私塾在线《高级软件架构师实战培训阶段二》

——跟着cc学架构系列精品教程

纯内存操作——应对复杂业务逻辑

- 概述

纯内存操作是提升性能的一大法宝，根据操作情况的不同，大致分成：

1：基本只读：缓存

2：少部分写，大多时候读，基本不会并发：缓存；写的时候同时修改缓存数据

3：并发较高的混合读写：缓存；写的时候要控制同步、单线程写入，且整个应用中应该只有一个地方来实现写入，典型应用：库存的查询和修改

- 库存的查询和修改的基本业务描述

- 解决方案

1：单一操作的应用 + 单个缓存 + 同步写入数据库

2：单一操作的应用 + 多个缓存 + 同步写入数据库

3：多个单一操作的应用 + 多个缓存 + 同步/异步 写入数据库

4：引入预分配机制

高并发业务的处理-1

- 典型例子：秒杀下订单

  一些常见的要注意的基本点如下描述，当然是包括但不限于这些内容。

- 准备阶段

1：系统最好独立部署

2：做好系统性能容量规划（7折计算），系统优化，容灾过载保护

3：做好系统的拆分，比如：按功能模块、按实时/非实时、按动态/静态等等

4：设置定时上架的时间

5：服务器时钟同步

6：动态生成下单页面的URL

- 前端页面

1：静态页面 + Ajax来获取动态内容，比如：实时库存、活动状态、当前时间等

2：做好CDN部署

3：静态页面和资源的缓存

4：JS对请求的过滤，比如：获取检验码、时间到了或者已售完自动结束等

高并发业务的处理-2

- Web端

1：F5/Lvs + Nginx来接收高并发的请求，并做负载均衡

2：Nginx + Lua + Redis 来做请求队列，并实现一些基本控制，比如：限流、帐号参加次数检查、同一IP请求数检查等

3：Varnish来缓存静态页面和静态资源

4：进入Tomcat集群，先做一个预处理，就是判断这些账户是否能参与活动，比如：帐号等级是否足够、帐号行为是否正常、是否在黑名单上等

- 逻辑层

1：按照Redis的请求队列进行先后处理

2：纯内存操作 + 异步

3：控制超卖

4：Redis里面存放着SKU的库存数据

5：处理成功的信息也存放在Redis里面

高并发业务的处理-3

- 其它

1：合理设计接口
2：应当时告知用户结果
3：考虑业务规则，比如减库存的时机
4：服务器尽量集群，并做HA，避免雪崩
5：缓存服务器如果要重启，要做预热
6：对抗作弊，比如：同一帐户同时发多个请求、同一IP同时发大量请求、秒杀器采用多账户多IP发送请求等