58 多个事务更新同一行数据时，是如何加锁避免脏写的?

**多个事务更新同一行数据时，是如何加锁避免脏写的?**

---

之前我们已经用很多篇幅给大家讲解了多个事务并发运行的时候，如果同时要读写一批数据，此时读和写时间的关系是如何协调的，毕竟要是你不协调好的话，可能就会有脏读、不可重复读、幻读等一系列的问题。

简单来说，脏读、不可重复读、幻读，都是别人在更新数据的时候，你怎么读的问题，读的不对，那就有问题，读的方法对了，那就不存在一系列问题了。

而你要解决一系列的数问题，其实就是依靠之前我们给大家讲的那套，基于undo log版本链条以及ReadView实现的mvcc机制。

现在开始我们接下来要用一系列的篇幅来研究另外一个问题了，那就是当有多个事务同时并发更新一行数据的时候，不就是会有脏写的问题吗?

我们之前讲过，脏写是绝对不允许的，那么这个脏写是靠什么防止的呢?

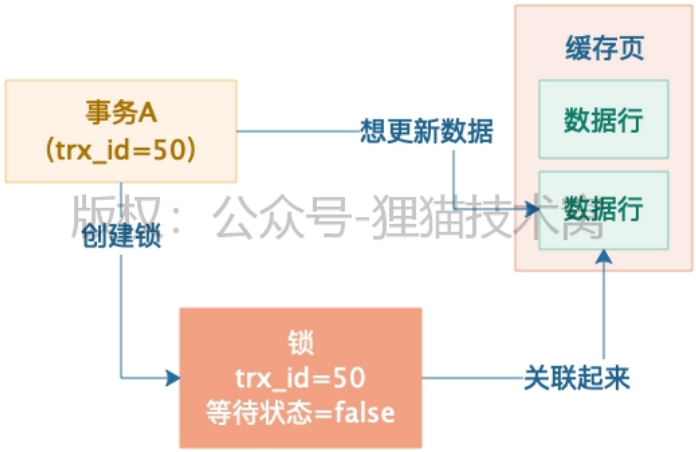
说白了，就是靠锁机制，依靠锁机制让多个事务更新一行数据的时候串行化，避免同时更新一行数据，今天我们就先对数据库的锁机制做一个初步的入门讲解。

在MySQL里，假设有一行数据摆在那儿不动，此时有一个事务来了要更新这行数据，这个时候他会先琢磨一下，看看这行数据此时有没有人加锁?

一看没人加锁，太好了，说明他是第一个人，捷足先登了。

此时这个事务就会创建一个锁，里面包含了自己的trx_id和等待状态，然后把锁跟这行数据关联在一起。

同时大家应该还记得，更新一行数据必须把他所在的数据页从磁盘文件里读取到缓存页里来才能更新的，所以说，此时这行数据和关联的锁数据结构，都是在内存里的，大家要明确这一点，如下图。





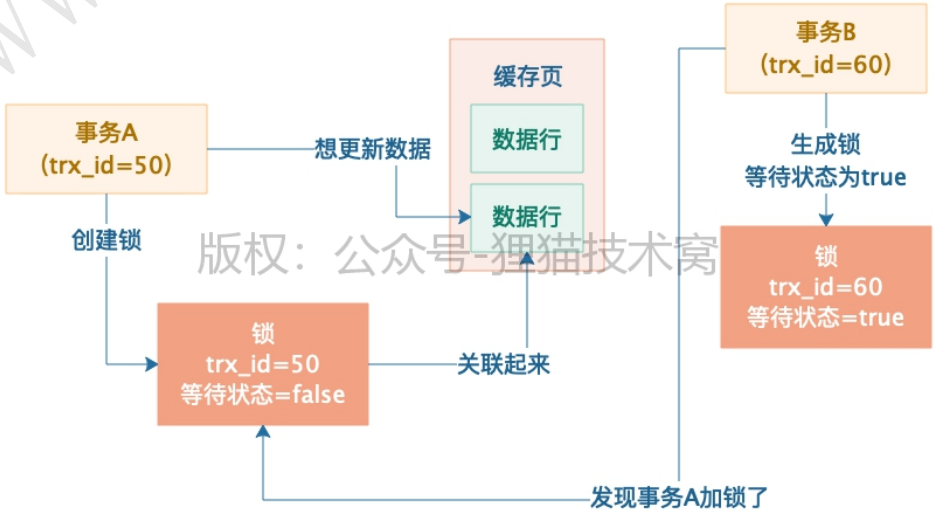

大家注意看上面的那个图，因为事务A给那行数据加了锁，所以此时就可以说那行数据已经被加锁了

那么既然被加锁了，此时就不能再让别人访问了！如果有朋友对加锁的概念不了解，可能是对编程语言不太了解，其实这个就跟Java里的加锁是一个概念。

现在呢，有另外一个事务B过来了，这个事务B就也想更新那行数据，此时就会检查一下，当前这行数据有没有别人加锁

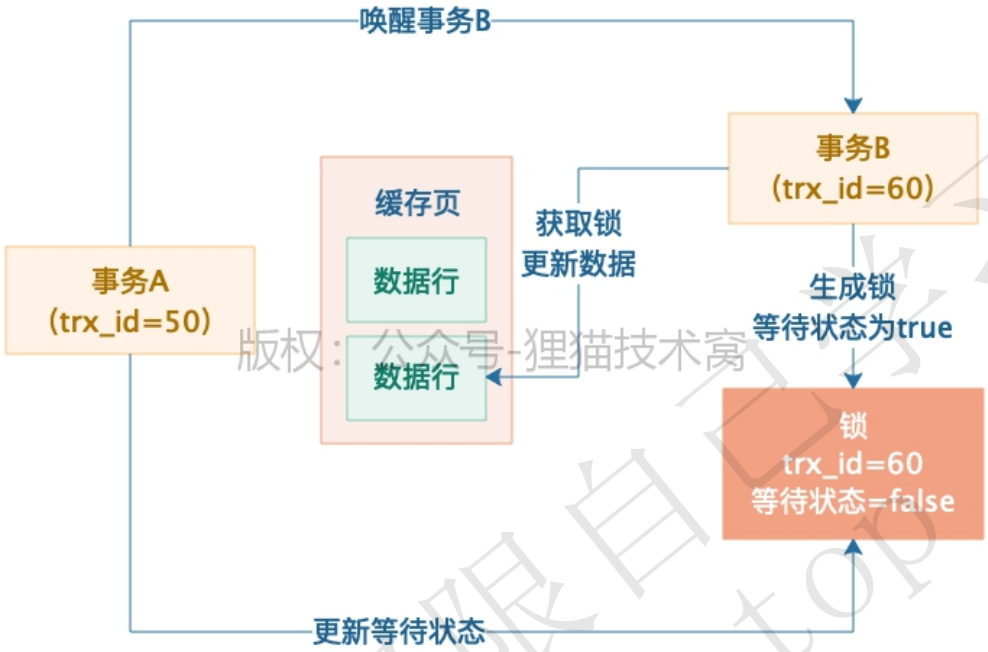
然而他一下子发现，真是糟糕啊，事务A这家伙太不地道了，居然抢先给这行数据加锁了，这怎么办呢？

事务B这个时候一想，那行，我也加个锁，然后等着排队不就得了，这个时候事务B也会生成一个锁数据结构，里面有他的trx_id，还有自己的等待状态，但是他因为是在排队等待，所以他的等待状态就是true了，意思是我在等着呢，如下图。



接着事务A这个时候更新完了数据，就会把自己的锁给释放掉了。锁一旦释放了，他就会去找，此时还有没有别人也对这行数据加锁了呢？他会发现事务B也加锁了

于是这个时候，就会把事务B的锁里的等待状态修改为false，然后唤醒事务B继续执行，此时事务B就获取到锁了，如下图。






上述就是MySQL中锁机制的一个最基本的原理，大家可以先好好理解一下，其实是跟Java里的锁机制，思路是完全类似的，从这种简单的锁里可以引申出很多其他的概念，比如读写锁，共享锁，独占锁，公平锁，非公平锁，等等。Java里的锁，也同样具备这些锁的概念。

**End**