Druid在360的实践与优化

nichuanlei@360.cn

360互联网安全中心

Agenda

- Druid在360的实践
- 功能优化
  - Virtual datasource
    - Naïve Virtual datasource
    - Improved Virtual datasource
  - 三个细节改进
- 未来计划
  - Druid realtime on flink

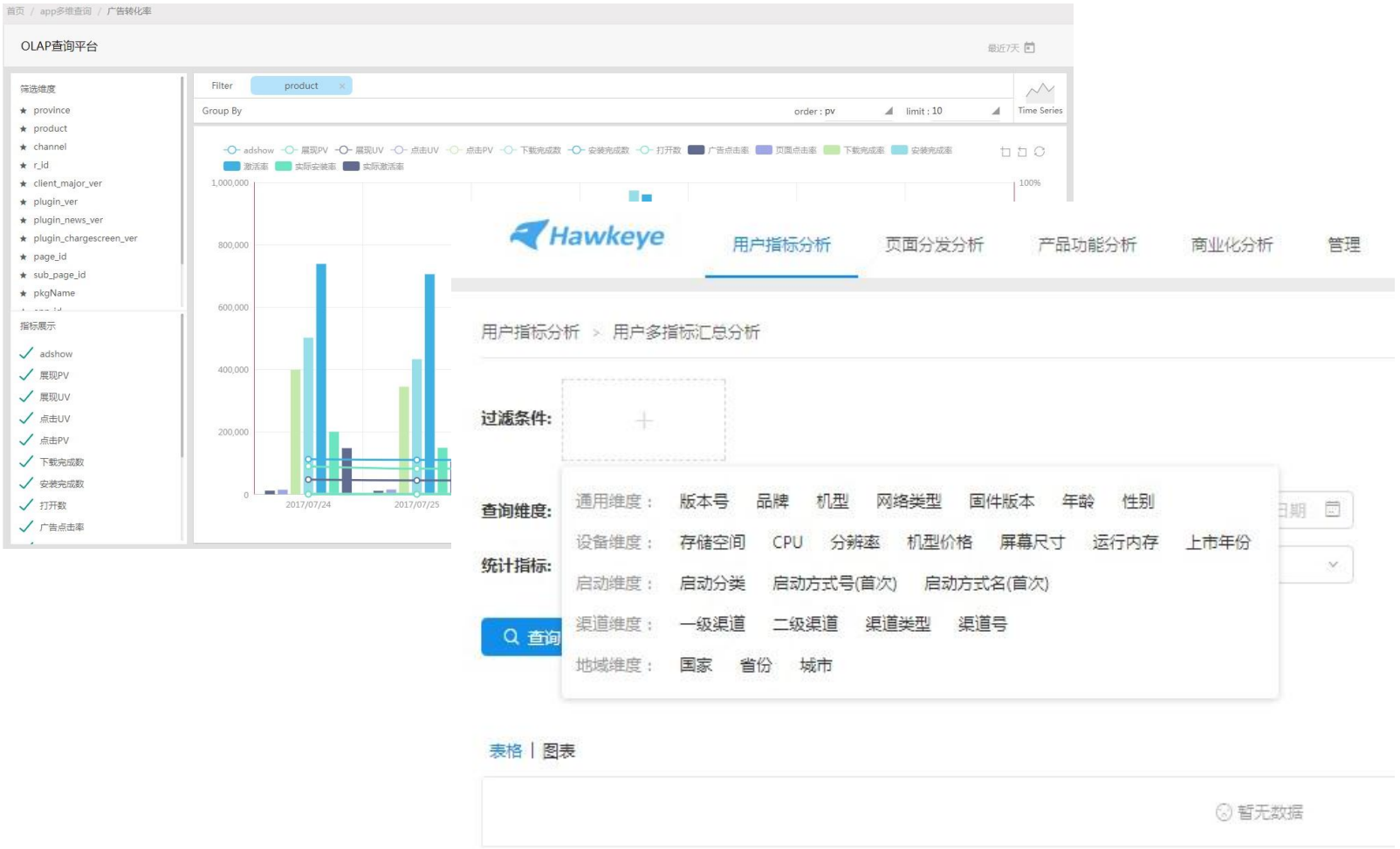
Druid在360的实践

360 互联网安全中心

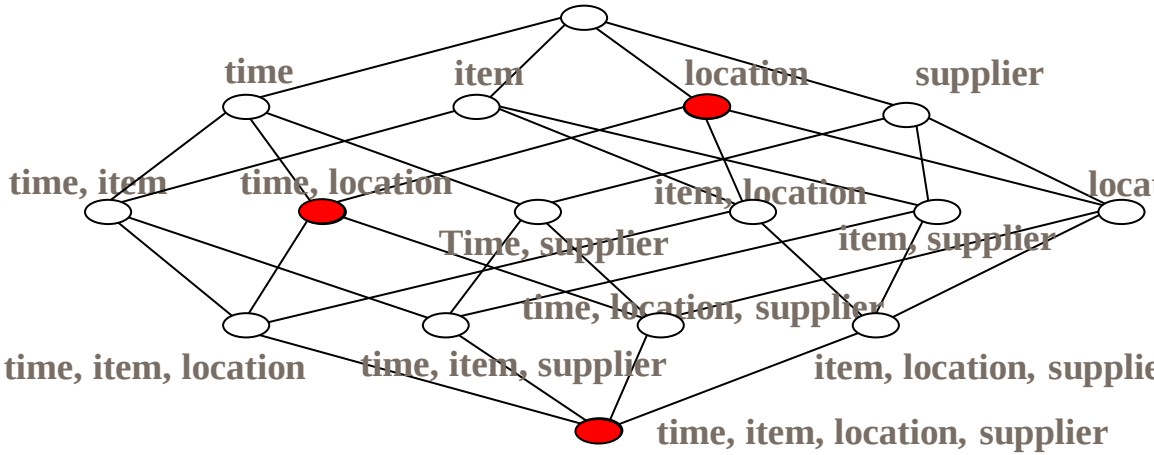
- Druid版本
  - 在0.10.0的基础之上修复bug、增加功能
- 集群状态
  - 两个集群，分别包含20台、10台机器
  - 机器配置
    - 内存：256G
    - CPU：24核
    - 磁盘：1.2T Flask卡
    - 网络：普通千兆
- 业务状态
  - 60个data source
  - 平均每天导入100亿条原始数据
  - 95%查询在3秒以内，平均查询1.4s

Druid在360的实践

- 前端交互页面



- Virtual datasource的思路



0-D(apex) cuboid

1-D cuboids

2-D cuboids

3-D cuboids

4-D(base) cuboid

datasource	c	dims
dm_dss_sj_user_di	129	br,is_active,md
dm_dss_sj_user_di	100	br
dm_dss_sj_user_di	74	br,is_active
dm_dss_sj_user_di	45	br,md
dm_dss_sj_user_di	28	is_active
dm_dss_sj_user_di	19	is_active,package
dm_dss_sj_user_di	19	br,is_active,v
dm_dss_sj_user_di	18	is_active,sk
dm_dss_sj_user_di	13	br,city
dm_dss_sj_user_di	10	br,is_active,sk
dm_dss_sj_user_di	9	sk
dm_dss_sj_user_di	9	is_active,md
dm_dss_sj_user_di	7	ch_catef,price,md,sk
dm_dss_sj_user_di	6	is_active,v
dm_dss_sj_user_di	6	is_active,is_online

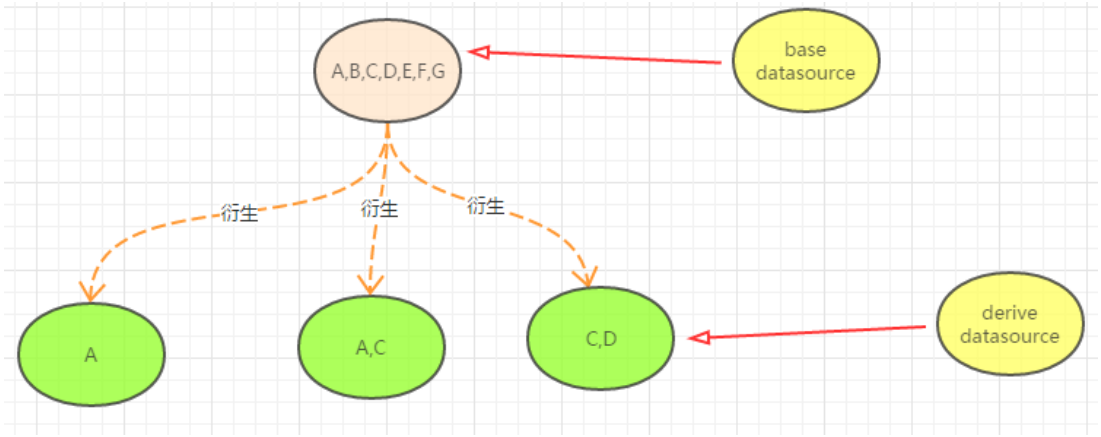
共41个维度

datasource	c	dims
sys-shbt_wukongtvsdk_logshare_event_hyper	432	event_key
sys-shbt_wukongtvsdk_logshare_event_hyper	51	packname,event_key
sys-shbt_wukongtvsdk_logshare_event_hyper	9	event_seg
sys-shbt_wukongtvsdk_logshare_event_hyper	6	packname
sys-shbt_wukongtvsdk_logshare_event_hyper	5	servercountry
sys-shbt_wukongtvsdk_logshare_event_hyper	5	channel
sys-shbt_wukongtvsdk_logshare_event_hyper	4	event_seg,event_key
sys-shbt_wukongtvsdk_logshare_event_hyper	2	serverprovince

共17个维度

功能优化

- Naïve Virtual datasource
  - 概念定义



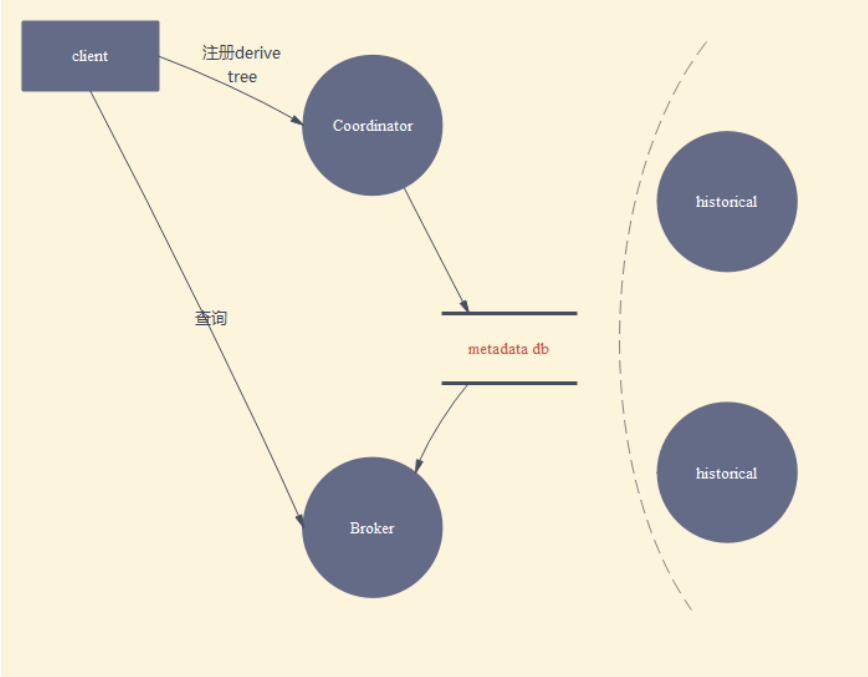
derive tree

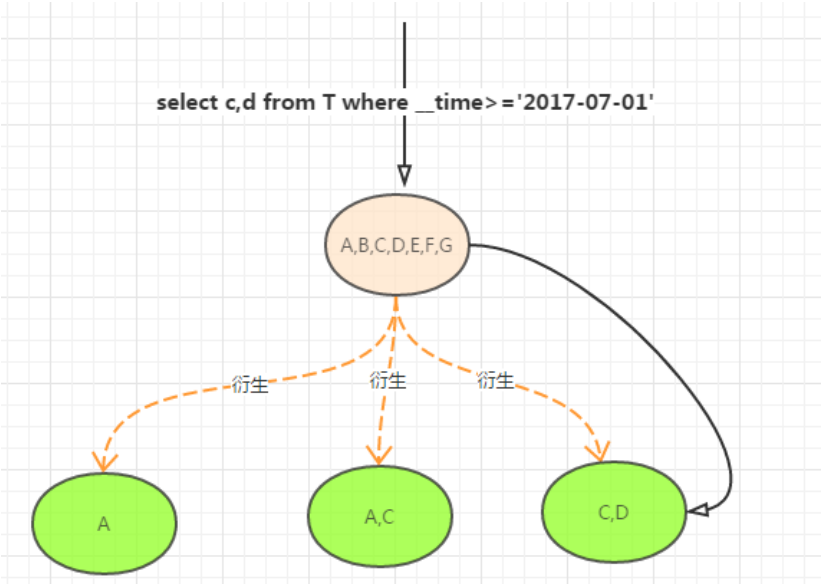
derive tree作为整体对外提供查询服务，我们把这个逻辑上的datasource称为virtual datasource

事实上，virtual datasource的schema与base datasource一致，作为druid用户对整个derive tree无感知

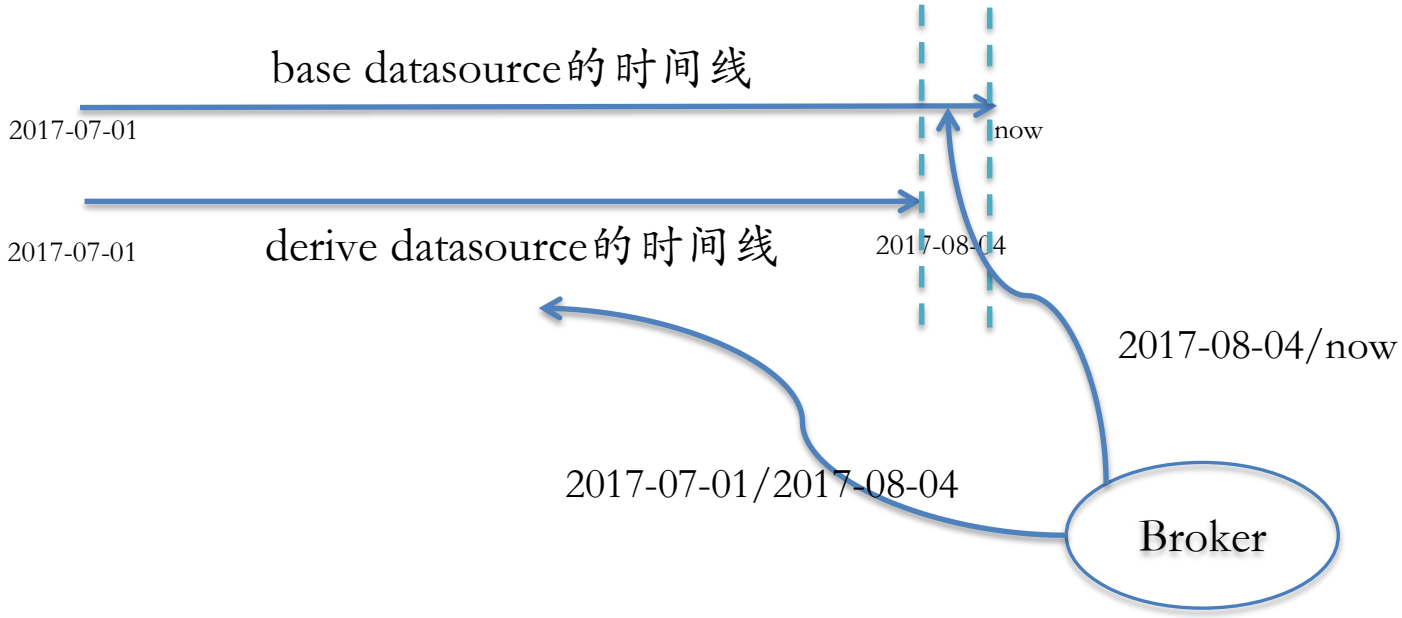
功能优化

- Naïve Virtual datasource（cont）
  - 实现思路





- Improved Virtual datasource
  - 简单版本的Virtual Datasource面临以下两个缺点
    - 在构建derive datasource期间，derive datasource不能发挥作用
    - 无法对实时灌入的datasource生效



功能优化

- Virtual datasource的效果
  - 空间开销（多消耗2%以内的磁盘存储空间）

以druid自带的wikiticker为例，我们在coordinator界面能看到左图的datasource

  - 性能提升
    - 某datasource 包含31个维度，每天30亿原始数据，build完成之后20G，平均查询延迟20秒
    - 为其构建3个derive datasource，每天消耗磁盘200M
    - derive datasource覆盖80%以上查询，平均延迟2秒

细节改进1

- Historical单核打满

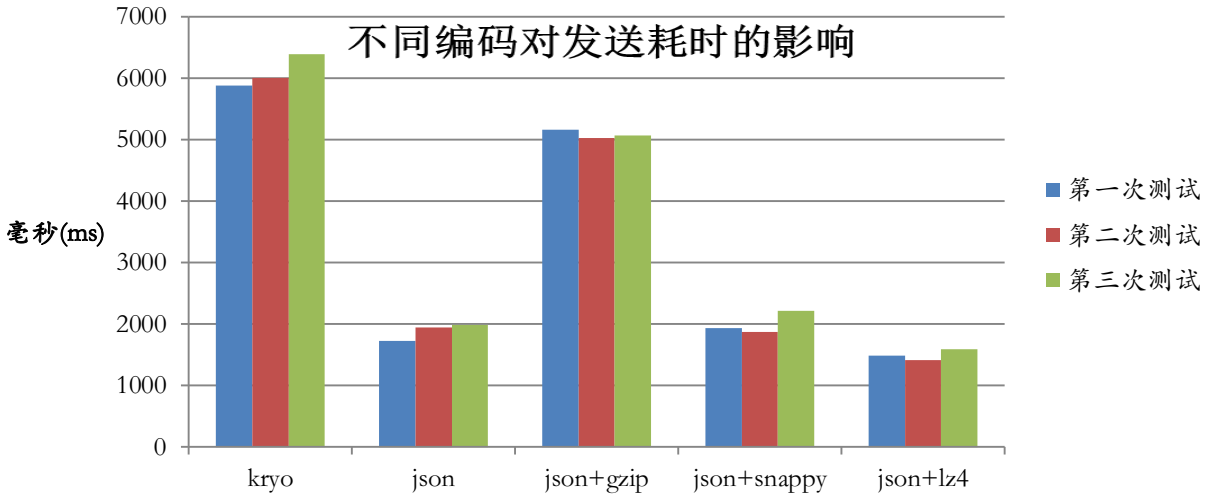
原因：historical并发地涉及到的segment处理完毕之后，由单个线程完成排序、序列化、压缩三个耗时功能

解决方案:引入异步机制，将三个耗时操作并行化

细节改进2

- 传输层编码

Broker与Historical之间通过HTTP协议交换数据，默认打开gzip压缩，也可以选择不压缩（plain json）；如果Historical返回158M结果数据，使用不同编码、压缩方式发送耗时对比如下



单核问题+传输层编码=30%的性能提升

细节改进3

- Hive数据导入
  - 问题：hdfs上的数据不包含时间信息，时间信息只包含在partition中
  - 怎么办？清洗一遍数据，将时间戳信息包含在hdfs中？

    简单修改代码支持以下这种时间戳的配置格式

```
"timestampSpec" : {
  "format" : "yyyyMMdd",
  "column" : "__constant__",
  "missingValue": "20150912"
}
```

实际上hdfs原始数据中并不存在__constant__列，druid在build的时候遇到这种特殊情况就将其解析为20150912这一固定日期

未来计划

360 互联网安全中心

- 新的实时导入方案
  - kafka indexing service的问题
    - realtime node与historical资源竞争
    - 进程崩溃从头开始消费kafka
    - 作为一个实时统计方案，未能与现有系统完美融合
  - 我们的方案(druid realtime on flink)
    - On yarn
    - 管理实时灌入过程中的segment（checkpoint机制）
    - 完善的配套
  - 更有价值的意义
    - 成为实时统计业务的统一入口

总结

360 互联网安全中心

- ✓ Thanks
- ✓ Question?

系统部 INFRASTRUCTURE SYSTEM