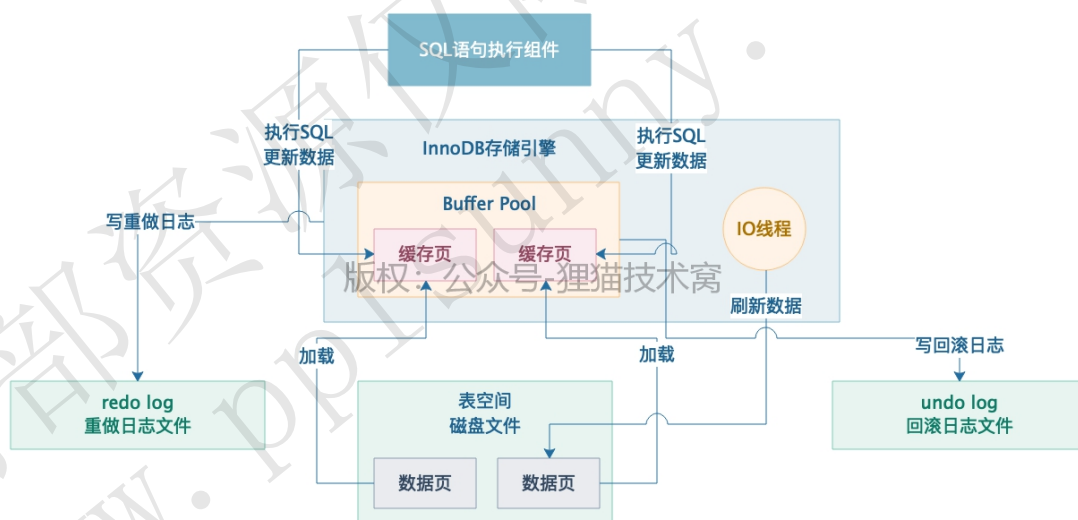


昨天我们讲解了undo log回滚日志的作用，说白了，就是你执行事务的时候，里面很多INSERT、UPDATE和DELETE语句都在更新缓存页里的数据，但是万一事务回滚，你必须要有每条SQL语句对应的undo log回滚日志，根据回滚日志去恢复缓存页里被更新的数据。

比如你执行了INSERT语句，那么你的undo log必须告诉你插入数据的主键ID，让你在回滚的时候可以删除这条数据；

如果你执行了DELETE语句，那么你的undo log必须记录下来被删除的数据，回滚的时候就得重新插入一条数据；

如果你执行了UPDATE语句，那么你必须记录下来修改之前的数据，回滚的时候就得把数据给更新回去，如下图所示。



那么今天我们就一起来看看这个INSERT语句的undo log日志到底长什么样子呢？

INSERT语句的undo log的类型是TRX\_UNDO\_INSERT\_REC，这个undo log里包含了以下一些东西：

- 这条日志的开始位置
- 主键的各列长度和值
- 表id
- undo log日志编号
- undo log日志类型
- 这条日志的结束位置

接下来我们来给大家解释一下，首先，一条日志必须得有自己的一个开始位置，这个没什么好说的吧？

那么主键的各列长度和值是什么意思？大家都知道，你插入一条数据，必然会有一个主键！

如果你自己指定了一个主键，那么可能这个主键就是一个列，比如id之类的，也可能是多个列组成的一个主键，比如“id+name+type”三个字段组成的一个联合主键，也是有可能的。

所以这个主键的各列长度和值，意思就是你插入的这条数据的主键的每个列，他的长度是多少，具体的值是多少。即使你没有设置主键，MySQL自己也会给你弄一个row\_id作为隐藏字段，做你的主键。

接着是表id，这个就不用多说了，你插入一条数据必然是往一个表里插入数据的，那当然得有一个表id，记录下来是在哪个表里插入的数据了。

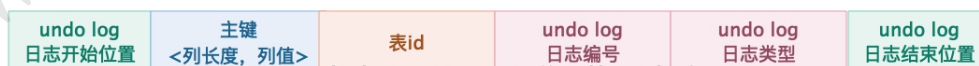
undo log日志编号，这个意思就是，每个undo log日志都是有自己的编号的。

而在一个事务里会有多个SQL语句，就会有多个undo log日志，在每个事务里的undo log日志的编号都是从0开始的，然后依次递增。

至于undo log日志类型，就是TRX\_UNDO\_INSERT\_REC，insert语句的undo log日志类型就是这个东西。

最后一个undo log日志的结束位置，这个自然也不用多说了，他就是告诉你undo log日志结束的位置是什么。

那么接着我们用一个图画一下这个INSERT语句的undo log回滚日志的结构，大家来看一眼，感受一下。



版权：公众号-狸猫技术窝

大家可以想象一下，有了这条日志之后，剩下的事儿就好办了

万一要是你现在的buffer pool的一个缓存页里插入了一条数据了，执行了insert语句，然后你写了一条上面的那种undo log，现在事务要是回滚了，你直接就把这条insert语句的undo log拿出来。

然后在undo log里就知道在哪个表里插入的数据，主键是什么，直接定位到那个表和主键对应的缓存页，从里面删除掉之前insert语句插入进去的数据就可以了，这样就可以实现事务回滚的效果了！

好了，今天先初步的看一下insert语句的undo log回顾日志，delete语句和update语句的回滚日志我们暂时就不细讲了，其实大家应该都能想象到他们是如何实现的。

**End**

内部资源仅限自己学习  
www.pp1sunny.top