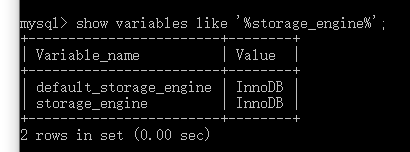
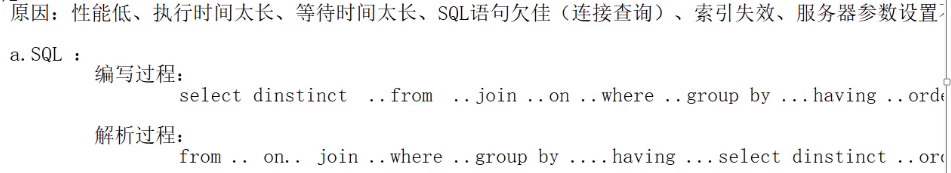
## sql优化

show engines ; 查询mysql数据库支持的引擎是MyISAM(性能优先，表锁) 还是InnoDB（默认，事务优先，行锁）

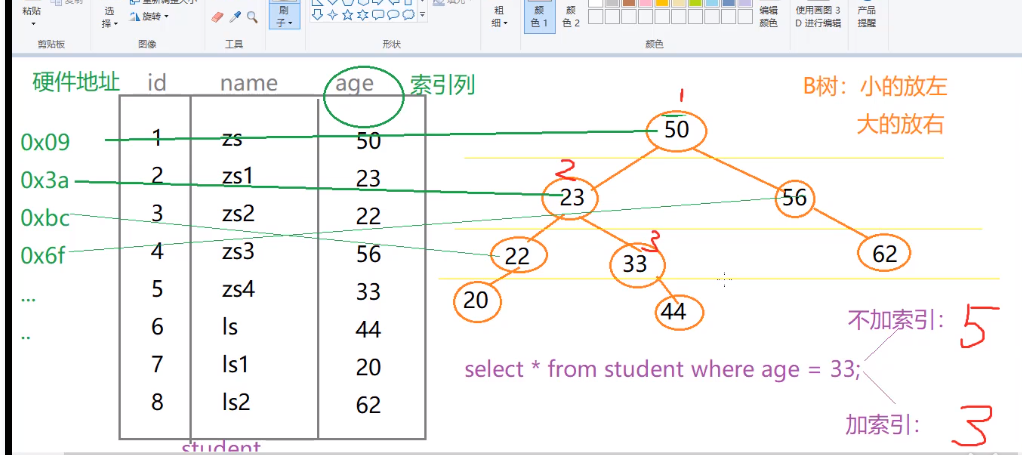
show engines \G ; 查询mysql数据库支持的引擎是MyISAM 还是InnoDB,格式化

show variables like '%storage\_engine%'; 查看当前的数据库引擎





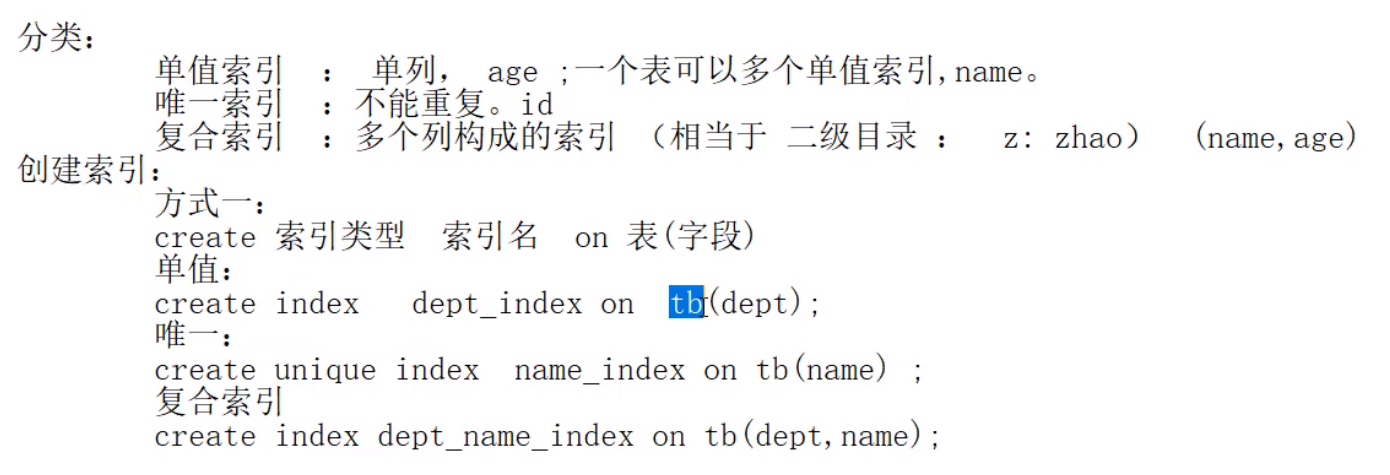
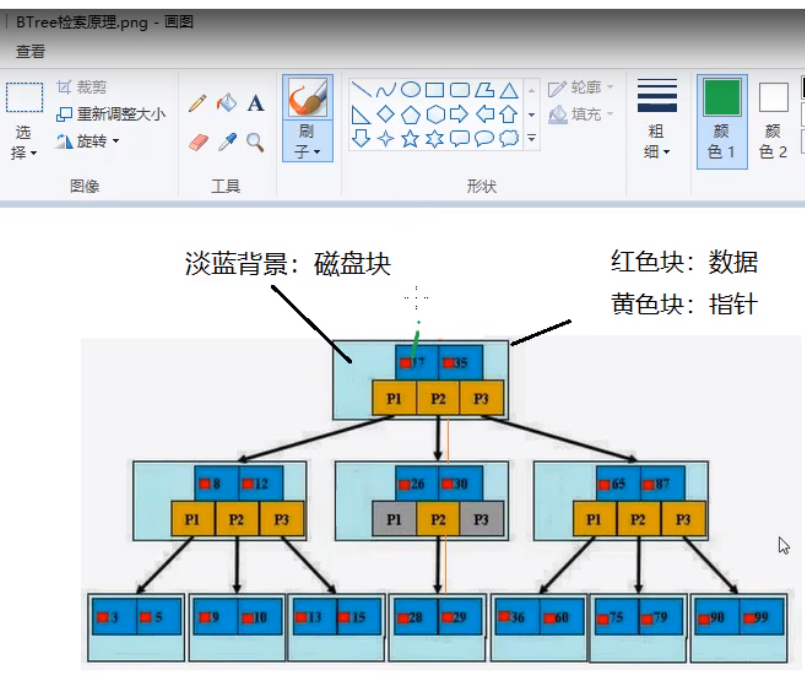
## 索引：数据结构 B树，更改索引前，先删除旧索引。

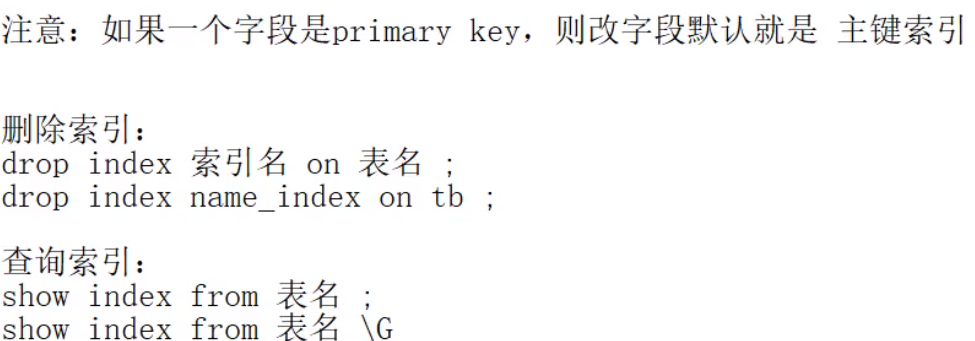
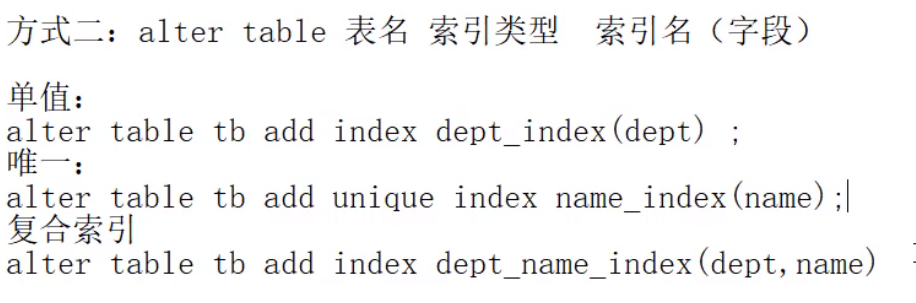


缺点：1.索引本身很大，可以放在内存或者硬盘中(默认) 2.索引不适用的情况 a少量数据 b频繁更新的字段(也会更新索引) c很少使用的字段 3.索引增加查询效率，但是会降低增删改的效率(因为要同步索引)

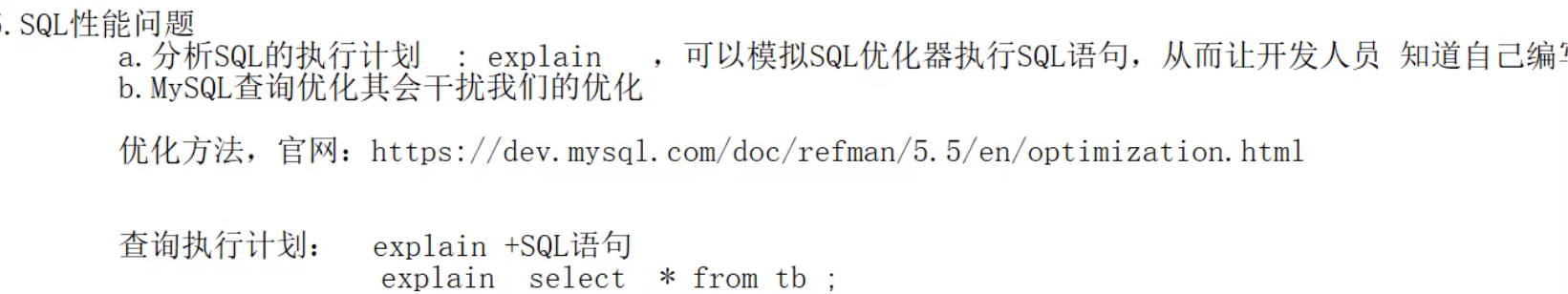
优点：1.提高查询效率，降低io使用率 2. 降低CPU使用率，因为b树索引本身就是一个排好序的结构。

b树检索原理



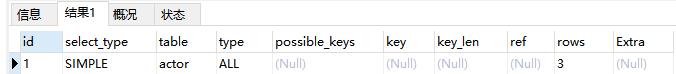


## sql性能问题



1. 使用EXPLAIN关键字可以模拟优化器执行SQL语句，分析查询语句或是结构的性能瓶颈, id越大执行优先级越高，id相同则从上往下执行，id为NULL最后执行。

explain select \* from actor;



* 1. id列的编号是select的序列号,有几个select就有几个id，并且id的顺序是按select出现的顺序增长的。
  2. select type列表示对应行是简单还是复杂的查询。

simple：简单查询。查询不包含子查询和union。

primary：复杂查询中最外层的select

subquery：包含在select中的子查询（不在from子句中）

derived：包含在from子句中的子查询。MySQL会将结果存放在一个临时表中，也称为派生表。

* 1. table列：这一列表示explain的一行正在访问哪个表
  2. type列这一列表示关联类型或访问类型，即MySQL决定如何查找表中的行，查找数据行对应的大概范围。依次从最优到最差的分别为：system>const>eq\_ref>ref>range>index>All
  3. possible\_keys列这一列显示select可能会使用哪些查询来查找。explain时可能会出现possible\_keys有列，而key显示为NULL的情况，这种情况是因为表中的数据不多，MySQL认为索引对此查询帮助不大，选择了全表扫描。  
     如果该列为NULL，则没有相关的索引。这种情况下，可以通过检查where子句看是否可以创造一个适当的索引来提高查询性能，然后用explain查看效果。
  4. key列 这一列显示MySQL实际采用哪个索引对该表的访问
  5. key\_len列 这一列显示了mysql在索引里使用的字节数，通过这个值可以估算出具体使用了索引中的哪些列。
  6. ref列 这一列显示了在key列记录的索引中，表查找值所用到的列或常量，常见的有： const(常量)，字段名等。一般是查询条件或关联条件中等号右边的值，如果是常量那么ref列是const，非常量的话ref列就是字段名。
  7. row列这一列是mysql估计要读取并检测的行数，注意这个不是结果集的行数。
  8. extra列 这一列是额外信息。

Using index：使用覆盖索引（结果集的字段是索引）

Using index condition：查询的列不完全被索引覆盖，where条件中是一个前导的范围

Using where：使用where语句来处理结果，查询的列未被索引覆盖

Using temporary：mysql需要创建一张临时表来处理查询。出现这种情况一般要进行优化，首先要想到是索引优化。

Using filesort：将用外部排序而不是索引排序，数据较小时从内存排序，否则需要在磁盘完成排序。这种情况下一般也是要考虑使用索引来优化的。

select tables optimized away：使用某些聚合函数（比如：max、min）来访问存在索引的某个字段

