# 一 基本架构

Server层和存储引擎

server层

连接器,分析器,优化器,执行器,查询缓存

Mysql接收查询后,会先查找缓存,如果能命中缓存则直接返回

存储引擎

Innodb,myisam,memory

# 二 索引

### 多列索引

多个单列索引,在做多条件查询的时候性能会很差,因为涉及到额外的连接操作

## 联合索引

联合索引符合左匹配原则,创建了(A,B),再创建(B,A)不属于冗余索引

比如对于(a,b)b只是局部有序的

因此根据最左匹配原则,在遇到范围查询的时候,就会停止匹配

SELECT \* FROM table WHERE a > 1 and b = 2;

这种需要对(b,a)建立索引,因为a是范围查询

SELECT \* FROM table WHERE a = 1 and b = 2 and c = 3;

对于这种查询,mysql会自动优化顺序,所以还是要按照区分度来创建索引

SELECT \* FROM `table` WHERE a = 1 ORDER BY b;

如何建立索引？ 这还需要想？一看就是对(a,b)建索引，当a = 1的时候，b相对有序，可以避免再次排序！ 那么

SELECT \* FROM `table` WHERE a > 1 ORDER BY b;

如何建立索引？ 对(a)建立索引，因为a的值是一个范围，这个范围内b值是无序的，没有必要对(a,b)建立索引。

SELECT \* FROM `table` WHERE a IN (1,2,3) and b > 1;

还是对(a，b)建立索引，因为IN在这里可以视为等值引用，不会中止索引匹配，所以还是(a,b)!

## 聚簇索引

这是一种数据存储方式,存在Innodb引擎中,主键索引为聚簇索引,如果没有主键mysql会选择一个唯一的非空的列作为聚簇索引,如果没有会自己隐式创建一个主键.

索引和数据存放在一起. 只能有一个聚簇索引,其他为二级索引

因此二级索引的回表查找要经历两次,在叶子节点存储了主键值,因此主键不可以过大,否则其他索引数据都会很大

## Index pushdown

减少回表,因此在using index的时候是不会有icp的

在索引存在范围查询的时候出现,例如以下语句

Select first\_name,last\_name,gender from employee where first\_name=’Amer’ and last\_name like ‘B%’

## Count()

两个作用,统计列值的数量或者统计行数.统计列值时不统计NULL

Myisam在做count(\*)的时候是比较快的

Innodb在count(\*) 会访问全部数据量,但是可以using index

# 三 innodb和myisam差别

1. innodb支持事务,myisam不支持
2. Innodb支持外键,myisam不支持
3. Innodb是聚簇索引,myisam不是. innodb索引和数据存放在一个文件,二myisam索引和数据分开存储,索引保存指向数据文件的指针,myisam主键索引和二级索引是独立的
4. Innodb不保存表的行数,所以count(\*)会很慢,myisam反之很快
5. Innodb的最小锁粒度是行,myisam最小锁粒度是表