DataSpace系统开发流程分析

> Q：dataSpace是什么?
>
> A：一个指标数据开发和管理系统

1、数据的重要性理解

星图作为一个交易平台，本质上也是一个售卖平台。数据是平台最重要的一趴，平台需要依赖投前数据吸引客户下单，需要依赖投中数据和投后数据度量所花的钱是否值，从而吸引客户复投。平台真正售卖的就是数据。类似于在其他电商平台买东西，当东西到手后我们会有感受，这东西好不好，这钱花的值不值。而星图平台就需要依赖数据来给客户这种反馈。

2、DataSpace之前数据开发的问题

- 开发流程

  数据来源

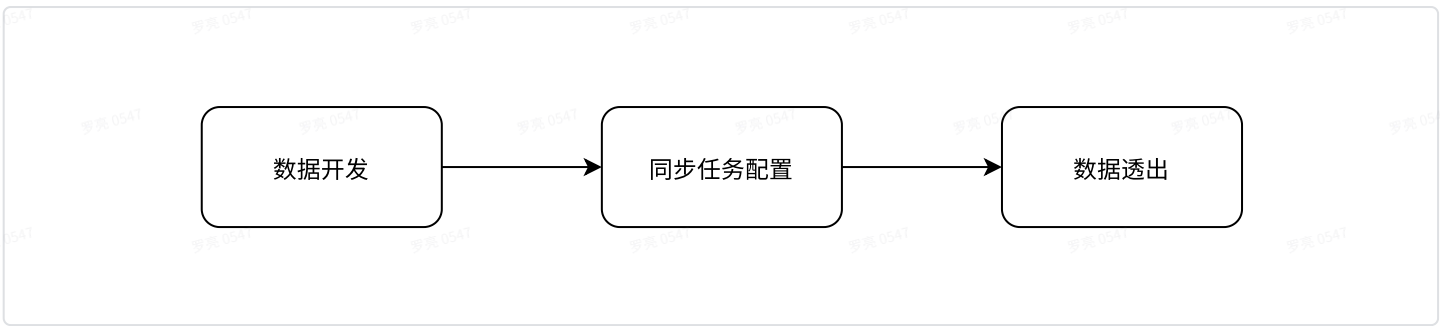
  service->mysql->hive->hive->mysql->service

这个流程中重要的点，是要依赖hive的算力实现数据分析（数仓和数据库的区别）

- 带来的问题（解决的方法）
  - 每天对mysql高读写访问（根据业务场景选择合适的存储）
  - 占用mysql存储空间（根据业务场景选择合适的存储）
  - 数据迭代带来的mysql表的频繁修改和DAO层代码的不断维护（设计统一的数据模型，提供统一的数据查询接口）
  - 数据的重复建设和口径不一（制定数据指标规范）
  - 数据的溯源困难和不易维护（提供平台用于任务配置）

3、DataSpace开发流程

> 仅针对我所熟知的离线数据开发进行分析，实时数据、Redis和es存储还不太了解

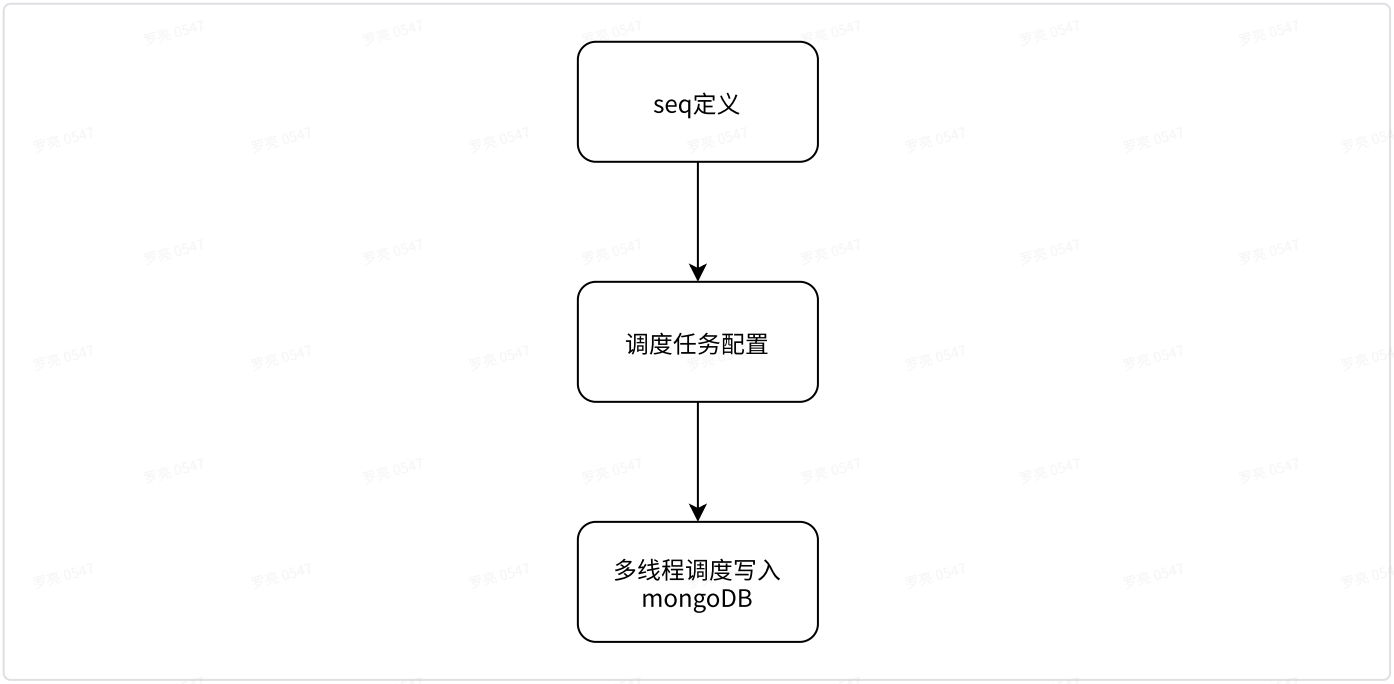


- 数据开发

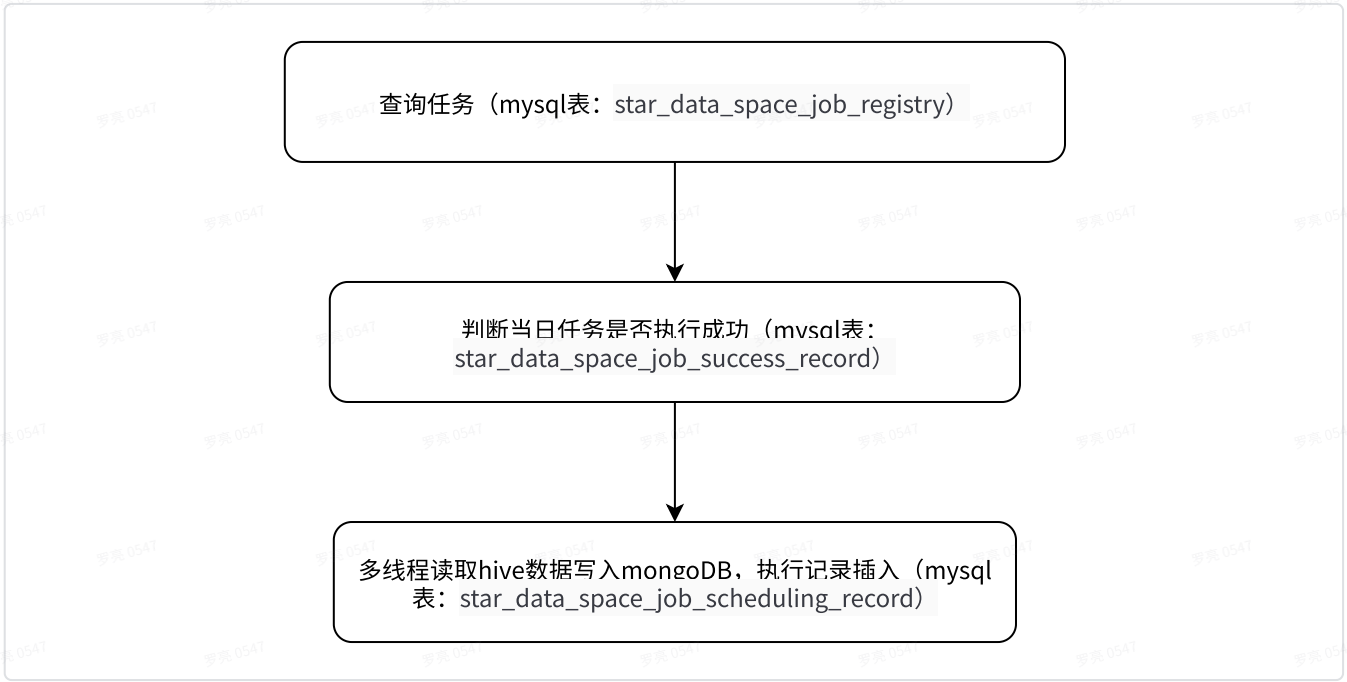
目前以hive开发为主，会写SQL（HSQL）就够了，目前用到的表都是数仓同学开发的成表，一般情况下我们只需做**数据集成**或者一些**简单指标计算**

Hive的基础概念和开发原则

- 同步任务配置
  - 开发流程



  - 任务调度流程



    - 任务查询线程数为**10**
    - 数据按**百万行**划分多个线程执行

取值应该和资源有关

- 数据透出

```
service AdStarDataSpaceService {
    GetDataItemFieldsResp GetDataItemFields(1: GetDataItemFieldsReq req)
    MGetDataItemFieldsResp MGetDataItemFields(1: MGetDataItemFieldsReq req)
    GetDataItemTimelineFieldsResp GetDataItemTimelineFields(1: GetDataItemTime
    MGetDataItemTimelineFieldsResp MGetDataItemTimelineFields(1: MGetDataItemT
    GetTdDataItemFieldsResp GetTdDataItemFields(1: GetTdDataItemFieldsReq req)
    MGetTdDataItemFieldsResp MGetTdDataItemsFields(1: MGetTdDataItemFieldsReq
    GetTdDataItemTimelineFieldsResp GetTdDataItemTimelineFields (1: GetTdDataI
    MGetTdDataItemTimelineFieldsResp MGetTdDataItemTimelineFields (1: MGetTdDa
}
```

  - 设计点
    - 批量查询最多20个：避免查询数据量过大，导致接口响应时间过长而超时
    - 布隆过滤器 空值筛选（已下线）
    - redis做缓存（已下线）
      - 下线原因，mysql存储向mongoDB存储切换，mongoDB提供了缓存机制，因此不需要redis来做缓存了

4、DataSpace离线数据存储选型

参考：data_space 存储迁移技术方案

- 目前已有指标数据的模型：

item_id,item_type,seq,(place) -> value 普通指标

item_id,item_type,rel_id,rel_type,seq,(place)->value 二维叉乘

item_id,item_type,seq,(place),start_date,end_date -> value 普通增量指标

item_id,item_type,rel_id,rel_type,seq,(place),start_date,end_date->value 二维叉乘增量

- 数据特点
  - 本质上是key-value,换而言之就是查询条件是固定的
  - 表的设计只对指标类型区分，需四张表存下所有指标数据
- mysql存储的瓶颈
  - **查询效率**，随着数据量越大，查询效率越低，联合索引能解决一定查询效率问题
  - **数据量大**，不易拓展，按目前数据模型，未来所有指标都只能由四张表存下，数据量大后，分库分表的代价较大，且关系型数据库，对表结构的修改（数据模型）需慎重
- mongodb的优势
  - 分片集群
    - 目的
      - 读写能力提升
      - 存储容量扩大
      - 高可用性
    - 分片策略
      - 分片键
      - 哈希分片
      - 范围分片
  - 复制集
    - 目的
      - 高可用性
  - 缓存机制（WT)
    - 目的
      - 利用内存空间提高读写效率
  - NoSQL的扩展能力
    - 目的
      - 易于数据模型拓展
- mysql的什么特性是dataSpace场景不需要的?

  mysql对事务的支持，在目前dataSpace的业务场景下，是完全没必要的，既不涉及连表查询，也不涉及频繁读写交替

  而且mongodb目前也支持事务了，对于DataSpace来说，可以有但没必要
- 为什么不选用redis或者Abase

不选用redis是因为内存资源太贵了，无法应对目前指标数据量庞大的问题

不选用abase是因为增量指标的存在，对于增量指标，可能需要查询长时间内的范围指标，对于key的设计需要把date带上，大范围查询对性能有一定影响（说白了就是不支持范围查询，要实现范围查询要依赖key的设计）