极客时间 训练营

架构实战营 - 模块4

第3课：存储架构模式 - 分片架构和分区架构

李运华

前阿里资深技术专家(P9)

教学目标

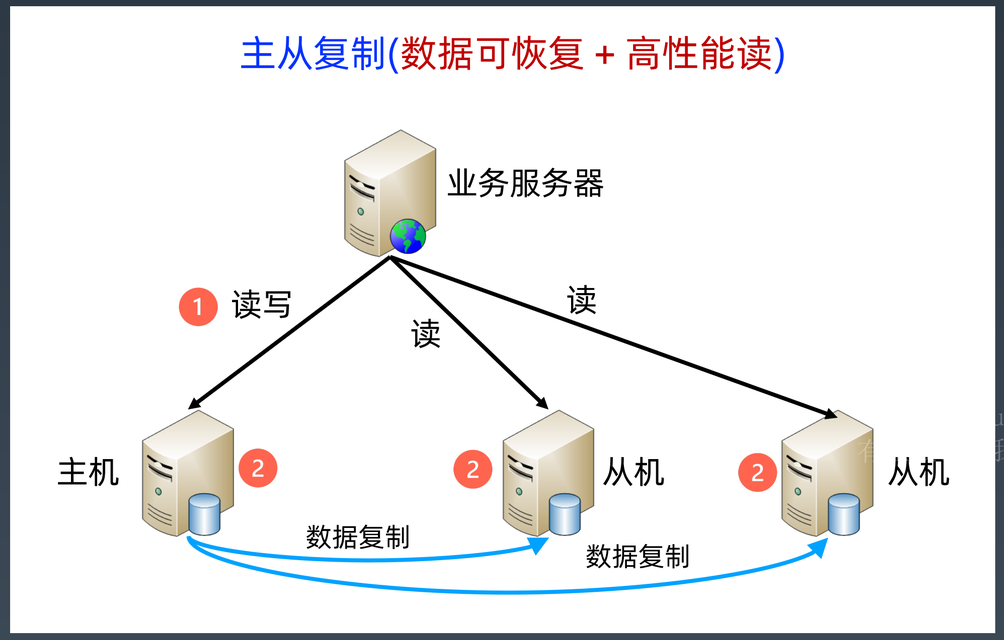
1. 掌握分片架构的设计和本质
2. 掌握分区架构的设计和本质

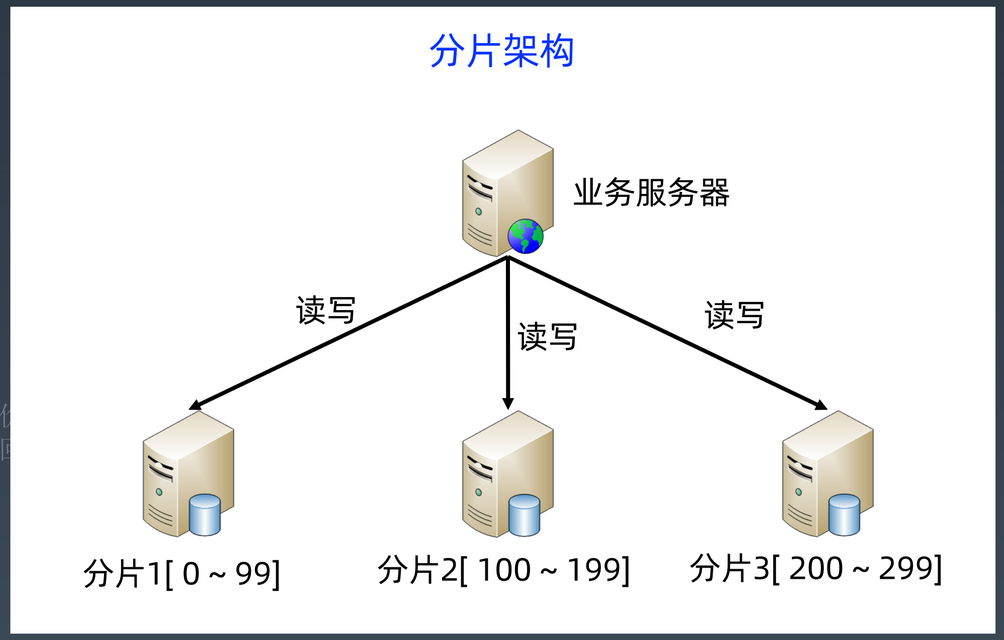
量变引起质变！

目录

分片架构

分片架构的本质



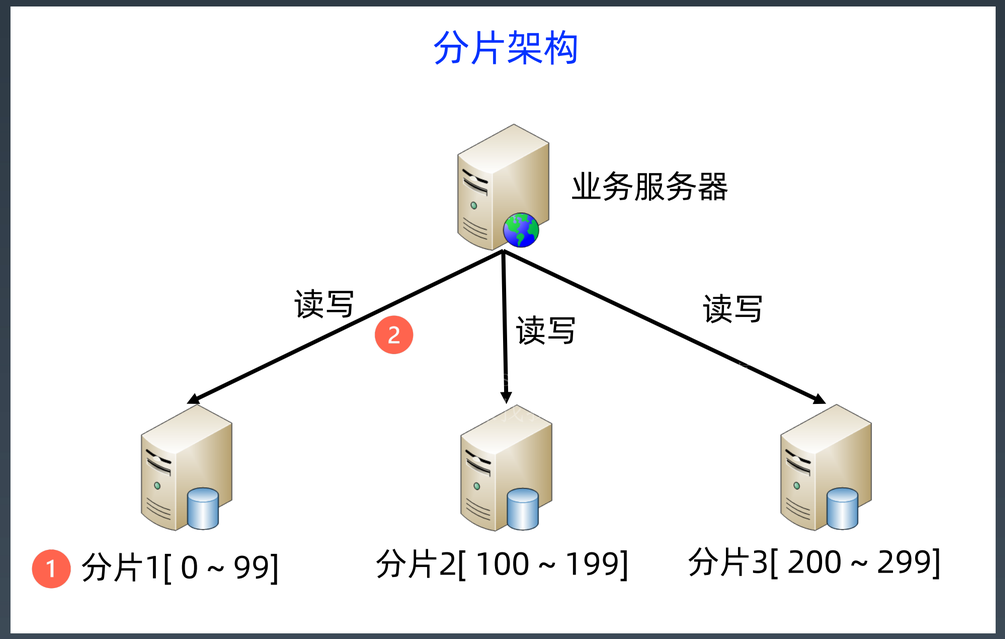


1. 只有主机承担写，写性能会存在瓶颈；
2. 每台机器保存全量数据，存储存在瓶颈

分片架构本质
通过叠加更多服务器来提升写性能和存储性能

主备架构的本质和主从架构的本质分别是什么？

分片架构设计核心



1.分片规则：数据按照什么规则分片
2.路由规则：业务服务器如何找到数据

分片架构设计核心 - 分片规则

极客时间 训练营

核心原则

选取基数比较大的某个数据键值，让数据均匀分布，避免热点分片

【基数 Cardinality】
被选的数据维度取值范围

【均匀】
数据在取值范围内是均匀分布的

分片数据

【主键】
适合主业务数据，例如数据库分片常用的用户 ID，订单 ID，Redis 分片的 key，MongoDB 的文档 ID

【时间】
适合流水型业务，例如创建日期，IoT 事件，动态

分片规则

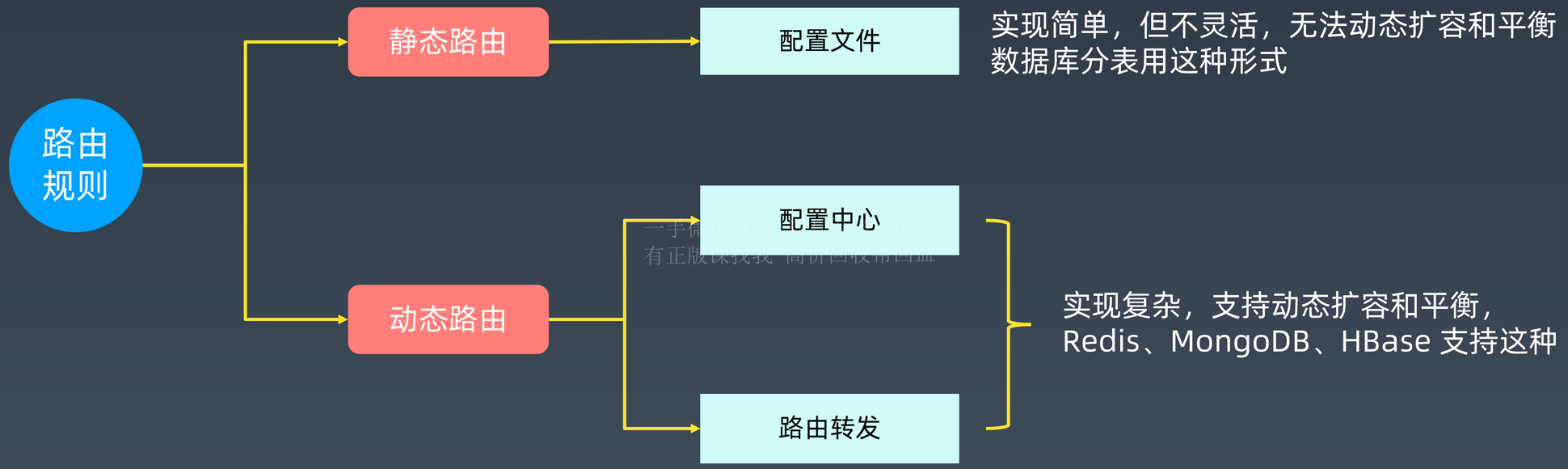
【Hash分片】
sharding key = hash（原始键值）分布均匀，不支持范围查询

【范围分片】
分布可能不均匀，支持范围查询

案例：
1. 某互联网业务按照用户年龄分片，每10岁一个分片，这样分片合理么？如果按照城市分片呢？
2. 微信朋友圈的动态适合用什么做分片？微博适合用 Hash 分片还是范围分片？

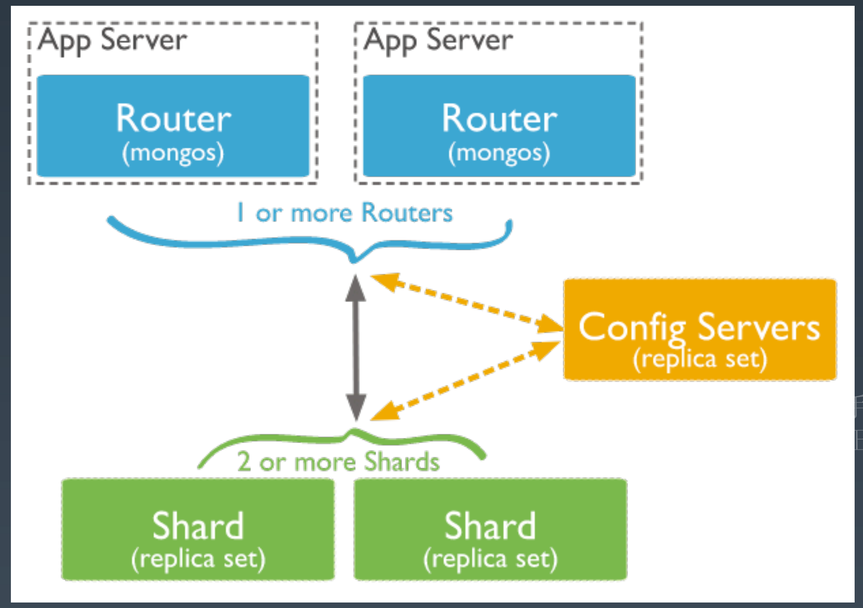
注意：数据均匀分布并不意味着读写请求均匀分布，例如微博

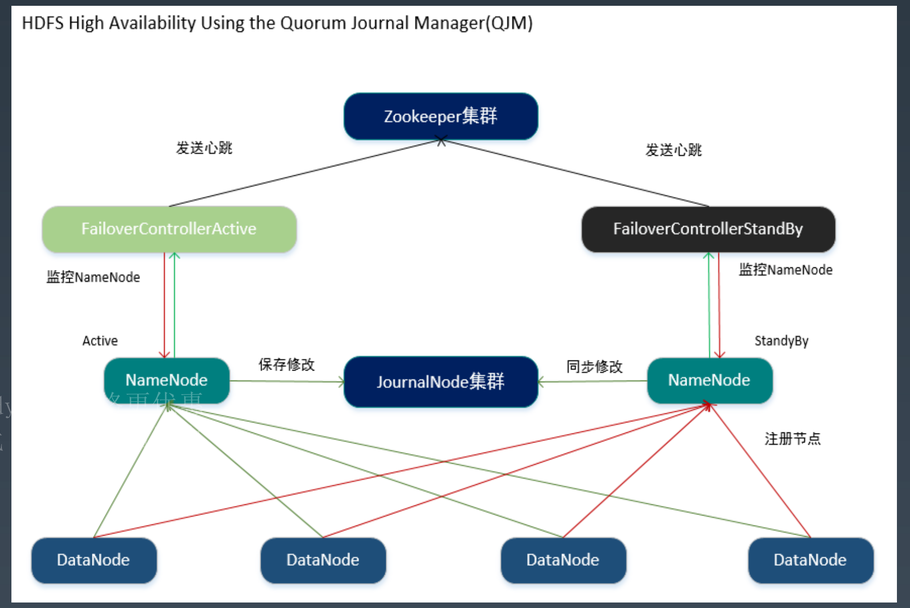
分片架构设计核心 - 路由规则



为什么数据库分表不实现动态路由？

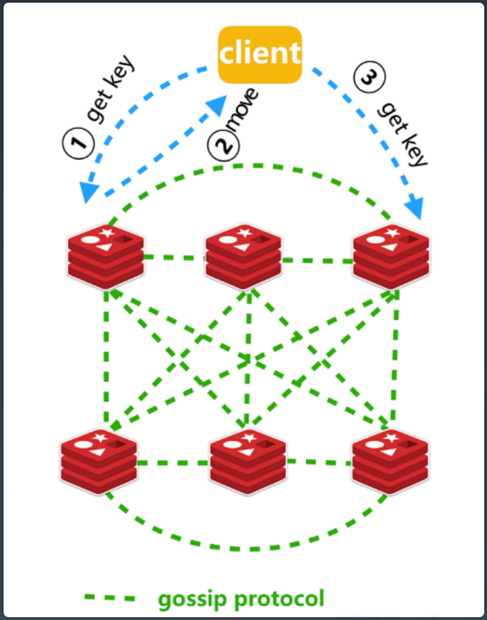
分片动态路由 - 配置中心



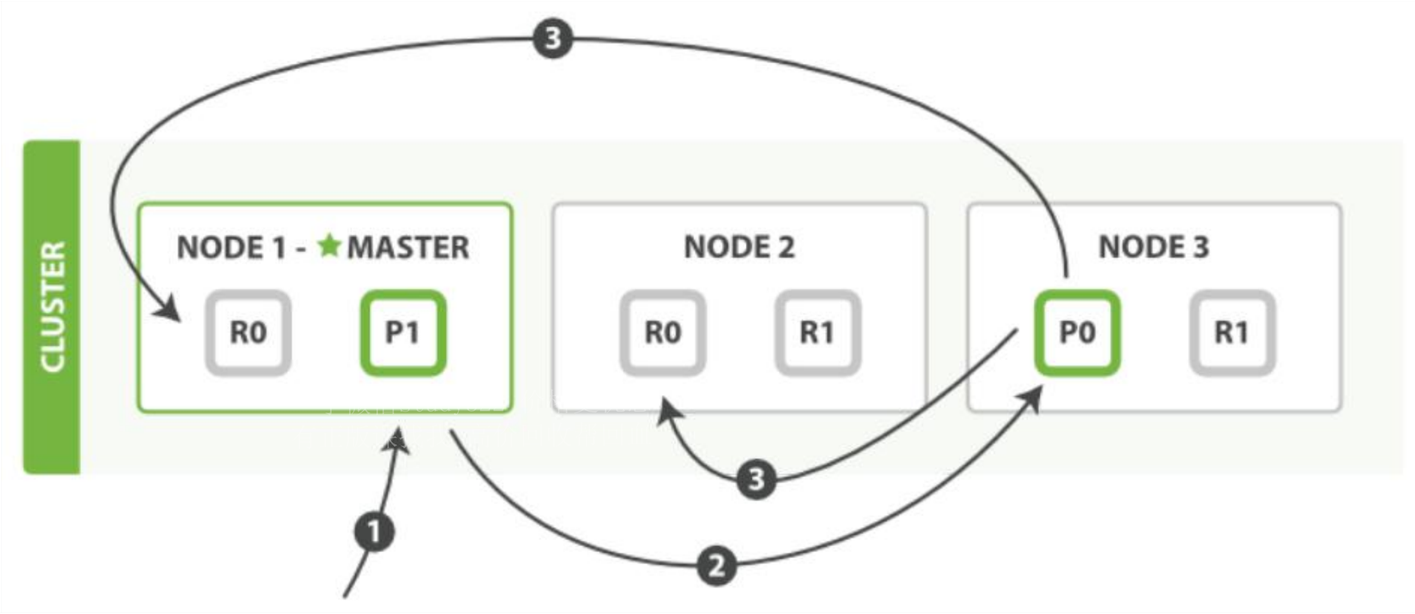


1. 由专属的配置中心记录分片信息，客户端需要向配置中心查询分片信息，然后发起读写操作。
2. 可以支持超大规模集群，节点数量可以达到几百上千
3. 架构复杂，一般要求独立的配置中心节点，配置中心本身又需要高可用，例如 MongoDB 用的是 replica set，HDFS 用的是 ZooKeeper（注意：HDFS 2.0版本以前的 Namenode 是单点）

分片动态路由 - 路由转发



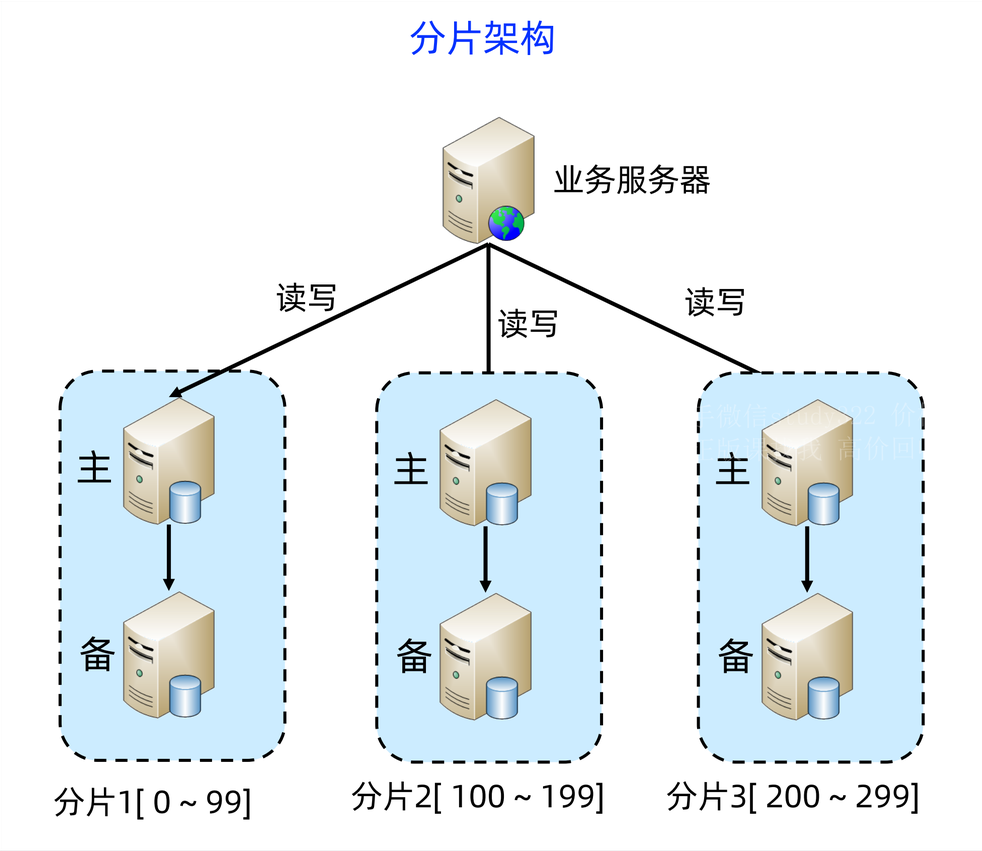
客户端重定向



服务端请求转发

1. 每个节点都保存所有路由信息，客户端请求任意节点皆可
2. 架构相对简单一些，一般通过 gossip 协议来实现分片信息更新
3. 无法支持超大规模集群，集群数量建议100以内（为什么？）

分片架构高可用方案1 - 独立备份

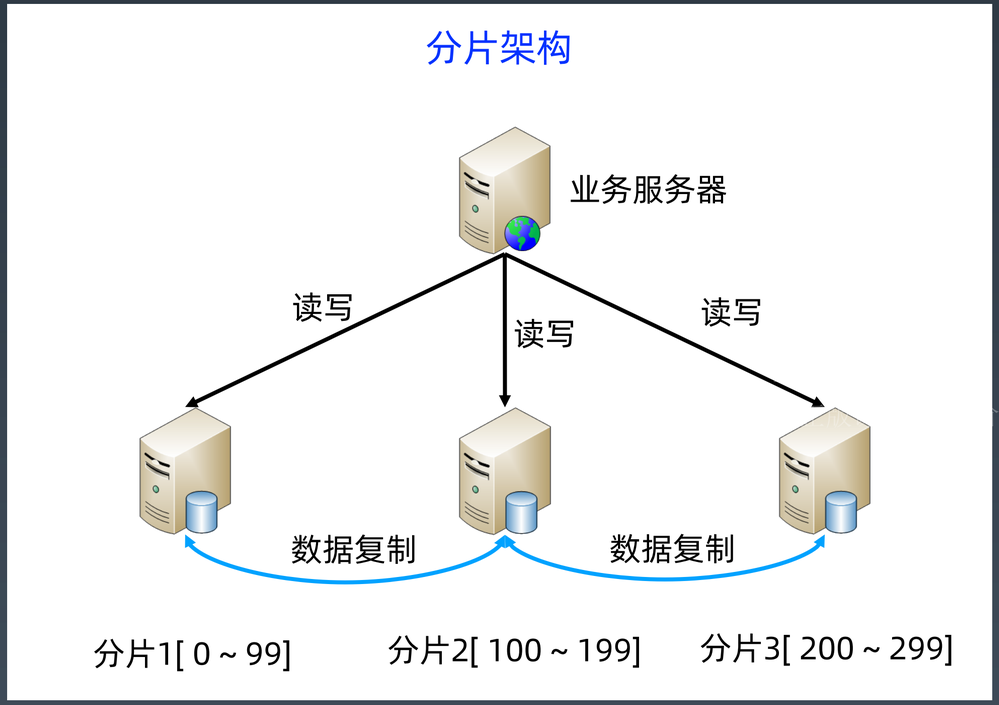


【原理】
每个分片有独立的备份节点，可以用主备、主从、集群选举等方式实现。

【优缺点】
1. 实现简单
2. 机器硬件成本比较高

【应用】
存储系统已经支持节点级别的复制

分片架构高可用方案2 - 互相备份

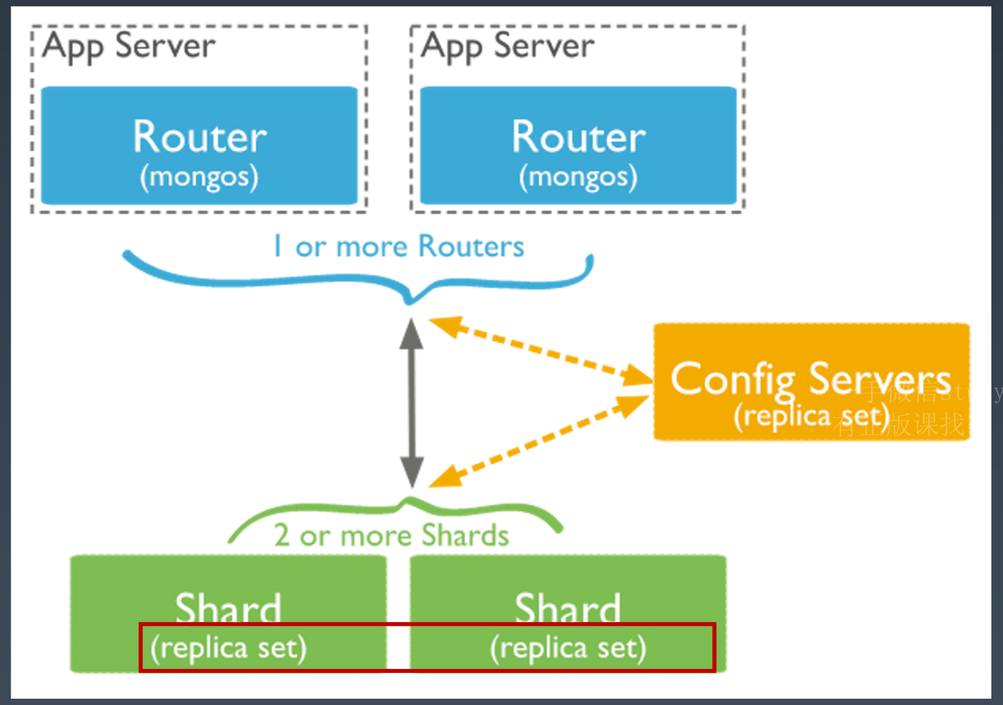


【原理】
分片之间的节点互相备份

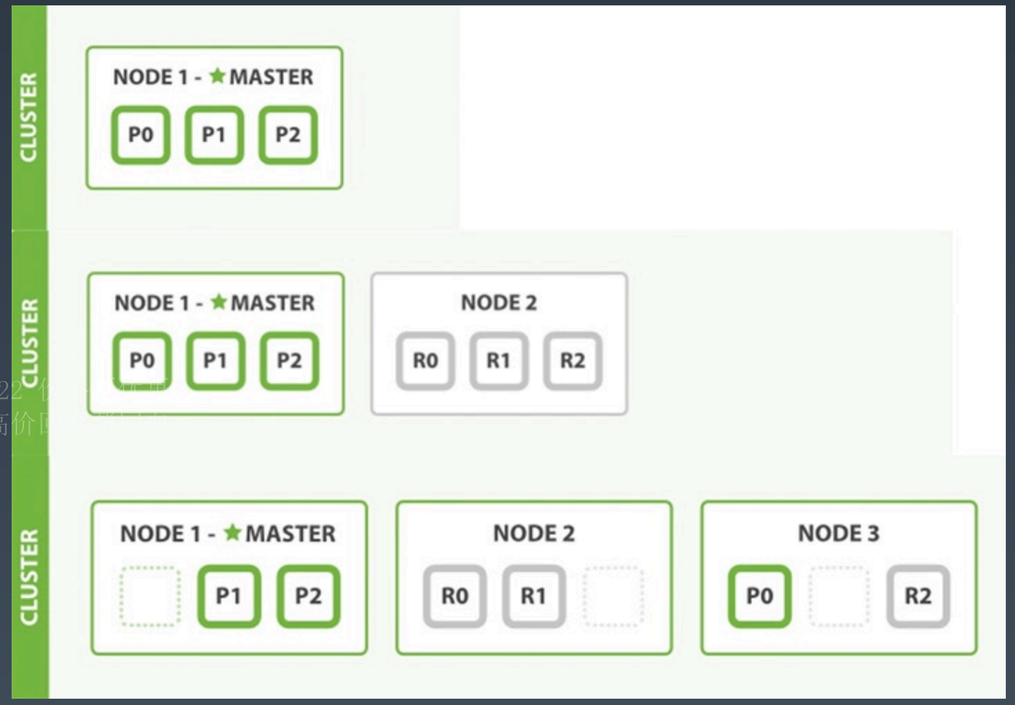
【优缺点】
1. 实现复杂
2. 机器硬件成本相对来说低，互相利用

【应用】
存储系统支持数据块级别的复制

分片架构高可用架构 案例



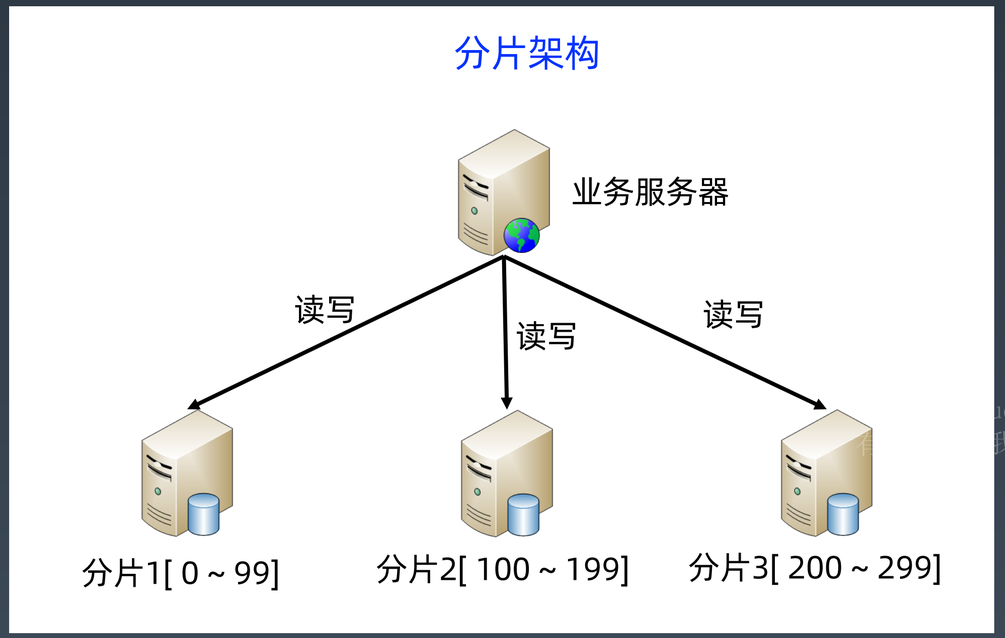
类似的有 MongoDB，Redis，MySQL

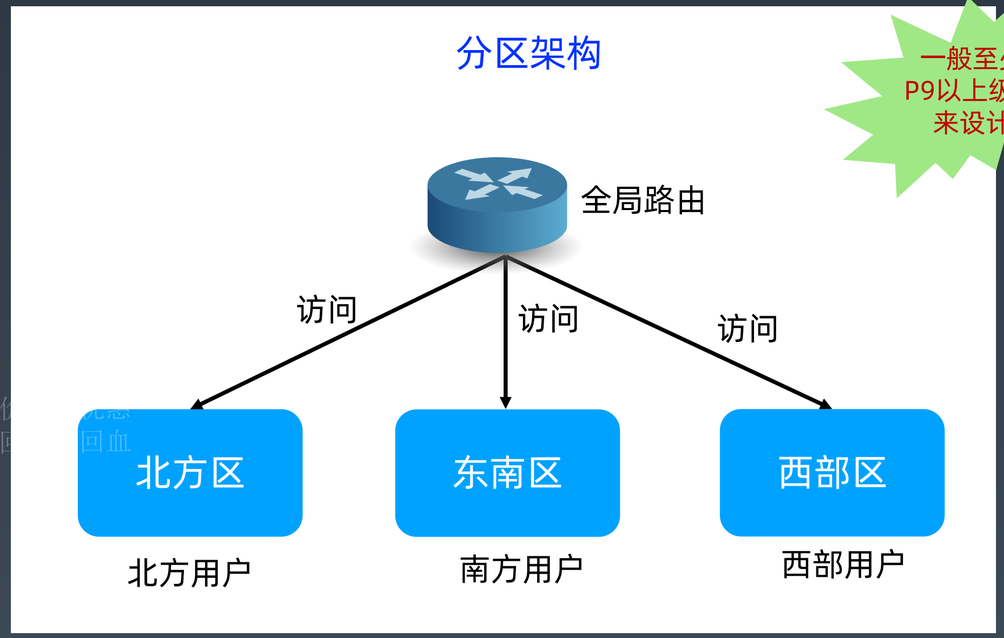


类似的有 HDFS、Elasticsearch

分区架构

分区架构的本质





分片架构缺陷

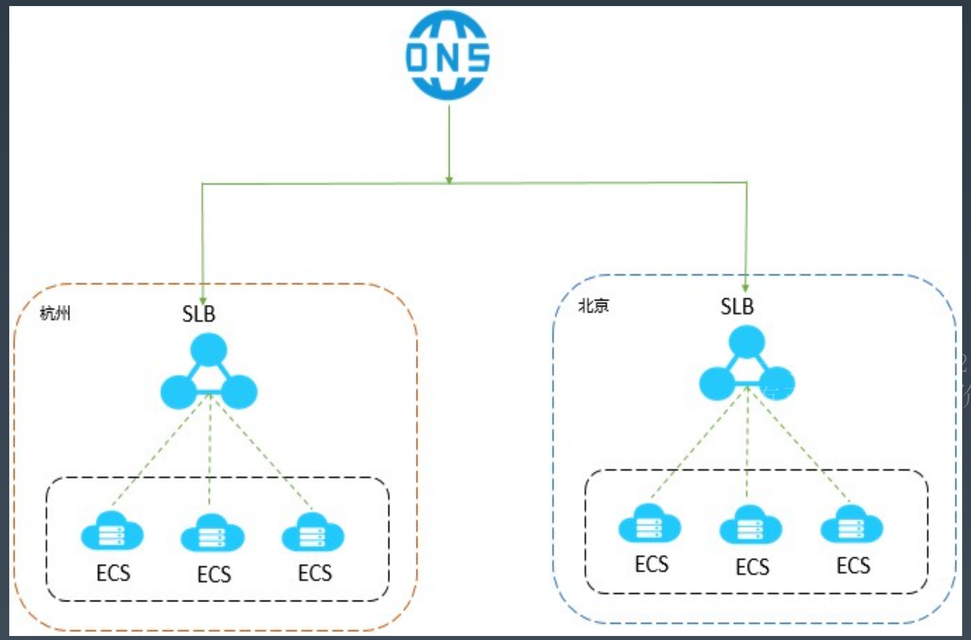
无法应对城市级别的故障

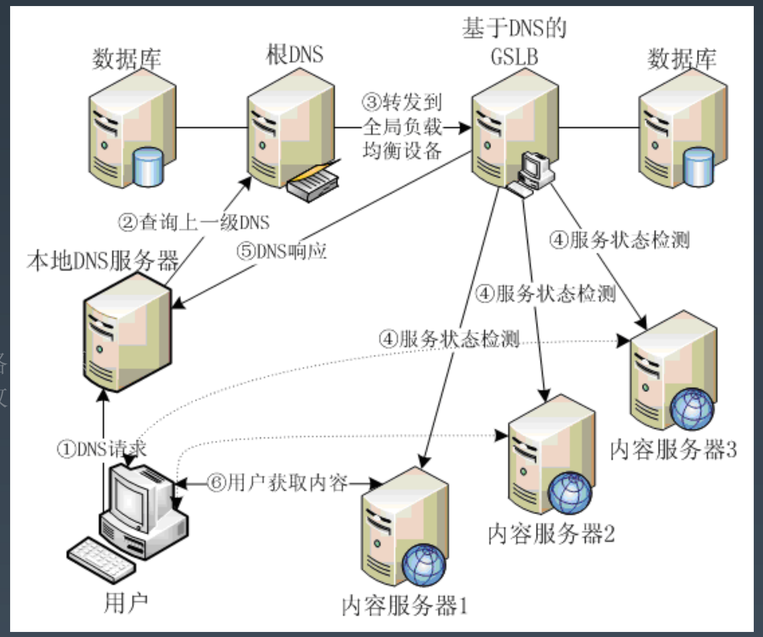
分区架构本质

通过冗余 IDC 来避免城市级别的灾难，并提供就近访问

分片架构可以跨城市部署么？

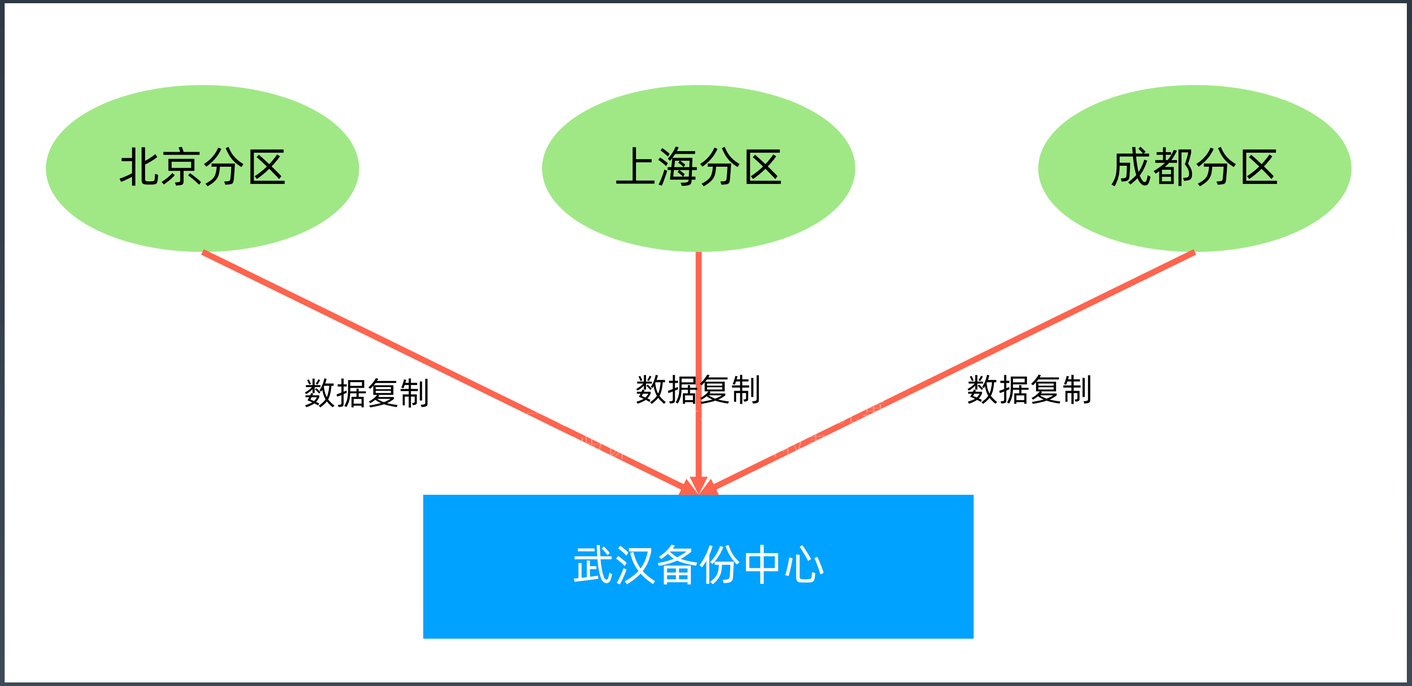
分区架构全局路由 - DNS 和 GSLB





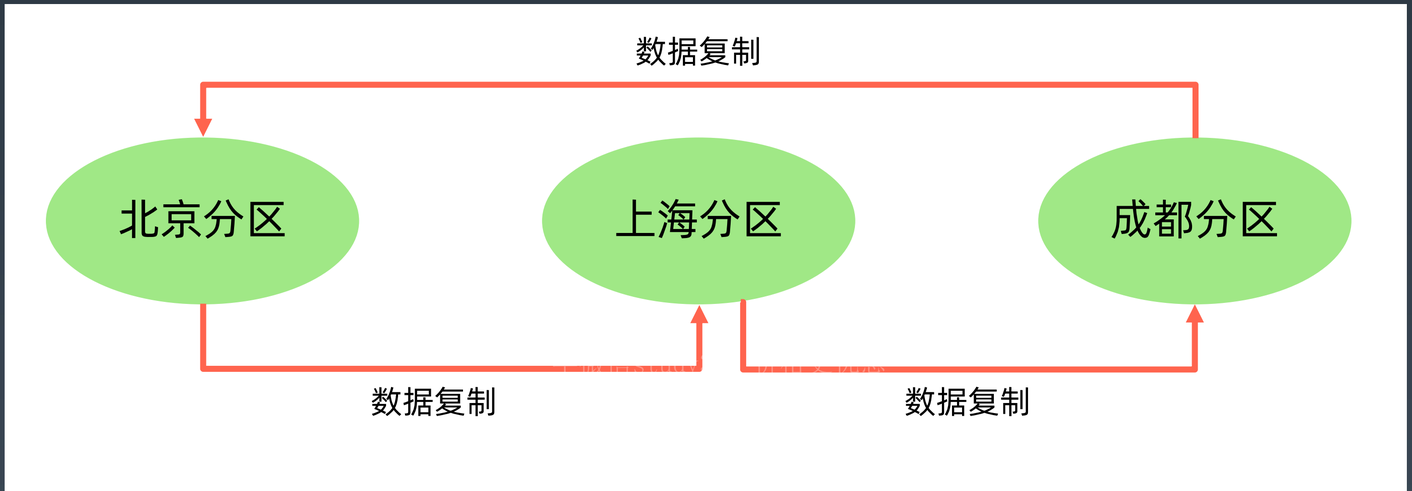
DNS：标准协议，通用，但基本只能实现就近接入的路由
GSLB：非标准，需要独立开发部署，功能非常强大，可以做状态监测、基于业务规则的定制路由

分区架构备份策略 - 集中式



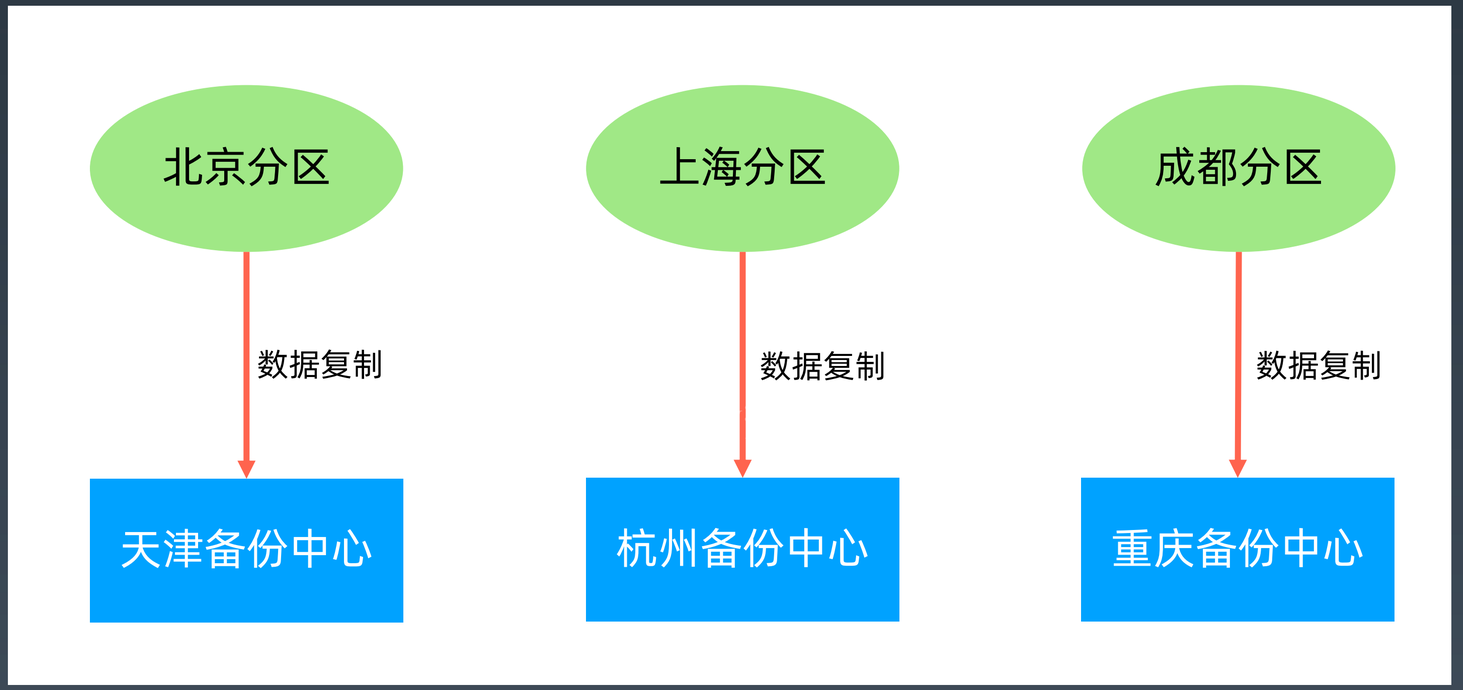
1. 设计简单，各分区之间并无直接联系，可以做到互不影响。
2. 扩展容易，如果要增加第四个分区（例如，西安分区），只需要将西安分区的数据复制武汉备份中心即可，其他分区不受影响。
3. 成本较高，需要建设一个独立的备份中心。

分区架构备份策略 - 互备式



1. 设计比较复杂，各个分区除了要承担业务数据存储，还需要承担备份功能，相互之间互相关联和影响。
2. 扩展麻烦，例如增加一个武汉分区。
3. 成本低，直接利用已有机房和网络

分区架构备份策略 - 独立式



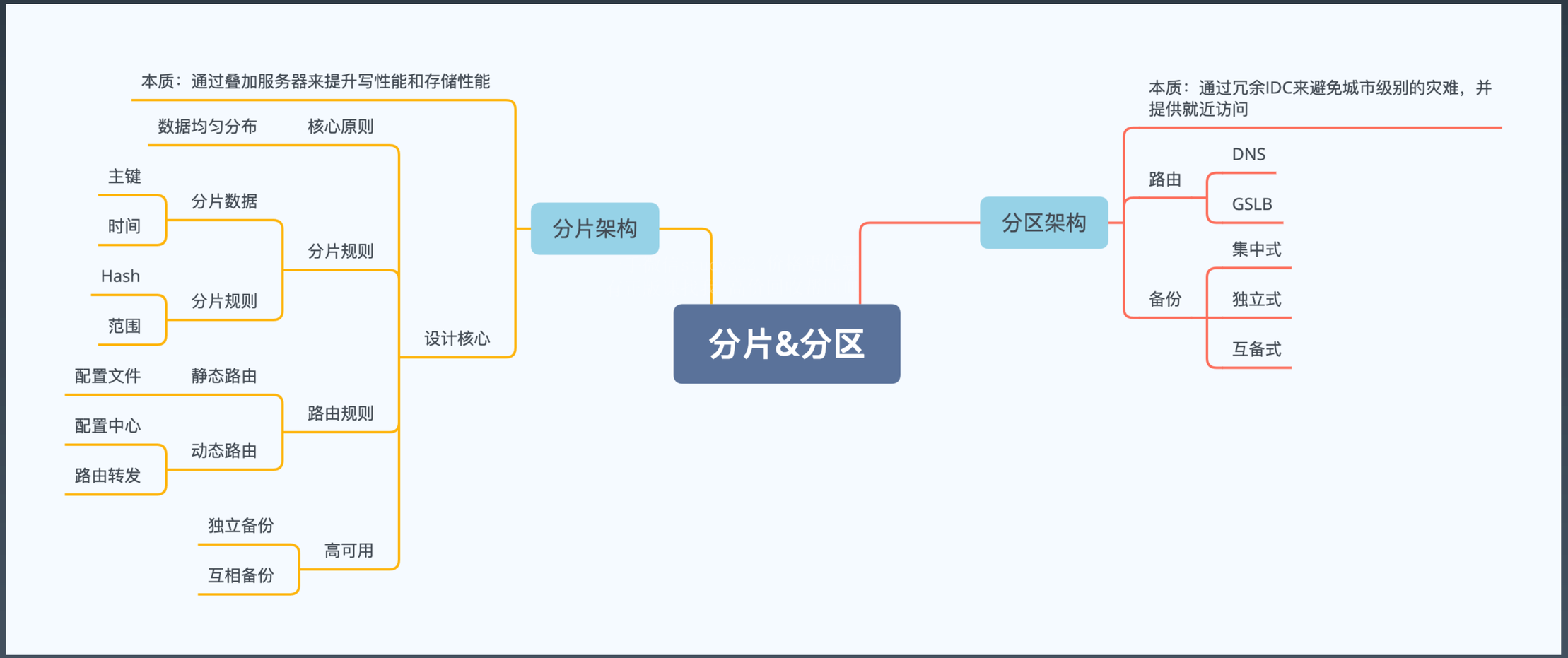
1. 设计简单，各分区互不影响。
2. 扩展容易，新增加的分区只需要搭建自己的备份中心即可。
3. 成本高，每个分区需要独立的备份中心，备份中心的场地成本是主要成本

分区架构备份策略对比

	集中式	互备式	独立式
成本	中	低	高
可扩展	高	低	高
复杂度	低	高	低

你认为哪种策略应用会多一些？

本节思维导图


分片&分区
分片架构
本质：通过叠加服务器来提升写性能和存储性能
设计核心
核心原则
数据均匀分布
分片规则
分片数据
主键
时间
分片规则
Hash
范围
路由规则
静态路由
配置文件
动态路由
配置中心
路由转发
高可用
独立备份
互相备份
分区架构
本质：通过冗余IDC来避免城市级别的灾难，并提供就近访问
路由
DNS
GSLB
备份
集中式
独立式
互备式

随堂测验

【判断题】

1. 主从复制无法解决主机写性能问题，因此要采用分片架构来提升写能力
2. 分片架构做到数据均匀分布后，读写就能够做到负载均衡
3. 分片架构需要结合复制架构才能具备高性能高可用特性
4. 分片架构如果跨城市部署，就相当于分区架构了
5. 分区架构的备份策略，成本和可扩展都是重要的考虑因素

【思考题】

既然数据集群就可以做到不同节点之间复制数据，为何不搭建一个远距离分布的集群来应对地理位置级别的故障呢？

Q&A

茶歇时间

八卦，趣闻，内幕……

THANKS