# MySQL | 两阶段锁协议

作者：douniwan
原文链接：https://ld246.com/article/1582358504156
来源网站：链滴

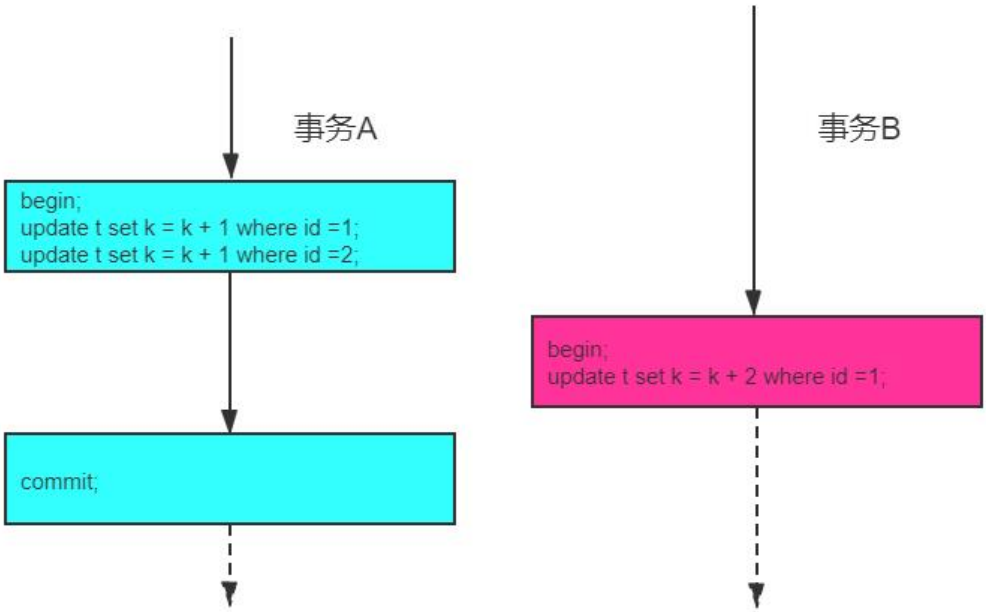
# 两阶段锁协议

## 两阶段锁协议定义

两阶段锁协议是InnoDB中的一种锁提交机制。

## 两阶段锁协议内容

先举个例子，假设有一个表 t，主键是 id，其中一个字段是 k，在下面的操作中，事务 B 的 update 句执行时，会是什么现象呢？




这个问题的结论取决于事务 A 执行完前两条语句后，持有哪些锁，以及在什么时候释放。实际上，事 A 持有两个记录的行锁，都是在 commit 的时候才释放的，所以事务 B 的 update 就会被阻塞，直事务 A 执行 commit 之后，事务 B 才能被继续执行。

也就是说，**在 InnoDB 事务中**，**行锁**是在需要的时候才加上的，但并不是不需要了就立刻释放，需要事**务结束时才释放**，这就是**两阶段锁协议，分为加锁阶段和解锁阶段，所有的 lock 操作都在 unlock 操作之后。**

## 两阶段锁协议的运用

知道两阶段锁协议后，我们就可以，来避免一些并发操作中的锁。如果你的事务中需要锁多个行，要最可能造成锁冲突、最可能影响并发度的锁尽量往后放。

举个例子：

假设你负责实现一个电影票在线交易业务，顾客 A 要在影院 B 购买电影票。我们简化一点，这个业需要涉及到以下操作：

1. 从顾客 A 账户余额中扣除电影票价；
2. 给影院 B 的账户余额增加这张电影票价；
3. 记录一条交易日志。

也就是说，要完成这个交易，我们需要 update 两条记录，并 insert 一条记录。当然，为了保证交易原子性，我们要把这三个操作放在一个事务中。那么，你会怎样安排这三个语句在事务中的顺序呢？

分析解决：

试想如果同时有另外一个顾客 C 要在影院 B 买票，那么这两个事务冲突的部分就是语句 2了。因为它要更新同一个影院账户的余额，需要修改同一行数据。根据两阶段锁协议，不论你怎样安排语句顺序，所有的操作需要的行锁都是在事务提交的时候才释放的。所以，如果你把语句 2 安排在最后，比如按 3、1、2 这样的顺序，那么影院账户余额这一行的锁时间就最少。这就最大程度地减少了事务之间的等待，提升了并发度。