



携程机票数仓建设之路

演讲人 张振华
2019年11月9日



张振华



本科、博士分别就读于中国科技大学、中国科学院大学
计算机专业

2015年加入携程机票研发部，负责大数据技术调研及在
机票的落地。

2018年起加入机票数据仓库团队，主要负责数据仓库技
术架构、性能优化、数仓规范制定、数据模型设计以及
数据应用开发

目录

- 1 机票数仓演进历程
- 2 数仓技术栈
- 3 数仓建设时的共性问题与实践
- 4 数据质量体系
- 5 应用案例

数仓技术演进历史

2008年

- Kettle
- Informaticas
- SQLSever
- BO

2014年

- Hadoop
- Hive
- HBase

2015年

- Kafka
- Spark
- Storm
- Zeus
- DataX

2016年

- ElasticSearch
- Presto
- ART

2018年

- Flink
- ClickHouse
- CrateDB
- ArtNova

当前技术栈



数仓的团队合作

产品

- 用户行为分析
- 运营可视化系统
- 埋点管理系统
- 大数据查询工具
- 取数、报表、数据分析

业务、财务

- 报表
- 运营可视化系统
- Adhoc取数
- 业财一体化系统

开发、测试

- 用户画像服务
- 大数据查询前端、服务
- 日志追踪系统
- 报文解析系统
- 流量回放系统

数据分析师

- 底表
- 自定义函数集
- 数据处理
- 分布式算法实现

营销

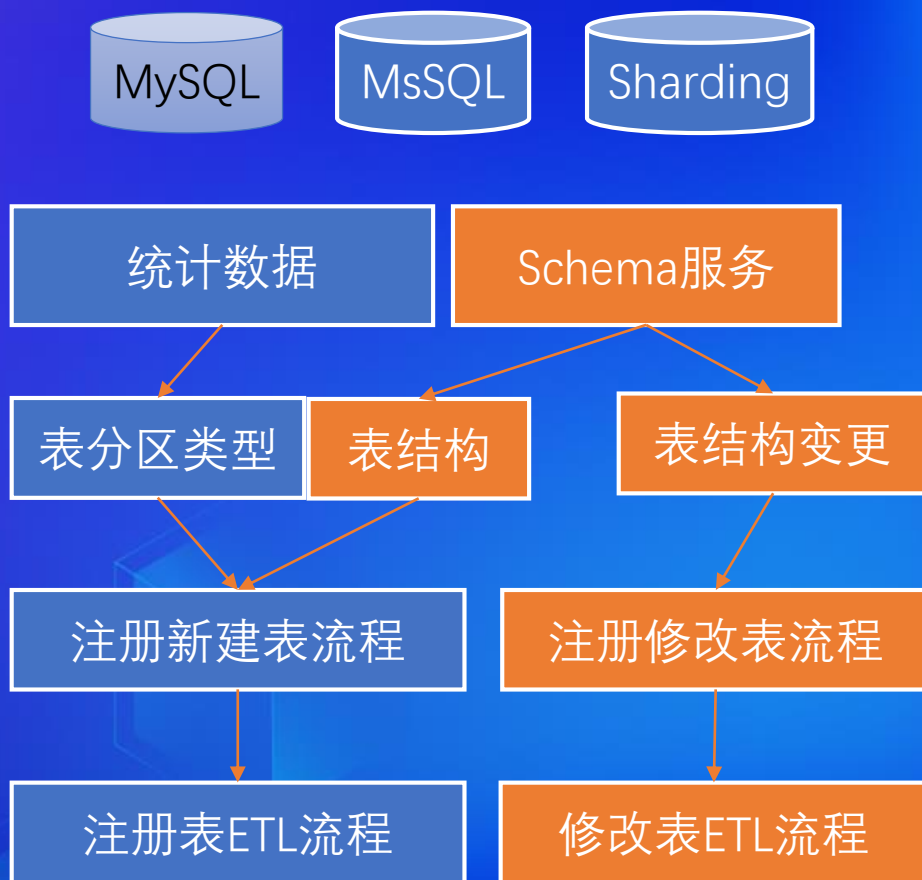
- 定投营销系统
- 效果分析

数仓建设内容



数据同步

DB->HIVE



- 覆盖所有表
- DataX
- 获取DB SCHEMA, 映射成HIVE SCHEMA,自动注册ETL zeus脚本,定期同步生产schema的变更
- 初始化脚本
 - 全量同步
- 分区类型
 - 全量分区
 - UPDATE DML的表
 - 增量分区
 - INSERT ONLY的表
- 无分区
 - 基础数据表
- Hint: 尽量不要物理删除, 采用逻辑删除、批量物理删除

数据同步

Kafka

Partition

Partition

Partition

Decoder

新建表

已有表

Schema
解析

建表

表结构获取

提取数据

HIVE表

Hive
Offset
Record
Table

Spark Job

基于spark-sql-kafka的二次开发

- 配置简单
- 稳定
- 方便异常监控

```
cat <<EOF > hamal.properties

cluster.name=
topic.name=
data.db.name=
log.db.name=di
log.table.name=logs
create.table=false

EOF

/opt/apr/... spark-2.3.0-bin-hadoop2.tgz master yarn --deploy-mode client \
--class ... hamal-ValueError: you need dependencies ... hamal.properties"
```

示例

数仓分层设计



宽表设计

便捷

Trade off

冗余

销售渠道

出票日期

主订单号

用户行为还原

用户原始行程

拆单类型

Clientcode
Sid
Vid

前端信息

设备标识

经纬度

订单宽表

串联
UUID

三字码
翻译

中转时长

行程信息

行程概况

往返停留时长

不惧冗余

便利使用

不变粒度

宽表

兼顾性能

丰富维度

数据分层-解析处理框架



1. 对非大数据开发同学虚化大数据处理框架
2. 各司其职，处理需求迭代周期短

聚合层数据解析

1. 聚合层返回给前端的用户查询结果
2. 数据格式zstd/gzip+ProtoBuf
3. 解析需求：
将航组、产品类型、乘客类型、票台、价格数据展平
4. 一些数值
 - a. 17年初上线，积累800TB
 - b. 日增80亿

元数据管理



基础信息

表结构描述

字段注释

存储

• 位置

• 格式

分区表

• 分区类型 增量/全量

• 分区统计

生命周期ttl

创建时间

表修改时间

使用热度



血缘关系

表与表

表与报表

字段

表与作业



标签

层级标签

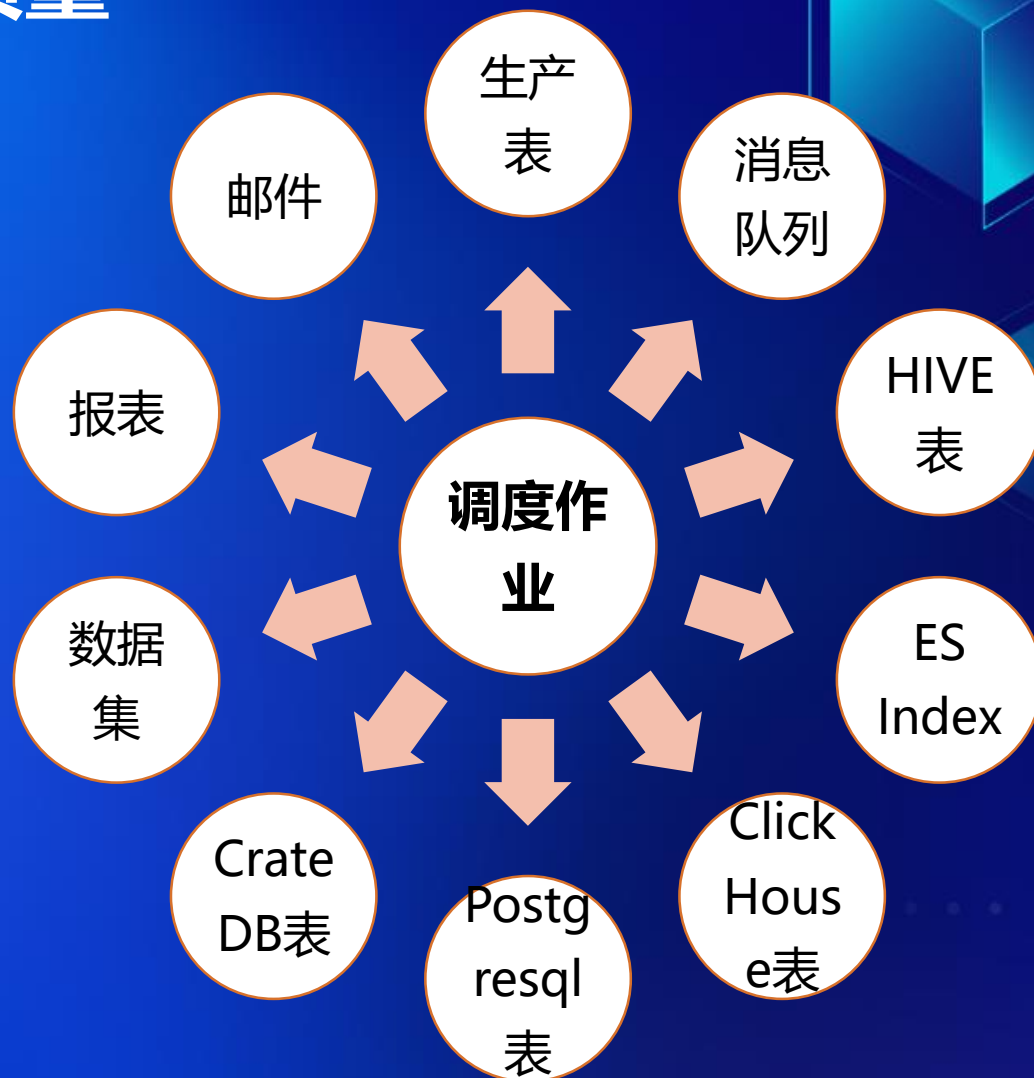
安全标签

重要等级标签

业务标签

数据质量

报表为啥又不对?



数据质量

```
[Stage 18:> (0 + 0) / 84]
[Stage 18:> (0 + 6) / 84][Stage 19:> (0 + 0) / 8]
[Stage 18=> (5 + 3) / 84][Stage 19:> (0 + 3) / 8]
[Stage 18=> (6 + 3) / 84][Stage 19:> (0 + 3) / 8]
[Stage 18=> (9 + 4) / 84][Stage 19:> (0 + 4) / 8]
[Stage 18=> (9 + 6) / 84][Stage 19:> (0 + 4) / 8]
[Stage 18=> (11 + 5) / 84][Stage 19:> (0 + 5) / 8]
[Stage 18=> (12 + 7) / 84][Stage 19:> (0 + 5) / 8]
[Stage 18=> (12 + 9) / 84][Stage 19:> (0 + 7) / 8]
[Stage 18=> (12 + 14) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (14 + 18) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (15 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (17 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (18 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (23 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (27 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (34 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (36 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (42 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (47 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (53 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (60 + 20) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (65 + 19) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (71 + 13) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (75 + 9) / 84][Stage 19:> (0 + 8) / 8]
[Stage 18=> (83 + 1) / 84][Stage 19:> (0 + 8) / 8]
[Stage 19:> (0 + 8) / 8]
[Stage 19=> (1 + 7) / 8]
[Stage 19=> (2 + 6) / 8]
[Stage 19=> (3 + 5) / 8]
[Stage 19=> (4 + 4) / 8]
[Stage 19=> (5 + 3) / 8]
[Stage 19=> (6 + 2) / 8]
[Stage 19=> (7 + 1) / 8]
[Stage 21:> (0 + 2) / 4]
[Stage 21=> (2 + 2) / 4]
[Stage 21=> (3 + 1) / 4]
[Stage 31:> (0 + 2) / 2]
[Stage 31=> (1 + 1) / 2]
```

```
19/10/27 03:45:39 INFO SparkSqlParser: Parsing command: `olap_flt`
19/10/27 03:45:40 [main] WARN InsertIntoHiveTable: Partition `olap`
```

```
Hadoop job information for Stage-123: number of mappers: 4; number of reducers: 0
```

```
2019-10-28 01:16:38,621 Stage-123 map = 0%, reduce = 0%
2019-10-28 01:17:39,016 Stage-123 map = 0%, reduce = 0%, Cumulative CPU 226.41 sec
2019-10-28 01:18:18,561 Stage-123 map = 25%, reduce = 0%, Cumulative CPU 388.72 sec
2019-10-28 01:19:18,981 Stage-123 map = 25%, reduce = 0%, Cumulative CPU 570.57 sec
2019-10-28 01:20:00,522 Stage-123 map = 50%, reduce = 0%, Cumulative CPU 706.4 sec
2019-10-28 01:20:21,361 Stage-123 map = 63%, reduce = 0%, Cumulative CPU 753.35 sec
2019-10-28 01:20:30,699 Stage-123 map = 88%, reduce = 0%, Cumulative CPU 773.8 sec
2019-10-28 01:21:31,254 Stage-123 map = 88%, reduce = 0%, Cumulative CPU 835.23 sec
2019-10-28 01:22:31,407 Stage-123 map = 88%, reduce = 0%, Cumulative CPU 900.54 sec
2019-10-28 01:23:32,312 Stage-123 map = 88%, reduce = 0%, Cumulative CPU 964.66 sec
2019-10-28 01:23:34,386 Stage-123 map = 100%, reduce = 0%, Cumulative CPU 966.41 sec
MapReduce Total cumulative CPU time: 16 minutes 6 seconds 410 msec
Ended Job = job_1569492437585_5442342
Stage-183 is selected by condition resolver.
Stage-22 is filtered out by condition resolver.
set heap size 2048MB
Execution log at: /tmp/hadoop-hive/hi-123/20191028004848_9d7db6aa-7f08-4fe0-9eff-a30ae97ab08f.log
```

```
2019-10-28 01:23:43
2019-10-28 01:23:45
2019-10-28 01:23:45 Upload
2019-10-28 01:23:45 End of local task; time taken: 1.281 sec.
```

```
Execution completed successfully
MapredLocal task succeeded
```

```
Launching Job 46 out of 50
```

```
Number of reduce tasks to get: 0 since
```

```
Starting Job = job_1569492437585_5443289
```

```
CONSOLE# Dr Elephant
```

```
Kill Command = /usr/bin/kill -9 job_1569492437585_5443289
```

```
Hadoop job information for Stage-121: number of mappers: 5; number of reducers: 0
```

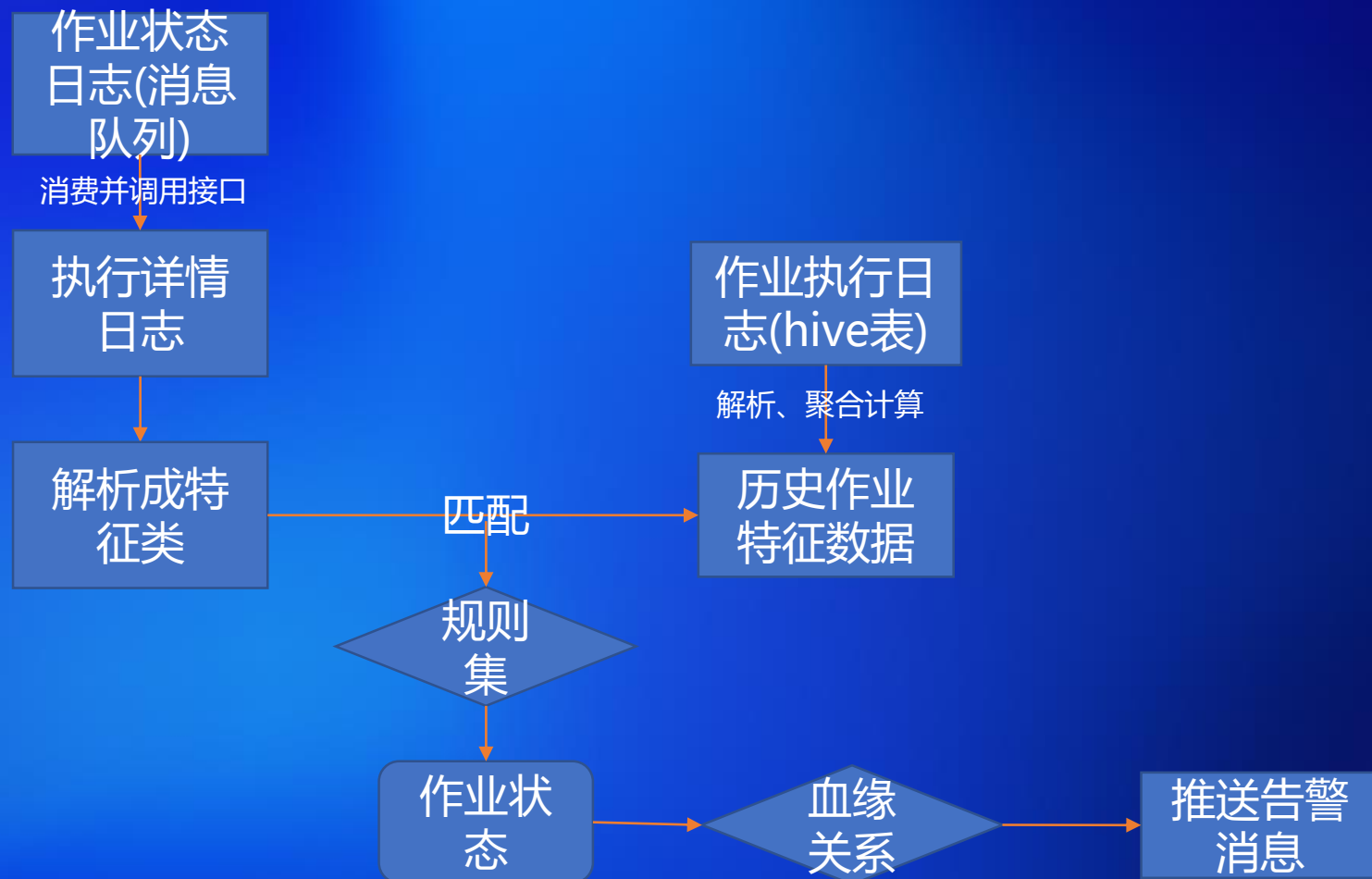
```
2019-10-28 01:24:09,629 Stage-121 map = 0%, reduce = 0%
2019-10-28 01:25:08,933 Stage-121 map = 20%, reduce = 0%, Cumulative CPU 333.1 sec
2019-10-28 01:25:15,132 Stage-121 map = 40%, reduce = 0%, Cumulative CPU 363.74 sec
2019-10-28 01:25:54,699 Stage-121 map = 60%, reduce = 0%, Cumulative CPU 508.57 sec
2019-10-28 01:26:08,166 Stage-121 map = 90%, reduce = 0%, Cumulative CPU 544.08 sec
2019-10-28 01:27:08,229 Stage-121 map = 90%, reduce = 0%, Cumulative CPU 610.68 sec
2019-10-28 01:27:51,613 Stage-121 map = 100%, reduce = 0%, Cumulative CPU 663.69 sec
MapReduce Total cumulative CPU time: 11 minutes 3 seconds 690 msec
Ended Job = job_1569492437585_5443289
Stage-25 is selected by condition resolver.
Stage-24 is filtered out by condition resolver.
Stage-26 is filtered out by condition resolver.
```

```
Moving data to: hdfs://ns/user/biuser/warehouse/etl/Tmp_FltDB.db/hive-staging_hive_2019-10-28_00-48-47_081_2221848282308500031-1/-ext-10002
Move
Tab
Mapk
Stage-Stage-105: Map: 1 Reduce: 1 Cumulative CPU: 6.98 sec HDFS Read: 37922 HDFS Write: 2745 SUCCESS
Stage-Stage-103: Map: 14 Reduce: 10 Cumulative CPU: 944.0 sec HDFS Read: 1182773312 HDFS Write: 612424717 SUCCESS
```

增量字节
数

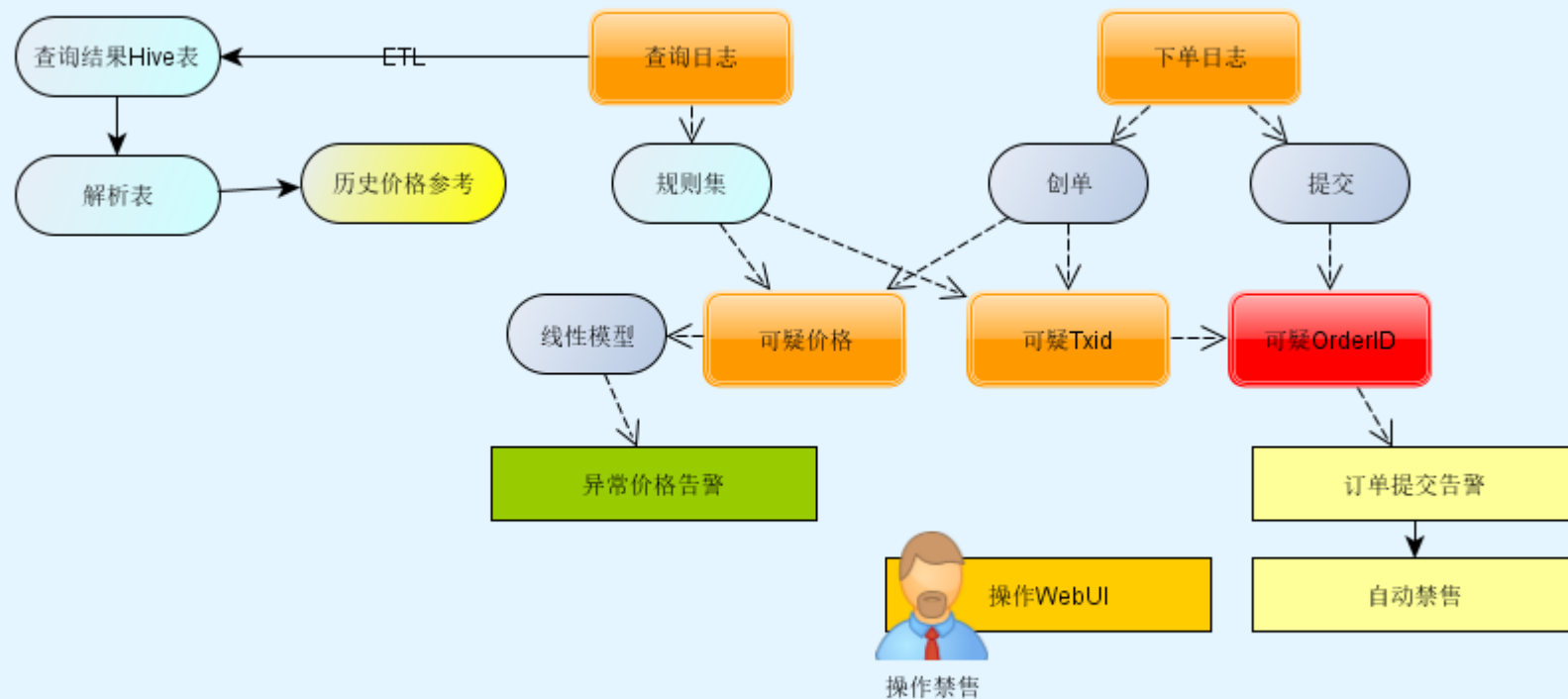
增量字节数

数据质量检验流程



数仓应用-价格监控

机票价格监控系统



- 毫秒级响应
- 发现数十起价格异常

数仓应用-日志追踪

Service-A

Service-B

Service-C

Topic-A

Topic-B

Topic-C

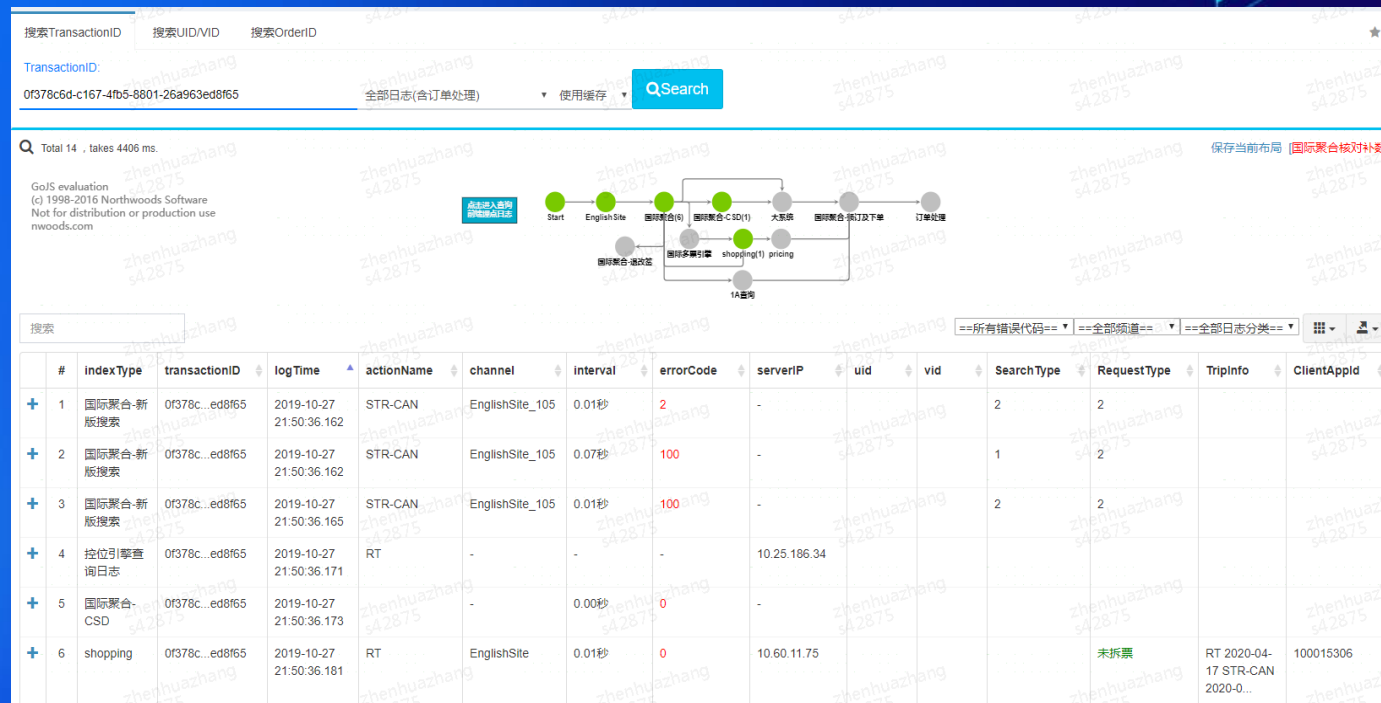
Flink

Txid:[Date List]

同步, topic+date=>index

Redis

ElasticSearch



数仓应用-报表取数

统一数据源

- 自身准确性
 - 业务规则校验
 - 历史数据规律
 - 其他数据源佐证

口径

- 建立不同口径取数模板
- 标准化口径管理

临时取数

- 中心化需求描述、脚本及数据
- 交叉review

报表管理

- 分类
- 僵尸报表识别
- 埋点



数仓规范

命名规范

- 正式/临时 库/表
 - 正式表: {层级}_{类别}_{业务}_{描述}
 - 临时表: tmp_{编号}_{业务}
- 视图
 - v_{相关表名}
- 流程名称
 - 与写入表名一致

脚本规范

- 建表语句
 - comment
- 脚本描述
 - 功能
 - 变更历史
 - 是否支持无改动刷数
- 提行
- 注释

检测工具

规范文档、维护及培训

数仓面临的挑战

非数据生产者，但必须理解数据

生产数据逻辑变更没法事先主动获知

貌似做不完的数据需求

数据源/种类繁多

历史遗留问题

流程过多，维护成本超高，僵尸报表难以辨别，资源浪费

可能带来的问题

1. 需求繁杂琐碎，资源碎片化
2. 数仓人员永远都在被动承接需求状态，无积极性和成就感
3. 人员晋升乏力，流失严重
4. 需求响应不及时，数据部门和需求方互相吐槽

数仓团队能做的

1. 让数仓工程师了解一类业务并熟悉其数据
2. 取数口径、脚本及数据定期分享
3. 数仓规范严格执行，对历史遗留问题做到不增加，有计划有节奏地去除
4. 自定义函数封装公用口径
5. 开发数仓实用工具，提升数据开发的效率
 - ① 脚本查询
 - ② 赋权流程
 - ③ 数据解析工具
 - ④ SQL助手
 - ⑤ 维表导入
6. 培训数据需求方SQL技能
7. 主动了解各需求方看数据视角，总结凝练，建设底表及可视化系统



本PPT来自2019携程技术峰会
更多信息请关注“携程技术中心”微信公众号~