Kyligence

Apache Kylin
实时OLAP实现

韩卿 | Apache Kylin PMC Chair
Luke.han@apache.org

关于...

- ✓韩卿 | Luke Han
  - -- Kyligence Inc 联合创始人 & CEO
  - -- Apache Kylin 联合创始人 & PMC Chair
  - -- ASF member & VP
  - -- Mentor of Apache RocktMQ & Apache Weex
  - -- Microsoft MVP
  - -- 前eBay全球分析架构部大数据产品负责人

关于Apache Kylin

- ✓首个来自中国的Apache顶级开源项目
- ✓连续两年荣获InfoWorld“ 最佳开源大数据工具奖”，今年更是与Google TensorFlow一起获得该奖。
- ✓社区活跃，2年内赢得超过100多家公司使用

Apache Kylin代表了亚洲国家，特别是中国，在开源社区中越来越高的参与度。

—Ted Dunning, Apache 孵化器副总裁

Apache Kylin 历史

项目开始 *Sep 2013*

正式开源 *Oct 2014*

加入Apache 孵化器项目 *Nov 2014*

InfoWorld: Bossie Award **最佳开源大数据工具奖** *Sep 2015*

毕业成为 Apache顶级项目 *Nov 2015*

Kyligence公司创建 *Mar 2016*

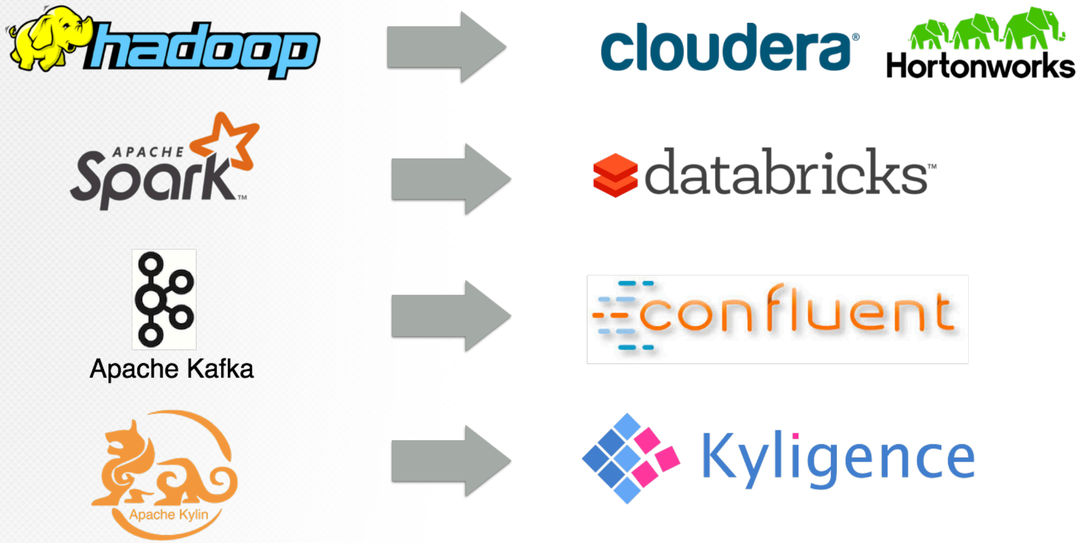
商业版KAP发布 *Aug 2016*

InfoWorld: Bossie Award **最佳开源大数据工具奖** *Sep 2016*

Apache Kylin 用户

关于Kyligence

- Kyligence’s vision is to unleash big data productivity for everyone's analytics needs.
- The company was founded by the team who created Apache Kylin™, a top open source OLAP engine built for interactive analytics at petabyte-scale data on Hadoop. Kyligence is the primary contributor to the open source Kylin project globally.
- Kyligence provides a leading intelligent data platform to simplify big data analytics from on-premises to cloud.



Kyligence

现状：Hadoop的生产力未得到充分释放

企业通常采用**混搭架构**构建数据分析生态系统

Hadoop的瓶颈

查询性能差

技术门槛高

人员不好招

混搭架构的问题

架构复杂

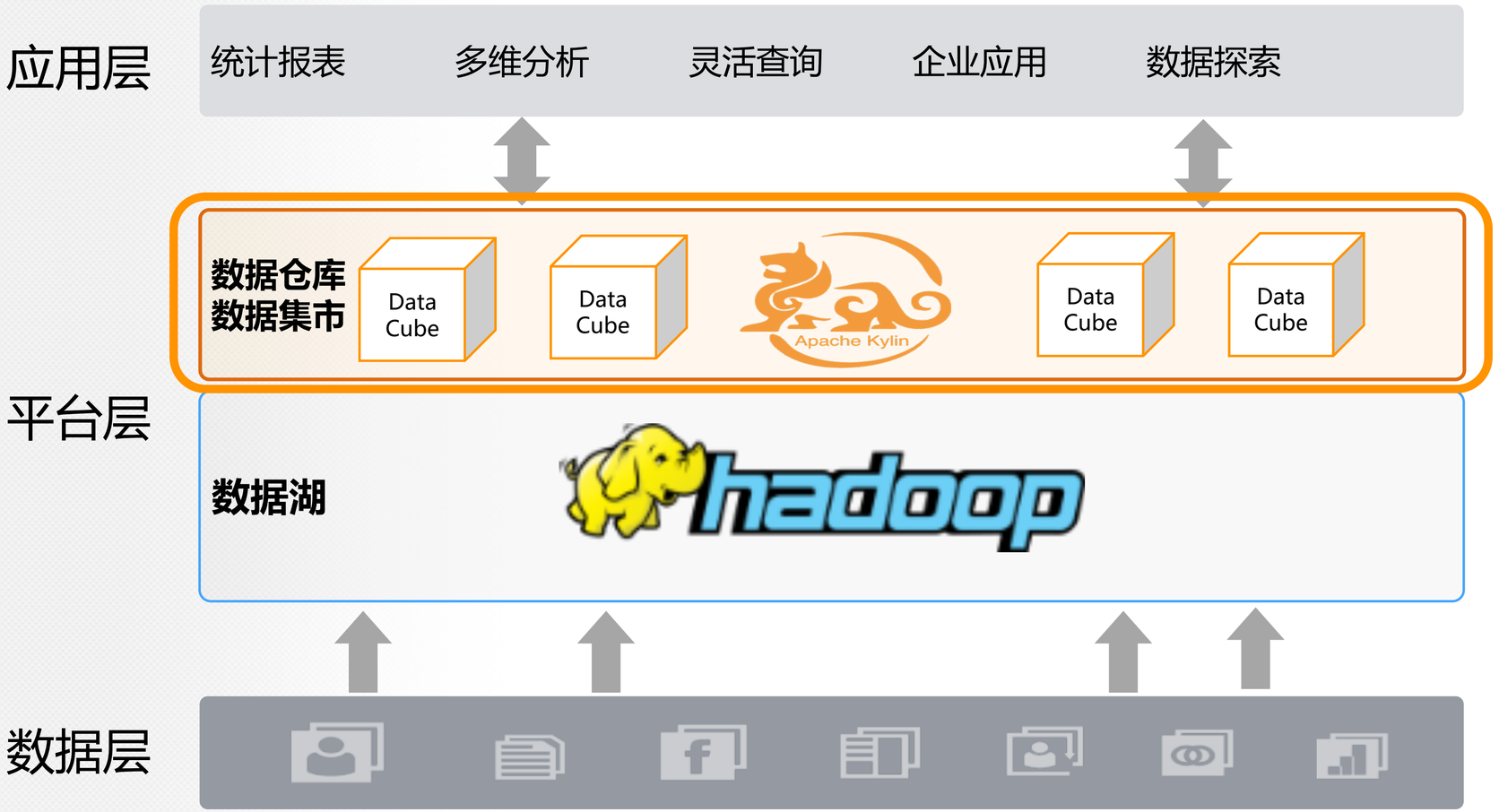
运维难度大

数据时效性差

数据冗余严重

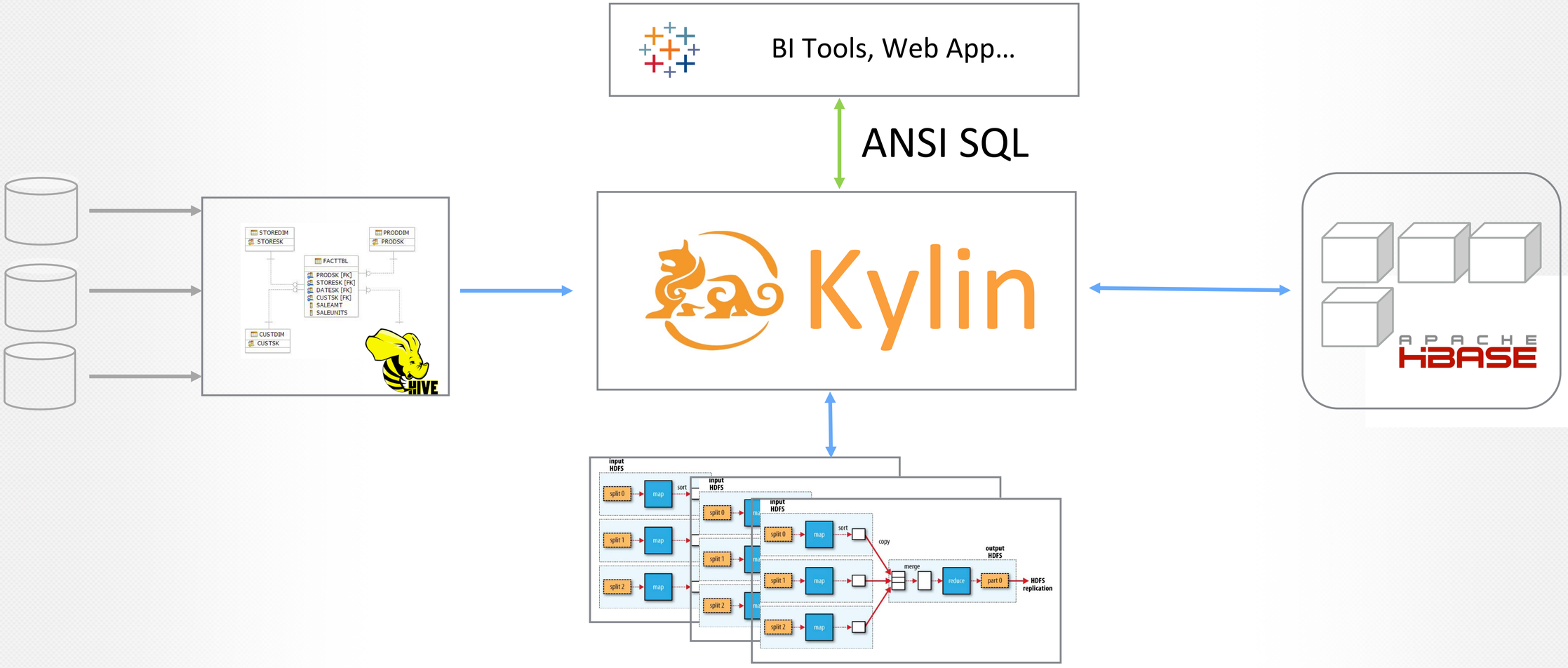
总体拥有成本高

Kylin：搭建用户和大数据之间的桥梁



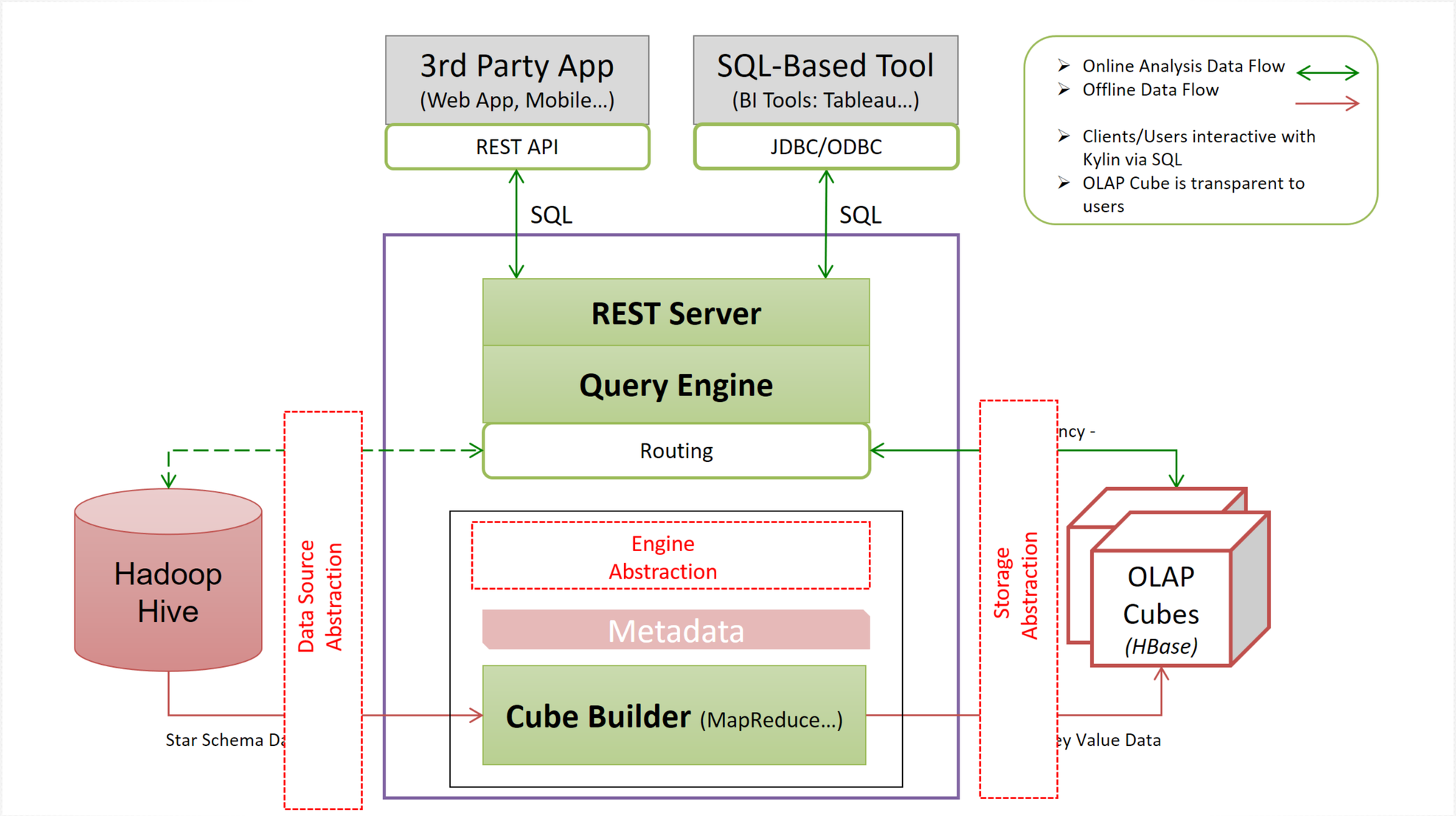
- **超高性能**：满足传统数据分析应用的时效性要求
- **高并发**：满足大数据时代爆发的数据分析需求
- **标准SQL**：降低技术门槛，屏蔽复杂的技术接口

Apache Kylin的架构

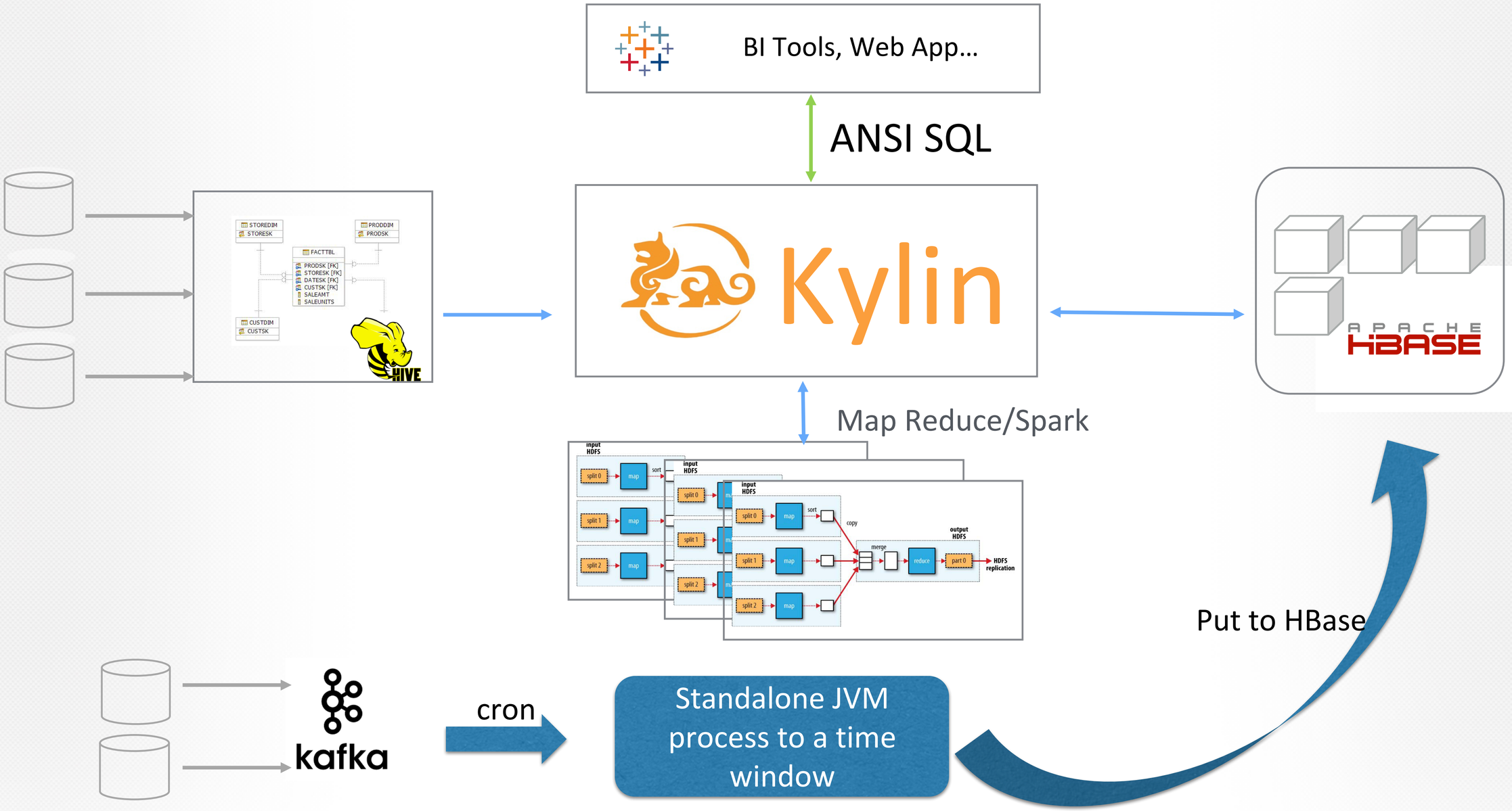


Map Reduce

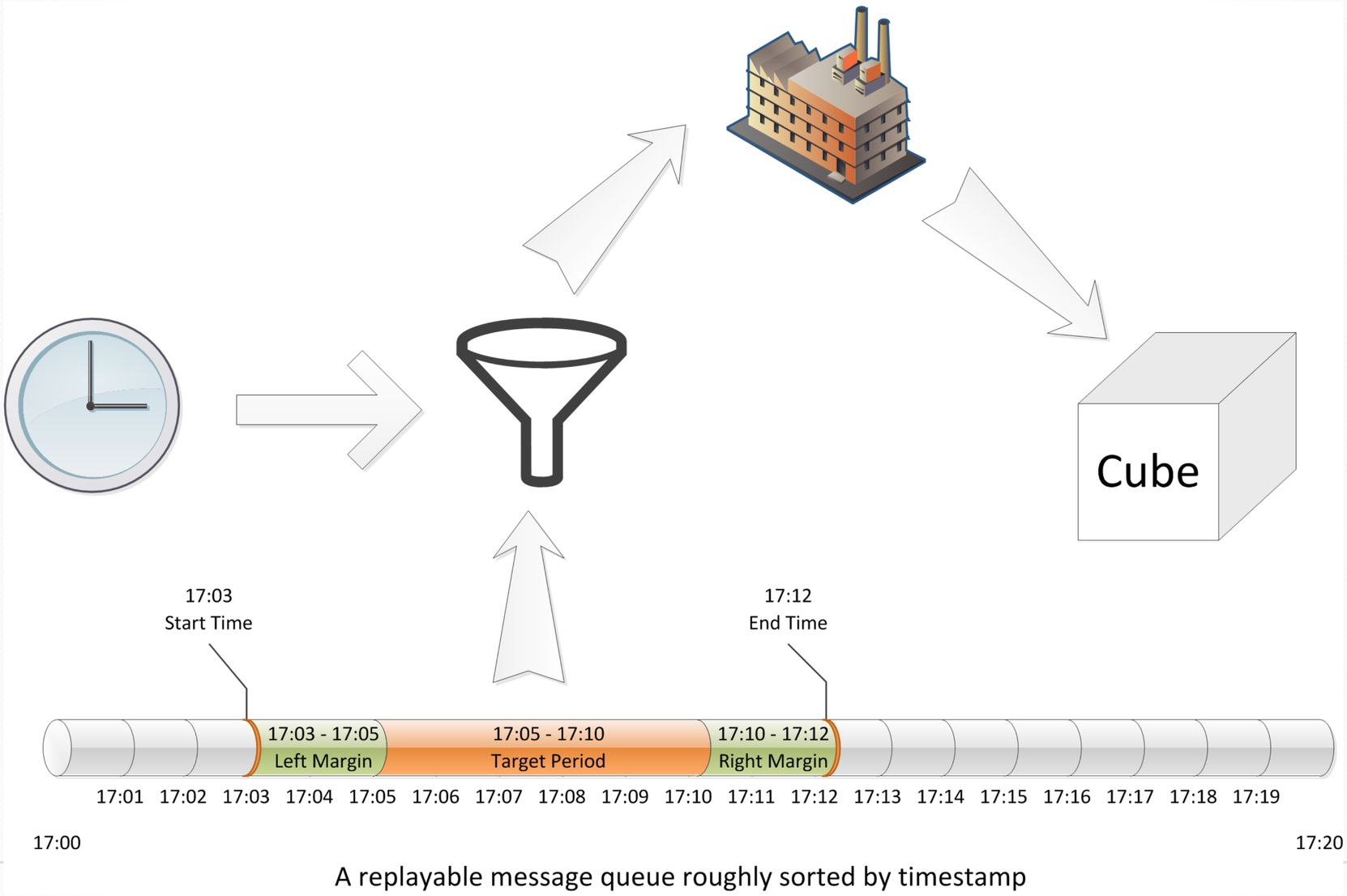
V1.5 核心组件解耦



V1.5的Streaming Cubing

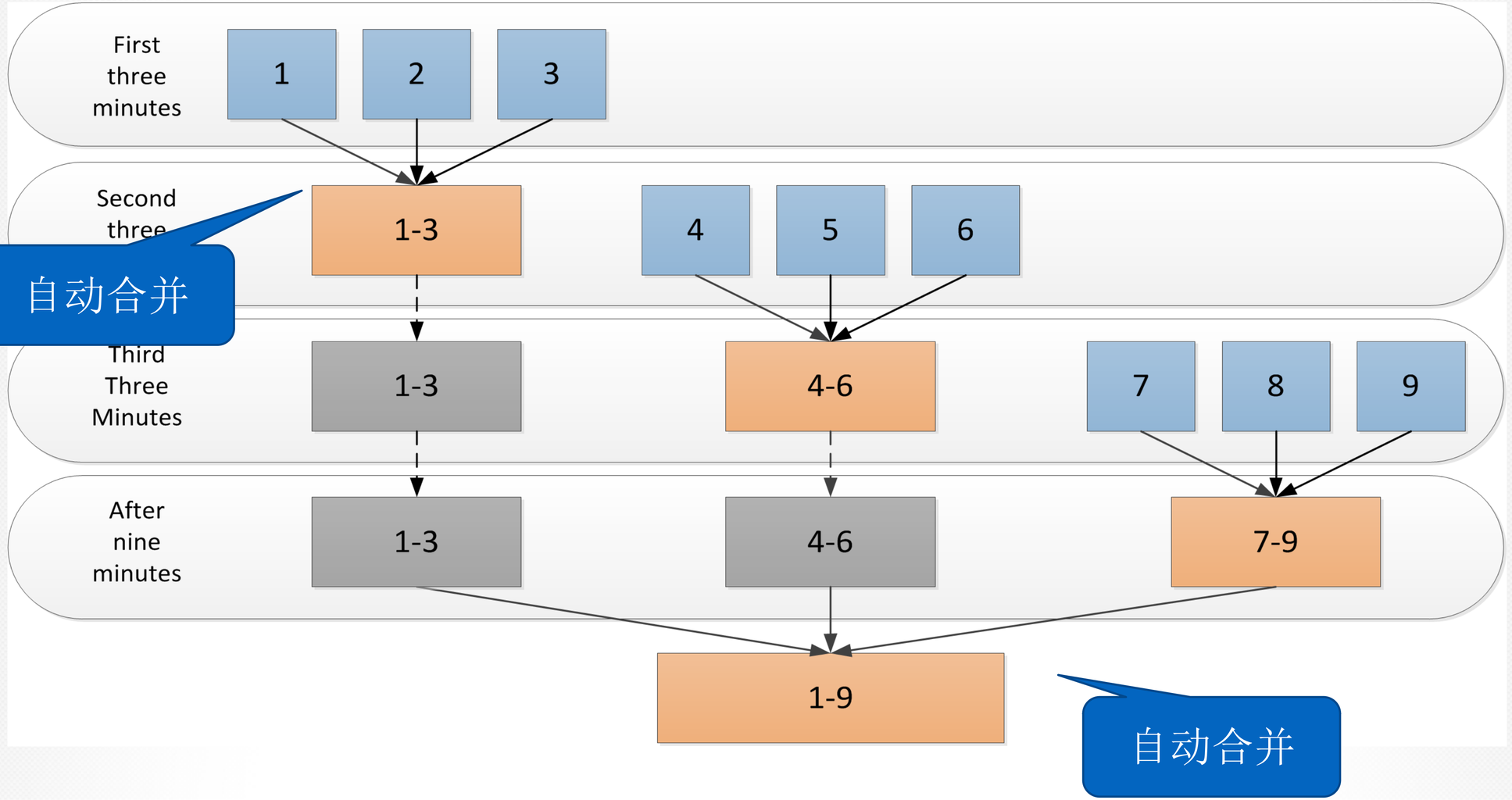


V1.5: 流数据分段 – 使用时间近似寻找



A replayable message queue roughly sorted by timestamp

小碎片逐渐合并成大Segment

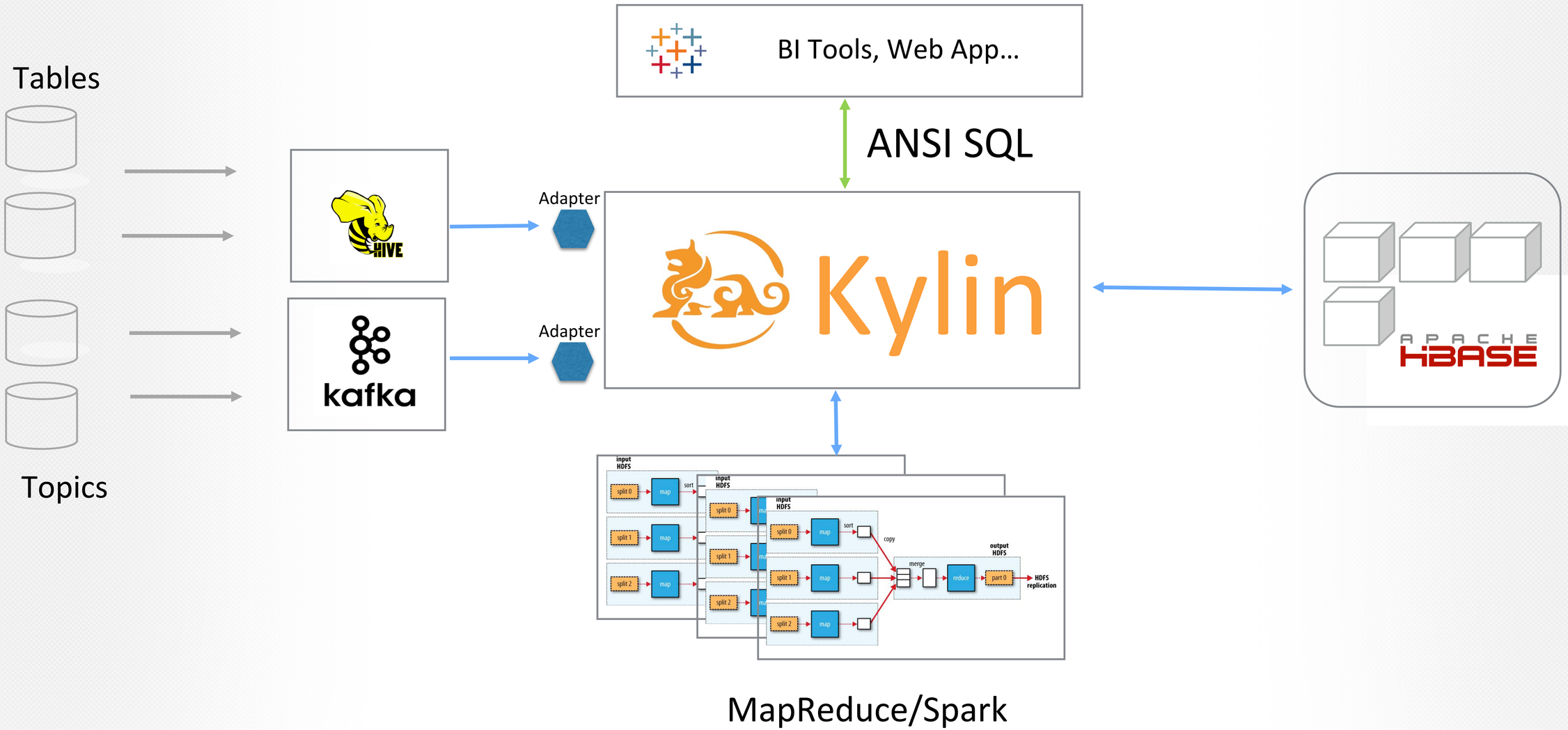


收获及缺陷

- 解决了从无到有的问题

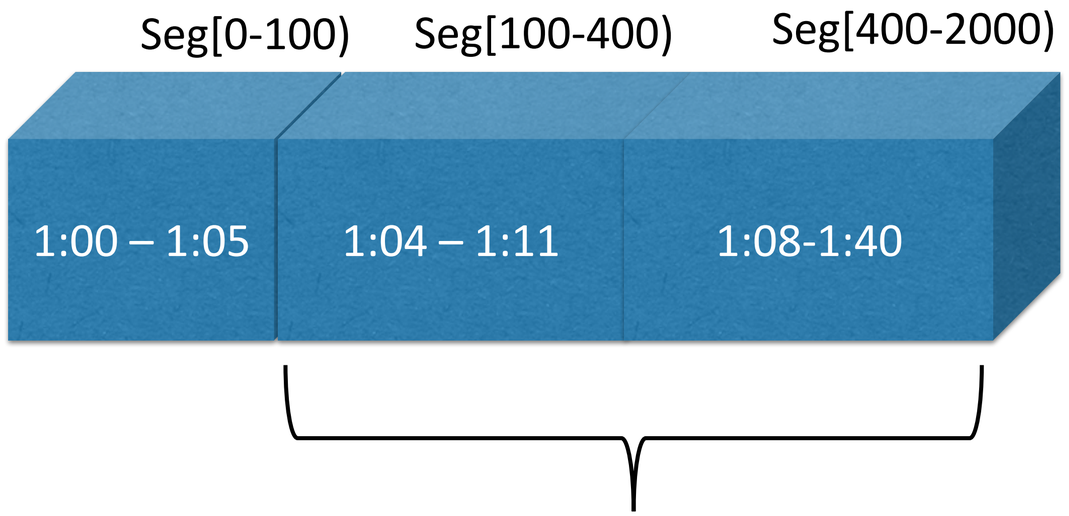
- 构建不能自动伸缩
- 近似二分查找会丢失数据
- 构建任务难以监控
- 错误恢复困难
- 整体运维成本高

v1.6 Streaming：插拔架构下的完美实现



Cube Segment按offset切分

- Segment按offset切分, 不能有重合
- 对于Hive数据源，分区时间做segment的offset，保证了向前的兼容
- 对于Kafka数据源，各个partition的offset之合，作为segment的offset
- Segment之间允许有时间值重合
- 使用offset，确保了数据一致性和查询准确性



When query for time 1:10， Kylin will scan all Segments contain this time

完全可伸缩

- 大部分代码重用既有构建引擎，易于维护
- 从几千条到几亿条数据，一次均可轻松构建
- 可随时暂停 / 恢复，可任意更改构建频率

其它改进

- 自动寻找开始和结束的offset
- 支持嵌套式JSON消息
- 支持自定义时间格式
- 允许多segment并行构建/合并
- 通过Rest API触发

Kylin Streaming先决条件

- Kafka版本0.10或以上
- Kafka消息为JSON格式，必须有一个timestamp字段
- 模型仅一张表（不支持多个Topic的join，也不支持topic与Hive表的join）

kafka

Apache Kylin

使用Kylin分析Twitter消息

- Demo环境： AWS 5台vm的Hadoop集群， Kafka 3个broker
- 测试数据：Twitter sample数据流，使用Twitter SDK反复获取，写入本地两个Kafka topic中;
- Topic “TwitterSample” 存储原始消息；Topic “TWITTER_TAG_STREAM2”存储从原始消息中抽出的HASH_TAG信息；

Twitter消息

./bin/kafka-console-consumer.sh --zookeeper 10.0.0.205:2181 --bootstrap-server \
10.0.0.207:6667,10.0.0.208:6667,10.0.0.209:6667 --topic TwitterSample


```
{
    "createdAt": "Dec 29, 2016 7:09:33 AM",
    "id": 814367738172997600,
    "text": "RT @DaisukeMurakami: Merry Christmas!🎄🎁
メリークリスマス！ベンジーサンタさんからプレゼントもらったかなー？ （笑）早く皆さんの前で滑りたいです！がんばります！💪
https://t.co/v4rW7rR2gO https://t.co…",
    "source": "<a href=\"http://twitter.com\" rel=\"nofollow\">Twitter Web Client</a>",
    "isTruncated": false,
    "inReplyToStatusId": -1,
    "inReplyToUserId": -1,
    "isFavorited": false,
    "isRetweeted": false,
    "favoriteCount": 0,
    "retweetCount": 0,
    "isPossiblySensitive": false,
    "lang": "ja",
    "contributorsIDs": [ ],
    "retweetedStatus": {
        "createdAt": "Dec 26, 2016 1:45:37 AM",
        "id": 813199055324971000,
        "text": "Merry Christmas!🎄🎁
メリークリスマス！ベンジーサンタさんからプレゼントもらったかなー？ （笑）早く皆さんの前で滑りたいです！がんばります！💪
https://t.co/v4rW7rR2gO https://t.co/6vbQN9pzBl",
        "source": "<a href=\"http://twitter.com/download/iphone\" rel=\"nofollow\">Twitter for iPhone</a>",
        "isTruncated": false,
        "inReplyToStatusId": -1,
        "inReplyToUserId": -1,
        "isFavorited": false,
        "isRetweeted": false,
        "favoriteCount": 3197,
        "retweetCount": 1526,
        "isPossiblySensitive": false,
        "lang": "ja",
        "contributorsIDs": [ ],
        "userMentionEntities": [ ],
        "urlEntities": [
```

Twitter Tag消息

./bin/kafka-console-consumer.sh --zookeeper 10.0.0.205:2181 --bootstrap-server \
10.0.0.207:6667,10.0.0.208:6667,10.0.0.209:6667 --topic TWITTER_TAG_STREAM2


```
{
    "id": 814370279929442300,
    "createdAt": "Dec 29, 2016 7:19:39 AM",
    "source": "Twitter Web Client",
    "favoriteCount": 0,
    "retweetCount": 0,
    "lang": "th",
    "userTimezone": "Pacific Time (US & Canada)",
    "hashTag": "teamexo"
}
```

将JSON消息映射成表结构

Need to input streaming source record here, will detect the source schema and create a table schema for streaming.

JSON

```
{
    "id": 814370279929442300,
    "createdAt": "Dec 29, 2016 7:19:39 AM",
    "source": "Twitter Web Client",
    "favoriteCount": 0,
    "retweetCount": 0,
    "lang": "th",
    "userTimezone": "Pacific Time (US & Canada)",
    "hashTag": "teamexo"
}
```

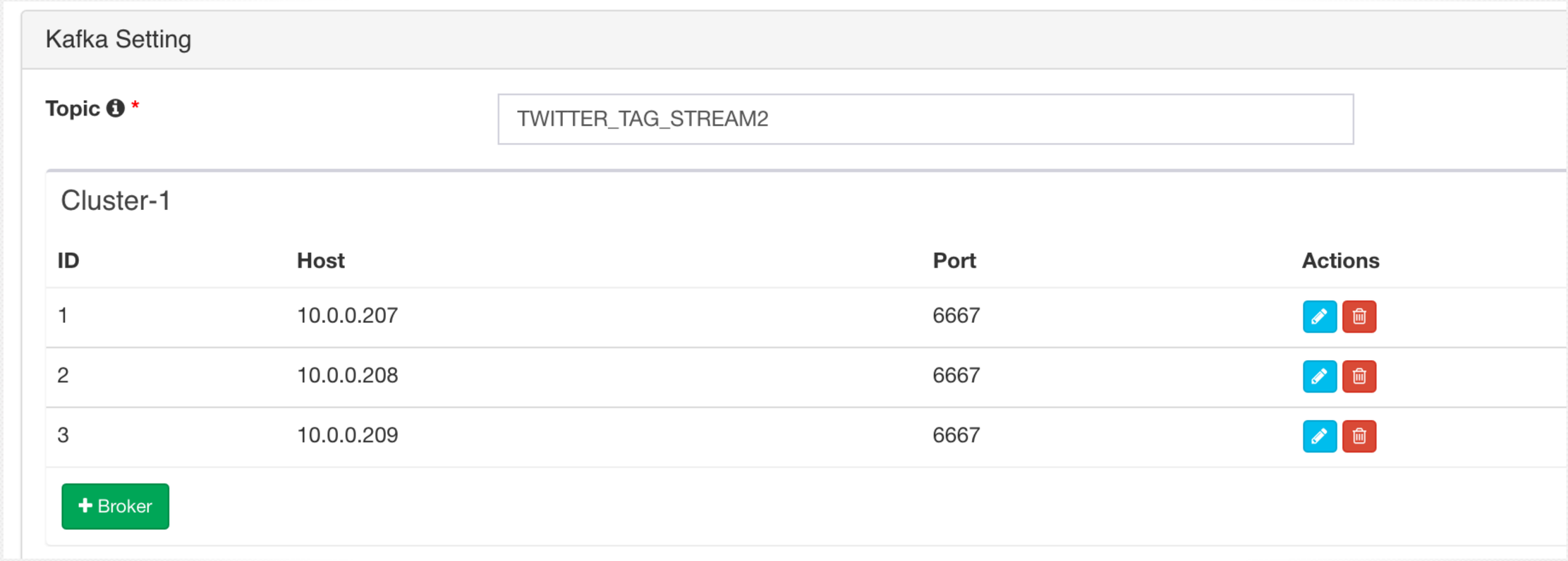
1. Choose 'timestamp' type column for streaming table.
2. By default, system will choose 'Default' as database, you can specify database like this 'database.table'
3. derived time dimensions are calculated from timestamp field to help analysis against different time granularities.

Table Name* TAG_TABLE111

	Column	Column Type	Comment
☐	id	int	
☑	createdAt	timestamp	timestamp
☑	source	varchar(256)	
☐	favoriteCount	int	
☐	retweetCount	int	
☑	lang	varchar(256)	
☑	userTimezone	varchar(256)	
☑	hashTag	varchar(256)	
☑	year_start	date	derived time dimension

Kyligence

输入Kafka集群信息

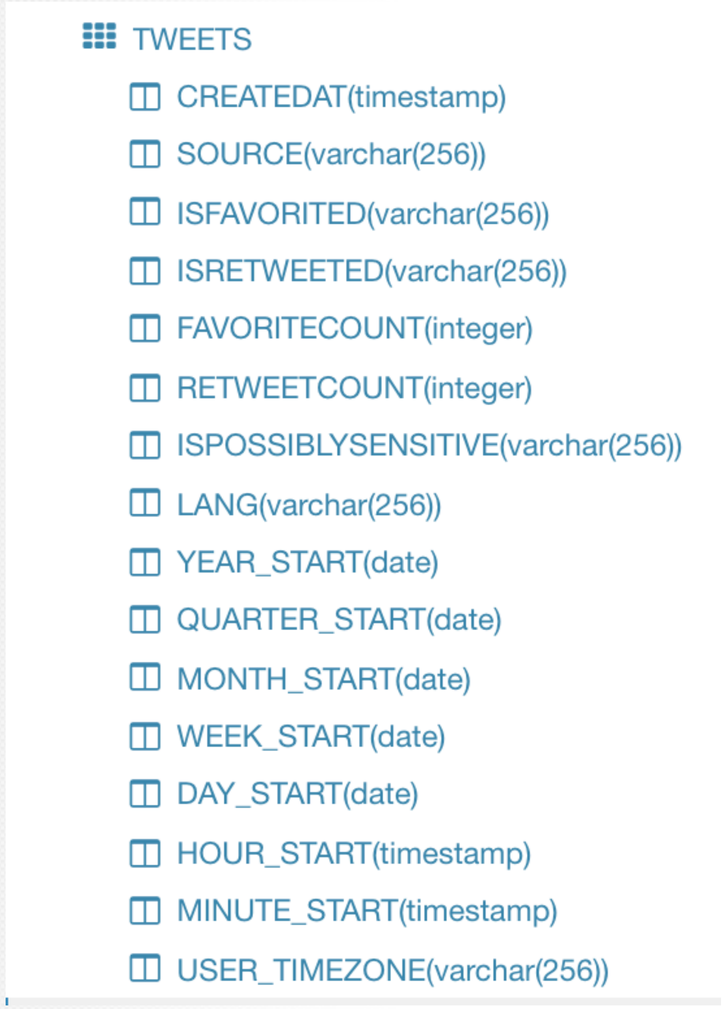


输入消息parser信息

Parser Setting

Parser Name	org.apache.kylin.source.kafka.TimedJsonStreamParser
Parser Properties	tsColName=createdAt;tsParser=org.apache.kylin.source.kafka.DateTimeParser;tsPattern=MMM dd, yyyy hh:mm:ss aa

表结构映射完成

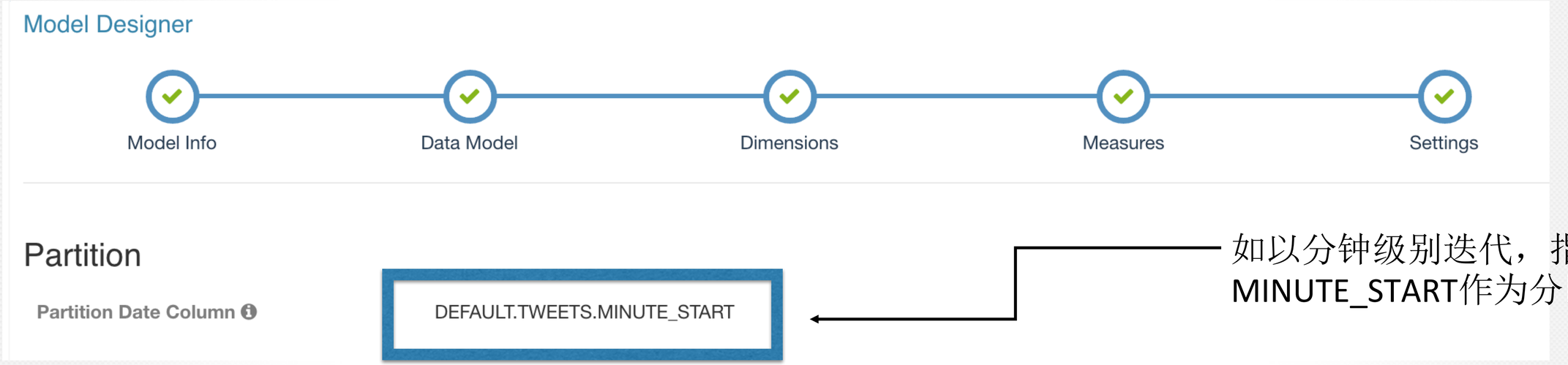


这样一张来源是Kafka的Table就生成了！从它开始创建Model和Cube

创建模型

与普通模型基本没有差异

- 须指定一个时间分区列，因为流式场景下时间是常用查询条件



创建Cube

与普通Cube没有差异；

kylin.hbase.region.cut	0.5
kylin.cube.algorithm	inmem

指定使用inmem算法
可以提速cube构建

Aggregation Groups

Visit aggregation group for more about aggregation group.

ID	Aggregation Groups	
1	Includes	["SOURCE","ISRETWEETED","ISFAVORITED","ISPOSSIBLYSENSITIVE","LANG","DAY_START","HOUR_START","MINUTE_START","USER_TIMEZONE"]
	Mandatory Dimensions	[]
	Hierarchy Dimensions	["DAY_START","HOUR_START","MINUTE_START"]
	Joint Dimensions	["ISRETWEETED","ISFAVORITED","ISPOSSIBLYSENSITIVE"]

对时间列指定hierarchy层级，
对超低基数维度指定joint，减少组合数

构建Cube

Web GUI点击触发，或

RESTful API触发：

```
curl -X PUT --user ADMIN:KYLIN -H "Content-Type: application/json;charset=utf-8"
-d '{ "sourceOffsetStart": 0, "sourceOffsetEnd": 9223372036854775807,
"buildType": "BUILD"}'
http://localhost:7070/kylin/api/cubes/twitter_tag_cube5/build2
```

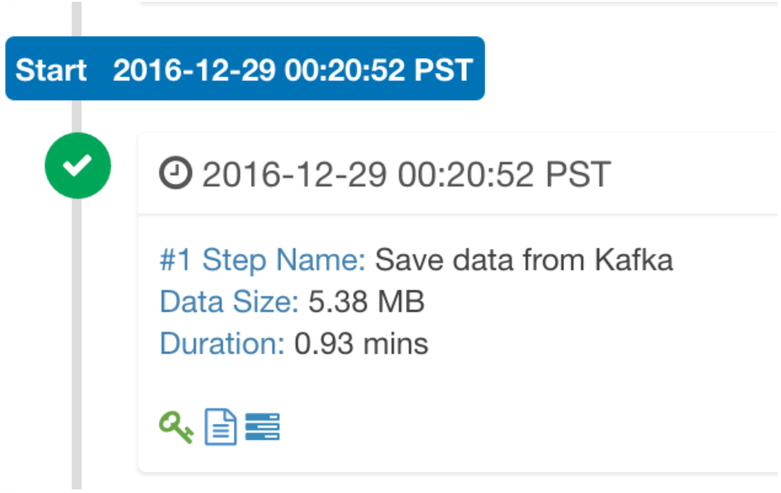
“sourceOffsetStart”: 0 代表从上次构建的结束点开始；

“sourceOffsetEnd”: 9223372036854775807 代表构建到kafka当下最新的消息为止；

Kylin使用Kafka API获取各个partition的最新offset信息，生成任务

构建Cube - continued

- 启动MR job，并行从每个partition获取新消息
- 后续构建将基于HDFS上的消息进行，与Hive类同



Twitter消息的分析场景

- 结合Twitter消息的内容，设计分析场景
- 场景1: 按时间、语言，分析消息流量变化
- 场景2: 按时间、客户端，分析流量变化
- 场景3: 按时间、语言，分析最热门的Tag
- 场景4: 对给定Tag，显示其随时间的频率变化

根据需求设计3个Cube

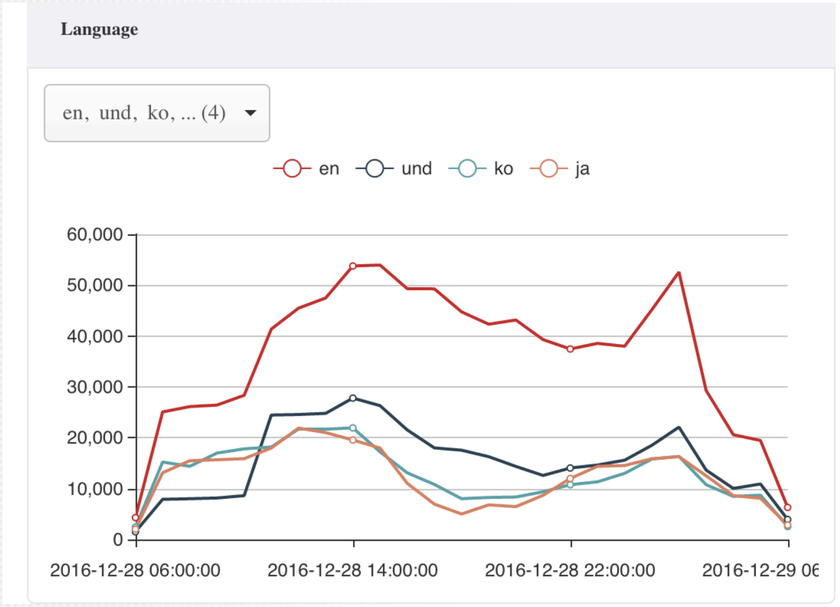
- Cube 1: 支持场景1&2，维度为时间、语言、客户端等；

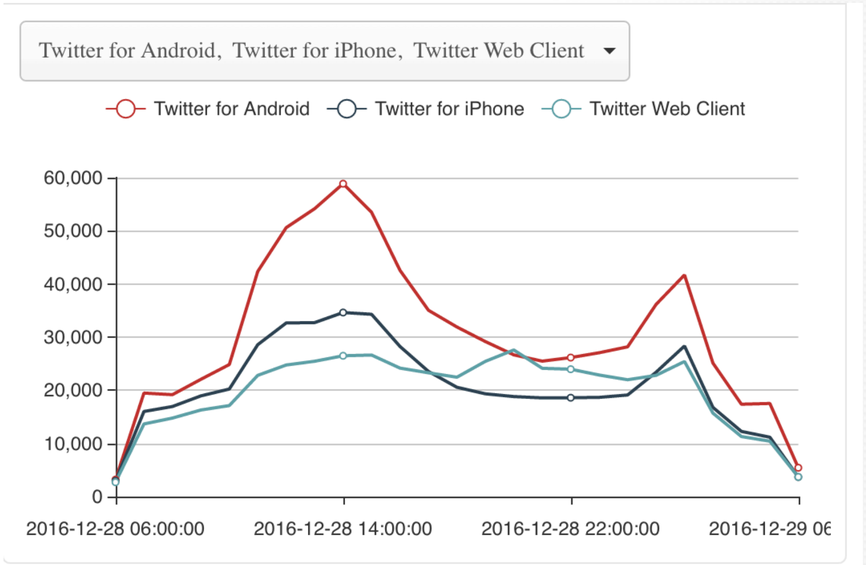
  度量为count(*)

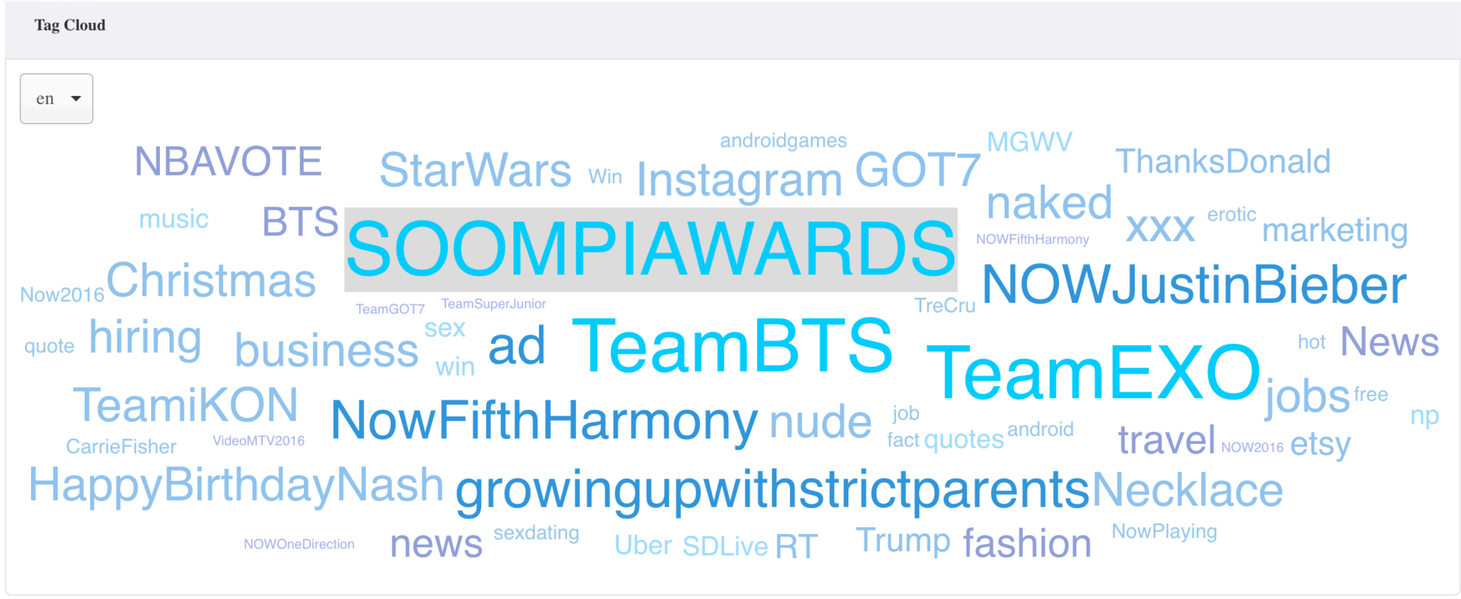
- Cube 2: 支持场景3，维度为时间、语言，度量为TopN

- Cube 3: 支持场景4，维度为时间、Tag，度量为count(*)

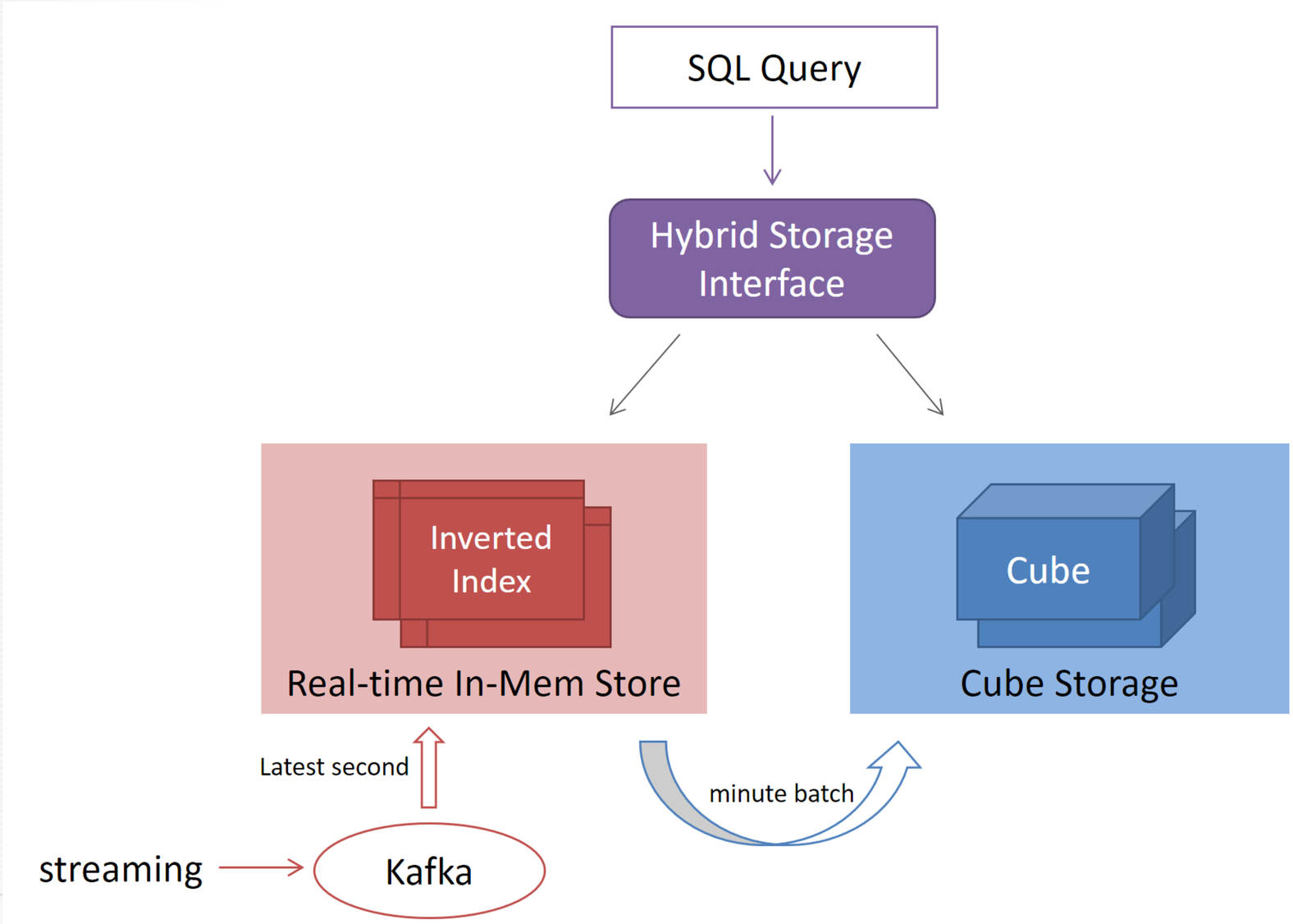
Dashboard







Streaming 还差多少？



未来工作

- 优化MapReduce引擎，提升构建速度
- 尝试持续构建引擎，Spark Streaming/Flink
- 补完实时节点

Kyligence

微信公众号： ApacheKylin