**Flink 实时数据仓项目设计与实现总结**

**一、项目背景**

**本项目旨在通过 Flink 构建一套实时数据仓库系统，打通 Kafka、Flink、HBase、ClickHouse 等组件，实现从用户行为日志、订单行为数据中抽取核心指标，支撑可视化大屏分析。**

**通过实时计算订单、支付、退款、用户行为等指标，商家可以实时监控业务趋势、分析用户行为偏好、定位转化瓶颈，进一步实现精细化运营与业务优化。**

**二、项目技术栈**

**技术栈与工具汇总**

|  |  |  |
| --- | --- | --- |
| 类别 | 技术 / 工具 | 说明 |
| 数据源 | Kafka、MySQL、Debezium | 行为日志 / 事务数据来源 |
| 流处理 | Apache Flink 1.17 | 实时计算核心组件 |
| 存储 | HBase、ClickHouse | 维度存储 / 明细指标落地 |
| 开发语言 | Java、Flink SQL | 主体逻辑编写 |
| 可视化 | Finereport | 实时大屏可视化 |
| 其他 | ClickHouseUtil、HbaseUtils | 异步缓存 / 工具封装类 |

**三、数据分层设计**

**1. ODS (原始数据)**

* 来源：Kafka 消费日志、MySQL 事务表 CDC
* 处理：使用 Flink SQL 进行消息格式化、后续搜集
* **重点：解析 Debezium 格式的 before/after 字段**：用于判断操作类型（INSERT/UPDATE/DELETE）

**2. DWD (事件明细层)**

* 按类型抽取日志：start/page/display/action/error
* 对订单、支付、退款三类效能实现 **order\_detail\_id 级别聚合容错效系列处理**
* 配合 HBase + 异步 join 的方式 enrich

**3. DIM (维度层)**

* 源数据库：MySQL
* 工具类：HbaseUtils
* 共合类：AsyncDimFunction，支持加盐rowkey
* 给 6 维度表：user\_info, sku\_info, spu\_info, trademark, category3, province

**4. DWS (主题层并行并聚合)**

* 行为合并：各种操作日志 (display/action/page/start)
* 订单合并：将三类流缓存经历后 join 聚合成一条 order\_detail 字段级数据
* 产生 enrich 并给后续 ADS 写 ClickHouse

**5. ADS (指标展示层)**

* 通过下列 Flink job 输出给 ClickHouse
  + AdsTradeStatsWindowJob
  + AdsUserBehaviorWindowJob

**四、指标设计与求值逻辑**

**1. 商品曝光/点击**

* 来源：dwd\_display\_log
* 计算：
  + 曝光 PV: 每条 display 记录计 1
  + 点击: item\_type 为 sku\_id 且 is\_click = 1
* 性能：通过产品分组进行数据统计

**2. PV / UV**

* 来源：dwd\_page\_log
* 分类：按应用版本 VC 统计
* 算法：
  + PV: 全部访问量
  + UV: 根据 mid 去重

**3. 会话**

* 来源：dwd\_page\_log
* 算法：
  + 每次新的 sid 记为一次会话
  + 统计 sid 数量和均值持续时长

**4. 实时订单**

* 来源：dwd\_order\_detail
* 按 SKU ID 分类
* 算法：
  + order\_ct 订单条数
  + order\_user\_ct 用户数
  + sku\_num 商品件数
  + order\_amount 总金额

**5. 实时支付**

* 来源：dwd\_payment
* 算法同上，计算 payment\_ct / payment\_user\_ct / payment\_amount

**6. 实时退款**

* 来源：dwd\_order\_refund
* 算法同上，退款数据添加 refund\_ct / refund\_user\_ct / refund\_amount

**7. Enriched 指标**

* 给 ClickHouse 的 enriched\_trade\_stats 表，根据 sku\_id 级分析产品全链路求值效果

**五．大屏指标与表结构介绍**

**1. 商品曝光与点击统计表（柱状图）**

来源表：ads\_user\_display\_click\_window

* stt, edt: 时间窗口开始/结束时间
* sku\_id: 商品 ID
* page\_id: 页面 ID
* ch: 渠道（如 web、小米、Appstore）
* disp\_ct: 曝光次数（由 display 数据窗口聚合得出）
* disp\_sku\_num: 曝光商品数

**2. 用户访问量统计表（PV/UV）**

来源表：ads\_user\_pv\_uv\_window

* stt, edt: 时间窗口
* vc: 版本号
* ch: 渠道
* ar: 地区
* is\_new: 是否新用户标识
* pv\_ct: 页面访问量（PageView）
* uv\_ct: 独立访客数（Unique Visitor）

**3. 用户会话行为统计表（饼图）**

来源表：ads\_user\_session\_window

* stt, edt: 时间窗口
* vc, ch, ar, is\_new: 用户维度
* sv\_ct: 会话数
* uj\_ct: 跳出会话数（无有效行为）

**4. 实时订单统计表（折线图）**

来源表：order\_stats

* stt, edt: 时间窗口
* sku\_id: 商品 ID
* order\_ct: 订单数
* order\_user\_ct: 下单用户数
* sku\_num: 下单商品件数
* order\_amount: 订单金额

**5. 实时退款统计表（气泡图）**

来源表：refund\_stats

* stt, edt: 时间窗口
* sku\_id: 商品 ID
* refund\_ct: 退款笔数
* refund\_user\_ct: 退款用户数
* refund\_amount: 退款金额

**6. 实时支付统计表（横向条形图）**

来源表：payment\_stats

* stt, edt: 时间窗口
* sku\_id: 商品 ID
* payment\_ct: 支付笔数
* payment\_user\_ct: 支付用户数
* payment\_amount: 支付金额

**7. 综合交易指标统计表（enriched\_trade\_stats）**

来源表：enriched\_trade\_stats

* 综合聚合了订单、退款等信息字段，用于多指标综合分析展示：
  + order\_ct, sku\_num, order\_amount
  + refund\_ct, refund\_user\_ct, refund\_amount

**六．主题域划分**

|  |  |  |
| --- | --- | --- |
| 主题域 | 说明 | 对应数据源 / 表 |
| 用户行为域 | 分析 PV、UV、会话、跳出 | dwd\_page\_log、dwd\_display\_log、dwd\_action\_log |
| 商品行为域 | 曝光、点击、加购、收藏 | dwd\_display\_log（点击）、dwd\_cart\_info（加购） |
| 交易订单域 | 下单、支付、退款、优惠券使用 | dwd\_order\_detail、dwd\_payment\_info、dwd\_order\_refund\_info |
| 用户维度域 | 用户画像、地域信息 | dim\_user\_info、dim\_province |
| 商品维度域 | 商品、品类、品牌 | dim\_sku\_info、dim\_spu\_info、dim\_base\_category3、dim\_base\_trademark |
| 地域分析域 | 地区维度订单/用户分布 | dim\_province + 订单表聚合 |
| 会话分析域 | 用户访问会话、停留时长 | dwd\_page\_log 中 session\_id 聚合 |

**全量表与增量表设计说明**

|  |  |  |  |
| --- | --- | --- | --- |
| 表类型 | 说明 | 示例表 | 特点 |
| 全量维度表 | 存储基础维度信息，随时间更新不频繁 | dim\_user\_info、dim\_sku\_info | HBase 存储，Flink 异步 join |
| 增量事实表 | 存储实时流入的业务事件数据 | dwd\_order\_detail、dwd\_payment\_info、dwd\_display\_log | Kafka → Flink 消费，每条为一次事件 |
| 中间宽表 | 拉宽后融合信息的宽表 | dwd\_order\_detail\_wide、dws\_user\_behavior | 多表 join 后的数据汇总 |
| 指标汇总表（ADS） | 按窗口输出指标，供大屏展示 | ads\_user\_pv\_uv\_window、ads\_trade\_stats | 按小时 / 分钟写入 ClickHouse，支持 BI 展示 |

**六、ClickHouse 优缺点分析**

**优点**

* 高压缩 + 列序存储，性能极高
* 适合 OLAP 场景
* 支持多端前端 BI 系统（如 FineBI）

**缺点**

* 不适合 OLTP 性能，更新支持弱
* 不适合很复杂的形成模型

**七、重点难点与解决方案**

|  |  |
| --- | --- |
| 难点 | 说明 |
| Debezium JSON 格式 | 前端 after/before 格式处理复杂，需要分析 op 字段，转成 Flink 可用 JSON |
| Flink 异步维表关联 | 自定义 AsyncDimFunction 、HbaseUtils，加盐 rowkey，减少红光 join 的性能压力 |
| Kafka 缓存分流 | 将行为、订单、支付、退款分流 topic，避免数据交叉形成崩溃 |
| 空值问题 | 应对不完整的订单、退款重新识别聚合逻辑 |

**八．关键处理逻辑说明**

**1. Kafka 数据格式**

* 日志类数据：原始 JSON，含 event\_type
* CDC 数据（Debezium）：包含 before, after, op 字段，需要反序列化成 Row 类型处理

**2. 维度表异步关联（Async Join）**

* 使用 Flink Async I/O + 自定义 AsyncDimFunction 实现异步查询 HBase
* 维度数据使用 Guava Cache 本地缓存，提升吞吐性能，避免频繁请求 HBase

**3. Flink 窗口统计**

* 所有聚合指标均基于事件时间（Event Time）
* 使用 Tumble、Slide、Session Window 对用户行为做窗口处理（如 UV 去重、曝光累计）

**4. 多流合并（union/order\_detail + pay + refund）**

* 在 DwdOrderDetailEnrichJob 中通过 order\_detail\_id 做数据拉宽，关联订单、支付、退款三条流
* 确保输出数据字段齐全、指标无空值

**5. 指标一致性保障**

* 所有指标基于事件时间
* 异步维表拉取 + 状态管理 + 精确一次语义 + Checkpoint
* 抽象通用 OutputTag 捕获脏数据（异常 JSON、维度缺失）

**涉及知识点分类**

**Flink 流处理基础**

* DataStream API / Table API / Flink SQL
* Watermark / EventTime / 状态管理
* Tumble/Sliding/Session Window
* Broadcast State、SideOutput

**异步维表关联（Async Join）**

* Async I/O
* 自定义 AsyncFunction
* HBase 查询与缓存优化

**数据解析与清洗**

* JSON 解析（Kafka → Debezium 格式）
* 字段规整、类型转换
* 异常数据捕获（脏数据处理）

**宽表建模与多流合并**

* KeyBy order\_detail\_id
* 支付流、退款流 join 聚合
* null 值处理、补偿机制

**数据落地与可视化**

* ClickHouse 建表规范（MergeTree、分区、排序键）
* FineVis 组件绑定
* 指标 → 图表映射关系构建

**总结**

本项目实现了一套高效的实时数据分析链路，支撑了电商业务中多维度、多指标的实时可视化看板。整体流程覆盖了从数据采集、加工、清洗、计算、维表增强、指标聚合、数据入仓，再到最终可视化展示的完整链路。

后续可以继续拓展：

* 用户画像画像聚合分析（新增标签宽表）
* 店铺级别的汇总指标
* 实时监控告警系统
* BI 工具与指标 API 接口对接