

一文理解MySQL的锁机制与死锁排查

发布于2021-06-25 21:49:53 阅读 213

MySQL的并发控制是在数据安全性和并发处理能力之间的权衡，通过不同的锁策略来决定对系统开销和性能的影响。

基础知识

为了后续的解释更加容易理解，这里列举一些基本概念的解释。

悲观锁

悲观锁指的是对数据被外界（包括本系统当前的其他事务，以及来自外部系统的事务处理）修改持保守态度，因此，在整个数据处理过程中，将数据处于锁定状态。

select...for update是MySQL提供的实现悲观锁的方式。在悲观锁的情况下，为了保证事务的隔离性，其它事务无法修改这些数据。

现在互联网高并发的架构中，受到fail-fast思路的影响，悲观锁已经比较少见了。

乐观锁

相对悲观锁而言，乐观锁机制采取了更加宽松的加锁机制。悲观锁依靠数据库的锁机制实现，以保证操作最大程度的独占性。但随之而来的就是数据库性能的大量开销，特别是对长事务而言，这样的开销往往无法承受。

而乐观锁机制在一定程度上解决了这个问题。乐观锁，大多是基于数据版本（Version）记录机制实现：通过为数据库表增加一个数字类型的 `version` 字段，当读取数据时，将 `version` 字段的值一同读出，数据每更新一次，对此 `version` 值+1。当我们提交更新的时候，判断数据库表对应记录的当前版本信息与第一次取出来的 `version` 值进行比对，如果数据库表当前版本号与第一次取出来的 `version` 值相等，则予以更新，否则认为是过期数据，返回更新失败。

锁的粒度

MySQL定义了两锁的粒度：表级、行级。

表锁

由MySQL Server控制，分为读锁和写锁。优点是开销小，加锁快；不会出现死锁；缺点是锁定粒度大，发生锁冲突的概率最高，并发度最低。表锁适合查询多、更新少的场景。

当对表加了读锁，则会话只能读取当前被加锁的表，其它会话仍然可以对表进行读取但不能写入。

当对表加了写锁，则会话可以读取或写入被加锁的表，其它会话不能对加锁的表进行读取或写入。

行锁

由存储引擎实现，InnoDB支持，而MyISAM不支持。优点是开销大，加锁慢；会出现死锁；缺点是锁定粒度最小，发生锁冲突的概率最低，并发度也最高。InnoDB引擎下默认使用行级锁。行级锁适合按索引更新频率高的场景。

InnoDB是MySQL最常用的存储引擎，本文以此为角度讲解MySQL的锁机制。有关MySQL存储引擎和有关B-Tree的知识，可以查看博客《[彻底搞懂MySQL的索引](#)》

行锁的分类

1. 记录锁

记录锁（Record lock）在唯一索引列或主键列记录上加锁，且该值存在，否则加锁类型为间隙锁。例如

```
SELECT a FROM t WHERE a = 12 FOR UPDATE
```

，对值为12的索引进行锁定，防止其它事务插入、删除、更新值为12的记录行。

2. 间隙锁

间隙锁（Gap Lock），只有在可重复读、串行化隔离级别才有，在索引记录之间的间隙中加锁，或者是在某一条索引之前或者之后加锁，并不包括该索引本身。

例如：

```
SELECT a FROM t WHERE a > 15 and a < 20 FOR UPDATE
```

，且a存在的值为1、2、5、10、15、20，则将(15,20)的间隙锁住。

间隙锁的范围：

1. 对主键或唯一索引当前读时，where条件全部精确命中(=或者in)，这种场景本身就不会出现幻读，所以只会加记录锁。
2. 没有索引的列当前读操作时，会加全表gap锁，生产环境要注意。（所有主键x锁，所有主键间隙gap锁）
3. 非唯一索引列，如果where条件部分命中(>、<、like等)或者全未命中，则会加附近Gap间隙锁。例如，某表数据如下，非唯一索引
2,6,9,9,11,15。 `delete from table where another_id = 9` 要操作非唯一索引列9的数据，gap锁将会锁定的列是(6,11)，该区间内无法插入数据。
4. 更多情况看文末的图片总结。

在使用范围条件检索并锁定记录时，间隙锁机制会阻塞符合条件范围内键值的并发插入，这往往会造成严重的锁等待。因此，在实际应用开发中，尤其是并发插入比较多的应用，要尽量优化业务逻辑，尽量使用相等条件来访问更新数据，避免使用范围条件。

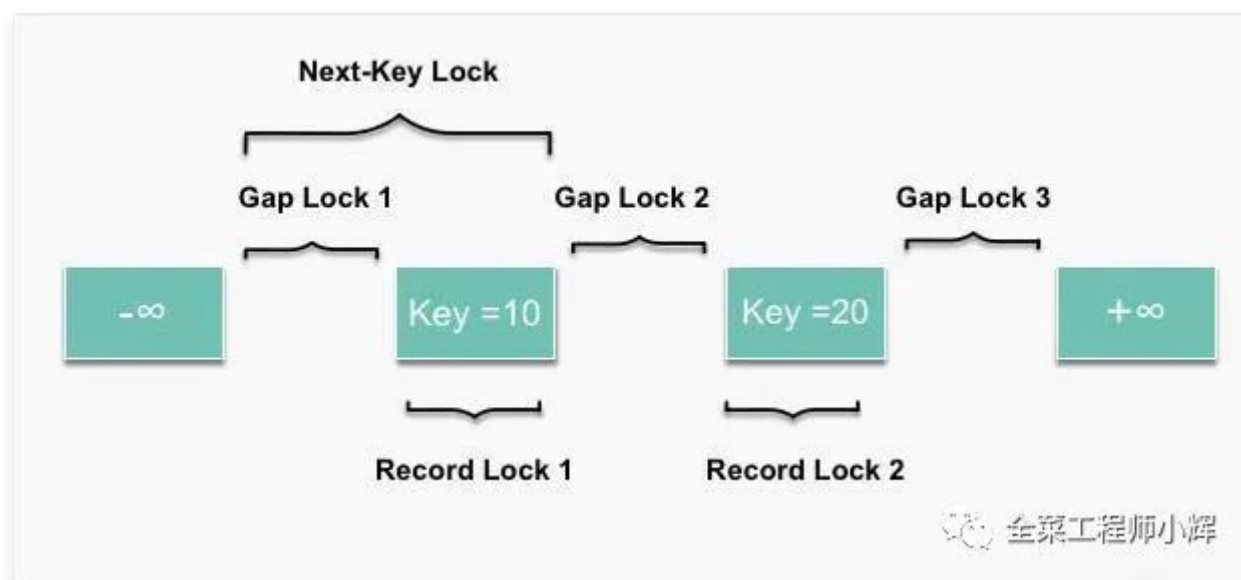
间隙锁和间隙锁之间是互不冲突的，间隙锁唯一的作用就是为了防止其他事务的插入，在RR（可重复读）级别下解决了幻读的问题。

例如id有3,4,5，间隙锁锁定id>3的数据，是指的4及后面的数字都会被锁定。这样的话加入新的数据id=6，就会被阻塞，从而避免了幻读。

快照读与当前读将在下一篇博客《一文理解MySQL的事务原则与事务隔离》进行详解

3. 临键锁

临键锁（Next-Key Lock）是记录锁和间隙锁的合集。只有在可重复读、串行化隔离级别才有。



例如一个索引有10,11,13,20这四个值。InnoDB可以根据需要使用记录锁将10, 11, 13, 20四个索引锁住，也可以使用间隙锁将 $(-\infty, 10)$, $(10, 11)$, $(11, 13)$, $(13, 20)$, $(20, +\infty)$ 五个范围区间锁住。而临键锁是记录锁和间隙锁的合集。

4. 插入意向锁

插入意向锁 (Insert Intention Locks)，是一种特殊的间隙锁，只有在执行INSERT操作时才会加锁，插入意向锁之间不冲突，可以向一个间隙中同时插入多行数据，但插入意向锁与间隙锁是冲突的，当有间隙锁存在时，插入语句将被阻塞，正是这个特性解决了幻读的问题。

假设有一个记录索引包含键值4和7，不同的事务分别插入5和6，每个事务都会产生一个加在4-7之间的插入意向锁，获取在插入行上的排它锁，但是不会被互相锁住，因为数据行并不冲突。

意向锁

innodb的意向锁主要用户多粒度的锁并存的情况。比如事务A要在一个表上加S锁，如果表中的一行已被事务B加了X锁，那么该锁的申请也应被阻塞。如果表中的数据很多，逐行检查锁标志的开销将很大，系统的性能将会受到影响。为了解决这个问题，可以在表级上引入新的锁类型来表示其所属行的加锁情况，这就引出了“意向锁”的概念。

举个例子，如果表中记录1亿，事务A把其中有几条记录上了行锁了，这时事务B需要给这个表加表级锁，如果没有意向锁的话，那就要去表中查找这一亿条记录是否上锁了。如果存在意向锁，那么假如事务A在更新一条记录之前，先加意向锁，再加X锁，事务B先检查该表上是否存在意向锁，存在的意向锁是否与自己准备加的锁冲突，如果有冲突，则等待直到事务A释放，而无须逐条记录去检测。事务B更新表时，其实无须知道到底哪一行被锁了，它只要知道反正有一行被锁了就行了。

意向锁的主要作用是处理行锁和表锁之间的矛盾，能够显示“某个事务正在某一行上持有了锁，或者准备去持有锁”。

锁的模式

共享锁和排它锁都是行级锁。意向共享锁和意向排他锁是表级锁。意向共享锁和意向排他锁都是系统自动添加和自动释放的，整个过程无需人工干预。

1. 共享锁

共享锁 (S锁, Shared Lock)：又称读锁，允许一个事务去读数据集，阻止其他事务获得该数据集的排他锁。共享锁与共享锁可以同时使用。举例：若事务T对数据对象A加上S锁，则事务T可以读A但不能修改A，其他事务只能再对A加S锁，而不能加X锁，直到T释放A上的S锁。这保证了其他事务可以读A，但在T释放A上的S锁之前不能对A做任何修改。

2. 排他锁

排他锁 (X锁, Exclusive Lock) : 又称写锁, 允许获取排他锁的事务更新数据, 阻止其他事务获得相同的数据集的共享锁和排他锁。排它锁与排它锁、共享锁都不兼容。举例: 若事务T对数据对象A加上X锁, 事务T可以读A也可以修改A, 其他事务不能再对A加任何锁, 直到T释放A上的锁。

3. 意向共享锁

意向共享锁 (IS锁, Intention Shared Lock) : 事务打算给数据行共享锁, 事务在给一个数据行加共享锁前必须先取得该表的IS锁。

4. 意向排他锁

意向排他锁 (IX锁, Intention Exclusive Lock) : 事务打算给数据行加排他锁, 事务在给一个数据行加排他锁前必须先取得该表的IX锁。

IS锁和IX锁的提出仅仅为了在之后加表级别的S锁和X锁时可以快速判断表中的记录是否被上锁, 以避免用遍历的方式来查看表中有没有上锁的记录, 也就是说其实IS锁和IX锁是兼容的, IX锁和IX锁是兼容的。

意向锁之间不会发生冲突, 但共享锁、排它锁、意向锁之间会发生冲突, 表级别各种锁的兼容性如下表所示。

兼容性	X	IX	S	IS
X	不兼容	不兼容	不兼容	不兼容
IX	不兼容	兼容	不兼容	兼容
S	不兼容	不兼容	兼容	兼容
IS	不兼容	兼容	兼容	兼容

5. 自增锁

AUTO-INC Locks, 自增锁, 是一种特殊的表锁。当表有设置自增 `auto_increment` 列, 在插入数据时会先获取自增锁, 其它事务将会被阻塞插入操作, 自增列+1后释放锁, 如果事务回滚, 自增值也不会回退, 所以自增列并不一定是连续自增的。(MySQL 从 5.1.22 版本开始, 引入了一种可选的轻量级锁 (mutex) 机制来代替AUTOINC锁。见于参考文档3)

6. 元数据锁

元数据锁（metadata lock），MySQL Server控制，表级锁，是维护表元数据的数据一致性，保证在表上有活动事务（显式或隐式）的时候，不可以对元数据进行写入操作。从MySQL5.5版本开始引入了MDL锁，来保护表的元数据信息，用于解决或者保证DDL操作与DML操作之间的一致性。

对于引入MDL，其主要解决了2个问题：

1. 事务隔离问题，比如在可重复隔离级别下，会话A在2次查询期间，会话B对表结构做了修改，两次查询结果就会不一致，无法满足可重复读的要求。
2. 数据复制的问题，比如会话A执行了多条更新语句期间，另外一个会话B做了表结构变更并且先提交，就会导致slave在重做时，先重做alter，再重做update时就会出现复制错误的现象。

每执行一条DML、DDL语句时都会申请MDL锁，DML操作需要MDL读锁，DDL操作需要MDL写锁（MDL加锁过程是系统自动控制，无法直接干预，读读共享，读写互斥，写写互斥），申请MDL锁的操作会形成一个队列，队列中写锁获取优先级高于读锁。

一旦出现MDL写锁等待，不但当前操作会被阻塞，同时还会阻塞后续该表的所有操作（不过在MySQL5.6的时候推出了online ddl机制，使得排队MDL写锁进行降级，防止对MDL读锁的阻塞）。

加锁时机

SELECT xxx 查询语句正常情况下为快照读，只加元数据读锁，直到事务结束。

SELECT xxx LOCK IN SHARE MODE 语句为当前读，加S锁和元数据读锁，直到事务结束。

SELECT xxx FOR UPDATE 语句为当前读，加X锁和元数据读锁，直到事务结束。

DML语句（INSERT、DELETE、UPDATE）为当前读，加X锁和元数据读锁，直到事务结束。

DDL语句（ALTER、CREATE等）加元数据写锁，且是隐式提交不能回滚，直到事务结束。

为什么DDL语句会隐式提交？因为DDL是数据定义语言，在数据库中承担着创建、删除和修改的重要的职责。一旦发生问题，带来的后果很可能是不可估量的。于是在每执行完一次后就进行提交，可以保证流畅性，数据不会发生阻塞，同时也会提高数据库的整体性能。

线上踩坑举例：由于DDL语句存在隐式提交，所以如果会话A开始了事务，进行了DML操作，然后进行了DDL操作，然后会话A回滚事务。此时会话A回滚的事务是一个空事务，因为DDL操作执行的时候会进行一次隐式提交

行锁锁住整表的场景

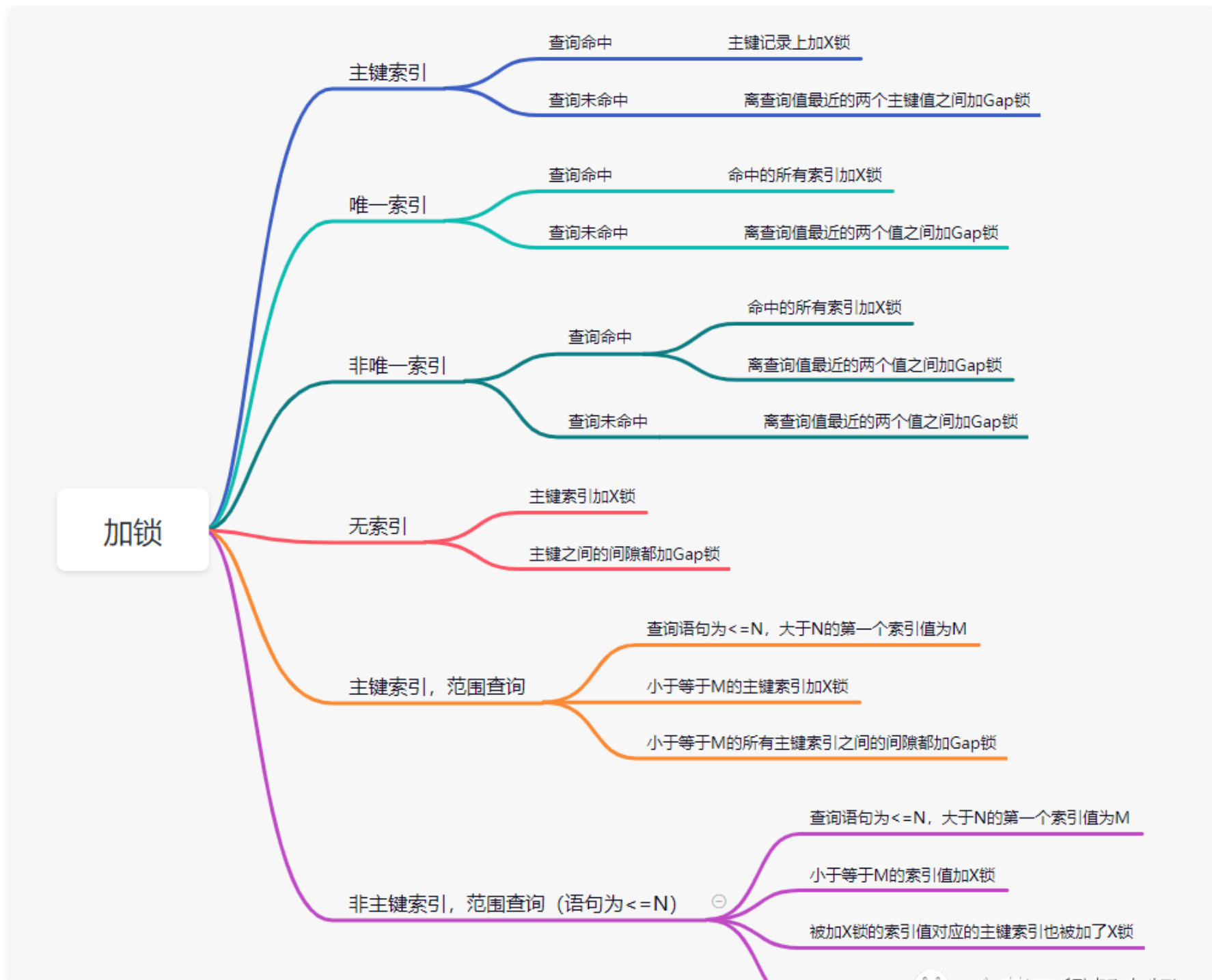
1. SQL语句没有使用索引会把整张表锁住。例如事务里进行整表update；用到前缀like；字段没有加索引；数据库优化将索引查询转全表扫描等等。

2. Mysql在5.6版本之前，直接修改表结构的过程中会锁表。

“查询每个表索引，并使用最佳索引，除非优化程序认为使用表扫描更有效。一次使用扫描是基于最佳索引是否跨越了表的30%以上，但是固定百分比不再决定使用索引还是扫描。现在，优化器更加复杂，并且根据附加因素（如表大小，行数和I / O块大小）进行估计。”见于参考文档1。

实验操作下的加锁情况分析

以下结论基于MySQL5.6，以InnoDB默认的RR级别来实验，只用来方便理解本文提到的锁机制。恐有纰漏，敬请谅解。



死锁排查

INFORMATION_SCHEMA提供对数据库元数据的访问、关于MySQL服务器的信息，如数据库或表的名称、列的数据类型或访问权限。其中有一个关于InnoDB数据库引擎表的集合，里面有记录数据库事务和锁的相关表。

MySQL有关事务和锁的四条命令：

1. `SELECT * FROM information_schema.INNODB_TRX;` 命令是用来查看当前运行的所有事务。
2. `SELECT * FROM information_schema.INNODB_LOCKS;` 命令是用来查看当前出现的锁。
3. `SELECT * FROM information_schema.INNODB_LOCK_WAITS;` 命令是用来查看锁等待的对应关系。
4. `show engine innodb status \G;` 命令是用来获取最近一次的死锁信息。

在查询结果中可以看到是否有表锁等待或者死锁。如果有死锁发生，可以通过 `KILL trx_mysql_thread_id` 来杀掉当前运行的事务。

查询事务与锁的命令

死锁是并发系统中常见的问题，同样也会出现在数据库MySQL的并发读写请求场景中。当两个及以上的事务，双方都在等待对方释放已经持有的锁或因为加锁顺序不一致造成循环等待锁资源，就会出现“死锁”。常见的报错信息为"Deadlock found when trying to get lock..."

MySQL死锁问题排查的常见思路：

1. 通过多终端模拟并发事务，复现死锁。
2. 通过上面四条命令，查看事务与锁的信息。
3. 通过explain可以查看执行计划。

发生死锁异常后，通过开启InnoDB的监控机制来获取实时的死锁信息，它会周期性（每隔 15 秒）打印 InnoDB 的运行状态到 mysqld服务的错误日志文件中。

InnoDB的监控较为重要的有标准监控（Standard InnoDB Monitor）和锁监控（InnoDB Lock Monitor），通过对应的系统参数可以将其开启。

```
1 | set GLOBAL innodb_status_output=ON; 开启标准监控
2 | set GLOBAL innodb_status_output_locks; 开启所监控
```

另外，MySQL 提供了一个系统参数 `innodb_print_all_deadlocks` 专门用于记录死锁日志，当发生死锁时，死锁日志会记录到 MySQL 的错误日志文件中。

另外，MySQL 提供了一个系统参数 `innodb_print_all_deadlocks` 专门用于记录死锁日志，当发生死锁时，死锁日志会记录到 MySQL 的错误日志文件中。

如何尽可能避免死锁

1. 合理的设计索引，区分度高的列放到组合索引前面，使业务SQL尽可能通过索引定位更少的行，减少锁竞争。
2. 尽量按主键/索引去查找记录，范围查找增加了锁冲突的可能性，也不要利用数据库做一些额外额度计算工作。比如有的程序会用到 `select ... where ... order by rand();` 这样的语句，类似这样的语句用不到索引，因此将导致整个表的数据都被锁住。
3. 大事务拆小。大事务更倾向于死锁，如果业务允许，将大事务拆小。
4. 以固定的顺序访问表和行。比如两个更新数据的事务，事务A更新数据的顺序为1，2;事务B更新数据的顺序为2，1。这样更可能会造成死锁。
5. 降低隔离级别。如果业务允许，将隔离级别调低也是较好的选择，比如将隔离级别从RR调整为RC，可以避免掉很多因为gap锁造成的死锁。

参考文档：

1. <http://www.searchdoc.cn/rdbms/mysql/dev.mysql.com/doc/refman/5.7/en/where-optimization.com.coder114.cn.html>
2. <https://dev.mysql.com/doc/refman/5.7/en/innodb-locking.html#innodb-record-locks>
3. <https://dev.mysql.com/doc/refman/5.7/en/innodb-auto-increment-handling.html>

文章分享自微信公众号：

全菜工程师小辉

复制公众号名称

本文参与 [腾讯云自媒体分享计划](#)，欢迎热爱写作的你一起参与！

如有侵权，请联系 yunjia_community@tencent.com 删除。

[举报](#)

点赞 2

[分享](#)[登录](#) 后参与评论

0 条评论

相关文章


一次MySQL死锁问题的排查与分析(一)

笔者负责的一个系统最近有新功能上线后突然在预警模块不定时报出MySQL死锁导致事务回滚。幸亏，上游系统采用了异步推送和同步查询结合的方式，感知到推送失败及时进行...

 Throwable


记一次神奇的Mysql死锁排查

说起Mysql死锁，之前写过一次有关Mysql加锁的基本介绍，对于一些基本的Mysql锁或者死锁都有一个简单的认识，可以看下这篇文章为什么开发人员需要了解分布式...

 用户5397975

C# 死锁的原理与排查方法详解

线程死锁是指由于两个或者多个线程互相持有对方所需要的资源，并且互相等待对方释放资源，导致这些线程都处于等待状态，无法继续执行。如果线程都不主动释放所占...

 用户9127601

全面了解mysql锁机制（InnoDB）与问题排查

MySQL/InnoDB的加锁，一直是一个常见的话题。例如，数据库如果有高并发请求，如何保证数据完整性？产生死锁问题如何排查并解决？下面是不同锁等级的区别

 蒋老湿

谈谈 MySQL 锁机制

因为数据也是一种供许多用户共享的资源，如何保证数据并发访问的一致性、有效性是所有数据库必须解决的一个问题，锁冲突也是影响数据库并发访问性能的一个重要因素，所以进...

 iMike

MySQL 中的 锁机制 详解

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。



suveng

【建议收藏】MySQL 三万字精华总结 —锁机制和性能调优（四）

在数据库中，除传统的计算资源（如CPU、RAM、I/O等）的争用以外，数据也是一种供许多用户共享的资源。数据库锁定机制简单来说，就是数据库为了保证数据的一致性，...



码农编程进阶笔记

史上最全MySQL锁机制

因为数据也是一种供许多用户共享的资源，如何保证数据并发访问的一致性、有效性是所有数据库必须解决的一个问题，锁冲突也是影响数据库并发访问性能的一个重要因素，所以进...



数据和云

重新学习Mysql数据库7：详解MyIsam与InnoDB引擎的锁实现

本文是微信公众号【Java技术江湖】的《重新学习MySQL数据库》其中一篇，本文部分内容来源于网络，为了把本文主题讲得清晰透彻，也整合了很多我认为不错的技术博客...



Java技术江湖

[精选]详细介绍MySQL中常见的锁

当数据库的隔离级别为Repeatable Read或Serializable时,我们来看这样的两个并发事务（场景一）：



码农编程进阶笔记

MySQL中锁机制超详细解析

锁是计算机协调多个进程或纯线程并发访问某一资源的机制。在数据库中，除传统的计算资源（CPU、RAM、I/O）的争用以外，数据也是一种供许多用户共享的资源。如何保...



博文视点Broadview

锁系列-Mysql中的锁

在计算机科学中，锁是在执行多线程时用于强行限制资源访问的同步机制，即用于在并发控制中保证对互斥要求的满足。目录： 1、行级锁、表级锁、页级锁 2、共享锁和排它...



ImportSource

MySQL 锁机制——必知必会

MyISAM表的读和写是串行的，但这是就总体而言的。在一定条件下，MyISAM表也支持查询和插入操作的并发进行。



高广超

再谈mysql锁机制及原理—锁的诠释

加锁是实现数据库并发控制的一个非常重要的技术。当事务在对某个数据对象进行操作前，先向系统发出请求，对其加锁。加锁后事务就对该数据对象有了一定的控制，在该事务释放...



周陆军

Mysql之锁、事务绝版详解---干货！

数据库锁定机制简单来说，就是数据库为了保证数据的一致性，而使各种共享资源在被并发访问变得有序所设计的一种规则。对于任何一种数



码农编程进阶笔记

MySQL锁机制及优化

总的来说，MySQL各存储引擎使用了三种类型（级别）的锁定机制：行级锁定，页级锁定和表级锁定。下面我们先分析一下MySQL这三种锁定的特点和各自的优劣所在。



lyb-geek

记一次排查DB死锁的分析

文章摘要 在线上环境遇到数据库死锁问题该如何分析并解决问题呢？虽然很多童鞋在学数据库课程时都了解数据库隔离级别、死锁和事务等概念，但在测试/线上环境遇到死锁却...



企鹅号小编


漫谈死锁

一 前言 死锁是每个MySQL DBA 都会遇到的技术问题，本文是自己针对死锁学习的一个总结,了解死锁是什么，MySQL如何检测死锁，处理死锁，死锁的案例，...



用户1278550

[更多文章](#)

社区	活动	资源	关于	云+社区
专栏文章	原创分享计划	技术周刊	视频介绍	 扫码关注云+社区 领取腾讯云代金券
阅读清单	自媒体分享计划	社区标签	社区规范	
互动问答	邀请作者入驻	开发者实验室	免责声明	
技术沙龙	自荐上首页		联系我们	
技术快讯	在线直播		友情链接	
团队主页	生态合作计划			
开发者手册				
腾讯云TI平台				

热门产品	域名注册	云服务器	区块链服务	消息队列	网络加速	云数据库	域名解析
	云存储	视频直播					
热门推荐	人脸识别	腾讯会议	企业云	CDN 加速	视频通话	图像分析	MySQL 数据库
	SSL 证书	语音识别					
更多推荐	数据安全	负载均衡	短信	文字识别	云点播	商标注册	小程序开发
	网站监控	数据迁移					