#### MySQL锁

**不同存储引擎支持不同锁机制**

**MyISAM、MEMORY存储引擎采用表级锁**

**InnoDB存储引擎支持行级锁，也支持表级锁，默认行级锁**

##### 行锁 表锁

行锁、表锁、库锁

1. 行锁：访问数据库，绑定整个行
2. 表锁：访问数据库，绑定整个表

区别

* 表锁

开销小，加锁快，不会出现死锁，锁定粒度大，发生锁冲突概率高，并发度最低。

* 行锁

开销大，开锁慢，会出现死锁，锁定粒度小，发生锁冲突的概率低，并发度高。

##### 乐观锁 悲观锁

悲观锁：

每次拿数据都会上锁、其他请求要拿数据只能block直到拿到锁 Synchronized

悲观锁适合写操作多的场景 先加锁可以保证写操作时数据正确。

乐观锁适合读操作多的场景，不加锁让读操作性能大幅提升。

乐观锁：

更新的时候会判断一下（多用于多读的应用类型，提高吞吐量）

CAS算法（Compare and swap）

需要读写的内存值V

进行比较的值 A

写入的新值 B

当且仅当预期A与内存值V相同时，将内存值V修改为B，否则返回V

###### 区别

乐观锁适用于写比较少的情况下，如果经常产生冲突，上层应用会不断retry，反倒降低性能，这种情况下用悲观锁比较合适。

##### 共享锁（悲观锁的一种）

Lock in share mode 共享锁保证并发读非常高效

同时访问资源

InnoDB什么时候使用表锁

1. 事务需要更新大部分数据、表大、使用默认行锁效率低，而且可能造成其他事务长时间锁等待和锁冲突
2. 事务设计多个表，比较复杂，容易引起死锁，造成大量事务回滚，故而一次性锁定事务涉及的表，从而避免死锁、减少数据库因事务回滚带来太多的开销

如果类似情况过多，就要使用MyISAM表

###### 表锁和行锁应用场景

表级锁使用与并发性不高，以查询为主，少量更新的 应用 （小型web应用）

行级锁适用于高并发环境下，对事务完整性要求较高的系统 （在线事务处理系统）

##### 独享锁（悲观锁的一种）

Java ReentrantLock

AQS