在读提交的事务隔离级别下，MVCC 机制是如何工作的?

原创 刘进坤 菜鸟飞呀飞 2020-11-22 22:47

前言

接前两篇文章:《一文搞懂 undo log 版本链与 ReadView 机制如何让事务读取到该读的数据》 和 《在 MySQL 中是如何通过 MVCC 机制来解决不可重复读和幻读问题的？》，本文接下来将介绍**「在读提交隔离级别下」**，MySQL 是如何解决脏读问题的？以及为什么会存在不可重复读的问题？

本文的内容严重依赖前两篇文章的知识，建议读者先阅读前两篇文章。

从上篇文章中我们可以知道，**「在可重复读隔离级别下」**，当事务开启时，会创建一个 ReadView 视图（什么是 ReadView 视图，可以去阅读上上篇文章），然后在整个事务期间内，都会依照这个视图来决定当前线程能读取到什么数据，**「且在当前事务期间内，出现多次查询时，都不会重新创建 ReadView 视图」**。

「而在读提交隔离级别下，在事务范围内，则会在每次查询前，都会重新创建 ReadView 视图」。

如何解决脏读问题

脏读问题指的是**「一个事务读到了另一个事务未提交的数据」**，如：事务 B 修改了数据，但是事务 B 还未提交，接着事务 A 去读取数据，读取到的是事务 B 刚修改的数据，接着事务 B 将数据回滚了，这就造成了事务 A 的脏读。

「在读提交隔离级别下，不存在脏读问题。」 为什么呢？答案依旧是 ReadView 机制。

假设现有一条数据，它的 row_trx_id=10，数据的值为 data0，它的 roll_pointer 指针为 null。

图1

假设现在有事务 A 和事务 B 并发执行，事务 A 的事务 id 为 20，事务 B 的事务 id 为 30。

现在事务 B 去修改数据，将事务修改为 data_B，然后不提交事务。虽然不提交事务，但是仍然会记录一条 undo log，因此这条数据的 undo log 的版本链就有两条记录了，新的这条 undo log 的 roll_pointer 指针会指向前一条 undo log，示意图如下。

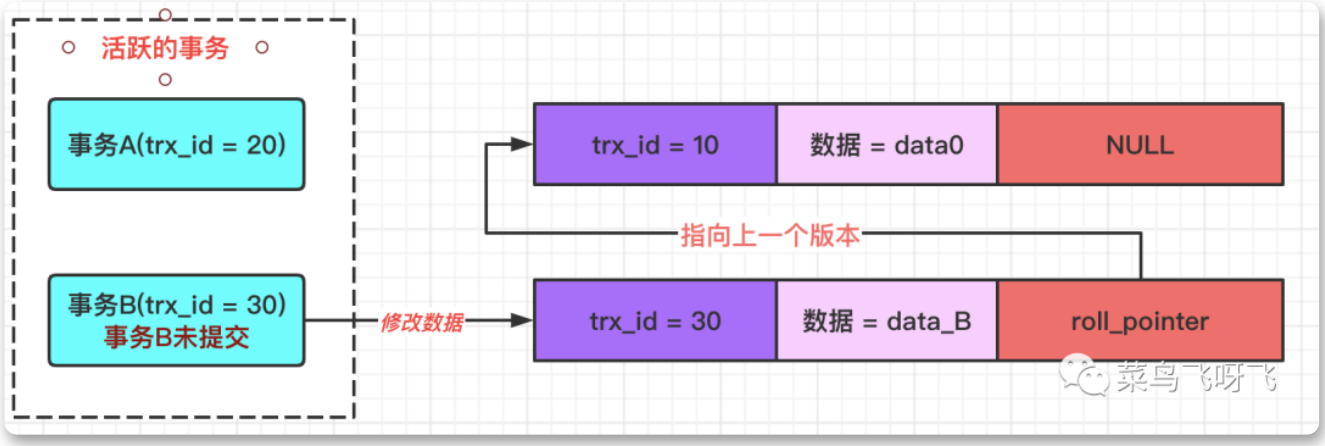

图2

接着事务 A(trx_id=20)去读取数据，那么此时 MySQL 会为事务 A 产生一个 ReadView，此时 ReadView 的内容如下：m_ids=[20,30]，min_trx_id=20，max_trx_id=31，creator_trx_id=20。

那么在 undo log 版本链中，数据最新版本的事务 id 为 30，**「这个值处于事务 A 的 ReadView 里 min_trx_id 和 max_trx_id 之间」**，并且在 m_ids 数组中包含有 30，这表示这个版本的数据是和自己同一时刻启动的事务修改的，因此这个版本的数据，数据 A 读取不到。所以需要沿着 undo log 的版本链向前找，接着会找到该行数据的上一个版本，也就是 trx_id=10 的版本，由于这个版本的数据的 trx_id=10，小于 min_trx_id 的值，因此事务 A 能读取到该版本的值，即事务 A 读取到的值是 data0。

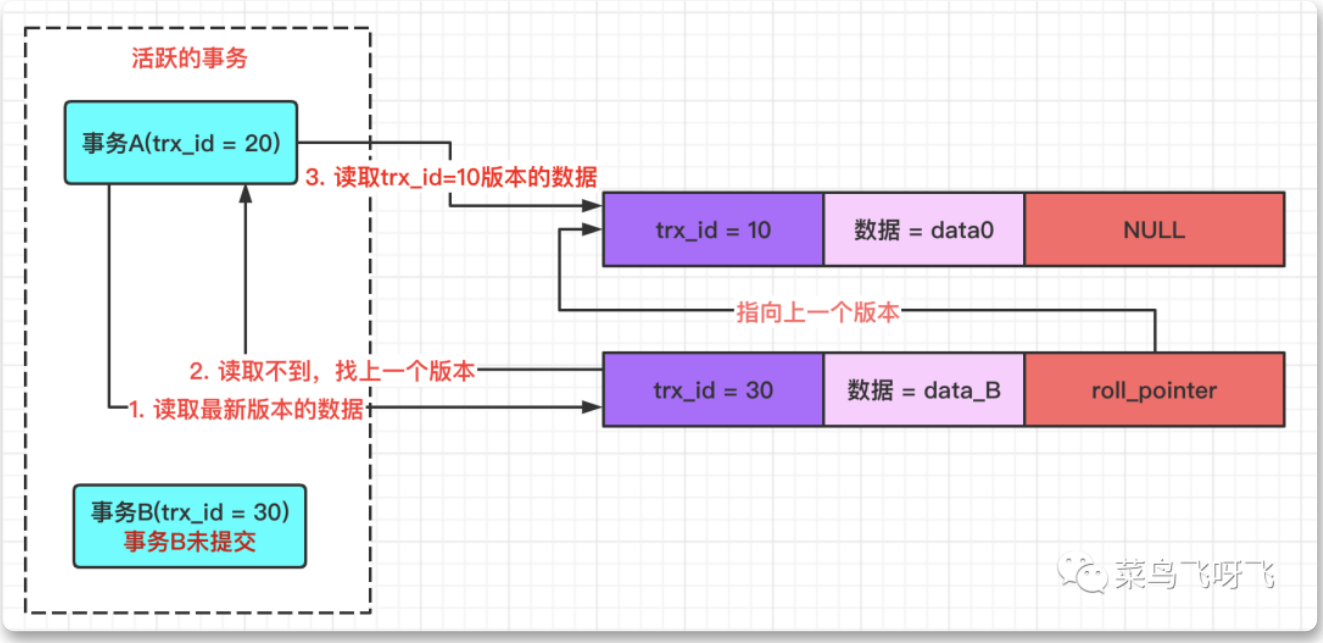

图3

这样依靠 ReadView 机制，在读提交隔离级别下就解决了脏读问题，避免了一个事务读取到另一个未提交事务所修改的数据。

存在不可重复读问题

「不可重复读指的是，在一个事务内，进行多次查询，前后查到的数据不一样。」

「在读提交隔离级别下，存在不可重复读的问题。」

为什么呢？**「原因就是在读提交隔离级别下，在一个事务内，每次查询，都会重新创建一个新的 ReadView 视图。」**

还是接着上面的例子说明。由于事务 B 将数据修改为 data_B 后，未提交事务，因此事务 A 在第一次查询时，不能读取到 data_B，只能读取到 data0。

当事务 A 第一次查询结束后，事务 B 提交事务(也就是事务 B 结束了)，然后事务 A 发起第二次查询，由于此时是读提交隔离级别，因此此时 MySQL 会为事务 A 再重新创建一个

ReadView 视图。

此时 ReadView 的内容如下：m_ids=[20]，min_trx_id=20，max_trx_id=31，creator_trx_id=20。

注意，此时由于事务 B 已经提交了，所以系统中活跃的事务就没有 30 了，因此事务 A 生成的视图中，m_ids 数组中只有 20，没有 30。

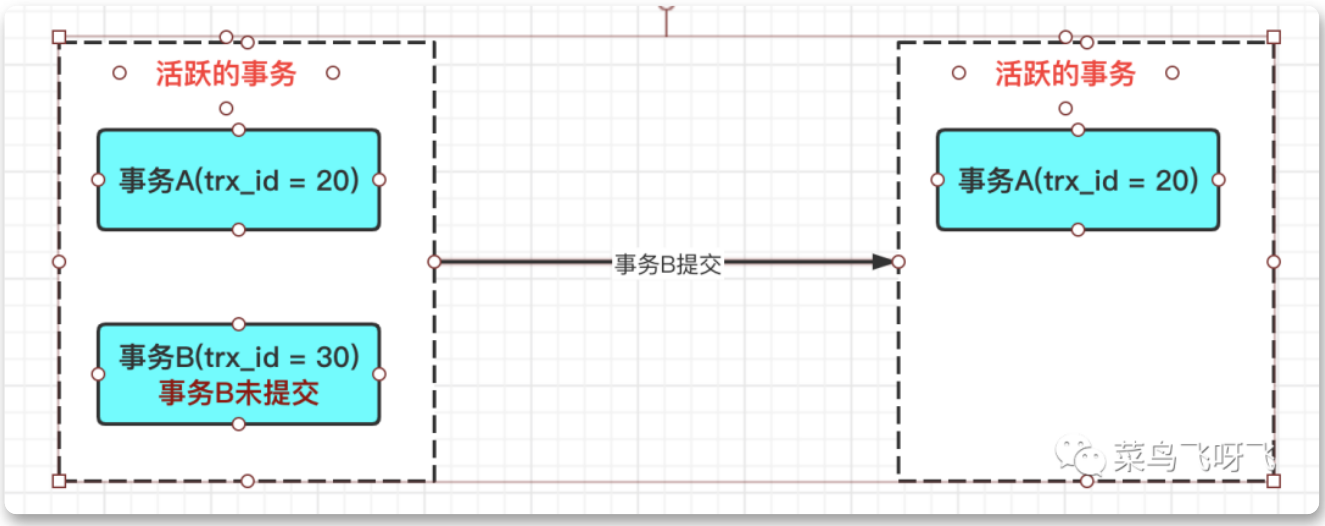

图4

那么在 undo log 版本链中，数据最新版本的事务 id 为 30，**「这个值处于事务 A 的 ReadView 里 min_trx_id 和 max_trx_id 之间」**，因此还需要判断这个数据版本的值是否在 m_ids 数组中，结果发现，30 不在 m_ids 数组中，这表示这个版本的数据虽然是和自己在同一时刻启动的事务所修改的，但是这个事务已经提交了，因此这个版本的数据，数据 A 能读取到，也就是说 A 这次查询到的数据为 data_B。

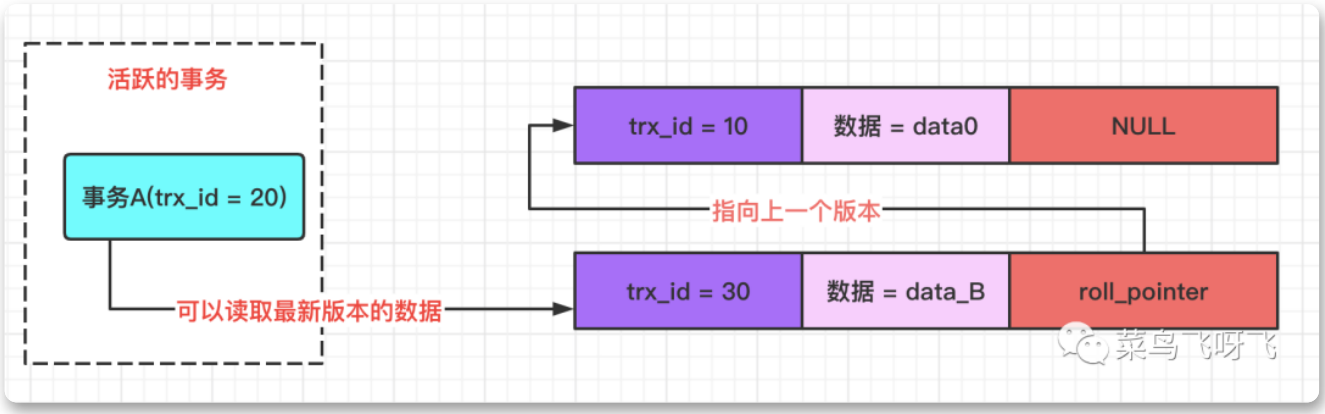

图5

在事务 A 中，第一次读取到的数据是 data0，第二次读取到的数据为 data_B，同一个事务内，前后两次读取到数据不一致。也就是说在读提交隔离级别下，出现了不可重复读的问题。

总结

本文结合 ReadView 机制，介绍了在 MySQL 的读提交隔离级别下是如何解决脏读问题的，以及为什么会存在不可重复读的问题。

「其核心原理在于 ReadView 机制以及在读提交隔离级别下，每次查询都会为事务重新创建 ReadView 视图。」

相关文章

- 数据库中事务的几种隔离级别分别解决了哪些问题
- 一文搞懂 undo log 版本链与 ReadView 机制如何让事务读取到该读的数据
- 在 MySQL 中是如何通过 MVCC 机制来解决不可重复读和幻读问题的？