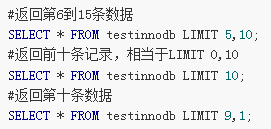
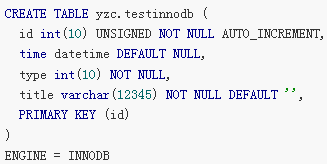
MySQL分页操作及其优化

一、LIMIT的基本用法

在我们使用查询语句的时候，经常要返回前几条或者中间某几行数据，这个时候我们就需要用到mysql为我们提供的分页功能了。分页在MySQL中使用的是LIMIT关键字，其用法如下：

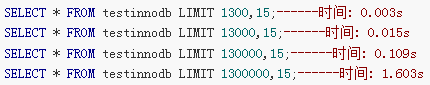
 LIMIT 子句可以被用于强制 SELECT 语句返回指定的记录数。日常使用中，其格式一般为：limit 当前页码\*页面容量-1 , 页面容量。LIMIT 接受一个或两个数字参数。参数必须是一个整数常量，不能是负数。如果给定两个参数，第一个参数指定第一个返回记录行的偏移量，第二个参数指定返回记录行的最大数目。初始记录行的偏移量是 0(而不是 1)。

 上面就是MySQL中LIMIT的基本用法，下面分析一下其在实际使用中的效率。测试表为testinnodb，建表语句如下：



向其中插入150W条数据。

首先测试从其中取出15条数据，偏移量分别设为0.13W、1.3W、13W、130W后其时间分别为多少



多次运行，其平均时间大致如上，通过以上的测试，可以发现LIMIT语句在偏移量比较小的时候效率比较好，但是当偏移量比较大的时候，其性能就会急剧下降。

对于LIMIT M,N来说，其中M为偏移量，N为需要取的数据的行数，其意思是扫描前面的M+N行，然后扔掉前M行，返回剩下的N行，这在一个高并发的应用里，如果M很大，每次查询需要扫描超过数万行，性能肯定大打折扣。

查看上面任一语句的执行计划，如下：

可以很明显的发现，上面的所有语句都是全表扫描，并且未使用任何的索引

# 二、常见的分页优化方法

还是对testinnodb表测试，假设每页数据为500条，总共有3000页，查询其中的第100页的下一页，也就是ID为50001-50500的记录。下面语句的执行时间都是在数据库中多次运行取时间的平均值。

## 方法一: 直接使用数据库提供的SQL语句

语句样式: MySQL中,可用如下方法:

 **适应场景:** 适用于数据量较少的情况(元组百/千级)   
 **缺点:** 全表扫描,速度会很慢。   
 **执行计划：**



## **方法二: 基于索引再排序** **语句样式:** MySQL中,可用如下方法:

适应场景: 适用于数据量多的情况(元组数上万). ORDER BY后的列对象是主键或唯一，所以,使得ORDER BY操作能利用索引被消除

原因: 索引扫描,速度会很快. 但MySQL的排序操作,只有ASC没有DESC(DESC是假的,未来会做真正的DESC).

执行计划：



## 方法三：MySQL自带的BETWEEN…AND… 语句查询优化

执行计划：



## 方法四： 如果需要查询 id 不是连续的一段，最佳的方法就是先找出 id ，然后用 in 查询

原理： MySQL 的索引对于in语句同样是有效的



## 方法五: 利用”子查询/JOIN+索引”快速定位元组的位置,然后再读取元组

原理：先找出第一条数据，然后大于等于这条数据的id的就是要获取的数据，减少了无用的IO操作，只对需要的数据执行SELECT \*操作

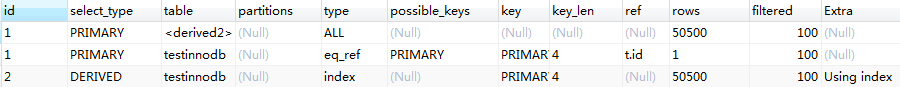
缺点：数据必须是连续的，否则可能需要先查询第一行所在的位置

**子查询：**

执行计划如下：

**JOIN联接两个表:**

执行计划：



## 方法六：存储过程类

原因: 存储过程中的SQL都是预先编译过的，这样子把操作封装在服务器，相对更快一些。

## 方法七: 反向查找优化法

原理：当偏移超过一半记录数的时候，先用排序，这样偏移就反转了

LIMIT偏移算法：

正向查找： (当前页 - 1) \* 页长度

反向查找： 总记录 - 当前页 \* 页长度

缺点： ORDER BY优化比较麻烦，要增加索引，索引影响数据的修改效率，并且要知道总记录数 ，偏移大于数据的一半

实例：

查询其中的第2001页的下一页，也就是ID为1000001-1000500的记录。

直接查找：

反向查找：

**三、包含排序操作的分页需求**

**解决思路： 建立合适的索引，先找出对应的主键，在根据主键查找需要的数据，减少不必要的IO操作。**

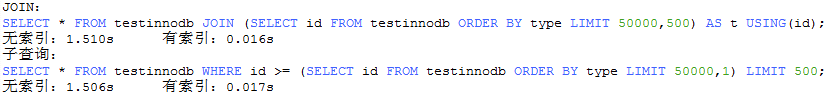
**假设现在的查找需求为，跟据type升序排序，查找第100页。**

**创建的索引为：**

 **直接查找：**

直接操作时，在有无索引来那个中条件下，虽然有优化，但是好像还是不太理想。

借鉴上面的方法，继续优化：

 现在的执行效率就可以达到我们的需求了。

四、**实际工作中对于LIMIT的需求**

**实际编码的过程中，我们经常会遇到类似于：“上一页 1 2 3 4 5 6 7 8 9 … 下一页”这样的翻页需求。这个时候我们又应该如何优化查询呢？**

**我们已经知道limit 10000,500的意思扫描满足条件的10500行，扔掉前面的10000行，返回最后的500行，大量的时间都浪费在前面的10000行上。所以对于上面的需求，如果LIMIT M,N不可避免的话，要优化效率，只有尽可能的让M小一些。**

**还是对testinnodb表测试，假设每页数据为500条，总共有3000页，假设当前是第10页，当前页的最大ID是5000，最小ID是4501。**

**当前的第10页的SQL如下：**

**比如要跳到第9页，SQL语句可以这样写：**

比如要跳到第8页，SQL语句可以这样写：

 跳转到第11页：

跳转到第12页：



原理还是一样，记录住当前页id的最大值和最小值，计算跳转页面和当前页相对偏移，由于页面相近，这个偏移量不会很大，这样的话M值相对较小，大大减少扫描的行数。其实传统的LIMIT M，N，相对的偏移一直是第一页，这样的话越翻到后面，效率越差，而上面给出的方法就没有这样的问题。