索引大家都知道是一种能加快数据查询的数据结构。

说到索引离不开存储引擎，因为索引是由存储引擎决定的。我们常见的存储引擎有Innodb,Myisam.其他存储引擎有：NDB,Archive,Federated,Maria。今天主要介绍下Innodb存储引擎。

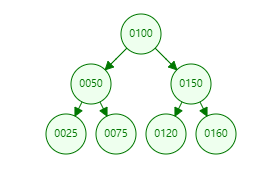
**innob 为什么使用B+树做索引，二叉树，B树不行吗**

数据的检索时间取决于磁盘次数，操作系统每次读取一个数据块(页)的数据到内存。

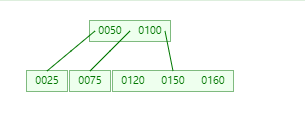
假设当前有如下数据：(数据结构可视化<https://www.cs.usfca.edu/~galles/visualization/Algorithms.html>)

25,50,75,100,120,150,160

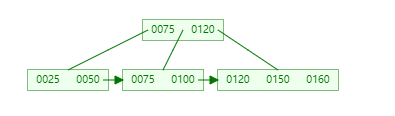
二叉树数据结构



B树



B+树



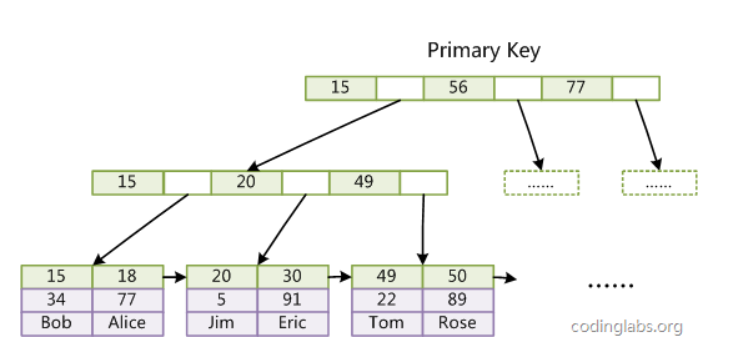
二叉树查找叶子节点，由于在所有节点的数据并不是整齐相邻排列，比如100,50,25,可能分布在3个数据块中。元素很多时会导致树很高，查找时会导致多次io操作(时间主要花费在寻址时间)。

B树和B+树 最显著的区别：B+树的所有数据都会出现在叶子节点上，并且叶子节点数据有序，节点之间通过指针相连。在遍历数据时，B+树只要定位到叶子节点然后遍历就行了，B树的遍历需要遍历整颗树。

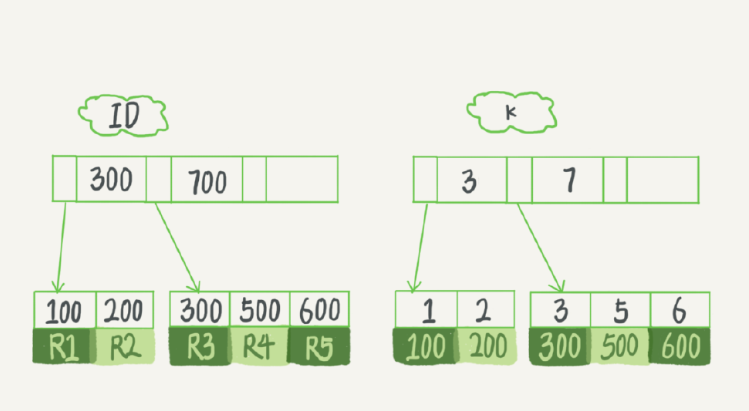
**这些图的结构网上博客都有，这些结构图结论怎么得来的，有什么证据？**

关于Innodb的索引结构先来看下这张图。Innodb 按索引组织数据，所有数据结构==主键索引结构

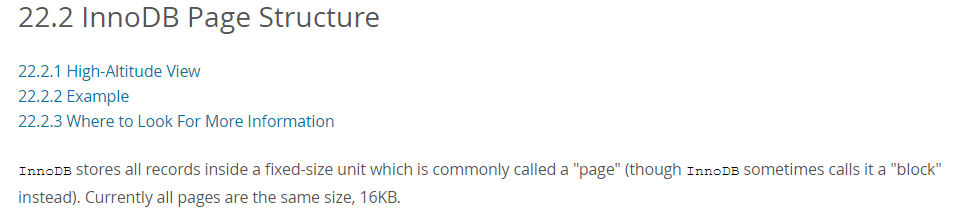
主键索引图：



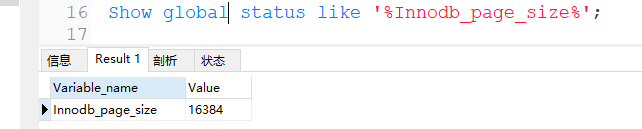
主键索引与普通所以区别



这些图的结构是怎么画出来的，或者说根据什么证据得出来的。这些树状结构归根到底还是由数据构成的，所以来分析下innodb的数据文件。



Innodb的数据以page(页)为单位进行存储,大小为16kb  
Show global status like ‘%Innodb\_page\_size%’;



16384/1024=16

我们先建一张表。下面的例子都是基于这张表来演示的。

create DATABASE ssm;

Create table t\_demo(

Id int primary key,

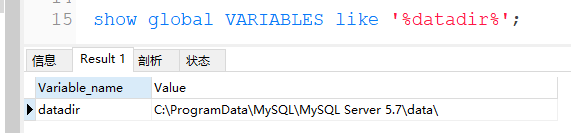
K int not null,

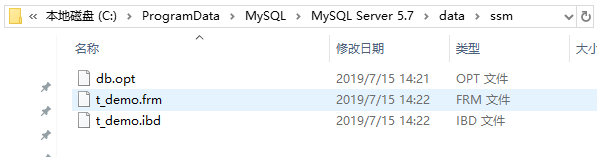
Name varchar(16),

Index index\_k(k)  
) ENGINE=InnoDB;

查看数据文件的位置：

Show global varables like ‘%datadir%’;





Db.opt 保存数据库字符编码等信息。

default-character-set=utf8

default-collation=utf8\_general\_ci

MySQL中.frm文件：保存了每个表的元数据，包括表结构的定义等，该文件与数据库引擎无关。

MySQL中.ibd文件：InnoDB引擎开启了独立表空间(my.ini中配置innodb\_file\_per\_table = 1)产生的存放该表的数据和索引的文件。

所以下面主要分析.dbd文件