MySQL 是如何解决幻读的

IT哈哈 昨天

一、什么是幻读

在一次事务里面，多次查询之后，结果集的个数不一致的情况叫做幻读。而多出来或者少的哪一行被叫做 幻行

二、为什么要解决幻读

在高并发数据库系统中，需要保证事务与事务之间的隔离性，还有事务本身的一致性。

三、MySQL 是如何解决幻读的

如果你看到了这篇文章，那么我会默认你了解了 脏读 、不可重复读与可重复读。

1. 多版本并发控制（MVCC）（快照读/一致性读）

多数数据库都实现了多版本并发控制，并且都是靠保存数据快照来实现的。以 InnoDB 为例，每一行中都冗余了两个字断。一个是行的创建版本，一个是行的删除（过期）版本。具体的版本号（trx_id）存在 information_schema.INNODB_TRX 表中。版本号（trx_id）随着每次事务的开启自增。

事务每次取数据的时候都会取创建版本小于当前事务版本的数据，以及过期版本大于当前版本的数据。

普通的 select 就是快照读。

```
select * from T where number = 1;
```

原理：将历史数据存一份快照，所以其他事务增加与删除数据，对于当前事务来说是不可见的。

2. next-key 锁（当前读）

next-key 锁包含两部分

- 记录锁（行锁）

- 间隙锁

记录锁是加在索引上的锁，间隙锁是加在索引之间的。（思考：如果列上没有索引会发生什么？）

```
select * from T where number = 1 for update;
select * from T where number = 1 lock in share mode;
insert
update
delete
```

原理：将当前数据行与上一条数据和下一条数据之间的间隙锁定，保证此范围内读取的数据是一致的。

其他：MySQL InnoDB 引擎 RR 隔离级别是否解决了幻读

引用一个 github 上面的评论 地址：

> Mysql官方给出的幻读解释是：只要在一个事务中，第二次select多出了row就算幻读。
>
> a事务先select，b事务insert确实会加一个gap锁，但是如果b事务commit，这个gap锁就会释放（释放后a事务可以随意dml操作），a事务再select出来的结果在MVCC下还和第一次select一样，接着a事务不加条件地update，这个update会作用在所有行上（包括b事务新加的），a事务再次select就会出现b事务中的新行，并且这个新行已经被update修改了，实测在RR级别下确实如此。
>
> 如果这样理解的话，Mysql的RR级别确实防不住幻读

有道友回复 地址：

> 在快照读读情况下，mysql通过mvcc来避免幻读。
>
> 在当前读读情况下，mysql通过next-key来避免幻读。
>
> select * from t where a=1;属于快照读
>
> select * from t where a=1 lock in share mode;属于当前读
>
> 不能把快照读和当前读得到的结果不一样这种情况认为是幻读，这是两种不同的使用。所以我认为mysql的rr级别是解决了幻读的。

先说结论，MySQL 存储引擎 InnoDB 隔离级别 RR 解决了幻读问题。

如引用一问题所说，T1 select 之后 update，会将 T2 中 insert 的数据一起更新，那么认为多出来一行，所以防不住幻读。看着说法无懈可击，但是其实是错误的，InnoDB 中设置了 快照读 和 当前读 两种模式，如果只有快照读，那么自然没有幻读问题，但是如果将语句提升到当前读，那么 T1 在 select 的时候需要用如下语法： select * from t for update (lock in share mode) 进入当前读，那么自然没有 T2 可以插入数据这一回事儿了。

注意

1. next-key 固然很好的解决了幻读问题，但是还是遵循一般的定律，隔离级别越高，并发越低。