

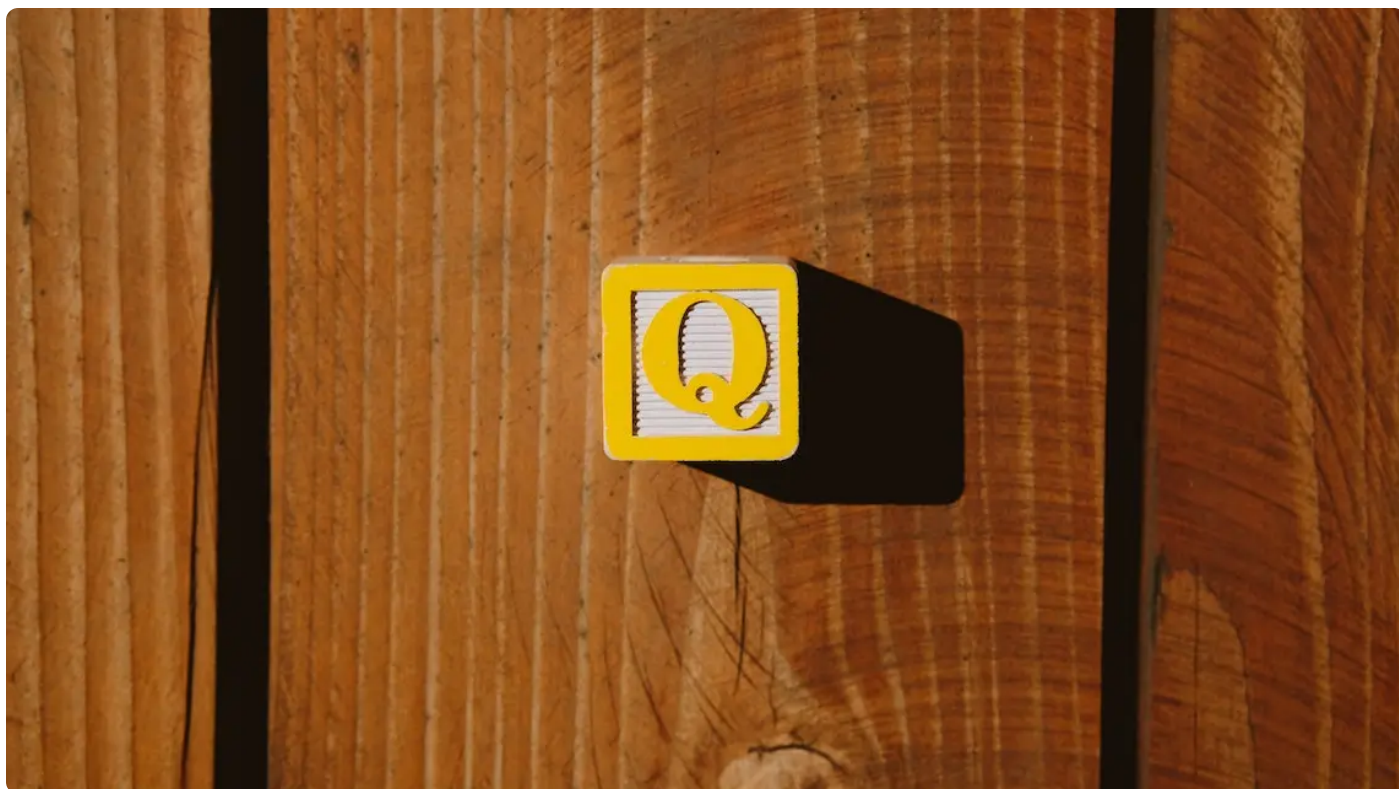
答疑课堂 | 思考题答案（一）

2022-12-12 徐长龙 来自北京



[课程介绍 >](#)

《高并发系统实战课》



讲述：宇新

时长 01:04 大小 1000.33K



你好，我是编辑小新。

今天是一节答疑课。我们的专栏已经步入尾声。除了紧跟更新节奏的第一批同学，也很开心看到有更多新朋友加入到这个专栏的学习中。

很多同学的留言也是这门课的亮丽风景，给专栏增色了不少。大部分的疑问，老师都在留言区里做了回复，期待更多同学在留言区里分享经验，提出问题或尝试解答他人的疑问，我们来共建一个共同学习、积极交流的良好氛围。

为了给你留下足够的思考和研究的时间，我们选择用加餐的方式，公布每节课的参考答案，也会精选一些优秀同学的答案展示出来。这里要提醒一下，建议你先做了自己的思考后，再核对答案。另外每节课都有超链接，方便你跳转回顾。

[🔗 第一节课](#)

Q: 请你思考一下，用户邀请其他用户注册的记录，属于历史记录还是关系记录？

A: 用户邀请其他用户注册的记录，我认为属于关系记录。



虽然这种记录有历史记录特征，但是被邀请注册的用户只能被邀请一次，所以总量是可控的。同时，这种表的用途很明确，表内记录的是关系记录，查询时会按邀请人或被邀请人 uid 进行查询。

留言区里也有不少精彩的答案，推荐你去看看。比如 @移横为固 的答案，这里我也复制过来：

一开始觉得注册邀请表应该作为历史表。思考了下作为关系表也是可以的。

在满足下面的注册邀请前提下：

1. 邀请人用类似二维码分享方式，注册人主动扫码注册（不使用点对点邀请，被邀请人可能不接受）。

2. 只能注册成功一次。

这样每一条邀请记录都是一个用户的注册记录：可以定义如下字段：（邀请者，注册人，注册时间，邀请方式）。

表的字段结构都非常简单，记录的总量最多就是账号量，并不会随时间不断膨胀。因此可以胜任关系表的查询需求。

在实际项目中，我们会遇到很多类似情况，需要我们预防超出预期的操作，核心在于我们怎么约束使用表的人，以及我们要怎么用表里的数据。

🔗 第二节课

Q1: 使用 BloomFilter 识别热点 key 时，有时会识别失误，进而导致数据没有找到，那么如何避免这种情况呢？

A1: 有一个特殊方法能降低概率，原始 key 通过 BloomFilter 检测一次，md5 后再通过另外一个 BloomFilter 再测一次。

Q2: 使用 BloomFilter 只能添加新 key, 不能删除某一个 key, 如果想更好地更新维护, 有什么其他方式吗?



A2: 请参考 Redis 的 Cuckoo Filter 的实现。

@不吃包子同学的思考也不错, 这里我一并放出供你参考。

针对 1.2 的问题, 搜索到了如下解决方案: 调整布隆过滤器参数或者用布谷鸟过滤器。

我想说说我自己的看法, 针对误判的情况, 能不能再加一层缓存? 比如说一个数据被误判为有, 则去查询数据库了, 这个时候为空, 记到缓存里面, 如果下次再访问该数据的时候, 直接从缓存返回。针对问题 2, 同样也维护一个删除的缓存。

🔗 第三节课

Q: 用户如果更换了昵称, 如何快速更换 token 中保存的用户昵称呢?

A: 在更换用户昵称, 同时更换修改端的 token。如果我们的用户有多个客户端, 那么可以利用缓存更新提及的 Version 版本号, 让客户端定期检测判断 token 是否需要更换。

对于这个问题, 置顶留言里 @徐曙辉同学的回答也很有趣:

如果我来做快速更换昵称的功能, 有两种方式:

a. 在用户修改昵称后, 内存中加入一个用户标识, 解析 token 后读取该标识, 有则返回特定 code, 让客户端重新拿 token。甚至可以不用客户端参与, 返回 301 重定向到获取新 token 的路由。

b. token 里面不存用户信息, 只存用户 ID, 需要用户信息的时候从缓存读。

徐同学的第一个解法很暴力, 但是很有趣。

第二个方式也很有意思, 这里我也补一个应用技巧: 我们可以通过设定固定网址 user/ 用户 uid/heaer.jpg 方式, 直接获取用户头像, 这样也不用考虑更新问题了。

围绕着我的补充，这个话题还有后续讨论，我也一并展示出来，仍然是徐同学的回答：

按这样做，头像可以 `http://xx.com/user/ 用户 ID/header.jpg`，静态文件可以，因为反正都是远程 `http` 渲染。但是昵称和其他信息都这样处理，每一项都放到远程地址性能不是很好，是不是可以 `http://xxx.com/user/ 用户 ID/info.json`，再反序列化呢？

这样确定是占了额外的存储空间，优点是不用查 `DB` 和缓存，减少它们的压力，在 `Web` 应用中，用户信息读取挺频繁。

我认为这个思路很优秀，建议尽量使用对象存储做。关于对象存储的话题，你还可以参考 [🔗 第二十一节课](#) 的内容，我在里面详细分享了对象存储如何管理小文件和大文本。

[🔗 第四节课](#)

Q：如果 `Otter` 同步的链路是环形的，那么如何保证数据不会一直循环同步下去？

A： `Otter` 在事务头尾插入同步标识，解析时会通过这种方式防止发起方再执行同样事务。

[🔗 第五节课](#)

Q：请你思考一下，为什么 `Raft` 集群成员增减需要特殊去做？

A：这是一个复杂的话题，我觉得后面这篇文章分析得相对完整，你可以点击 [🔗 这里](#) 查看原文。

[🔗 第六节课](#)

Q：这节课中的有些概念与 `DDD` 是重合的，但是仍有一些细小的差异，请你对比一下 `MVC` 三层方式和 `DDD` 实现的差异。

A：这个问题没有标准答案。我们结合同学的回答一起看看。

@Geek_994f3b 同学的回答是这样的：

我个人觉得两者只是作用域范围不同，从程序的角度看，MVC 模式用在线程间 (单体应用)，而 DDD 用在进程间 (微服务)，那么 MVC + RPC 协议 + 业务拆分 \approx DDD (个人愚见：)，像是在单体上多套了一层。



@徐曙辉同学的回答是这样的：

MVC 是项目目录功能分层设计，偏框架，而 DDD 更多是业务实体领域和彼此之间的关系，偏业务。

再补充一下我的想法，建议结合贫血模型和充血模型区别，以及领域模型和 Service 的区别来考虑这个问题。

🔗 第七节课

Q1：请你思考一下，通过原子操作 + 拆开库存方式实现库存方案时，如何减少库存为 0 后接口缓慢的问题？

A1：我们可以再设置一个 key，标注还有哪些 key 有库存。

Q2：我们这节课的内容并不仅仅在讲库存，还包含了大量可实现的锁的使用方式，请你分享一些实践过程中常见但不容易被发现的精妙设计。

A2：这道题没有标准答案，希望你做一个有心人，在实践中多多关注各种锁的有趣设计。

🔗 第八节课

Q：用什么方法能够周期检查出两个系统之间不同步的数据？

A：在数据上增加修改时间或版本号，每次更新的时候同步更新版本号，通过版本号能够很好地帮助我们识别哪个数据是最新的。

我们再看看 @LecKeyT 同学的回答：

每条数据都有唯一的数据标识（一般是自增 id，或者有规律一串数字唯一 id），而且一般都是小到大，根据这个最大值应该就能判断出来。如果数据不同步应该找到对应数据节点做

补偿操作。

看到他的回答后，我又追加了一个提问“如何避免更新操作同一条数据”？你也可以自行思考一下，再继续往下看。

后面的回答同样来自 @LecKeyt 同学：

更新带来的数据不一致的情况，我个人认为要看具体业务，如果是实时性要求不高的可以用事件队列处理，如果要求强一致性那最好的方式应该是分布式事务保证了。

🔗 第九节课

Q：现在市面上有诸多分布式实现方式，你觉得哪一种性能更好？

A：建议考虑使用 AT 或 Seata 方式。

以上就是用户中心和电商系统这两个章节的思考题答案，希望能带给你一些启发。接下来，老师还会针对剩余的课后思考题，以及你的提问来作出解答。有任何问题，还是跟以前一样，欢迎你在留言区多多互动。

和这次答疑加餐一起来的，还有一份暂停更新的说明。老师因为不可抗力无法备稿，专栏本周三、周五暂停更新两期，预计下周一恢复更新。你可以利用这个空档时间，复习之前的内容，我们下周再见！

分享给需要的人，Ta购买本课程，你将得 18 元

📄 生成海报并分享

👍 赞 1 💡 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。



技术领导力实战笔记 2022


从实操中提升你的领导力

TGO 鲲鹏会

数十位优秀管理者的真知灼见

肖军 / 苏宁金科 CTO
王璞 / DatenLord 联合创始人
郭炜 / 前易观数据 CTO
肖德时 / 前数人云 CTO
林晓峰 / GrowingIO 副总裁
于游 / 马泷医疗集团 CTO
王植萌 / 去哪儿网高级技术总监
胡广寰 / 酷家乐技术 VP
舒超 / 星汉未来 CTO



新版升级：点击「 请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。

精选留言 (1)

 写留言



TableBear 

2022-12-12 来自内蒙古

弱弱问一句，徐老师不会是 了吧

作者回复: 家父.....

共 3 条评论 >

