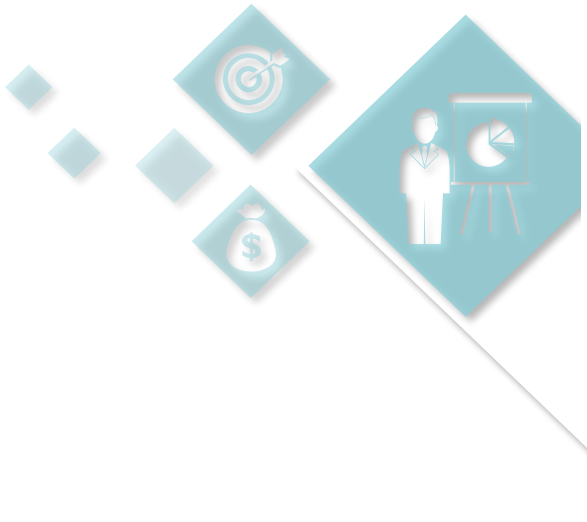


“新代帐指标值实时接入”技术方案分享

部门：数据智能部
日期：2021-08-13
分享人：杨成凯

目 录

CONTENTS

云帐房
YunZhangFang.com

01 需求背景

本需求的业务背景介绍

- 业务含义：“指标值”指的是云帐房企业的税表数据。
- 业务逻辑：报税产品组使用帆软、章鱼报表等可视化工具配置好各种类型的税表;每个企业在每个税期都会生成自己的一份税表数据（即指标值），用于申报。
- 源系统与数据量：在老代帐系统中，指标值存储在mysql业务数据库中，并通过分库分表的方式，存储在12个set中，每个帐期的数据约为35亿到80亿之间。新代帐系统目前将指标值存储在Hbase中。

源系统与数据量

- 本图截取自kudu监控工具，老代帐每个税期的指标值数据量在35亿到80亿之间。
- http://xxx:3000/d/eGKlWzZnz/teng-xun-yun-kudu?orgId=1
- 用户名密码：xxx/xxx

Kudu表操作记录

Kudu表	Delete总数	Insert总数	Update总数	Upsert总数	现存行数	实时增量	间内增量	间内增长率	Tablets总数	存储占用
impala::fintax_account.ztkm_kmye	240,988,542	0	0	11,314,220,760	12,842,731,034	167	2,601,776	0.0404%	63	296 GB
impala::yzf_report.rpt_zbz202012	0	0	0	143,349,153	8,184,834,530	40	106,651	0.00261%	105	716 GB
impala::yzf_report.rpt_zbz20213	0	0	0	307,206,043	6,368,661,256	6	93,561	0.00294%	84	565 GB
impala::yzf_report.rpt_zbz20209	0	0	0	24,493,213	6,261,073,662	0	17,433	0.000557%	105	535 GB
impala::yzf_report.rpt_zbz201912	0	0	0	900,567,236	5,597,912,629	0	21,950	0.000784%	105	457 GB
impala::yzf_report.rpt_zbz20206	0	0	0	21,631,310	5,283,444,627	0	16,757	0.000634%	126	460 GB
impala::yzf_report.rpt_zbz20216	0	0	0	7,787,365,411	5,200,460,106	29	378,429	0.0145%	84	533 GB
impala::yzf_report.octopus_qushu_table	0	0	0	9,388,217,169	4,573,282,189	83	4,865,379	0.211%	336	528 GB
impala::yzf_report.rpt_zbz202011	0	0	0	24,867,249	3,973,314,766	0	20,315	0.00102%	105	304 GB
impala::yzf_report.rpt_zbz202010	0	0	0	22,760,664	3,641,963,652	0	23,412	0.00129%	105	266 GB
impala::yzf_report.rpt_zbz20211	0	0	0	152,140,266	3,554,653,121	0	69,793	0.00393%	105	270 GB

Kudu Table实时明细

本需求的业务背景介绍

- **核心业务诉求**：报税产品组需要关联其他大表如任务表，全局查询某些指标值。而直接使用mysql直接查询，往往需要几个小时的时间才能查询出来。因此希望通过大数据的技术手段，尽量在几分钟内完成查询。

核心业务诉求拆解

- 查询快：

1. 查询尽量在几分钟内完成，速度越快越好。
2. 对于报税产品组来说，查询指标值的原因有两个：
3. 第一个是检查税表配置是否正确，在这种情况下，查询速度越快，越能帮助报税产品组定位到错误，从而增加税表的准确性。
4. 第二个是客户需要自己的税表中指标值数据用于分析。同样，越快的查询出数据，就越能提升用户的体验。

- 数据准：

数据准体现在两个方面：

1. 数据不能丢。在实时同步的过程中，数据不能丢失，一旦丢失，就无法保证数据完整。
2. 数据实时同步要快。数据应尽量在每一个时刻与源系统中保持完全一致，数据延迟不能过大。

核心业务诉求拆解

- 支持大表关联：

在实际查询过程中，经常需要将指标值与企业表、任务中心等大表关联。因此必须保证指标值数据能与其他表进行关联。

- 查询时候sql逻辑要简单：

接入到大数据平台后，遇到跨帐期、跨地区查询的时候，sql越简单越好，数据最好只存在一个表中。

- 数据初始化过程应简单、快速：

在系统上线或实时数据丢失时候，能在几个小时内完成数据初始化、并且数据初始化应简单易用。

云帐房
YunZhangFang.com

02

技术选型

技术选型

技术选型主要根据核心业务诉求与实际数据量进行。

技术选型-核心业务诉求拆解

- 查询快：

使用kudu存储结构+impala/presto查询引擎能满足此需求。
kudu是由cloudera公司出品的存储系统，对于scan与随机访问都有很好的性能。对于几十亿数据量的大表，配合impala或presto，能在几分钟内查询出结果。

因此指标值数据应存储在kudu中。

- 数据准：

目前实时工具主要有canal、 StreamSets、 SparkStreaming、 Flink三种工具。

在本需求中，除了实时同步，还需要对每条数据进行实时拆分、保证端到端一致性、具备容错与重启机制。这里综合比较采用Flink作为实时同步工具。

技术选型-核心业务诉求拆解

- 支持大表关联：

kudu中已经存放了任务表，将指标值存入到kudu中可直接支持大表关联。

- 查询时候sql逻辑要简单：

经估算，2021年指标值一年的数据量在400亿到600亿之间，且存在年内跨帐期取数的场景。因此使用range+hash的方式设计kudu表，单表存储一年的指标值数据。这样只需查询一张表便可跨帐期查询。

- 数据初始化过程应简单、快速：

经过评估，新代帐指标值采用双写的方式，写入到HBase和ES中。
为不干扰线上环境稳定性，可使用Hive连接ES、Presto从Hive写入到Kudu的方式完成数据初始化。
使用workflow的方式按照帐期调度Hive同步数据。

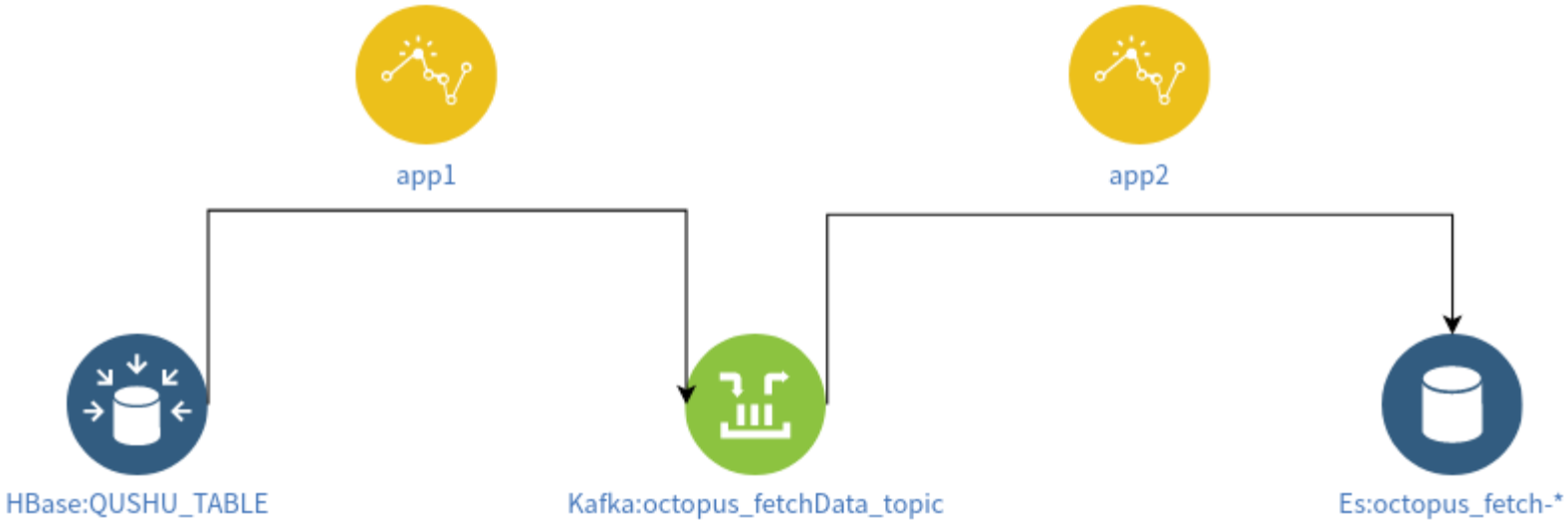
技术选型结论

- 使用kudu存储引擎存储数据。
- 使用Flink实时接入数据。
- 使用hive与presto做数据初始化。

云帐房
YunZhangFang.com

03 技术方案

技术方案-当前算税支撑组的应用架构



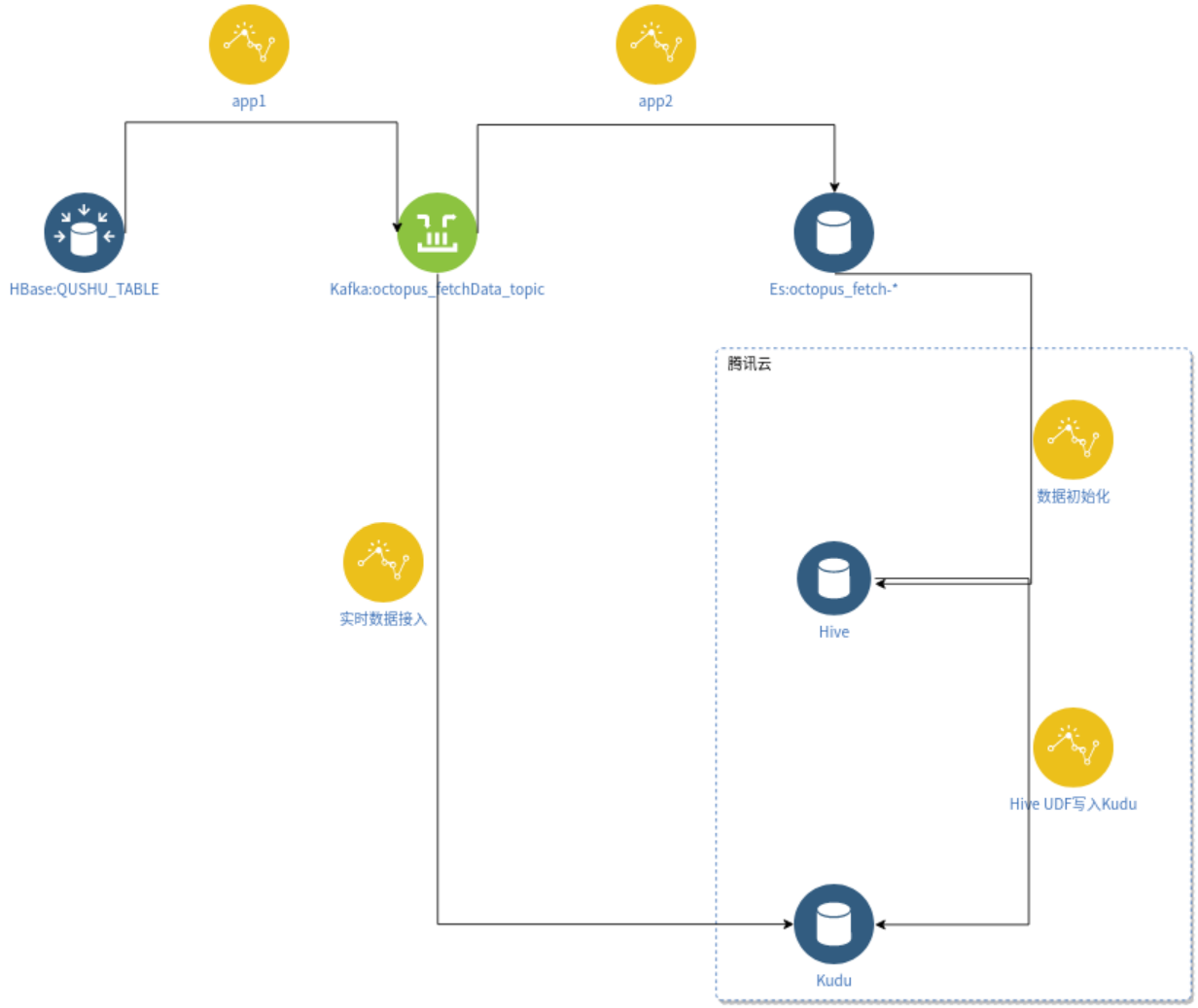
当前算税支撑组采用双写的方式将数据写入到HBase与ES中。
类似基于binlog的mysql主从同步方案，线上应用读写HBase。
app1、app2将数据同步到Kafka与Es，通过ES进行查询分析。

技术方案

通过应用架构、数据架构、网络拓扑架构完成技术方案，其中：

- 应用架构，主要关注采用如何的技术栈、应用间如何完成数据交换。
- 数据架构，主要关注数据的转换。
- 网络拓扑架构，主要关注应用之间的网络拓扑关系。

技术方案-应用架构设计



1、为避免消耗线上HBase 的资源,数据实时接入端通过 Kafka 另起消费者订阅消费进行。

2、数据初始化则从 Es 端进行。使用hive的es connector连接es，将数据同步到大数据平台的hive临时表中。

3、使用hive udf将数据拉窄。

4、最后通过presto将hive的初始化数据写入到kudu中。

技术方案-数据架构设计


```
"areaName":"全国",
"boxId":"778944412830113793",
"cells":[
 {
  "isChange":"NOCHANGE",
  "location":"A1",
  "value":"1"
 },
 {
  "isChange":"NOCHANGE",
  "location":"A2",
  "value":"1"
 },
 {
  "isChange":"NOCHANGE",
  "location":"A4",
  "value":"Y"
 },
 {
  "isChange":"NOCHANGE",
  "location":"B1",
  "value":"0"
 }
],
"createTime":"2021-05-132 15:21:50",
"dzQyId":"722839913816526849",
"dzQyName":"测试",
"fetchDataTime":1620804083847,
"id":"840244025863835648_2021_4_778944412830113793_778944412834308097_sheet1",
"kjnd":"2021",
"kjqj":"4",
"nsqxdm":"4",
"parentBoxId":"778941549412327424",
"qyId":"840244025863835648",
"qyName":"青岛小企业年报测试",
"sbszId":"3702040002",
"sheetName":"sheet1",
"systemId":"4988129485804011153"
}
```


Hbase中的Cells，代表了这个Sheet页中的指标值格子

HBase中的Rowkey，标注了这条数据是属于某个企业某个帐期某个boxid的sheet1页面

- 在HBase中Rowkey的设计原则为"企业id+会计年度+会计期间+boxId+版本号+SheetName"
- HBase 中 Cells 族列中存储了指标值的位置与值,如 B3 指的是 B 列第三行，
- 因此左图的一条数据包含了某个企业某个帐期某个税表的sheet页中所有指标值格子。

技术方案-数据架构设计

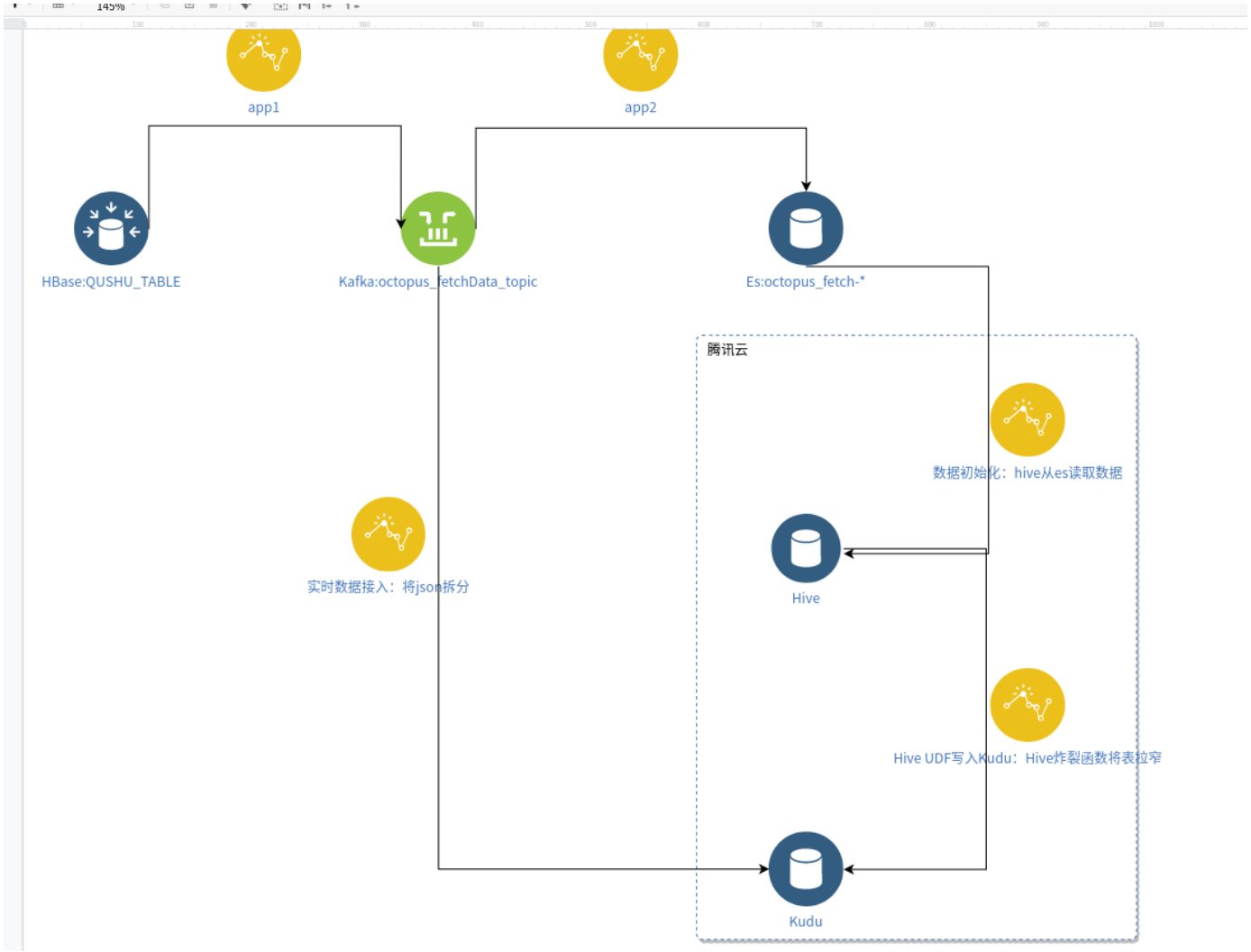
```
CREATE TABLE test.octopus_qushu_table_2021q3 (
  id string not null comment 'hbase中的rowkey', --
  840244025863835648_2021_4_778944412830113793_778944412834308097_sheet1,
  cell_location string not null comment '格子的坐标如A1',
  cell_is_change string comment '格子是否变换',
  cell_value string comment '格子的值',
  area_id string comment '区域id', -- 0,
  area_name string comment '区域名称', --全国,
  box_id string comment 'boxid', -- 778944412830113793,
  create_time string comment '创建时间', -- 2021-05-13215:21:50,
  dz_qy_id string comment '代账公司id', -- 722839913816526849,
  dz_qy_name string comment '代账公司名称', -- 测试,
  fetch_data_time bigint comment '取数时间', -- 1620804083847,
```

这条数据为某个企业202104帐期某个税表的sheet页面中的某个格子。

```
  kjnd string comment '会计年度', -- 2021,
  kjqj string comment '会计期间', -- 4,
  nsqxdm string comment '纳税期限代码', -- 4,
  parent_box_id string comment '上级boxid', -- 778941549412327424,
  qy_id string comment '企业id', -- 840244025863835648,
  qy_name string comment '企业名称', -- 青岛小企业年报测试,
  sbsz_id string comment '申报税种id', -- 3702040002,
  sheet_name string comment 'sheet页名称', -- sheet1,
  system_id string comment '', -- 4.98813_e+17
    PRIMARY KEY (id,cell_location)
)
partition by hash(id,cell_location) partitions 14
COMMENT '新代账hbase实时同步指标值表 2021第2季度'
stored as kudu;
```

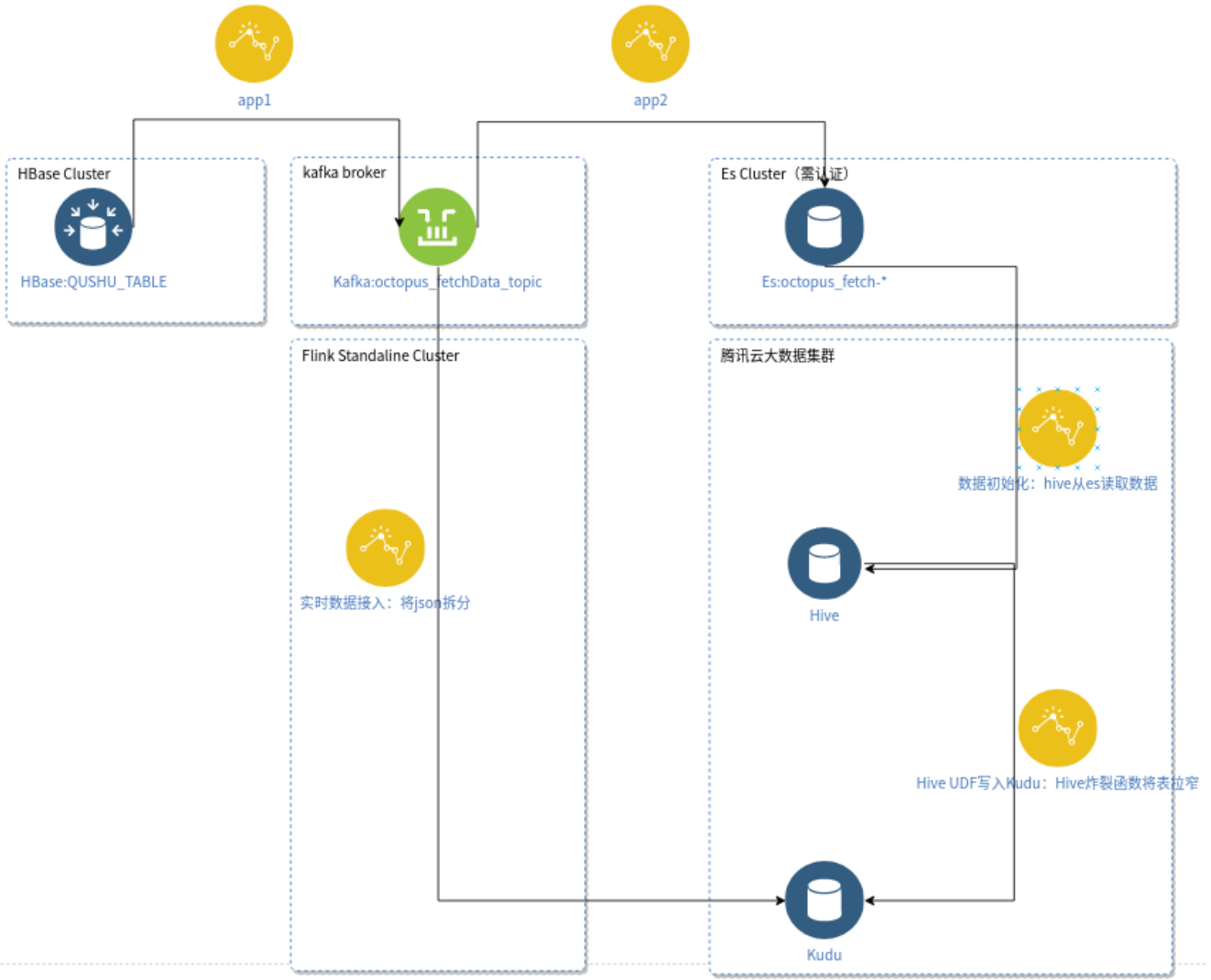
- 为了便于用户查询，kudu建表方式为将HBase中Cells族列中的所有cells进行拆分，也就是将整个表拉窄。
- 表中数据的最细粒度变为Hbase中Rowkey+cell位置。
- 因此左图的一条数据包含了某个企业某个帐期某个税表的sheet页中的某个格子。

技术方案-数据架构设计



- 因此实时端将kafka中json进行拆分，写入到kudu中。
- 离线端使用hive的炸裂函数，将数据拆分把表拉窄。

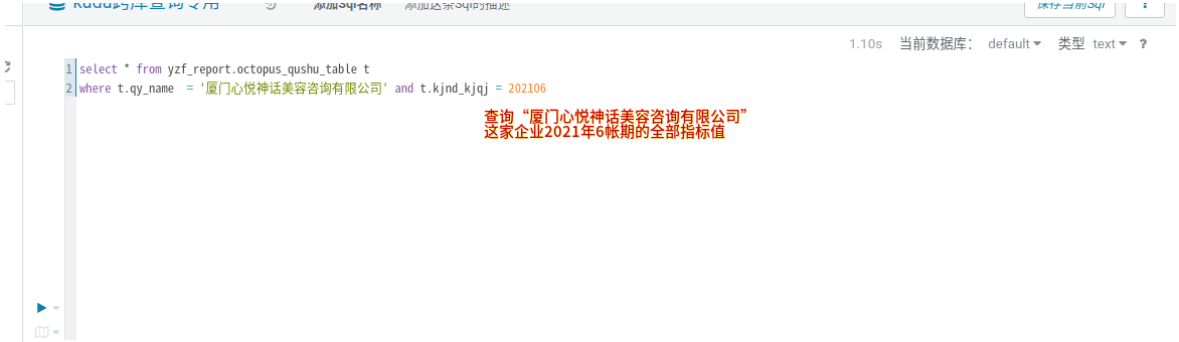
技术方案-网络拓扑架构设计



本图展示了各个应用所在的集群或网络位置

技术方案-效果

- 该技术方案已于2021年8月10日完成上线。
- 截至2021年8月15日，通过数据初始化与实时接入，已完成了45亿行指标值的数据接入。
- 单表查询、大表关联查询基本在10秒到5分钟完成。



云帐房
YunZhangFang.com

04 监控、报警

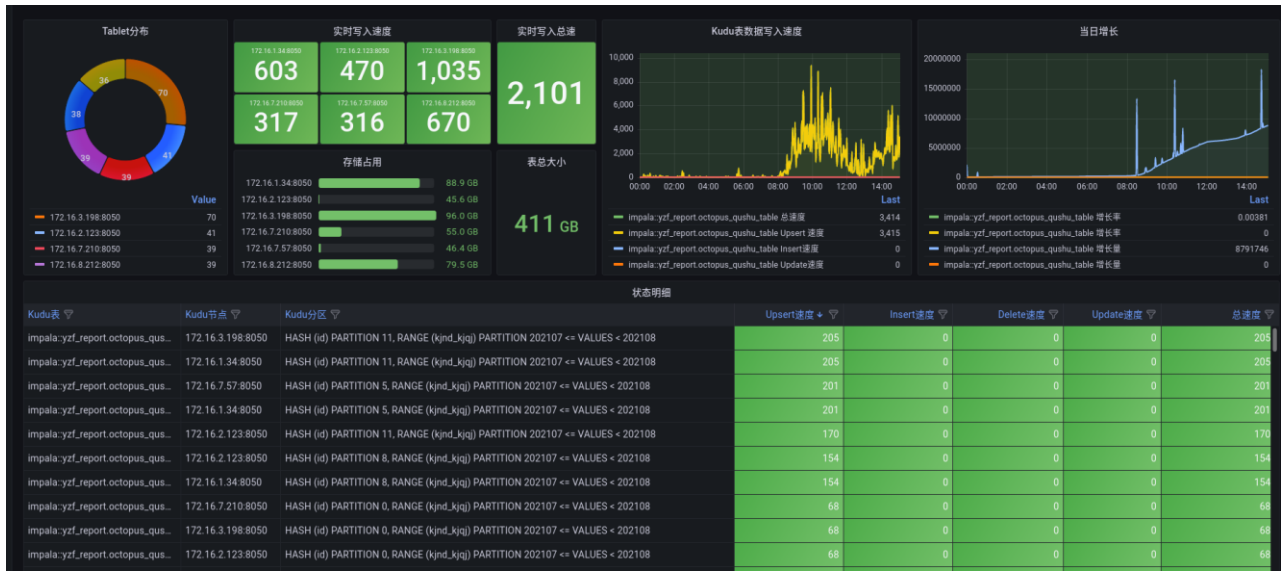
监控与报警

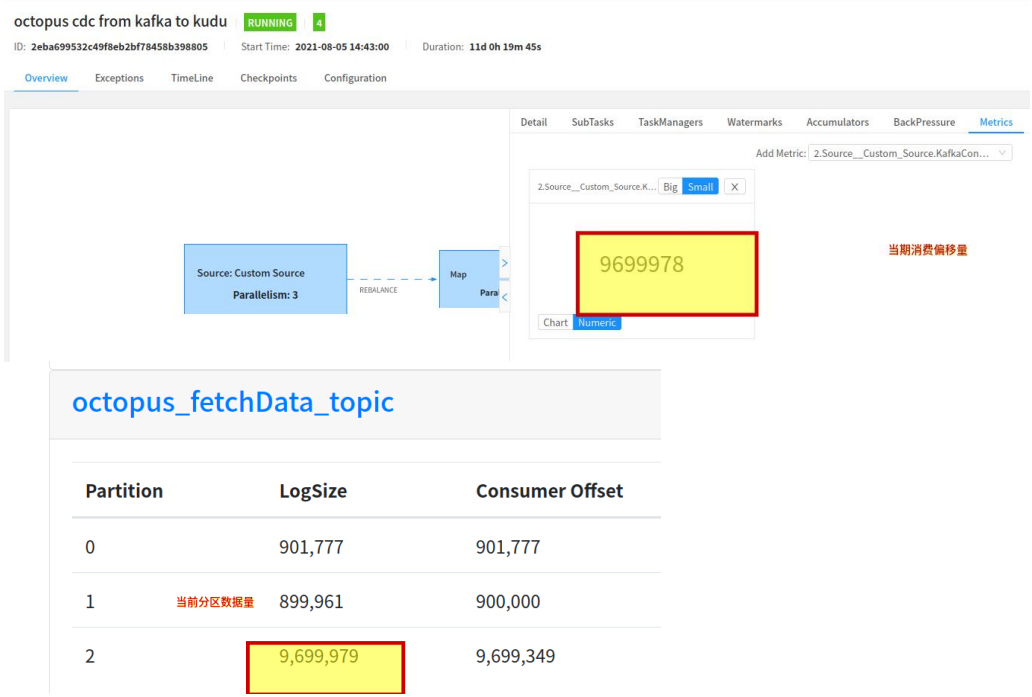
本项目的监控与报警主要关注以下两个方面：
1、数据同步过程中的延时。

2、实时应用在运行时发生的异常。

监控

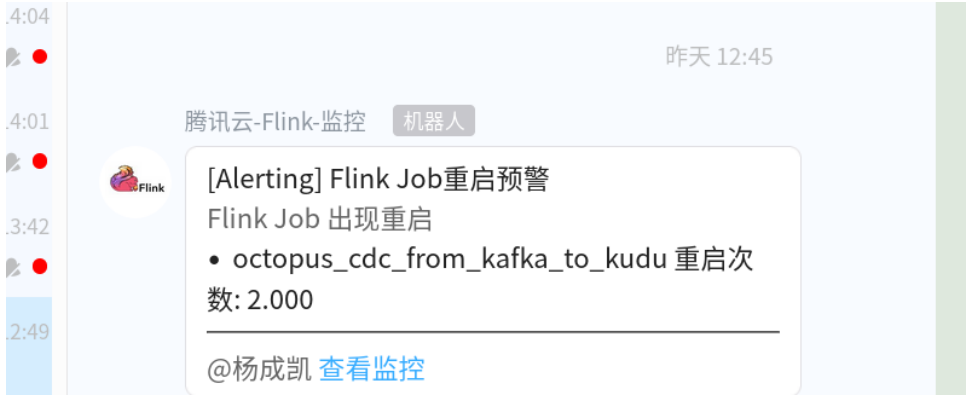
- 可通过flink的metrics与kafka manager实时查看当前的消费进度。
- 通过grafana查看kudu表的实时写入速度、当日增长量。





告警

- 如果消费组的lag过大，工作流异常，会通过钉钉告警的方式进行提醒：



05 总结与启发

总结与启发

- 对于大数据相关项目，应重点关注项目的未来数据量、tps等指标，评估好项目所需资源。
- 平时要做好各个组件的监控与预警，要提前评估好集群整体的资源。
- 在监控与预警机制完善情况下，当技术方案定下来后开始开发前，要提前在线上进行模拟验证，保证方案可行

THE END