

YOH0-SEARCH

说明文档

版本： V1.0

编写时间： 2016 年 6 月 3 日

修改历史

版本	时间	内容描述	修改人
1.0	2016/6/3	文档创建	王楠

1.数据库模型

是同步商品那边的的表，表结构不再详细介绍。

2.代码结构

yoho-search-core 搜索服务的公共核心模块，包括 dal 和 core

--bin maven 操作脚本

--core

-----com.yoho.search.dal.service mybatis 服务类

 --index 存放索引相关类

 -----config 配置文件实体

 -----es es 客户端

 -----service 索引相关接口与实现

 --task 任务描述和任务分配

 --utils 工具类

-----resources esmapping json 文件

--dal

-----com.yoho.search.dal 实体与 mybatis mapper

-----resources mybatis 配置文件

yoho-search-producer 生产者模块

--bin maven 操作脚本

--deploy 部署脚本

--producer-web 业务逻辑代码

-----com.yoho.search.canal 阿里 mysql 数据库 binlog 的增量订阅&消费组件

 ---mq 消息队列 FastJson 转换器

-----resources spring canal rabbitmq 等配置文件

yoho-search-consumer 消费者模块

--bin maven 操作脚本

--deploy 部署脚本

--consumer-web 业务逻辑代码

-----com.yoho.search.job 索引重建定时任务

 ---mq mq 监听器

 ---service 索引请求相关 controller

-----resources spring 等配置文件

yoho-search-service 提供搜索服务模块

--bin maven 操作脚本

--deploy 部署脚本

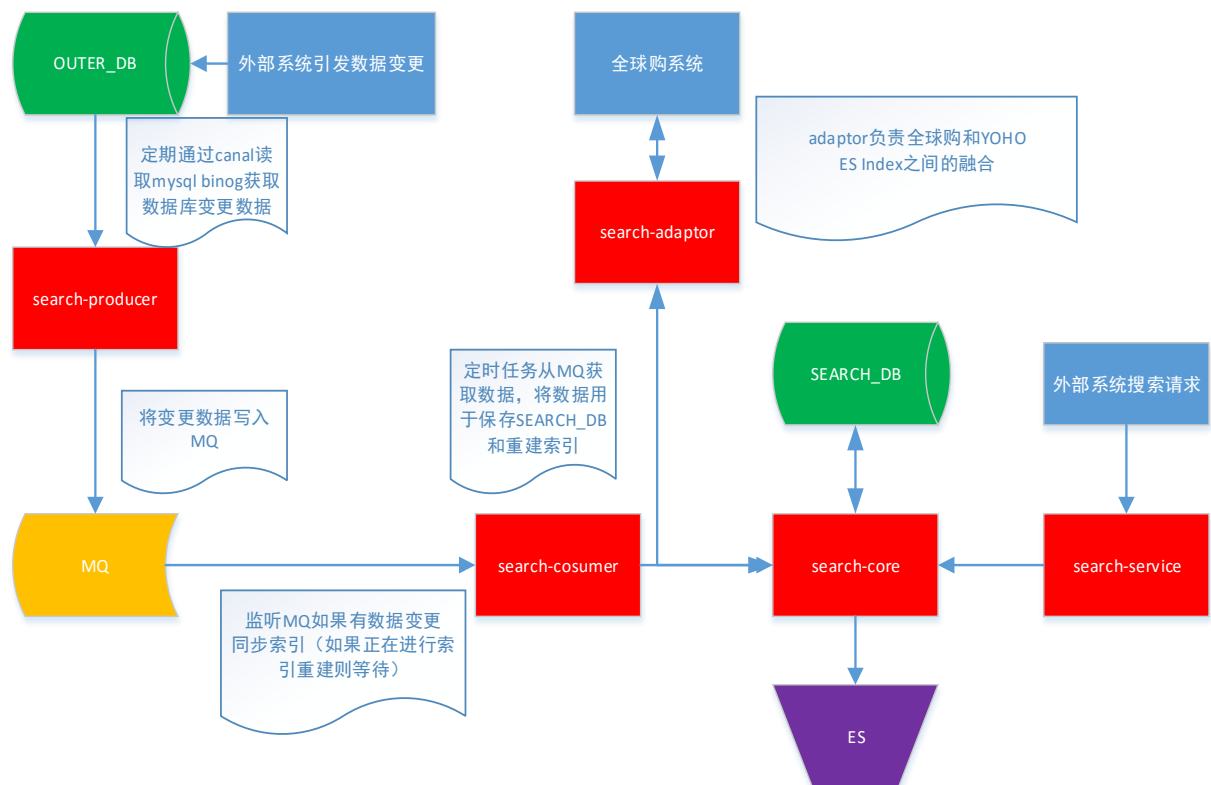
--search-service 业务逻辑代码
 -----com.yoho.search.restapi 存放 controller 类
 --- service 存放 service 类
 ---vo 存放 vo 实体
 -- search-service -web web 壳子，存放 spring 等配置文件

yoho-search-adaptor 全球购和 yoho 商品适配

--bin maven 操作脚本
 --deploy 部署脚本
 --adaptor-service 业务逻辑代码
 -----com.yoho.adaptor.model 存放实体类
 --- restapi 存放 controller
 ---service 存放 service
 ---util 存放工具类
 -- search-adaptor -web web 壳子，存放 spring 等配置文件

3.业务逻辑

总架构图



producer 模块

生产者模块，分布式部署

search-producer处理流程

类：CustomSimpleCanalClient
方法：afterPropertiesSet()
在spring容器加载的时候调用，判断这个producer是master还是slave，如果是master则启动client（30s循环一次）



类：CustomSimpleCanalClient
方法：startClient()
--> 1.startShopClient(); startOperationClient();
--> 2.handleCanalMessage ()
--> 3.publishToRedis()（此方法名有点问题）

订阅的表：

yh_shops.brand,yh_shops.goods,yh_shops.goods_images,yh_shops.product,yh_shops.product_color,yh_shops.product_price,yh_shops.product_sort,yh_shops.product_standard_relation,yh_shops.product_style_relation,yh_shops.size,yh_shops.storage,yh_shops.standard,yh_shops.style,yh_shops.product_activities_link,yh_shops.product_search,yh_shops.parameter_make

启动canal client服务 采用异步线程方式
生产数据到内存队列

- 1.启动client
- 2.获取数据
- 3.数据操作记录保存进mq

consumer 模块

消费者模块

search-consumer处理流程

定时任务1 每日索引重建

indexRebuildJobMethod

频率：每天3点

执行：类-->IndexRebuildJob

方法-->execute()

execute()

--> rebuildIndexWithlog() 重建所有索引：

--> 1.rebuildFlagService.waitingRebuildingIndex()

--> 2. 【search-core模块】indexService.rebuild();

1.判断是否在rebuilding，如果在，则等待20000ms
2.执行索引重建（重建策略：1. 建一个新索引 2. 向新导入数据 3. 删除原索引的别名 4. 为新索引增加别名 目前重建可以先停掉canal，暂停同步更新，但不会影响线上搜索服务。）

rebuild()详情

3.重建数据详情

类：xxxIndexBuilder

方法：buildAll()

1.从数据库读取数据

2.调用beanToMap方法转换成es接受的数据结构

3.向tempIndex中添加数据

0.标志更新索引状态为true

1.创建临时索引

2.针对批量插入数据，进行索引的优化设置

3.重建数据

4.恢复原来的索引设置

5.索引替换：将老索引的索引名加到新索引上，并删除老索引

6.标志更新索引状态为false

定时任务2 每小时重建tblproductIndex

rebuildTblProductIndex

频率：每天小时的第17分钟

执行：类-->IndexRebuildJob

方法--> rebuildTblProductIndex()

--> rebuildIndexWithlog() 重建索引：tblproduct

--> 1.rebuildFlagService.waitingRebuildingIndex()

--> 2.indexService.rebuild();

定时任务3 全球购product到yoho productIndex 适配

SyncTplProductIndex

频率：每小时的第30分钟

执行：类-->IndexRebuildJob

方法--> syncTplProductIndex()

--> 1.rebuildFlagService.waitingRebuildingIndex()

--> 2.indexService.syncTplDateToIndex();

--> 0.调用xxxIndexBuilder 中的syncIndex()（目前只有productIndexBuilder中的有逻辑）

--> 1.调用search-adaptor 中的接口：

TplProductAdaptorController.adaptorTplToYoho 获取数据

--> 2. 调用beanToMap方法转换成es接受的数据结构

--> 3.向xxxindex 中添加数据

MQ监控

xxxMqListner

---> onMessage()

---> updateData()或deleteData()-->DB and ES(清除索引中数据)

updateData()

---> saveOrUpdate();---> DB

---> updateXXXIndex() ----> ES

adaptor 模块

全球购和 yoho 商品双向适配

search-adaptor模块
开放了两个restAPI

目前全球购和yoho商品在搜索引擎中的索引是不同的，分别是tplproduct 和 productindex。由于productindex的字段远远多于tplproduct的字段，所以整融合过程会分为几个阶段，循序渐进来完成2个商品共用一个索引。
适配模式。通过开发适配器，把全球购的字段转化成yoho商品的字段，全球购商品更新到productindex中

适配器设计

1 全球购商品融合到productindex

(1) 适配器通过调用全球购的url+key获取全球购商品，参考：

wget

`${tbl.product.url}getproductscountforsearch?client_type=iphone&client_secret=xx`

其中xx为客户端key，需要进行md5的转化，转化如下：

`clientkey= MD5Util.string2MD5("client_type=iphone&private_key="+
${tbl.product.private.key});`

`${tbl.product.url}`:全球购url，可以找周硕要

`${tbl.product.private.key}`: 全球购产品密钥

(2) 获取全球购产品后，需要参考yoho商品productindex 封装成商品list。

`List<ProductIndex> globalProducts = ProductAdapotr.changed(List<tplproduct>);`

(3) 搜索这边会每隔一段时间（默认间隔为1个小时）获取globalProducts，之后会进行product_index的更新，这样全球购商品就能在product_index里面搜索得到。

2 yoho部分商品（部分品牌下的商品）融合到全球购索引 商品适配模块需要支持逆向适配，其中：（1）部分brand品牌下的商品需要按照全球购的字段转成全球购商品列表。可直接从yh_shops库product表中根据brand查到对应的product，然后按照全球购的字段转成全球购的商品列表

`List<product> = XXservice.getProduct(String brandname);`

`List<tplproduct> yohoProducts =`

`ProductAdapotr.changedfromProductIndex(List<product>);`

(2)全球购对着部分商品列表进行存储计算。（3）搜索引擎会定时从全球购获取商品列表，并进行tplproduct更新的操作。

注：全球购同步过来的数据，skn，id，productid 都是负数

service 模块

简单搜索代码逻辑:

1. 拼装 SearchParam

- > **setFilter 过滤参数**
 - >constructFilter ()
 - > 使用 ES 包中的 BoolFilterBuilder, 常用的方法 must() mustNot() should()
 - >上面的方法中塞入 FilterBuilders.termsFilter(field, values)
- > **setQuery 查询参数**
 - > QueryBuilders.matchAllQuery()
 - 或--- >constructQuery ()
 - > 使用 ES 包中的 MultiMatchQueryBuilder, 常用的方法 operator() field()
- > **setAggregationBuilders 聚合参数**
 - > constructAggregations()
 - >使用 List<AbstractAggregationBuilder> list
 - >使用 AggregationBuilders 常用方法: terms() field() size() subAggregation() order() addRange() addHits addSort()
- > **setSearchType 查询类型**

2. 设置查询条数

3. 设置查询结果返回字段

4. 设置排序字段

5. 执行查询 doSearch(indexName, searchParam)

6. 将 searchResult 转化为 map 返回--需要转化为需要的结构

7. 返回 jsonMap

开放 RESTAPI 接口提供搜索服务, 包括搜索数据的封装和处理



搜索文档整理【持续更新中】.pdf

参数和接口详见胡古飞写的文档

core 模块

包括 mybatis 服务, 调用 es 客户端逻辑, indexBuilder 等底层逻辑

4.搜索相关文档地址

公司文档项目: <http://git.yoho.cn/tanling/api-interfaces/tree/master/%E6%90%9C%E7%B4%A2>

ES 权威指南: <http://es.xiaoleilu.com/>