# 背景

参考：

<https://blog.csdn.net/w892824196/article/details/82591115>

mysql5.6开始支持在线ddl，在线ddl能够提供下面的好处；

1. 提高生产环境的可用性
2. 在ddl执行期间，获得性能和并发性的平衡，可以指定LOCK从句与algorithm从句，lock=exclusize会阻塞整个表的访问，lock=shared会允许查询但不允许dml，lock=none允许查询和dml操作，lock=default或是没有指定，mysql使用最低级别的锁，algorithm指定是拷贝表还是不拷贝表直接内部操作
3. 只对需要的地方做改变，不是创建一个新的临时表。

之前ddl操作的代价是很昂贵的，许多的alter table语句是创建一个新的，按需要的选项创建的空表，然后拷贝已经存在的行到新表中，在更新插入行的索引，在所有的行被拷贝之后，老的表被删除，拷贝的表被重命名成原来表的名。

在5.5和5.1优化了的create index和drop index避免了表拷贝的行为，这个特色叫快速索引创建，5.6增强了在改变的时候dml还能处理，叫在线ddl。

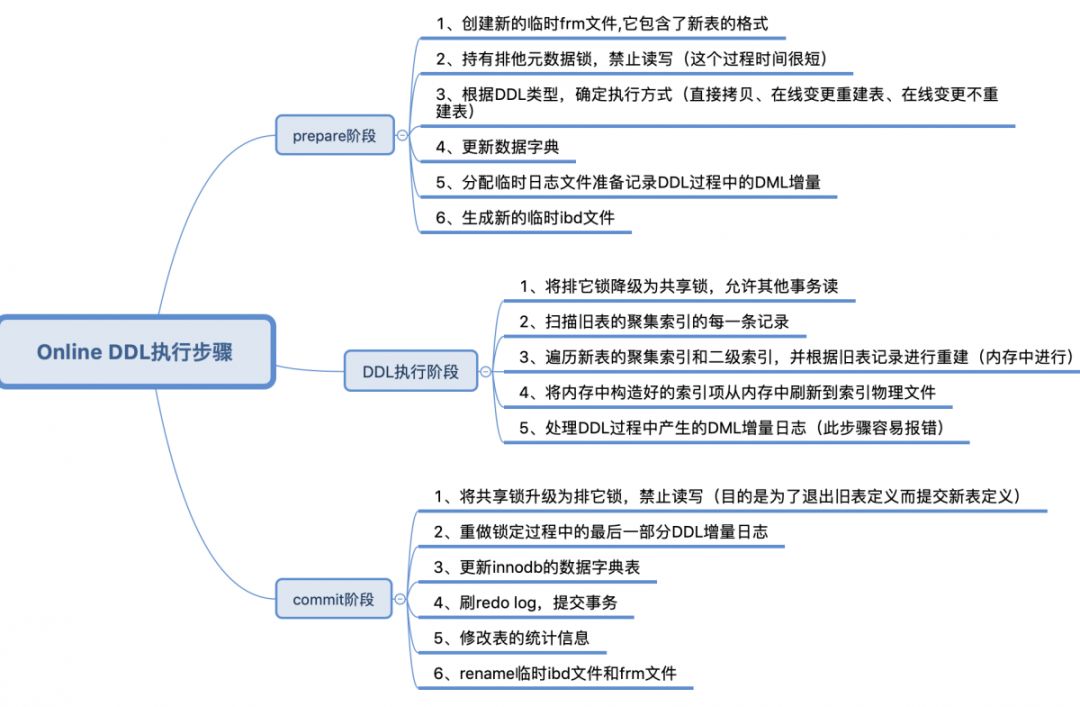
一些alter语句允许并发的dml，但是仍然需要拷贝表，这些操作的表拷贝要比之前版本的快。

当ddl在改变表的时候，表是否被锁住取决于操作的内部工作方式及alter table的lock从句，在线ddl语句总是等待访问表的事务提交或回滚，因为ddl语句在准备的过程中会要求一个短暂的排他请求。因为要记录并发dml操作产生的改变，并在最后应用这些改变，在线的ddl会花费更长的时间。

要看ddl是否使用了临时表还是内部操作的，可以查看语句执行结果中有多少行收到了影响，如果是0行，那么就没有复制表，如果是非0，那么就是复制了表。

# 过程

online ddl操作的执行过程一般被分为3个阶段，如下：



## 阶段1：初始化阶段（准备阶段）

在初始化阶段，服务器将考虑存储引擎功能，语句中指定的操作以及用户指定的ALGORITHM和LOCK 选项，以确定在操作期间允许多少并发。在此阶段，将使用共享的元数据锁来保护当前表定义。

## 阶段2：执行

在此阶段，准备并执行该语句。元数据锁是否升级到排它锁取决于初始化阶段评估的因素。如果需要排他元数据锁，则仅在语句准备期间进行短暂锁定。

## 阶段3：提交阶段

在提交表定义阶段，将元数据锁升级为排它锁，以退出旧表定义并提交新表定义，在获取排它锁的过程中，如果其他事务正在占有元数据的排它锁，那么本事务的提交操作可能会出现锁等待。

# 种类

DDL的种类有很多，比较常见的包含：

索引操作

主键操作

列操作

外键操作

表操作

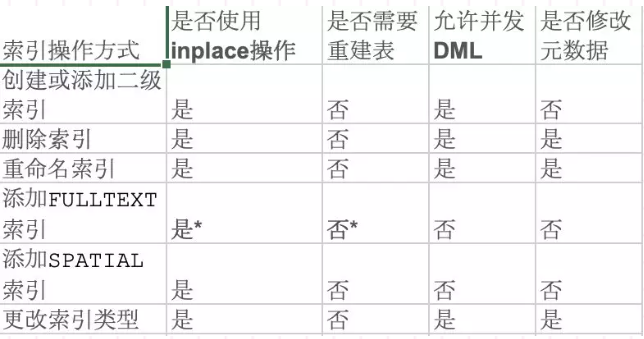
表空间操作

分区操作

每个操作里面又包含了很多种类，比如，索引操作中包含新增索引、删除索引等操作，列操作中有新增列、修改列、删除列等等，这些ddl操作执行过程中的状态究竟是什么样的？我们一一来看。

## 索引DDL操作

可以用下面的表来表示：

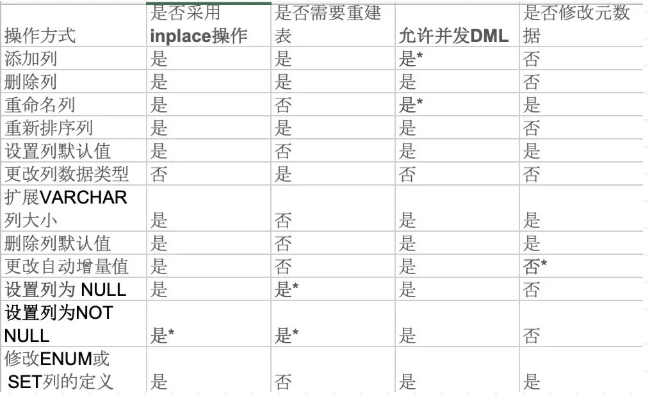


从上面的表中可以看出，创建或者添加二级索引的时候，使用了inplace的操作，不需要重建表，并且允许并发的DML，也就是说，在创建索引的过程中，原表是可读可写的。它数据新增元数据的操作，没有修改数据库的元数据。

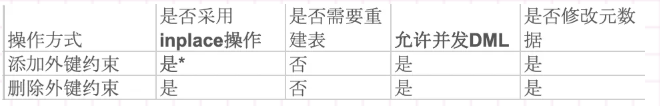
## 主键DDL操作



## 列DDL操作



## 外键操作



## 表操作



# 特点

## 优点

**online ddl操作支持表的本地更改(无需拷贝)和并发DML操作**，一般有以下几个优点：

1、一般的线上环境都是比较忙碌的，想要在一个大表中比较平滑的执行DDL变更几乎不太可能，但是线上的环境又不会接受几分钟的延迟，使用online ddl操作可以尽可能的降低这种影响。

2、online ddl中支持lock语法，lock语法可以微调对表的并发访问程度：

使用lock=none的方法可以开启表的读取和写入，

使用lock=shared方法可以允许对表进行读取，而关闭表的写入功能，

使用lock=exclusive可以禁止对表进行读写，组织并发查询和DML

换句话来说，lock语法可以平衡数据库服务并发和性能之间的竞争，但是需要注意的是：该方法有可能出现失败的情况，如果该方法不可用，该alter table 的操作会立即停止。

3、online ddl中支持algorithm的语法，该参数有两个取值，一个是copy，另外一个是inplace，来看官方文档说明：

COPY：对原始表的副本执行操作，并将表数据从原始表逐行复制到新表。不允许并发DML。

INPLACE：操作避免复制表数据，但可以在适当位置重建表。在操作的准备和执行阶段可以简短地获取表上的独占元数据锁定。通常，支持并发DML。

默认情况下，MySQL5.7使用inplace的方法，而不是copy表结构的方法。因此，与传统的表复制方法相比，online ddl可以降低磁盘上的消耗和IO上的开销。

简单总结，online ddl的3个优点：

a、降低线上变更表的影响时间

b、平衡数据库服务并发性和性能之间的竞争

c、降低磁盘和IO消耗

官方文档中给的常用的在线变更表结构的例子如下：

ALTERTABLEtbl\_name

ADDPRIMARYKEY(column),

ALGORITHM=INPLACE,LOCK=NONE;

## 系统空间

Online DDL对系统空间的要求：

a、如果DDL需要拷贝表数据，则需要额外的空间来保存中间临时表

b、如果DDL执行过程中支持并发DML，则DML操作产生的临时日志文件需要占用额外的系统空间

c、如果DDL执行过程中需要对数据进行排序，则需要额外的系统空间来存储额外的临时排序文件

## 限制

1、使用lock=none模式的时候，不允许有外键约束，如果表中有外键的时候，使用Online DDL会出现一些问题

2、持有元数据锁的其他事务可能导致Online DDL阻塞，Online DDL可能导致其他需要获取元数据锁的事务超时

3、执行Online DDL的执行线程和并行DML不是同一个执行线程，所以并行的DML在执行过程中可能会报错，Duplicate Key的错误

4、optimize table操作会使用重建表的方法来释放聚集索引中未使用的空间，它类似alter table的操作，因为要重建表，它的处理效率不高。

5、再对大表进行online ddl的操作时，还需要注意以下3点：

a、没有任何操作能够停止Online DDL操作或者限制该操作过程中IO和磁盘使用率

b、一旦中间发生问题，回滚的代价非常昂贵

c、大表的Online DDL会导致复制出现巨大的延迟，这一点在主从复制架构中需要考虑到

综上所述，在对大表进行Online DDL的时候，有两种方法：

1、使用pt-osc或者gh-ost等在线变更的工具进行变更√

~~2、提前准备好故障报告，直接在线上进行变更，该方法纯属娱乐×~~

# 失败

官方文档上给出了可能失败的几种情况：

1、手工指定的algorithm和存储引擎中的算法出现冲突

2、在一些必须使用排它锁的场合手工指定锁的类型为share或者为none

3、需要拷贝表的时候系统磁盘空间溢出或者DDL过程中的并发DML临时日志文件过大导致超过了参数innodb\_online\_alter\_max\_size的值

4、当前系统有不活跃的事务占用了元数据锁，导致锁等待超时

5、DDL添加唯一二级索引的时候，并发DML中插入了重复键值的记录，此时会造成alter table的操作回滚