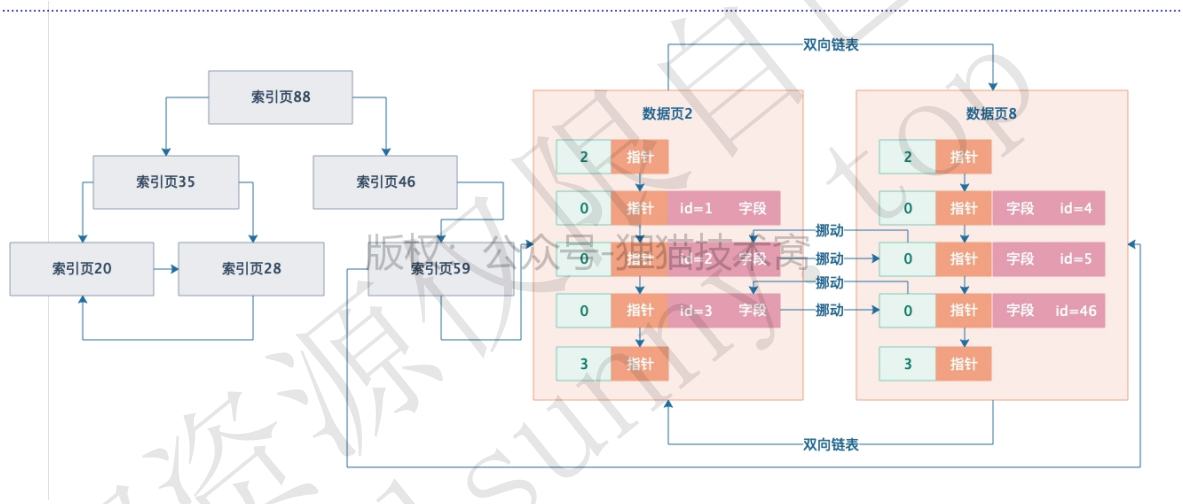
72 一个表里是不是索引搞的越多越好？那你就大错特错了！

一个表里是不是索引搞的越多越好？那你就大错特错了！

今天我们来稍微停一下脚步，做一个简单的关于索引知识的总结，然后再给大家分析一下索引的优点和缺点。

首先呢，我们都知道，正常我们在一个表里灌入数据的时候，都会基于主键给我们自动建立聚簇索引，这个聚簇索引大概看起来就是下面的样子。



随着我们不停的在表里插入数据，他就会不停的在数据页里插入数据，然后一个数据页放满了就会分裂成多个数据页，这个时候就需要索引页去指向各个数据页

然后如果数据页太多了，那么索引页里里的数据页指针也就会太多了，索引页也必然会放满的，此时索引页也会分裂成多个，再形成更上层的索引页。

最后这么逐步的演化下来，你就会看到上面那个图了！这个过程我们之前都详细分析过了，相信大家看一下文字说明就知道怎么回事！

默认情况下MySQL给我们建立的聚簇索引都是基于主键的值来组织索引的，聚簇索引的叶子节点都是数据页，里面放的就是我们插入的一行一行的完整的数据了！

在一个索引B+树中，他有一些特性，那就是数据页/索引页里面的记录都是组成一个单向链表的，而且是按照数据大小有序排列的；然后数据页/索引页互相之间都是组成双向链表的，而且也都是按照数据大小有序排列的，所以其实B+树索引是一个完全有序的数据结构，无论是页内还是页之间。

正是因为这个有序的B+树索引结构，才能让我们查找数据的时候，直接从根节点开始按照数据值大小一层一层往下找，这个效率是非常高的。

然后如果是针对主键之外的字段建立索引的话，实际上本质就是为那个字段的值重新建立另外一颗B+树索引，那个索引B+树的叶子节点，存放的都是数据页，里面放的都是你字段的值和主键值，然后每一层索引页里存放的都是下层页的引用，包括页内的排序规则，页之间的排序规则，B+树索引的搜索规则，都是一样的。

但是唯一要清晰记住的一点是，假设我们要根据其他字段的索引来搜索，那么只能基于其他字段的索引B+树快速查找到那个值所对应的主键，接着再次做回表查询，基于主键在聚簇索引的B+树里，重新从根节点开始查找那个主键值，找到主键值对应的完整数据。

以上就是我们之前给大家分析过的完整的MySQL的B+树索引原理了，包括B+树索引的数据结构，排序规则，以及你插入的时候他形成的过程，基于B+树查询的原理，以及不同字段的索引是有独立B+树的和回表查询过程，就给大家完整总结好了。

那么今天我们就站在这个总结的基础之上，给大家最后提一个结论，你在MySQL的表里建立一些字段对应的索引，好处是什么？

好处显而易见了，你可以直接根据某个字段的索引B+树来查找数据，不需要全表搜索，性能提升是很高的。

但是坏处呢？索引当然有缺点了，主要是两个缺点，一个是空间上的，一个是时间上的。

空间上而言，你要是给很多字段创建很多的索引，那你必须会有很多棵索引B+树，每一棵B+树都要占用很多的磁盘空间啊！所以你要是搞的索引太多了，是很耗费磁盘空间的。

其次，你要是搞了很多索引，那么你在进行增删改查的时候，每次都需要维护各个索引的数据有序性，因为每个索引B+树都要求页内是按照值大小排序的，页之间也是有序的，下一个页的所有值必须大于上一个页的所有值！

所以你不停的增删改查，必然会导致各个数据页之间的值大小可能会没有顺序，比如下一个数据页里插入了一个比较小的值，居然比上一个数据页的值要小！此时就没办法了，只能进行数据页的挪动，维护页之间的顺序。

或者是你不停的插入数据，各个索引的数据页就要不停的分裂，不停的增加新的索引页，这个过程都是耗费时间的。

所以你要是一个表里搞的索引太多了，很可能就会导致你的增删改的速度就比较差了，也许查询速度确实是可以提高，但是增删改就会受到影响，因此通常来说，我们是不建议一个表里搞的索引太多的！

那么怎么才能尽量用最少的索引满足最多的查询请求，还不至于让索引占用太多磁盘空间，影响增删改性能呢？这就需要我们深入理解索引的使用规则了，我们的SQL语句要怎么写，才能用上索引B+树来查询！

End