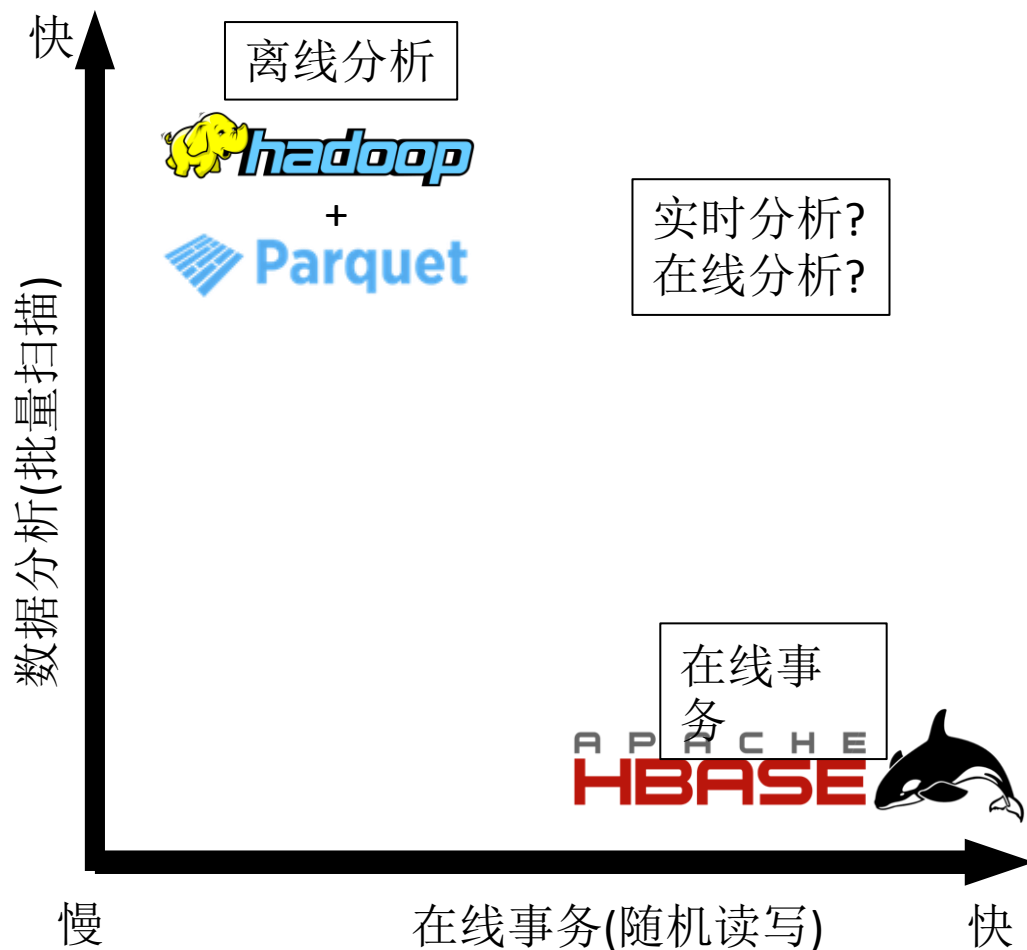


使用Kudu搭建OLAP服务

常冰琳

小米工程师 Kudu PPMC & Committer

- 现状
- Kudu 目标和设计
- OLAP 服务架构
- 性能测试



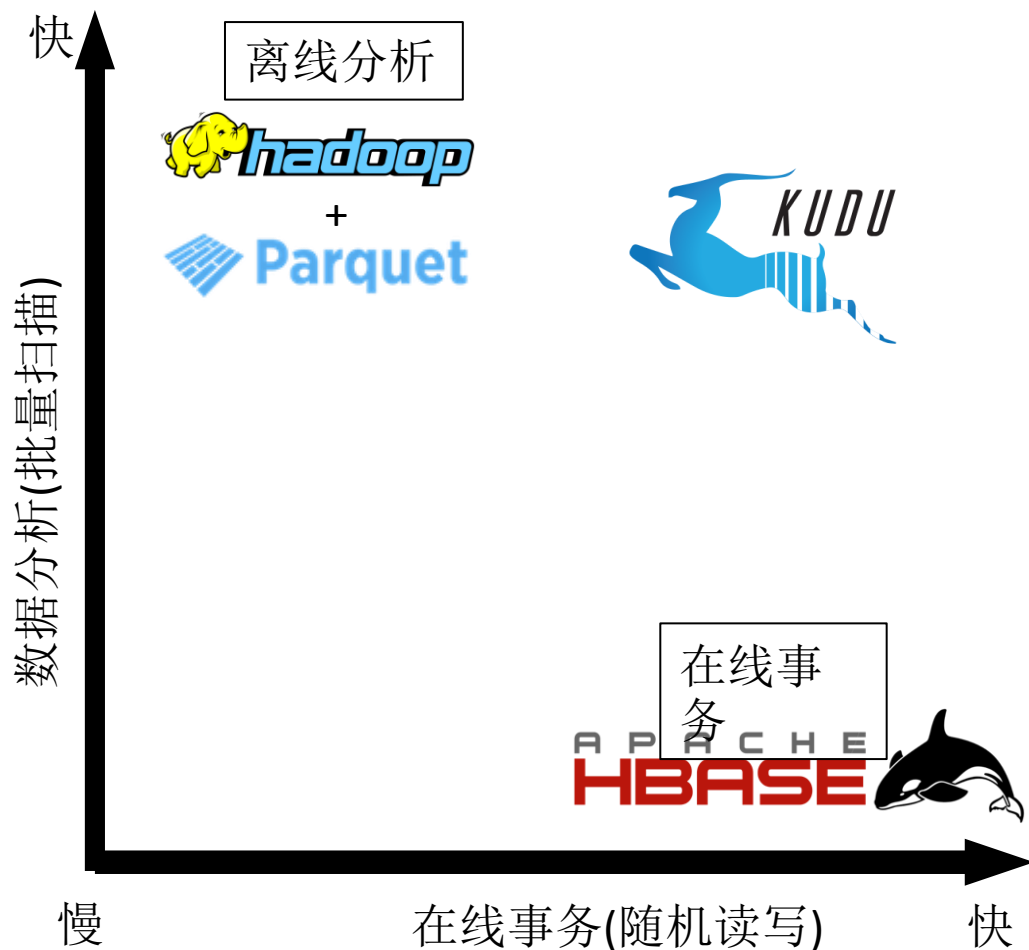
HDFS

批量写入
顺序扫描
批量离线分析

HBase

随机读写
高可用性
在线事务处理

如何能够**实时分析**在线事务系统实时产生的数据?
如何能够提供**在线分析**服务?



支持随机读写

- 接近HBase的吞吐性能
- 更低的延迟: 1ms (SSD)

高性能的顺序读

- 接近HDFS的读性能
- 列存储, 接近Parquet性能

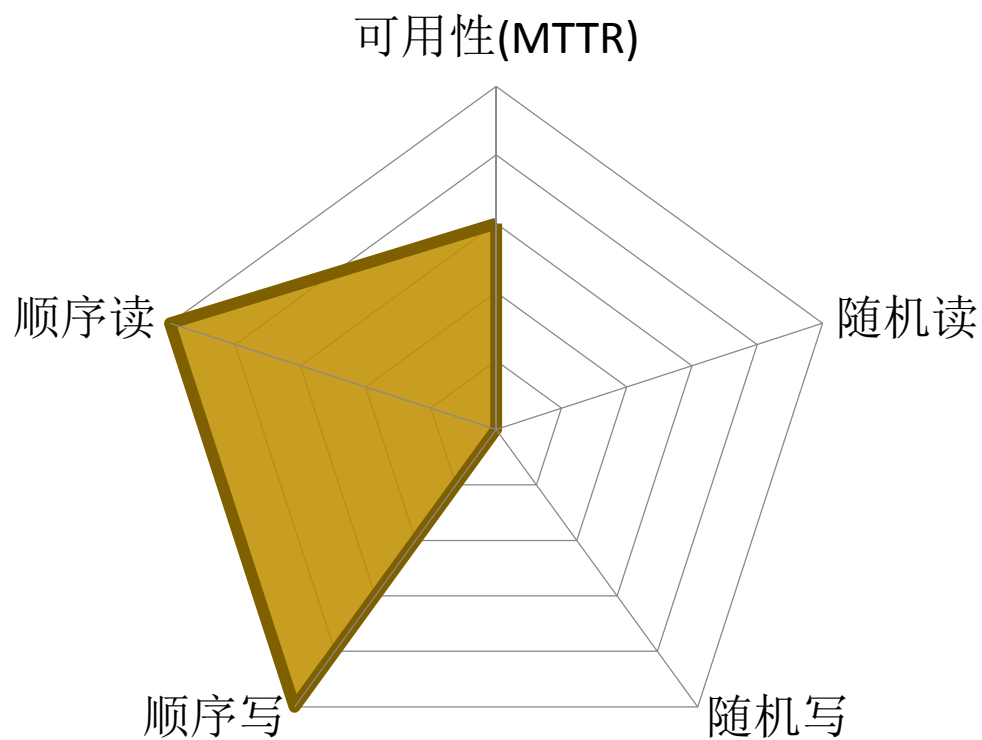
高可用

- Raft一致性协议

类关系数据库数据模型

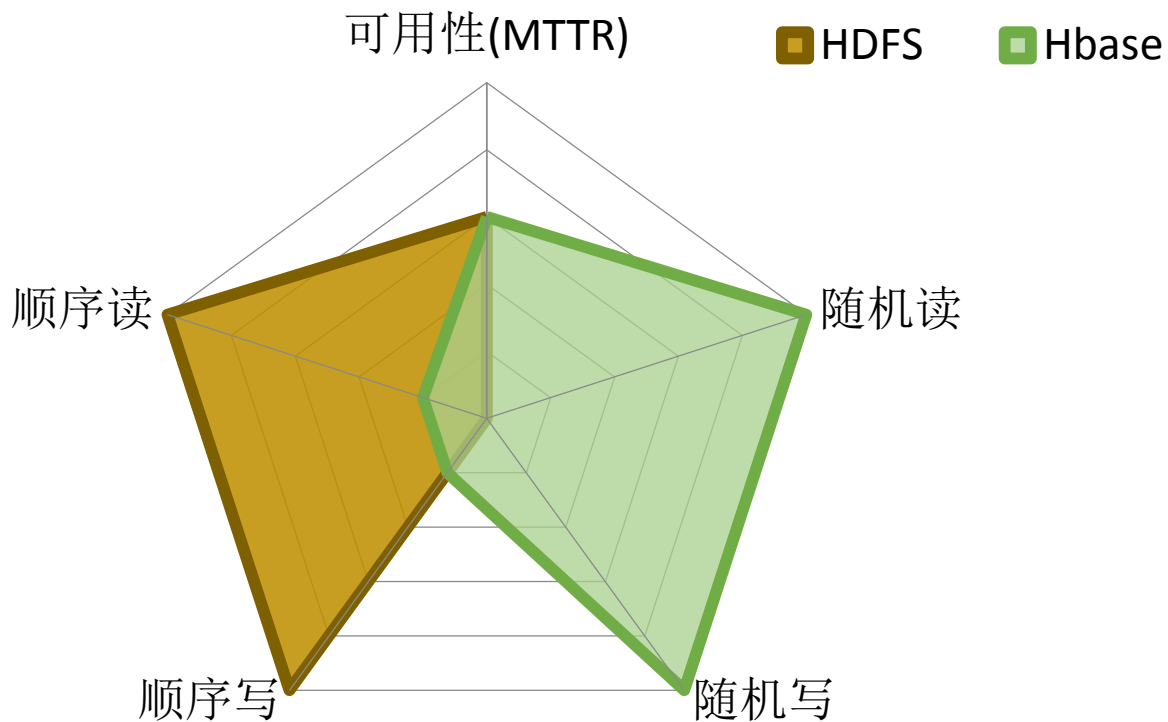
- 支持事务(单行)
- 计算引擎支持
Impala/Spark/Drill/...

- 存储越来越快
 - NAND Flash
 - 450K read/250K write IOPS
 - 3D Xpoint
 - 性能接近内存，但是可持久化
 - 性能比NAND还要快1000倍
- 内存越来越大
- CPU成为新瓶颈, 存储系统需要优化CPU
- 列存储应用在线事务系统也是可行的



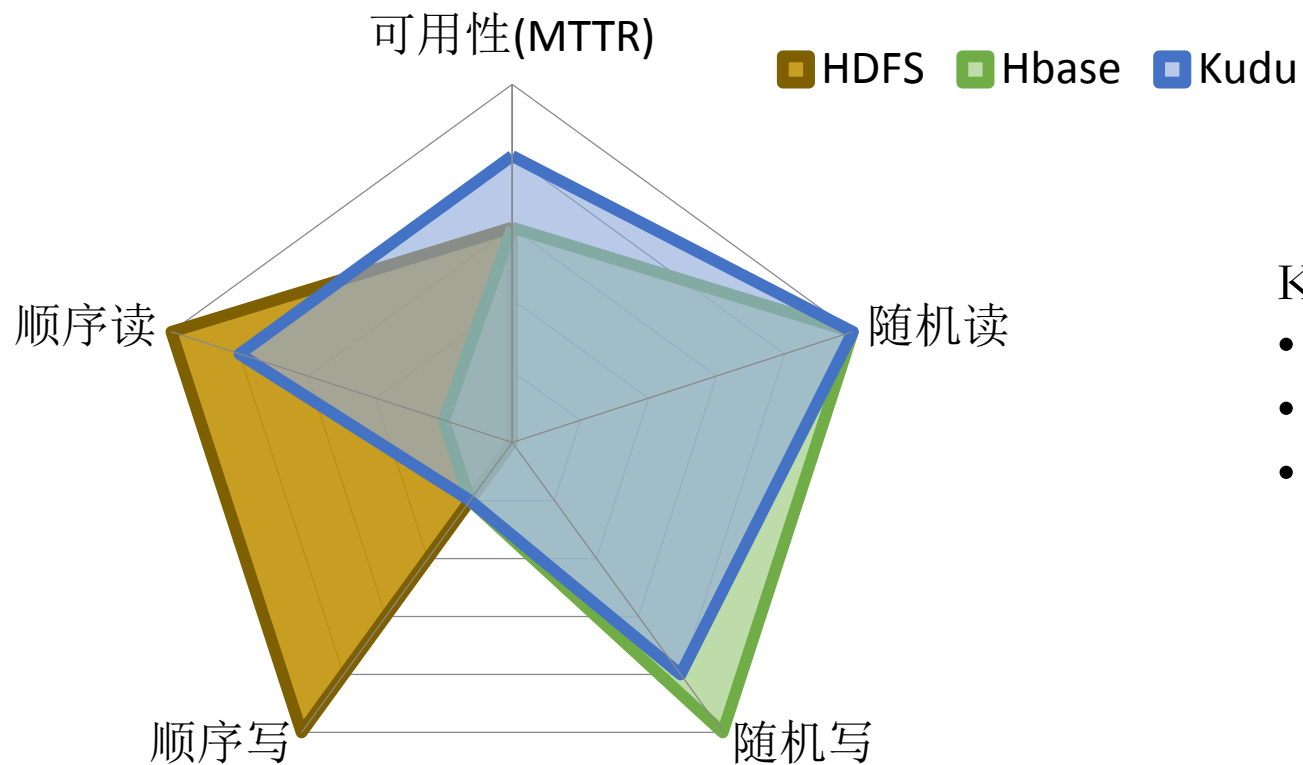
HDFS

- 离线数据存储
- ETL等预处理
- 复杂分析



HBase

- 在线事务处理
- 计算中的随机查询
- 流计算状态存储



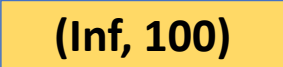
Kudu

- 在线分析
- 实时数据分析
- 事务数据分析

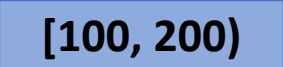
- 2012.10 Cloudera发起
- 2014.9 小米加入
- 2015.10 对外公布，发布Beta版本
- 2015.12 Apache Incubator
- 2015.12 小米生产环境上线
- 2016Q3 计划1.0 Release

- 数据模型：类似关系数据库表
 - 有限固定列, 支持alter table
 - 强类型(bool, int8/16/32/64, float/double, string etc.)
 - 主键索引（支持多列）
- NoSQL API
 - Insert/Update/Delete/Scan/Get*
 - C++/Java/Python
- Hadoop生态集成
 - Impala/Spark/Drill/MapReduce/Ibis(Python)

Range Partition



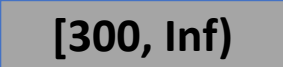
$(-\infty, 100)$



$[100, 200)$

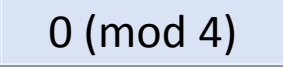


$[200, 300)$



$[300, \infty)$

Hash Partition



$0 \pmod{4}$



$1 \pmod{4}$

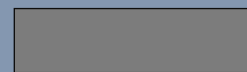
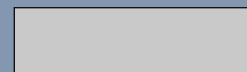
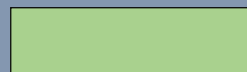
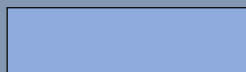


$2 \pmod{4}$

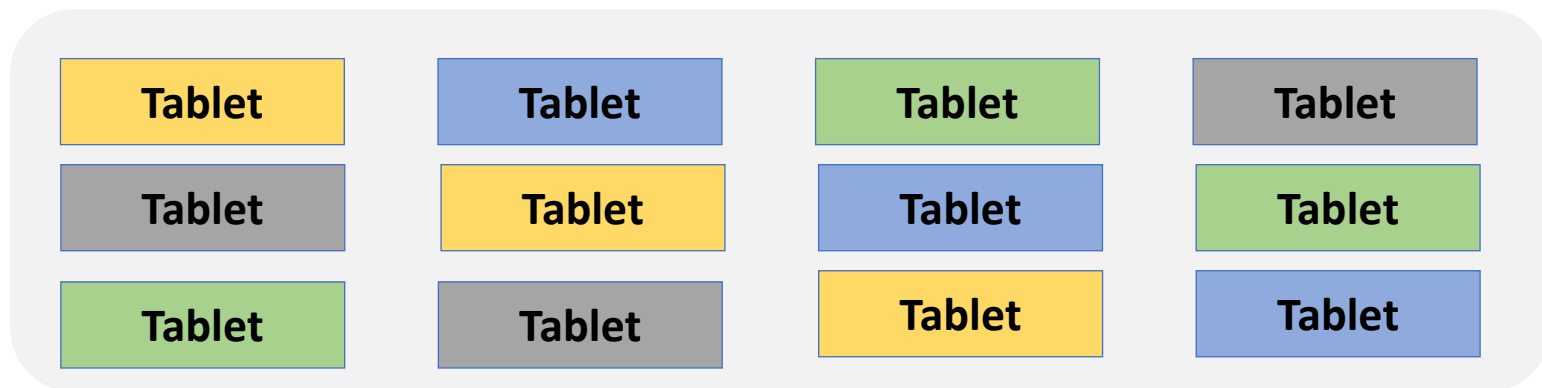


$3 \pmod{4}$

Range + Hash



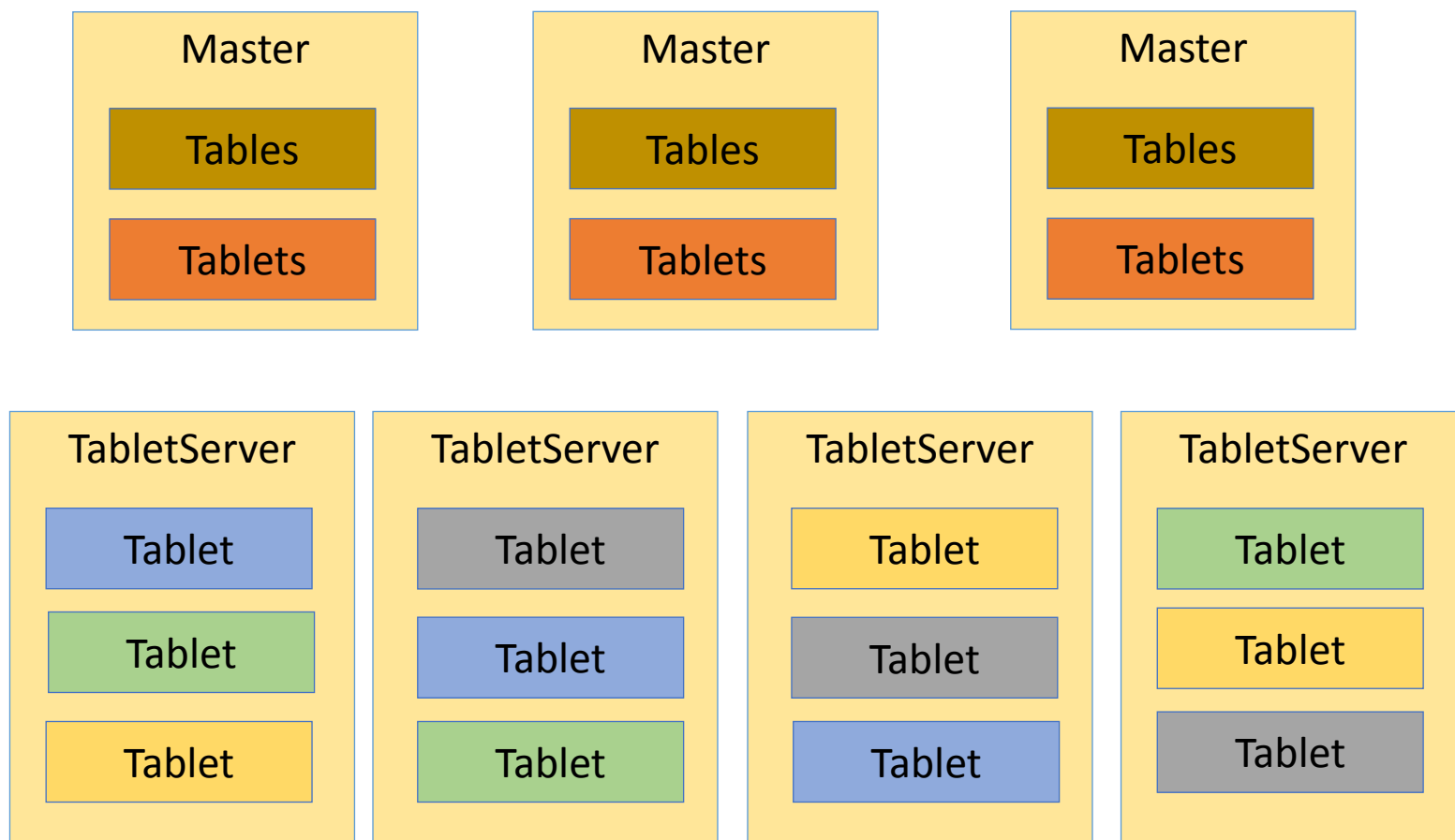
社区新进展: Range分区支持非全覆盖, 动态修改
适合日志和时间序列, 维护一个动态时间窗口
例如: 每天动态生成新分区, 删除几天前的老分区



每个Tablet维护多个(3或5)副本, 使用Raft一致性协议

- 强一致
- 低延迟: 多数写成功即可返回 (没有gc和大compaction)
- 高可用: 低MTTR, Leader失败检测+重选主时间短
- 允许从Follower进行Snapshot Read
- HybridTime支持外部一致性(类似Google Spanner)

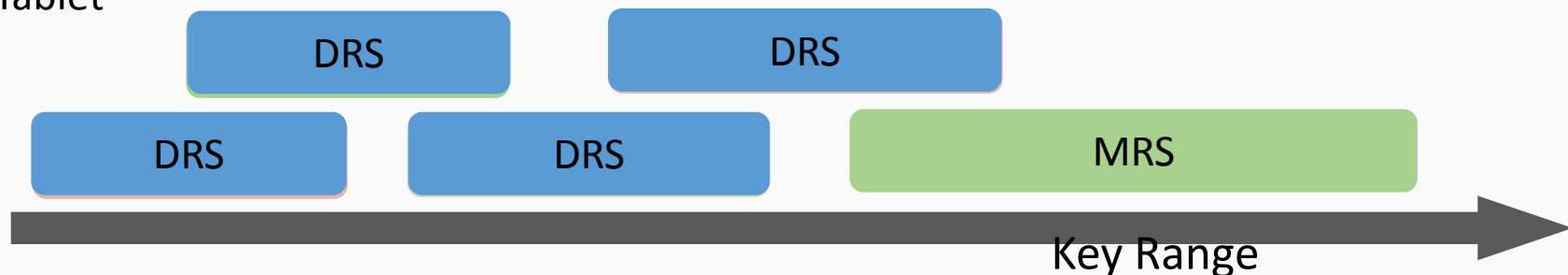
- Follower临时失败
 - Leader仍可取得多数，不影响读写
 - 5分钟内恢复，仍然可以重新加入
- Leader临时失败
 - 心跳停止，follow心跳超时后重新选主
 - 5分钟内恢复，变为follower
- 永久失败
 - 踢掉该follower，master重选一个replica
 - 从leader拷贝数据后，加入成为新的follower



- 类似TabletServer, Raft保证多副本一致性
- 管理两个元表(Tables, Tablets)
- Tables
 - 表的schema, parititon等元数据
- Tablets
 - 所有tablet的元信息,状态信息(tablet到tablet server映射)
- 所有元数据缓存在内存
 - 查询非常快($99.99^{\text{th}} < 1\text{ms}$)
- 客户端缓存
 - 减少与maser通信, 避免master压力过大

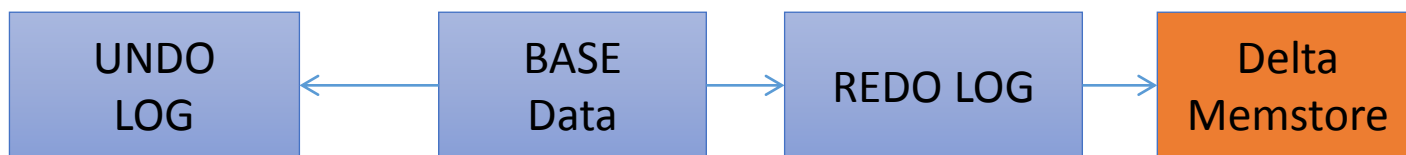
- 类似LSM(Hbase/BigTable), 但不完全相同
- 每个Tablet包含:
 - 一个MemoryRowSet(MRS)
 - 多个DiskRowSet(DRS), 列存储, 类似parquet
- Insert写入MRS, MRS满或者定时Flush为DRS

Tablet



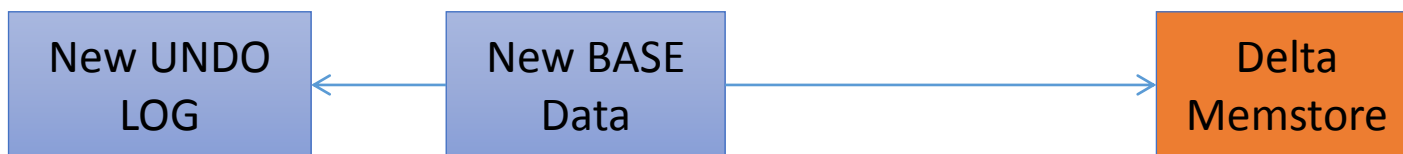
Before

读最新数据需要合并base和redo

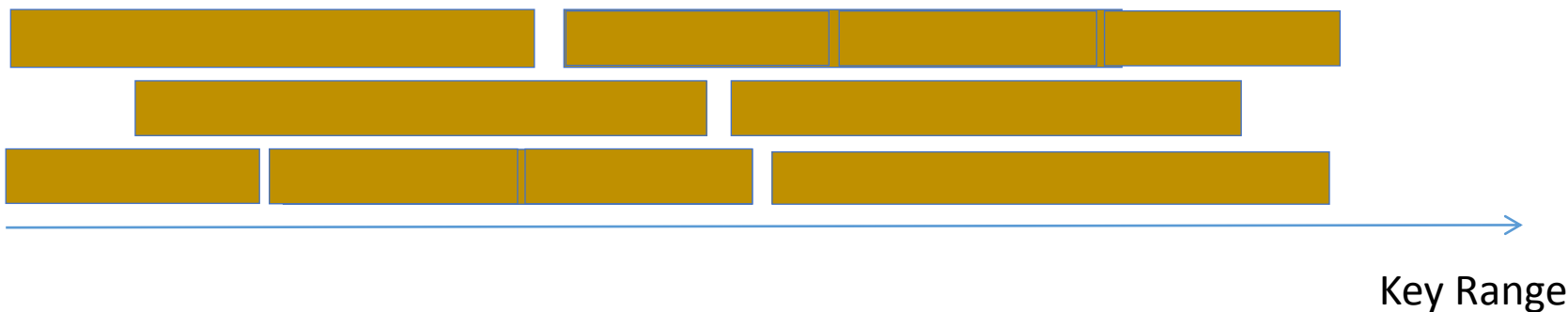


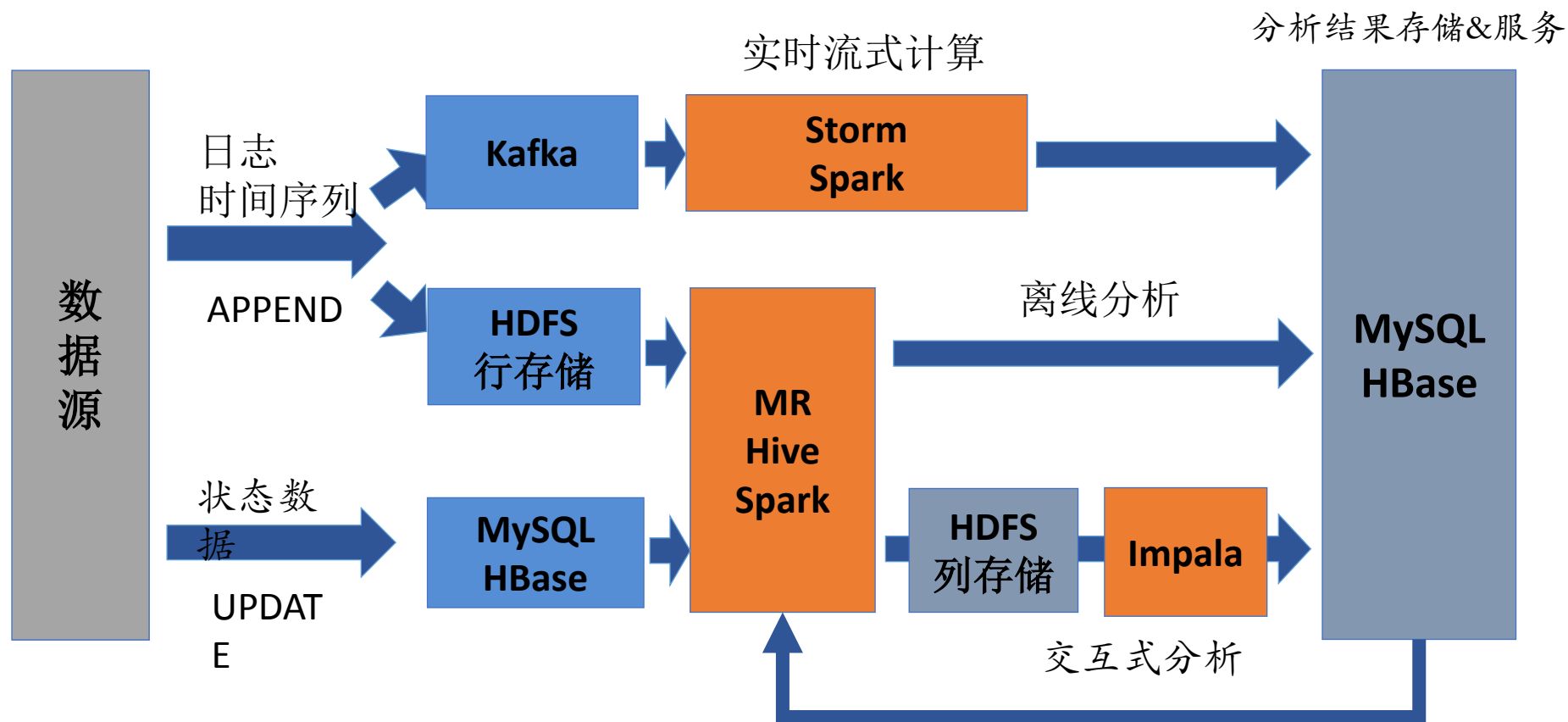
After

读最新数据仅需要读取base



- DRS 区间覆盖过多影响读写效率
- Compaction 减少区间覆盖
- 频繁地做小规模 Compaction, 不做大 Compaction
- 每次 Compaction 严格限制 IO 量(128MB)
- 对随机读写延迟影响很小
- 顺序写不会导致 Compaction





- 简化数据流，降低延迟
 - 不需要定时从行存储转化为列存储
 - 支持需要update的分析型应用
- 统一元数据和存储格式
 - 避免数据格式转化和不一致问题
- 分析结果存储和服务
 - 分析结果重复使用
 - 统一SQL管理

- 事务型存储，主键索引
 - 存储层支持去重，Exactly Once语义
 - 简化计算层容错处理
- 数据到达时间有延迟
 - 可按时间排序，简化或加速分析处理
 - 问题：Hive/MR/Spark需读取多余日志
 - 问题：流式计算窗口问题

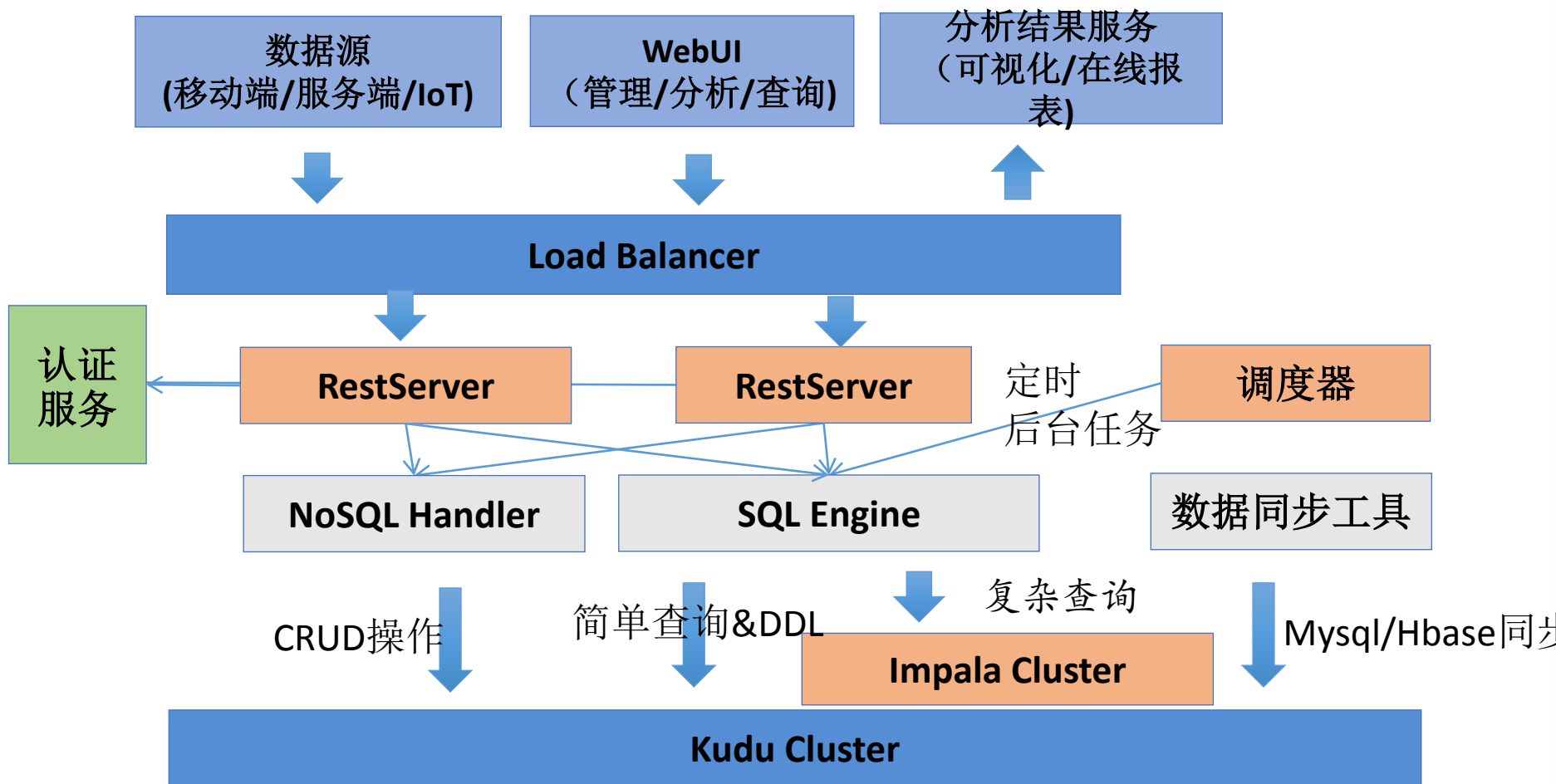
```
// 创建表(持久化数据流)
create table words (ts int key, seqid int key, word string);

// 实时数据插入
insert into words values (xxx, xxx, xxx);

// 实时查询最近两分钟top10 Word
select ts/60, word, count(*) as cnt from words
  where ts >= 1462856700 and ts < 1462856820
 group by ts/60
 order by cnt desc
 limit 10;

// 分析结果保存为Kudu表, 直接提供随机查询服务, 如在线报表
create table wordcount (ts int key, rank int key, word string, cnt int)
insert overwrite wordcount select ...
```

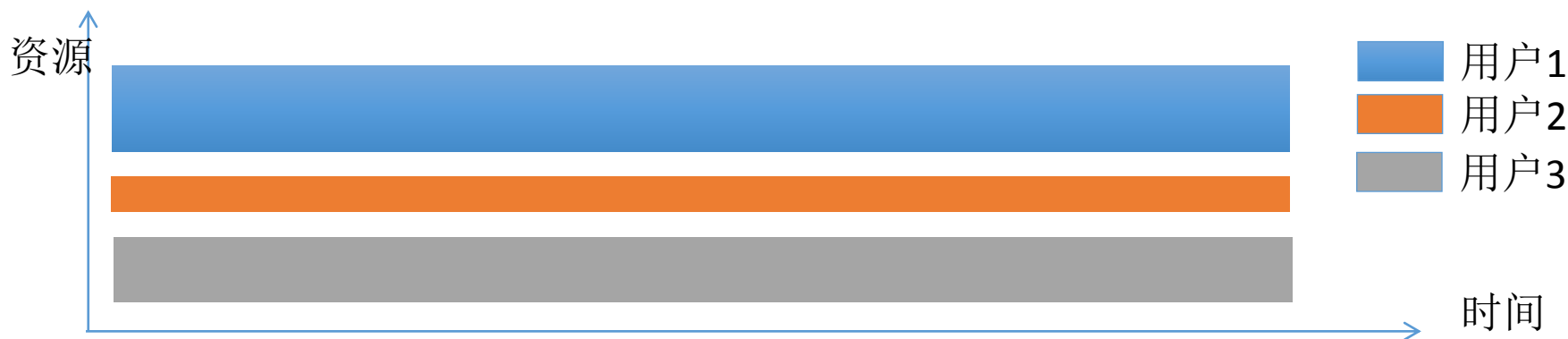
- ◆ 为生态链企业提供数据分析服务
- ◆ 像数据库一样简单易用（REST + SQL）
 - ◆ OLAP数据库包含部分OLTP功能
 - ◆ 导入/查询/管理/权限/Quota统一集中管理
- ◆ 良好的可扩展性
- ◆ 一定程度的隔离



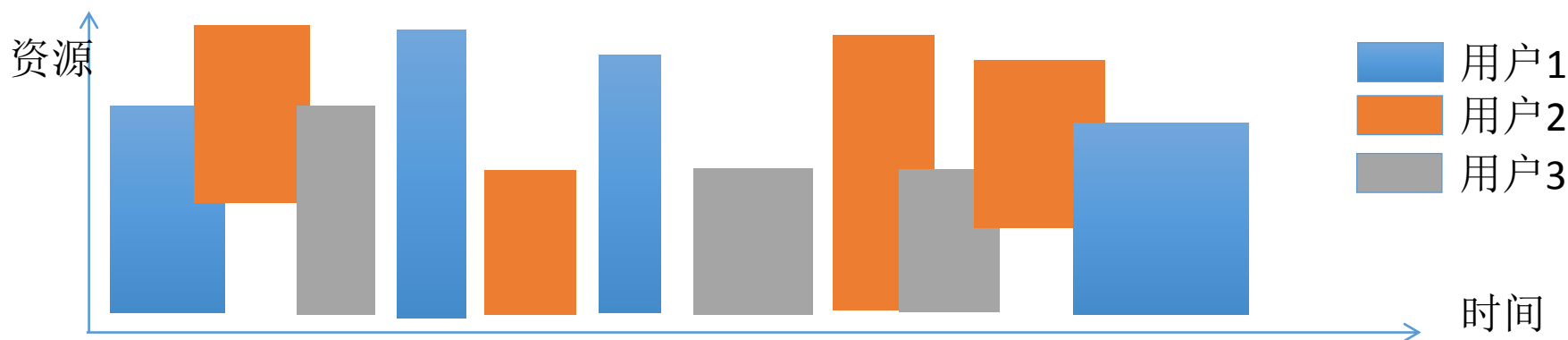
- 主要用于数据导入和结果服务
- Kudu API封装为REST服务
 - 通用性和易用性
 - 认证和权限验证
- 水平可扩展，高性能
- CRUD (Insert/Update/Delete/Get)
 - 同步写：低延迟
 - 批量写：高吞吐

- 主要用于分析查询，数据和任务管理
- SQL封装为REST服务
- 对SQL请求做语法解析，分析查询类型
- 元数据管理、DDL、简单查询
 - 直接操作Kudu
- 复杂查询
 - 转发Impala
- 任务管理，扩展库，数据导入导出等功能
 - 添加SQL扩展语法

- NoSQL服务隔离
 - 写请求压力预期相对稳定
 - 每个分区写请求有固定Quota
 - 根据应用需求规划Partition数(类似AWS Kinesis)
- 实现
 - Kudu添加Partition级别Write Throttling
 - Token Bucket算法, 细粒度时间(0.1s)



- SQL查询服务隔离
 - 查询不稳定, 突发
 - 预期执行时间短, 但占用资源多
- 实现
 - 每个查询收集聚合Impala和Kudu资源占用
 - Token Bucket, 粗粒度时间(1min)
 - 限制大查询, 添加后台查询模式(类似BigQuery)

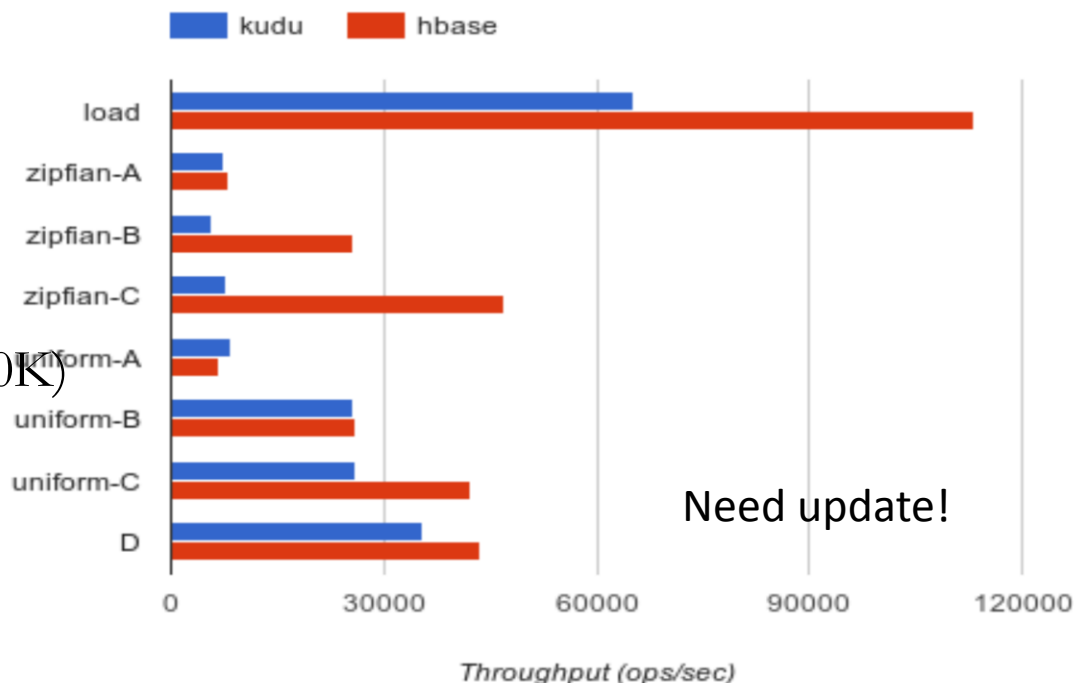


总结:

设计上牺牲写换取读

实际测试结果:

- 随机写差距不明显(20K-30K)
- 99th延迟控制很好(~5ms)
- 频繁更新性能比较差
- 随机读性能很好 (>50%)
- Scan性能>40倍
 - HBase ~ 150K cell/s
 - Kudu ~ 6M cell/s



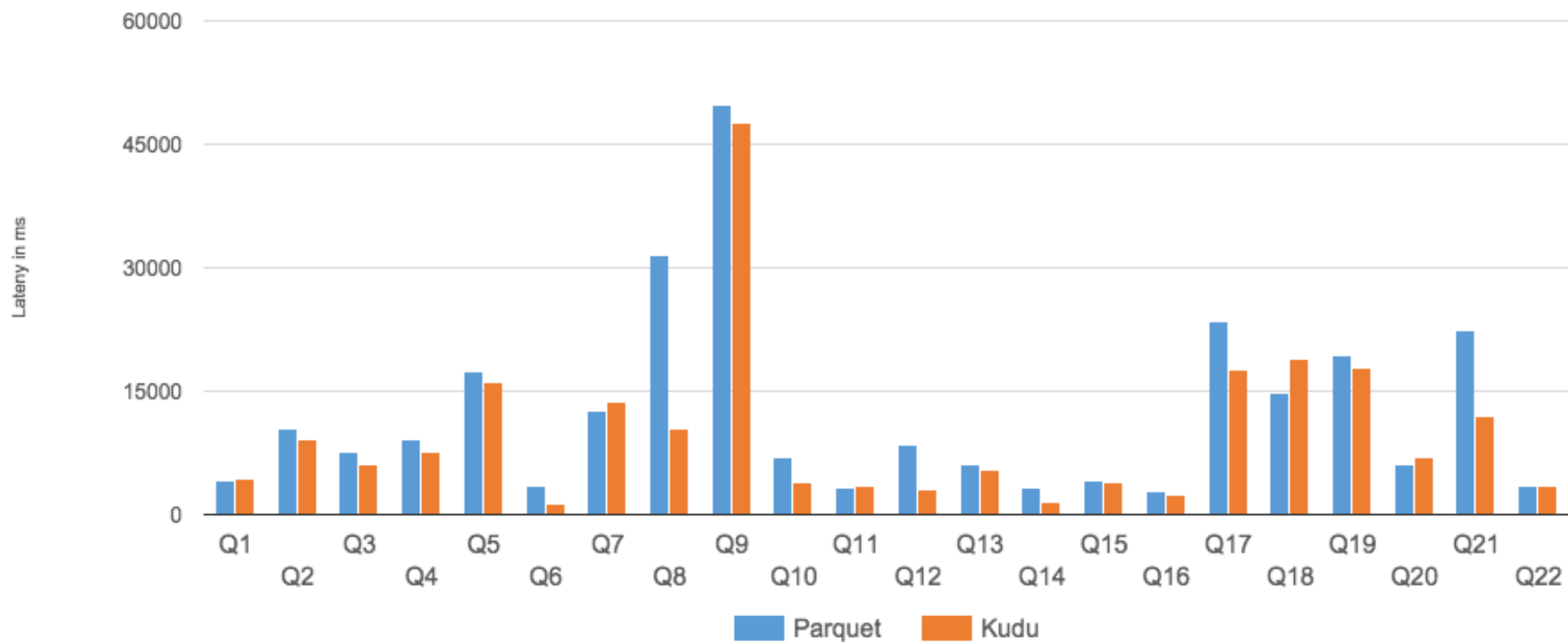
社区新进展:

RPC性能优化: 性能>5倍提升

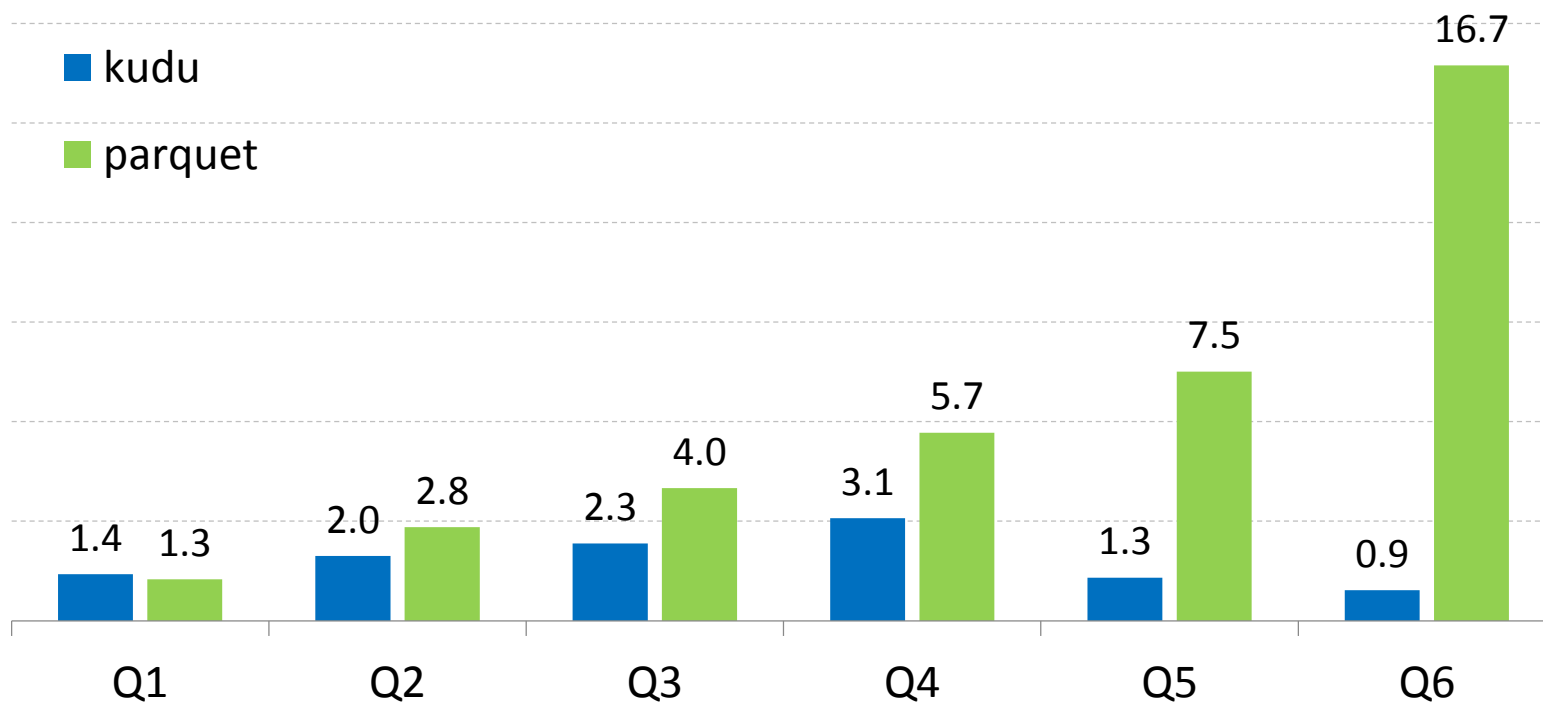
Get API (单节点800K QPS)

支持高性能随机读的应用, 例如: 计算结果服务
MultiGet, 支持Index Join

TPC-H SF 100 @75 nodes



小米应用，71节点，260亿记录



谢谢

getkudu.io

@常冰琳