



【VIP直播课】

深入剖析MySQL索引原理

青山



关注咕泡 码上升职加薪





曾就职于国内顶级FinTech公司

多年Java从业经验,常年从事金融领域的项目研发,拥有传统金融和大型互联网金融项目的架构设计经验。 经历了消费金融公司从单体架构到大型分布式、微服务系统的演变,支撑千万级客户和日均过亿放款量。

青山-咕泡盆鱼宴

课程目标



- 1、理解索引的本质
- 2、通过推演掌握索引底层的数据结构
- 3、掌握在不同存储引擎中索引的落地方式
- 4、掌握索引的创建和使用原则



关注咕泡 码上升职加薪



适合了解MySQL架构,知道存储引擎的概念,了解索引基本类型(普通索引、唯一索引、主键索引),了解在MySQL中创建索引基本语法的同学





- 表的索引越全越好,因为不管什么情况都能用到索引,对吗?
- 为什么不要在性别字段上建索引?
- 为什么不建议使用身份证作为主键?
- 模糊匹配 like abc%, like %2673%, like %888都用不到索引,对吗?
- 不要使用select * , 写明具体查询字段 , 为什么?







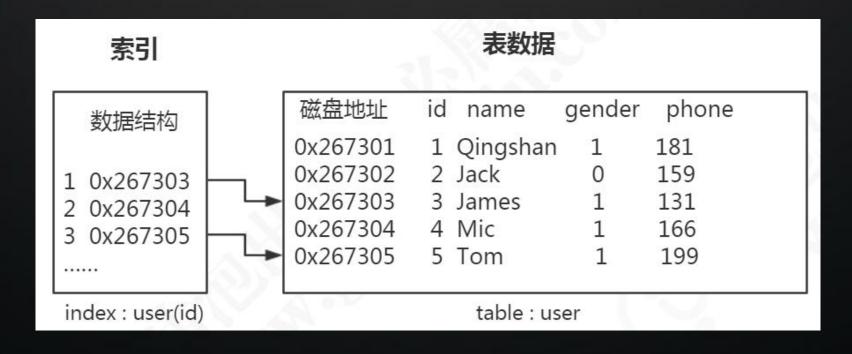
索引到底是什么?



关注咕泡 码上升职加薪



数据库索引,是数据库管理系统(DBMS)中一个排序的数据结构,以协助快速查询、更新数据库表中数据。





关注咕泡 码上升职加薪

专业IT教育培训,做技术人的指路明灯,职场生涯的精神导师

咕泡学院官网:http://www.gupaoedu.com



索引类型:Normal、Unique、Fulltext





索引,应该选择一种什么数据结构?



关注咕泡 码上升职加薪





MySQL索引数据模型推演



关注咕泡 码上升职加薪



猜猜我双十一买了多少钱?



关注咕泡 码上升职加薪



有序数组:

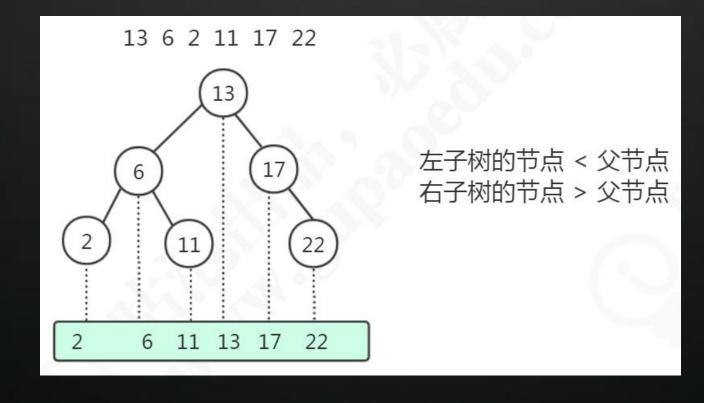
单链表:



关注咕泡 码上升职加薪

二叉查找树(Binary Search Tree)



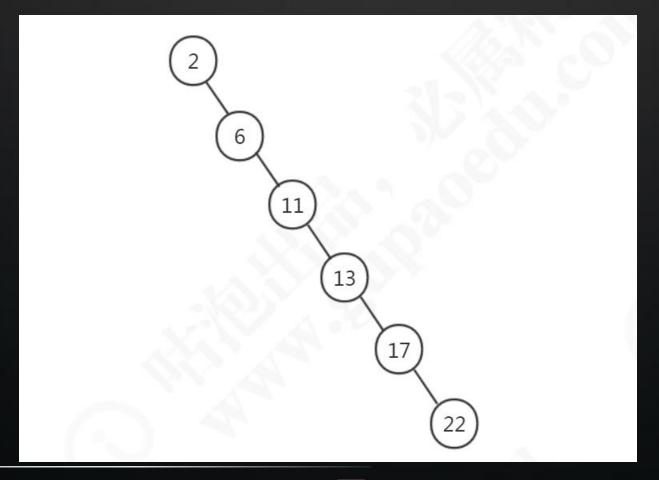


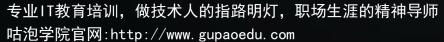


关注咕泡 码上升职加薪



顺序插入: 2、6、11、13、17、22





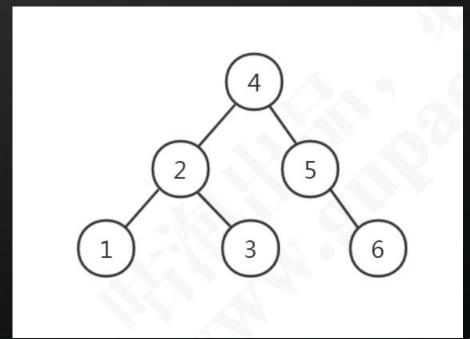


关注咕泡 码上升职加薪



平衡二叉树:左右子树深度差绝对值不能超过1

顺序插入:1、2、3、4、5、6





关注咕泡 码上升职加薪



插入:1、2

再插入:3





关注咕泡 码上升职加薪



插入:7、6

再插入:5

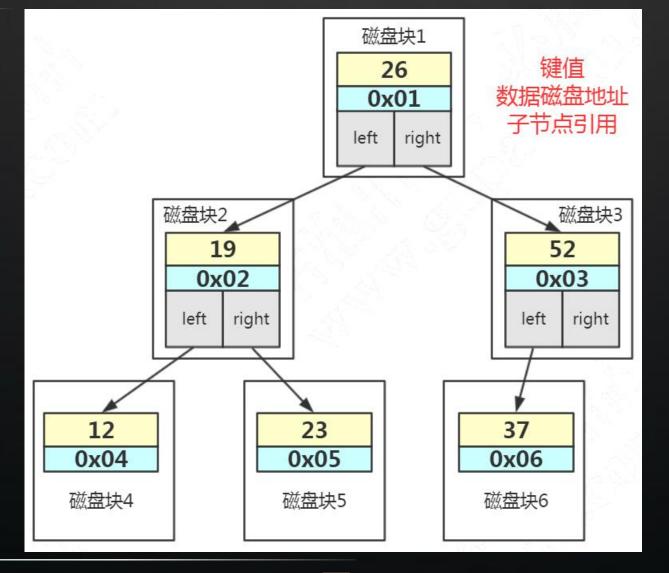




关注咕泡 码上升职加薪

AVL Tree -数据存储



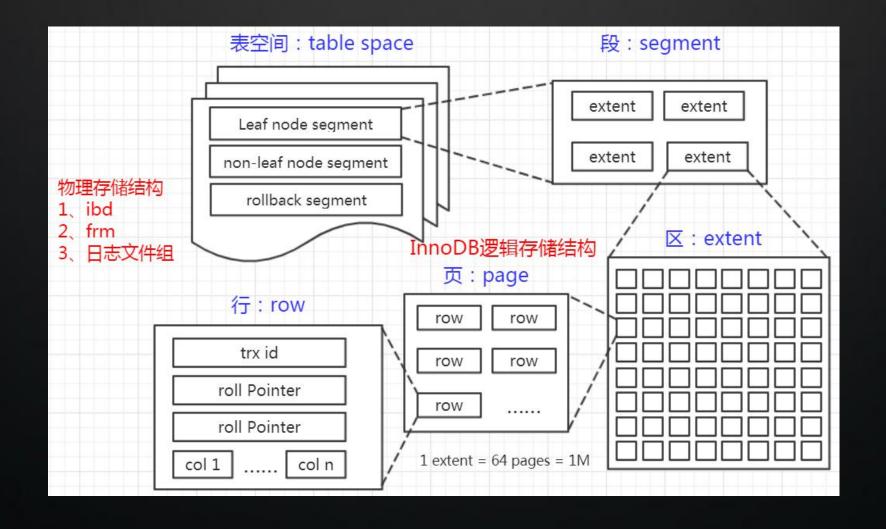




关注咕泡 码上升职加薪

InnoDB 逻辑存储结构



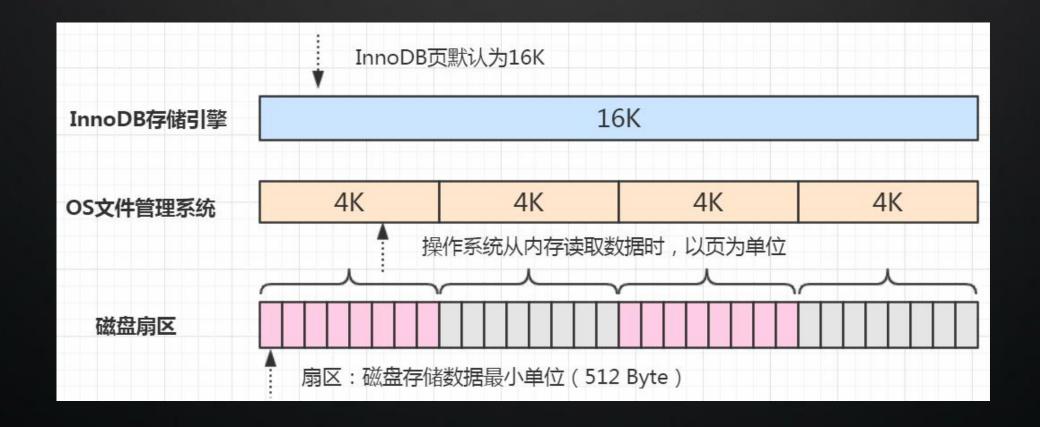




关注咕泡 码上升职加薪

文件系统中的页

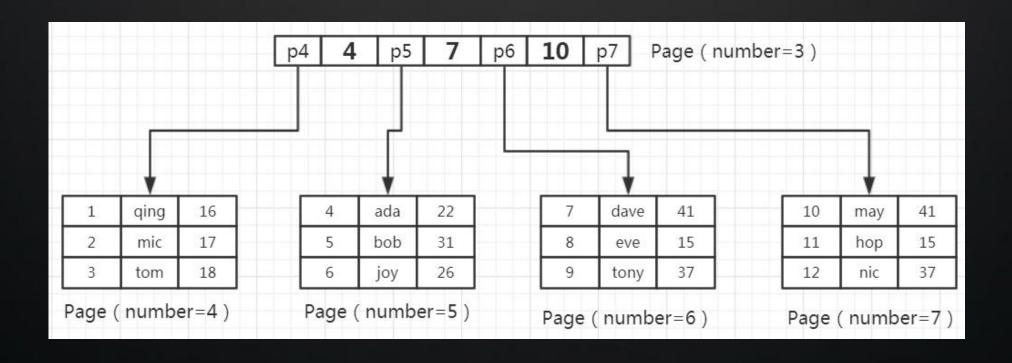






关注咕泡 码上升职加薪



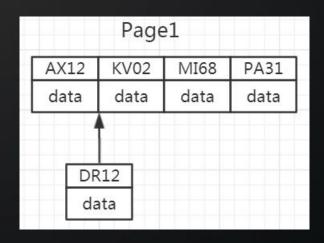




关注咕泡 码上升职加薪



ge2	Pag			ge1	Pa	
	18	17	16		2	1
	data	data	data		data	data
19						
data						





关注咕泡 码上升职加薪



Antelope[ˈæntɪləʊp](羚羊)是InnoDB内置的文件格式,有两种行格式:

- REDUNDANT[rɪˈdʌndənt] Row Format
- COMPACT Row Format (5.6默认)

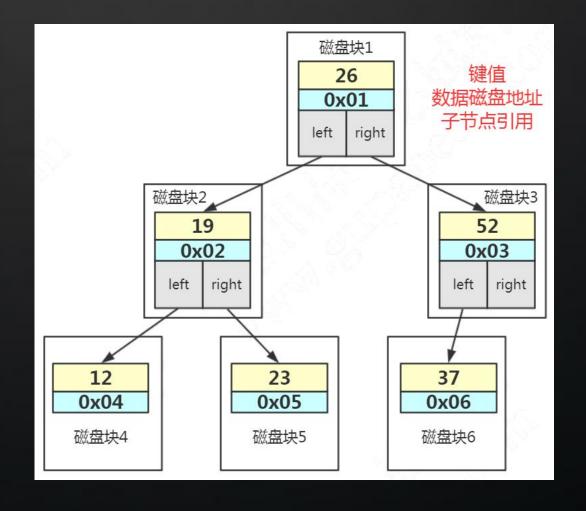
Barracuda[ˌbærəˈkjuːdə](梭子鱼)是InnoDB Plugin支持的文件格式,新增了两种行格式:

- DYNAMIC Row Format (5.7默认)
- COMPRESSED Row Format





- ●操作系统与磁盘交互
- ●树的深度

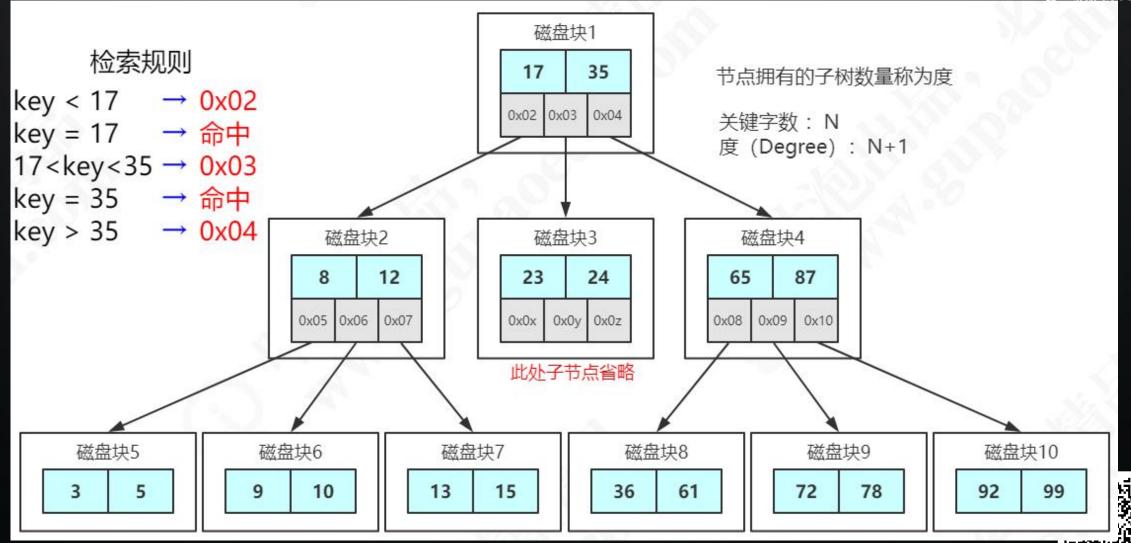




关注咕泡 码上升职加薪

多路平衡查找树(Balanced Tree B树)





专业IT教育培训,做技术人的指路明灯,职场生涯的精神导师咕泡学院官网:http://www.gupaoedu.com



●分裂、合并

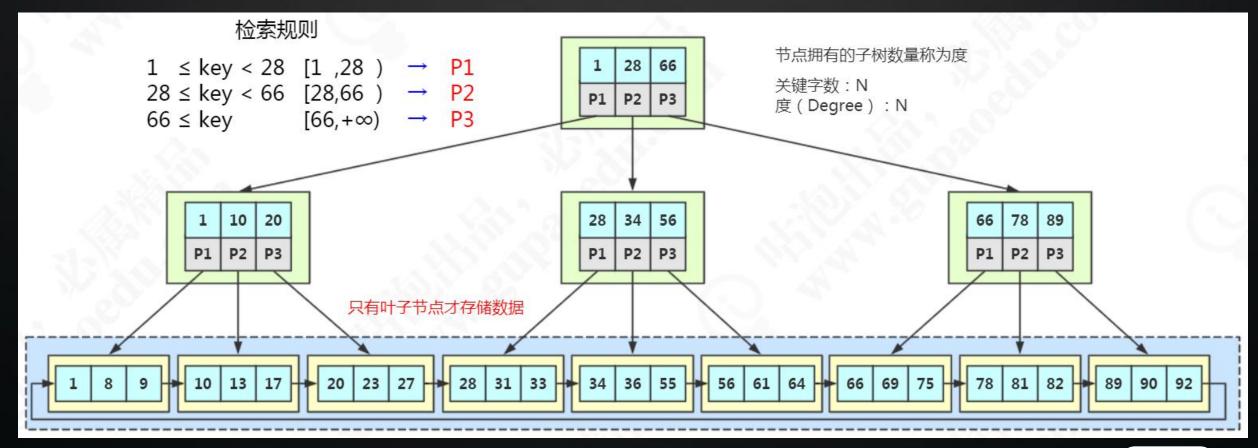




关注咕泡 码上升职加薪

B+Tree 加强版多路平衡查找树

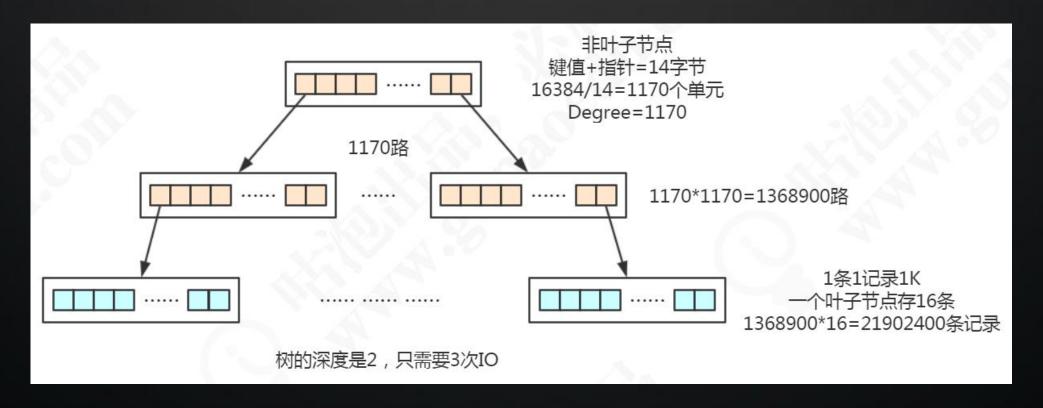






关注咕泡 码上升职加薪







关注咕泡 码上升职加薪



- ●B Tree能解决的问题, B+Tree都能解决
- ●扫库、扫表能力更强
- ●磁盘读写能力更强
- ●排序能力更强
- ●效率更加稳定



关注咕泡 码上升职加薪

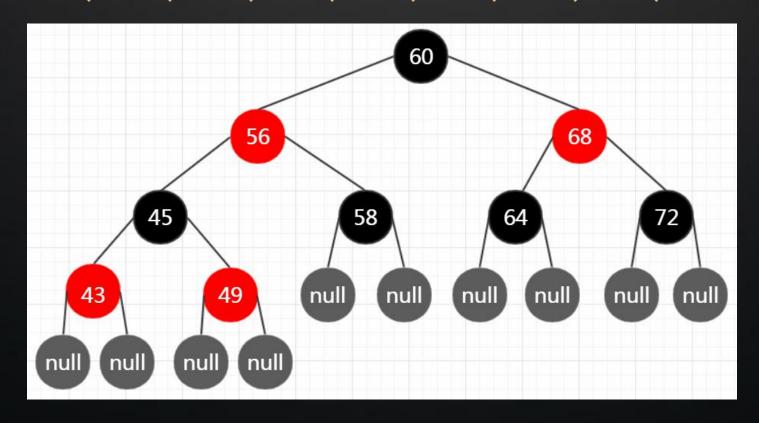


- 1、节点分为红色或者黑色。
- 2、根节点必须是黑色的。
- 3、叶子节点都是黑色的NULL节点。
- 4、红色节点的两个子节点都是黑色(不允许两个相邻的红色节点)。
- 5、从任意节点出发,到其每个叶子节点的路径中包含相同数量的黑色节点。





插入:60、56、68、45、64、58、72、43、49

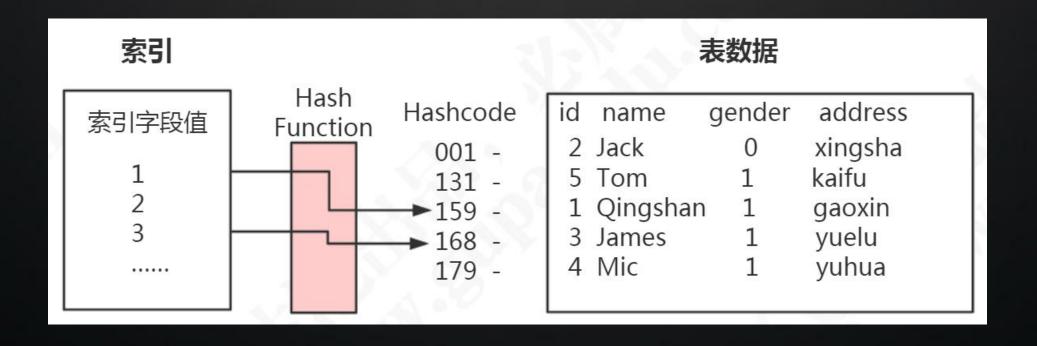




关注咕泡 码上升职加薪



索引方式: Hash、B Tree







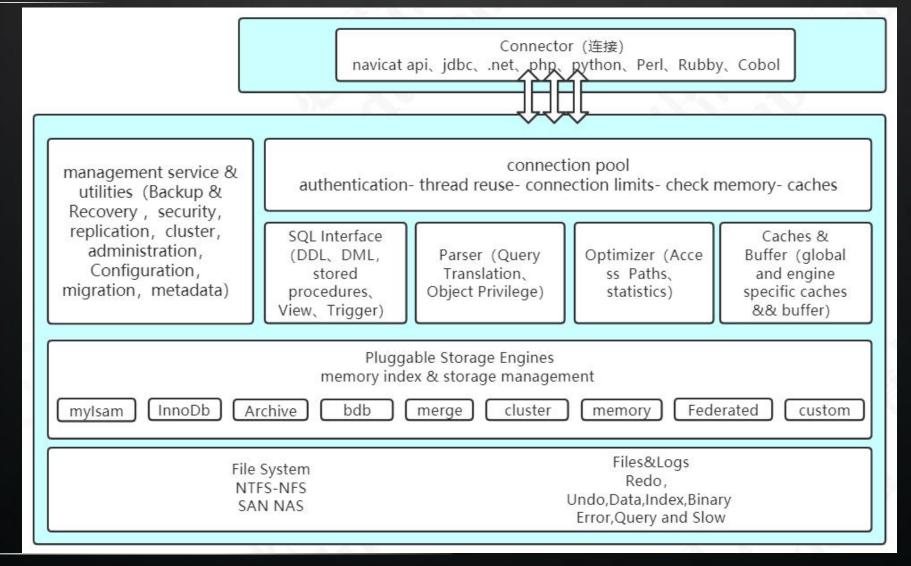


存储引擎中索引如何落地?



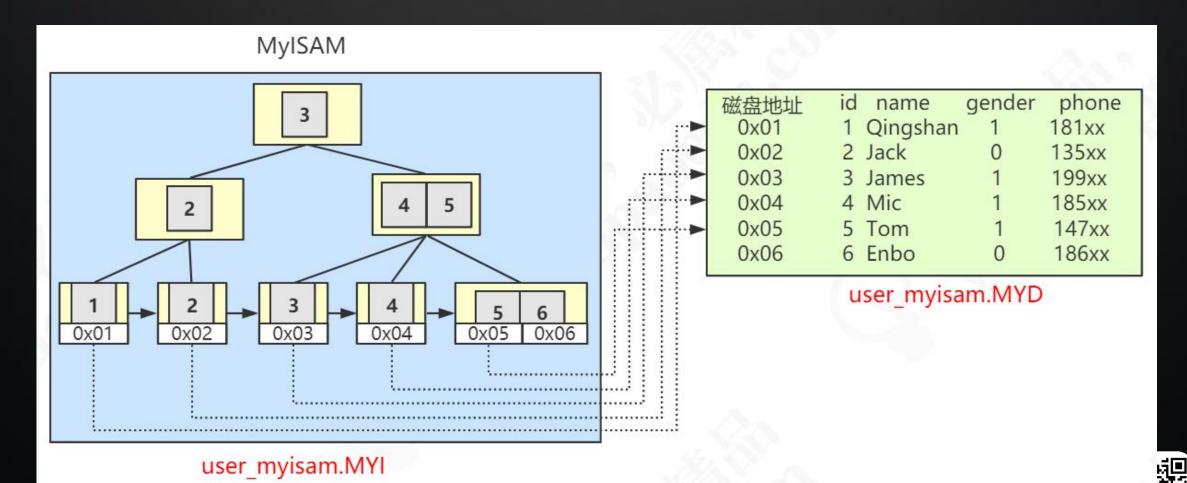
关注咕泡 码上升职加薪



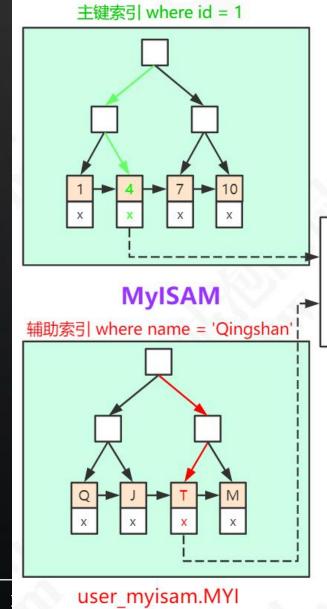








My I SAM-辅助索引



磁盘地址 gender phone name Qingshan 181xx 0x267301 0x267302 Jack 135xx 0x267303 Tom 199xx 0x267304 10 Mic 185xx

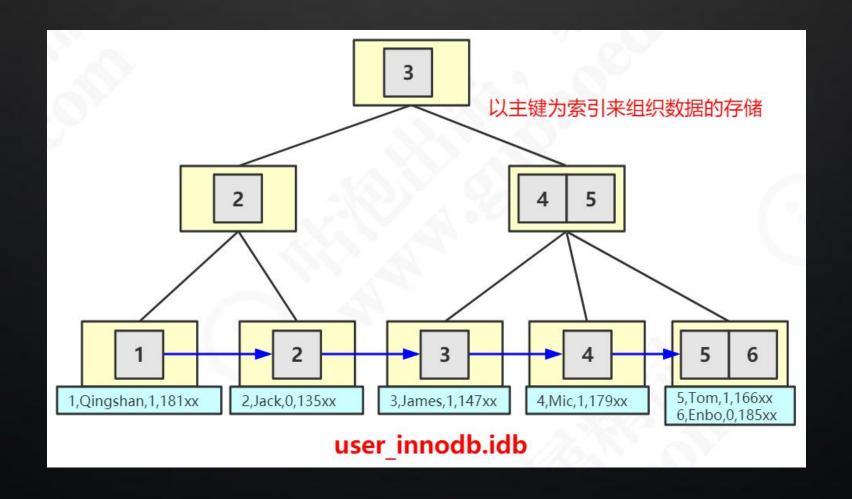
user_myisam.MYD



设据泡学院 gupaoedu.com

关注咕泡 码上升职加薪

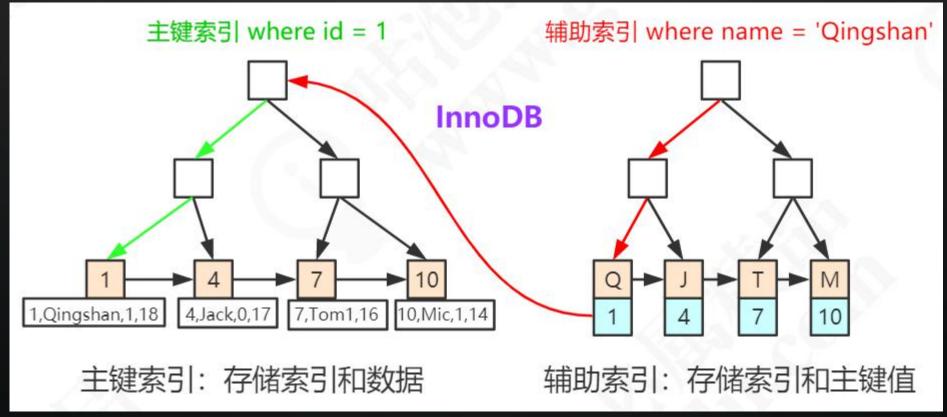






关注咕泡 码上升职加薪





● 没有主键索引怎么办?



关注咕泡 码上升职加薪





索引的使用原则



关注咕泡 码上升职加薪



在所有的字段上创建索引.....

```
常规
    DDL
PEOPLE_PEOPLE_TOTAL_QKLJF_RYSSCS_QKL_302_SCORE varchar(50) DEFAULT NULL,
PRIMARY KEY ('id'),
KEY 'ZJF INDEX' ('ZJF'),
     'JFSJ INDEX' ('JFSJ'),
     'CSRQ INDEX' ('CSRQ'),
     `SJHM INDEX` (`SJHM`),
      'NL INDEX' ('NL'),
      'SFZH INDEX' ('SFZH'),
      'XB INDEX' ('XB'),
      'HJDXZQH INDEX' ('HJDXZQH'),
     `HJDXXDZ INDEX` (`HJDXXDZ`),
      IS_GANGDOM_INDEX' ('IS_GANGDOM'),
      DELETE_FLAG_INDEX' ('DELETE_FLAG'),
     'MZ INDEX' ('MZ')
 ENGINE=InnoDB DEFAULT CHARSET=utf8;
```



关注咕泡 码上升职加薪



离散度公式:count(distinct(column_name)):count(*)

id	name	gender	phone
1	青山	0	1366666666
2	岑粒	0	13800722654
3	章机轻	1	16607463532
4	于琴旺	0	15205286470
5	皮明业	1	15901557881
6	伊颚	0	15003812562
7	宁慨	1	19900275915
8	明鹕	0	15104237001
9	蒋椴	0	13206846562
10	魏仿巡	1	15104584161

gender 和 phone,哪一列的离散度更高?



关注咕泡 码上升职加薪

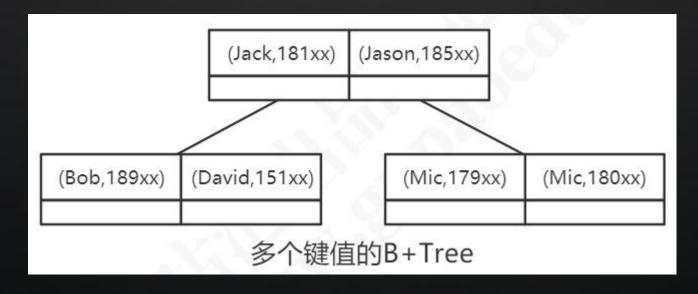
专业IT教育培训,做技术人的指路明灯,职场生涯的精神导师

咕泡学院官网:http://www.gupaoedu.com

(2) 联合索引最左匹配



ALTER TABLE user_innodb add INDEX `comidx_name_phone` (`name`,`phone`);





关注咕泡 码上升职加薪



INDEX `comidx_name_phone` (`name`,`phone`);

- ① SELECT * FROM user_innodb WHERE name= '青山' AND phone = '13666666666';
- ② SELECT * FROM user_innodb WHERE phone = '13666666666' AND name= '青山';
- ③ SELECT * FROM user_innodb WHERE name= '青山';
- SELECT * FROM user_innodb WHERE phone = '13666666666666';



关注咕泡 码上升职加薪

下面这种做法是对的吗?



常用的查询SQL:

- > SELECT * FROM user_innodb WHERE name= ? AND phone = ?;
- > SELECT * FROM user_innodb WHERE name= ?;

以下做法对吗?

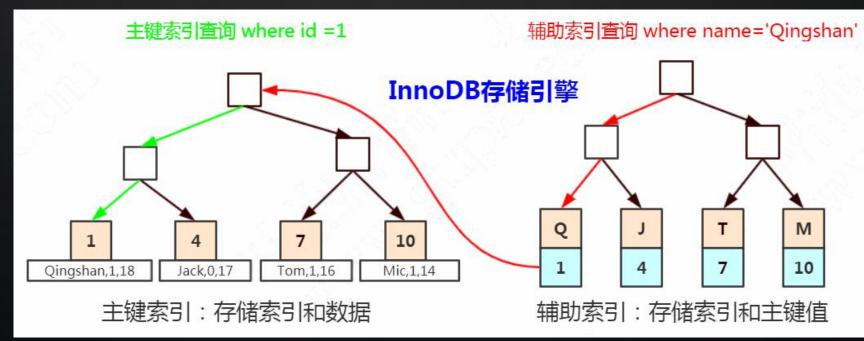
- ① CREATE INDEX idx_name on user_innodb(name);
- ② CREATE INDEX idx_name_phone on user_innodb(name,phone);



关注咕泡 码上升职加薪



- 什么是回表?
- 什么是覆盖索引?





关注咕泡 码上升职加薪



INDEX `comidx_name_phone` (`name`,`phone`);

EXPLAIN SELECT * FROM user_innodb WHERE name= '青山';

EXPLAIN SELECT name,phone FROM user_innodb WHERE name= '青山';

EXPLAIN SELECT name FROM user_innodb WHERE name= '青山' AND phone = '13666666666';



关注咕泡 码上升职加薪



last_name	first_name		emp_no	last_name	first_name	
XX	XX		1	XX	XX	
XX	XX	-	2	wang	d99	
XX	XX		3	XX	XX	
XX	XX	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	4	XX	XX	
XX	XX		5	wang	0dc	
wang	d99	20	6	XX	989	
wang	0dc		7	XX	e38	
wang	0zi	-	8	wang	0zi	1



关注咕泡 码上升职加薪



索引的创建与使用原则



关注咕泡 码上升职加薪



- 1、在用于where判断order排序和join的(on)字段上创建索引。
- 2、索引的个数不要过多。
- 3、区分度低的字段,例如性别,不要建索引。
- 4、频繁更新的值,不要作为主键或者索引。
- 5、符合索引把散列性高(区分度高)的值放在前面。
- 6、创建复合索引,而不是修改单列索引。
- 7、过长的字段,怎么建立索引?
- 8、为什么不建议用无序的值(例如身份证、UUID)作为索引



关注咕泡 码上升职加薪

什么时候用不到索引



- 1、索引列上使用函数(replace\SUBSTR\CONCAT\sum countary avg)、表达式
- 2、字符串不加引号,出现隐式转换
- 3、like条件中前面带%
- 4、负向查询能用到索引吗? <> != NOT in







谢谢你的鼓励和支持

青山老师



美**洼**晴泡 码上升职加薪

专业IT教育培训,做技术人的指路明灯,职场生涯的精神导师咕泡学院官网:http://www.gupaoedu.com





专业IT教育培训,做技术人的指路明灯,职场生涯的精神导师咕泡学院官网:http://www.gupaoedu.com



关注咕泡 码上升职加薪