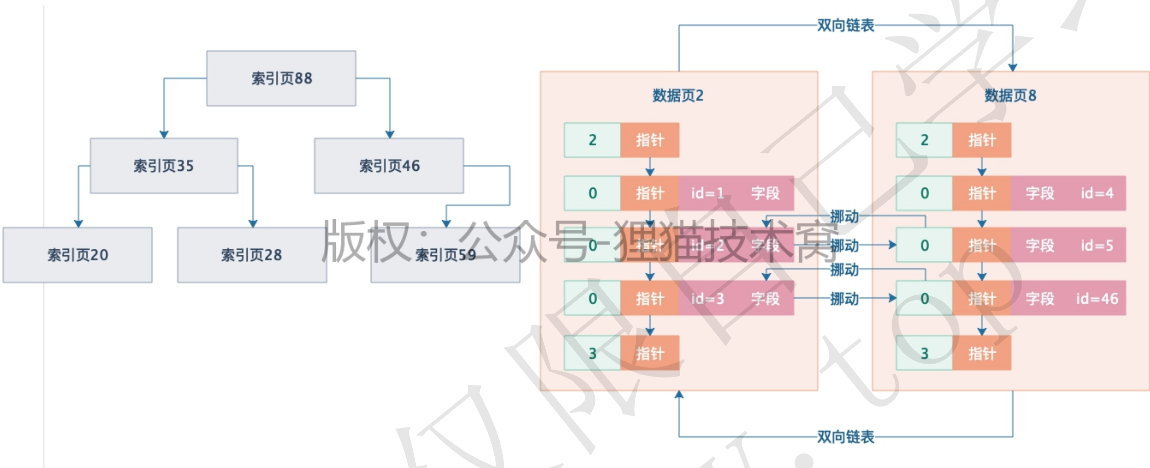
69 更新数据的时候，自动维护的聚簇索引到底是什么？

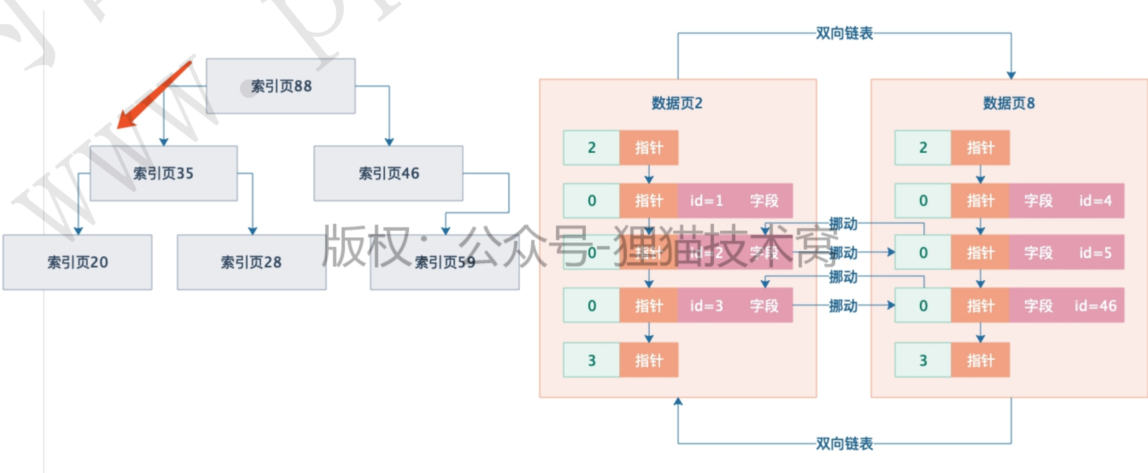
更新数据的时候，自动维护的聚簇索引到底是什么？

上一次我们给大家讲了一下基于主键如何组织一个索引，然后建立索引之后，如何基于主键在索引中快速定位到那行数据所在的数据页，再如何进入数据页快速到定位那行数据，大家看下面的图。

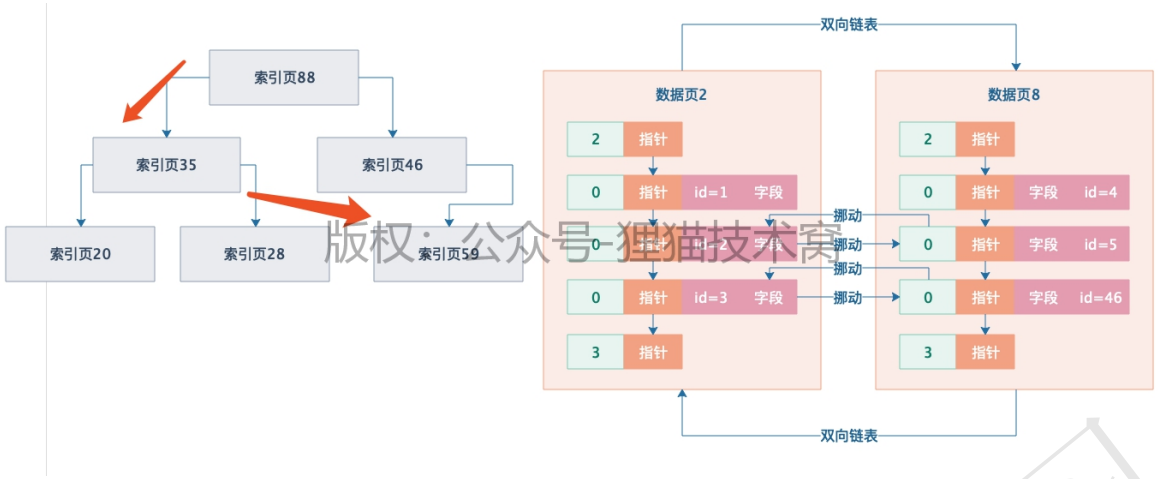


我们今天就先基于上面的图，把按照主键来搜索数据的过程重新再次给大家来梳理一遍，接着讲完了这个，其实大家也就理解今天的主题，**聚簇索引**了。

首先呢，现在假设我们要搜索一个主键id对应的行，此时你就应该先去顶层的索引页88里去找，通过二分查找的方式，很容易就定位到你应该去下层哪个索引页里继续找，如下图所示，我们给一个图示出来。

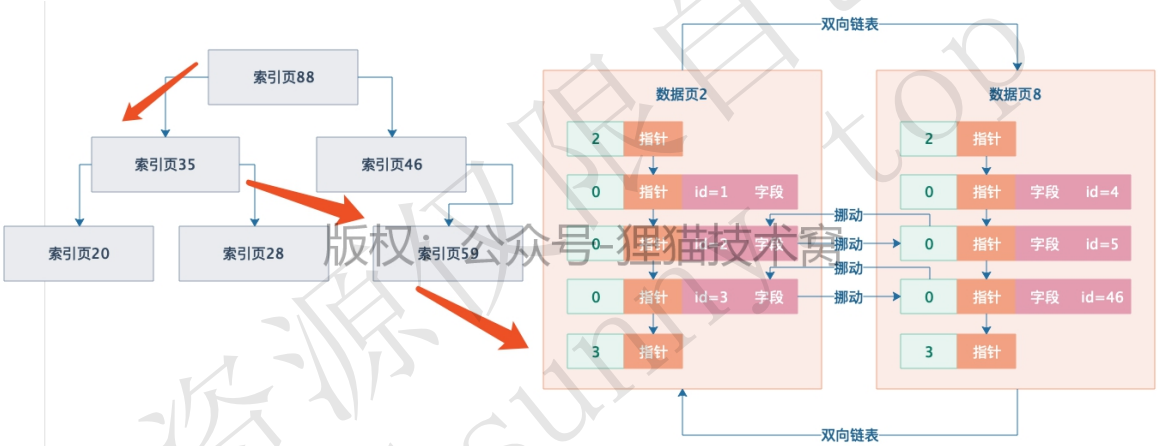


比如现在定位到了下层的索引页35里去继续找，此时在索引页35里也有一些索引条目的，分别都是下层各个索引页（20，28，59）和他们里面最小的主键值，此时在索引页35的索引条目里继续二分查找，很容易就定位到，应该再到下层的哪个索引页里去继续找，如下图所示。



我们这里看到，可能从索引页35接着就找到下层的索引页59里去了，此时索引页59里肯定也是有索引条目的，这里就存放了部分数据页页号（比如数据页2和数据页8）和每个数据页里最小的主键值

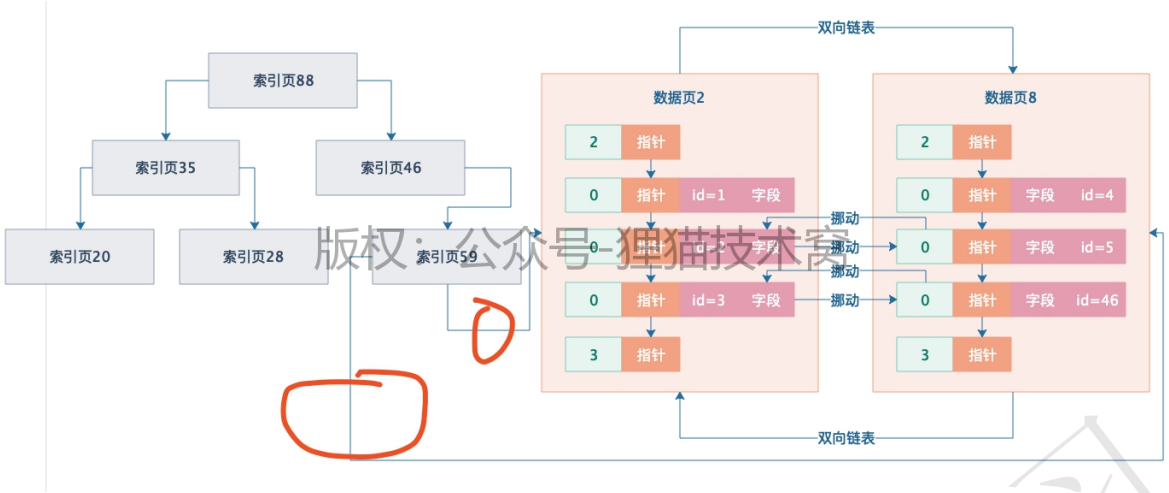
此时就在这里继续二分查找，就可以定位到应该到哪个数据页里去找，如下图所示。



接着比如进入了数据页2，里面就有一个页目录，都存放了各行数据的主键值和行的实际物理位置

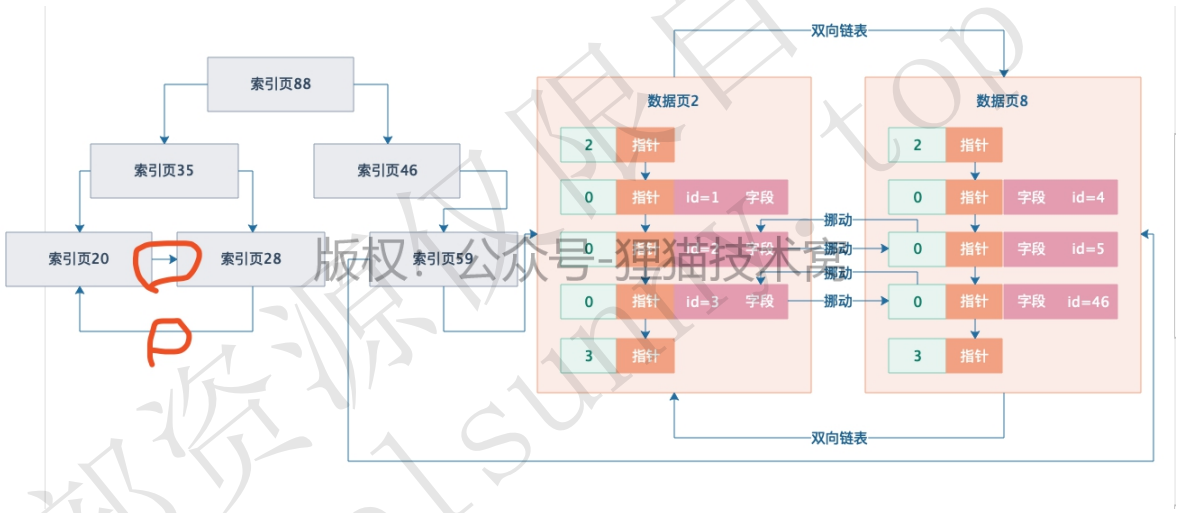
此时在这里直接二分查找，就可以快速定位到你要搜索的主键值对应行的物理位置，然后直接在数据页2里找到那条数据即可了。

这就是基于索引数据结构去查找主键的一个过程，那么大家有没有发现一件事情，其实最下层的索引页，都是会有指针引用数据页的，所以实际上索引页之间跟数据页之间是有指针连接起来的，如下图。



另外呢，其实索引页自己内部，对于一个层级内的索引页，互相之间都是基于指针组成双向链表的，如下面图示

大家可以看看，这个同一层级内的索引页组成双向链表，就跟数据页自己组成双向链表是一样的。



不知道大家把上面的图连起来看，有没有发现一些特点，就是说假设你把索引页和数据页综合起来看，他们都是连接在一起的，看起来就如同一颗完整的大的B+树一样，从根索引页88开始，一直到所有的数据页，其实组成了一颗巨大的B+树。

在这颗B+树里，最底层的一层就是数据页，数据页也就是B+树里的叶子节点了！

所以，如果一颗大的B+树索引数据结构里，叶子节点就是数据页自己本身，那么此时我们就可以称这颗B+树索引为聚簇索引！

也就是说，上图中所有的索引页+数据页组成的B+树就是聚簇索引！

其实在InnoDB存储引擎里，你在对数据增删改的时候，就是直接把你的数据页放在聚簇索引里的，数据就在聚簇索引里，聚簇索引就包含了数据！比如你插入数据，那么就是在数据页里插入数据。

如果你的数据页开始进行页分裂了，他此时会调整各个数据页内部的行数据，保证数据页内的主键值都是有顺序的，下一个数据页的所有主键值大于上一个数据页的所有主键值

同时在页分裂的时候，会维护你的上层索引数据结构，在上层索引页里维护你的索引条目，不同的数据页和最小主键值。

然后如果你的数据页越来越多，一个索引页放不下了，此时就会再拉出新的索引页，同时再搞一个上层的索引页，上层索引页里存放的索引条目就是下层索引页页号和最下主键值。

按照这个顺序，以此类推，如果你的数据量越大，此时可能就会多出更多的索引页层级来，不过说实话，一般索引页里可以放很多索引条目，所以通常而言，即使你是亿级的大表，基本上大表里建的索引的层级也就三四层而已。

这个聚簇索引默认是按照主键来组织的，所以你在增删改数据的时候，一方面会更新数据页，一方面其实会给你自动维护B+树结构的聚簇索引，给新增和更新索引页，这个聚簇索引是默认就会给你建立的。

End