为什么阿里巴巴禁止使用 count(列名)或 count(常量)来替代 count(*)

温柔的养猫人 2020-04-21 1789浏览量

简介： 本文介绍了COUNT函数的用法，主要用于统计表行数。主要用法有COUNT(*)、COUNT(字段)和COUNT(1)。

> 作者 | Hollis

数据库查询相信很多人都不陌生，所有经常有人调侃程序员就是CRUD专员，这所谓的CRUD指的就是数据库的增删改查。

在数据库的增删改查操作中，使用最频繁的就是查询操作。而在所有查询操作中，统计数量操作更是经常被用到。

关于数据库中行数统计，无论是MySQL还是Oracle，都有一个函数可以使用，那就是COUNT。

认识COUNT

关于COUNT函数，在MySQL官网中有详细介绍：

- `COUNT(expr) [over_clause]`

  Returns a count of the number of non-`NULL` values of *`expr`* in the rows retrieved by a `SELECT` statement. The result is a `BIGINT` value.

  If there are no matching rows, `COUNT()` returns 0.

  This function executes as a window function if *`over_clause`* is present. *`over_clause`* is as described in Section 12.21.2, "Window Function Concepts and Syntax".

```
mysql> SELECT student.student_name,COUNT(*)
       FROM student,course
       WHERE student.student_id=course.student_id
       GROUP BY student_name;
```

  `COUNT(*)` is somewhat different in that it returns a count of the number of rows retrieved, whether or not they contain `NULL` values.

简单翻译一下：

1、COUNT(expr) ，返回SELECT语句检索的行中expr的值不为NULL的数量。结果是一个BIGINT值。

2、如果查询结果没有命中任何记录，则返回0

3、但是，值得注意的是，`COUNT(*)` 的统计结果中，会包含值为NULL的行数。

即以下表记录

```
create table #bla(id int,id2 int)
insert #bla values(null,null)
```

```
insert #bla values(1,null)
insert #bla values(null,1)
insert #bla values(1,null)
insert #bla values(null,1)
insert #bla values(1,null)
insert #bla values(null,null)
```

使用语句count(*),count(id),count(id2)查询结果如下：

```
select count(*),count(id),count(id2)
from #bla
results 7 3 2
```

除了`COUNT(id)`和`COUNT(*)`以外，还可以使用`COUNT(常量)`（如`COUNT(1)`）来统计行数，那么这三条SQL语句有什么区别呢？到底哪种效率更高呢？为什么《阿里巴巴Java开发手册》中强制要求不让使用`COUNT(列名)`或`COUNT(常量)`来替代`COUNT(*)`呢？

1. 【强制】不要使用 count(列名)或 count(常量)来替代 count(*)，count(*)是 SQL92 定义的标准统计行数的语法，跟数据库无关，跟 NULL 和非 NULL 无关。
   说明：count(*)会统计值为 NULL 的行，而 count(列名)不会统计此列为 NULL 值的行。

COUNT(列名)、COUNT(常量)和COUNT(*)之间的区别

前面我们提到过`COUNT(expr)`用于做行数统计，统计的是expr不为NULL的行数，那么`COUNT(列名)`、`COUNT(常量)`和`COUNT(*)`这三种语法中，expr分别是`列名`、`常量`和`*`。

那么`列名`、`常量`和`*`这三个条件中，`常量`是一个固定值，肯定不为NULL。`*`可以理解为查询整行，所以肯定也不为NULL，那么就只有`列名`的查询结果有可能是NULL了。

所以，`COUNT(常量)`和`COUNT(*)`表示的是直接查询符合条件的数据库表的行数。而`COUNT(列名)`表示的是查询符合条件的列的值不为NULL的行数。

除了查询得到结果集有区别之外，`COUNT(*)`相比`COUNT(常量)`和`COUNT(列名)`来讲，`COUNT(*)`**是SQL92定义的标准统计行数的语法，因为他是标准语法，所以MySQL数据库对他进行过很多优化。**

> SQL92，是数据库的一个ANSI/ISO标准。它定义了一种语言（SQL）以及数据库的行为（事务、隔离级别等）。

COUNT(*)的优化

前面提到了`COUNT(*)`是SQL92定义的标准统计行数的语法，所以MySQL数据库对他进行过很多优化。那么，具体都做过哪些事情呢？

这里的介绍要区分不同的执行引擎。MySQL中比较常用的执行引擎就是InnoDB和MyISAM。

MyISAM和InnoDB有很多区别，其中有一个关键的区别和我们接下来要介绍的`COUNT(*)`有关，那就是**MyISAM不支持事务，MyISAM中的锁是表级锁；而InnoDB支持事务，并且支持行级锁。**

因为MyISAM的锁是表级锁，所以同一张表上面的操作需要串行进行，所以，**MyISAM做了一个简单的优化，那就是它可以把表的总行数单独记录下来，如果从一张表中使用COUNT(*)进行查询的时候，可以直接返回这个记录下来的数值就可以了，当然，前提是不能有where条件。**

MyISAM之所以可以把表中的总行数记录下来供COUNT(*)查询使用，那是因为MyISAM数据库是表级锁，不会有并发的数据库行数修改，所以查询得到的行数是准确的。

但是，对于InnoDB来说，就不能做这种缓存操作了，因为InnoDB支持事务，其中大部分操作都是行级锁，所以可能表的行数可能会被并发修改，那么缓存记录下来的总行数就不准确了。

但是，InnoDB还是针对COUNT(*)语句做了些优化的。

在InnoDB中，使用COUNT(*)查询行数的时候，不可避免的要进行扫表了，那么，就可以在扫表过程中下功夫来优化效率了。

从MySQL 8.0.13开始，针对InnoDB的`SELECT COUNT(*) FROM tbl_name`语句，确实在扫表的过程中做了一些优化。前提是查询语句中不包含WHERE或GROUP BY等条件。

我们知道，COUNT(*)的目的只是为了统计总行数，所以，他根本不关心自己查到的具体值，所以，他如果能够在扫表的过程中，选择一个成本较低的索引进行的话，那就可以大大节省时间。

我们知道，InnoDB中索引分为聚簇索引（主键索引）和非聚簇索引（非主键索引），聚簇索引的叶子节点中保存的是整行记录，而非聚簇索引的叶子节点中保存的是该行记录的主键的值。

所以，相比之下，非聚簇索引要比聚簇索引小很多，所以**MySQL会优先选择最小的非聚簇索引来扫表。所以，当我们建表的时候，除了主键索引以外，创建一个非主键索引还是有必要的。**

至此，我们介绍完了MySQL数据库对于COUNT(*)的优化，这些优化的前提都是查询语句中不包含WHERE以及GROUP BY条件。

COUNT(*)和COUNT(1)

介绍完了`COUNT(*)`，接下来看看`COUNT(1)`，对于，这二者到底有没有区别，网上的说法众说纷纭。

有的说`COUNT(*)`执行时会转换成`COUNT(1)`，所以COUNT(1)少了转换步骤，所以更快。

还有的说，因为MySQL针对`COUNT(*)`做了特殊优化，所以`COUNT(*)`更快。

那么，到底哪种说法是对的呢？看下MySQL官方文档是怎么说的：

> InnoDB handles SELECT COUNT(*) and SELECT COUNT(1) operations in the same way. There is no performance difference.

画重点：`same way`，`no performance difference`。**所以，对于COUNT(1)和COUNT(*)，MySQL的优化是完全一样的，根本不存在谁比谁快！**

那既然`COUNT(*)`和`COUNT(1)`一样，建议用哪个呢?

建议使用`COUNT(*)`！因为这个是SQL92定义的标准统计行数的语法，而且本文只是基于MySQL做了分析，关于Oracle中的这个问题，也是众说纷纭的呢。

COUNT(字段)

最后，就是我们一直还没提到的COUNT(字段)，他的查询就比较简单粗暴了，就是进行全表扫描，然后判断指定字段的值是不是为NULL，不为NULL则累加。

相比`COUNT(*)`，`COUNT(字段)`多了一个步骤就是判断所查询的字段是否为NULL，所以他的性能要比`COUNT(*)`慢。

总结

本文介绍了COUNT函数的用法，主要用于统计表行数。主要用法有`COUNT(*)`、`COUNT(字段)`和`COUNT(1)`。

因为`COUNT(*)`是SQL92定义的标准统计行数的语法，所以MySQL对他进行了很多优化，MyISAM中会直接把表的总行数单独记录下来供`COUNT(*)`查询，而InnoDB则会在扫表的时候选择最小的索引来降低成本。当然，这些优化的前提都是没有进行where和group的条件查询。

在InnoDB中`COUNT(*)`和`COUNT(1)`实现上没有区别，而且效率一样，但是`COUNT(字段)`需要进行字段的非NULL判断，所以效率会低一些。

因为`COUNT(*)`是SQL92定义的标准统计行数的语法，并且效率高，所以请直接使用`COUNT(*)`查询表的行数！

参考资料：《极客时间——MySQL实战45讲》

> 来源 | HollisChuang's Blog

SQL　缓存　Oracle　关系型数据库　MySQL　Java　程序员　数据库　索引