项目分点：

1. 集群规模：

（12 台物理机：128G 内存，8T 机械硬盘，2T 固态硬盘，20 核 40 线程，戴尔 4 万多一台）

2. 框架结构，画出来

（日志部分：日志服务器，落盘日志，flume，kafka，hdfs，hive，mysql

业务数据部分：mysql-sqoop-hdfs-hive）

3.框架：

（一）Flume（留了问题：flume，take 出小文件怎么处理，可以根据时间 10min 一次，或者 128M 一次落盘。）

① 我们选了单层 1.7 版本 flume，12 台物理机上部署了四个 flume 节点，

② 对于日志文件，我们保存 30 天，flume 使用的是 tailDir Source，这是 1.6 版本没有的。具有断点续传功能和读取多目录文件的功能。Memory channel 他比较快。Kafkasink 将 event 采集至 kafka。

③ 由于 flume 的 put 和 take 事务机制，所以他的数据采集准确性比较好。

④ 另外我们还使用了他的拦截器，用来做日志类型区分，和数据轻度过滤，检查时间戳不对的，json 数据不完整的，就舍弃。

⑤ 为了 sink 到 kafka 的不同主题，使用选择器，将不同类型的 event 发送至不同主题。

⑥ 我们为了监控 flume 的性能还是用了监控器，可以查看 put 和 take 机制尝试次数和成功次数，

⑦ 如果 put 的尝试次数低于成功次数，说明 flume 集群性能有问题，那我们可以考虑优化 flume，可以修改配置文件 flume-env 文件把内存从默认 1G 调到 4G

（二）Kafka（留问题：kafka 挂了怎么办，数据重复的问题）

① 搭建 kafka 集群，前后以 flume 分别作为生产者和消费者，我们根据日常的数据量，以及峰值速度，部署了三台 kafka，

② Kafka 的数据保存七天，副本数是 2，大概给 kafka 集群给了 1T 的资源。

③ 日常每天的数据量大概在 60-70G

④ 一般设置 8-10 个分区，不同的主题可能会不一样。

⑤ 我负责 8 个 topic，（有：浏览主题，评论主题，收藏，启动，故障日志，消息通知，广告，内容推送等）

⑥ 针对生产者，我们的 ack 设置 1，leader 收到了，就回应生产者 offset，还可以设置 0，这个很容易丢数据，设置-1 的话，也可以，leader 和 follower 都收到数据，才返回消息。

⑦ 针对消费者，我们使用 range 的分区分配策略，还可以选择 roundRobin，只不过 roundroubin 对消费者要求线程数一样，并且一个消费者组只消费同一个主题。Range 没有这些限制，而且分配也均匀。

⑧ Kafka 还有个问题就是相关 ISR 的副本同步队列问题，因为 leader 负责消费者消费内容。所以 leader 挂了谁上。就有副本同步队列。会根据 leader 和 follower 之间的延迟条数和延迟时间来判断，后面延迟条数被抛弃了。

⑨ Kafka 还可以设置多目录，可以优化读写性能。

⑩ 内存我们一般调为 4-5G

（三）HDFS（问题：HDFS 读写流程，shuffle 机制，Hadoop 优化。Yarn 调度器，yarn 任务提交流程，集群的搭建过程）

① Hadoop 作为集群的文件存储系统，在部署的时候要注意配置 HA。

② 也要注意 namenode 和 datanode 的通信，有两个参数可以提升他们通信的顺畅度。

③ 注意把 HDFS 的 namenode 文件 edits 和 fsimage 配置在不同目录下。可以提升

namenode 性能

（四）Hive（hive 的架构，动态分区与静态分区，四个 by，窗口函数，时间系统函数，hive 的优化）

① Hive 作为数据仓库的重要工具，他底层是 MR 程序，因为 MR 程序，代码书写很复杂，所以为了方便只熟悉 sql 语句的程序员利用 MR ，就有了 Hive。

② Hive 的数据存储到 HDFS 上，元数据信息，存储在 mysql 上，记得给 mysql 进行备份，使用主从 master 和 slave 结构。以免元数据被破坏了

③ Hive 的分析框架还是 MR，hive 的底层将 HQL 转换成抽象语法树，然后换成查询块，转换成逻辑查询计划，优化或，编程物理 MR 计划，选择最优的给 Mr 去执行

（五）数仓

① 我们的业务主要分为几个部分，根据物流一部分是相关特快直送的，一部分是在国内保税仓发货的，然后营销层面会有一些限时特卖板块，还有针对 vip 的专享活动，outlet 的特价商品。

② 针对我们的数仓的输入源，我们前面讲过来的日志数据，其实还包括业务数据，业务数据我们从 sqoop 里面导进来。

③ 要注意的是，sqoop 是 MR 任务。耗时较长，有可能失败，所以要做好事务一致性处理，他有两个参数可以利用--staging-table，--clear-staging-table。会讲数据先导入导临时表，如果失败了，就把临时表给删了，重新执行任务。当我们使用 sqoop 把数据从 hive 往 mysql 里面导的时候，hive 底层的 null 是‘\N’，所以 sqoop 导的时候要额外加参数 --null-string

④ kafka 主题：

日志： 浏览主题，广告，商品相关的，

业务：购物车， 退货，物流信息(根据产品的来源，有两种，香港特快直送，闪电保税仓。一个从香港发货，一个从内地的保税仓发货)，订单，评论主题，评分。

⑤ 我们的数据仓库，搭建了四层，ods，dwd，dws，ads 层。

⑥ Ods 层基本上就是一些原始数据，dwd 层是根据 ods 层表进行解析，然后清楚脏数据，利用 spark 做 ETL 工具进行过滤脏数据，敏感信息处理，比如电话，身份证脱敏，掩码，加密，截取等。Dwd 层得到的是比较干净的，能用的数据。Dws 层是根据 dwd 层轻度聚合来的数据，主要是一些宽表，ads 就是聚合后的数据，数据量很小，一些重要的指标数据结果，可以导入到 mysql。

⑦ 我们数据仓库是基于维度建模，主要使用星型模型。

⑧ 把表分为四类，实体表，主要是一些对象表比如用户，商家，商品等。

⑨ 维度表，是指指一些业务状态，编号的解释，又叫码表，像地区表，订单状态，支付方式，审批状态。状态分类。

⑩ 事实表分为周期型事实表和事务型事实表。如果需要后面状态还会改变的就是周期型事实表，一旦确定了，就是事务性事实表。

11 对于不同的表我们使用不同的同步策略，

同步策略包括全量表，增量表，新增及变化，拉链表

日志表：（商品点击，商品详情，商品详情页表，广告表，错误日志表，消息通知表等）

(1) 商品点击：用户的基本信息字段，动作，商品 id，种类等。

(2) 商品详情页：入口，上一页面来源，商品 id，加载时间，种类。

(3) 广告表：入口，内容，行为，展示风格等。

(4) 错误日志：错误详情

(5) 消息通知表：通知类型，展示时间，通知内容等

这些记录性质的，都使用每日增量

业务表：（购物车，评分，评论，订单表，订单详情表，退货表，用户表，商家表，商品分类表（一级，二级，三级），支付流水，物流信息等）

(1) 购物车详情：用户 id ，商品 id，商品价格，商家 id ，商品型号，商品分类等
同步策略：这属于周期型事实表因为他可能会随时改变，所以得用每日新增及变化。

(2) 评分表：评分时间，评分用户，评分商品 ，分数等，
同步策略：这是事务性事实表，一般可以用每日增量就可以了。但是如果电商用户对某件商品用过之后，又想改评论，所以用每日新增及变化

(3) 评论表：评论时间，评论用户，评论商品，评论内容，
同步策略：这个跟评分差不多，用每日新增及变化

(4) 订单表：订单状态，订单编号，订单金额，支付方式，支付流水，创建时间等
同步策略：因为订单的状态会随时发生改变，比如下单，支付，商家发货，用户收到货，确认收货，等这一系列的状态会比较长，然后订单也比较多。所以，要做历史快照信息的话，最好使用拉链表。

(5) 订单详情表：订单编号，订单号，用户 id，商品名称，商品价格，商品数量，创建时间等。
同步策略：这属于记录信息的，直接用每日新增。

(6) 退货流程表：用户 id，订单编号，退货商品 id，退货状态，时间等
同步策略：这种需要分阶段操作的，需要用户申请退货，平台审核，通过后，用户寄货，上传订单号，跟踪物流信息，商家收货，平台退款。这一系列的流程，可能需要很长时间。为了查看每个历史时间点的切片信息，我们需要做拉链表。

(7) 用户表：用户 id，性别，等级，vip，注册时间等等。
同步策略：因为表不是很大，每次做全量表。

(8) 商家表：商家 id，商家地址，商家规模等级，商家注册时间，商家分类信息。
同步策略：每次做每日全量

(9) 商品分类表：一级，二级，三级，
同步策略：表很小，或者不怎么改变的直接默认值。使用每日全量

(10) 商品表：商品 id，商品各级分类 id，商品的型号，商品的价格等
同步策略：有些可能做每日新增。但是他的数据量基本固定，并且没那么大，但是可能随着商品的上架，下架，更新起来很麻烦，所以我们直接做每日全量，

(11) 支付流水：支付方式，支付订单，支付用户 id，支付商家，支付时间等。
同步策略：每日新增。

(12) 物流信息表：快递公司名称，快递单号，状态，订单编号，等
同步策略：他是属于事务性事实表，像流水信息，直接用每日增量

总结：

① 实体表，不大，就可以做每日全量

② 对于维度表，比如说商品分类，这种不是很大，也可以做每日全量
有一些不太会发生改变的维度，就可以固定保存一份值，比如说：地区，种族，等

③ 像事务型事实表，比如说交易流水，操作日志，出库信息，这种每日比较大，且需要历史数据的，就根据时间做每日新增，可以利用分区表，每日做分区存储。

④ 像这种周期型事实表的同步策略，比如订单表，有周期性变化，需要反应不同时间点的状态的，就需要做拉链表。记录每条信息的生命周期，一旦一条记录的生命周期结束，就开始下一条新的记录。并把当前的日期放生效开始日期。

离线指标：

1.日活/周活/月活统计：（每日的根据 key 聚合，求 key 的总数）

2.用户新增：每日新增（每日活跃设备 left join 每日新增表，如果 join 后，每日新增表的设备 id 为空，就是新增）

3.用户留存率：（一周留存）10 日新增设备明细 join 11 日活跃设备明细表，就是 10 日留存的。注意每日留存，一周留存

4.沉默用户占比：只在当天启动过，且启动时间在一周前

5.

6.用户在线时长统计

7.区域用户订单数（根据区域分区，然后求订单数）

8.区域订单总额（根据区域分区，求订单总额。）

9.区域用户订单访问转化率（以区域分组成单数/访问数）

10.区域客单价（订单总额度/下订单总人数）

11.总退货率（退货商品数/购买商品总数）

12.各区域退货率（根据区域分组）

13.GMV（成交总额）

14.物流平均时长（用户收货时间-物流发货时间）求平均

15.每周销量前十品类

16.每周各品类热门商品销量前三

17.各区域热门商品销量前五（有利于后期铺货）

18.各区域漏斗分析

19.商品评价人数占比（该商品的总评价人数/该商品的总购买人数）

20.各品牌商家总销售额。

21.各品类中销量前三的品牌

22.购物车各品类占比（说明大家想买的东西，便于后期铺货。）

23.每周广告点击率。（看到这个广告的人数/点击这个广告商品的人数）

24.vip 用户每日，周订单总额

25.每日限时特卖产品占比（限时特卖产品总额/每日交易总额）

26.香港特快直送渠道总交易额占比（香港特快直送渠道总额/每日商品交易总额）

27.香港特快直送渠道总交易单占比

28.国内保税仓渠道总交易额占比（国内保税仓总额/每日商品交易总额）

29.国内保税仓渠道总交易单占比

30.各区域页面平均加载时长（考察各地区网络问题。后台访问是否稳定）

31.页面单跳转化率统计

32.获取点击下单和支付排名前 10 的品类

33.各类产品季度复购率

使用 HIVE 的（）

(1) 日活/周活/月活统计：（每日的根据 key 聚合，求 key 的总数）

```
select
    mid_id,
    concat_ws('|', collect_set(user_id)) user_id,
concat_ws('|', collect_set(version_code)) version_code,
from dwd_start_log
```

```
where dt='2019-02-10'
group by mid_id;
```

```
date_add(next_day('2019-02-10','MO'),-7),当前这周的周一日期
    date_add(next_day('2019-02-10','MO'),-1),当前这周的周日
    concat(date_add(  next_day('2019-02-10','MO'),-7),  '_'  ,
date_add(next_day('2019-02-10','MO'),-1)
)
from dws_uv_detail_day
where      dt>=date_add(next_day('2019-02-10','MO'),-7)      and
dt<=date_add(next_day('2019-02-10','MO'),-1)
group by mid_id;
月活
where                date_format(dt,'yyyy-MM')                  =
date_format('2019-02-10','yyyy-MM')要求年月时间符合要求就可以了
```

(2) 用户新增：每日新增（每日活跃设备 left join 每日新增表，如果 join 后，每日新增表的设备 id 为空，就是新增）

```
from  dws_uv_detail_day  ud  left  join  dws_new_mid_day  nm  on
ud.mid_id=nm.mid_id
where ud.dt='2019-02-10' and nm.mid_id is null;
```

(3) 用户留存率：（一周留存）10 日新增设备明细 join 11 日活跃设备明细表，就是 10 日留存的。注意每日留存，一周留存

```
    `create_date`    string  comment '设备新增时间',
    `retention_day`  int comment '截止当前日期留存天数'
    nm.create_date,
    1 retention_day
from dws_uv_detail_day ud join dws_new_mid_day nm   on ud.mid_id
  =nm.mid_id
where                  ud.dt='2019-02-11'                  and
nm.create_date=date_add('2019-02-11',-1);
```

```
ADS: create external table ads_user_retention_day_count
(
    `create_date`       string  comment '设备新增日期',
    `retention_day`     int comment '截止当前日期留存天数',
    `retention_count`    bigint comment  '留存数量'
)  COMMENT '每日用户留存情况'

select
    create_date,
    retention_day,
    count(*) retention_count
from dws_user_retention_day
```

```
留存率: create external table ads_user_retention_day_rate
(
     `stat_date`         string comment '统计日期',
     `create_date`       string  comment '设备新增日期',
     `retention_day`     int comment '截止当前日期留存天数',
     `retention_count`    bigint comment  '留存数量',
```

```
    `new_mid_count`     string  comment '当日设备新增数量',
    `retention_ratio`   decimal(10,2) comment '留存率'
```

(4) 沉默用户数：不是新用户，只在注册当天启动过，且启动时间在一周前

求沉默用户数（根据 dws 判断：第一次启动时间 min（dt）<一周前，并且根据 mid count，结果值只有 1，说明，用户的所有记录，只启动过一次）

```
        dws_uv_detail_day
    where
        dt<='2019-02-03'
    group by
        mid_id
    having
        count(*)=1
    and
        min(dt)<date_add('2019-02-03',-7)t1;
```

(5) 流失用户：最近一月未登陆我们称之为流失用户（）

根据日活 dws 表，查看如果最大的启动时间超过 14 天以前，那就是流失用户

```
from
    dws_uv_detail_day
group by
    mid_id
having max(dt)<=date_sub('2019-02-03',7))t1;
```

(6) 本周回流用户：本周回流=本周活跃-本周新增-上周活跃

本周活跃 mid left join 本周新增， left join 上周活跃 然后 join 的新字段都是 null 的

(7) 最近连续三周活跃用户

最近 3 周连续活跃的用户,通常是周一对前 3 周的数据做统计,该数据一周计算一次

dws_uv_detail_wk 利用周日活：

```
from
(select
    mid_id
from dws_uv_detail_wk
where
wk_dt>=concat(date_sub(next_day('2019-02-03','MO'),7*3),'_',date_
sub(next_day('2019-02-03','MO'),7*3-6))
and
wk_dt<=concat(date_sub(next_day('2019-02-03','MO'),7),'_',date_su
b(next_day('2019-02-03','MO'),1))
group by mid_id
having count(*)=3)t1;
```

(8) 本周内连续登陆三天的用户

1.查询出最近 7 天的活跃用户,并对用户活跃日期进行排名

```
select
    mid_id,
    dt,
    rank() over(partition by mid_id order by dt) rank
from dws_uv_detail_day
where dt>=date_sub('2019-02-03',6) and dt<='2019-02-03';t1
```

2.计算用户活跃日期及排名之间的差值

```
select
```

```
    mid_id,
    date_sub(dt,rank)
from t1;t2
```

3.对同一个用户分组,将差值相同个数大于等于 3 的数据取出,即为连续 3 天及以上活跃的用户

```
select
    mid_id
from t2
group by mid_id,date_diff
having count(*)>=3;t3
```

4.统计最近 7 天连续 3 天活跃的用户数

```
select count(*) from t3;
```

(9) GMV 每日/每周/每月成交总额：根据用户行为宽表，sum 订单总额

(10) 业务指标：当日新增占日活的比率

```
select
    '2019-02-10',
    sum(uc.dc) sum_dc,
    sum(uc.nmc) sum_nmc,
    cast(sum(   uc.nmc)/sum(   uc.dc)*100   as   decimal(10,2))
new_m_ratio
from
(
    select
        day_count dc,
        0 nmc
    from ads_uv_count
    where dt='2019-02-10'

    union all
    select
        0 dc,
        new_mid_count nmc
    from ads_new_mid_count
    where create_date='2019-02-10'
)uc;
```

(11) 用户行为之漏斗分析：

```
create external  table ads_user_action_convert_day(
   `dt` string COMMENT '统计日期',
   `total_visitor_m_count`  bigint COMMENT '总访问人数',
   `order_u_count` bigint    COMMENT '下单人数',
   `visitor2order_convert_ratio`  decimal(10,2) COMMENT '访问到下单转化率',
   `payment_u_count` bigint    COMMENT '支付人数',
   `order2payment_convert_ratio` decimal(10,2) COMMENT '下单到支付的转化率'
 ) COMMENT '用户行为漏斗分析'
select
    '2019-02-10',
    uv.day_count,日活
    ua.order_count,日订单
    cast(ua.order_count/uv.day_count*100   as     decimal(10,2))
visitor2order_convert_ratio,
    ua.payment_count,
    cast(ua.payment_count/ua.order_count*100 as  decimal(10,2))
```

```
order2payment_convert_ratio
from
(
    select
        sum(if(order_count>0,1,0)) order_count,下单人数
        sum(if(payment_count>0,1,0)) payment_count 支付人数
    from dws_user_action
    where dt='2019-02-10'
)ua, ads_uv_count  uv
where uv.dt='2019-02-10'
```

(12) 品牌复购率（一个季度之内，同一个品牌买两次的人数，买三次的人数。/此品牌购买的总人数）

(13) 用户在线时长统计（进入后台时间戳-启动时间戳，根据用户 sum，然后求平均值）
(14) 一周销量前十品类：用户行为宽表，过滤一周的数据，根据 group by 品类 Count 数
(15) 各区域漏斗分析，与漏斗分析类似，根据区域分组

使用 Spark 的指标：
1. 每周各品类热门商品销量前三（取每周各热门品类，然后取用户行为宽表的几个字段，热门品类，用户 id，商品 id。然后用热门品类过滤。得到属于热门品类的数据，再根据热门品类，商品 id，去聚合。去前三。）
2. 各区域热门商品销量前五：取用户行为宽表，然后得到里面的数据，可以转化成样例类的 rdd。然后根据区域分组，然后求商品销量，前五的。
3. 商品评价人数占比（该商品的总评价人数/该商品的总购买人数）：根据商品 key 分类，拿到评价人数，再拿到总购买人数
4. 各品牌商家总销售额。 根据商家为 key，拿去用户行为宽表的数据，求对应的销售额
5. 各品类中销量前三的品牌
6. 购物车各品类占比：以品牌为 key，数量为 value。从购物车宽表中获取数据。然后根据品牌分类，求总数。（说明大家想买的东西，便于后期铺货。）
7. 每周广告点击率。（看到这个广告的人数/点击这个广告商品的人数）
8. vip 用户每日，周订单总额
9. 每日限时特卖产品占比（限时特卖产品渠道总额/每日交易总额）
10. 香港特快直送渠道总交易额占比（香港特快直送渠道总额/每日商品交易总额）
11. 香港特快直送渠道总交易单占比
12. 国内保税仓渠道总交易额占比（国内保税仓总额/每日商品交易总额）
13. 国内保税仓渠道总交易单占比
14. 各区域页面平均加载时长（页面加载成功时间戳-请求时间戳）（考察各地区网络问题。后台访问是否稳定）
15. 获取点击下单和支付排名前 10 的品类

实时指标（spark streaming 做）：
1. 每日日活实时统计
2. 每日订单量实时统计
3. 一小时内日活实时统计

4. 一小时内订单数实时统计
5. 一小时内交易额实时统计
6. 一小时内广告点击实时统计
7. 一小时内区域订单数统计
8. 一小时内区域订单额统计
9. 一小时内各品类销售 top3 商品统计
10. 用户购买明细灵活分析（根据区域，性别，品类等）

1. 准备三个具体的指标。比较难，又有对运营，营销又非常有价值的。帮助他们做了什么事，讲讲怎么做的

寻找潜在 VIP：

1. 上一周连续 3 天登录，且上周内下过一单的

先过滤取出上周内下过一单，又是非 vip 的人。（从订单明细表）

再根据他们每日的最早启动时间，用 rank 窗口函数进行排序。那么排序的这个字段就应该是以 1 为公差的等差数列（left join 用户活跃表日）

然后再用 date-sub 去将启动日期与 rank 计算，得到了日期差值，根据这个日期差值进行分组，计算这个差有几个。

>3 就是我们所需要的用户。

找出来之后，给她短信，后台消息推送 vip 活动免费一个月体验。减税，免邮，享受会员价等活动。

2. 过去一个月内下单商品大于 8 件，且下单次数大于 2

使用用户订单详情表：取出过去一个月的非 vip 用户购买详情。

计算每个用户的下单商品数，下单次数>2 （group by userID，sum（购买件数），count（distinct 订单号）》2）

推送消息，给免费 vip 活动体验

这部分的用户在接下来的三个月时间里，真正转换成 vip 的有 35%的人，所以这个指标还挺有意义的

商品季度/半年复购率（购买过这个商品两次以上的用户数/这个季度购买这种商品的总人数）：

3. 用户购买明细表。

把上个季度的用户购买详情表过滤出来。group by 用户 id 商品 id 分组，求出用户对于某个商品下单的总次数。

然后用 sum if（判断订单单数>2），订单单数>1 的人数，求比率,

然后对比率根据品类排名，求每个品类中 比率排名前十的。用 row_number<11.分区取品类，排序取复购率。

这些商品，是我们的重要维系的商品，要及时补货。然后复购率高说明，受用户喜欢，可以推荐，给用户发送小样，尝试，增大转化率。

4.品牌复购率：

差不多，把具体商品，改成品牌 id。各类商品下的品牌复购率（每月来算）

5.每周各品类热门商品销量前三（取每周各热门品类，然后取用户行为宽表的几个字段，热门品类，用户 id，商品 id。然后用热门品类过滤。得到属于热门品类的数据，再根据热门品类，商品 id，去聚合。去前三。）
6.各区域热门商品销量前五：取用户行为宽表，然后得到里面的数据，可以转化成样例类的 rdd。然后根据区域分组，然后求商品销量，前五的。
7.各品类中销量前三的品牌
8.购物车各品类占比：以品牌为 key，数量为 value。从购物车宽表中获取数据。然后根据品牌分类，求总数。（说明大家想买的东西，便于后期铺货。

数据健康问题：
物流信息：有的客户物流信息上显示收到货了，但是快递可能没有送到他手里，然后过程中有丢失的情况。那么我们的物流计算时长，如果单纯按照物流信息来就会出现偏差，所以我们物流到货时间都是以用户，确认收货为准。也不会差很大。
用户的隐私信息，电话号码：我们使用自己的一套脱敏技术，将每个电话号码的 4-11 位，加 1，然后 4-7 位与 8-11 位顺序调换。后期我们需要用到他们的隐私信息，电话进行，营销，发送消息是，就把他转换过来。

数据倾斜问题：
1. 用时间维度表去 join 过去一整年的用户购买明细表，查看，用户集中购买的月份和季节。分析用户的行为。之前不是默认的。（默认开启 mapJoin 嘛）
2. 小表 join 大表的问题。后面这个优化了，但是小表不能超过 512M.我们数据量没那么大，应该是可以的。

比如说算品类销售排名的时候，group by 品类，求销售总量是，某一品类像面膜，可能销售量特别大，占 60%多，那么有一个任务就会执行特别久。半天出不来。设置推测执行也差不多，就应该是数据倾斜导致的问题
Map 端部分聚合

这里需要修改的参数为：

hive.map.aggr=true（用于设定是否在 map 端进行聚合，默认值为真）
hive.groupby.mapaggr.checkinterval=100000（用于设定 map 端进行聚合操作的条目数）

- 有数据倾斜时进行负载均衡

此处需要设定 hive.groupby.skewindata，当选项设定为 true 是，生成的查询计划有两个 MapReduce 任务。在第一个 MapReduce 中，map 的输出结果集合会随机分布到 reduce 中， 每个 reduce 做部分聚合操作，并输出结果。这样处理的结果是，相同的 Group By Key 有可 能分发到不同的 reduce 中，从而达到负载均衡的目的；第二个 MapReduce 任务再根据预处 理的数据结果按照 Group By Key 分布到 reduce 中（这个过程可以保证相同的 Group By Key 分布到同一个 reduce 中），最后完成最终的聚合操作。