1 宏观

1.1 数据库锁

- 数据库锁适用于集群
- 粒度小，更方便控制

1.2 代码锁

- 需要复杂的处理，才能作用于集群
- 粒度大

2 微观

2.1 行锁 & 表锁

2.1.1 说明

只有「明确」指定主键，才会执行锁，否则将会执行表锁

2.1.2 示例

假设有个表 products ，字段id、name、type，id是主键。

- 无锁

```
# 明确指定主键，但不存在该主键的值(没有数据，当然不会有锁)
SELECT * FROM products WHERE id=-1 FOR UPDATE;
```

- 行锁

```
# 明确指定主键
SELECT * FROM products WHERE id=3 FOR UPDATE;
SELECT * FROM products WHERE id=3 AND type=1 FOR UPDATE;
```

- 表锁

```
# 主键不明确
SELECT * FROM products WHERE name='Mouse' FOR UPDATE;
SELECT * FROM products WHERE id<>'3' FOR UPDATE;
SELECT * FROM products WHERE id LIKE '3' FOR UPDATE;
```

2.1.3 注意

- 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。
- 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。

2.2 行锁算法

2.2.1 Record Lock（普通行锁）

- 对于键值在条件范围内，且存在的记录，使用" Record Lock "，即普通的行锁机制；

2.2.2 Gap Lock（间隙锁）

- 对于键值在条件范围内但并不存在的记录，叫做" 间隙（GAP) "，InnoDB会对这个“间隙”加锁，这种锁机制就是所谓的" Gap Lock "(间隙锁)；

2.2.3 Next-Key Lock（行 & 间隙）

- 对于存在于不存在的数据同时加锁，则称为" Next-Key Lock "；
- **Next-Key Lock包含Record Lock和Gap Lock；**

```
# 假如user表中只有101条记录，empid的值是1,2,...,100,101
# 范围条件的检索，会对值为101的记录加锁，也会对大于101（不存在）加锁
# 由于两个锁同时存在，则此处为 Next-Key Lock
select * from  user where user_id > 100 for update;
```

2.3 表锁算法

2.3.1 意向锁

- 当一个事务带着表锁去访问一个被加了行锁的资源，那么，此时，这个行锁就会升级为意向锁，将表锁住。
- 常用的意向锁有：意向共享锁，意向排它锁，共享意向排它锁

2.3.2 自增锁

- 事务插入自增类型的列时获取自增锁

> 如果一个事务正在往表中插入自增记录，所有其他事务的插入必须等待

3 实现

3.1 共享锁 & 排它锁

> 行锁和表锁是锁粒度的概念，共享锁和排它锁是他们的具体实现

3.1.1 共享锁（S）：读锁

- 允许一个事务去读一行，阻止其他事务获取该行的排它锁。
- 多事务时，只能加共享读锁，不能加排他写锁；单事务时，可以加任何锁。
- 一般理解为：能读，不能写。

3.1.2 排它锁（X）：写锁

- 允许持有排它锁的事务读写数据，阻止其他事物获取该数据的共享锁和排它锁。
- 其他事务不能获取该数据的任何锁，直到排它锁持有者释放。
- 不能获取任何锁，不代表不能无锁读取。

注意

- 排它锁指的是，在某个事务获取数据的排它锁后，其他事务不能获取该数据的任何锁，**并不代表其他事务不能无锁读取该数据。**
    - 无锁

        - **select ... from**
    - 共享锁
        - **select ... lock in share mode**
    - 排它锁
        - **update**
        - **delete**
        - **insert**
        - **select ... for update**
- MySQL8.0 中，使用 FOR SHARE 替代了 LOCK IN SHARE MODE，但仍然支持 LOCK IN SHARE MODE；
  虽然是等价的，但是 FOR SHARE 支持 NOWAIT 、 SKIP LOCKED 等，配合自旋，可以实现高效的等待队列。

3.2 乐观锁 & 悲观锁

> 不管是什么锁都需要增加，需加失败重试

3.2.1 乐观锁

- 通过版本号来进行更新的操作属于乐观锁

```
update tab set name = 'xxx' where id = 1 and version = xxx
```

3.2.2 悲观锁

> 共享锁 & 排它锁都是悲观锁的具象实现

- 显示地控制行或表锁属于悲观锁