1. undo日志
2. 概述

当事务执行过程中突然中止，为了保证事务的原子性，需要回滚回原来的样子，每当要对一条记录进行改动时，都需要记录下一些信息，为了回滚而记录的东西称为撤销日志，即undo日志。

1. 事务id
2. 分配事务id

如果某个事务在执行过程中对某个表执行了增删改操作，那么InnoDB就会给它分配一个唯一的事务id。对于只读事务，只有在它第一次对某个用户创建的临时表执行增删改操作时，才会分配事务id；对于读写事务，只有在它第一次对某个表执行增删改操作时，才会分配事务id。有时，虽然我们开启了一个读写事务，但是该事务中都是查询语句，就不会被分配事务id了。

1. 事务id的生成

服务器在内存中维护一个全局变量作为事务id，只要分配了一个就会增加1，每当这个变量的值为256的倍数，就将该值刷新到系统空间表中页号为5的页面中一个名为Max\_Trx ID的属性中。InnoDB聚簇索引记录的行格式中有个trx\_id隐藏列，记录的就是对这个聚簇索引记录进行改动的语句所在的事务id。

1. undo日志的格式

一般每对一条记录进行一次改动，就对应着一条undo日志，但在某些更新中也会对应着两条undo日志，一个事务在执行过程中会生成很多undo日志，从0开始编号，记为undo\_no。

1. INSERT操作对应的undo日志

插入操作，最终的结果都是将这条记录放到一个索引页中，如果希望回滚这个操作，就把这条记录删除，即在写undo日志时，只要记下这条记录的主键信息即可，不论主键包含几个列， 记录下每个相关列的存储空间大小和真实值。虽然说插入记录是向聚簇索引和二级索引都插入了一条记录，但是写入undo日志只针对聚簇索引，因为可以根据主键找到二级索引中的记录。

在聚簇索引记录中还有一个隐藏列roll\_pointer，它是指向记录对应的undo日志的指针。

1. DELETE操作对应的undo日志

我们知道，插入到页面中的记录会根据记录头信息中的next\_record属性组成一个单向链表，称为正常记录链表；被删除的记录也会根据next\_record组成一个垃圾链表，该链表中的存储空间可以被重用。Page Header中有一个PAGE\_FREE 属性，指向垃圾链表的头结点。

当使用DELETE操作删除一个记录时，将正常记录链表中的最后一条记录删除，这个删除过程分为两步：

1. 将记录的delted\_flag标识设为1，这个阶段称为delete mark，此时的记录既不在正常记录链表中，也不再垃圾链表中，处于中间阶段。
2. 当该删除语句所在的事务提交后，会有专门的线程将该记录加入到垃圾链表中，这个阶段称为purge。此后这条记录算作真正被删除了，占用的存储空间也可以重用了。注意，加入到垃圾链表采用的是头插法。

由于事务提交后，我们无需回滚事务，对于DELETE操作的回滚，只需要考虑对delete mark过程进行回滚即可。对一条记录进行delete mark操作之前，需要把该记录的trx\_id和roll\_pointer的旧值都记录到对应的undo日志的trx\_id和roll\_pointer中，这样我们可以通过这个undo日志找到上一次对该记录进行修改时产生的undo日志。roll\_pointer指针串联成的undo链表称为版本链。

1. UPDATE操作对应的undo日志

在执行UPDATE操作时，InnoDB对于更新主键和不更新主键有两种处理。

1. 不更新主键

又可以细分为被更新的列占用的存储空间不变化与变化。如果更新的列不变化，可以就地更新，即直接在原纪录的基础上修改对应列的值；如果更新的列空间变化，则需要先将该记录从聚簇索引中删除，再根据新值创建一条新的记录插入页面中。这里的删除操作是直接加入垃圾链表，修改统计信息，如果新创建的记录占用的存储空间不超过旧记录的空间，可以直接重用旧空间。

1. 更新主键

对于更新主键的操作，意味着这条记录在聚簇索引中的位置将会发生改变，首先将旧记录进行delete mark操作，之所以只进行delete mark操作，是因为别的事务也可能同时访问这个记录，如果直接删除加入垃圾链表别的事务就访问不到了。再根据更新后的值创建一条新记录，插入聚簇索引页中。这种情况会产生两条undo日志，一条进行delete mark操作，一条进行插入操作。

1. 通用链表结构

在写入undo日志的过程中，会用到多个链表，很多链表都有相同的结构。在某个表空间内，我们可以通过一个页的页号和在页内的偏移量来定位一个节点的位置。链表节点通常包括指向前后节点的页号和偏移量，占用12字节；链表基节点包括了链表的头节点，尾结点和链表长度等信息，16字节。

1. undo日志类型

基本分为两类，一种是由INSERT语句产生，或者UPDATE语句中更新主键的语句产生的undo日志，称为insert undo日志；另一种是DELETE, UPDATE语句产生的，称为update undo日志。一个页面只能存储一种类型的undo日志，不同类型的undo日志是不能混杂在一起使用的。

1. undo页面链表

一个事务包含很多语句，一条记录可能对多条记录修改，每次修改都需要记录1-2条undo日志，这些日志可能在一个页面中放不下，需要放到多个页面中，这就需要将这些undo页面连成不同的链表了。在一个事务执行过程中，会有多种类型的语句，会产生不同类型的undo日志，只能放在不同的页中，所以在一个事务的执行过程中需要两个undo页面的链表，一个是insert undo链表，一个是update undo链表。同时InnoDB中该对普通表和临时表的undo日志有所区分，因此一个事务最多有4个以undo日志为节点的链表。这些链表都是按需分配的，只有在有需要时才会创建。