Elasticsearch技术应用

目录

- Elasticsearch 介绍
- Elasticsearch在百度的应用
- Elasticsearch的一些改进

Baidu 百度

目录

- Elasticsearch 介绍
- Elasticsearch在百度的应用
- Elasticsearch的一些改进

Baidu 百度

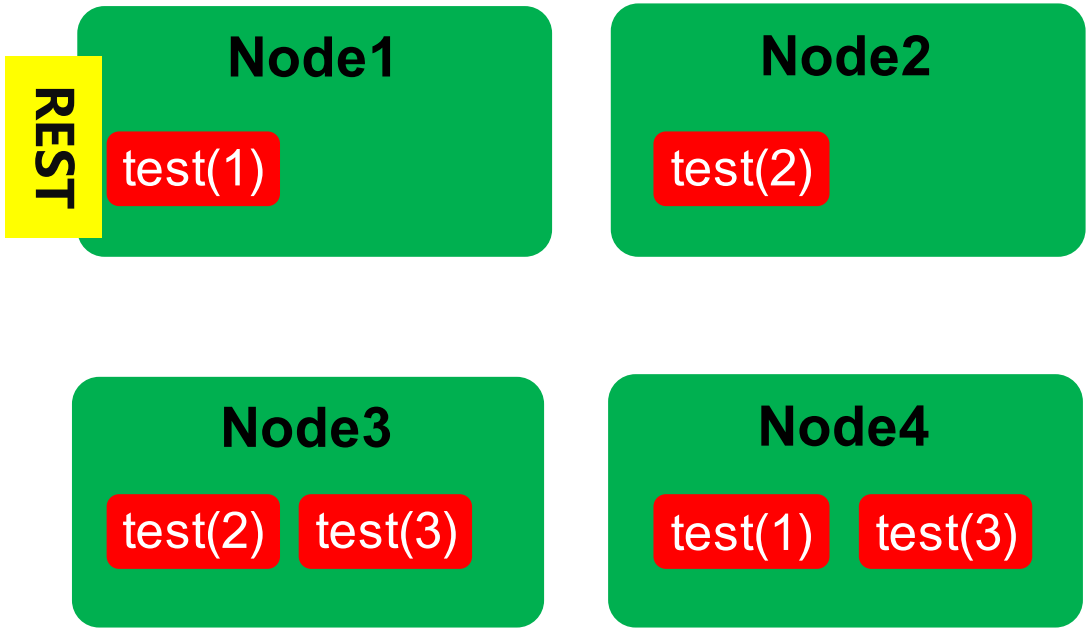
Elasticsearch 介绍

- 构建在Lucene上的搜索引擎
- 分布式的架构
  - 分布式索引
  - 多副本高可靠
- 全文检索系统 -->数据分析平台

Baidu 百度

Elasticsearch介绍

```
POST /test
{
    "settings": {
        "number_of_replicas": 2,
        "number_of_shards": 3
    }
}
```

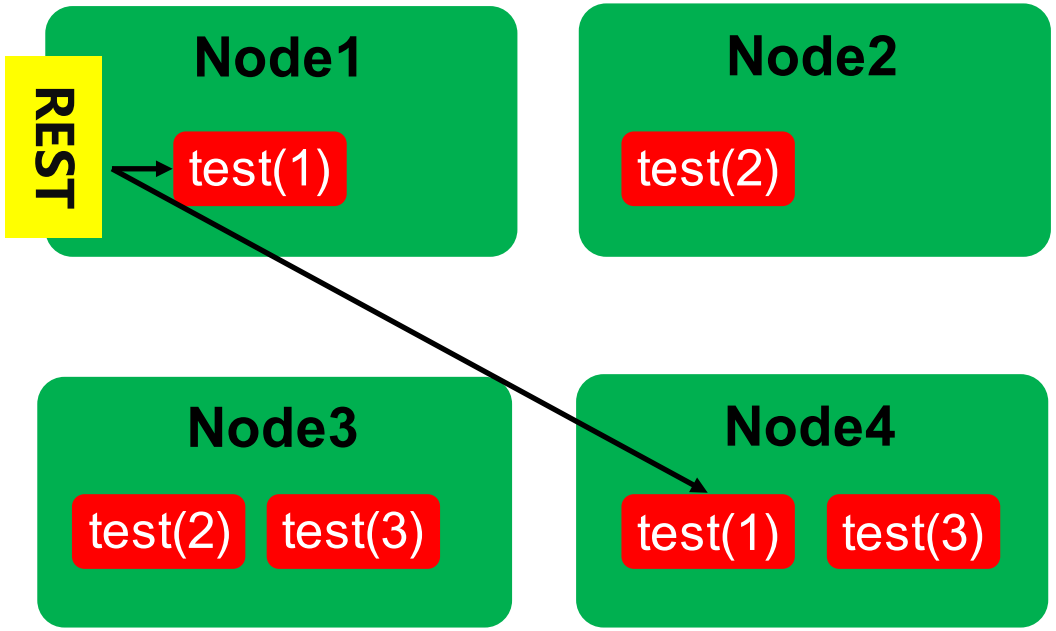


- ✓ Index类似于database，是数据的物理管理方式
- ✓ 一个Index被划分为多个shard
- ✓ 每个shard可以有多个副本
- ✓ 通过REST API来访问

Baidu 百度

Elasticsearch介绍

```
POST /test/test_table1/2
{
    "name": "yiguolei",
    "age": "29",
    "others": {
        "address": "beijing",
        "phone": "13439910228"
    }
}
```

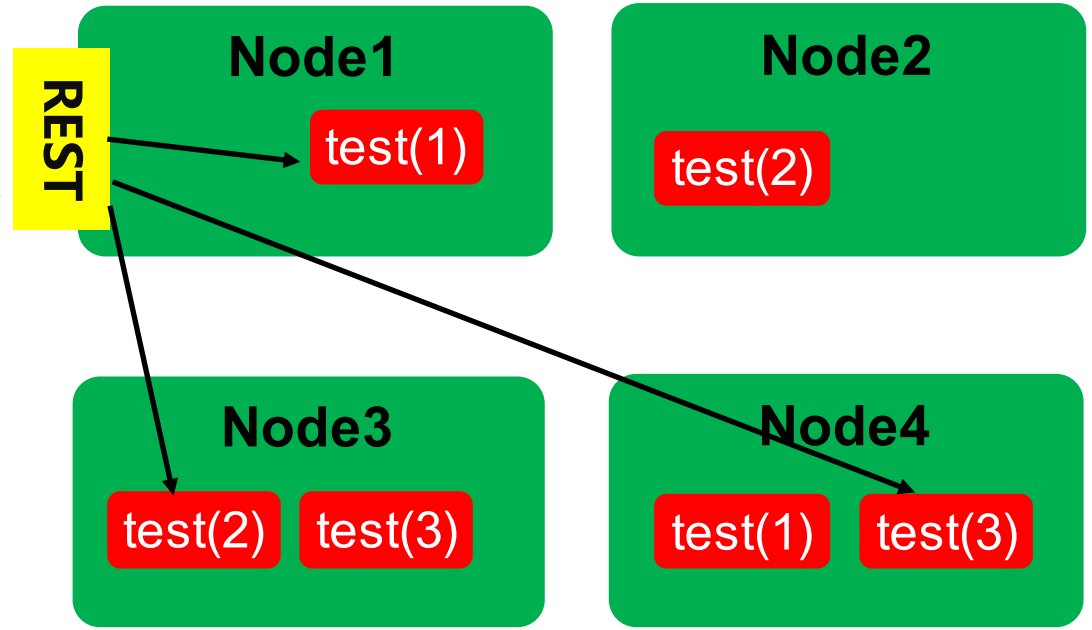


- ✓ 接收json格式的文档
  - ✓ 非结构化—普通文本
  - ✓ 半结构化—日志，邮件文档
- ✓ 利用id hash来将数据划分到各个shard上
- ✓ 在导入时支持对文本进行分词处理

Baidu 百度

Elasticsearch介绍

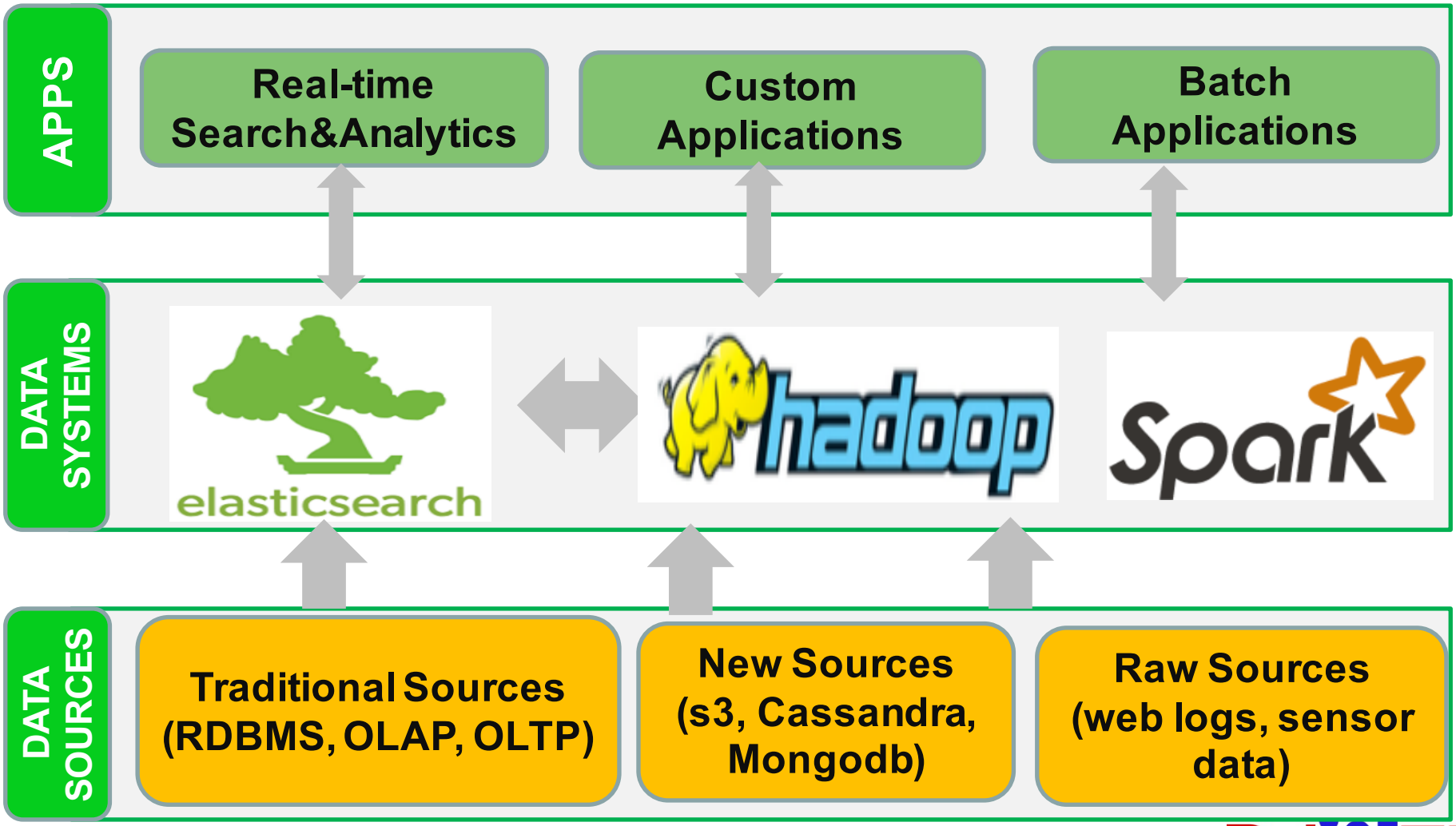
```
POST /test/_search
{
    "query": {
        "term": {
            "age": 29
        }
    }
}
```



- ✓ 用json来描述查询语句
- ✓ MPP的方式查询所有相关的shard
- ✓ 查询支持模糊匹配，聚合计算等功能

Baidu 百度

生态


APPS
Real-time Search&Analytics
Custom Applications
Batch Applications
DATA SYSTEMS
elasticsearch
hadoop
Spark
DATA SOURCES
Traditional Sources (RDBMS, OLAP, OLTP)
New Sources (s3, Cassandra, Mongodb)
Raw Sources (web logs, sensor data)

Baidu 百度

典型客户

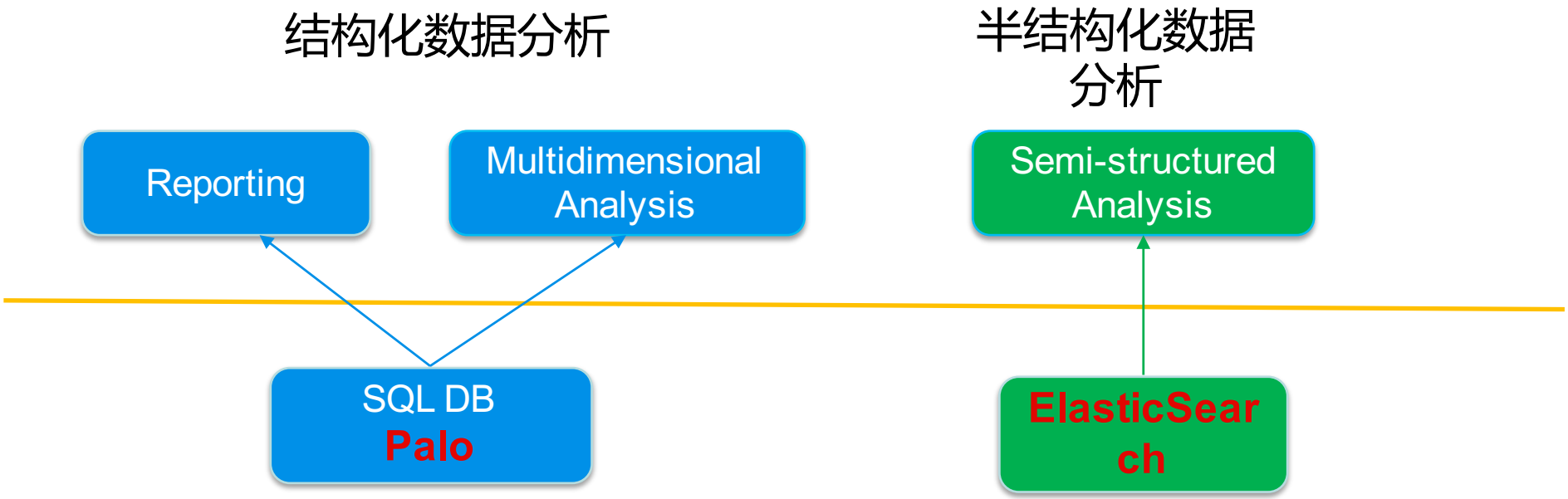
- 支撑对源码、repo的搜索，对用户操作审计、代码异常分析
- 400万用户， 800万repo，20亿文件，30TB存储

目录

- Elasticsearch 介绍
- Elasticsearch在百度的应用
- Elasticsearch的一些改进

Baidu 百度

ES 定位



- ✓ **Function： query、analysis**
- ✓ **Interactive Data Analysis vs. Batch Data Processing**
- ✓ **Palo: a MPP-based Interactive Data Analysis SQL DB**
- ✓ **Elasticsearch: a Distributed Real-time Document Indexer with Support for Online Analytics**

Baidu 百度

Elasticsearch的优势

多维分析	✓ 任何字段都可以建立索引、使用多个索引进行分析 ✓ 丰富的分析函数 avg, sum, max, min, count ✓ 文本分词能力
实时性	✓ 数据实时更新 ✓ 实时检索,实时分析,毫秒级响应
Flexible Schema	✓ Json格式的数据,支持嵌套、集合、父子关系多种模式 ✓ 字段动态增加
Easy to Use	✓ Standalone的架构可以独立部署、没有大量依赖 ✓ RESTful API: 查询 & 管理
数据加载	✓ 单节点100GB/小时的加载能力满足大部分需求
其他	✓ 扩展性、可用性、share nothing…

Baidu 百度

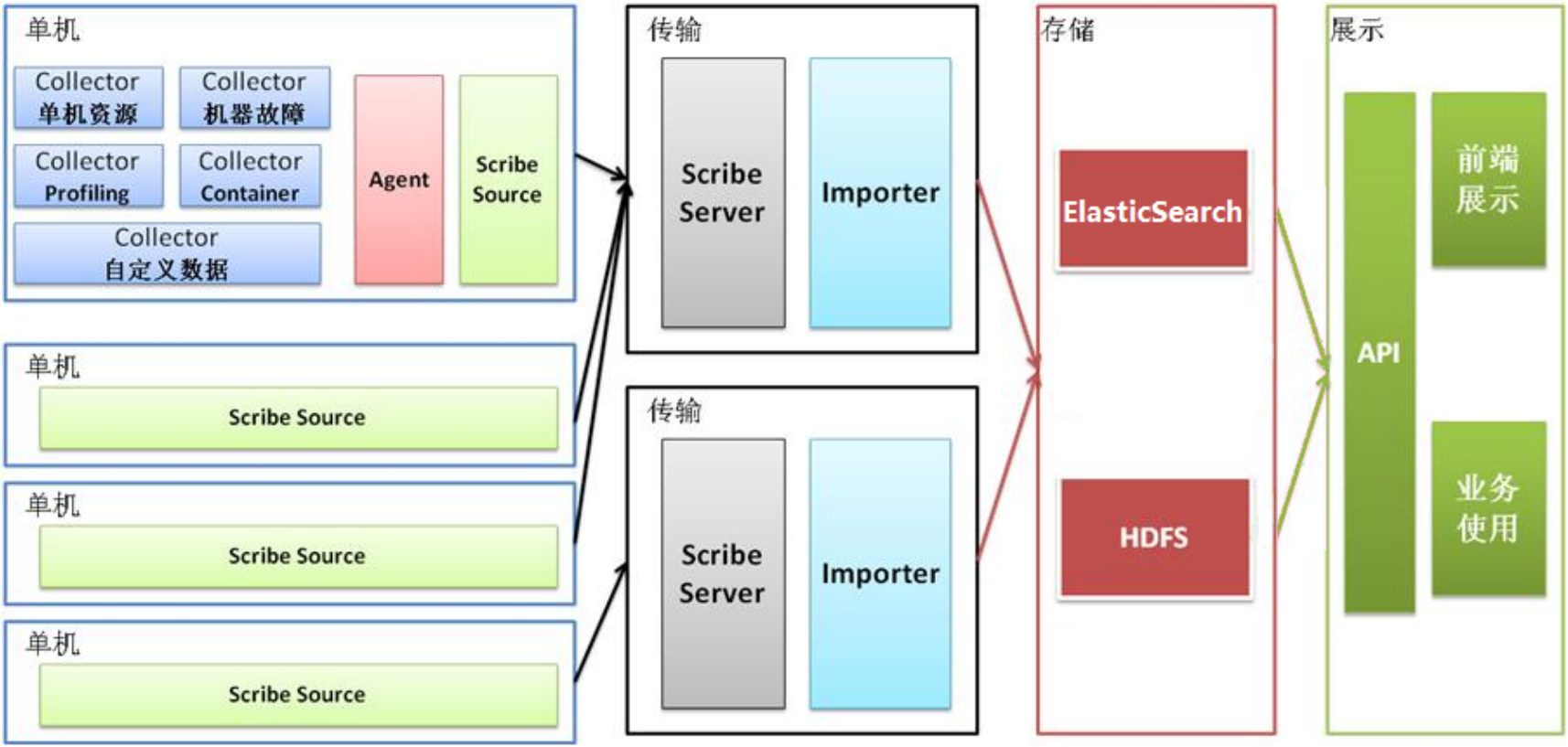
Elasticsearch在百度的应用

- 2013年10月开始使用
- 目前覆盖百度内部40多个业务线
- 单集群每天导入30TB+数据，总共每天60TB+
- 单集群最大100台机器，200个ES节点
- 共使用近400台机器，启动700+ES节点

Baidu 百度

ES实践-监控业务

百度的ELK架构



Baidu 百度

ES实践-监控业务

- 主要挑战
  - 字段不确定
  - 数据量较大，每天30TB+数据，24小时不间断导入
  - 小时级任意维度聚合分析毫秒级返回，天级秒级返回

Baidu 百度

ES实践-监控业务

- 动态字段
  - 一律接受 OR 一律拒绝
  - 按照规则来接受、处理

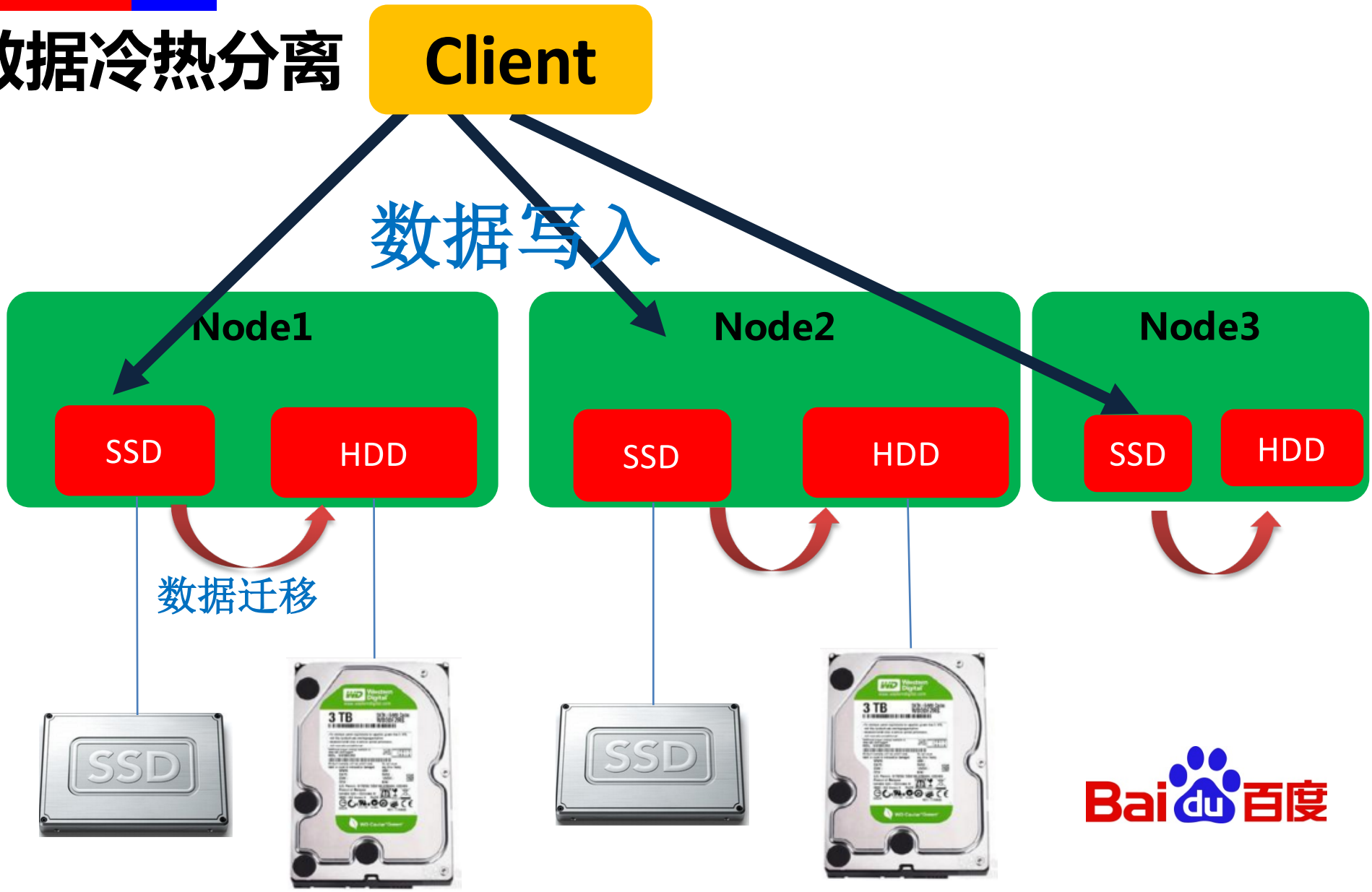
```
"mappings": {
    "_default_": {
        "dynamic_templates": [
            { "kvs": {
                  "path_match":"*_kvs.*",
                  "mapping": {
                      "index":"no",
                      "store":false,
                      "doc_values":true,
                      "type": "long"
                  }
            }}
        ]
```

```
POST /casio-machines-20151017/all
{
   "instance":1001,
   "long_kvs":{"mem":12}
}
```

Baidu 百度

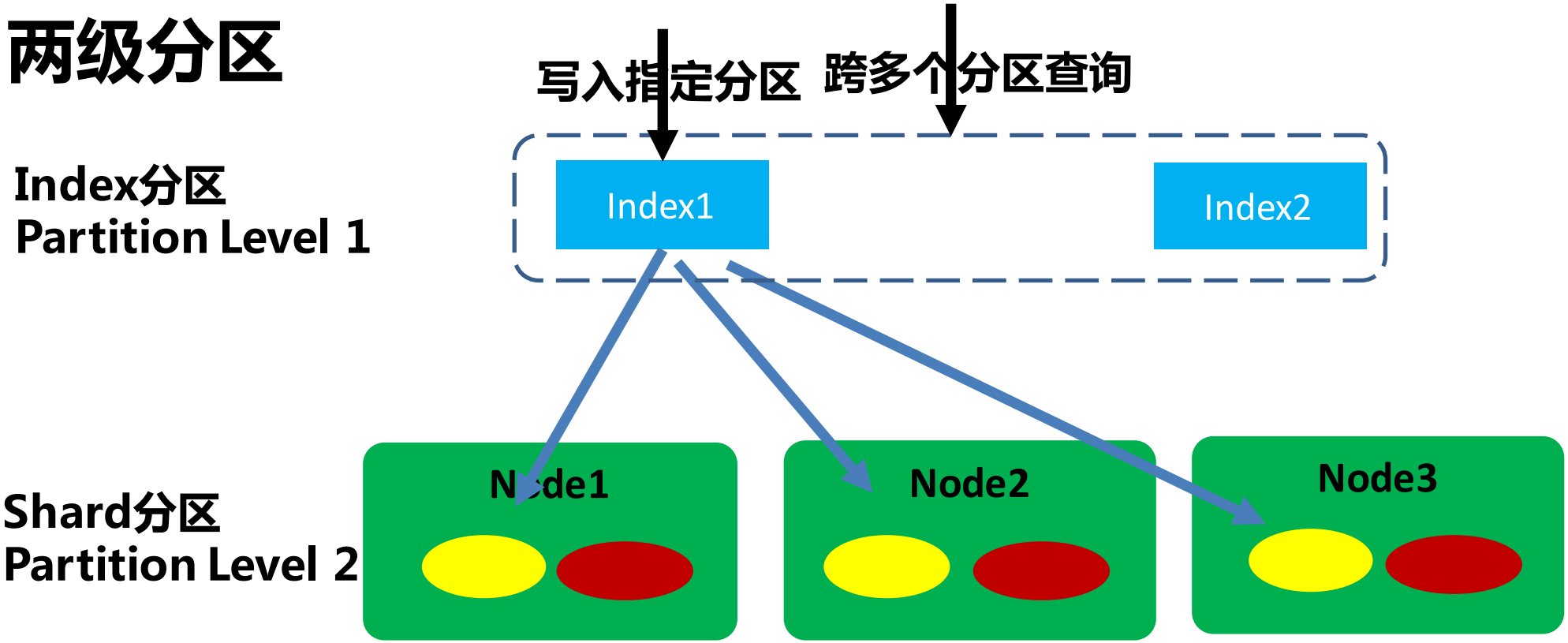
ES实践-监控业务

数据冷热分离



ES实践-监控业务

两级分区



- ✓ 按照时间做一级分区，建立index
  - ✓ 相同类型的index用alias的方式结合起来，跨index查询
  - ✓ 过期的数据可以自动删除
  - ✓ 新数据所在的index使用SSD存储
- ✓ 使用用户指定的key做二级分区

Baidu 百度

ES实践-百度糯米

与OLTP集成

- ✓ 线性扩展
- ✓ Flexible Schema
- ✓ 高性能

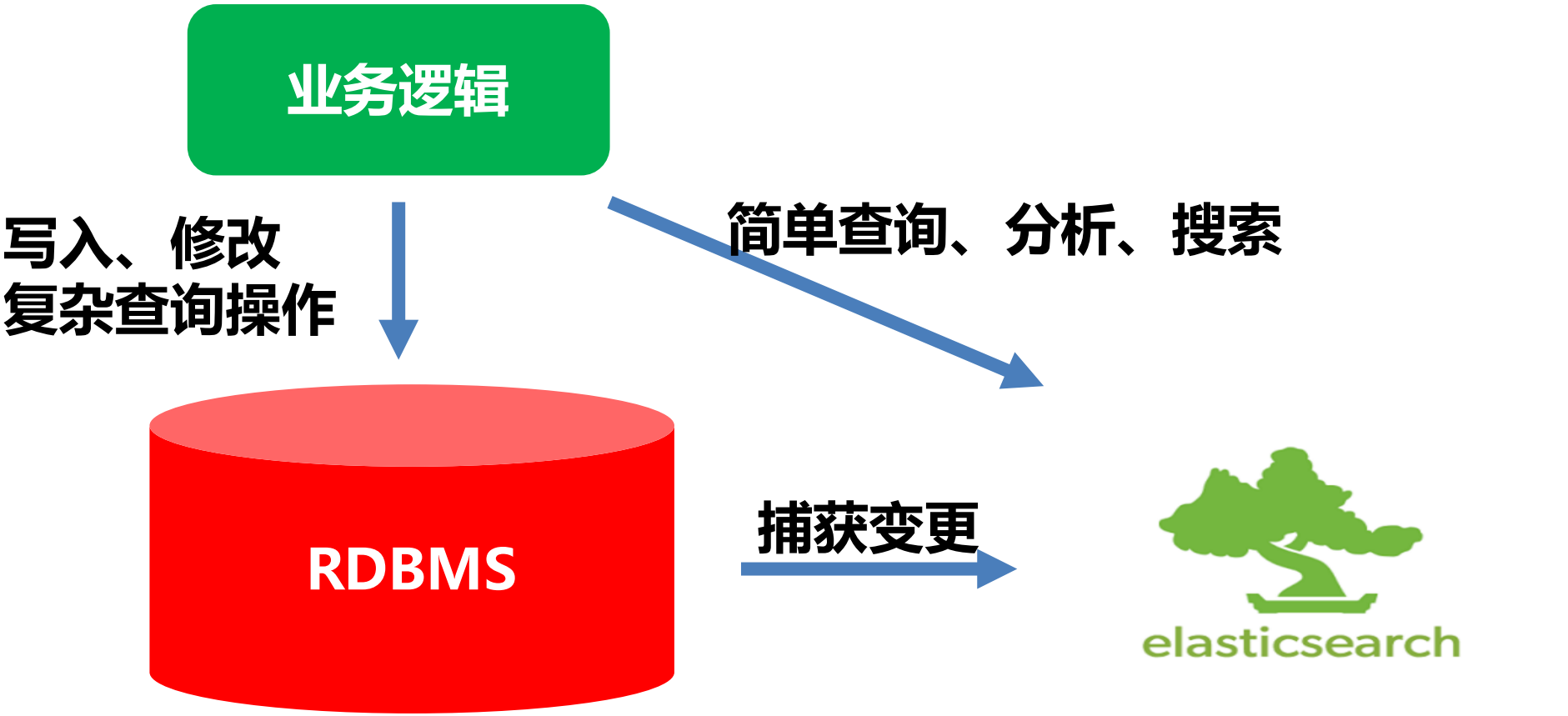
Elasticsesarch

- ✓ 强大的事务支持
- ✓ 多表join能力

RDBMS

Baidu 百度

ES实践-百度糯米



- ✓ 用户写入或者修改操作-->RDBMS
- ✓ 复杂的查询 --> RDBMS
- ✓ 简单的查询 --> Elasticsearch
- ✓ 搜索请求 --> Elasticsearch

Elasticsesarch
+
RDBMS

Baidu 百度

ES实践-百度糯米

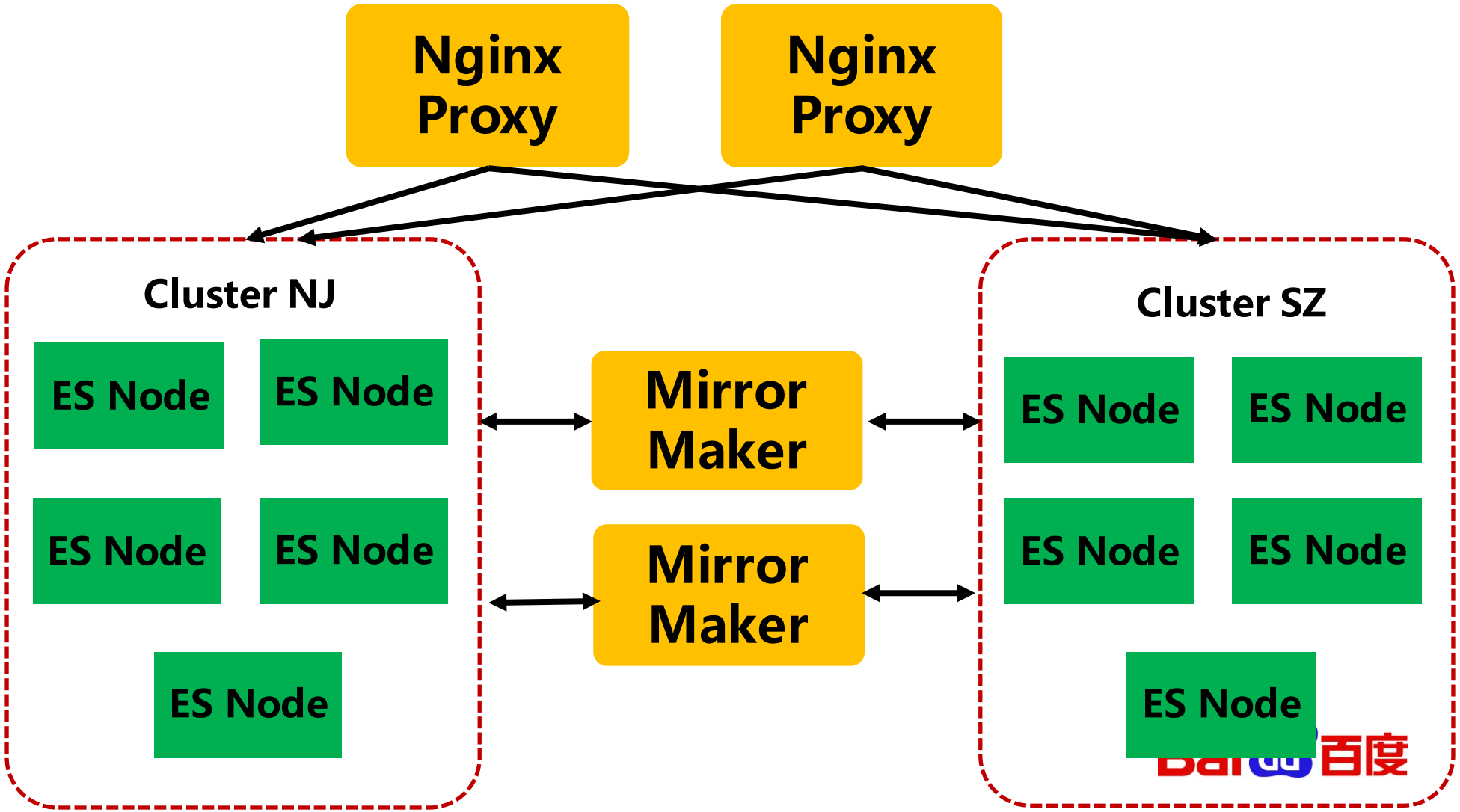
- 捕获变更的几种方式
  - CDC系统
    - 优点：应用不需要做任何工作，实时性比较高，异步做，不影响插入性能
    - 缺点：依赖binlog格式解析工具，并且一条记录的多次修改，会被执行多次update
  - 触发器
    - 优点：比较方便，容易使用
    - 缺点：对在线业务有影响
  - 业务层标记
    - 可以做到对数据同步的精确控制，多次更新可以一次修改
    - 缺点：业务层需要修改业务逻辑，modify_time, is_del等标记

ES实践-百度糯米

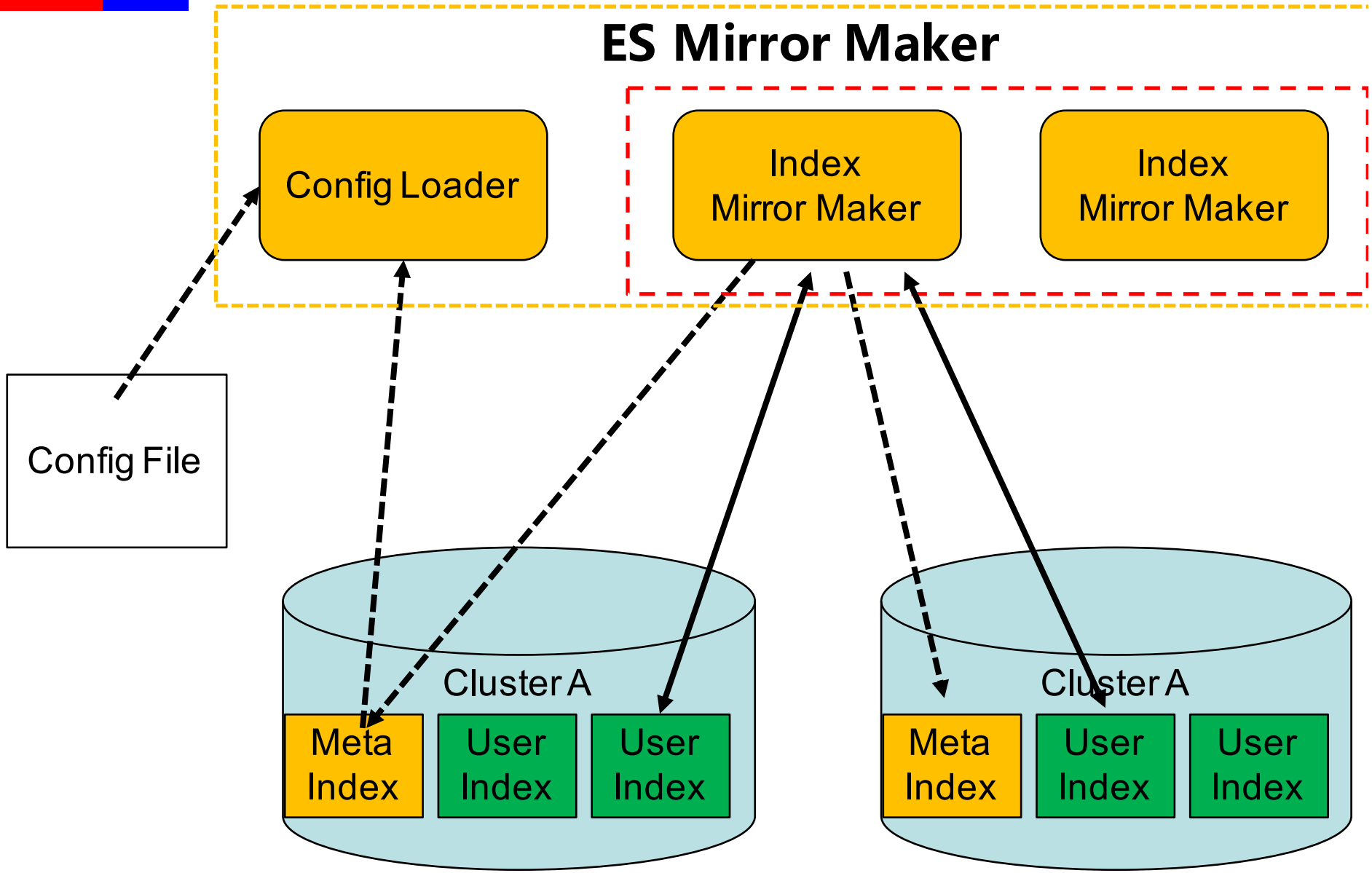
- 跨机房部署
  - 背景
    - 集群由于一些bug导致服务不可用或者数据丢失
    - 单个机房可能出现故障导致服务不可访问
  - 双机房的现实条件
    - 网络延时比较高
    - 两个机房之间偶尔断线
    - 同步的双机房方案不可取

Baidu 百度

ES实践-百度糯米


Nginx Proxy
Nginx Proxy
Cluster NJ
ES Node
ES Node
ES Node
ES Node
ES Node
Mirror Maker
Mirror Maker
Cluster SZ
ES Node
ES Node
ES Node
ES Node
ES Node

ES实践-百度糯米


ES Mirror Maker
Config Loader
Index
Mirror Maker
Index
Mirror Maker
Config File
Cluster A
Meta
Index
User
Index
User
Index
Cluster A
Meta
Index
User
Index
User
Index

ES实践-百度糯米

- Mirror Maker实现细节
  - —增量获取
    - 每一条数据都有唯一的时间戳
    - Mirror Maker通过记录的时间戳来获取增量的数据
  - —冲突检测
    - 每条数据都有唯一的版本信息
    - 版本的值绑定到时间戳上或者由用户指定
  - —循环复制
    - ES内部有版本冲突检测机制，版本冲突的不会被写入
    - MirrorMaker在获取到两个集群的数据后需要在内存里做比对，冲突的数据不传送
  - —通过Config来实现复制时间的回退

Baidu 百度

目录

- Elasticsearch 介绍
- Elasticsearch在百度的应用
- Elasticsearch的一些改进

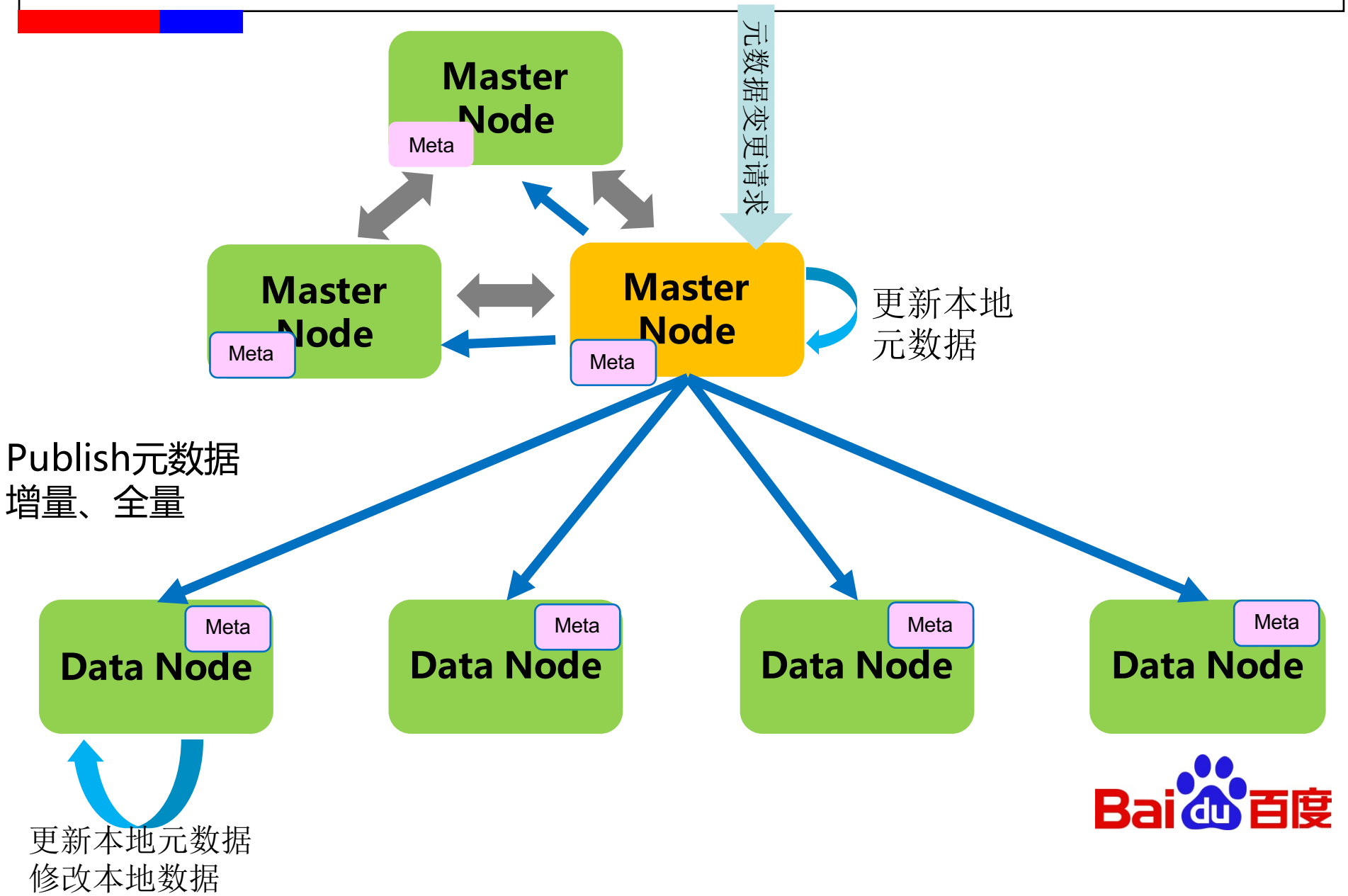
Baidu 百度

Elasticsearch的改进

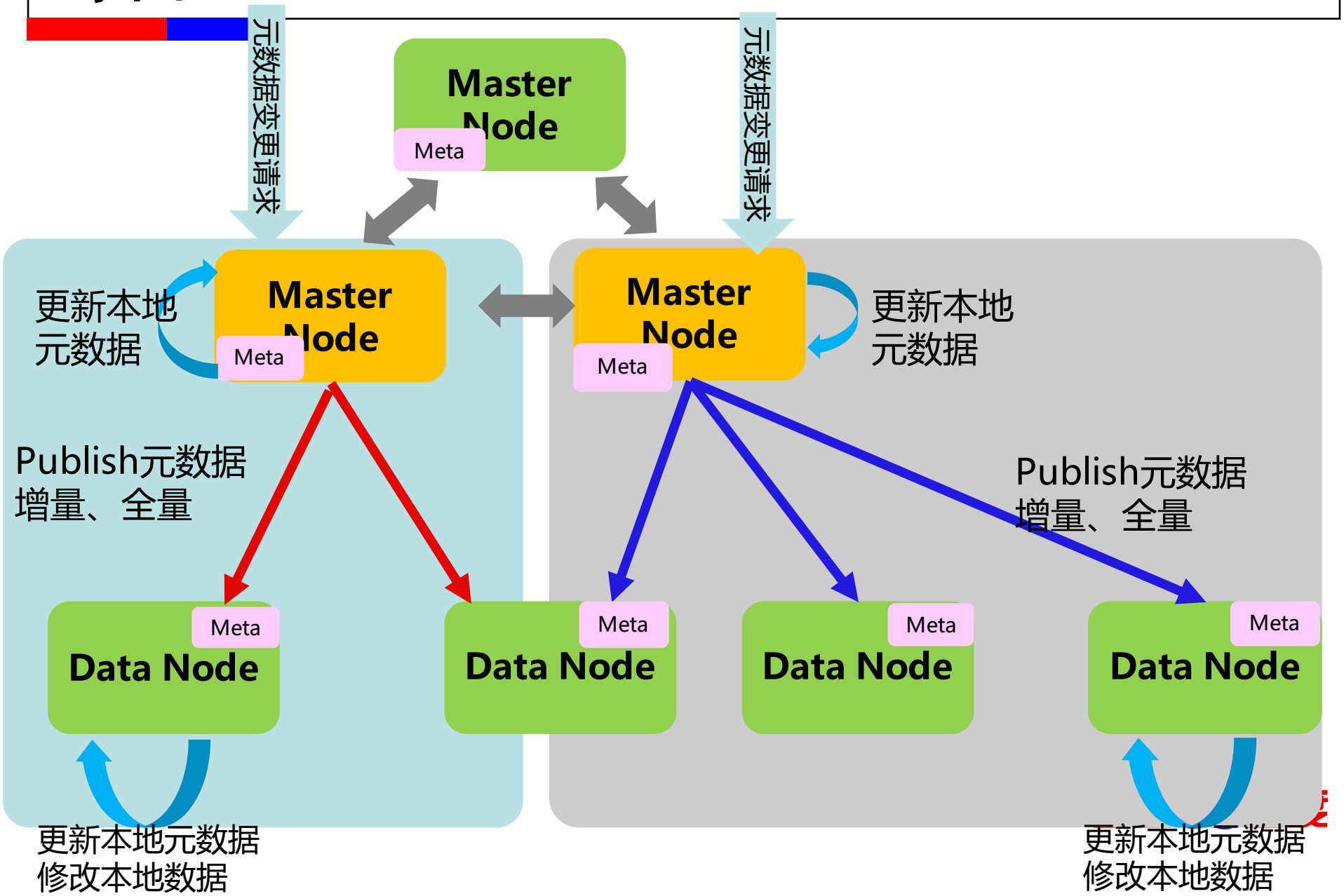
一、脑裂问题

Baidu 百度

原因


Master Node
Meta
元数据变更请求
Master Node
Meta
Master Node
Meta
更新本地元数据
Publish元数据
增量、全量
Data Node
Meta
Data Node
Meta
Data Node
Meta
Data Node
Meta
更新本地元数据
修改本地数据

原因

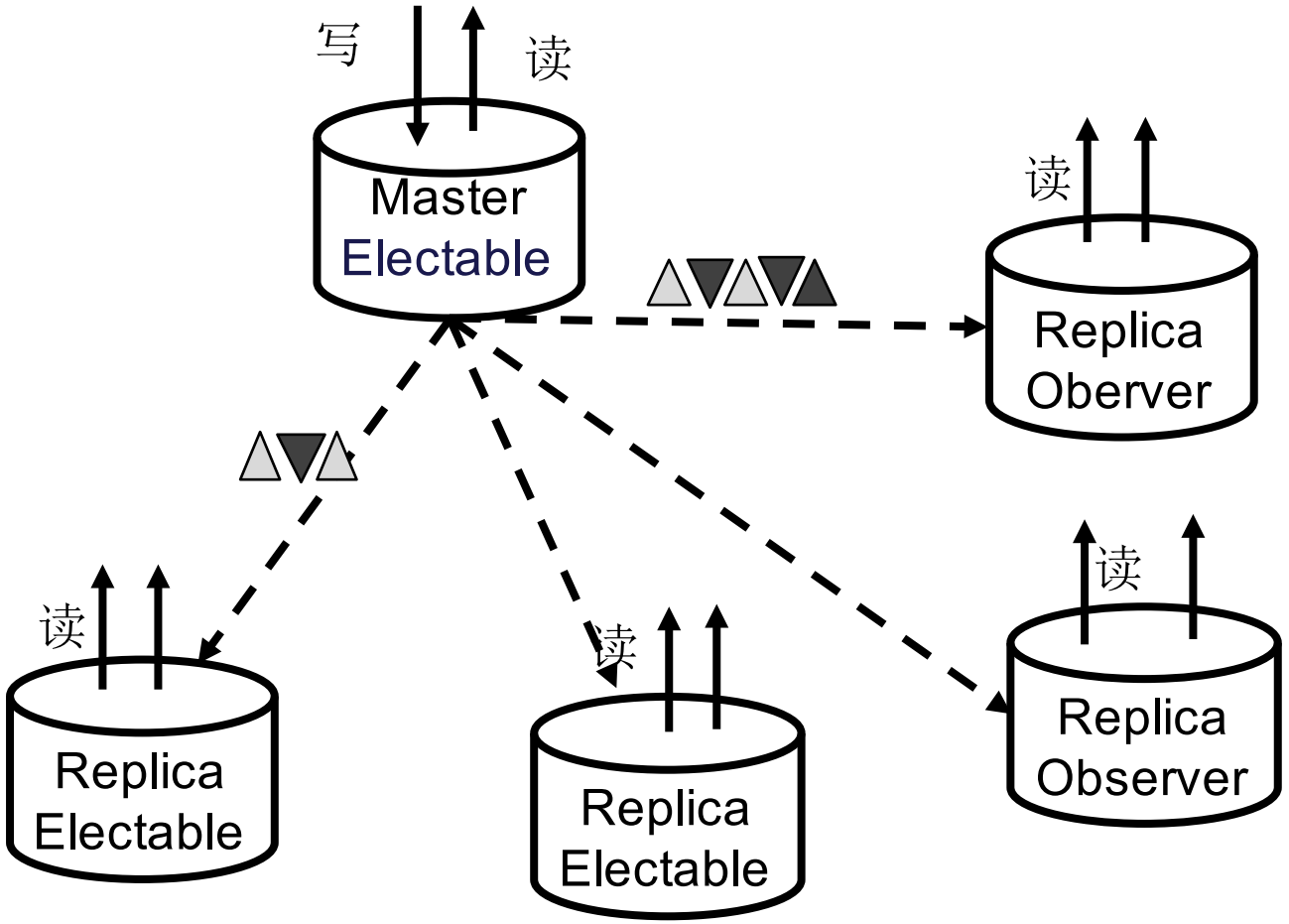

元数据变更请求
Master Node
Meta
元数据变更请求
更新本地元数据
Master Node
Meta
Master Node
Meta
更新本地元数据
Publish元数据
增量、全量
Publish元数据
增量、全量
Meta
Data Node
Meta
Data Node
Meta
Data Node
Meta
Data Node
更新本地元数据
修改本地数据
更新本地元数据
修改本地数据

原因

- 分析：
  - 当一个节点是主时可以随意修改并广播元数据
  - 元数据的修改没有Quorum的方式修改
- 改进思路
  - 需要在多个Master节点之间实现可靠的分布式一致性算法
  - 可选方案：
    - RAFT & PAXOS
    - 实现起来费时费力，不确定能work

Baidu 百度

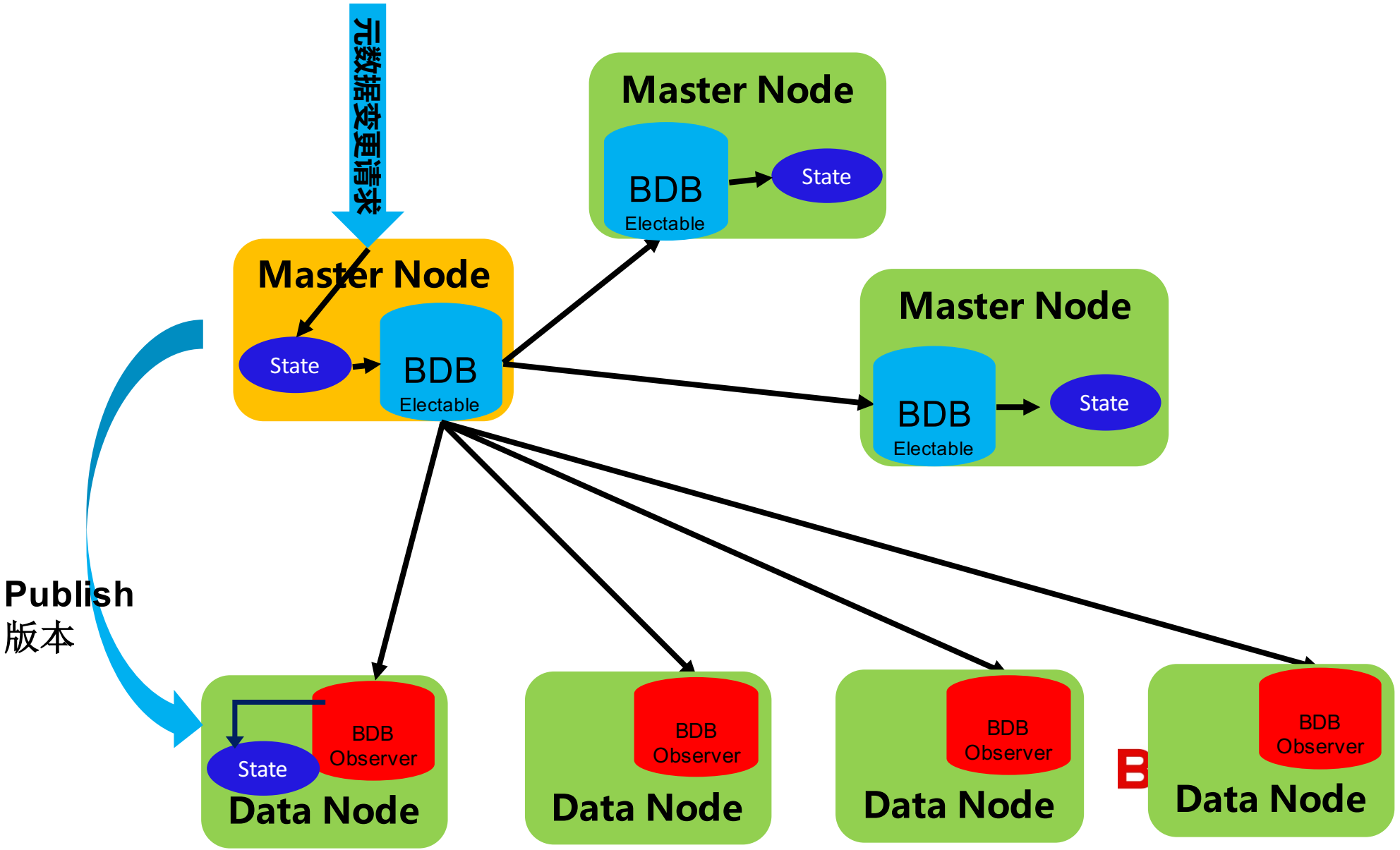
改进方案—BDB JE HA



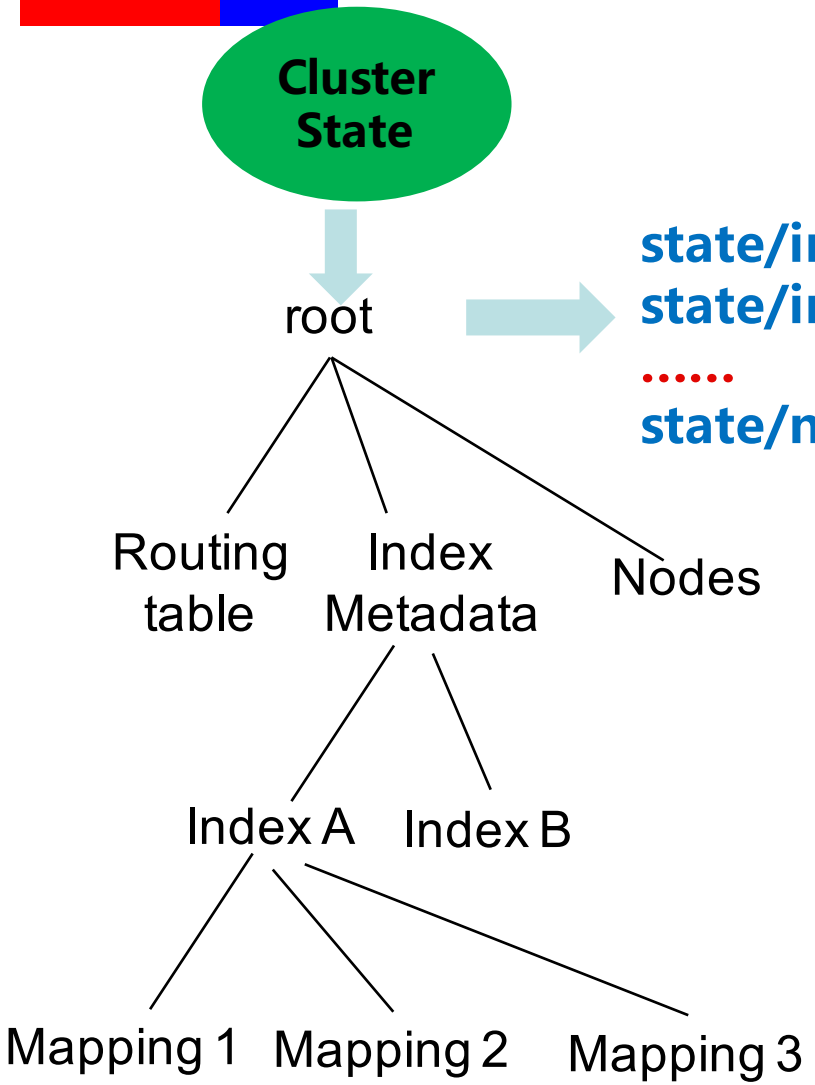
- ✓ 类似Paxos的选举机制
- ✓ 数据在Electable Nodes上需要Quorum写入成功
- ✓ Observer Nodes 异步的接受数据，不参与选举

Baidu 百度

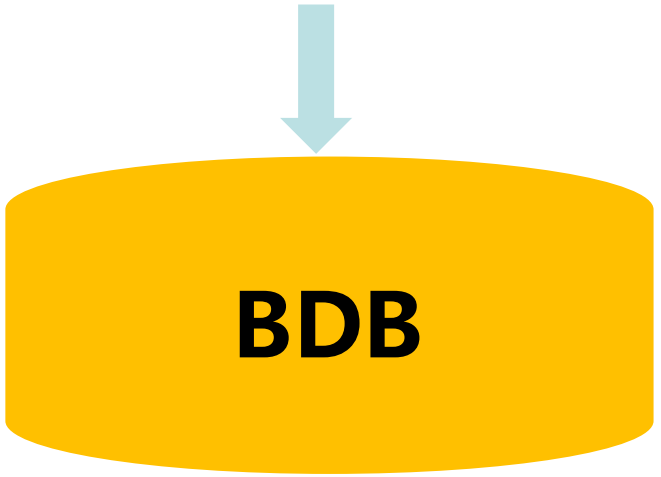
改进方案


元数据变更请求
Master Node
BDB
Electable
State
Master Node
State
BDB
Electable
Master Node
BDB
Electable
State
Publish
版本
State
BDB
Observer
Data Node
BDB
Observer
Data Node
BDB
Observer
Data Node
BDB
Observer
Data Node

改进方案



state/indexmeta/indexa/mapping1 mappinginfo1
state/indexmeta/indexa/mapping2 mappinginfo2
……
state/nodes nodesinfo



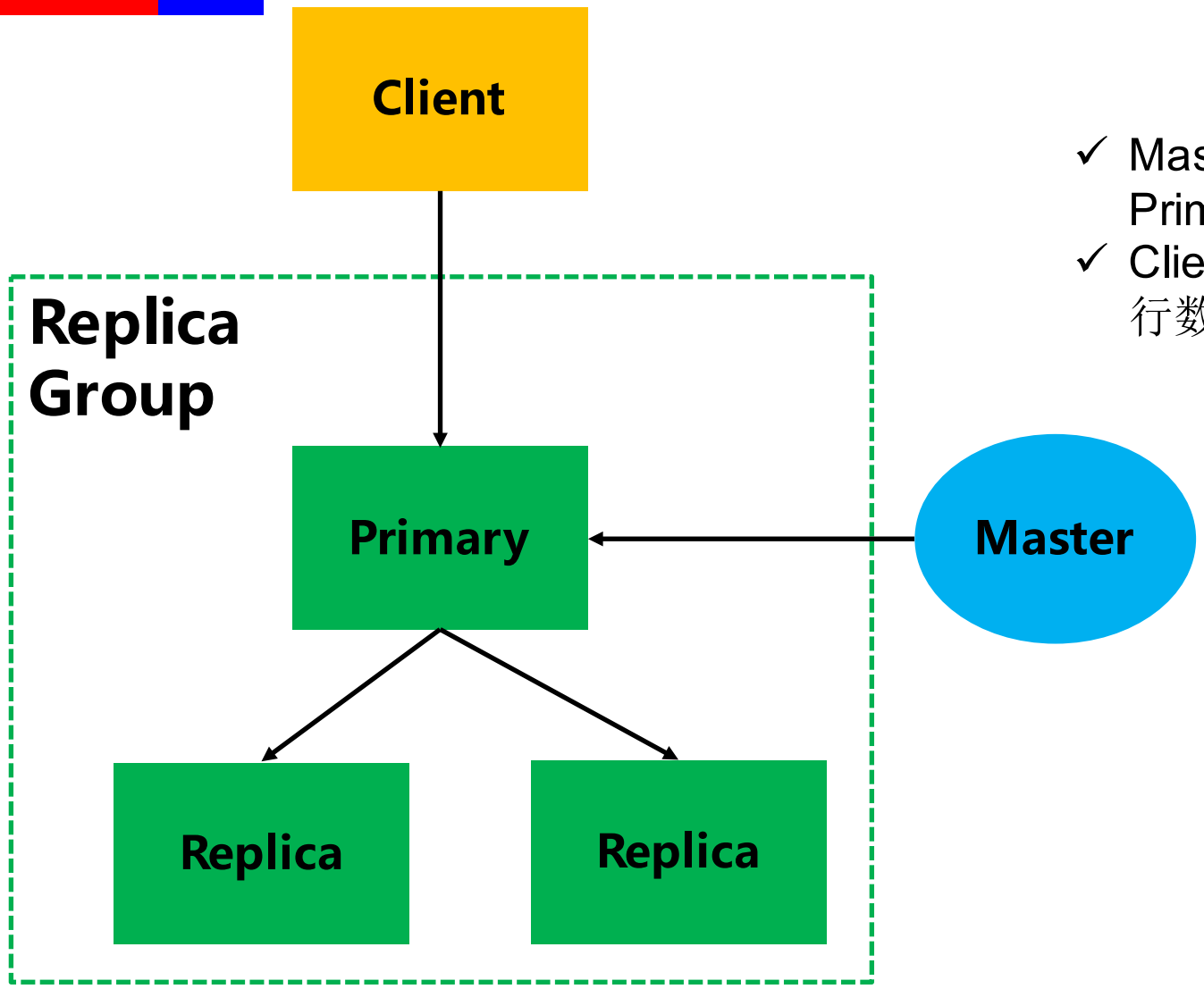
Baidu 百度

Elasticsearch的改进

二、多副本一致性

Baidu 百度

现状

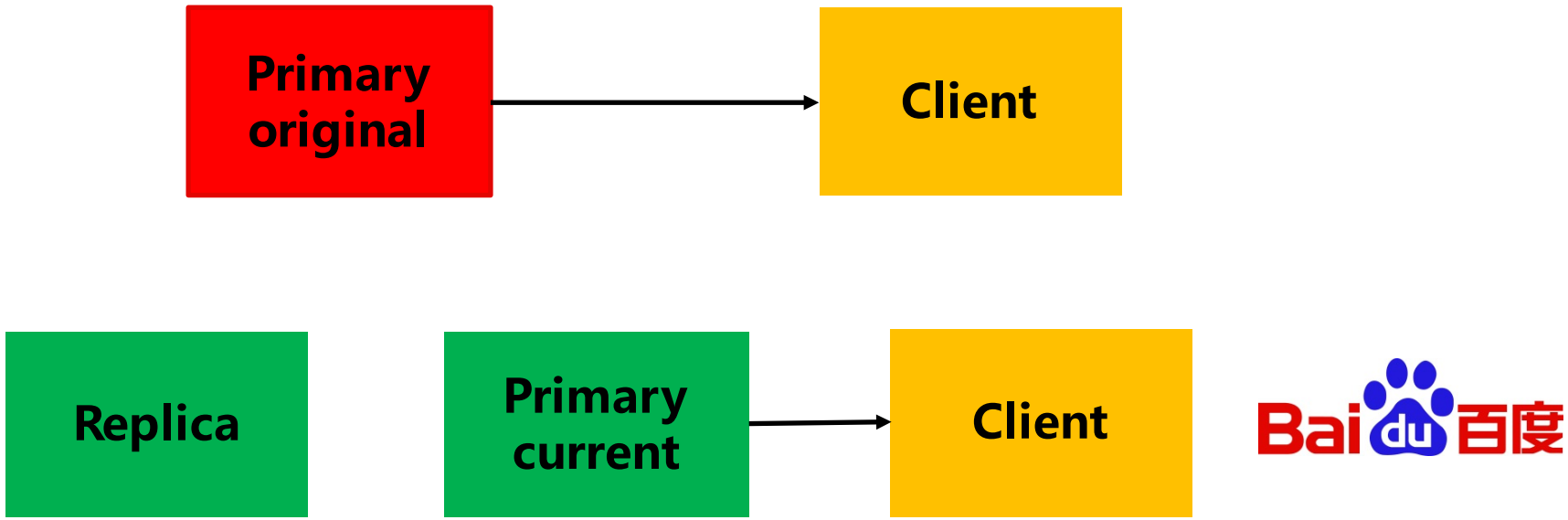


- ✓ Master指定一个副本作为Primary
- ✓ Client通过连接Primary进行数据写入或读取

Baidu 百度

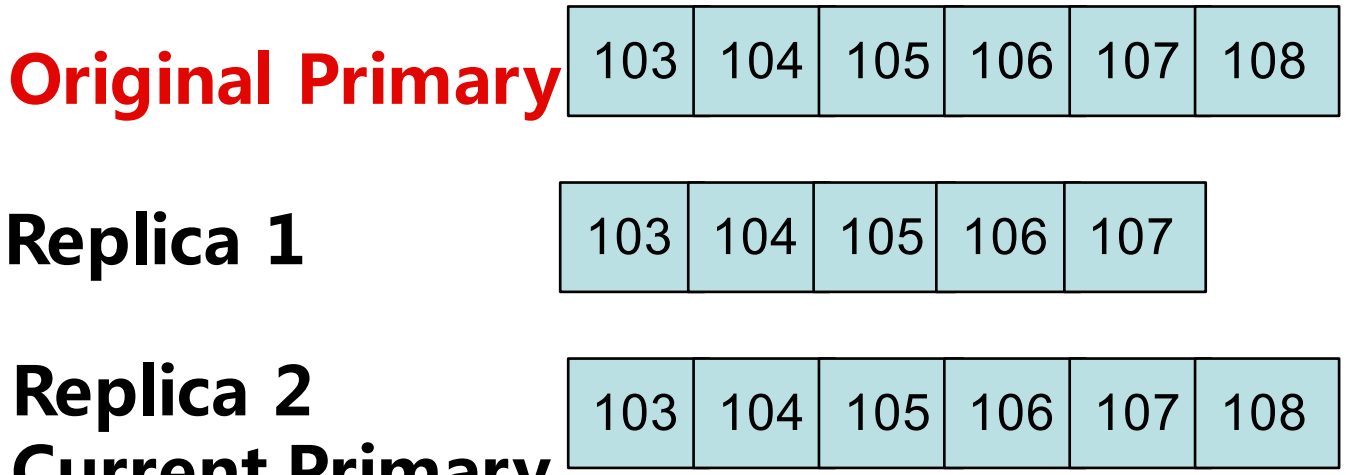
问题一

- 各个节点的元数据虽然都是从Master获取，但是时间有先后，在某个时间可能是不一致的
  - Client可能连接到的是一个旧的primary，然后读取的是旧数据



问题二

- 新的Primary产生后，各个副本之间没有做数据检查，副本接受到新数据后也没有做数据检查

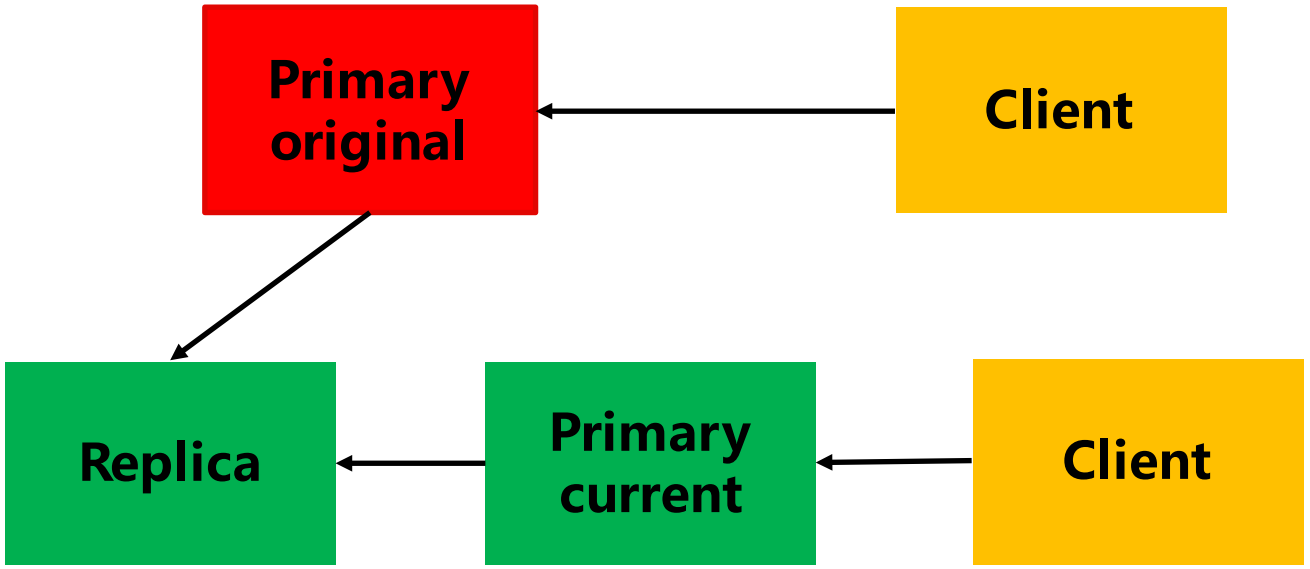


Log号为108的数据，在新Primary上还有，但是当Replica2挂掉，Replica1被选为主时，这条数据又看不见了，从客户端来看就是“丢数据”

Baidu 百度

问题三

- Replica上没有Fencing机制，不能拒绝处于“假死”状态的Primary的写请求
  - Replica和current primary的数据是不一致的



Bai du 百度

Raft的问题

- Raft
  - Raft必须在quorum时工作
    - 为了节省存储空间，一些服务是两副本，要求主从复制即可，只要有一个副本可以工作即可
  - 需要依赖心跳包来保持Leader Lease
    - 心跳包的数量随着ReplicaGroup的数量增长
  - Leader的选择是随机的
    - 不一定能保证Leader分布均匀，可控读写都从Leader走，容易导致一个节点压力过大
    - 需要在Leader选举时做一些策略
- 现状
  - 已经存在了一个可靠的Master
  - 可能存在更简单的一致性算法

Baidu 百度

一致性设计的关键点

- 数据存储一致性
  - 选择一个数据完整的副本作为Leader（Primary）
    - 包含所有已经通知client写入成功的数据
  - 在发生Leader切换时，Leader需要确保Follower（Replica）在接收新的写入请求之前，数据跟Leader是一致的
  - Follower要按照顺序接收Primary的写入消息
  - Follower需要能够拒绝其他不合法的Leader的写入请求
- 数据读取一致性
  - 用户写入数据后，再次读取时能够读到最新的数据

Baidu 百度

算法的思路—Leader选举

- 由Master来指派Leader，而不是选举
  - 当副本数小于quorum时，仍然能够工作
- 选择方式：
  - 根据balance策略，确保Leader在各个节点上的分布比较均衡
  - 从一个副本组中选择一个处于“健康”状态的副本作为Leader
    - 健康状态的副本上存储着所有已经通知Client写入成功的数据
      - 在原有Leader Commit之前一定是先发送了Log，如果Log写入失败，那么就会被移除，不会进入Commit阶段，所以如果这个副本是健康的，那么它一定存有commit之前的log。
    - 健康状态：上一个副本组状态下，正常的节点

Baidu 百度

算法的思路—Leader工作流程

- 建立副本组内Leader-Follower关系
  - 向所有的follower发送fencing消息
    - 确保follower不会接受别的leader发送的消息
  - 将所有follower上的数据做数据恢复
    - 保证在新的数据写入之前，所有健康的副本上的数据是一致的
    - 日志同步 & 数据文件同步
  - 恢复完数据后将follower的状态修改为“健康”
- 接受Client的写入
  - 按照Client的写入顺序，将log向各个处于“健康”状态的节点
  - 处于数据恢复状态的节点不发送数据
  - 发送不成功的节点，需要从副本组中移除
- Master根据meta_version判断Leader的移除或者修改请求是否合理

Baidu 百度

算法的思路—Lease机制

- Leader
  - Leader定时向Master申请Lease
  - 当Lease过期后，Leader不能继续提供读写请求
  - Master只有Lease过期后才能指定新的Leader

Baidu 百度

Elasticsearch的改进

三、与Imapla整合

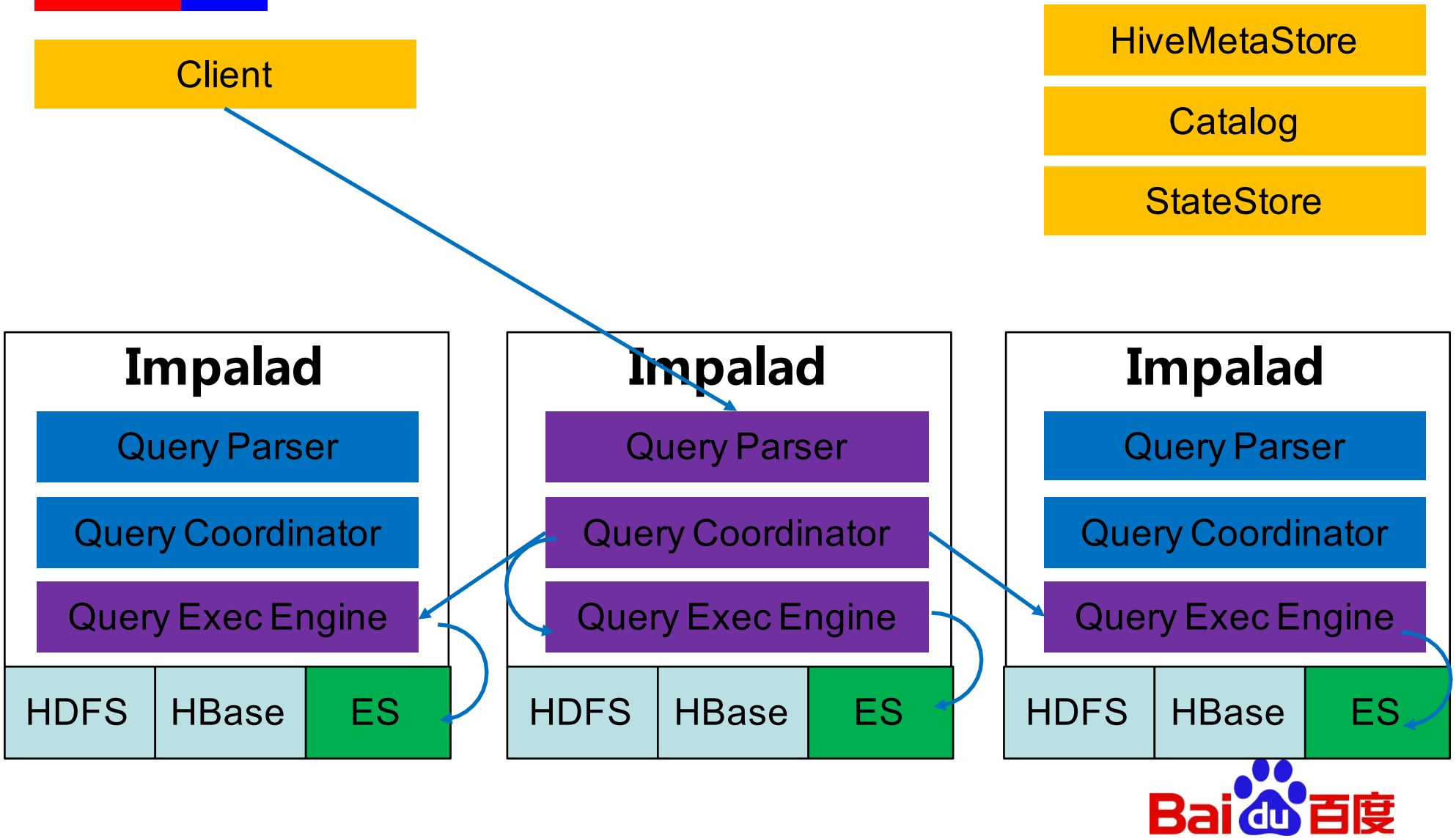
Baidu 百度

Elasticsearch现状

- ES的特点
  - 适合搜索
  - 有一定的分析能力
- 缺陷
  - JSON方式的查询语言
  - 缺少JOIN能力
- 现有的方案
  - 大多都是简单的SQL转Rest解析
  - 对现有的SQL工具支持比较弱

Baidu 百度

Elasticsearch现状



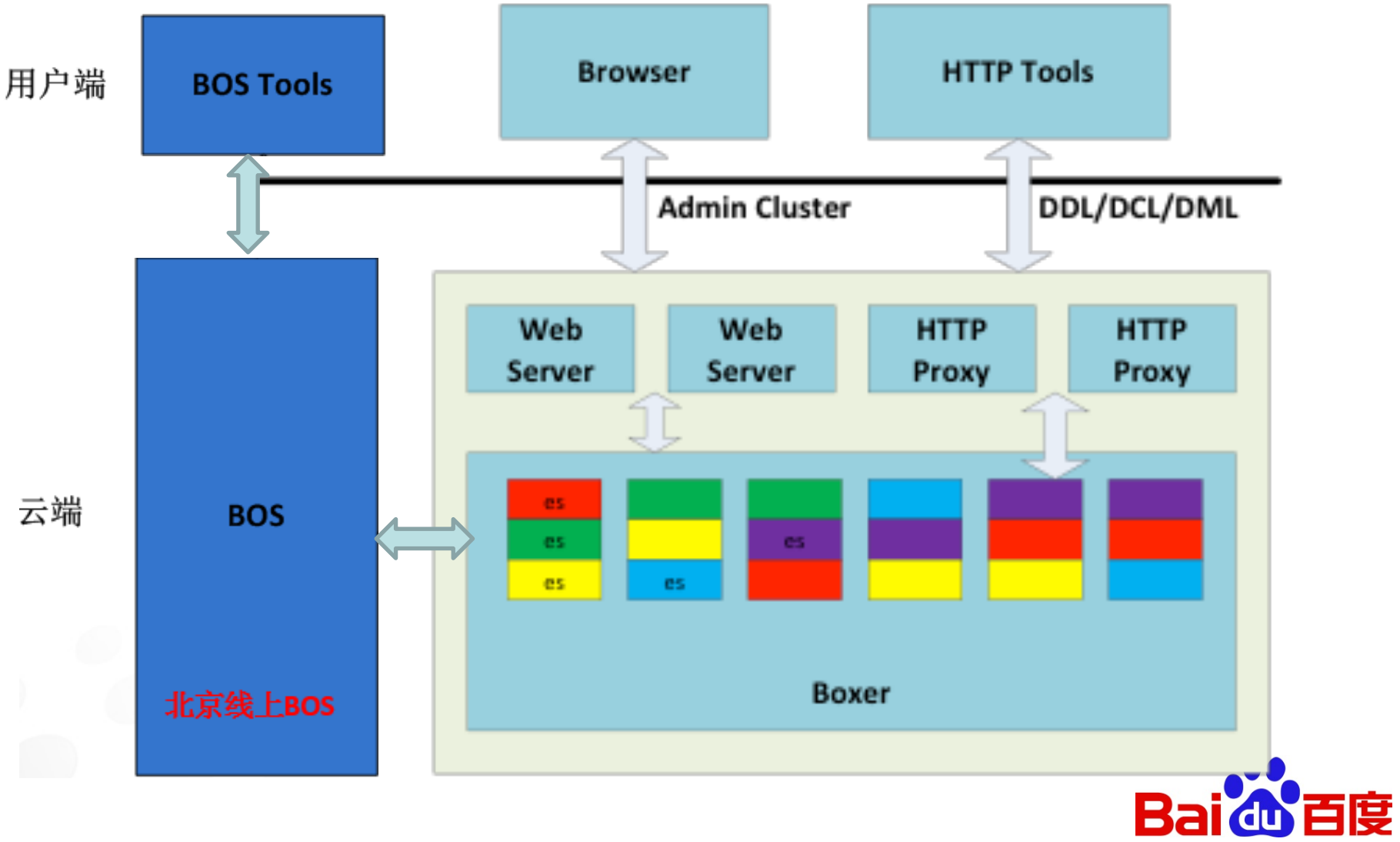
Baidu 百度

目标

- Elasticsearch
  - 分布式存储层
  - 利用索引完成数据的过滤
- Impala
  - 查询层
  - 完成SQL解析、JOIN、聚合等

Baidu 百度

ES实践-开放云



Baidu 百度

ES实践-开放云

- Palo 云服务
  - http://bce.baidu.com/product/palo.html
- Elasticsearch 云服务
  - http://bce.baidu.com/product/bes.html
- 如有兴趣，请联系 palo-rd@baidu.com

Baidu 百度

谢谢大家

Baidu 百度