INSERT ... ON DUPLICATE KEY UPDATE产生death lock死锁原理

前言

编辑

我们在实际业务场景中，经常会有一个这样的需求，插入某条记录，如果已经存在了则更新它如果更新日期或者某些列上的累加操作等，我们肯定会想到使用INSERT ... ON DUPLICATE KEY UPDATE语句，一条语句就搞定了查询是否存在和插入或者更新这几个步骤，但是使用这条语句在msyql的innodb5.0以上版本有很多的陷阱，即有可能导致death lock死锁也有可能导致主从模式下的replication产生数据不一致。

正文

正如前言说的那样，在实际业务中，曾经有过一个需求就是插入一条业务数据，如果不存在则新增，存在则累加更新某一个字段的值，于是乎就想到了使用insert... on duplicate key update这个语句，但是有一天去测试环境查看错误日志时，却发现了在多个事务并发执行同一条insert...on duplicate key update 语句时，也就是insert的内容相同时，发生 了死锁。

对于insert...on duplicate key update这个语句会引发dealth lock问题，官方文档也没有相关描述，只是进行如下描述：

An INSERT ... ON DUPLICATE KEY UPDATE statement against a table having more than one unique or primary key is also marked as unsafe. (Bug #11765650, Bug #58637)

也就是如果一个表定义有多个唯一键或者主键时，是不安全的，这又引发了以一个问题，见https://bugs.mysql.com/bug.php?id=58637

也就是

当mysql执行INSERT ON DUPLICATE KEY的 INSERT时，存储引擎会检查插入的行是否会产生重复键错误。如果是的话，它会将现有的

行返回给mysql，mysql会更新它并将其发送回存储引擎。当表具有多个唯一或主键时，此语句对存储引擎检查密钥的顺序非常敏感。根据这个顺序，
存储引擎可以确定不同的行数据给到mysql，因此mysql可以更新不同的行。存储引擎检查key的顺序不是确定性的。例如，InnoDB按照索引添加到
表的顺序检查键。

insert ... on duplicate key 在执行时，innodb引擎会先判断插入的行是否产生重复key错误，如果存在，在对该现有的行加上S（共享锁）锁，如果返回该行数据给mysql,然后mysql执行完duplicate后的update操作，然后对该记录加上X（排他锁），最后进行update写入。

如果有两个事务并发的执行同样的语句，那么就会产生death lock，如：

时间	事务T1	事务T2
time1	执行INSERT ON DUPLICATE KEY	
time2	key以存在，获取该记录的S锁，获取该记录	执行INSERT ON DUPLICATE KEY
time3	对读取的记录进行修改	key以存在，获取该记录的S锁，获取该记录
time4		对读取的记录进行修改
time5	把修改写进存储引擎，给该记录加上X锁，T2存在S锁，所以等待T2释放S锁	
time6		把修改写进存储引擎，给该记录加上X锁，T1存在S锁，所以等待T1释放S锁
time7		死锁
time8		

https://bugs.mysql.com/bug.php?id=58637

编辑

https://bugs.mysql.com/bug.php?id=21356

解决办法：

1、尽量不对存在多个唯一键的table使用该语句

2、在有可能有并发事务执行的insert 的内容一样情况下不使用该语句