MySQL 分页查询优化

作者：InkDP
原文链接：https://ld246.com/article/1603213585917
来源网站：链滴

前言

Mysql慢查询优化，一直是开发中不可避免的问题，当然面试的时候也是。

今天的面试中，面试的最后一道题：“如何提供分页查询”，我自信的写下 LIMIT，认为此题十拿九，面试官此后的问题为当 offset到一定数量的时候怎么优化，因为之前没有遇到过类似的问题，而且没有量特别大的分页，所以这个问题只能作罢。

复盘

回家后弄了个大概有快20W数据的表，实测一下，查询速度是否会因为 limit边大而边长。

本地 single_planet

```
1  SELECT * FROM buried_point ORDER BY id ASC LIMIT 0, 100;
2  SELECT * FROM buried_point ORDER BY id ASC LIMIT 190000, 100;
```

信息 结果 1 结果 2 解释 1 剖析 状态

sql	message
SELECT * FROM buried_point ORDER BY id ASC LIMIT 0, 100	OK, Time: 0.000000s
SELECT * FROM buried_point ORDER BY id ASC LIMIT 190000, 100	OK, Time: 0.056000s

原文链接：MySQL 分页查询优化

如上图所示，同样的查询条件下，因为 limit增大查询速度确实变慢了很多。

why?

对于limit子句 LIMIT [offset,] row_count，官网说明如下

- The offset specifies the offset of the first row to return. Theoffset of the first row is 0, not 1.
- The row_count specifies the maximum number of rows to return.
- 翻译一下就是：
- offset参数指定要返回的第一行的偏移量。第一行的偏移量为0，而不是1。
- count指定要返回的最大行数。
-

因为要偏移到 offset处，所以就要先扫描前 offset行，所以随着 limit边大，也就越来越慢。

解决

随意Google一下，就找到了两种解决办法，分别贴出对于SQL以作参考

```
1  SELECT * FROM buried_point ORDER BY id ASC LIMIT 190000, 100;
2  SELECT * FROM buried_point JOIN (SELECT id FROM buried_point ORDER BY id ASC LIMIT 190000, 100) as tmp using(id);
3  SELECT a.* FROM buried_point AS a, (SELECT id FROM buried_point ORDER BY id ASC LIMIT 190000, 100) b WHERE a.id = b.id
```

信息 结果 1 解释 1 结果 2 结果 3 剖析 状态

sql	message
SELECT * FROM buried_point ORDER BY id ASC LIMIT 190000, 100	OK, Time: 0.050000s
SELECT * FROM buried_point JOIN (SELECT id FROM buried_point ORDER BY id ASC LIMIT 190000, 100...	OK, Time: 0.035000s
SELECT a.* FROM buried_point AS a, (SELECT id FROM buried_point ORDER BY id ASC LIMIT 190000, 1...	OK, Time: 0.034000s

因为数据量小，然后数据没有特意去设计，所以整体来说效果一般，但是还是有所提升

思考

针对limit的优化，更多的应该是让limit去尽量少的偏移数据，具体步骤如下：

- 使用索引列或者主键作为 order by操作列
- 记录上次查询的主键，作为下次查询时主键的筛选条件

```sql
SELECT * FROM buried_point ORDER BY id ASC LIMIT 190000, 100;
SELECT * FROM buried_point WHERE id > 191712 LIMIT 100; -- 19172为上一次查询后组件
```

信息 结果 1 结果 2 解释 1 剖析 状态

sql	message
SELECT * FROM buried_point ORDER BY id ASC LIMIT 190000, 100	OK, Time: 0.049000s
SELECT * FROM buried_point WHERE id > 191712 LIMIT 100	OK, Time: 0.000000s

参考：性能优化之分页查询