MySQL锁类型以及子查询锁表问题、解锁

MySQL中select * for update锁表的范围

MySQL中select * for update锁表的问题

由于InnoDB预设是Row-Level Lock，所以只有「明确」的指定主键，MySQL才会执行Row lock (只锁住被选取的资料例)，否则MySQL将会执行Table

Lock (将整个资料表单给锁住)。 举个例子: 假设有个表单products ，里面有id跟name二个栏位，id是主键。

例1: (明确指定主键，并且有此笔资料，row lock)

SELECT * FROM products WHERE id='3' FOR UPDATE;

SELECT * FROM products WHERE id='3' and type=1 FOR UPDATE;

例2: (明确指定主键，若查无此笔资料，无lock)

SELECT * FROM products WHERE id='-1' FOR UPDATE;

例2: (无主键，table lock)

SELECT * FROM products WHERE name='Mouse' FOR UPDATE;

例3: (主键不明确，table lock)

SELECT * FROM products WHERE id<>'3' FOR UPDATE;

例4: (主键不明确，table lock)

SELECT * FROM products WHERE id LIKE '3' FOR UPDATE;
注1: FOR UPDATE仅适用于InnoDB，且必须在交易区块(BEGIN/COMMIT)中才能生效。

注2: 要测试锁定的状况，可以利用MySQL的Command Mode ，开二个视窗来

做测试。
在MySql 5.0中测试确实是这样的

另外：MyAsim 只支持表级锁，InnerDB支持行级锁 添加了(行级锁/表级锁)锁的数据不能被其它事务再锁定，也不被其它事务修改

（修改、删除）。是表级锁时，不管是否查询到记录，都会锁定表。

今天碰到诡异的表死锁问题。
首先Tomcat报错：

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)

使用 show engine innodb status .
查看mysql死锁。
发现是 update语句中把子查询中的表给死锁了。比如
update table_a set comments = (select count(1) from table_b where id =
table_a.id) where id = 123;
把table_b给锁住了。
搜索了一个，发现是mysql的问题。
http://shen2.cn/2013/06/sub-query-in-update-locked-table/

最后发现这个不是mysql bug,
mysql 默认的隔离级别是REPEATABLE-READ，oracle默认数据隔离级别是READ-COMMITTED 。所以在mysq中
update ... select * from
insert into .... select * from
这些语句中，都会锁住子表的row.
将mysql 隔离级别改成
SET session TRANSACTION ISOLATION LEVEL READ COMMITTED;
问题就解决了。

解锁
第一种
show processlist;
找到锁进程，kill id ;

第二种
mysql>UNLOCK TABLES;
锁表
锁定数据表，避免在备份过程中，表被更新
mysql>LOCK TABLES tbl_name READ;
为表增加一个写锁定：
mysql>LOCK TABLES tbl_name WRITE;

以上文章非原创，只做收藏用。