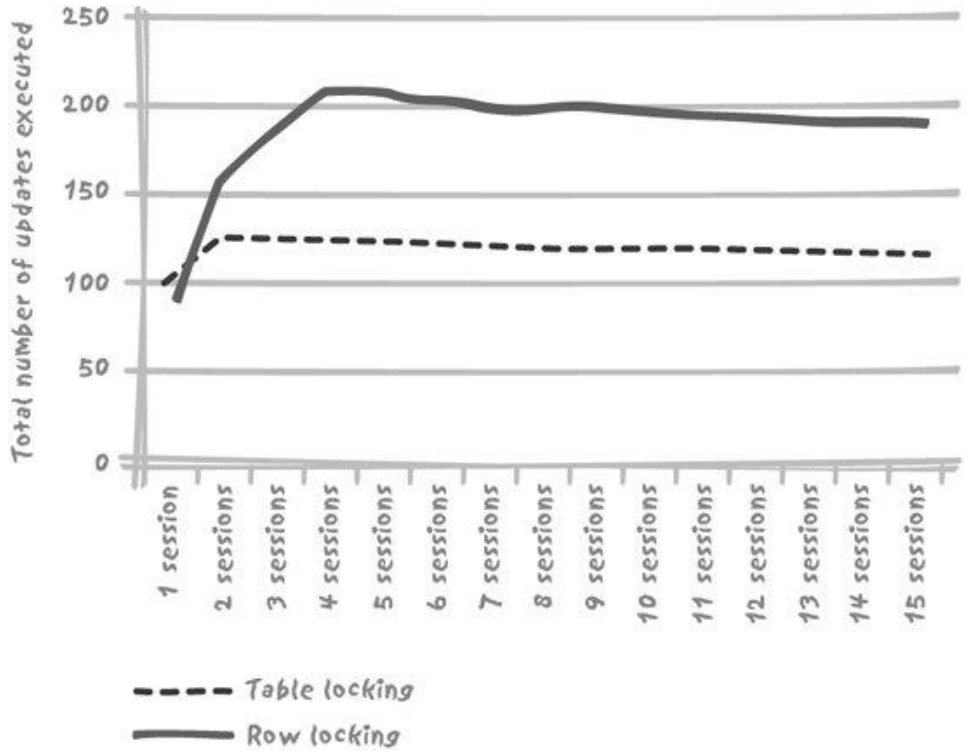
加锁

- 锁的粒度
  - 整个数据库、存储被修改的表的那部分物理单元、要修改的表、包含目标数据的的块和页、包含受影响数据的记录、记录中的字段

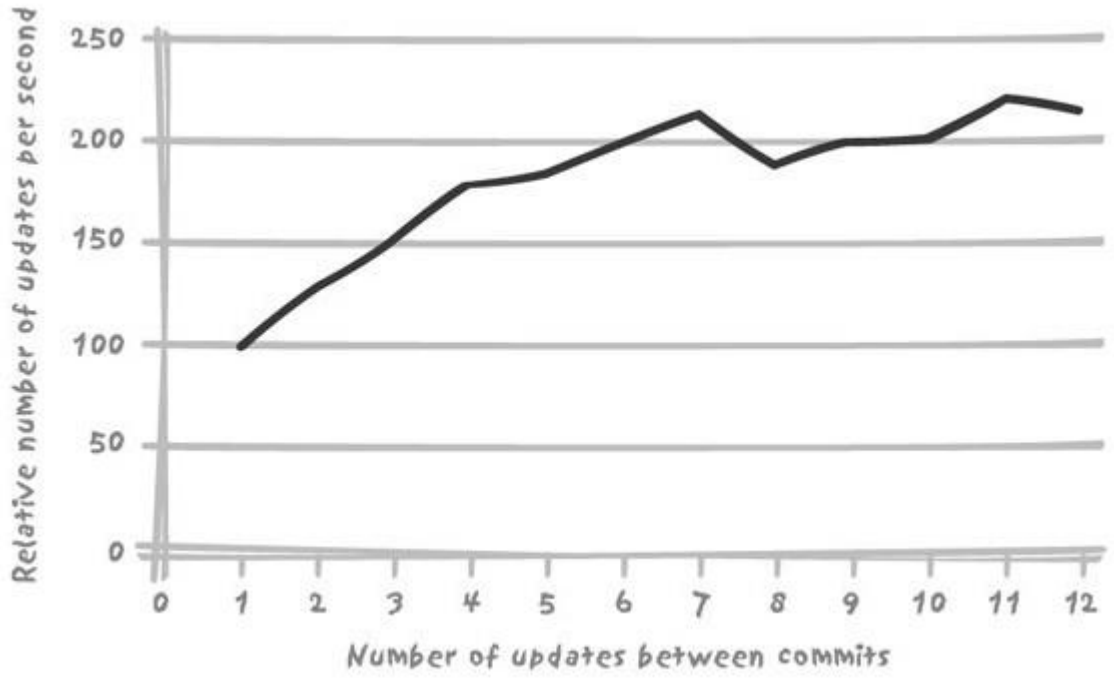


加锁处理

- 不要随便使用表级锁
- 尽量缩短加锁时间
  - Delete没有where，用truncate
- 索引也需要维护
- 语句性能高，未必程序性能高
  - 尽可能避免SQL语句上的循环处理
  - 尽量减少程序和数据库之间的交互次数
  - 充分利用DBMS提供的机制，使跨机器交互的次数降至最少
  - 把所有不重要不必须的SQL语句放在逻辑工作单元之外

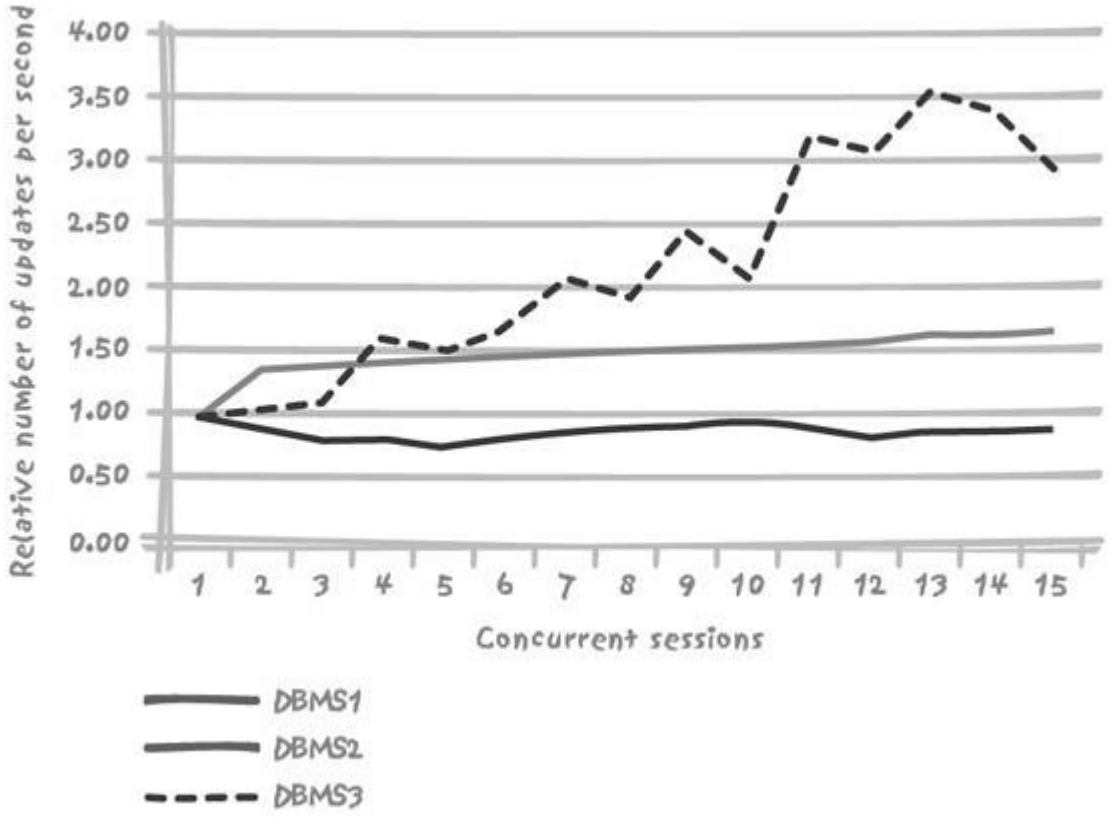
加锁与提交

- 想要使加锁时间最短，必须频繁的提交
- 但如果每个逻辑单元完成后都提交会增加大量开销



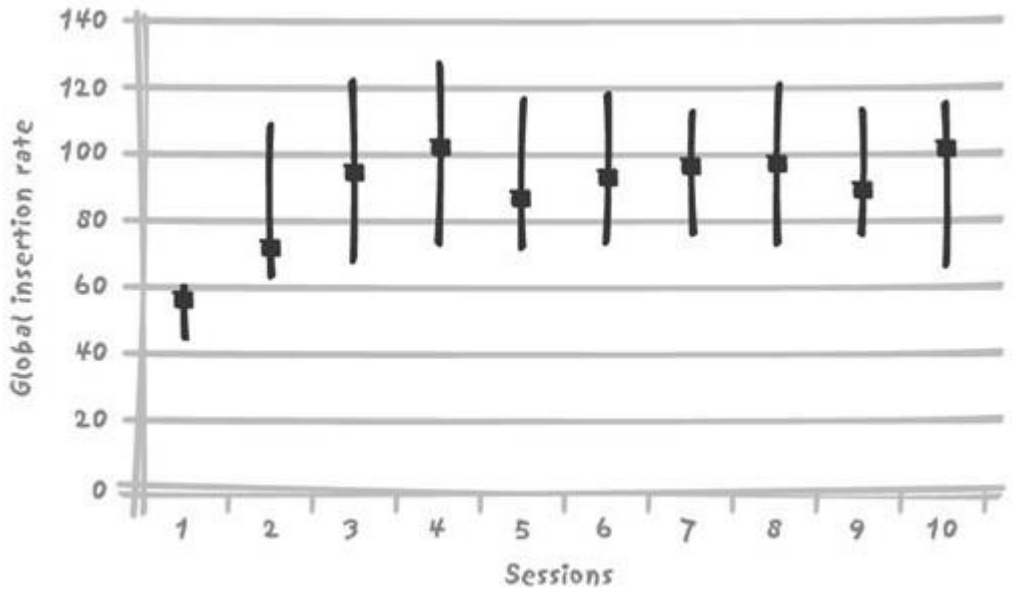
加锁与可伸缩性

- 与表级锁相比，行级锁能产生更佳的吞吐量
- 行级锁大都性能曲线很快达到极限
- 但是不同的数据库产品不太一样



资源竞争

- 插入与竞争
  - Table 是有14个字段、两个唯一性索引的表
  - 主键为系统产生的编号，而真正的键是由短字符串和日期值组成的复合键，必须满足唯一性约束

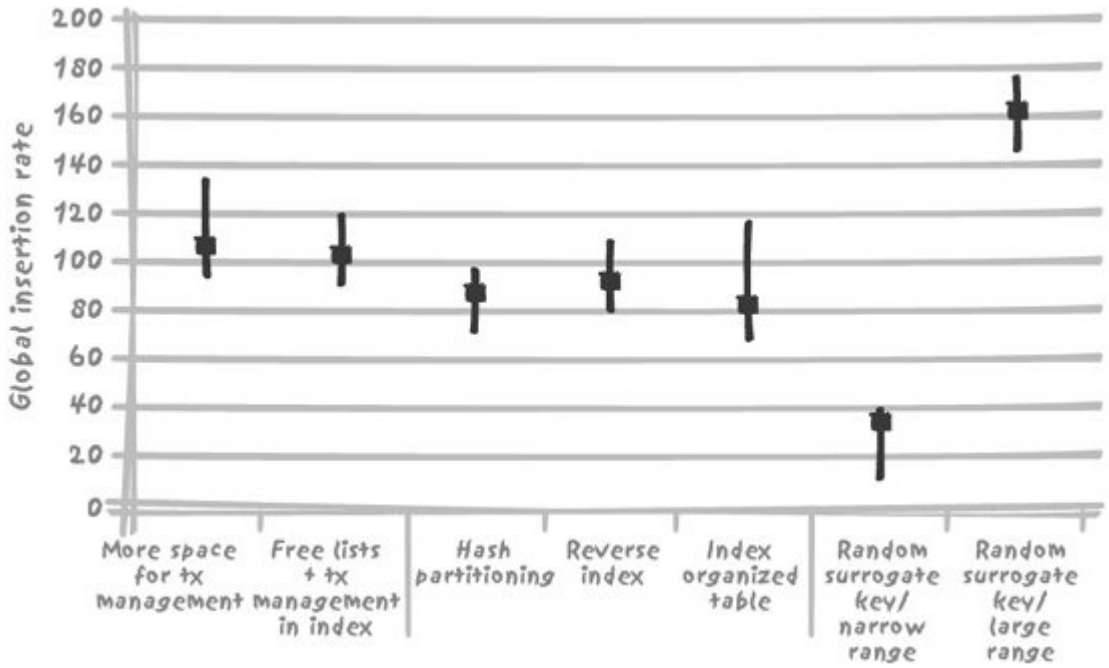


资源竞争

- DBA解决方案
  - 事务空间（Transaction space）
  - 可用列表（Free list）
- 架构解决方案
  - 分区（Partitioning）
  - 逆序索引（Reverse index）
  - 索引组织表（Index organized table）
- 开发解决方案
  - 调节并发数
  - 不使用系统产生值

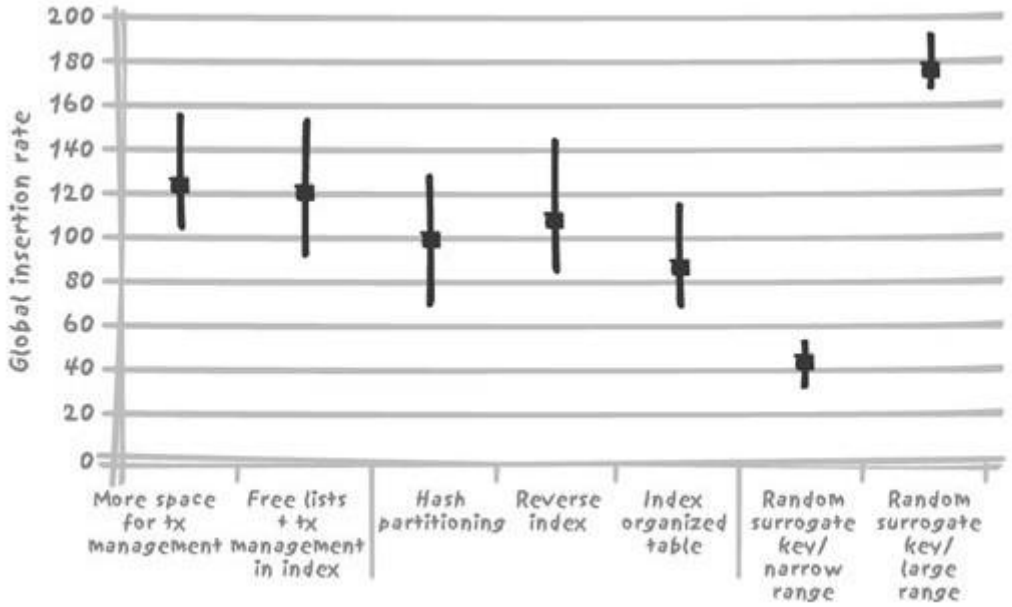
资源竞争

- 限制insert操作之间竞争的技术



资源竞争

- Session数较少时竞争限制技术的表现



资源竞争

- 上述案例的瓶颈是主键索引
- Session的差异说明，有些技术需要已处于饱和状态的CPU提供资源，所以不能带来性能上的改善
- 上述案例避免竞争的方法是避免使用顺序产生的代理键……

- 总结：与加锁不同，数据库竞争是可以改善的。架构师、开发者和DBA都可以从各自的角度改善竞争。

End

下一讲，我们讲讲大数据量的问题。