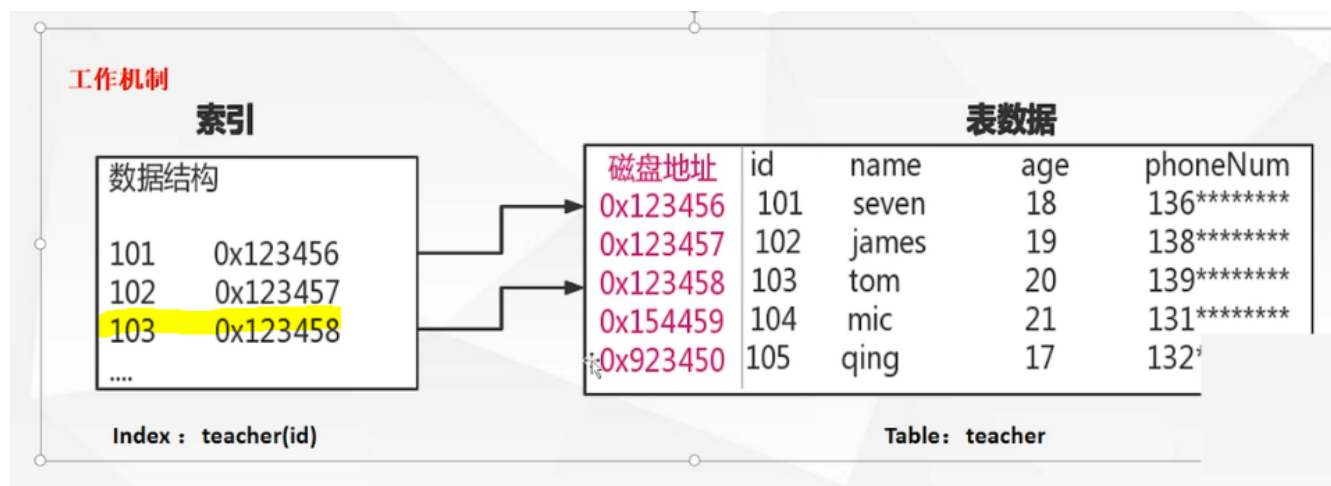


正确的创建**适合**的索引,是提升数据库查询性能的**基础**

索引的定义

索引是为了加速对表中数据行的检索而创建的一种分散存储的**数据结构**,B+树



索引是一种硬盘式索引,即数据过大,索引的容量也会随之变大,此时就会保存一部分在硬盘中

频繁插入和修改的列不建议添加索引,会带来性能的消耗,因为索引的数据结构机制是多路平衡查找树,当你插入或修改数据的时候,需要调整树的结构(左旋右旋),这就会消耗性能

搜索效率不足

一般来说,在树结构中数据处的深度决定着它的搜索时IO次数

节点数据内容太少

每一个磁盘块(节点/页)保存的关键字数据量太小了

没有很好的利用操作系统和磁盘的数据交换特性和**磁盘预读能力**(空间局部性原理)

B+树的话,如果数据量越大,树的高度会越矮

MySQL中的B+Tree

- B+节点关键字搜索采用闭合区间
- B+非叶节点不保存数据相关信息，只保存关键字和子节点的引用
- B+关键字对应的数据保存在叶子节点中
- B+叶子节点是顺序排列的，并且相邻节点具有顺序引用的关系

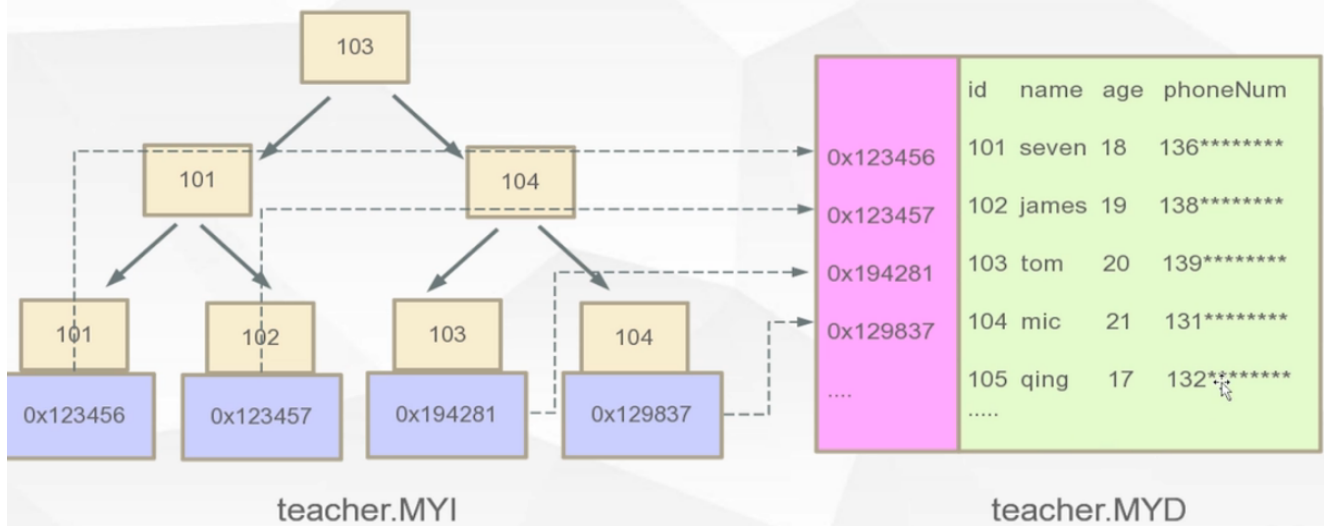


Mysql为什么选用B+Tree?

- B+树是B-树的变种（PLUS版）多路绝对平衡查找树，他拥有B-树的优势
- B+树扫库、表能力更强
- B+树的磁盘读写能力更强
- B+树的排序能力更强
- B+树的查询效率更加稳定（仁者见仁、智者见智）

Mysql中B+Tree索引体现形式-Myisam

Myisam



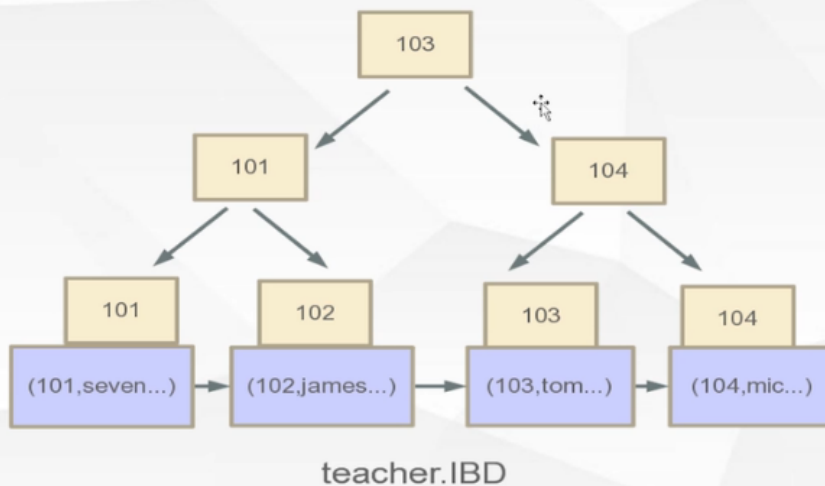
技术人的指路明灯,做职场生涯的精神导师

Mysql中B+Tree索引体现形式-Innodb

innodb

以主键为索引来组织数据的存储

聚集索引:
数据库表中数据的
物理顺序与键值的逻辑
顺序(索引)顺序相同

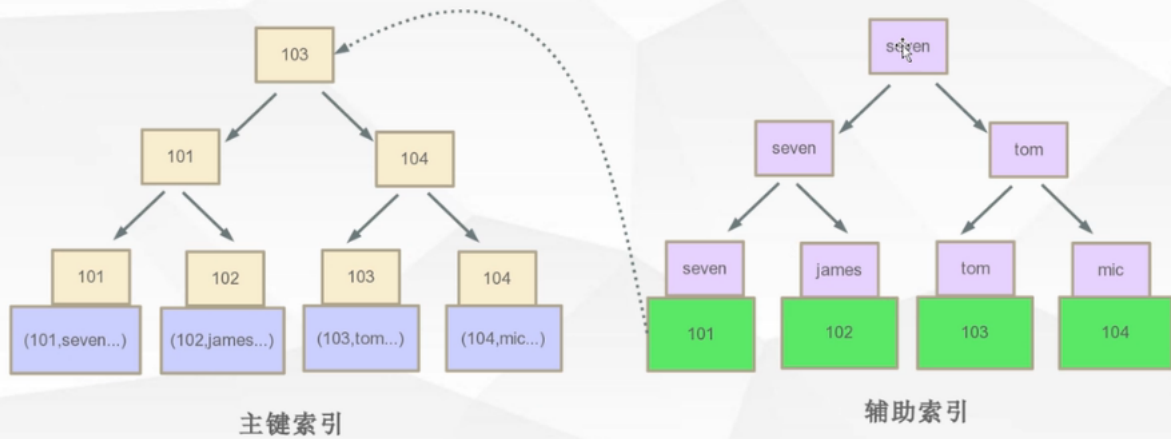


人的指路明灯,做职场生涯的精神导师

聚集索引:数据库行中数据的物理顺序和逻辑顺序相同 InnoDB利用主键存储只有一个聚集索引

Mysql中B+Tree索引体现形式-Innodb

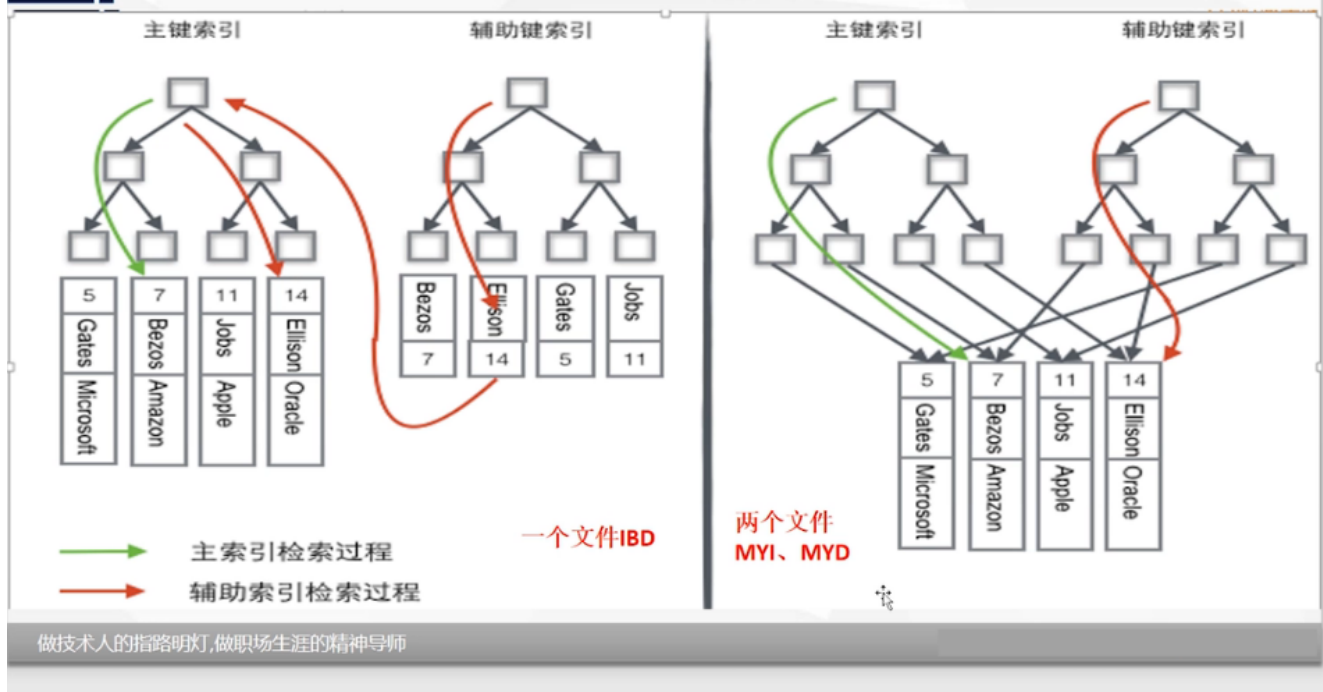
innodb



做技术人的指路明灯,做职场生涯的精神导师

innodb有个隐藏列 rowid,没有指定主键会用rowid作为自增默认主键

Innodb VS Myisam



做技术人的指路明灯,做职场生涯的精神导师

索引的几大原则

innodb 是 98%的选择比例

列得离散性

列的离散性

```
mysql> mysql> select * from distribution;
```

name	zoneDesc	sex
张二狗	0755	男
李大力	010	男
赵美丽	020	女
钱大妈	0731	女
孙漂亮	010	女
张三疯	0738	男
牛大宝	0755	男
东门庆	020	男
潘银莲	0755	女

```
9 rows in set (0.00 sec)
```

```
mysql>
```

找出图中三列中离散性最好的列？

计算公式：

$\text{count}(\text{distinct col}) : \text{count}(\text{col})$

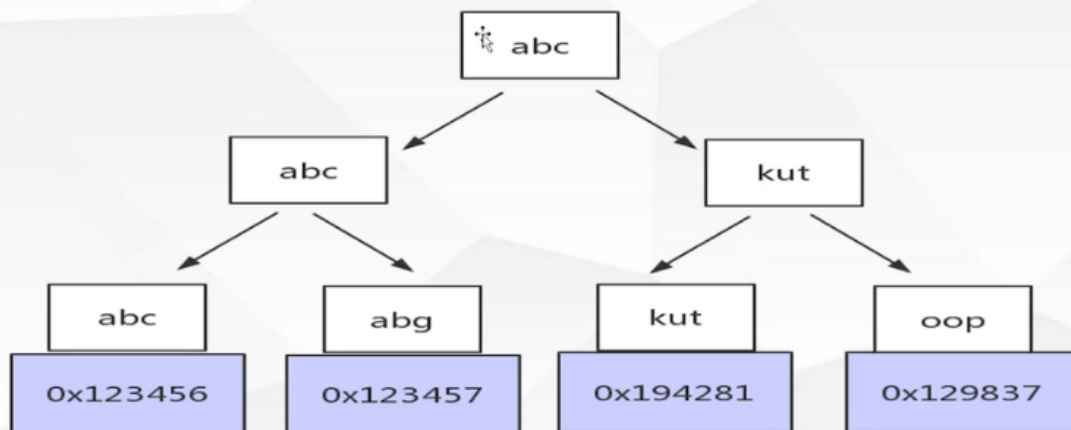
离散性越高
选择性就越好

离散性太差(即重复值太多)不建议添加索引,因为选择过多,无法判断具体一个值。

认真想一下,如果大家都一样,我加索引不就多此一举了吗?

最左匹配原则

对索引中关键字进行计算(对比),一定是从左往右依次进行,且不可跳过



联合索引? so easy

单列索引

节点中关键字[name]

联合索引

节点中关键字[name,phoneNum]

单列索引是特殊的联合索引

联合索引列选择原则

- 经常用的列优先【最左匹配原则】
- 离散度高的列优先【离散度高原则】
- 宽度小的列优先【最少空间原则】

【最左匹配原则】



【离散度高原则】



【最少空间原则】

机灵的李二狗

经排查发现最常用的sql语句:

```
select * from users where name = ? ;
```

```
select * from users where name = ? and phoneNum = ?;
```

机灵的李二狗的解决方案:

```
create index idx_name on users(name);
```



```
create index idx_name_phoneNum on users(name,phoneNum);
```



覆盖索引

如果查询的列，通过索引项的信息可直接返回，则该索引称之为查询SQL的覆盖索引

表: teacher

索引: PK(id) key(name,phoneNum) unique(teacherNo)

哪些SQL使用了覆盖索引?

select teacherNo from teacher where teacherNo= ?



select id,teacherNo from teacher where teacherNo= ?



select name,phoneNum from teacher where teacherNo= ?



select phoneNum from teacher where name = ?



术人的指路明灯,做职场生涯的精神导师



面试常客MySQL索引 你不知道的那些事

主要内容

全值匹配我最爱，最左前缀要遵守；
带头大哥不能死，中间兄弟不能断；
索引列上少计算，范围之后全失效；
Like百分写最左，覆盖索引不写星；
不等空值还有or，索引失效要少用；


seven 老师


2月28日 15:00



咕泡学院

索引用了函数的话,因为函数有返回值,不确定,所以优化器会认为用不上索引,


这些你从原理层理解了么？


索引列的数据长度满足业务的情况下能少则少，对吗？ 

表的索引越全越好，因为这样不管什么情况我都能用到索引，对吗？ 

Where 条件中 like 9999%, like %999%、like %999 三种方式都用不到索引？  


Where 条件中 NOT IN 和 <> 条件无法使用索引，对吗？ 

多用指定列查询，只返回自己想到的数据列，少用select *; 

SELECT name, phoneNum FROM teacher WHERE CONCAT(name, '1') = 'seven1' 使用key(name) 

联合索引：

联合索引中如果不是按照索引最左列开始查找，无法使用索引； 

联合索引中精确匹配最左前列并范围匹配另外一列可以用到索引； 

联合索引中如果查询中有某个列的范围查询，则其右边的所有列都无法使用索引； 