大生命值大于6000，最大法力大于1700的英雄，然后按照最大生命值和最大法力值之和从高到低进行排序。

```
SQL：SELECT name, hp_max, mp_max FROM heros WHERE hp_max > 6000 AND mp_max > 1700 ORDER B
```

运行结果：（23条记录）

name	hp_max	mp_max
廉颇	9328	1708
牛魔	8476	1926
刘邦	8073	1940
……	……	……
孙尚香	6014	1756

如果AND和OR同时存在WHERE子句中会是怎样的呢？假设我们想要查询最大生命值加最大法力值大于8000的英雄，或者最大生命值大于6000并且最大法力值大于1700的英雄。

```
SQL：SELECT name, hp_max, mp_max FROM heros WHERE (hp_max+mp_max) > 8000 OR hp_max > 6000 A
```

运行结果：（33条记录）

name	hp_max	mp_max
廉颇	9328	1708
牛魔	8476	1926
白起	8638	1666
……	……	……
孙尚香	6014	1756

你能看出来相比于上一个条件查询，这次的条件查询多出来了10个英雄，这是因为我们放宽了条件，允许最大生命值+最大法力值大于8000的英雄显示出来。另外你需要注意到，当WHERE子句中同时存在OR和AND的时候，AND执行的优先级会更高，也就是说SQL会优先处理AND操作符，然后再处理OR操作符。

如果我们对这条查询语句OR两边的条件增加一个括号，结果会是怎样的呢？

```
SQL：SELECT name, hp_max, mp_max FROM heros WHERE ((hp_max+mp_max) > 8000 OR hp_max > 6000)
```

运行结果：

name	hp_max	mp_max
廉颇	9328	1708
牛魔	8476	1926
刘邦	8073	1940
……	……	……
孙尚香	6014	1756

所以当WHERE子句中同时出现AND和OR操作符的时候，你需要考虑到执行的先后顺序，也就是两个操作符执行的优先级。一般来说()优先级最高，其次优先级是AND，然后是OR。

如果我想要查询主要定位或者次要定位是法师或是射手的英雄，同时英雄的上线时间不在2016-01-01到2017-01-01之间。

```
SQL：
SELECT name, role_main, role_assist, hp_max, mp_max, birthdate
FROM heros
WHERE (role_main IN ('法师', '射手') OR role_assist IN ('法师', '射手'))
AND DATE(birthdate) NOT BETWEEN '2016-01-01' AND '2017-01-01'
ORDER BY (hp_max + mp_max) DESC
```

你能看到我把WHERE子句分成了两个部分。第一部分是关于主要定位和次要定位的条件过滤，使用的是role_main in ('法师', '射手') OR role_assist in ('法师', '射手')。这里用到了IN逻辑运算符，同时role_main和role_assist是OR（或）的关系。

第二部分是关于上线时间的条件过滤。NOT代表否，因为我们要找到不在2016-01-01到2017-01-01之间的日期，因此用到了NOT BETWEEN '2016-01-01' AND '2017-01-01'。同时我们是在对日期类型数据进行检索，所以使用到了DATE函数，将字段birthdate转化为日期类型再进行比较。关于日期的操作，我会在下一篇文章中再作具体介绍。

这是运行结果（6条记录）：

name	role_main	role_assist	hp_max	mp_max	birthdate
张良	法师		5799	1988	2015-10-26
貂蝉	法师	刺客	5611	1960	2015-12-15
干将莫邪	法师		5583	1946	2017-05-22
周瑜	法师		5513	1974	2015-11-10
百里守约	射手	刺客	5611	1784	2017-08-08
芈月	法师	坦克	6164	100	2015-12-08

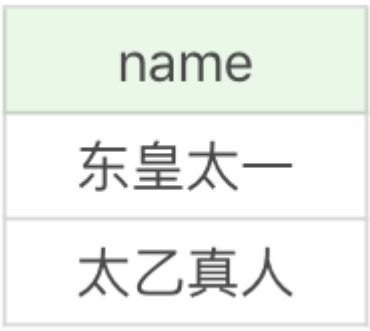
使用通配符进行过滤

刚才讲解的条件过滤都是对已知值进行的过滤，还有一种情况是我们要检索文本中包含某个词的所有数据，这里就需要使用通配符。通配符就是我们用来匹配值的一部分的特殊字符。这里我们需要使用到LIKE操作符。

如果我们想要匹配任意字符串出现的任意次数，需要使用（%）通配符。比如我们想要查找英雄名中包含“太”字的英雄都有哪些：

```
SQL：SELECT name FROM heros WHERE name LIKE '%太%'
```

运行结果：（2条记录）



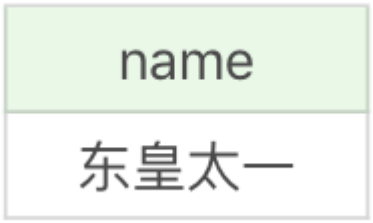
name
东皇太一
太乙真人

需要说明的是不同DBMS对通配符的定义不同，在Access中使用的是（*）而不是（%）。另外关于字符串的搜索可能是需要区分大小写的，比如'liu%'就不能匹配上'LIU BEI'。具体是否区分大小写还需要考虑不同的DBMS以及它们的配置。

如果我们想要匹配单个字符，就需要使用下划线(_)通配符。（%）和（_）的区别在于，（%）代表一个或多个字符，而（_）只代表一个字符。比如我们想要查找英雄名除了第一个字以外，包含“太”字的英雄有哪些。

```
SQL：SELECT name FROM heros WHERE name LIKE '_%太%'
```

运行结果（1条记录）：



name
东皇太一

因为太乙真人的太是第一个字符，而_%太%中的太不是在第一个字符，所以匹配不到“太乙真人”，只可以匹配上“东皇太一”。

同样需要说明的是，在Access中使用（?）来代替（_），而且在DB2中是不支持通配符（_）的，因此你需要在使用的时候查阅相关的DBMS文档。

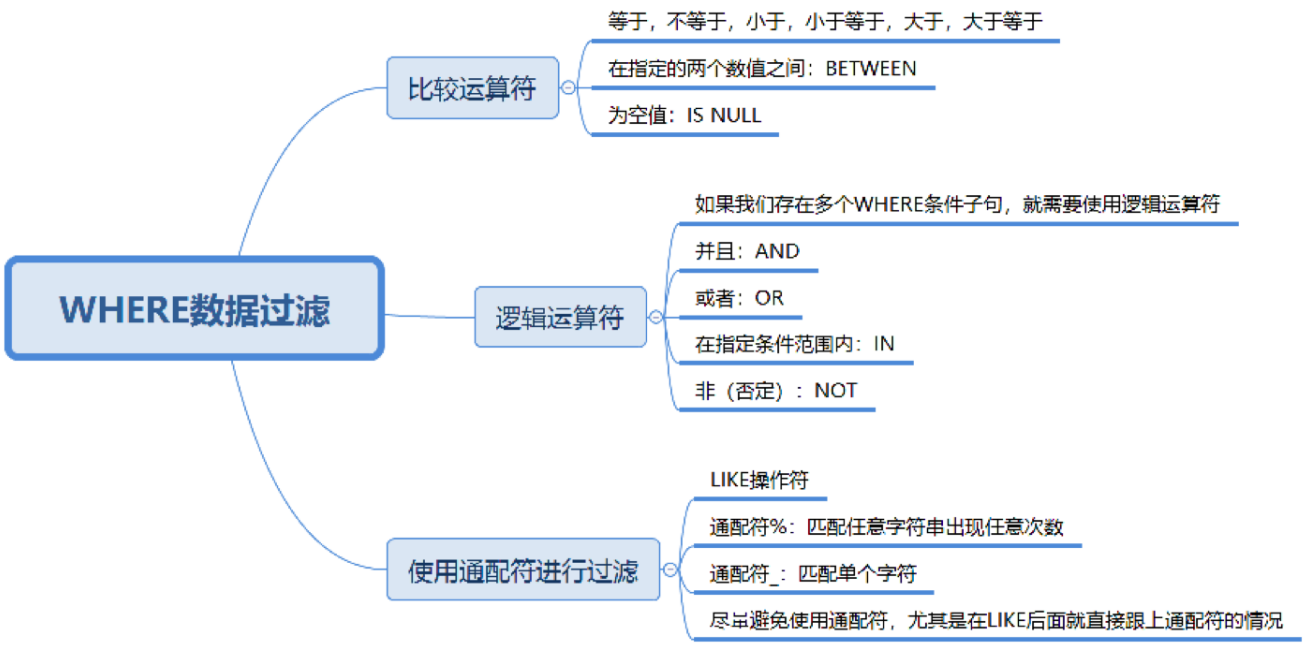
你能看出来通配符还是很有用的，尤其是在进行字符串匹配的时候。不过在实际操作过程中，我还是建议你尽量少用通配符，因为它需要消耗数据库更长的时间来进行匹配。即使你对LIKE检索的字段进行了索引，索引的价值也可能会失效。如果要让索引生效，那么LIKE后面就不能以（%）开头，比如使用LIKE '%太%'或LIKE '%太'的时候就会对全表进行扫描。如果使用LIKE '太%'，同时检索的字段进行了索引的时候，则不会进行全表扫描。

总结

今天我对SQL语句中的WHERE子句进行了讲解，你可以使用比较运算符、逻辑运算符和通配符这三种方式对检索条件进行过滤。

比较运算符是对数值进行比较，不同的DBMS支持的比较运算符可能不同，你需要事先查阅相应的DBMS文档。逻辑运算符可以让我们同时使用多个WHERE子句，你需要注意的是AND和OR运算符的执行顺序。通配符可以让我们对文本类型的字段进行模糊查询，不过检索的代价也是很高的，通常都需要用到全表扫描，所以效率很低。只有当LIKE语句后面不用通配符，并且对字段进行索引的时候才不会对全表进行扫描。

你可能认为学习SQL并不难，掌握这些语法就可以对数据进行筛选查询。但实际工作中不同人写的SQL语句的查询效率差别很大，保持高效率的一个很重要的原因，就是要避免全表扫描，所以我们会考虑在WHERE及ORDER BY涉及到的列上增加索引。



你能说一下WHERE子句中比较运算符、逻辑运算符和通配符这三者各自的作用吗？以heros数据表为例，请你编写SQL语句，对英雄名称、主要定位、次要定位、最大生命和最大法力进行查询，筛选条件为：主要定位是坦克或者战士，并且次要定位不为空，同时满足最大生命值大于8000或者最大法力小于1500的英雄，并且按照最大生命和最大法力之和从高到底的顺序进行排序。

欢迎你在评论区写下你的思考，也欢迎点击请朋友读，把这篇文章分享给你的朋友或者同事。

精选留言

Samson

15

老师，似乎有个问题：“（%）和（_）的区别在于，（%）代表一个或多个字符，而（_）只代表一个字符。”

%似乎是代表0个或任意正整数个字符，而不是一个或多个，因为如果是一个或多个的话，那么第一个例子中的太乙真人就匹配不到了。

2019-06-24

一步

13

就是要避免全表扫描，所以我们会考虑在 WHERE 及 ORDER BY 涉及到的列上增加索引

---

where 条件字段上加索引是可以明白的，但是为什么 order by 字段上还要加索引呢？这个时候已经通过 where条件过滤得到了数据，已经不需要在筛选过滤数据了，只需要在排序的时候根据字段排序就好了。不是很明白

2019-06-24

作者回复

这是一个很好的问题，关于ORDER BY字段是否增加索引：

在MySQL中，支持两种排序方式：FileSort和Index排序。Index排序的效率更高，

Index排序：索引可以保证数据的有序性，因此不需要再进行排序。

FileSort排序：一般在内存中进行排序，占用CPU较多。如果待排结果较大，会产生临时文件I/

O到磁盘进行排序，效率较低。

所以使用ORDER BY子句时，应该尽量使用Index排序，避免使用FileSort排序。

当然具体优化器是否采用索引进行排序，你可以使用explain来进行执行计划的查看。

优化建议：

1、SQL中，可以在WHERE子句和ORDER BY子句中使用索引，目的是在WHERE子句中避免全表扫描，ORDER BY子句避免使用FileSort排序。

当然，某些情况下全表扫描，或者FileSort排序不一定比索引慢。但总的来说，我们还是要避免，以提高查询效率。

一般情况下，优化器会帮我们进行更好的选择，当然我们也需要建立合理的索引。

2、尽量Using Index完成ORDER BY排序。

如果WHERE和ORDER BY相同列就使用单索引列；如果不同使用联合索引。

3、无法Using Index时，对FileSort方式进行调优。

2019-06-24

flow

6

关于通配符匹配里的 % 相当于正则表达式里的 .* 表示匹配大于等于0个任意字符，
所以 % 太 % 匹配的是 [大于等于0个任意字符]太[大于等于0个任意字符]，[东皇]太[一] 和 []太[乙真人]都符合；
而 _% 相当于正则表达式里的 .+ 表示匹配至少一个，即大于等于1个，
所以 '_% 太 % 匹配的是 [大于等于1个字符]太[大于等于0个字符]，只有 [东皇]太[一] 符合。

2019-06-24

作者回复

对的 解释正确

2019-06-24

一步

5

所以使用到了 DATE 函数，将字段 birthdate 转化为日期类型再进行比较

对于日期的比较，不是可以直接进行比较吗？ 对于上面的例子，直接可以使用 birthdate 字段进行时间比较，为什么还要使用DATE函数转换一下呢？

2019-06-24

陈扬鸿

3

老师，你好，现在mysql8已经没有frm文件，一旦数据字典丢失，没有表结构就无法恢复单个ibd文件的数据，如何通过mysql8的 sdi文件生成创建表的ddl语句。

2019-06-25

作者回复

在MySQL8.0版本之前，当我们ALTER TABLE时系统崩溃了，则会遗留.frm，.ibd文件。而在8.0版本之后，MySQL默认的InnoDB存储引擎实现了原子DDL。原子 DDL 操作写入了内部隐藏的系统表，即mysql.innodb_ddl_log中，也就是说明在DDL执行过程中如果出现了失败，是可以回滚的。

需要说明的是：DDL如果正常运行结束后，ddl_log中的相应日志也会被删除。如果这中间崩溃了，重启时会根据事务是否提交了来判断是做redo，还是undo DLL操作。

2019-06-25

悟空

3

老师关于通配符给的解释，不够清晰！
说明如下：
SQL：SELECT name FROM heros WHERE name LIKE '_% 太 %'

因为太乙真人的太是第一个字符，而_%太%中的太不是在第一个字符，所以匹配不到“太乙真人”，只可以匹配上“东皇太一”。

说明：
"_"：匹配任意一个字符，包括可以匹配到“太乙真人”的太字。
但是，整体的通配符 '_% 太 %'，需要后面继续匹配到一个"太"字符，显然，"太乙真人"不符合了，如果是，"太乙真人太太"，就可以匹配到。

2019-06-24

作者回复

哈哈 太乙真人太太 这个解释的好。也就是在第一个字符之后能匹配上“太”字。

2019-06-24

stormsc

2

作业 mysql: select name 英雄名称, role_main 主要定位, role_assist 次要定位,hp_max 最大生命值,mp_max 最大法力值 from heros where (role_main in ('坦克','战士')
AND role_assist is not null) AND (hp_max > 8000 or mp_max <1500) ORDER BY (hp_max+mp_max) DESC

2019-06-25

作者回复

正确，同时采用了列别名的方式。

2019-06-25

[illegible]

1

SELECT name, role_main, role_assist,hp_max,mp_max
FROM heros
WHERE (role_main IN ('坦克','战士') AND role_assist IS NOT NULL)
AND (hp_max > 8000 or mp_max < 1500)
ORDER BY (hp_max+mp_max) DESC;

2019-06-25

极客星星

1

你好 老师 不是很明白您说的对where语句建索引是什么意思 通过sql语句怎么实现
谢谢

2019-06-25

作者回复

多谢提问，这句话我说的比较省略。想表达的意思是，如果你使用了WHERE子句，对于某个字段进行了条件筛选，那么这个字段你可以通过建立索引的方式进行SQL优化。
因为我们在进行SQL优化的时候，应该尽量避免全表扫描。所以当我们使用WHERE子句对某个字段进行了条件筛选时，如果我们没有对这个字段建立索引，就会进入到全表扫描，因此可以考虑对这个字段建立索引。

当然你也需要注意 索引是否会失效。因此除了考虑建立字段索引以外，你还需要考虑：
1、不要在WHERE子句后面对字段做函数处理，同时也避免对索引字段进行数据类型转换
2、避免在索引字段上使用<>，!=，以及对字段进行NULL判断（包括 IS NULL, IS NOT NULL ）
3、在索引字段后，慎用IN和NOT IN，如果是连续的数值，可以考虑用BETWEEN进行替换
因为在WHERE子句中，如果对索引字段进行了函数处理，或者使用了<>,!=或NULL判断等，都会造成索引失效。

2019-06-25

stormsc 1

有个问题想问老师：
SELECT name,role_main,role_assist from heros where role_assist is not null LIMIT 5
这样限定的查询结果为5条数据，是随机选择的5条数据吗？

2019-06-25

作者回复

感谢提问，不是随机的5条。最简单的方式，你可以多重复几次，然后看下结果有没有变化。你会发现，每次运行的结果都是一样的，因此不是随机的。
如果想实现随机5条数据，可以采用下面的方式：
SELECT name,role_main,role_assist, RAND() as r FROM heros WHERE role_assist IS NOT NULL ORDER BY r LIMIT 5

2019-06-25

Krison 1

打卡，坚持学习

2019-06-24

野马 1

SELECT 英雄名称,主要定位,次要定位,最大生命,最大法力FROM heros WHERE （主要定位 IN ('坦克','战士')OR次要定位 IS NOT NULL）AND 最大生命>8000 AND 最大法力<1500 ORDER BY (最大生命+最大法力) DESC

2019-06-24

hlz-123 1

where子句WHERE 子句中比较运算符、逻辑运算符和通配符这三者各自作用？
1、比较运算符，比较数值的大小，数值类型可以是整数，浮点数，字符串，布尔类型等等。

2、逻辑运算符，定义where子句中多个条件之间的关系。

3、通配符，对文本类型字段进行模糊查询。

Mysql查询语句：

SELECT name,role_main,role_assist,hp_max,mp_max FROM heros
WHERE (role_main in ('坦克','战士') AND role_assist is not null)
AND (hp_max>8000 OR mp_max<1500) order by (hp_max+mp_max) DESC;

2019-06-24

作者回复

解释的很好，最后一个SQL查询也正确

2019-06-24

啦啦啦 1

打卡打卡

2019-06-24

mj4ever 0

SELECT name FROM heros WHERE name LIKE '_%太%'，最后一句似乎多了空格

2019-06-30

莫莫 0

SELECT `name`,role_main,role_assist,hp_max,mp_max FROM heros
WHERE (role_main in('坦克','战士') AND role_assist IS NOT NULL) AND (hp_max>8000 OR mp_max<1500)
ORDER BY (hp_max+mp_max) DESC

2019-06-30

华夏 0

SELECT name, role_main, role_assist, hp_max, mp_max
FROM heros
WHERE (role_main IN ('坦克', '战士') AND role_assist IS NOT NULL)
AND (hp_max > 8000 OR mp_max < 1500)
ORDER BY (hp_max+mp_max) DESC;

+----------+----------+------------+--------+--------+
| name | role_main | role_assist | hp_max | mp_max |
+----------+----------+------------+--------+--------+
牛魔	坦克	辅助	8476	1926
刘邦	坦克	辅助	8073	1940
程咬金	坦克	战士	8611	0
张飞	坦克	辅助	8341	100
亚瑟	战士	坦克	8050	0
吕布	战士	坦克	7344	0
关羽	战士	坦克	7107	10
花木兰	战士	刺客	5397	100

```
+----------+----------+------------+--------+--------+
```

8 rows in set (0.00 sec)

2019-06-30

H

0

是不是也需要优化下sql

2019-06-30

Neo

0

打卡：

SELECT name, role_main, role_assist, hp_max, mp_max FROM heros WHERE role_main IN('坦克','战士') AND role_assist IS NOT NULL AND (hp_max > 8000 OR mp_max < 1500) ORDE R BY (hp_max+mp_max) DESC;

2019-06-30

ttttt

0

对英雄名称、主要定位、次要定位、最大生命和最大法力进行查询，筛选条件为：主要定位是坦克或者战士，并且次要定位不为空，同时满足最大生命值大于 8000 或者最大法力小于 1500 的的英雄，并且按照最大生命和最大法力之和从高到底的顺序进行排序。

```
SELECT name, role_main, role_assist, hp_max, mp_max FROM heros
WHERE role_main IN ('坦克', '战士')
AND role_assist IS NOT NULL
AND (hp_max > 8000 OR mp_max < 1500)
ORDER BY (hp_max + mp_max) DESC;

+----------+----------+------------+--------+--------+
| name | role_main | role_assist | hp_max | mp_max |
+----------+----------+------------+--------+--------+
| 牛魔 | 坦克 | 辅助 | 8476 | 1926 |
| 刘邦 | 坦克 | 辅助 | 8073 | 1940 |
| 程咬金 | 坦克 | 战士 | 8611 | 0 |
| 张飞 | 坦克 | 辅助 | 8341 | 100 |
| 亚瑟 | 战士 | 坦克 | 8050 | 0 |
| 吕布 | 战士 | 坦克 | 7344 | 0 |
| 关羽 | 战士 | 坦克 | 7107 | 10 |
| 花木兰 | 战士 | 刺客 | 5397 | 100 |
+----------+----------+------------+--------+--------+
```

8 rows in set (0.00 sec)

2019-06-27

作者回复

结果正确

2019-06-28