## 索引

### 介绍

#### 1. 为什么要有索引

首先，无索引的表，查询时，是按照顺序存续的方法扫描每个记录来查找符合条件的记录，这样很慢，而索引则可以对查询进行性能优化。索引相当于字段的音序表。

#### 2.索引的原理

索引的目的在于提高查询效率，本质是通过不断地缩小想要获取数据的范围来筛选出最终想要的结果，同时把随机事件变成顺序的事件。也就是说，有了这种索引机制，我们可以总是用同一种查找方式来锁定数据。

稍有算法基础的同学会想到搜索树，其平均复杂度是lgN，具有不错的查询性能。但这里我们忽略了一个关键的问题，复杂度模型是基于每次相同的操作成本来考虑的。而数据库实现比较复杂，一方面数据是保存在磁盘上的，另外一方面为了提高性能，每次又可以把部分数据读入内存来计算，因为我们知道访问磁盘的成本大概是访问内存的十万倍左右，所以简单的搜索树难以满足复杂的应用场景。

#### 3.磁盘IO与预读

考虑到磁盘IO是非常高昂的操作，计算机操作系统做了一些优化，当一次IO时，不光把当前磁盘地址的数据，而是把相邻的数据也都读取到内存缓冲区内，因为局部预读性原理告诉我们，当计算机访问一个地址的数据的时候，与其相邻的数据也会很快被访问到。每一次IO读取的数据我们称之为一页(page)。

#### 4.索引的数据结构

b+树

### MySQL的索引管理

### 功能

索引的功能就是加速查找

MySQL中的primary key，unique，联合唯一也都是索引，这些索引除了加速查找外还有约束的功能

### 分类

#### 1.普通索引 index：加速查找

#### 2.唯一索引

主键索引：primary key：加速查找+约束（不为空，且唯一）

唯一索引：unique ：加速查找+约束（唯一）

#### 3.联合索引

-primary key（id，name）联合主键索引

-unique（id，name）：联合唯一索引

-index（id，name）：联合普通索引

#### 4.全文索引 fulltext：用于搜索很长一篇文章的时候，效果最好

### 索引的两大类型

#### Hash类型的索引：查询单条快，查询范围慢

#### Btree类型的索引：b+树，层数越多，数据量指数级增长（innode引擎默认）

#### 不同的存储引擎

InnoDB 支持事务，支持行级别锁定，支持 B-tree、Full-text 等索引，不支持 Hash 索引；

MyISAM 不支持事务，支持表级别锁定，支持 B-tree、Full-text 等索引，不支持 Hash 索引；

#### 建立索引的原则

1) 定义主键的数据列一定要建立索引。

2) 定义有外键的数据列一定要建立索引。

3) 对于经常查询的数据列最好建立索引。

4) 对于需要在指定范围内的快速或频繁查询的数据列;

5) 经常用在WHERE子句中的数据列。

6) 经常出现在关键字order by、group by、distinct后面的字段，建立索引。如果建立的是复合索引，索引的字段顺序要和这些关键字后面的字段顺序一致，否则索引不会被使用。

7) 对于那些查询中很少涉及的列，重复值比较多的列不要建立索引。

8) 对于定义为text、image和bit的数据类型的列不要建立索引。

9) 对于经常存取的列避免建立索引

9) 限制表上的索引数目。对一个存在大量更新操作的表，所建索引的数目一般不要超过3个，最多不要超过5个。索引虽说提高了访问速度，但太多索引会影响数据的更新操作。

#### 创建索引