链滴

MySQL 事务

作者：helly

原文链接：https://ld246.com/article/1510315627285

来源网站：链滴

InnoDB与MyISAM的最大区别：InnoDB支持事务，且InnoDB支持行级锁和表级锁（默认是行级锁，而MyISAM只支持表级锁。

一、事务定义

事务(Transaction)，是访问数据库的一个操作序列（程序的执行单元），要么所有都执行成功，不然失败。

二、事务的四个特性（ACID）

（1）原子性（Atomicity）

事务要么全部被执行，要么就全部不被执行。

（2）一致性（Consistency）

事务必须使数据库从一个一致性状态变换到另一个一致性状态.

一致状态的含义是数据库中的数据应满足完整性约束。

（3）隔离性（Isolation）

多个事务并发执行时，事务与事务之间互不干扰。

事务正确提交之前，它可能的结果不会显示给任何其他事务。

（4）持久性（Durability）

一旦事务提交，他对数据库的修改应该永久保存在数据库中。

三、事务的并发问题

多个并发执行的事务访问数据库中相同的数据时，如果没有采取必要的隔离机制，就会导致各种并发题。

（1）第一类丢失更新

撤销一个事务时，把其他事务已提交的更新数据覆盖。

（2）脏读

一个事务读到另一个事务未提交的更新数据。

（3）不可重复读

一个事务读到另一个事务已提交的更新数据。

（4）幻读

一个事务读到另一个事务已提交的新插入的数据。

不可重复读查询的都是同一个数据项，而幻读针对的是一批数据整体（比如数据的个数）。

（5）第二类丢失更新

这是不可重复读中的特例，一个事务覆盖另一个事务已提交的更新数据。

四、四种隔离级别

既要求高的隔离性（安全性），又要求高并发性，这种是不可能的任务。根据各种锁的操作机制出现一个事务隔离级别。即相同情况下的输入，不同隔离级别结果不同。

（1）Serializable (串行化)

一个事务在执行过程中完全看不到其他事务对数据库所做的更新。

可避免脏读、不可重复读、幻读的发生。

（2）Repeatable read (可重复读)

一个事务在执行过程中可以看到其他事务已经提交的新插入的记录，但是不能看到其他事务对已有记的更新。

可避免脏读、不可重复读的发生。

（3）Read committed (读已提交)

一个事务在执行过程中可以看到其他事务已经提交的新插入的记录，而且能看到其他事务已经提交的已有记录的更新。

可避免脏读的发生。

（4）Read uncommitted (读未提交)

一个事务在执行过程中可以拷贝其他事务没有提交的新插入的记录，而且能看到其他事务没有提交的已有记录的更新。

最低级别，任何情况都无法保证。

隔离级别：Serializable > Repeatable read > Read committed > Read uncommitted

在MySQL数据库中，支持上面四种隔离级别，默认的为Repeatable read (可重复读)。

MySQL数据库中查看当前事务的隔离级别：select @@tx_isolation;

MySQL数据库中设置事务的隔离 级别：

set [glogal | session] transaction isolation level 隔离级别名称;

或set tx_isolation='隔离级别名称';

注：设置数据库的隔离级别一定要是在开启事务之前。

其他数据库不一定完全实现上述4个隔离级别。

五、锁

共享锁（读锁）和排他锁（写锁）

粒度锁

1 表级锁

开销小，加锁快；不会出现死锁；锁定粒度大，发生锁冲突的概率最高，并发度最低。

存储引擎通过总是一次性同时获取所有需要的锁以及总是按相同的顺序获取表锁来避免死锁。

表级锁更适合于以查询为主，并发用户少，只有少量按索引条件更新数据的应用，如Web 应用

2 页级锁

BDB存储引擎采用的，这个存储引擎还支持表级锁。

3 行级锁

开销大，加锁慢；会出现死锁；锁定粒度最小，发生锁冲突的概率最低，并发度也最高。

行级锁只在存储引擎层实现，而Mysql服务器层没有实现。 行级锁更适合于有大量按索引条件并发更少量不同数据，同时又有并发查询的应用

MyISAM表级锁模式

MyISAM加表锁方式

查询表级锁争用情况

InnoDB的行锁模式

共享锁（S）

排他锁（X）

InnoDB的表锁模式（两个意向锁）

意向共享锁（IS）：事务在获取一条记录的共享锁之前必须先获取它所属表的意向共享锁。

意向排他锁（IX）：事务在获取一条记录的排他锁之前必须先获取它所属表的意向排他锁。

如果一个事务请求的锁模式与当前的锁兼容，InnoDB就将请求的锁授予该事务；反之，如果两者两不兼容，该事务就要等待锁释放。

默认情况下，表锁和行锁都是自动获得的，不需要额外的命令。如果我们需要显示进行锁表或进行事可控制，可以通过事务控制语句和锁定语句来完成。

对于 UPDATE、 DELETE 和 INSERT 语句， InnoDB

会自动给涉及数据集加排他锁（X）。

对于普通 SELECT 语句，InnoDB 不会加任何锁。

事务可以显式给记录集加共享锁或排他锁：

1 SELECT * FROM table_name WHERE ... LOCK IN SHARE MODE

2 SELECT * FROM table_name WHERE ... FOR UPDATE

优先考虑把数据库系统的隔离级别设为Read Commited，它能够避免脏读，而且具有较好的并发性。尽管它会导致不可重复读、虚读和第二类丢失更新这些并发问题，在可能出现这类问题的个别场合可以由应用程序采用悲观锁或乐观锁来控制。

MVVC一致性读

show status like "innodb_row_lock%" // 检查innodb_row_lock状态变量来分析系统上的行锁争夺况

六、事务的传播行为

规定了事务方法和事务方法发生嵌套调用时事务如何进行传播

有9种，一般我们只用一种：如果没有事务就创建一个

service层一个事务方法调用另一个不是事务的方法，如何处理这个不是事务的方法 - 为它创建一个事务

二、两种事务模式

（1）自动提交模式

每个SQL语句都是一个独立的事务，当数据库系统执行完一个SQL语句后，会自动提交事务。

（2）手动提交模式

必须由数据库客户程序显示指定事务开始边界和结束边界。

通过上述的分析，我们也理解了事务、锁和分离水平的概念，但锁和事务以及分离水平关系如何呢？际上，事务是解决多条sql执行执行过程的原子性、一致性、隔离性、持久性的整体解决方案，而事分离水平则是并发控制的整体解决方案，其实际是综合利用各种类型的锁来解决并发问题。锁是数据并发控制的内部基础机制。对应用开发人员来说，只有当事务分离水平无法解决并发问题和需求时，有必要在语句中手动设置锁。关于锁的锁定，对于UPDATE、DELETE和INSERT语句，InnoDB会自给涉及数据集加排他锁（X）；对于普通SELECT语句，InnoDB不会加任何锁，事务可以通过以下语句示给记录集加共享锁或排他锁。请注意InnoDB行锁是通过给索引上的索引项加锁来实现的，也就是，只有通过索引条件检索数据，InnoDB才使用行级锁，否则，InnoDB将使用表锁。

参考：

http://www.cnblogs.com/kristain/articles/2038397.html

http://www.letiantian.me/2014-06-18-db-undo-redo-checkpoint/

https://www.zhihu.com/question/30272728

http://donghui.blog.51cto.com/2709336/692586

MySQL的进阶实战篇