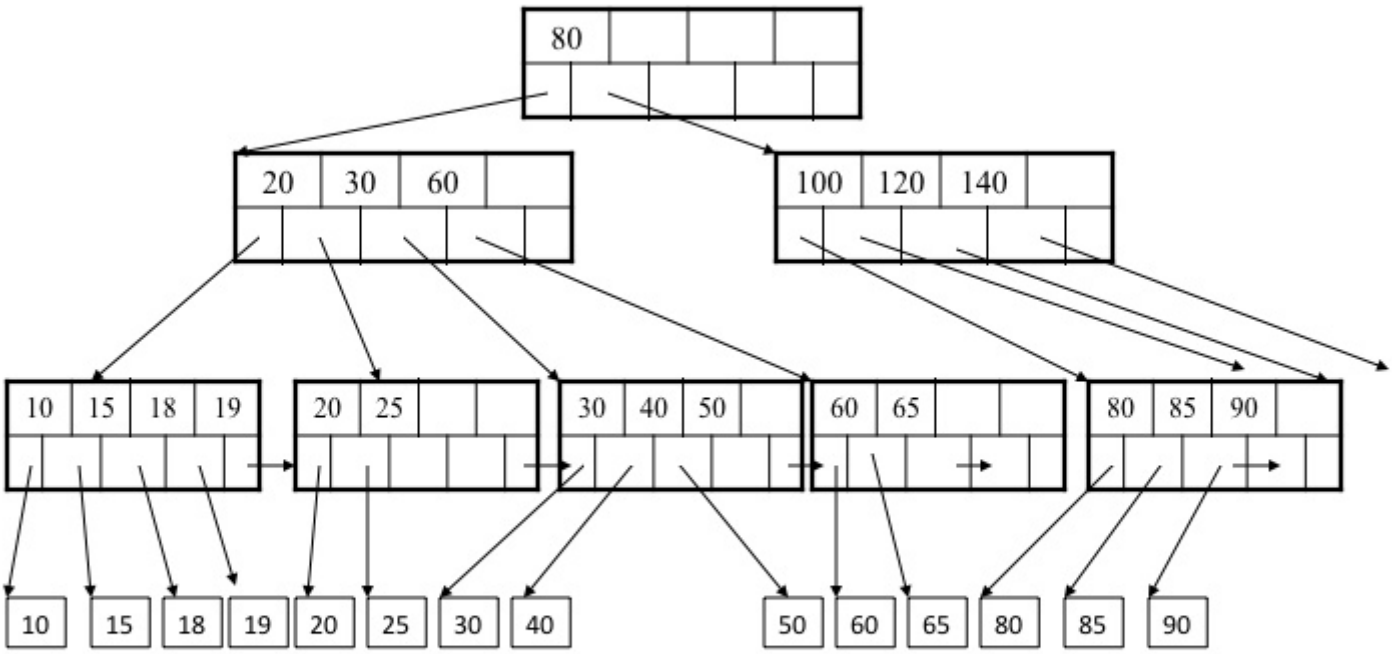
知乎



基于外存（磁盘）的B+树实现

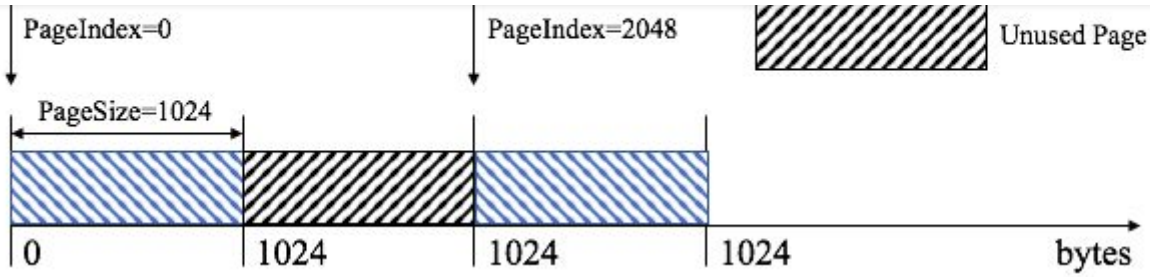
游凯超
清华软院，机器学习。提问请在知识星球APP搜索"答疑坊"

网上可以找到很多B+树的实现，但是大部分都是在内存中实现的。**整棵树都在内存中的B+树就是耍流氓**。不与文件打交道的B+树就失去了B+树的意义。

B+树可以看做是基于外存的多值的key-value pair（一个key可能对应多个value）。所以我们还需要存储多值的情况。在外存中，存储多个值可不是像内存中新建一个链表那么简单。

宏观存储结构

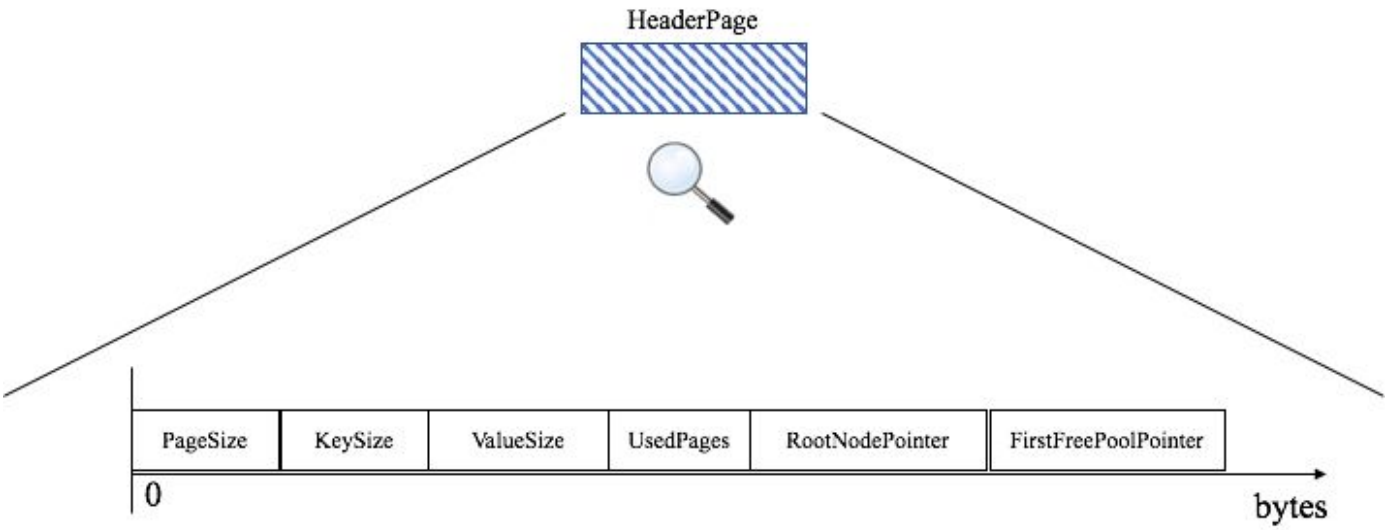
为了合理地存储B+树，文件需要被划分为以页块为单位的存储形式：



其中每个页块就是一个B+树的节点。

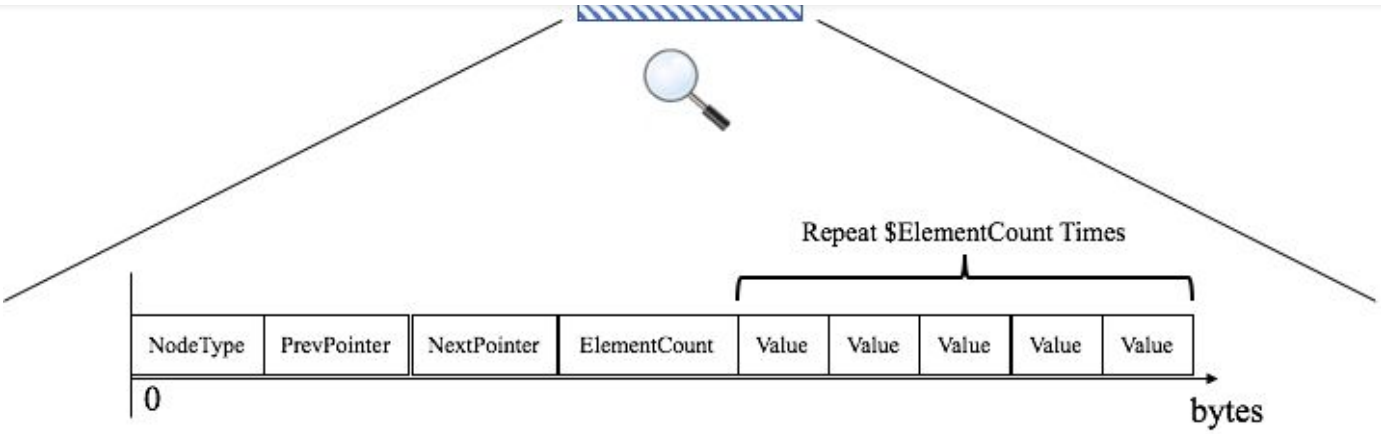
B+树的节点有很多种，下面分类型介绍一种可能的B+树各种节点的存储方法：

元信息节点（头节点）



头节点用于从磁盘中恢复B+树，它存储了B+树的元信息，包括页面大小、键值对大小、使用的总页数、根节点存储位置，首个空闲池节点存储位置等信息。

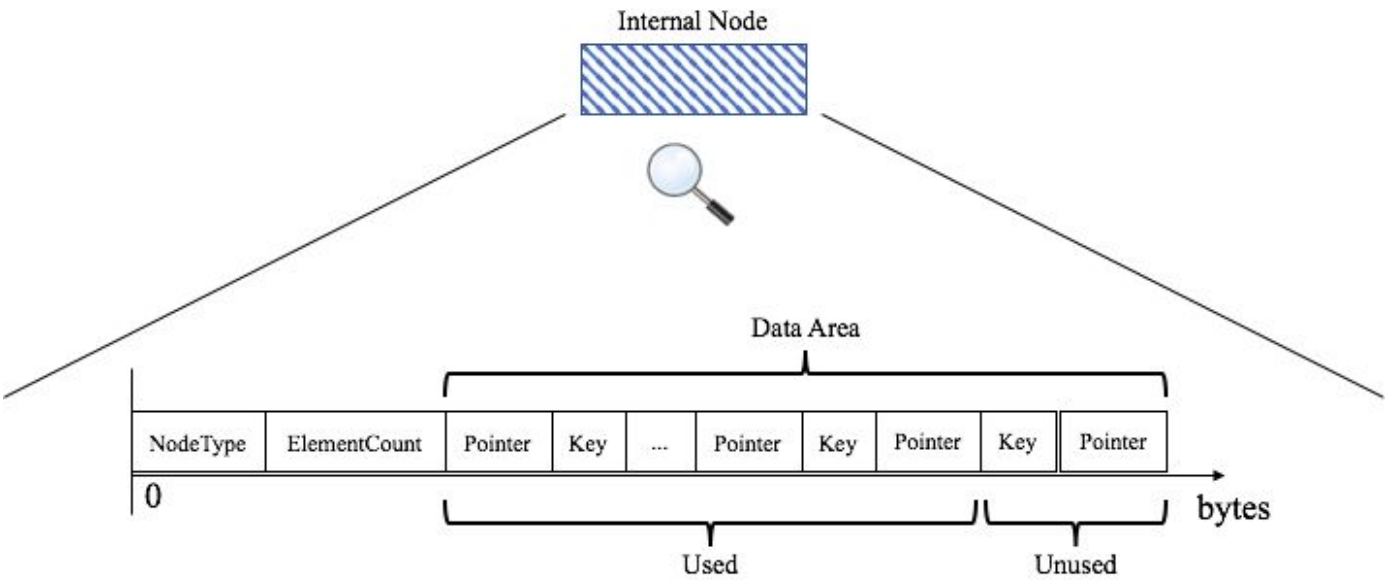
空闲池节点



空闲池节点是存储在外存中的要求，其功能是存储可用的空闲节点位置，实现空间回收。在我们的存储结构设计中，一棵B+树对应一个文件，但这个文件可能并不是紧凑的（例如B+树执行删除操作时就可能导致文件中的某些页被置空）。因此，需要用空闲池节点记录空闲页面的位置，以实现空闲空间的再利用。

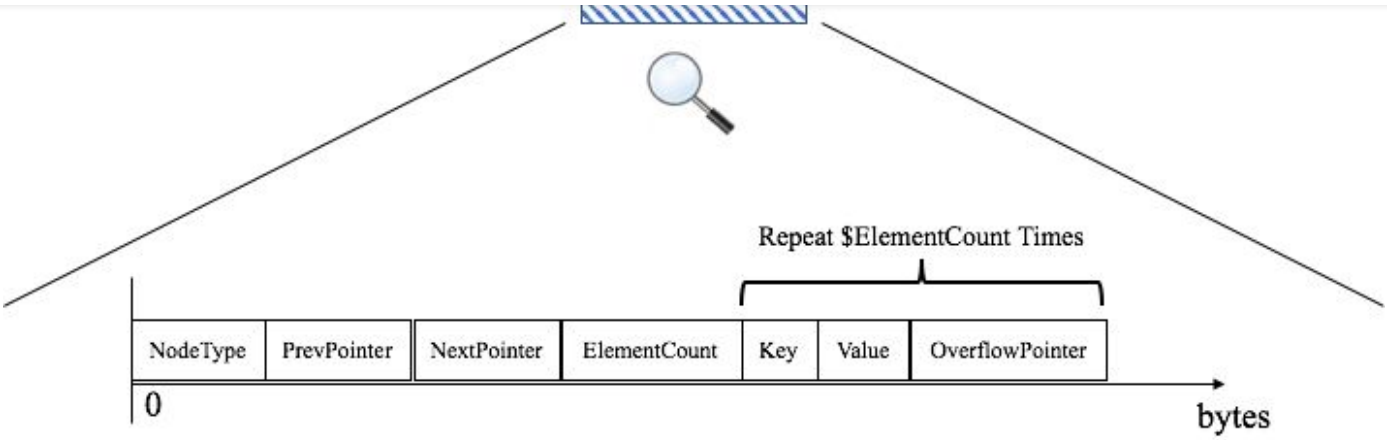
从这里我们可以看到，在外存中实现B+树时，需要自己实现对于空间的管理。

内部节点



内部节点的功能是确定检索路径，只存储用来分界的key值和该key值对应的分界后的节点位置。

叶节点

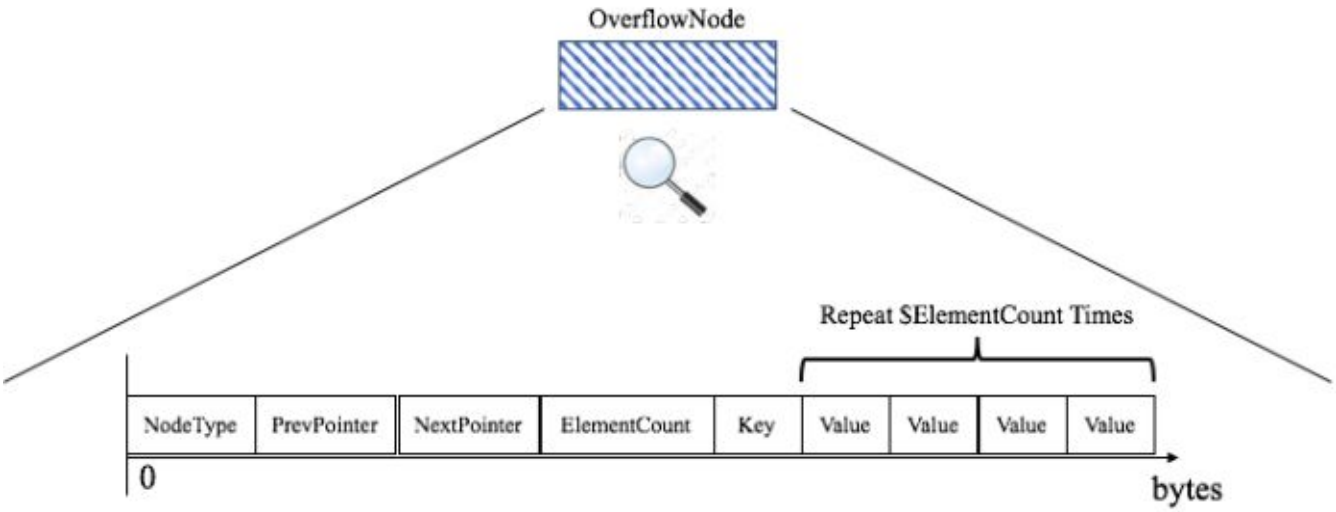


叶节点的功能是存储具体数据，其数据存储的主体部分是若干key-value键值对。

B+树的叶节点是彼此链接的，PrevPointer和NextPointer用来标识前驱和后继节点的位置，方便遍历所有叶子节点。

对于non-unique的树（即一个key可以对应多个value），每个key后面跟着溢出节点指针OverflowPointer。如果一个key对应多于一个value，则第二个及之后的value会被存储在溢出页中（没有溢出页的情况下，OverflowPointer会被置为-1）。

溢出节点



溢出节点对应一个key有多个value的情况，第二个及之后的value存储会被在溢出节点中。为了使用方便，溢出节点中也存储了这些value对应的key。

总结

知乎

好了。

基于外存实现时，我们还要面对外存速度比内存访问慢的问题。这时，把大量的读写操作先在内存中拼接并合成一次读写操作，会对性能带来极大的提升。

发布于 2019-05-29

数据库 数据结构 B/B+树

推荐阅读

[译]从磁盘结构到B+树

廖长江

平衡二叉树、B树、B+树、B*树 理解其中一种你就都明白了

1、平衡二叉树 概念平衡二叉树是基于二分法的策略提高数据的查找速度的二叉树的数据结构； 特点：平衡二叉树是采用二分法思维把数据按规则组装成一个树形结构的数据，用这个树形结构的数据...

勤劳的小手

5 条评论 切换为时间排序

写下你的评论...

知乎用户 9 个月前

有没有源码可以参考一下?

赞

binave

▲ 赞同 12 ▼ 5 条评论 分享 收藏

知乎

钞票 8 个月前

写的有点简单，详细一点就好了

赞

张潇 8 个月前

出发点很好，期待完善补全

赞

游凯超 (作者) 7 个月前

给出代码：github.com/youkaichao/m...

赞

▲ 赞同 12 ▼ 5 条评论 分享 收藏