1.数据库出现死锁，收到报警邮件

```
16:50:09.292 [http-nio-8080-exec-23] ERROR c.m.s.c.c [redacted] xception: {}
org.springframework.dao.DeadlockLoserDataAccessException:
### Error updating database.  Cause: com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: Deadlock found
when trying to get lock; try restarting transaction
### The error may involve com.[redacted] -Inline
### The error occurred while setting parameters
### SQL: update `order`    SET updated_at = now()    where tid = ?
### Cause: com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: Deadlock found when trying to get lock; try
restarting transaction
; SQL []; Deadlock found when trying to get lock; try restarting transaction; nested exception is
com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: Deadlock found when trying to get lock; try restarting
transaction
at
org.springframework.jdbc.support.SQLErrorCodeSQLExceptionTranslator.doTranslate(SQLErrorCodeSQLExceptionTranslator.java:263)
at
org.springframework.jdbc.support.AbstractFallbackSQLExceptionTranslator.translate(AbstractFallbackSQLExceptionTranslator.java:73)
at org.mybatis.spring.MyBatisExceptionTranslator.translateExceptionIfPossible(MyBatisExceptionTranslator.java:73)
at org.mybatis.spring.SqlSessionTemplate$SqlSessionInterceptor.invoke(SqlSessionTemplate.java:446)
at com.sun.proxy.$Proxy124.update(Unknown Source)
at org.mybatis.spring.SqlSessionTemplate.update(SqlSessionTemplate.java:294)
at org.apache.ibatis.binding.MapperMethod.execute(MapperMethod.java:62)
at org.apache.ibatis.binding.MapperProxy.invoke(MapperProxy.java:59)
--
at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:49)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
at java.lang.Thread.run(Thread.java:748)
Caused by: com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: Deadlock found when trying to get lock; try
restarting transaction
at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:62)
at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)
at java.lang.reflect.Constructor.newInstance(Constructor.java:423)
```

2.查找到相关代码，发现分析不出来问题

```
    waybillBean.setLastStatus(waybill.getDbStatus());
    waybillBean.setLcStatus(waybill.getLcStatus());
    waybillBean.setStatus(waybill.getStatus());
    waybillBean.setDmOperator(waybill.getDmOperator());
    waybillBean.setDmMobile(waybill.getDmMobile());
    logger.info("OrderServiceImpl updateWaybillInfo waybillBean {}", waybillBean);
    waybillDao.updateWaybillInfoByBean(waybillBean);

    //更新订单状态
    OrderBean orderBean = new OrderBean();
    orderBean.setTid(waybill.getOrderId());
    orderBean.setStatus(waybill.getOrderStatus());
    orderBean.setUpdatedAt(waybill.getTime());
    logger.info("OrderServiceImpl updateWaybillInfo orderBean {}", orderBean);
    orderDao.updateOrderInfoByBean(orderBean);

    //插入流转信息
```

3.学习查找关于数据库死锁的资料如下：

http://hedengcheng.com/

https://www.cnblogs.com/LBSer/p/5183300.htm

https://www.cnblogs.com/huanongying/p/7021555.html

4.通过数据库命令查看数据库，获取相关数据库死锁的相关的事务和sql语句，通过查看日志发现，事物1需要获取waybill表中的锁，事务2持有waybill表中的锁，

需要获取订单表中的锁。

show engine innodb status;

```
*** (1) TRANSACTION:
TRANSACTION 2421467629, ACTIVE 0 sec starting index read
mysql tables in use 1, locked 1
LOCK WAIT 6 lock struct(s), heap size 1184, 4 row lock(s), undo log entries 1
MySQL thread id 28223606, OS thread handle 0x7f58a5e3a700, query id 2451340715 10.10.58.249 tms Searching rows for update
UPDATE
    `waybill`
    SET
    status = 'REJECTED'
    ,last_status = 'ACCEPT'
    ,lc_status = null
    ,updated_at = now()

      ,dm_operator='张海超'


      ,dm_mobile='15226598163'


    WHERE
    order_tid = '192307435189458 46'
*** (1) WAITING FOR THIS LOCK TO BE GRANTED:
RECORD LOCKS space id 301 page no 128209 n bits 216 index `PRIMARY` of table `tms_production`.`waybill` trx id 2421467629 lock_mode X locks rec but not gap waiting
Record lock, heap no 102 PHYSICAL RECORD: n_fields 18; compact format; info bits 0
 0: len 4; hex 83429a00; asc  B  ;;
 1: len 6; hex 00009054a9ee; asc     T  ;;
 2: len 7; hex 3c0000044d23fa; asc <   M# ;;
 3: len 5; hex 99a0390a5e; asc    9 ^;;
 4: len 5; hex 99a0390c89; asc    9  ;;
 5: len 12; hex 333031363638303033303134; asc 301668003014;;
 6: len 17; hex 313932333037343335313839343538343436; asc 19230743518945846;;
 7: len 11; hex 31333731373730363137333; asc 13717706173;;
 8: len 9; hex e6a281e6b0b8e58ba4; asc         ;;
 9: len 8; hex 43414e43454c4544; asc CANCELED;;
 10: len 6; hex 4d545f504f50; asc MT_POP;;
 11: len 6; hex 414343455054; asc ACCEPT;;
 12: len 0; hex ; asc ;;
 13: SQL NULL;
 14: SQL NULL;
 15: len 6; hex 43414e43454c; asc CANCEL;;
 16: SQL NULL;
 17: SQL NULL;

*** (2) TRANSACTION:
TRANSACTION 2421467630, ACTIVE 0 sec starting index read, thread declared inside InnoDB 5000
mysql tables in use 1, locked 1
5 lock struct(s), heap size 1184, 3 row lock(s), undo log entries 1
MySQL thread id 28215627, OS thread handle 0x7f5ba579f700, query id 2451340717 10.10.58.249 tms updating
update `order`
   SET updated_at = now()
  where tid = '19230743518945846'
*** (2) HOLDS THE LOCK(S):
RECORD LOCKS space id 301 page no 128209 n bits 216 index `PRIMARY` of table `tms_production`.`waybill` trx id 2421467630 lock_mode X locks rec but not gap
Record lock, heap no 102 PHYSICAL RECORD: n_fields 18; compact format; info bits 0
 0: len 4; hex 83429a00; asc  B  ;;
 1: len 6; hex 00009054a9ee; asc     T  ;;
 2: len 7; hex 3c0000044d23fa; asc <   M# ;;
 3: len 5; hex 99a0390a5e; asc    9 ^;;
```

5.通过分析上面数据库引擎日志，找到两个事务对应的代码

事务1对应的代码：

```
TransactionStatus status = dataSourceTransactionManager.getTransaction(def);
try {
    if (!orderBean.getStatus().equals(orderStatus)) {
        orderDao.updateOrderInfo(orderParams);
    }
    if (!waybillBean.getStatus().equals(waybillStatus)) {
        waybillDao.updateWaybillInfo(waybillParams);
    }
    if (insertTrace) {
        waybillTraceBean.setCreatedAt(new Date());
        waybillTraceBean.setUpdatedAt(new Date());
```

事务2对应的代码：

```
waybillBean.setLastStatus(waybill.getDbStatus());
waybillBean.setLcStatus(waybill.getLcStatus());
waybillBean.setStatus(waybill.getStatus());
waybillBean.setDmOperator(waybill.getDmOperator());
waybillBean.setDmMobile(waybill.getDmMobile());
logger.info("OrderServiceImpl updateWaybillInfo waybillBean {}", waybillBean);
waybillDao.updateWaybillInfoByBean(waybillBean);

//更新订单状态
OrderBean orderBean = new OrderBean();
orderBean.setTid(waybill.getOrderId());
orderBean.setStatus(waybill.getOrderStatus());
orderBean.setUpdatedAt(waybill.getTime());
logger.info("OrderServiceImpl updateWaybillInfo orderBean {}", orderBean);
orderDao.updateOrderInfoByBean(orderBean);

//插入流转信息
```

这时，一下子恍然大悟，两个事物更新SQL语句的顺序刚好相反，并且持有对方需要的X锁，刚好满足死锁的条件。

6.继续查看服务器日志，看看什么场景，满足上面的场景。

场景1：三方物流公司回调取消订单

```
16:50:09.292 [http-nio-8080-exec-23] ERROR [redacted] Exception: {}
org.springframework.dao.DeadlockLoserDataAccessException:
### Error updating database.  Cause: com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: Deadlock found
when trying to get lock; try restarting transaction
### The error may involve [redacted].OrderDao.updateOrderInfoByBean-Inline
### The error occurred while setting parameters
### SQL: update `order`    SET updated_at = now()    where tid = ?
### Cause: com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: Deadlock found when trying to get lock; try
restarting transaction
; SQL []; Deadlock found when trying to get lock; try restarting transaction; nested exception is
com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: Deadlock found when trying to get lock; try restarting
transaction
at
org.springframework.jdbc.support.SQLErrorCodeSQLExceptionTranslator.doTranslate(SQLErrorCodeSQLExceptionTranslator.java:263)
at
org.springframework.jdbc.support.AbstractFallbackSQLExceptionTranslator.translate(AbstractFallbackSQLExceptionTranslator.java:73)
at org.mybatis.spring.MyBatisExceptionTranslator.translateExceptionIfPossible(MyBatisExceptionTranslator.java:73)
at org.mybatis.spring.SqlSessionTemplate$SqlSessionInterceptor.invoke(SqlSessionTemplate.java:446)
at com.sun.proxy.$Proxy124.update(Unknown Source)
at org.mybatis.spring.SqlSessionTemplate.update(SqlSessionTemplate.java:294)
at org.apache.ibatis.binding.MapperMethod.execute(MapperMethod.java:62)
at org.apache.ibatis.binding.MapperProxy.invoke(MapperProxy.java:59)
--
at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:49)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
at java.lang.Thread.run(Thread.java:748)
Caused by: com.mysql.jdbc.exceptions.jdbc4.MySQLTransactionRollbackException: Deadlock found when trying to get lock; try
restarting transaction
at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:62)
at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)
at java.lang.reflect.Constructor.newInstance(Constructor.java:423)
```

场景2：站长通过管理系统手动取消订单

```
[www@idc01-tms-api-06 skylark]$ cat skylark.log |grep "19230743518945846"
16:50:08.915 [http-nio-8080-exec-26] INFO  c.m.s.c.openapi.OrderInfoController - order_operation params:{optType=2, lcStatus=REJECTED, reason_type=REJECTED_PRODUCT, d
[redacted], remark=, lc_status=REJECTED, reasonType=REJECTED_PRODUCT, dm_name=[redacted], dmName=%E5%BC%A0%E6%B5%B7%E8%B6%85, tid=19230743518945846, dm_mobile=15
226598163}
16:50:08.915 [http-nio-8080-exec-26] INFO  c.m.s.s.impl.OrderStatusServiceImpl - OrderStatusServiceImpl changeOrderStatus params:{optType=2, lcStatus=REJECTED, reason
_type=REJECTED_PRODUCT, dmMobile=[redacted], remark=, lc_status=REJECTED, reasonType=REJECTED_PRODUCT, dm_name=[redacted], dmName=%E5%BC%A0%E6%B5%B7%E8%B6%85, tid=192307
43518945846, dm_mobile=[redacted]}
16:50:08.918 [http-nio-8080-exec-26] INFO  c.m.s.s.impl.AmsRequestServiceImpl - AmsRequestServiceImpl sendRejected url:http://[redacted]/synReje
ct.json, params:{"reason":"REJECTED_PRODUCT","orderNo":"19230743518945846","createTime":"2018-06-28 16:50:08","remark":"","createUser":"[redacted]"}
```

上面两个场景都需要各自在一个事务中修改订单和运单表，并且更新的订单和运单相同且更新顺序相反，在相同时间点刚好满足死锁条件，好在数据库在死锁的情况下，回超时回滚事务，其中一个事务会主动释放锁。