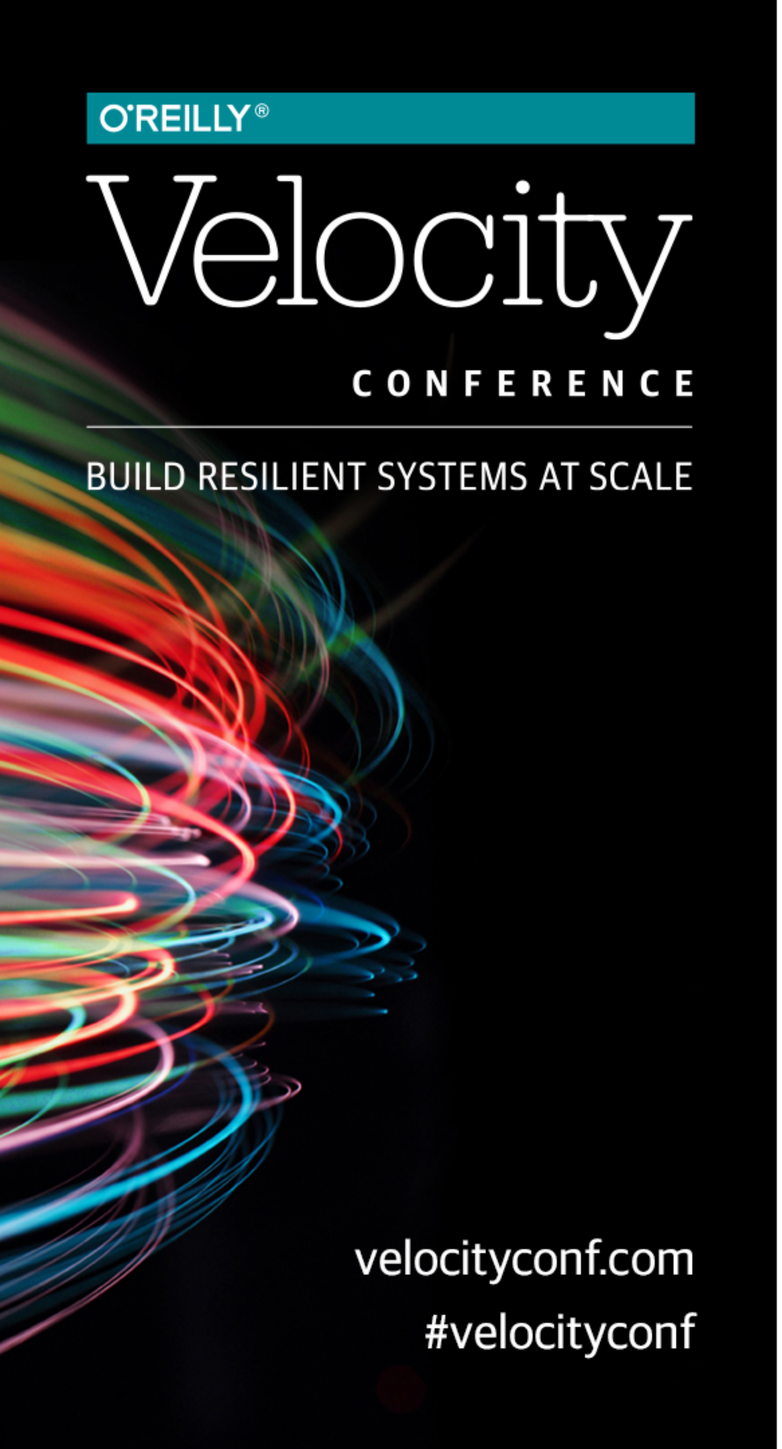



# 滴滴弹性在线存储平台

周充
zhouchong@didichuxing.com

# WHY ?

- 快速的业务发展，多变的应用层需求，需要高性能、高可用、大容量、易于使用的NoSQL存储解决方案

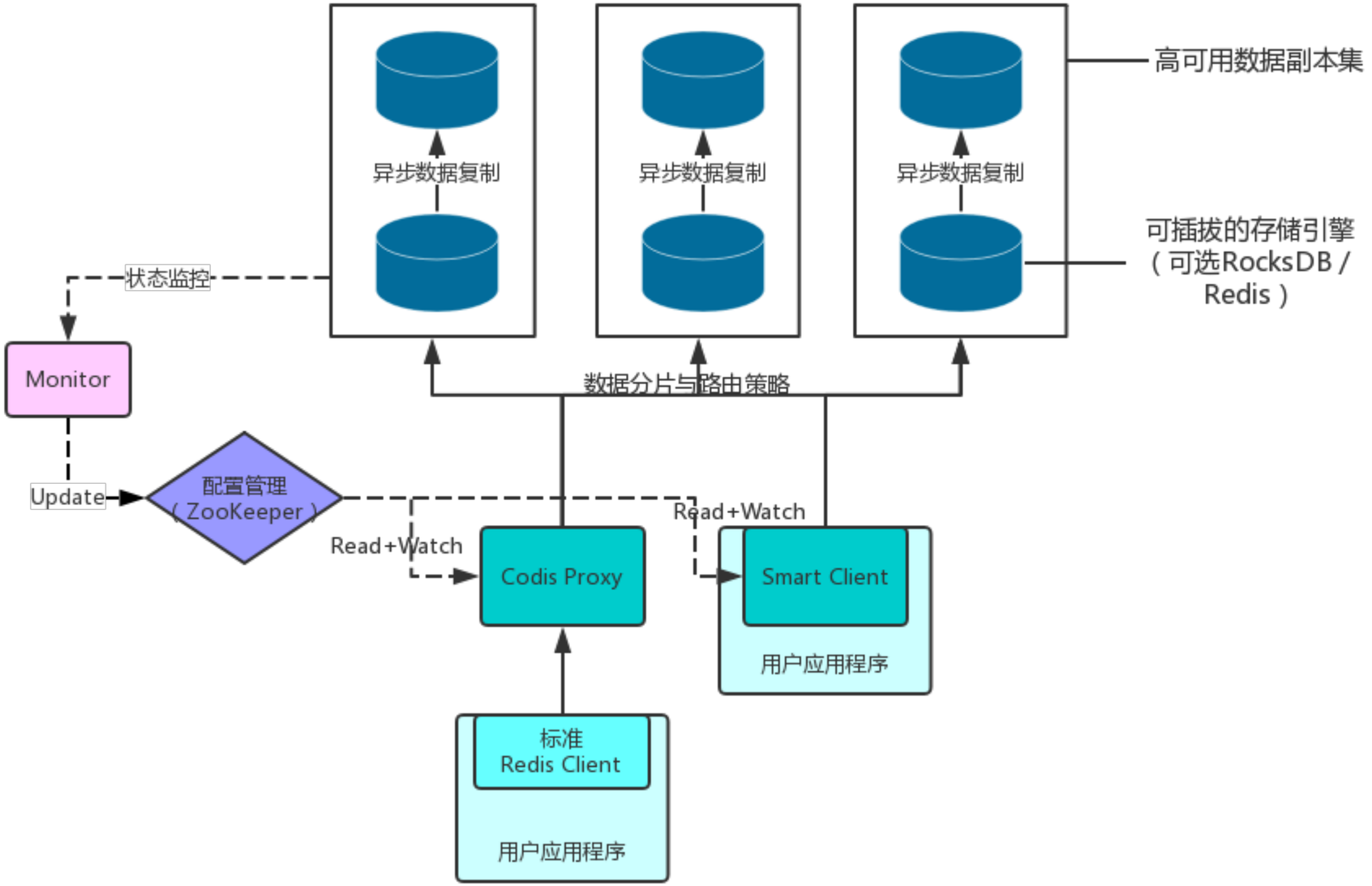
- 业界现有的方案无法满足应用的需求
  - Redis Cluster，Codis .....
  - Hbase，Cassandra
  - MongoDB

- 高响应度的业务需求定制开发

# WHAT ?

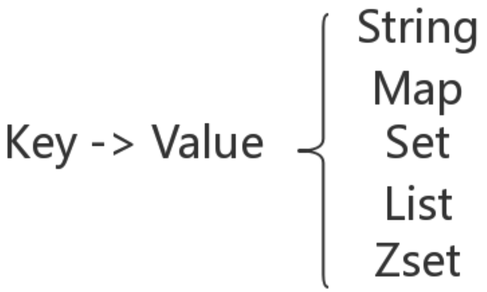
- NoSQL存储平台：支持在线业务与大数据场景
- 高性能：
  - 时延：内存 ~1ms； 磁盘（SSD） < 5ms
  - 吞吐量：内存单机100w QPS；磁盘（SSD）单机20w QPS
- 大容量：保证时延与吞吐量前提下单机最高存储T级别数据
- 高可用： 故障秒级恢复
- 易伸缩
  - 负载与容量线性扩容
  - 动态扩容，对业务透明

# 系统架构


异步数据复制
异步数据复制
异步数据复制
高可用数据副本集
可插拔的存储引擎
（可选RocksDB / Redis）
状态监控
Monitor
数据分片与路由策略
Update
配置管理
（ZooKeeper）
Read+Watch
Read+Watch
Codis Proxy
Smart Client
用户应用程序
标准
Redis Client
用户应用程序

# 数据模型

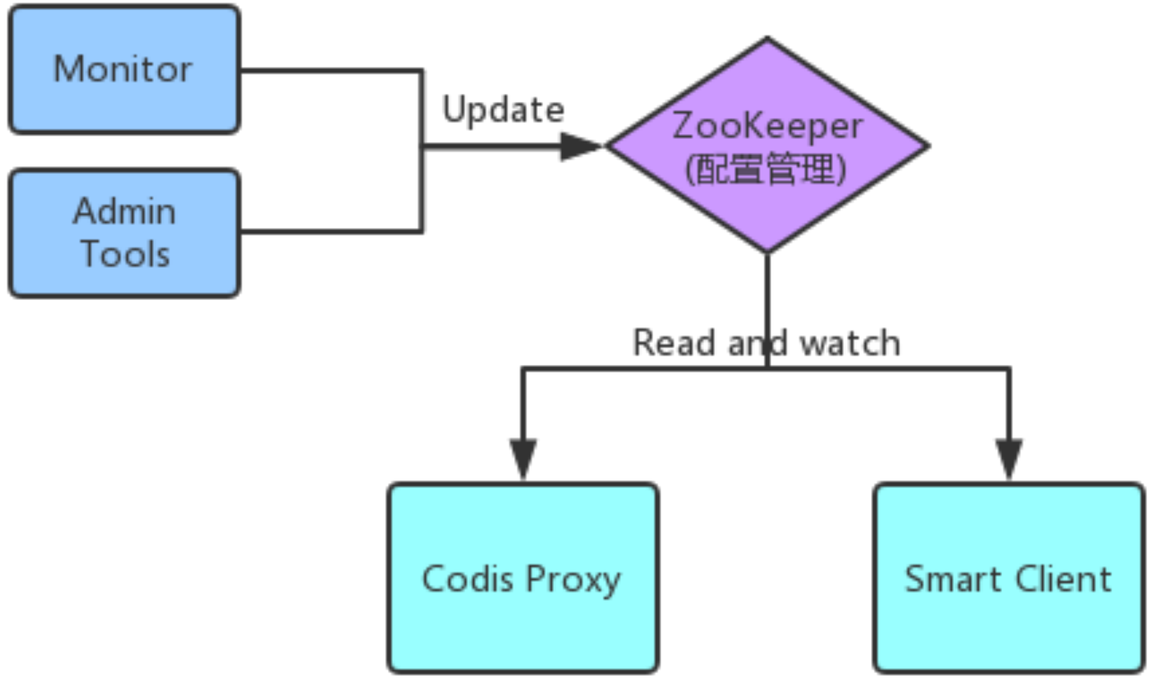
- KV数据模型
  - Redis数据结构与协议




- 二维表数据模型
  - 支持主键范围查询(必须)、其他字段条件过滤：
  - *SELECT f1, f2 FROM t WHERE primary > v1 AND primary <= v2 AND f3 < v3 OFFSET vx LIMIT y;*

# 数据分片

- 数据路由表
  - 保存从Key到高可用数据副本集的映射关系
  - 由Proxy/Client启动时从ZooKeeper载入，在本地内存缓存
  - 保持Watch，一旦数据路由表有变更便进行重载



# 数据分片

- KV数据模型
  - Slot： 数据路由的最小单位
  - SlotIndex = Hash(Key) % SLOT_NUM
  - 数据分布更均衡

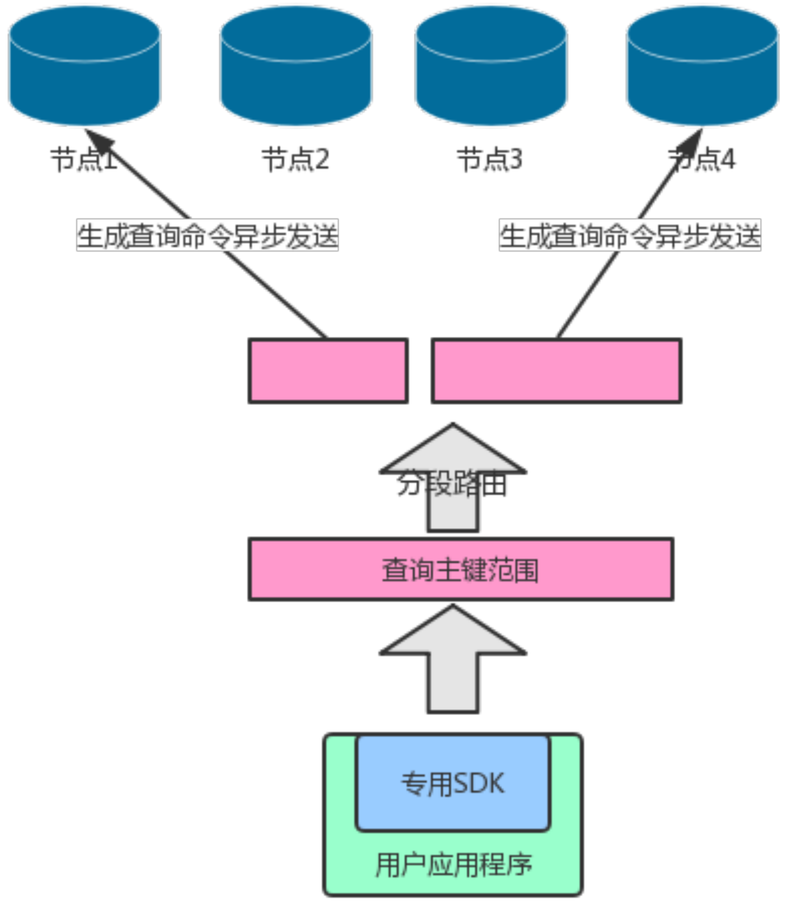
| KV数据模型路由表 | | |
|---|---|---|
| Slot1 | Master Instance | Slave Instance |
| Slot1 | Master Instance | Slave Instance |
| ... | ... | ... |

- Table数据模型
  - Region: 标识Key范围的起始与结束
  - Region可动态分裂与合并
  - 可支持主键的范围查询
  - 需要执行数据均衡

| 二维表数据模型数据路由表 | | | | |
|---|---|---|---|---|
| Region 1 | 主键Start | 主键End | Master Instance | Slave Instance |
| Region 2 | 主键Start | 主键End | Master Instance | Slave Instance |
| ... | ... | ... | ... | ... |

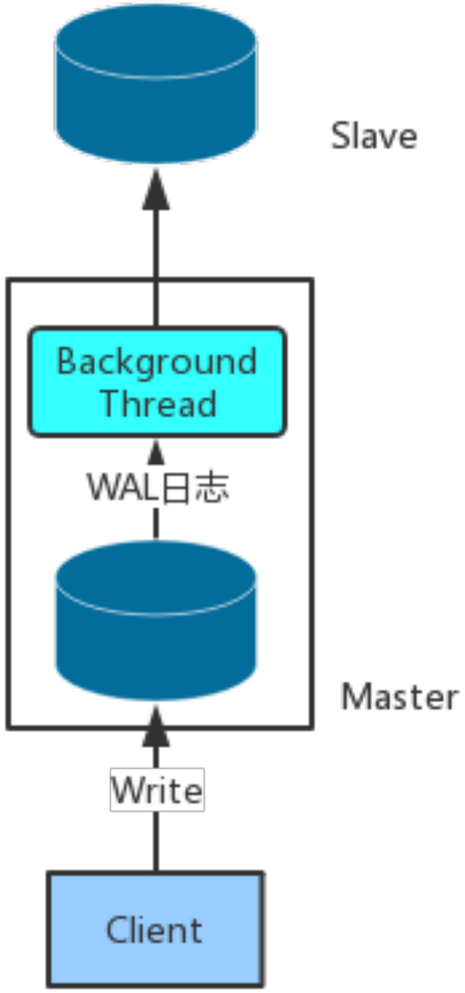
# 数据分片

- Table数据模型的路由过程
  - *SELECT f1, f2 FROM t WHERE primary > v1 AND primary <= v2 AND f3 < v3 OFFSET vx LIMIT y;*
  - 根据主键范围进行分段路由
  - 使用异步IO将请求发给多个节点
  - 客户端在收到所有节点响应后Merge最终结果集



# 数据同步

- 性能第一
- 异步线程同步降低请求时延
- RocksDB存储引擎基于WAL日志实现
  - 主机使用异步IO向备机发送WAL日志
  - 备机落盘成功后响应确认ACK(WAL seq)
  - 支持断点续传,降低数据同步带来的IO波动
- 主备数据时延： <10ms
- 主备切换时极小窗口数据不一致
- 保证最终一致
  - 故障节点重启后继续同步到对端节点
  - 使用PPTP协议校准微秒级时间戳解释数据冲突



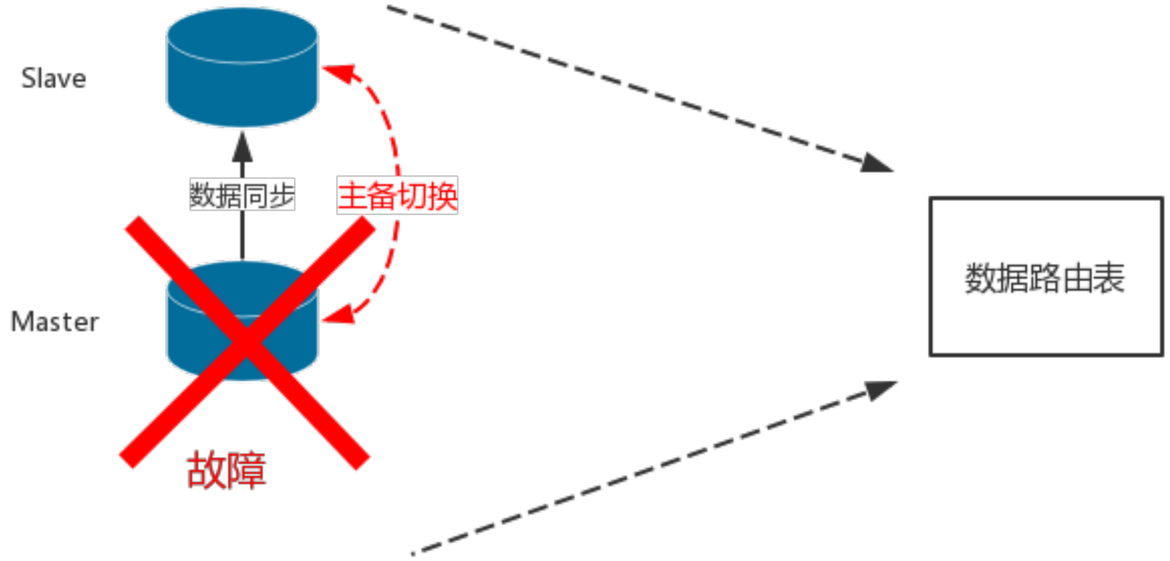
# 故障检测

- 每个集群都部署多个（3~5）Monitor实例
- 每个Monitor定时对每个节点进行心跳检测，并将结果记录在ZK中，形成节点健康状态表
- 当多数Monitor都判定某个节点状态异常时则任务该节点故障

| | Instance 1 (MASTER) | Instance 2 (SLAVE) | ... |
|---|---|---|---|
| Monitor 1 | 0 | 3 | ... |
| Monitor 2 | -1 | 3 | ... |
| ... | ... | ... | ... |

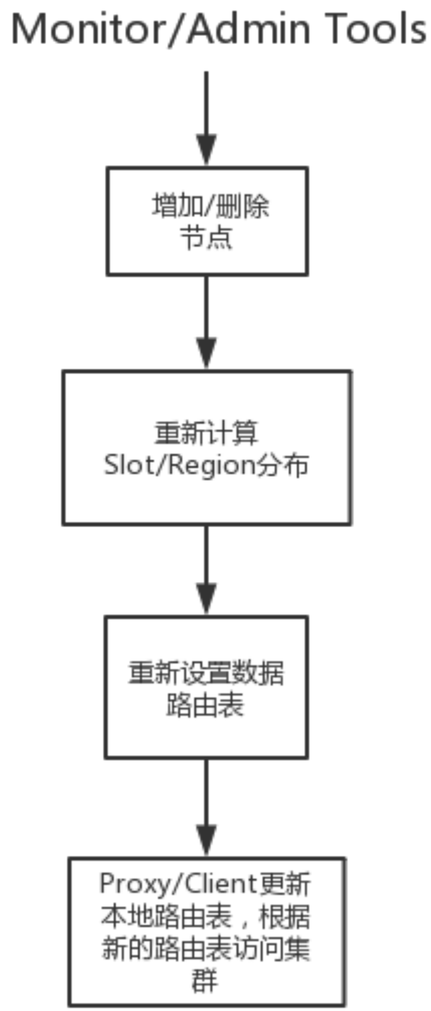
# 故障转移

- 当判定某个节点故障后所有Monitor会在ZK上竞争互斥锁
- 竞争成功的Monitor最终判断一次是否满足故障转移的条件
- 若满足故障转移的条件则更新路由表完成主备切换，故障转移
- 整个故障转移过程在3-5秒内完成



# 弹性伸缩

- 存储与负载能力的线性扩缩容
- 扩缩容过程是对上层业务透明的，执行期间写入、读取正常执行
- 50%的时延增加
- 扩缩容的过程即是Slot/Region在格节点间的重新分布过程



# 数据迁移

- 扩缩容的关键任务是要完成Slot/Region在节点间的透明迁移
  - 首先将Slot/Region标记为迁移状态
  - Monitor观察到Slot/Region进入迁移状态后就向对应节点发送迁移命令

| Slot Index | 迁出Instance | 迁出Instance Slave | 迁入Instance | 迁入Instance Slave |
|---|---|---|---|---|

| Region Index | Region Start | Region End | 迁出Instance | 迁出Instance Slave | 迁入Instance | 迁入Instance Slave |
|---|---|---|---|---|---|---|

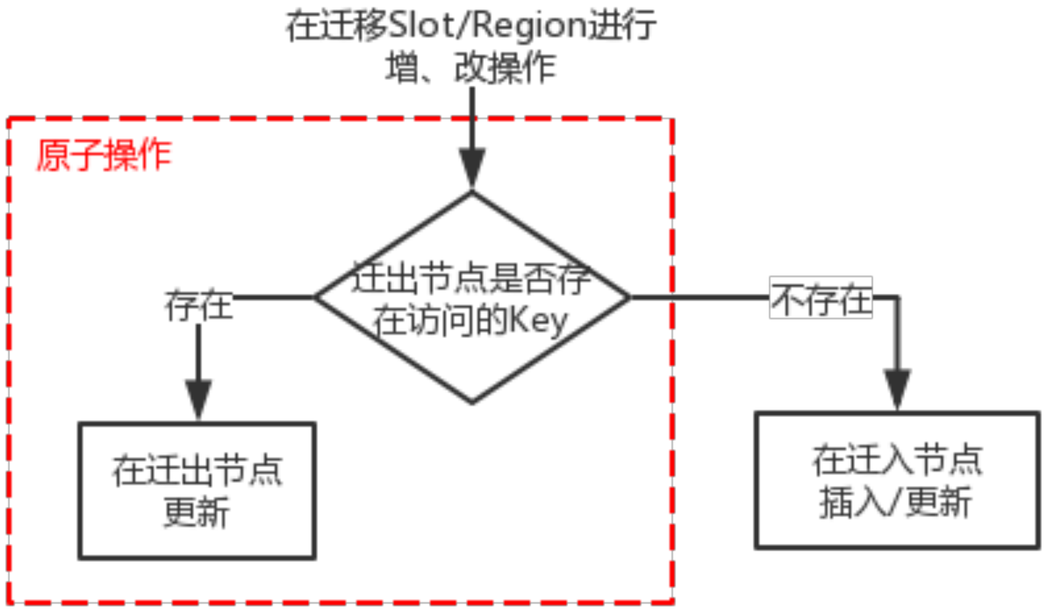
# 数据迁移

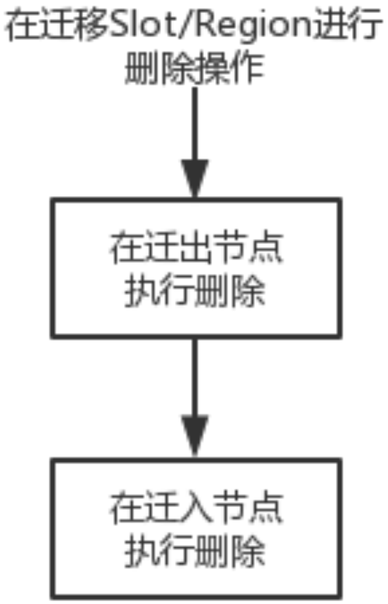
- 迁移命令执行时
  - Redis存储引擎使用Scan命令扫描符合要求的Key集合
  - RocksDB在Key前缀中编码Slot，保证相同的Slot连续存储，迁移时连续扫描需要迁移的Key集合

- 节点间使用单Key迁移命令进行数据迁移
  - 单Key迁移命令保证在迁入节点落盘成功之后再从迁出节点删除

# 数据迁移

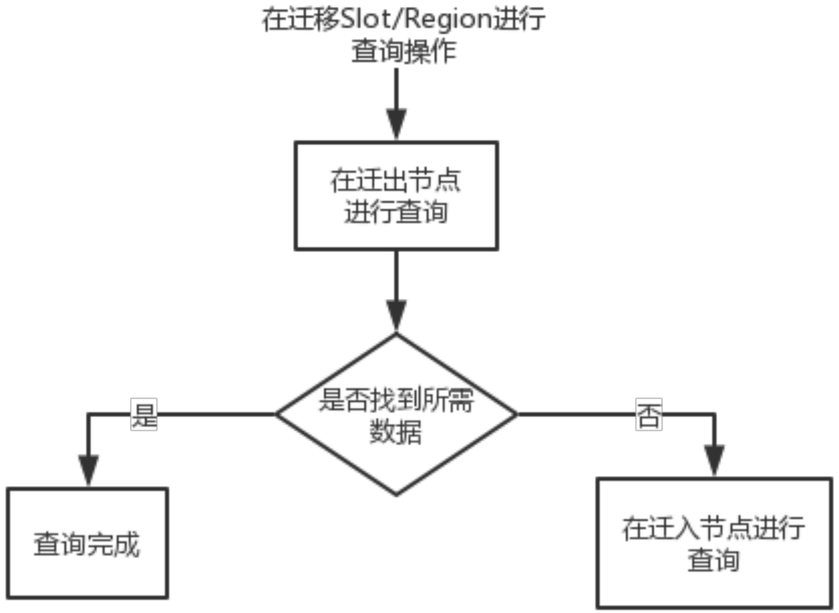
- 迁移期间的Slot/Region使用专门的写策略保证数据一致性



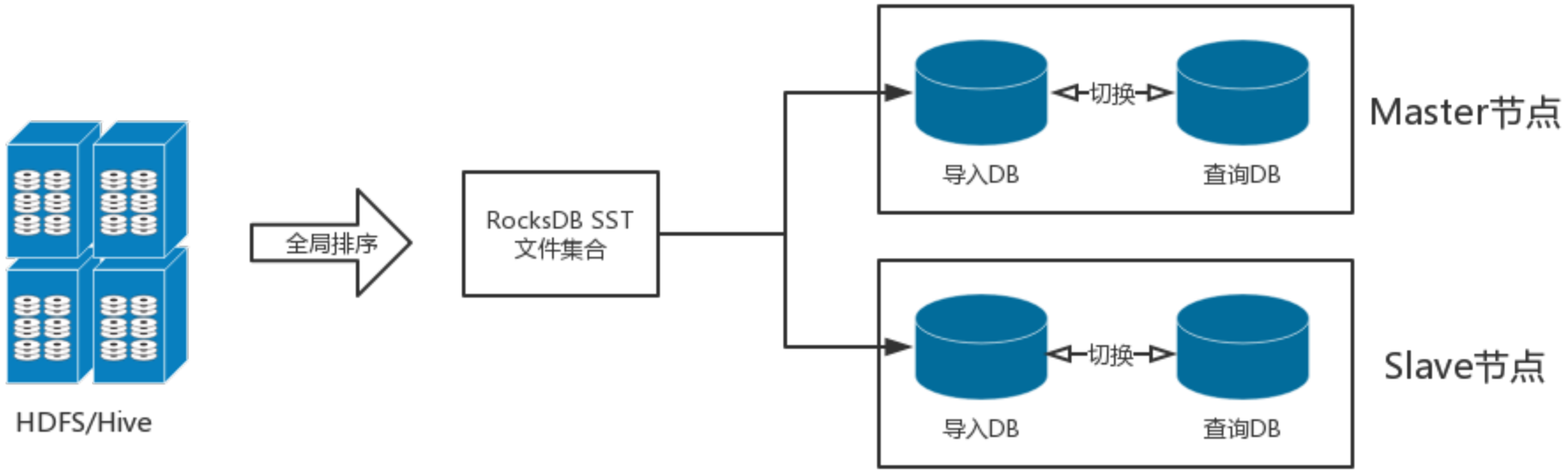
# 数据迁移

- 迁移期间的Slot/Region使用专门的读策略保证数据一致性



# 高速数据载入

- 支持大数据场景
  - 支持RocksDB存储引擎
  - 单节点最快载入速度 ~300MB/s
  - 大批量数据载入时不影响前台业务查询



# *Q & A*