数据库索引

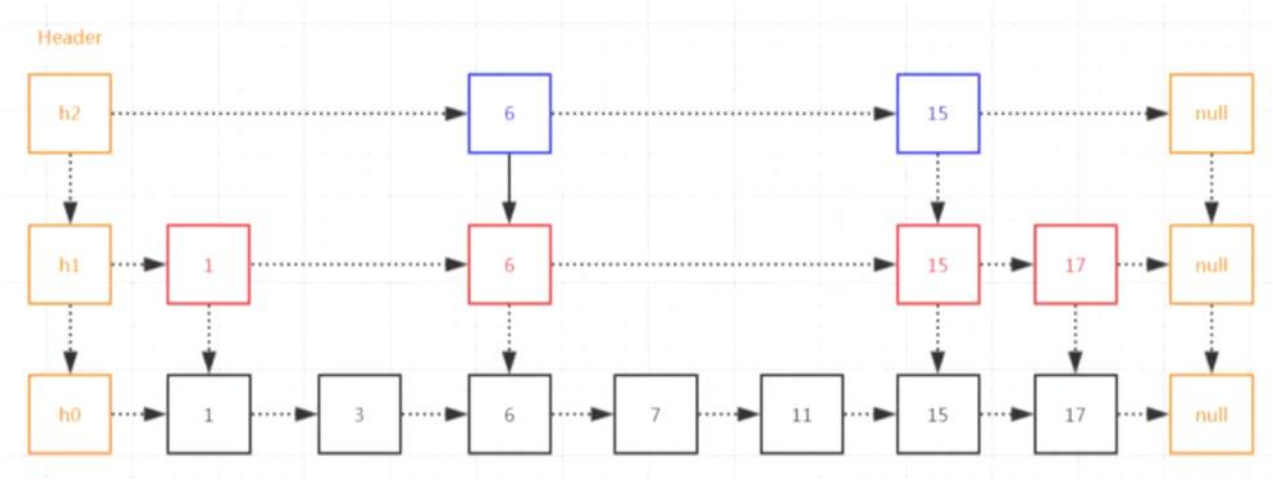
2019年7月16日 星期二 22:27

- 为什么要索引
  - 为了在查询数据库时不再需要一行一行的扫描整个表
  - 缩小查询的行的数目来加快搜索
- 索引是什么
  - 是存储表中一个特定列的值的数据结构（最常见的为B树）
  - 索引是基于表的某一列创建的
  - 关键
    - 索引包含表中某一列的值
    - 并且这些值存储在一个数据结构中
- 索引是如何提高性能的
  - 如果有索引，那么在查询时
    - 用索引去定位名字为“ZHAO”的地方
    - 因为在索引内这些值已经被排序，所以速度会变快
  - 索引存储的是表中相应行的指针，而不是具体的那一行
- 索引的缺点
  - 索引本身需要维护，耗费时间
  - 索引本身占用空间
  - 对表进行维护时还需要对索引进行维护，降低了维护速度
- mysql索引类型
  - normal
    - 表示普通索引
  - unique
    - 表示唯一的，不允许重复的索引
  - full textl
    - 表示全文搜索的索引（用于搜索很长的一篇文章）
- 索引常以B树和B+树构成，详见知乎B树
- 单列索引
  - 常用的一个列字段的索引
- 多列索引
  - 含有多个列字段的索引（name,age,place）
  - where中必须含有首列字段(如name)

跳表

2019年7月17日 星期三　　23:11

- 是一种随机化的数据结构，实质上是一种 可以进行二分查找的有序链表
- 在原有的有序链表的基础上增加了多级索引，通过索引来实现快速查找
-



数据库事务

2019年7月16日 星期二 22:27

事务基本

2019年7月16日 星期二 22:40

- 事务是什么
  - 构成单一逻辑工作单元的操作集合
  - 保证逻辑事务整体要么全部执行成功，要么全部不执行
  - 简化错误恢复并使应用程序更加可靠
  - 一个逻辑单元想要成为事务，必须满足ACID属性
  - 是数据库运行中的逻辑工作单位
- 事务隔离
  - 同一时间，只允许一个事务请求同一数据
  - 不同的事务之间彼此没有任何干扰

事务的四个特性

2019年7月16日 星期二 22:28

- 原子性
  - 事务包含的所有操作要么全部成功，要么全部回滚
  - 事务的操作如果成功就必须完全应用到数据库，操作失败则不能对数据库有任何影响
- 一致性
  - 一个事务执行之前和执行之后都必须处于一致性状态
  - 一致性状态：所有事务对一个数据的读取结果都是相同的
  - 例
    - 拿转账来说，不管A和B之间如何转账，A和B两个用户的存款加起来总数不能被改变
- 隔离性
  - 多个用户并发访问数据库时（比如同时操作同一张表时），事务不能被其他事务的操作所干扰，多个并发事务之间相互隔离
  - 一句话形容：一个事务所做的修改在最终提交之前，对其他事务是不可见的
  - 4个隔离级别
    - 读取未提交内容
      - 最低的隔离级别
        - 一个事务可以读到另一个事务尚未最终提交的结果
    - 读取提交内容
      - 只有在事务提交之后，更新结果才会被其他事物看见
    - 可重复读
      - 同一事务中，对于同一份数据的多次读取结果总是相同的
    - 可串行化
      - 串行化执行，牺牲了系统的并发性
      - 可以解决并发事务的所有问题
- 持久性
  - 一个事务一旦被提交了，那么对数据的改变就是永久性的（即使发生崩溃）

并发一致性问题

2019年7月18日 星期四 22:44

在并发环境下，事务的隔离性很难得到保证，会发生很多并发一致性问题
这里就要用到4种隔离机制去解决

- 丢失修改
  - 2个事务都对一个数据进行了修改，但后修改的事务覆盖了前面的
- 脏读
  - 1事务修改了一个数据，2事务随后读取它
  - 但1事务的操作被撤销
  - 2事务所读取到的错误数据就是脏读
- 不可重复读
  - 2事务读取了一个数据，但这时1事务修改了这个数据
  - 然后2事务又读取了一遍这个数据
  - 导致2事务两次读取结果不同
- 幻影读
  - 1事务读取某个范围内的数据，2事务在这个范围内插入了新的数据
  - 1事务再次读取该范围内数据
  - 导致1事务两次读取的结果不同
- 4种隔离机制的效果（对勾表示解决不了）
  -

隔离级别	脏读	不可重复读	幻影读
未提交读	√	√	√
提交读	×	√	√
可重复读	×	×	√
可串行化	×	×	×

MySQL

2019年7月18日 星期四 22:53

MySQL的引擎

2019年7月18日 星期四 22:58

- 最常用的有两种
- InnoDB
- MyIsam
- 区别

InnoDB	MyIsam
支持事务	不支持事务
性能慢	性能快
不保存行的个数	保存行的个数

- 总结
  - I是事务型的存储引擎，提供了对ACID事务的支持，实现了4中隔离级别，分别是（巴拉巴拉巴拉）
  - M是默认引擎，不支持事务，效率较低。但保存了表的行数，执行count（）时只需要读取行数即可
  - 因此当数据容量很大时，选择I；当读操作远多于写操作时，且不需要事务时，选择M

MVCC机制

2019年7月18日 星期四 22:53

- 是一种并发控制机制
- MySQL的InnoDB存储引擎实现隔离级别的一种方式
    - 可以实现未提交读和可重复读
- 通过保存数据在某个时间点的快照来实现
    - 在每行记录后面保存两个隐藏的列
    - 分别保存这个行的创建版本号和删除版本号

inner join 和 left join

2019年7月17日 星期三 23:35

- left join
  - 返回包括左表中的所有记录，和右表中相关字段相等的记录
- right join
  - 返回右表中的所有记录，和左表中匹配的记录
- inner join
  - 只返回两个表相匹配的记录