DDL 阻塞：

方法一：sys.schema_table_lock_waits

```sql
mysql> select * from sys.schema_table_lock_waits\G
*************************** 1. row ***************************
               object_schema: sbtest
                 object_name: t1
           waiting_thread_id: 62
                 waiting_pid: 25
             waiting_account: root@localhost
           waiting_lock_type: EXCLUSIVE
       waiting_lock_duration: TRANSACTION
               waiting_query: alter table sbtest.t1 add c1 datetime
          waiting_query_secs: 17
 waiting_query_rows_affected: 0
 waiting_query_rows_examined: 0
          blocking_thread_id: 61
                blocking_pid: 24
            blocking_account: root@localhost
          blocking_lock_type: SHARED_READ
      blocking_lock_duration: TRANSACTION
     sql_kill_blocking_query: KILL QUERY 24
sql_kill_blocking_connection: KILL 24
*************************** 2. row ***************************
               object_schema: sbtest
                 object_name: t1
           waiting_thread_id: 62
                 waiting_pid: 25
             waiting_account: root@localhost
           waiting_lock_type: EXCLUSIVE
       waiting_lock_duration: TRANSACTION
               waiting_query: alter table sbtest.t1 add c1 datetime
          waiting_query_secs: 17
 waiting_query_rows_affected: 0
 waiting_query_rows_examined: 0
          blocking_thread_id: 62
                blocking_pid: 25
            blocking_account: root@localhost
          blocking_lock_type: SHARED_UPGRADABLE
      blocking_lock_duration: TRANSACTION
     sql_kill_blocking_query: KILL QUERY 25
sql_kill_blocking_connection: KILL 25
```

2 rows in set (0.00 sec)
```

只有一个 alter 操作，却产生了两条记录，而且两条记录的 Kill 对象还不一样，其中一条 Kill 的对象还是 alter 操作本身。

在定位问题时，这 N*2 条记录完全是个噪音。

这个时候，就需要我们对上述记录进行过滤了。

过滤的关键是 blocking_lock_type 不等于 SHARED_UPGRADABLE。

SHARED_UPGRADABLE 是一个可升级的共享元数据锁，加锁期间，允许并发查询和更新，常用在 DDL 操作的第一阶段。

所以，阻塞 DDL 的不会是 SHARED_UPGRADABLE。

故而，针对上面这个 case，我们可以通过下面这个查询来精确地定位出需要 Kill 的会话。

```sql
SELECT sql_kill_blocking_connection
FROM sys.schema_table_lock_waits
WHERE blocking_lock_type <> 'SHARED_UPGRADABLE'
 AND waiting_query = 'alter table sbtest.t1 add c1 datetime';
```

方法二：Kill DDL 之前的会话

导致 DDL 被阻塞的操作，无非两类：

1. 表上有慢查询未结束。
2. 表上有事务未提交。

第一类比较好定位，通过 show processlist 就能发现。

第二类 information_schema.innodb_trx 中肯定会有记录，如 session1 中的事务，在表中的记录如下

```sql
mysql> select * from information_schema.innodb_trx\G
*************************** 1. row ***************************
```
```

```
                    trx_id: 421568246406360
                 trx_state: RUNNING
               trx_started: 2022-01-02 08:53:50
     trx_requested_lock_id: NULL
          trx_wait_started: NULL
                trx_weight: 0
       trx_mysql_thread_id: 24
                 trx_query: NULL
       trx_operation_state: NULL
         trx_tables_in_use: 0
         trx_tables_locked: 0
          trx_lock_structs: 0
     trx_lock_memory_bytes: 1128
           trx_rows_locked: 0
         trx_rows_modified: 0
   trx_concurrency_tickets: 0
       trx_isolation_level: REPEATABLE READ
         trx_unique_checks: 1
    trx_foreign_key_checks: 1
trx_last_foreign_key_error: NULL
 trx_adaptive_hash_latched: 0
 trx_adaptive_hash_timeout: 0
          trx_is_read_only: 0
trx_autocommit_non_locking: 0
       trx_schedule_weight: NULL
1 row in set (0.00 sec)
```

trx_mysql_thread_id 是线程 id ，结合 information_schema.processlist

下面这个 SQL 定位出执行时间早于 DDL 的事务。

```sql
SELECT concat('kill ', i.trx_mysql_thread_id, ';')
FROM information_schema.innodb_trx i, (
    SELECT MAX(time) AS max_time
    FROM information_schema.processlist
    WHERE state = 'Waiting for table metadata lock'
      AND (info LIKE 'alter%'
      OR info LIKE 'create%'
      OR info LIKE 'drop%'
      OR info LIKE 'truncate%'
      OR info LIKE 'rename%'
  )) p
```

```
WHERE timestampdiff(second, i.trx_started, now()) > p.max_time;
```

当前正在执行的查询也会显示在 information_schema.innodb_trx 中。

所以，上面这个 SQL 同样也适用于慢查询未结束的场景

MySQL 5.7 中使用 sys.schema_table_lock_waits 的注意事项

sys.schema_table_lock_waits 视图依赖了一张 MDL 相关的表 - performance_schema.metadata_locks。

该表是 MySQL 5.7 引入的，会显示 MDL 的相关信息，包括作用对象、锁的类型及锁的状态等。

但在 MySQL 5.7 中，该表默认为空，因为与之相关的 instrument 默认没有开启。MySQL 8.0 才默认开启。

```sql
mysql> select * from performance_schema.setup_instruments where name='wait/lock/metadata/sql/mdl';
+-----------------------------+---------+-------+
| NAME                        | ENABLED | TIMED |
+-----------------------------+---------+-------+
| wait/lock/metadata/sql/mdl | NO      | NO    |
+-----------------------------+---------+-------+
1 row in set (0.00 sec)
```

所以，在 MySQL 5.7 中，如果我们要使用 sys.schema_table_lock_waits ，必须首先开启 MDL 相关的 instrument。

开启方式很简单，直接修改 performance_schema.setup_instruments 表即可。

具体 SQL 如下。

```sql
UPDATE performance_schema.setup_instruments SET ENABLED = 'YES', TIMED = 'YES'
WHERE NAME = 'wait/lock/metadata/sql/mdl';
```

但这种方式是临时生效，实例重启后，又会恢复为默认值。

建议同步修改配置文件。

```sql
[mysqld]
performance-schema-instrument='wait/lock/metadata/sql/mdl=ON'
```

TRUNCATE

MySQL 8.0 的 `truncate` 实现方式基本和 `drop` 实现方式相同，包括主要的耗时位置（都在 `row_drop_table_for_mysql`、`os_file_delete_func`）都是相同的。

MySQL 5.7 的 `truncate` 和 `drop` 实现差异较大，整个实现过程几乎是完全独立的代码。`truncate` 使用 `row_truncate_table_for_mysql`，`drop` 使用 `row_drop_table_for_mysql` ； `truncate` 操作的主要的耗时有 `dict_drop_index_tree`、`os_file_truncate`。

其中：`row_drop_table_for_mysql` 主要是调用 `btr_drop_ahi_for_table` 执行 AHI 的 page 页的删除。`os_file_delete_func` 主要调用 `unlink` 执行文件的清理。

假如需要 `truncate` 的表分配的 fd 为 43，`truncate` 过程中，会先将表 `rename`。这个时候这个 fd 会被关闭，43 就被释放了。然后执行 `create table` 操作。一般这个间隙过程很短，因此新建立的表可以使用被释放的 43 了，所以会看到 fd 没有变化。

如果 `rename` 之后，在内部执行 `create table` 之前，又打开了新文件，那这时候 fd 43 就会被其它打开的文件持有，`truncate` 之后表的 fd 也就会发生变化。

> 注意：MySQL 8.0 是真正使用 `rename` + `create` + `drop`实现的 `truncate`，但 MySQL 5.7 是通过文件的 `truncate` 实现的。

总结

\1. 执行 show processlist ，如果 DDL 的状态是 Waiting for table metadata lock ，则意味着这个 DDL 被阻塞了。

\2. 定位导致 DDL 被阻塞的会话，常用的方法有两种：

- **sys.schema_table_lock_waits**

```sql
SELECT sql_kill_blocking_connection
FROM sys.schema_table_lock_waits
WHERE blocking_lock_type <> 'SHARED_UPGRADABLE'
  AND (waiting_query LIKE 'alter%'
  OR waiting_query LIKE 'create%'
  OR waiting_query LIKE 'drop%'
  OR waiting_query LIKE 'truncate%'
  OR waiting_query LIKE 'rename%');
```

这种方法适用于 MySQL 5.7 和 8.0。

注意，MySQL 5.7 中，MDL 相关的 instrument 默认没有打开。

- **Kill DDL 之前的会话**

```sql
SELECT concat('kill ', i.trx_mysql_thread_id, ';')
FROM information_schema.innodb_trx i, (
    SELECT MAX(time) AS max_time
    FROM information_schema.processlist
    WHERE state = 'Waiting for table metadata lock'
      AND (info LIKE 'alter%'
      OR info LIKE 'create%'
      OR info LIKE 'drop%'
      OR info LIKE 'truncate%'
      OR info LIKE 'rename%'
  )) p
WHERE timestampdiff(second, i.trx_started, now()) > p.max_time;
```

如果 MySQL 5.7 中 MDL 相关的 instrument 没有打开或在 MySQL 5.6 中，可使用该方法。