SQLServer索引结构及其使用之六 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/143/2021_2022_SQLServer_E7_c102_143008.htm 即，用not exists来代替not in，但我们前面已经谈过了，二者的执行效率实际上是没有区别的。既便如此，用TOP 结合NOT IN的这个方法还是比用游标要来得快一些。 虽然用not exists并不能挽救上个存储过程的效率，但使用SQL SERVER中的TOP关键字却是一个非常明智的选择。因为分页优化的最终目的就是避免产生过大的记录集，而我们在前面也已经提到了TOP的优势，通过TOP 即可实现对数据量的控制。 在分页算法中，影响我们查询速度的关键因素有两点：TOP和NOT IN。TOP可以提高我们的查询速度，而NOT IN会减慢我们的查询速度，所以要提高我们整个分页算法的速度，就要彻底改造NOT IN，同其他方法来替代它。我们知道，几乎任何字段，我们都可以通过max(字段)或min(字段)来提取某个字段中的最大或最小值，所以如果这个字段不重复，那么就可以利用这些不重复的字段的max或min作为分水岭，使其成为分页算法中分开每页的参照物。在这里，我们可以用操作符“ > ”或“ Select top 10 * from table1 where id>200于是就有了如下分页方案：0select top 页大小 *from table1 where id>(0select max (id) from (0select top ((页码-1)*页大小) id from table1 order by id) as T) order by id 在选择即不重复值，又容易分辨大小的列时，我们通常会选择主键。下表列出了笔者用有着1000万数据的办公自动化系统中的表，在以GID（GID是主键，但并不是聚集索引。）为排序列、提取gid,fariqi,title字段，分别以第1、10、100、500、1000、1万

、10万、25万、50万页为例，测试以上三种分页方案的执行速度：（单位：毫秒） 从上表中，我们可以看出，三种存储过程在执行100页以下的分页命令时，都是可以信任的，速度都很好。但第一种方案在执行分页1000页以上后，速度就降了下来。第二种方案大约是在执行分页1万页以上后速度开始降了下来。而第三种方案却始终没有大的降势，后劲仍然很足。 在确定了第三种分页方案后，我们可以据此写一个存储过程。大家知道SQL SERVER的存储过程是事先编译好的SQL语句，它的执行效率要比通过WEB页面传来的SQL语句的执行效率要高。下面的存储过程不仅含有分页方案，还会根据页面传来的参数来确定是否进行数据总数统计。--获取指定页的数据：CREATE PROCEDURE pagination3@tblName varchar(255), -- 表名@strGetFields varchar(1000) = *, -- 需要返回的列 @fldName varchar(255)=, -- 排序的字段名@PageSize int = 10, -- 页尺寸@PageIndex int = 1, -- 页码@doCount bit = 0, -- 返回记录总数, 非 0 值则返回@OrderType bit = 0, -- 设置排序类型, 非 0 值则降序@strWhere varchar(1500) = -- 查询条件 (注意: 不要加 where)AS 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com