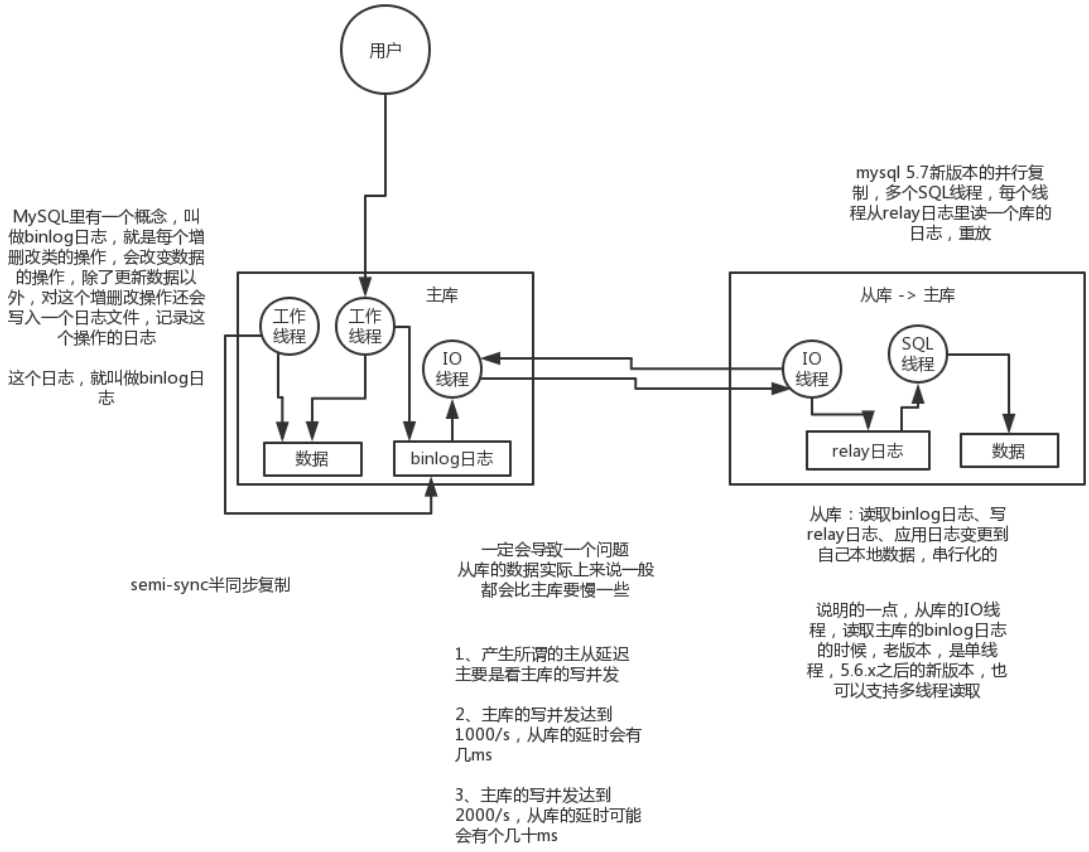
mysql读写分离

##问题

```
```

```
读写分离  主库负责写，从库负责读
```

主从复制



主从库的复制，从库的同步数据有一定的延迟

因为主库的写入操作是可以并发执行的，而从库的SQL线程对数据的恢复是单线程的，串行化的，单线程执行速度会慢一些

主从复制的数据的丢失

当对数据的操作已经写入了binlog日志中了，但还没来得及同步到从库时，主库宕机。

这个所谓半同步复制，

semi-sync复制，指的就是主库写入binlog日志之后，就会将强制此时立即将数据同步到从库，从库将日志写入自己本地的relay log之后，接着会返回一个ack给主库，主库接收到至少一个从库的ack之后才会认为写操作完成了。没有收到ack就认为写操作没有成功，半同步指知识吧binlog日志拉到从库，还没有对数据进行恢复

解决主从复制的延迟：

所谓并行复制，指的是从库开启多个线程，并行读取relay log中不同库的日志，然后并行重放不同库的日志，这是库级别的并行。

（3）mysql主从同步延时问题（精华）

场景：刚进行查询之后立马进行查询时，发现查询不到，就是因为主库写入后还没有同步到从库上时，对从库进行了查询的操作。

线上确实处理过因为主从同步延时问题，导致的线上的bug，小型的生产事故

show status，Seconds_Behind_Master，你可以看到从库复制主库的数据落后了几ms

其实这块东西我们经常会碰到，就比如说用了mysql主从架构之后，可能会发现，刚写入库的数据结果没查到，结果就完蛋了。。。。

所以实际上你要考虑好应该在什么场景下来用这个mysql主从同步，建议是一般在读远远多于写，而且读的时候一般对数据时效性要求没那么高的时候，用mysql主从同步

所以这个时候，我们可以考虑的一个事情就是，你可以用mysql的并行复制，但是问题是那是库级别的并行，所以有时候作用不是很大

所以这个时候。。通常来说，我们会对于那种写了之后立马就要保证可以查到的场景，采用强制读主库的方式，这样就可以保证你肯定的可以读到数据了吧。其实用一些数据库中间件是没问题的。

一般来说，如果主从延迟较为严重

1、分库，将一个主库拆分为4个主库，每个主库的写并发就500/s，此时主从延迟可以忽略不计
2、打开mysql支持的并行复制，多个库并行复制，如果说某个库的写入并发就是特别高，单库写并发达到了2000/s，并行复制还是没意义。28法则，很多时候比如说，就是少数的几个订单表，写入了2000/s，其他几十个表10/s。
3、重写代码，写代码的同学，要慎重，当时我们其实短期是让那个同学重写了一下代码，插入数据之后，直接就更新，不要查询
4、如果确实是存在必须先插入，立马要求就查询到，然后立马就要反过来执行一些操作，对这个查询设置直连主库。不推荐这种方法，你这么搞导致读写分离的意义就丧失了