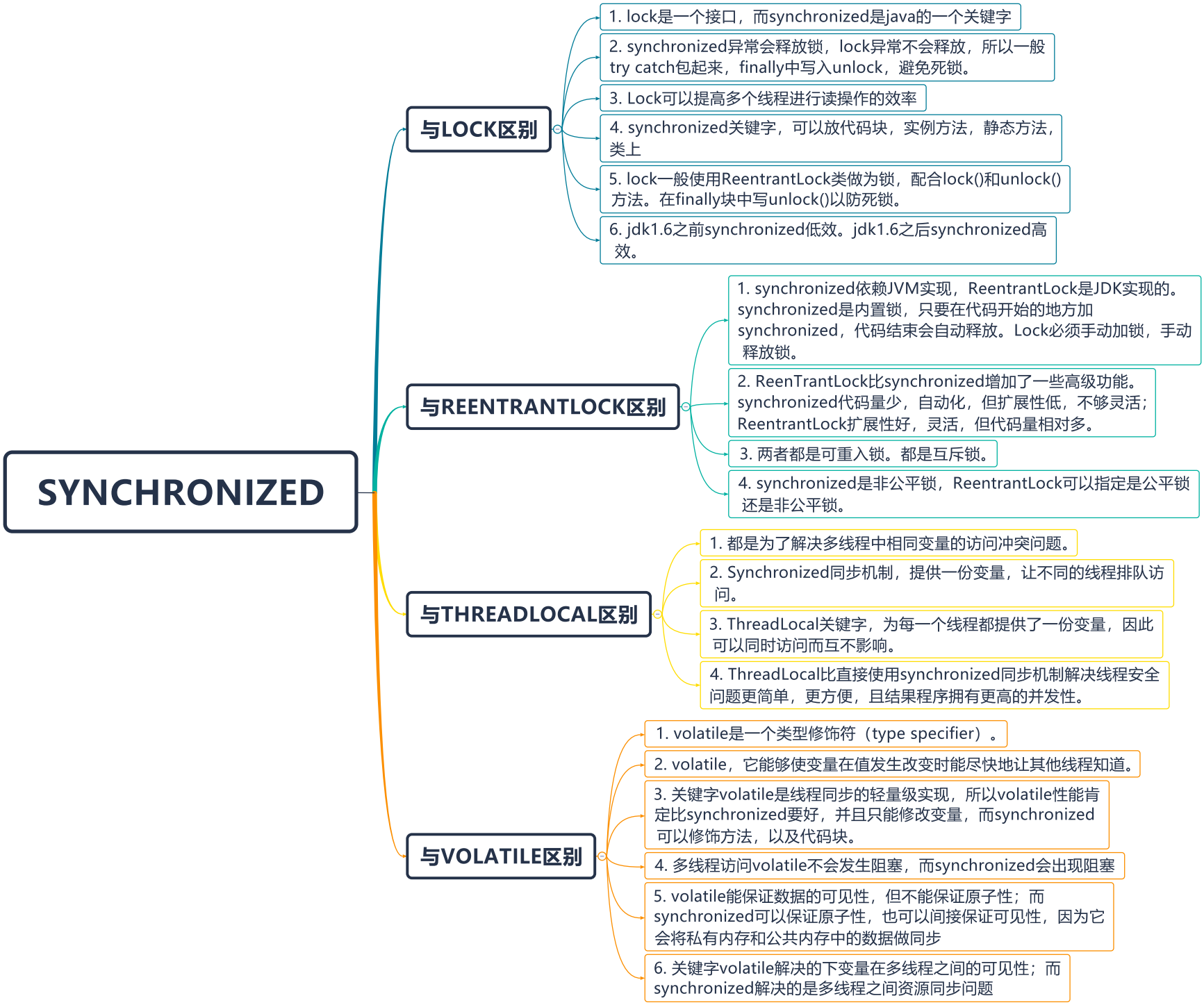
SYNCHRONIZED
与LOCK区别
1. lock是一个接口，而synchronized是java的一个关键字
2. synchronized异常会释放锁，lock异常不会释放，所以一般try catch包起来，finally中写入unlock，避免死锁。
3. Lock可以提高多个线程进行读操作的效率
4. synchronized关键字，可以放代码块，实例方法，静态方法，类上
5. lock一般使用ReentrantLock类做为锁，配合lock()和unlock()方法。在finally块中写unlock()以防死锁。
6. jdk1.6之前synchronized低效。jdk1.6之后synchronized高效。
与REENTRANTLOCK区别
1. synchronized依赖JVM实现，ReentrantLock是JDK实现的。synchronized是内置锁，只要在代码开始的地方加synchronized，代码结束会自动释放。Lock必须手动加锁，手动释放锁。
2. ReenTrantLock比synchronized增加了一些高级功能。synchronized代码量少，自动化，但扩展性低，不够灵活；ReentrantLock扩展性好，灵活，但代码量相对多。
3. 两者都是可重入锁。都是互斥锁。
4. synchronized是非公平锁，ReentrantLock可以指定是公平锁还是非公平锁。
与THREADLOCAL区别
1. 都是为了解决多线程中相同变量的访问冲突问题。
2. Synchronized同步机制，提供一份变量，让不同的线程排队访问。
3. ThreadLocal关键字，为每一个线程都提供了一份变量，因此可以同时访问而互不影响。
4. ThreadLocal比直接使用synchronized同步机制解决线程安全问题更简单，更方便，且结果程序拥有更高的并发性。
与VOLATILE区别
1. volatile是一个类型修饰符（type specifier）。
2. volatile，它能够使变量在值发生改变时能尽快地让其他线程知道。
3. 关键字volatile是线程同步的轻量级实现，所以volatile性能肯定比synchronized要好，并且只能修改变量，而synchronized可以修饰方法，以及代码块。
4. 多线程访问volatile不会发生阻塞，而synchronized会出现阻塞
5. volatile能保证数据的可见性，但不能保证原子性；而synchronized可以保证原子性，也可以间接保证可见性，因为它会将私有内存和公共内存中的数据做同步
6. 关键字volatile解决的下变量在多线程之间的可见性；而synchronized解决的是多线程之间资源同步问题