【大数据高并发1 负载均衡】

平均轮询

加权轮询

ip-hash

根据ip散列分配

解决用户持久化问题

会话粘滞

但会带来资源分配不均衡的问题

某个服务器挂了，一批ip就没法响应了，没法做到高可用

fair

根据承载量、响应速度进行动态分配

url-hash

用户持久化

会话粘滞 ip-hash具有这个效果

cookie

客户端

有风险

增加带宽

session共享

StateServer

SqlServer

Redis（首选）

token

JWT

与cookie差不多，

【jwt vs cookie】

【大数据高并发2 分库分表】

分区：数据库本身提供的功能，为了解决单表数据量太大

垂直分库：电商平台分成了存储库、订单库、用户库

水平分库：华南 华北 华东，或者不同时间段；每个库结构一致，数据不一样

各种拆分方式的特点、缺陷

跨库查询怎么办？

1.数据拷贝，总数据报表，为报表服务

2.即时查询，

垂直分：service

水平分：

统计数据式，比如分页，在主库设置统计表（博客园）

业务妥协：钱能解决的问题，就用钱解决

钱解决不了的，用技术解决

技术解决不了，用业务解决

分表：

垂直分

不同字段查询、修改不一样

文章表：标题、作者、时间，查询多

正文不多

通过外键、主从表的方式关联

水平分

时间、地区、类别

【大数据高并发3 全文检索、搜索引擎、Redis秒杀rank】

Redis，Remote Dictionary Server

单线程：没有线程安全问题/原子性

redis单线程可达10w QPS，Query Per Seconds

不够再多线程

防止超卖：

100件商品，10000个并发请求

还有下单成功，取消支付的

关系型数据库，锁+事务，数据库奔溃

Redis，初始化100个库存，单线程

decr，减1返回结果，结果小于0停止秒杀

只有100个请求到数据库。

取消订单，incr，加1

缓存：

数据统计：

刷礼物，zSet，有序列表，一个key对一组数据

【大数据高并发4 业务垂直切分】

队列

效果好

但成本高，有延迟

队列可以提升响应能力，削峰

高可用，集群，固化

可扩展，业务和UI分离

分布式，成本巨大

CAP