mysql索引类型与数据存储

一.索引类型

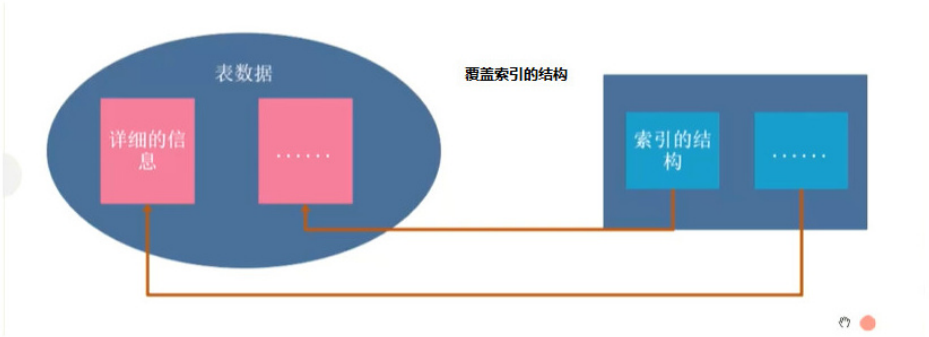
主键索引:就是我们的主键(一般选用唯一的自增的id)

唯一索引:与普通索引类似，不同的就是：索引列的值必须唯一，但允许有空值。如果是组合索引，则列值的组合必须一。

单索引:普通索引

全文索引:针对于中文进行的分词搜索

覆盖索引:指一个查询语句的执行只用从索引中就能够取得，不必从数据表中读取。也可以称之为实现了索引覆盖。



二.innodb和myisam索引存储数据的区别

1.innodb中主键索引和普通索引是什么关系?

普通图索引依赖于主键索引

2.区别

(1).物理文件方面

myisam:

.MYD:存储数据　.MYI:存储索引

特点是索引和数据分开存储

innodb:

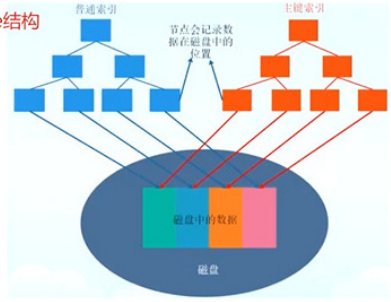
.idb　.ibdata

特点是数据和索引实际没有分开

(2).索引

myisam:普通索引与主键索引在索引的指向方向都是指定为实际的数据在磁盘中位置

btree结构

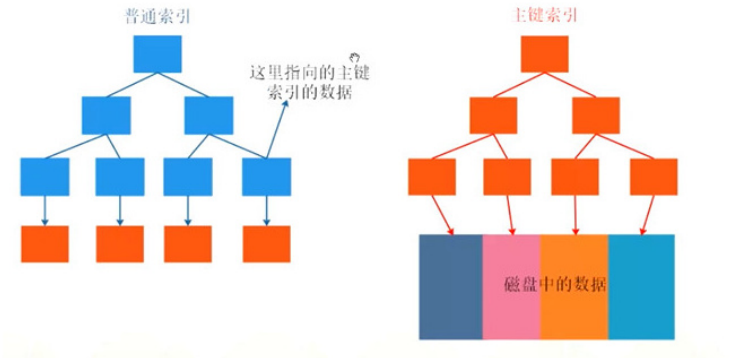


innodb:

alter table stu add index idx_sex(sex);

实际上idx_sex:是sex与id(主键索引)建立的联合索引=>idx_gender(gender,id);

这种设计与innodb中的行锁有关系



三.innodb回表问题

1.回表:主要是普通索引引起的问题,存储引擎为innodb

2.当SQL语句查询的时候用到了普通所索引,获取的数据不存在于索引中,就会出现回表问题.

例如:

```
select id,name where name='shenjian'
select id,name,sex where name='shenjian'
```

多查询了一个属性，为何检索过程完全不同？

InnoDB有两大类索引：

聚集索引(clustered index):InnoDB 聚集索引 的叶子节点存储行记录，因此， InnoDB必须要有，且只有一个聚集索引

（1）如果表定义了PK，则PK就是聚集索引；

（2）如果表没有定义PK，则第一个not NULL unique列是聚集索引；

（3）否则，InnoDB会创建一个隐藏的row-id作为聚集索引；

总结:所以PK查询(主键查询)非常快，直接定位行记录。

普通索引(secondary index):InnoDB 普通索引 的叶子节点存储主键值。

普通索引的查询过程(通常情况下，需要扫码两遍索引树):

1.

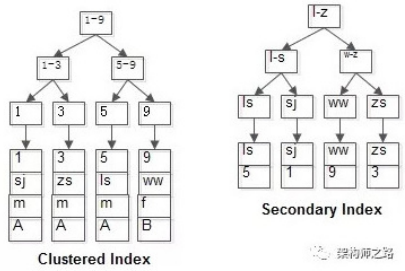
表中有四条记录:

1, shenjian, m, A

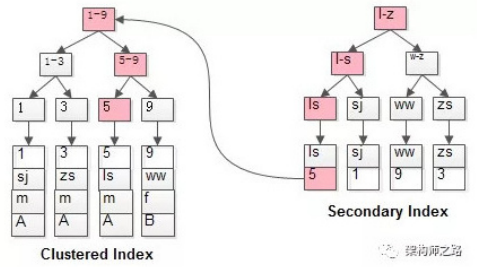
3, zhangsan, m, A

5, lisi, m, A

9, wangwu, f, B



2.



如 **粉红色** 路径，需要扫码两遍索引树：

(1) 先通过普通索引定位到主键值id=5;

(2) 在通过聚集索引定位到行记录;

3.回表本质:根据普通索引查询到主键,又去主键索引去磁盘中检索数据,解决办法就是:

实现索引覆盖,常见的方法是：将被查询的字段，建立到联合索引里去。

4.mysql对于索引检索数据的占比于性能对比

(1)非using index 最高 30%(Extra)

(2)哪些场景可以利用索引覆盖来优化SQL？

全表count查询优化

分页查询...

四.hash索引

(1)hash加密 hash算法

通过hash算法进行加密成32/64位的字符串

mysql会根据我们的字段进行hash计算生成字符串记录磁盘数据的地址

(2)没有准确的排序->无序,离散 特点查询效率高->存在问题不能进行范围查询(where id > 1)

(3)hash索引的创建:

```
alter table customers add index idx_gender_monthsalary using hash(gender);
show indexes from customers;
```