【最佳实践】实时计算 Flink 版在金融行业的实时数仓建设实践

行业背景

- 行业现状：

金融是现代经济的核心。我国金融业在市场化改革和对外开放中不断发展，金融总量大幅增长。金融稳定直接关系到国家经济发展的前途和命运，金融业是国民经济发展的晴雨表。对我国金融业发展现状进行客观分析，对金融业发展趋势进行探索，有助于消除金融隐患,使金融业朝着健康、有序方向发展。

- 大数据在其行业中的作用：

1. 金融服务和产品创新：借助社交网络等平台产生的海量用户和数据记录着用户群体的兴趣和偏好情绪等信息, 对客户行为模式进行分析，可以带来更贴近客户需求的产品创新。

2. 增强用户体验：通过大数据分析对客户进行画像，结合客户画像特征，为用户提供个性化服务，增强用户体验。

业务场景

某保险公司开发了个金融类APP，该公司在APP中会投放保险广告、发布优惠活动，用户通过APP进行投保等操作。
业务的构建涉及到几个端：

1. APP：应用程序，用户访问入口，用户通过APP点击浏览保险广告、优惠活动等，并进行投保下单。
2. 后台系统：

a.运营人员：
（1）根据用户提交的订单统计指定时间段的总投保人数和总投保金额，辅助优化运营方案。
（2）对用户的日常行为做出分析,分析出每个用户比较关注的信息,作为推荐系统的数据来源。
b:业务经理：
对重点客户的投保金额变动进行监控，将投保金额变动较大的重点客户推送给业务经理，业务经理针对性开展客户挽留等操作。

技术架构

架构解析：

数据采集：该场景中，数仓的数据主要来源于APP等系统的埋点信息，被实时采集至DATAHUB作为Flink的输入数据。
实时数仓架构：该场景中，整个实时数仓的ETL和BI部分的构建，全部通过Flink完成，Flink实时读取DATAHUB的数据进行处理，并与维表进行关联查询等操作，最终实时统计的结果输入到下游数据库RDS中。

业务指标

- 运营数据分析
  - 每小时的投保人数
  - 每小时的总保费
  - 每小时总保单数
- 用户行为监控
  - 用户原投保金额
  - 用户现投保金额
- 用户行为分析
  - 用户最后访问的页面类型
  - 用户最后访问的页面地址

数据结构

场景一：运营数据分析

> 本场景用于计算每小时总投保人数和总保费。
> 用户投保会生成一份订单,订单内容包括用户id、用户姓名、订单号等。flink实时读取订单信息，用where过滤出大于当前小时时间段的数据（数据过滤），然后根据用户id做分组用last_value函数获取每个用户最终生成的订单信息（订单去重），最后按照小时维度聚合统计当前小时的总保费和总投保人数。

输入表

```
CREATE   TABLE  user_order
(
    id                          bigint    comment '用户id'
    ,order_no                    varchar    comment '订单号'
    ,order_type                  bigint    comment '订单类型'
    ,pay_time                    bigint  comment '支付时间'
    ,order_price                 double    comment '订单价格'
    ,customer_name               varchar    comment '用户姓名'
    ,customer_tel                varchar    comment '用户电话'
    ,certificate_no              varchar    comment '证件号码'
    ,gmt_created                 bigint  comment '创建时间'
    ,gmt_modified                bigint  comment '修改时间'
```

```
    ,account_payble             double      comment '应付金额'

) WITH (
       type='datahub',
     topic='user_order'
       ...
)
```

输出表

```
CREATE    TABLE hs_order (
    biz_date              varchar COMMENT 'yyyymmddhh'
    ,total_premium         DOUBLE COMMENT '总保费'
    ,policy_cnt            BIGINT COMMENT '保单数'
    ,policy_holder_cnt     BIGINT COMMENT '投保人数'
    ,PRIMARY KEY (biz_date)
) WITH
 (
   type='rds',
   tableName='adm_pfm_zy_order_gmv_msx_hs'
   ...
 )
 ;
```

业务代码

1：数据清洗

```
create view  last_order
as
select
     id                                  as id
    ,last_value(order_no)                as order_no
    ,last_value(customer_tel)            as customer_tel
    ,last_value(gmt_modified)            as gmt_modified
    ,last_value(account_payble)          as account_payble
    from user_order
    where gmt_modified  >= cast(UNIX_TIMESTAMP(FROM_UNIXTIME(UNIX_TIMESTAMP(), 'y
    group by id
;
```

2：数据汇总

```
insert into hs_order
select
biz_date
,cast (total_premium as double) as total_premium
,cast (policy_cnt as bigint) as policy_cnt
,cast (policy_holder_cnt as bigint) as policy_holder_cnt
```

```
from (
select
    from_unixtime(cast(gmt_modified/1000 as bigint),'yyyyMMddHH')      as biz_dat
    ,sum(coalesce(account_payble,0))  as total_premium
    ,count(distinct order_no)   as policy_cnt
    ,count(distinct customer_tel)  as policy_holder_cnt
from  last_order a
group by
from_unixtime(cast(gmt_modified/1000 as bigint),'yyyyMMddHH')
)a
;
```

场景二：用户行为监控

> 本场景对投保金额变动较大（总保额变动大于15%）的重点用户进行监控。
> Flink实时读取用户新建订单数据，新建订单包括用户的id和现投保金额，通过where过滤没有保存成功的订单。维表中存储了业务经理关注的重点用户数据（如原投保金额），通过新建订单中的用户id与维表进行关联查询，如果维表中存在此用户且总保额下降15%以上，则将该用户的详细信息推送给业务经理，并且在维表中更新该用户投保金额及投保信息。

输入表

```
create table update_info
(
 id             bigint      comment '用户id'
,channel        varchar     comment '渠道编号'
,open_id        varchar     comment '订单id'
,event          varchar     comment '事件类型'
,now_premium  varchar     comment '现投保金额'
,creator        varchar     comment '创建人'
,modifier       varchar     comment '最后修改人'
,gmt_modified   bigint      comment '修改时间'
,now_info       varchar     comment '现投保信息'
) with (
    type = 'datahub',
    topic = 'dh_prd_dm_account_wechat_trace'
    ...

);
```

维表

```
 create table raw_info
(
     id                 bigint  comment '用户id'
    ,raw_premium      varchar  comment '原投保金额'
    ,raw_info           varchar  comment '原投保信息'
    ,PRIMARY KEY(id)
    ,PERIOD FOR SYSTEM_TIME
```

```
) WITH (
    type='ots',
    tableName='adm_zy_acct_sub_wechat_list'
    ...
);
```

输出表

```
create table update_info
(
     id               bigint  comment '用户id'
    ,raw_info         varchar comment '原投保信息'
    ,now_info         varchar comment '现投保信息'
    ,raw_premium      varchar comment '原投保金额'
    ,now_premium      varchar comment '现投保金额'
    ,PRIMARY KEY(id)
) WITH (
    type='rds',
    tableName='wechat_activity_user'
    ...
);
```

业务代码：

```
create view info_join as
select
      t1.id               as  id
    ,t2.raw_info          as  raw_info
    ,t1.now_info          as  now_info
    ,t2.raw_premium     as raw_premium
    ,t1.now_premium     as now_premium
from update_info t1
inner join raw_info FOR SYSTEM_TIME AS OF PROCTIME() as t2
on t1.id = t2.id ;
```

```
insert into update_info
select
     id                       as id
    ,raw_info                 as raw_info
    ,now_info                 as now_info
    ,raw_premium              as raw_premium
    ,now_premium              as now_premium
from info_join where now_premium<raw_premium*0.85
;
```

```
insert into raw_info
select
     id                       as id
```

```
    ,now_premium                as raw_premium
    ,now_info                   as raw_info
from info_join
;
```

场景三：用户行为分析

> 本场景记录用户最后访问的页面名称和类型，作为用户画像的特征值。
>
> Flink读取用户浏览APP页面的日志信息，如用户id、页面名称和页面类型等信息，根据用户id和设备id进行分组，通过last_value函数获取用户最后一次访问页面的名称和类型，输出到RDS作为推荐系统的输入数据，在下次用户登录的时候为其推送相关广告信息，提升用户广告点击率和下单的成功率。

输入表

```
create table user_log
(
 log_time                bigint  comment '日志采集日期(Linux时间)'
,device_id               varchar  comment '设备id'
,account_id              varchar  comment '账户id'
,bury_point_type         varchar  comment '页面类型或埋点类型'
,page_name               varchar  comment '页面名称或埋点时一级目录'
) WITH (
    type = 'datahub',
    topic = 'edw_zy_evt_bury_point_log'
    ...
);
```

输出表

```
create table user_last_log
(
     account_id         varchar  comment 'account_id'
    ,device_id          varchar    comment  '设备id'
    ,bury_point_type    varchar  comment '页面类型'
    ,page_name          varchar  comment '页面名称'
    ,primary key(account_id)
) WITH (
    type='rds',
    tableName='adm_zy_moblie_charge_exchg_rs'
    ...

);
```

业务代码

```
insert into user_last_log
select
    account_id
    ,device_id
    ,last_value(bury_point_type)  as bury_point_type
    ,last_value(page_name)  as page_name
from user_log
where account_id is not null
group by account_id,device_id
```