奈学教育

Architecture

百万架构师训练营之核心技术实践篇（上）

主讲人：孙玄

创始人简介

奈学教育

奈学教育科技

创始人&CEO

转 转

首席架构师
技术委员会主席
大中台技术负责人

58集团

技术委员会主席
高级系统架构师

百度

资深研发工程师

毕业

浙江大学

擅长领域

架构设计、大数据
机器学习、技术管理等

对外分享

业界顶级大会
百万年薪架构
直播大课品牌创始人

目录

NX 奈学教育

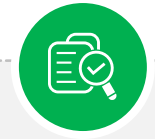
NX 奈学教育

01.互联网高可用设计

高可用对象

- 服务
- 架构

高可用是什么



7*24

任何人
任何时间
任何地点
任何方式
访问服务
正确结果

不高可用原因

单机硬件

硬件故障

生命周期

进程（Process）

潜在Bugs

性能极限

高可用评估方式

- 服务高可用传统的评估方式
  - 一段时间（比如一年）的停机时间占比
    - 停机时间/总时间
  - 2个9
    - 一年停机的时间不能超过88小时
  - 3个9
    - 一年停机的时间不能超过9小时
  - 4个9
    - 一年停机的时间不能超过53分钟
  - 5个9
    - 一年停机的时间不能超过6分钟
  - 1个9
    - ?
- 服务高可用科学的评估方式
  - 一段时间（比如一年）的停机影响请求量占比
    - 停机时间影响请求量/总的请求量

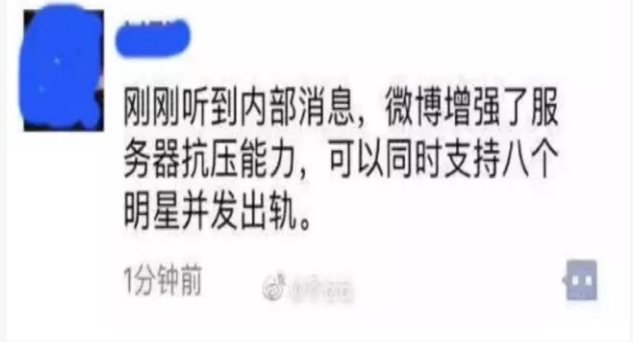
微博服务突发大流量

架构设计难点在哪里？

微服务架构如何应对突发流量？

微博服务突发大流量

为什么微博还是搞不定突发流量？



微服务高可用设计落地实践

架构高可用案例

- 单体架构高可用实践
- SOA架构高可用实践
- 水平分层架构高可用实践
- 微服务架构高可用实践
- 服务网格架构高可用实践
- 中台架构高可用实践
- 云原生架构高可用实践

互联网高可用设计

NX 奈学教育

高可用案例

- 网关层已具备热切换能力
  - 热开关切换
- 网关层不具备热切换能力
  - 防火墙限制只出不进
    - IPTABLES

架构高可用的本质到底是什么？

NX 奈学教育

02.服务无状态化设计与实践

服务无状态化本质

- **本质**
  - 同一服务（进程）（**比如网关层**）冗余部署N份，N份完全对等
  - 请求提交到任何一冗余服务上，处理结果完全一样

- **典型案例剖析（业务逻辑层）**
  - 电商分类信息存储设计
    - 一级类 / 二级类 / 三级类
      - 手机 / 华为 / Mate X

服务无状态化目的

- 快速扩容服务
- 弹性缩容服务

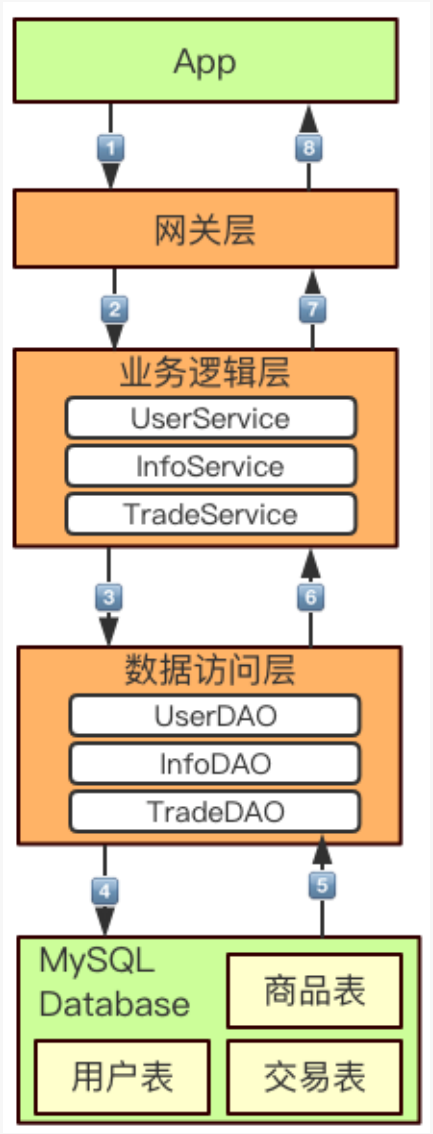
服务无状态化企业级案例剖析

- **用户Session案例**
  - 登录方式
    - 用户名+密码
    - 手机号+验证码
  - 登录成功
    - 生成用户凭证（session）
    - AES(UID+Timestmap+校验码)

- 用户Session数据存放在哪里?
- 用户Session数据如何存放?

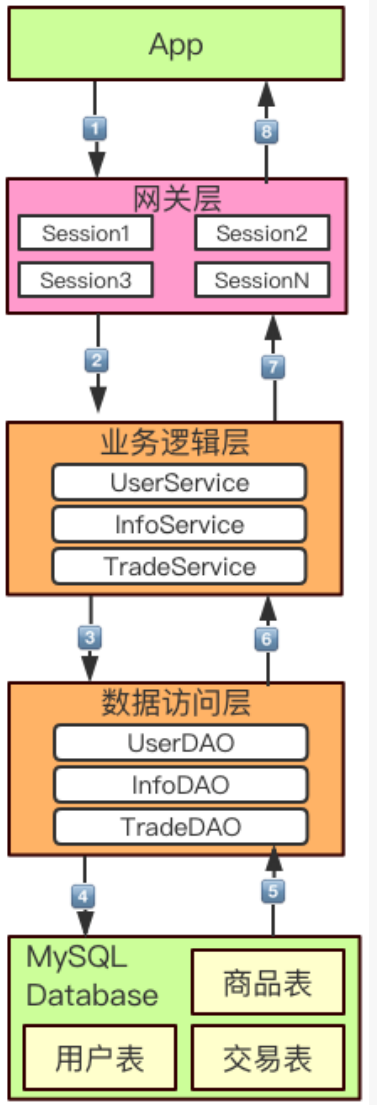
服务无状态化案例

- 用户Session数据存放哪里
  - 网关层
- 用户Session数据如何存放
  - 直接存放网关层
  - 存在外部存储



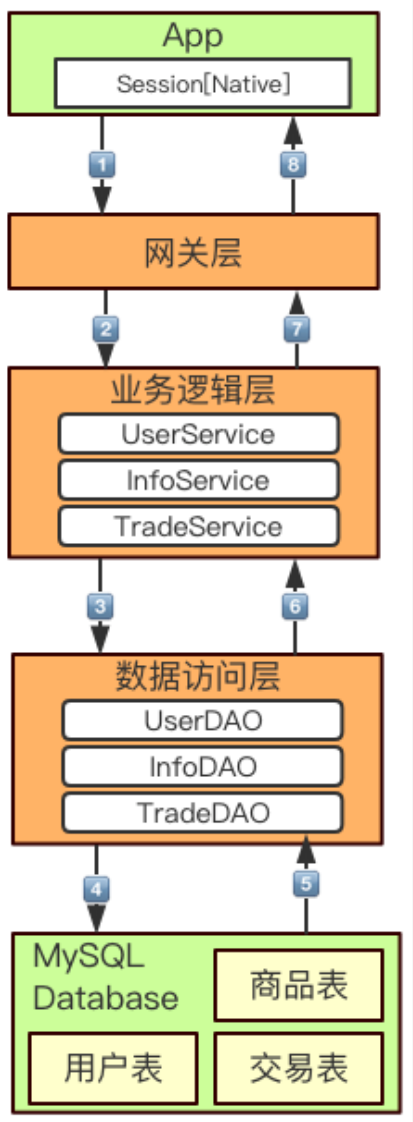
服务无状态化案例

- 用户Session数据直接存储网关层
  - 网关层有状态化



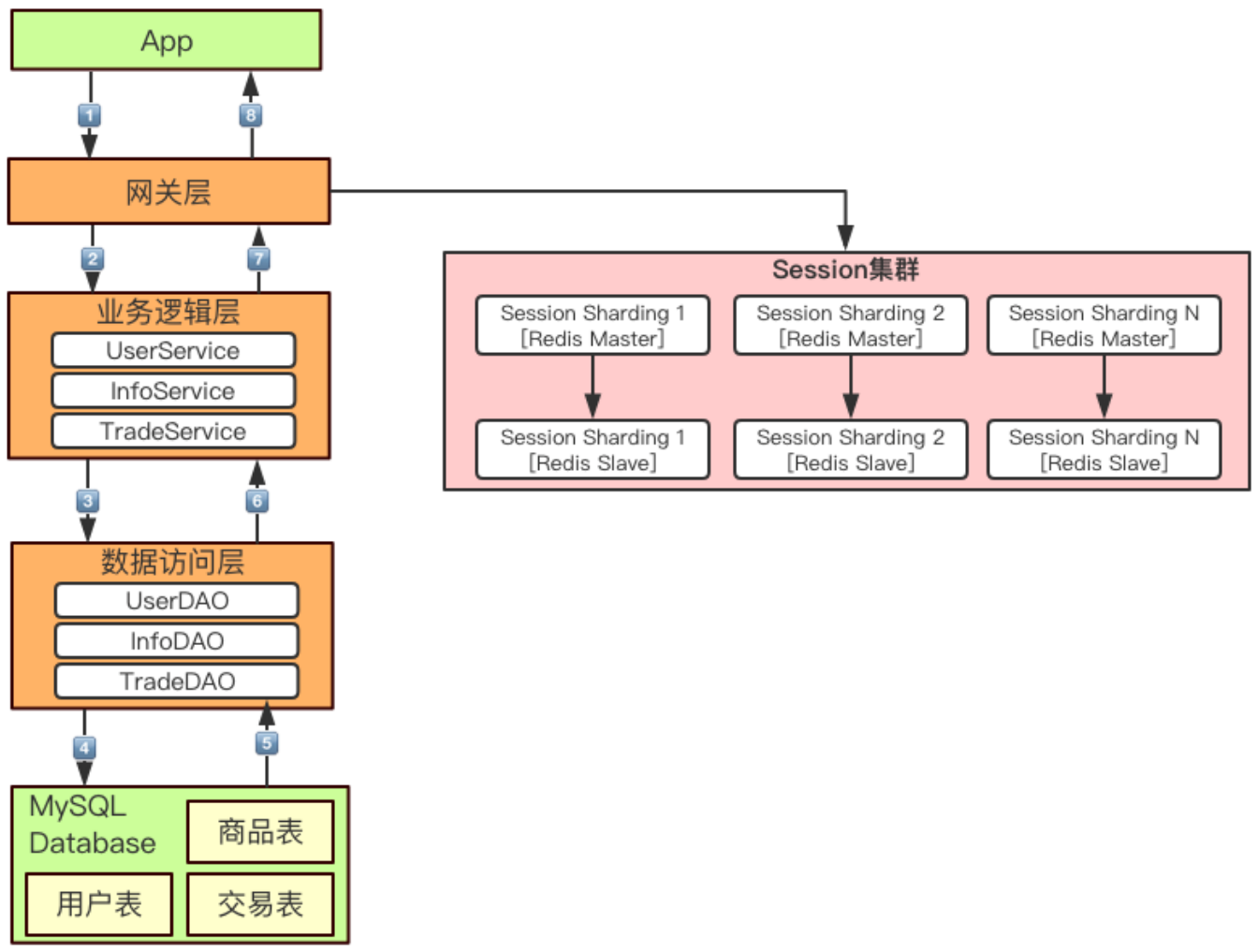
服务无状态化案例

- 用户Session数据存储客户端
  - 网关层无状态化
  - Session丢失问题
  - JWT(JSON Web Token)



服务无状态化案例

- 用户Session数据外部存储
  - 网关层无状态化
  - Session数据高可用



有状态化（Stateful）服务如何改造成无状态化（Stateless）服务？

分布式Session企业级落地代码案例深入剖析

NX 奈学教育

03.服务负载均衡设计与实践

负载均衡系统

- 硬件
  - F5
  - A10
  - Radware
- Nginx
  - 7层、4层
- HAProxy
  - 4层或7层
- 反向代理 VS 正向代理

狭义负载均衡

广义负载均衡

负载均衡算法

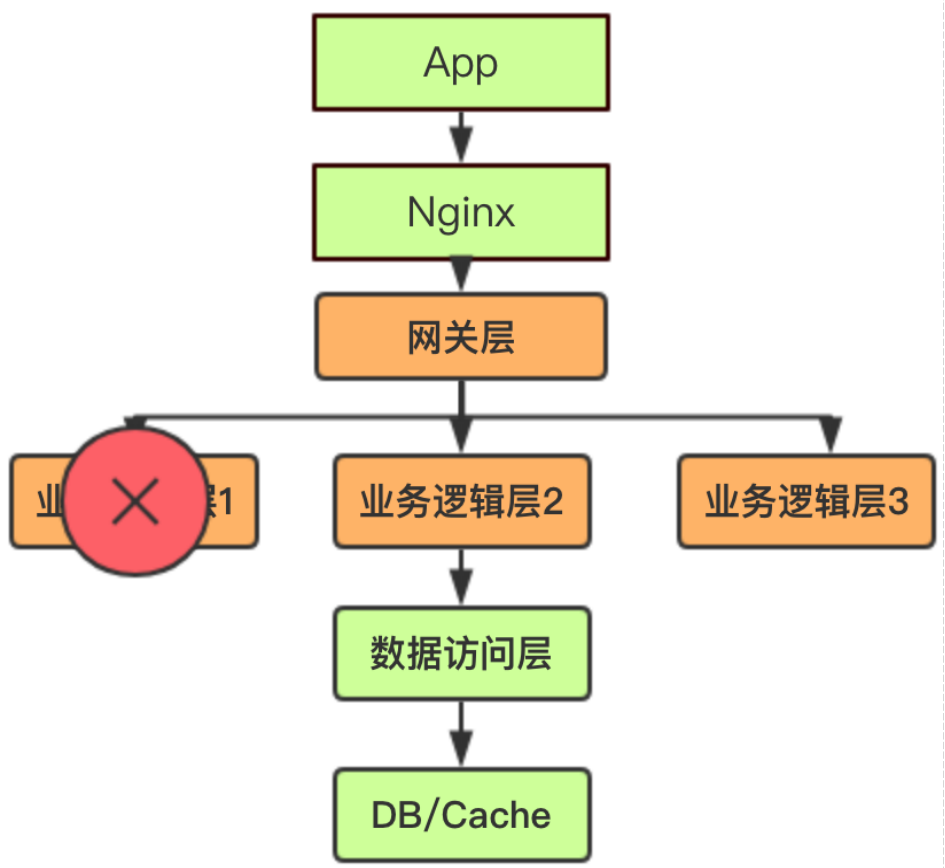
- Dubbo LoadBalance
  - Random
    - 随机，按权重设置随机概率
  - ConsistentHash
    - 一致性Hash，相同参数的请求总是发到同一提供者
  - ……

负载均衡系统和负载均衡算法区别联系？及应用场合（以微服务架构为例）？

广义负载均衡案例剖析

- 完整的故障处理恢复机制
  - 故障自动发现
  - 故障服务自动摘除
  - 请求自动重试
  - 服务恢复自动发现

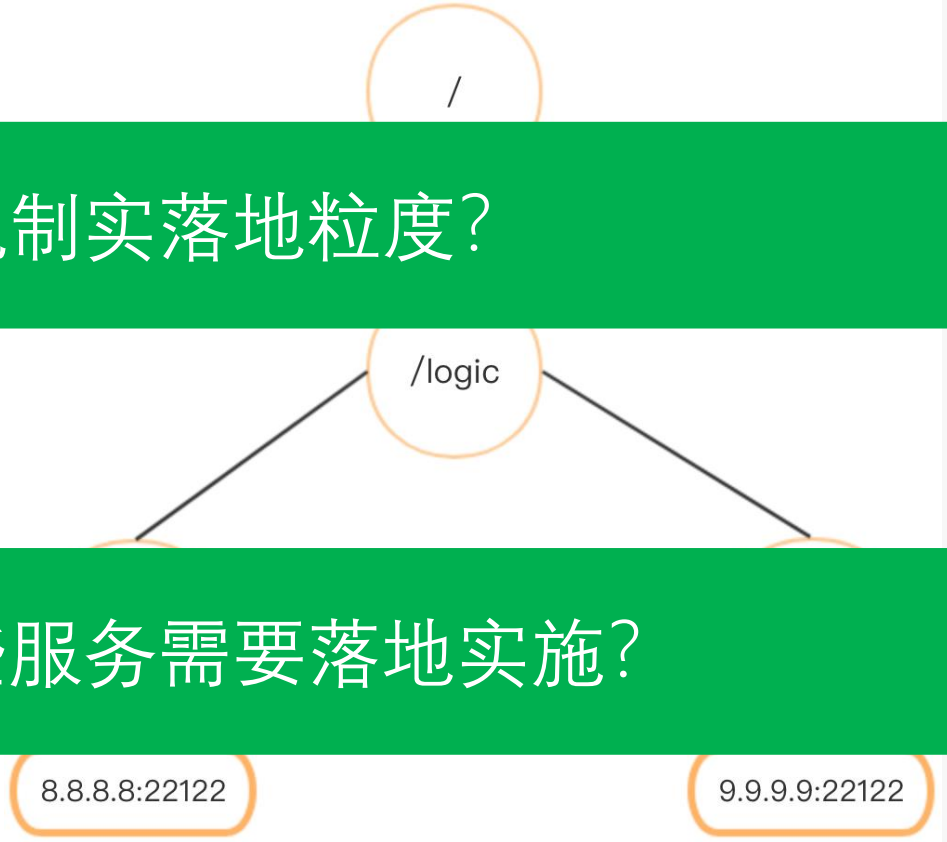
水平分层架构案例



水平分层架构案例

- 业务逻辑层1故障
  - 谁来发现
    - 网关层
    - ……
  - 如何发现
    - 所有问题
    - 机器类型
      - 物理机/虚拟机
        - Kill Process
      - 容器化
        - Kill Process
        - Kill Docker/Pod

ZK核心构成



微服务架构中，企业级熔断机制实落地粒度?

微服务架构中，广义负载均衡哪些服务需要落地实施?

NX 奈学教育

04.服务幂等设计与实践

业务场景驱动

- **业务场景一：用户转账**
  - 用户A转帐给用户B一百万RMB
    - 用户A转账超时
    - 用户A重试
- **业务场景二：用户下单**
  - 同一用户短时间内只能下单一次

业务场景共性

- **业务场景一：用户转账重试**
  - 保证转帐结果正确性
  - 请求幂等
- **业务场景二：同一用户短时间下单一次**
  - 保证下单结果的正确性
  - 业务幂等

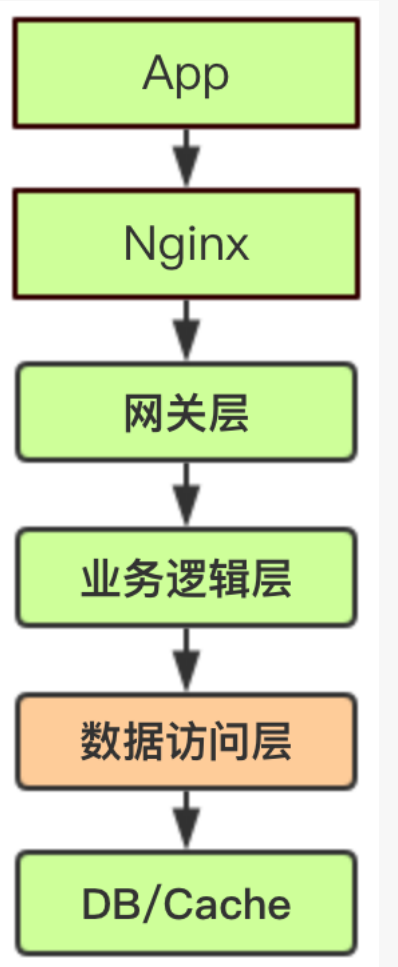
请求幂等本质

- **请求幂等**
  - 保证请求重复执行和执行一次结果相同
  - f…f(f(x)) = f(x)
    - x是参数
    - f是执行函数/ 方法

- **请求分类**
  - 读请求
  - 写请求
    - Insert、Update、Delete

请求幂等本质

- 请求幂等
  - 架构层面
    - 反向代理层
    - 网关层
    - 业务逻辑层
    - 数据访问层
    - DB层



请求幂等剖析

- CRUD
  - Create/Insert
    - Insert user values(uid,name,age,sex,ts);
      - Insert user values(58, '玄姐', 18, 男, 2020.04.01);
  - Read/Select
    - Select * from user where uid=58;
  - Update
    - Update user set age=18 where uid=58;
    - Update user set age++ where uid=58;
  - Delete
    - Delete user where uid=58;
    - Delete user where uid in bottom 10;

请求幂等剖析

- CRUD
  - CRD
  - Update
    - 案例一：年龄增加1岁
      - 增加where条件
        - where age=18
        - 相对修改转换成绝对修改
          - set age=19
    - 案例二：🖥电商平台购买商品
      - 商品价格¥10000
      - 确认收货
        - 订单状态
        - 打款
        - 状态修改
      - 分布式事务

请求幂等案例

- **电商下订单企业级案例剖析**
  - Insert请求
  - OrderID如何产生

请求幂等案例

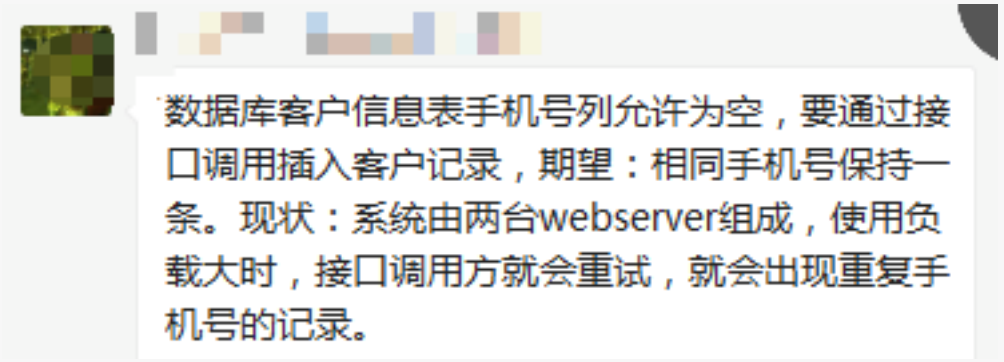
- **电商下订单企业级案例剖析**
  - Update请求
  - 订单状态修改如何做

业务幂等本质

- 同一用户短时间允许下单一次
  - 用户是共享资源
  - 保证用户操作串行化
    - 锁的问题
    - 进程冗余部署多份
      - 分布式锁问题

业务幂等本质

- 企业级案例剖析



分布式服务幂等企业级落地案例深入剖析

要点回顾

NX 奈学教育

01 高可用设计

02 服务无状态化设计

03 服务广义负载均衡设计

04 服务幂等设计

05 分布式锁设计

06 分布式事务设计

07 服务降级设计

08 服务限流 / 熔断设计

09 服务灰度发布设计

10 服务全链路压测设计

11 高并发设计

NX 奈学教育

欢迎关注本人公众号
“架构之美”