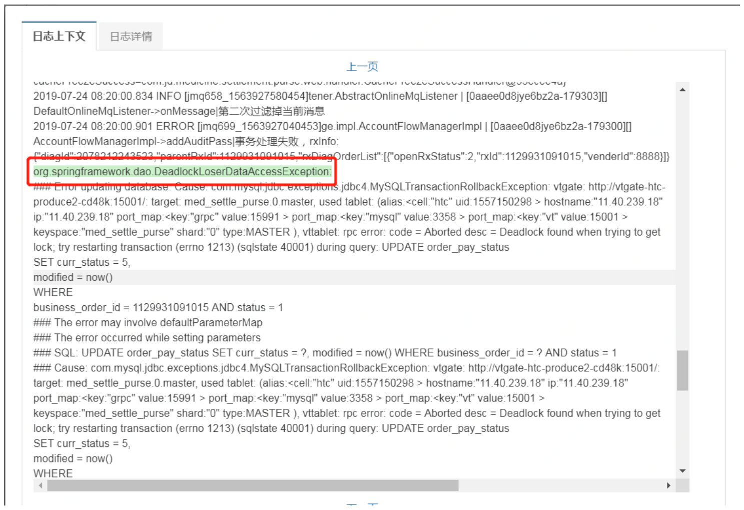
一次mysql死锁的排查

背景

pop购药上线后解冻操作经常发生死锁，报错日志如下：



```
2019-07-24 08:20:00.834 INFO [jmq658_1563927580454]tener.AbstractOnlineMqListener | [0aaee0d8jye6bz2a-179303][]
DefaultOnlineMqListener->onMessage|第二次过滤掉当前消息
2019-07-24 08:20:00.901 ERROR [jmq699_1563927040453]ge.impl.AccountFlowManagerImpl | [0aaee0d8jye6bz2a-179300][]
AccountFlowManagerImpl->addAuditPass|事务处理失败，rxInfo:
{"diagId":2078212243523,"parentRxId":1129931091015,"rxDiagOrderList":[{"openRxStatus":2,"rxId":1129931091015,"venderId":8888}]}
org.springframework.dao.DeadlockLoserDataAccessException:
### Error updating database. Cause: com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: vtgate: http://vtgate-htc-
produce2-cd48k:15001/: target: med_settle_purse.0.master, used tablet: (alias:<cell:"htc" uid:1557150298 > hostname:"11.40.239.18"
ip:"11.40.239.18" port_map:<key:"grpc" value:15991 > port_map:<key:"mysql" value:3358 > port_map:<key:"vt" value:15001 >
keyspace:"med_settle_purse" shard:"0" type:MASTER ), vttablet: rpc error: code = Aborted desc = Deadlock found when trying to get
lock; try restarting transaction (errno 1213) (sqlstate 40001) during query: UPDATE order_pay_status
SET curr_status = 5,
modified = now()
WHERE
business_order_id = 1129931091015 AND status = 1
### The error may involve defaultParameterMap
### The error occurred while setting parameters
### SQL: UPDATE order_pay_status SET curr_status = ?, modified = now() WHERE business_order_id = ? AND status = 1
### Cause: com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: vtgate: http://vtgate-htc-produce2-cd48k:15001/:
target: med_settle_purse.0.master, used tablet: (alias:<cell:"htc" uid:1557150298 > hostname:"11.40.239.18" ip:"11.40.239.18"
port_map:<key:"grpc" value:15991 > port_map:<key:"mysql" value:3358 > port_map:<key:"vt" value:15001 >
keyspace:"med_settle_purse" shard:"0" type:MASTER ), vttablet: rpc error: code = Aborted desc = Deadlock found when trying to get
lock; try restarting transaction (errno 1213) (sqlstate 40001) during query: UPDATE order_pay_status
SET curr_status = 5,
modified = now()
WHERE
```

死锁sql语句

session1	session2

`DELETE from order_pay_status`
`where id in (`
`  select b.id from (`
`    select id from order_pay_status`
`    where id > 3`
`    order by id`
`    limit 500`
`  ) b`
`)`
	update order_pay_status set curr_status = 4 where id = 9;

死锁日志

```
*** (1) TRANSACTION:
TRANSACTION 24836, ACTIVE 9 sec starting index read
mysql tables in use 1, locked 1
LOCK WAIT 6 lock struct(s), heap size 1136, 3 row lock(s), undo log entries 2
MySQL thread id 374, OS thread handle 140602328172288, query id 6711 10.0.56.104 root updating
UPDATE order_pay_status
SET curr_status = 4,
modified = now()
WHERE
id = 9
*** (1) WAITING FOR THIS LOCK TO BE GRANTED:
RECORD LOCKS space id 251 page no 3 n bits 72 index PRIMARY of table `med_settle_purse`.`order_pay_status` trx id 24836 lock_mode X
locks rec but not gap waiting
```

```
*** (2) TRANSACTION:
TRANSACTION 24842, ACTIVE 9 sec starting index read
mysql tables in use 2, locked 2
5 lock struct(s), heap size 1136, 4 row lock(s)
MySQL thread id 373, OS thread handle 140602736256768, query id 6717 10.0.56.104 root updating
DELETE from order_pay_status
where id in (
select b.id from (
select id from order_pay_status
where id > 0
AND DATE_FORMAT(created,'%Y-%m-%d') < DATE_FORMAT('2019-05-02 09:20:48.976','%Y-%m-%d')
order by id
limit 500
) b
)
*** (2) HOLDS THE LOCK(S):
RECORD LOCKS space id 251 page no 3 n bits 72 index PRIMARY of table `med_settle_purse`.`order_pay_status` trx id 24842 lock mode S
Record lock, heap no 1 PHYSICAL RECORD: n_fields 1; compact format; info bits 0
0: len 8; hex 73757072656d756d; asc supremum;;

*** (2) WAITING FOR THIS LOCK TO BE GRANTED:
RECORD LOCKS space id 251 page no 3 n bits 72 index PRIMARY of table `med_settle_purse`.`order_pay_status` trx id 24842 lock_mode X
waiting
```

死锁特征

1：lock_mode X locks rec but not gap waiting

2：hold lock mode S，lock_mode X waiting

3：隔离级别：RR

4：索引：主键索引

四种类型的行锁

1：记录锁（LOCK_REC_NOT_GAP）: lock_mode X locks rec but not gap

2：间隙锁（LOCK_GAP）: lock_mode X locks gap before rec

3：Next-key 锁（LOCK_ORNIDARY）: lock_mode X

4：插入意向锁（LOCK_INSERT_INTENTION）: lock_mode X locks gap before rec insert intention

重现步骤

session1	session2

```
DELETE from order_pay_status
where id in (
  select b.id from (
    select id from order_pay_status
    where id > 3
    order by id
    limit 500
  ) b
)
```
首先子查询对符合条件的记录逐行加S锁）（包括id=9的记录）	

	update order_pay_status set curr_status = 4 where id = 9;
	
	其次更新语句对id=9的记录加排他锁，等待

```
DELETE from order_pay_status
where id in (
  select b.id from (
    select id from order_pay_status
    where id > 3
    order by id
    limit 500
  ) b
)
```
执行delete操作对所有记录加排他锁，等待	

deadLock	deadLock

分析

1：mysql的锁机制是公平锁，即有个锁队列，先到先得。

2：不同事物下S锁与X锁互斥

3：锁的范围是锁整个事务，事务不结束，锁不释放。

清楚上面的概念后就不难得出结论了：

1. 事物A对id=9的记录加S锁，处于锁队列的第一个位置，并加锁成功，。
2. 事物B对id=9的记录加X锁，处于锁队列的第二个位置，因为第一个位置的S锁不是自己事物内的所以互斥加锁失败，处于等待。
3. 事物A对id=9的记录加X锁，处于锁队列的第三个位置，因为锁队列是先到先得的，第二位置的X锁不是自己事物内的锁所以互斥，处于等待。

这样就出现了锁等待，发生死锁。

解决

经过上面的分析，不难看出问题出现在了S锁上。如果没有S锁就不会发生死锁。那么S锁是怎么加的呢？因为删除语句做了子查询，为了防止读出数据不一致，所以读的时候加S锁，然后再执行删除。所以解决办法就是删除子查询，把session1的语句拆成两部分，第一部分普通查询（普通查询不会加S锁实际上是快照查），第一部分的结果作为第二部分的入参执行真正的删除操作。这样就不会加S锁，也不会出现死锁了。

资料

mysql发生死锁的种类有很多，这只是一个案例，更多案例https://github.com/aneasystone/mysql-deadlocks

参考博客：解决mysql死锁之路