**数据库锁机制**

**锁：资源的并发访问控制**

**锁的分类：**

1. **操作类型分类：读锁/写锁**
2. **锁定粒度分类：表锁/行锁**

**读锁(共享S锁)：针对同一份数据，多个读操作可以同时进行而不会相互阻塞。**

**写锁(排他X锁)：当前写操作没有完成前，它会阻塞其他写锁和读锁。**

**表锁：偏向MyISAM存储引擎，开销小，加锁快，无死锁，锁定粒度大，发生锁冲突的概 率最高，并发度最低。**

**查看表的锁定状态：show open tables;#In use = 0代表该表上没有锁**

**手动增加表锁(不会自动释放，必须手动释放)：lock table 表名 read/write;**

**手动释放表锁(不会自动释放，必须手动释放)：unlock tables;**

**注意：只会释放当前事务锁住的表**

**查看系统上的表锁定情况：**

**show status like ‘table%’;**

**Table\_locks\_immediate：产生表级锁定的次数**

**Table\_locks\_waited：出现表级锁定导致的等待的次数，此值较高说明存在较为严 重的表级锁争抢情况**

**MyISAM存储引擎：**

**MyISAM在执行查询语句前，会自动给涉及到的所有表加读锁(读完自动释放)，在执行 更新操作前，会自动给涉及到的表加写锁(写完自动释放)。**

**读写锁对数据操作和系统性能的影响：**

**对MyISAM表加读锁(读操作)：**

**不影响当前线程和其他线程对该表的读操作，当前线程对该表的更新操作将 会报错，其他线程对该表的更新操作将会阻塞，直至表锁释放时才会执行。 并且加了表读锁的线程在释放锁住的表之前不能对任何其他表进行读或写 操作(无论是否加锁)。**

**对MyISAM表加写锁(写操作)-：**

**不影响当前线程对该表的读或写操作。其他线程对该表的查询或更新操作 将会阻塞，直至表锁释放时才会执行。并且加了表读锁的线程在释放锁住 的表之前不能对任何其他表进行读或写操作(无论是否加锁)。**

**注意：无论是加表读锁还是表写锁，一个线程在对第二张表加锁时会自动释放掉第 一张表上的锁。即一个线程最多同时锁定一张表。**

**MyISAM的读写锁调度是写优先，这也是MyISAM不适合做以写为主的引擎。因为写锁 后，其他线程不能做任何操作，大量的更新操作会使查询很难得到锁，从而造成永远 阻塞。**