# [电商大数据应用之用户画像](https://segmentfault.com/a/1190000007694510)

# 课程目标

## 1.1 了解用户画像是什么

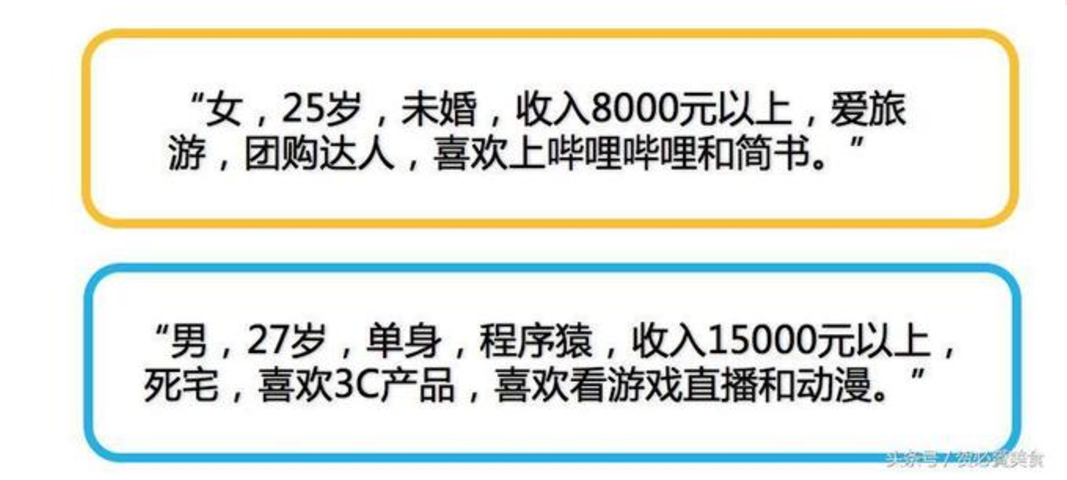
## 1.2 用户画像的作用是什么

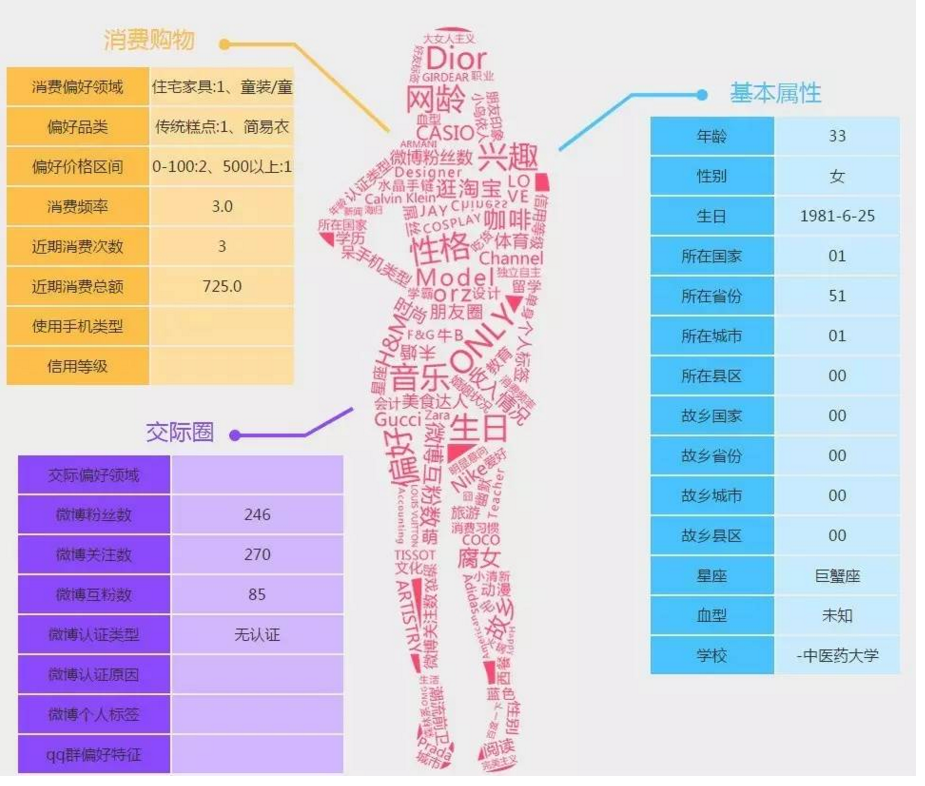
## 1.3 学会使用SparkSQL构建用户画像

# 前言

大数据时代已经到来，企业迫切希望从已经积累的数据中分析出有价值的东西，而用户行为的分析尤为重要。利用大数据来分析用户的行为与消费习惯，可以预测商品的发展的趋势，提高产品质量，同时提高用户满意度。本课程是基于大型电商公司的真实用户画像中提练出的精华内容，旨在培养学员了解用户画像的内容，掌握构建用户画像的方法。

# 初识用户画像





右边是一个人的基本属性，通过一个人的基本属性我们可以了解到这个人的基本信息，左边上图是通过消费购物信息来描述一个人特征，左边下图是通过交际圈信息来描述一个人特征，通过不同的维度，去描述一个人，认识一个人，了解一个人。这就是我们今天所要讲到的用户画像。

用户画像：也叫用户信息标签化、客户标签；根据用户社会属性、生活习惯和消费行为等信息而抽象出的一个标签化的用户模型。从电商的角度看，根据你在电商网站上所填的信息和你的行为，可以用一些标签把你描绘出来，描述你的标签就是用户画像。构建用户画像的核心工作即是给用户贴“标签”，而标签是通过对用户信息分析而来的高度精炼的特征标识。

# 构建电商用户画像的重大意义

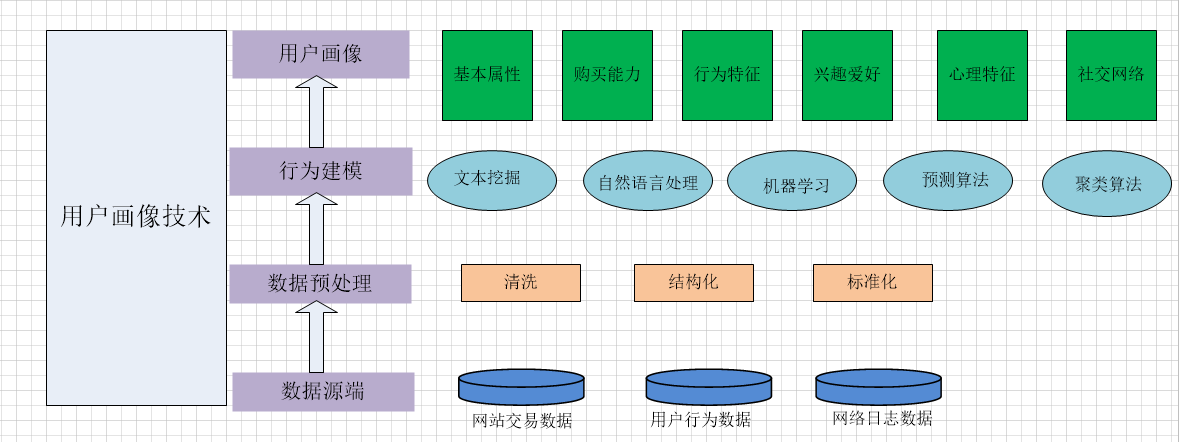
罗振宇在《时间的朋友》跨年演讲举了这样一个例子：当一个坏商家掌握了你的购买数据，他就可以根据你平时购买商品的偏好来决定是给你发正品还是假货以此来提高利润，且不说是否存在这种情况，但这也说明了利用用户画像可以做到“精准营销”，当然这是极其错误的用法。

其作用大体不离以下几个方面：

* 1、精准营销，分析产品潜在用户，针对特定群体利用短信邮件等方式进行营销；
* 2、用户统计，比如中国大学购买书籍人数 TOP10；
* 3、数据挖掘，构建智能推荐系统，利用关联规则计算，喜欢红酒的人通常喜欢什么运动品牌，利用聚类算法分析，喜欢红酒的人年龄段分布情况；
* 4、进行效果评估，完善产品运营，提升服务质量，其实这也就相当于市场调研、用户调研，迅速下定位服务群体，提供高水平的服务；
* 5、对服务或产品进行私人定制，即个性化的服务某类群体甚至每一位用户（个人认为这是目前的发展趋势，未来的消费主流）。比如，某公司想推出一款面向5－10岁儿童的玩具，通过用户画像进行分析，发现形象＝“喜羊羊”、价格区间＝“中等”的偏好比重最大，那么就给新产品提供了非常客观有效的决策依据。
* 6、业务经营分析以及竞争分析，影响企业发展战略

# 如何构建电商用户画像

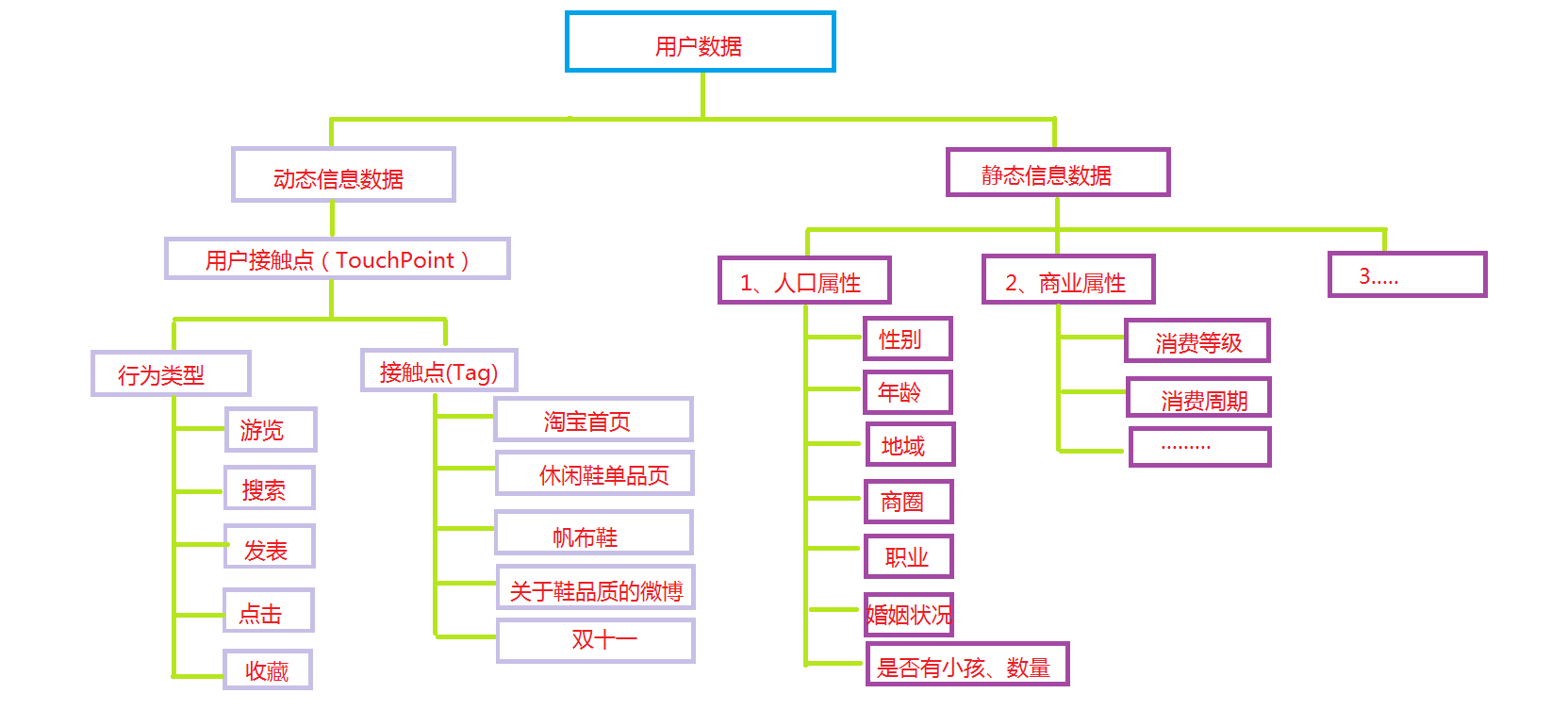
## 5.1 构建电商用户画像技术和流程



构建一个用户画像，包括数据源端数据收集、数据预处理、行为建模、构建用户画像

有些标签是可以直接获取到的，有些标签需要通过数据挖掘分析到！

## 5.2 源数据分析



用户数据分为2类：动态信息数据、静态信息数据

静态信息数据来源：

* 用户填写的个人资料，或者由此通过一定的算法，计算出来的数据
* 如果有不确定的，可以建立模型来判断，比如用户的性别注册没有填写，可以建立模型，根据用户的行为来判断用户性别是什么，或者它的概率

动态信息数据来源：

* 用户行为产生的数据：注册、游览、点击、购买、签收、评价、收藏等等。
* 用户比较重要的行为数据：游览商品，收藏商品、加入购物车、关注商品

根据这些行为特性可以计算出：用户注册时间、首单时间、潮妈族、纠结商品、最大消费、订单数量、退货数量、败家指数、品牌偏好等等。

## 5.3 目标分析

用户画像的目标是通过分析用户行为，最终为每个用户打上标签，以及该标签的权重。

如，红酒 0.8、李宁 0.6。

标签：表现了内容，用户对该内容有兴趣、偏好、需求等等。

权重：表现了指数，用户的兴趣、偏好指数，也可能表现用户的需求度，可以简单的理解为可信度，概率。

## 5.4 用户画像建模

### 5.4.1 用户基本属性表

根据用户所填写的属性标签和推算出来的标签。用于了解用户的人口属性的基本情况和按不同属性维度统计。

作用：按人口属性营销、比如营销80后，对金牛座的优惠，生日营销。

主要数据来源：用户表、用户调查表、孕妇模型表、马甲模型表。

用户表：记录用户最基本的属性特性。

用户调查表：补充用户的其他基本信息。

用户所填写的基本信息：用户ID、用户名、密码、性别、手机号、邮箱、年龄、户籍省份、身份证编号、注册时间、收货地址等

用户所填信息计算得到的指标：

生日、星座、城市等级、手机前几位、手机运营商、邮件运营商

用户调查表得到：学历、收入、职业、婚姻、是否有小孩、是否有车有房、使用手机品牌。

根据算法得到:

身高、体重、性别模型、孩子性别概率、潜在汽车用户概率、是否孕妇、孩子年龄概率、手机品牌、更换手机频率、是否有小孩，是否有车，使用手机档次，疑似马甲标准、疑似马甲账号数、用户忠诚度、用户购物类型。

模型算法---性别模型

* 用户自己也填写了性别，但仍然要用算法算一次性别

|  |  |  |
| --- | --- | --- |
| 用户性别 | 1男 0女 -1未识别 | 1. 商品性别得分 2. 用户购买上述商品计算用户性别等得分 3. 最优化算法训练阀值，根据阀值判断 |
| 孩子性别 | 0 仅有男孩  1仅有女孩  2男女都有  3无法识别 | 1. 选择男孩女孩商品 2. 确定用户购买商品的男女性别比例 3. 训练阀值，判断孩子性别,同用户性别类似 |

* 性别验证方法

随机抽样几千条数据让客户打电话确认。

与用户自己填的性别做对比，确认百分比。

模型算法---用户汽车模型

|  |  |  |
| --- | --- | --- |
| 用户是否有车 | 1有 0 没有  -1 未识别 | 根据用户购买车相关产品  判断用户是否有车 |
| 潜在汽车用户 | 1有 0 没有  -1 未识别 | 用户游览或者搜索汽车  用户数据判断 |

模型算法---用户忠诚度模型

* 忠诚度越高的用户越多，对网站的发展越有利

|  |  |  |
| --- | --- | --- |
| 用户忠诚度 | 1忠诚型用户  2偶尔型用户  3投资型用户  4游览型用户  -1未识别 | 总体规则是判断+聚类算法  1、游览用户型：只游览不购买的  2、购买天数大于一定天数的为忠诚用户  3、购买天数小于一定天数，大部分是有优惠才购买的  4、其他类型根据购买天数，购买最后一次距今时间，购买金额进行聚类 |

模型算法---用户身高尺码模型

|  |  |  |
| --- | --- | --- |
| 男性用户身高尺码 | xxx-xxx身高段，-1未识别 | 用户购买服装鞋帽等用户判断 |
| 男性身材 | 1偏瘦、2标准、3偏胖4肥胖、-1未识别 | 用户购买服装鞋帽等用户判断 |
| 女性用户身高尺码 | xxx-xxx身高段，-1未识别 | 用户购买服装鞋帽等用户判断 |
| 女性身材 | 1偏瘦、2标准、3偏胖4肥胖、-1未识别 | 用户购买服装鞋帽等用户判断 |

模型算法---用户马甲标志模型

* 马甲是指一个用户注册多个账号
* 多次访问地址相同的用户账号是同一个人所有
* 同一台手机登陆多次的用户是同一个人所有
* 收货手机号相同的账号同一个人所有

模型算法---手机相关标签模型

* 对于手机营销参考意义比较大
* 使用手机品牌: 最常用手机直接得到
* 使用手机品牌档次：根据档次维表
* 使用多少种不同的手机：手机登陆情况
* 更换手机频率（月份):按时间段看手机登陆情况

### 5.4.2 客户消费订单表

根据客户消费的情况提取的客户标签，用于了解用户的消费总体情况，

最终的目的根据用户消费习惯与消费能力做营销。

主要数据来源：订单表、退货表、用户表、购物车表

订单表可以得到相关标签：

|  |
| --- |
| 第一次消费时间、  最近一次消费时间、  首单距今时间、  尾单距今时间------分析用户什么时候来购买商品以及多久没有购买了。  最小消费金额、  最大消费金额、  累计消费次数(不含退拒)、  累计消费金额（不含退拒）、  累计使用代金券金额、  累计使用代金券次数。-----分析用户总体消费情况。  客单价（含退拒）、  近60天客单价（含退拒）-----分析用户消费水平。  常用收货地址、  常用支付方式----分析用户常用的消费属性，方便做定向营销。 |

退货表可以得到相关标签：

|  |
| --- |
| 近30天购买次数(不含退拒)、  近30天购买金额（不含退拒）  近30天购买次数(含退拒)、  近30天购买金额（含退拒）----分析用户最近的消费能力。  退货商品数量、  退货商品金额、  拒收商品数量、  拒收商品金额、  最近一次退货时间-----分析用户拒收和退货习惯。 |

购物车表可以得到相关标签：

|  |
| --- |
| 最近30天购物车次数、  最近30天购物车商品件数、  最近30天购物车提交商品件数、  最近30天购物车放弃件数、  最近30天购物车成功率------分析用户购物车使用习惯 |

订单表和用户表可以得到相关标签：

|  |
| --- |
| 学校下单总数、  单位下单总数、  家里下单总数、  上午下单总数、  下午下单总数、  晚上下单总数----分析用户购物时间与地点习惯。 |

### 5.4.3 客户购买类目表

根据客户购买类目的情况提取客户标签，用于了解类目的购买人群情况和针对某一类目的营销等。

主要数据来源：订单表、购物车表、类目维表

类目维表可以得到相关标签：

|  |
| --- |
| 一级分类ID、  一级分类名称、  二级分类ID、  二级分类名称、  三级分类ID、  三级分类名称-----分析用户都购买了哪些类目。 |

电商的三级类目：

京东商城：



淘宝：



订单表和类目维表可以得到相关标签：

|  |
| --- |
| 近30天购买类目次数、  近30天购买类目金额、  近90天购买类目次数、  近90天购买类目金额、  近180天购买类目次数、  近180天购买类目金额、  累计购买类目次数、  累计购买类目金额----分析用户最近都购买了哪些类目。  最近一次购买类目时间、  最后一次购买类目距今天数----分析用户多久没有购买这个类目。 |

购物车表和类目维表可以得到相关标签：

|  |
| --- |
| 近30天购物车类目次数、  近30天购物车类目金额、  近90天购物车类目次数、  近90天购物车类目金额----分析用户最近都挑中哪些类目。 |

### 5.4.4 用户访问信息表

根据客户访问的情况提取相关客户标签。

用于了解用户的访问总体情况，方便根据客户游览习惯做营销

主要数据来源：点击流日志行为表（PC/APP端）

点击流日志行为表可以得到相关标签:

|  |
| --- |
| 最近一次APP/PC端访问日期、  最近一次APP/PC端访问使用操作系统、  最近一次APP/PC端访问使用游览器、  最近一次访问IP地址、  最近一次访问城市、  最近一次访问的省份-----分析用户最近一次访问情况。  第一次APP/PC端访问日期、  第一次APP/PC端访问使用操作系统、  第一次APP/PC端访问使用游览器、  第一次访问IP地址、  第一次访问城市、  第一次访问的省份-----分析用户第一次访问情况。  近7天APP/PC端访问次数、  近30天APP/PC访问次数、  近60天APP/PC端访问次数、  近90天APP/PC端访问次数、  近180天APP/PC端访问次数、  近365天APP/PC端访问次数----分析用户APP/PC端访问次数。  近30天PC/APP端访问天数、  近30天PC/APP端访问并购买次数、  近30天PC/APP端访问PV、  近30天PC/APP端访问平均PV、  近30天PC/APP端最常用的游览器、  近30天PC/APP端不同IP数、  近30天PC/APP端最常用IP-----分析用户访问详情。  近30天0-5点访问的次数、  近30天6-7点访问的次数、  近30天8-9点访问的次数、  近30天10-12点访问的次数、  近30天13-14点访问的次数、  近30天15-17点访问的次数、  近30天18-19点访问的次数、  近30天20-21点访问的次数、  近30天22-23点访问的次数----分析用户喜欢在哪个时间上网访问。 |

# 电商用户画像环境搭建

众所周知，Hive的执行任务是将hql语句转化为MapReduce来计算的，Hive的整体解决方案很不错，但是从查询提交到结果返回需要相当长的时间，查询耗时太长。这个主要原因就是由于Hive原生是基于MapReduce的，那么如果我们不生成MapReduce Job，而是生成Spark Job，就可以充分利用Spark的快速执行能力来缩短HiveHQL的响应时间。

本项目采用SparkSql与hive进行整合，通过SparkSql读取hive中表的元数据，把HiveHQL底层采用MapReduce来处理任务,导致性能慢的特点，改为更加强大的Spark引擎来进行相应的分析处理，快速的为用户打上标签构建用户画像。

## 6.1 环境准备

* 1、搭建hadoop集群（见文档）
* 2、安装hive构建数据仓库（见文档）
* 3、安装spark集群（见文档）
* 4、sparksql 整合hive

## 6.2 sparksql整合hive

[Spark](https://www.iteblog.com/archives/tag/spark/) SQL主要目的是使得用户可以在[Spark](https://www.iteblog.com/archives/tag/spark/)上使用SQL，其数据源既可以是RDD，也可以是外部的数据源（比如文本、[Hive](https://www.iteblog.com/archives/tag/hive/)、Json等）。[Spark](https://www.iteblog.com/archives/tag/spark/) SQL的其中一个分支就是Spark on [Hive](https://www.iteblog.com/archives/tag/hive/)，也就是使用Hive中HQL的解析、逻辑执行计划翻译、执行计划优化等逻辑，可以近似认为仅将物理执行计划从MR作业替换成了Spark作业。SparkSql整合hive就是获取hive表中的元数据信息，然后通过SparkSql来操作数据。

### 第一步：将hive-site.xml拷贝到spark安装家路径的conf目录下

node03执行以下命令来拷贝hive-site.xml到所有的spark安装服务器上面去

cd /export/servers/hive-1.1.0-cdh5.14.0/conf

cp hive-site.xml /export/servers/spark-2.2.0-bin-2.6.0-cdh5.14.0/conf/

scp hive-site.xml node02:/export/servers/spark-2.2.0-bin-2.6.0-cdh5.14.0/conf/

scp hive-site.xml node01:/export/servers/spark-2.2.0-bin-2.6.0-cdh5.14.0/conf/

### 第二步：将mysql的连接驱动包拷贝到spark的jars目录下

node03执行以下命令将连接驱动包拷贝到spark的jars目录下，三台机器都要进行拷贝

cd /export/servers/hive-1.1.0-cdh5.14.0/lib

cp mysql-connector-java-5.1.38.jar /export/servers/spark-2.2.0-bin-2.6.0-cdh5.14.0/jars/

scp mysql-connector-java-5.1.38.jar node02:/export/servers/spark-2.2.0-bin-2.6.0-cdh5.14.0/jars/

scp mysql-connector-java-5.1.38.jar node01:/export/servers/spark-2.2.0-bin-2.6.0-cdh5.14.0/jars/

## 6.3 测试sparksql整合hive是否成功

先启动hadoop集群，在启动spark集群，确保启动成功之后node01执行命令：

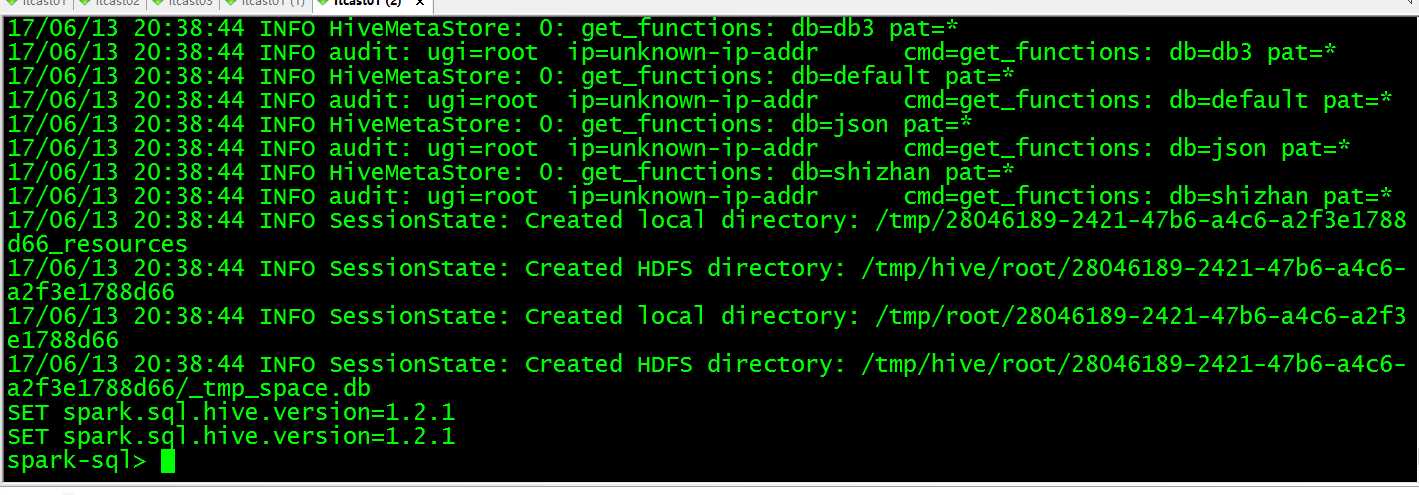
cd /export/servers/spark-2.2.0-bin-2.6.0-cdh5.14.0

bin/spark-sql --master spark://node01:7077 --executor-memory 1G --total-executor-cores 2

指明master地址、每一个executor的内存大小、一共所需要的核数、

mysql数据库连接驱动。

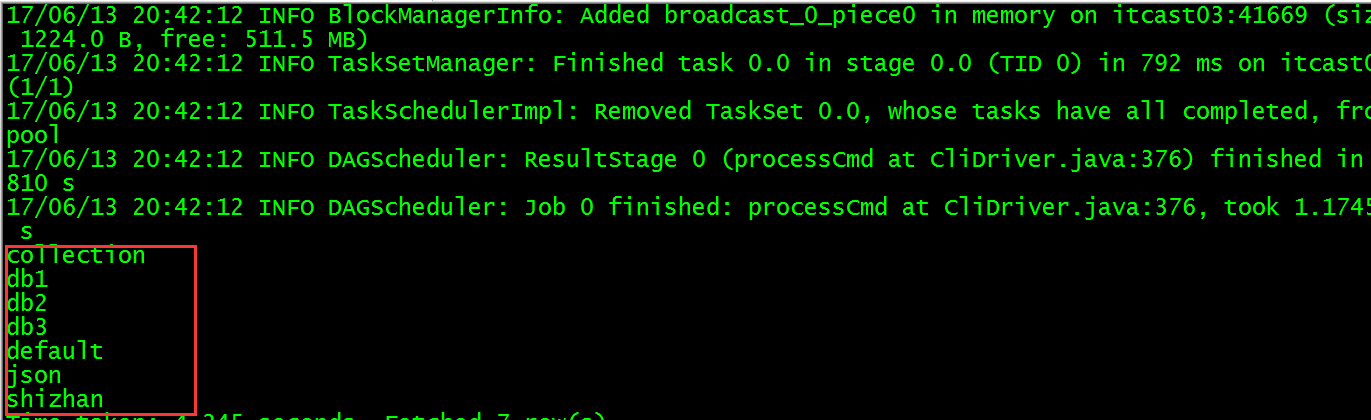
执行成功后的界面：进入到spark-sql 客户端命令行界面



接下来就可以通过sql语句来操作数据库表：

查看当前有哪些数据库

show databases;



看到以上结果，说明sparksql整合hive成功！

日志太多，我们可以修改spark的日志输出级别(conf/log4j.properties)

前方高能：

在spark2.0版本后由于出现了sparkSession，在初始化sqlContext的时候，会设置默认的spark.sql.warehouse.dir=spark-warehouse,

此时将hive与sparksql整合完成之后，在通过spark-sql脚本启动的时候，还是会在哪里启动spark-sql脚本，就会在当前目录下创建一个spark.sql.warehouse.dir为spark-warehouse的目录，存放由spark-sql创建数据库和创建表的数据信息，与之前hive的数据息不是放在同一个路径下（可以互相访问）。但是此时spark-sql中表的数据在本地，不利于操作，也不安全。

所有在启动的时候需要加上这样一个参数：

--conf spark.sql.warehouse.dir=hdfs://node01:8020/user/hive/warehouse

保证spark-sql启动时不在产生新的存放数据的目录，sparksql与hive最终使用的是hive同一存放数据的目录。

如果使用的是spark2.0之前的版本，由于没有sparkSession，不会有spark.sql.warehouse.dir配置项，不会出现上述问题。

最后的执行脚本；

node01执行以下命令重新进去spark-sql

cd /export/servers/spark-2.2.0-bin-2.6.0-cdh5.14.0

bin/spark-sql \

--master spark://node01:7077 \

--executor-memory 1G --total-executor-cores 2 \

--conf spark.sql.warehouse.dir=hdfs://node01:8020/user/hive/warehouse

# 电商用户画像数据仓库建立

## 7.1 数据仓库准备工作

**为什么要对数据仓库分层？**

* 用空间换时间，通过大量的预处理来提升应用系统的用户体验（效率），因此数据仓库会存在大量冗余的数据；
* 如果不分层的话，如果源业务系统的业务规则发生变化将会影响整个数据清洗过程，工作量巨大
* 通过数据分层管理可以简化数据清洗的过程，因为把原来一步的工作分到了多个步骤去完成，相当于把一个复杂的工作拆成了多个简单的工作，把一个大的黑盒变成了一个白盒，每一层的处理逻辑都相对简单和容易理解，这样我们比较容易保证每一个步骤的正确性，当数据发生错误的时候，往往我们只需要局部调整某个步骤即可。

数据仓库标准上可以分为四层：ODS（临时存储层）、PDW（数据仓库层）、MID（数据集市层）、APP（应用层）

**ODS层：**

为临时存储层，是接口数据的临时存储区域，为后一步的数据处理做准备。一般来说ODS层的数据和源系统的数据是同构的，主要目的是简化后续数据加工处理的工作。从数据粒度上来说ODS层的数据粒度是最细的。ODS层的表通常包括两类，一个用于存储当前需要加载的数据，一个用于存储处理完后的历史数据。历史数据一般保存3-6个月后需要清除，以节省空间。但不同的项目要区别对待，如果源系统的数据量不大，可以保留更长的时间，甚至全量保存；

**PDW层：**

为数据仓库层，PDW层的数据应该是一致的、准确的、干净的数据，即对源系统数据进行了清洗（去除了杂质）后的数据。这一层的数据一般是遵循数据库第三范式的，其数据粒度通常和ODS的粒度相同。在PDW层会保存BI系统中所有的历史数据，例如保存10年的数据

**MID层：**

为数据集市层，这层数据是面向主题来组织数据的，通常是星形或雪花结构的数据。从数据粒度来说，这层的数据是轻度汇总级的数据，已经不存在明细数据了。从数据的时间跨度来说，通常是PDW层的一部分，主要的目的是为了满足用户分析的需求，而从分析的角度来说，用户通常只需要分析近几年（如近三年的数据）的即可。从数据的广度来说，仍然覆盖了所有业务数据。

**APP层：**

为应用层，这层数据是完全为了满足具体的分析需求而构建的数据，也是星形或雪花结构的数据。从数据粒度来说是高度汇总的数据。从数据的广度来说，则并不一定会覆盖所有业务数据，而是MID层数据的一个真子集，从某种意义上来说是MID层数据的一个重复。从极端情况来说，可以为每一张报表在APP层构建一个模型来支持，达到以空间换时间的目的数据仓库的标准分层只是一个建议性质的标准，实际实施时需要根据实际情况确定数据仓库的分层，不同类型的数据也可能采取不同的分层方法。

这里我们采用的是京东的数据仓库分层模式，是根据标准的模型演化而来。

数据仓库分层：

BDM：缓冲数据，源数据的直接映像

FDM：基础数据层，数据拉链处理、分区处理

GDM：通用聚合

ADM：高度聚合

先把数据从源数据库中抽取加载到BDM层中，

然后FDM层根据BDM层的数据按天分区

## 7.2 数据仓库基本表介绍

|  |  |  |
| --- | --- | --- |
| BDM层数据表  (贴源缓存层) | 订单表 | itcast\_bdm\_order |
| 订单明细表 | itcast\_bdm\_order\_desc |
| 订单商品表 | itcast\_bdm\_order\_goods |
| 用户表 | itcast\_bdm\_user |
| 购物车表 | itcast\_bdm\_order\_cart |
| 用户上网记录表 | itcast\_bdm\_user\_pc\_click\_log  itcast\_bdm\_user\_app\_click\_log |

|  |  |  |
| --- | --- | --- |
| FDM层数据表  (拉链表、分区表) | 用户宽表 | itcast\_fdm\_user\_wide |
| 购物车表 | itcast\_fdm\_order\_cart |
| 订单表 | itcast\_fdm\_order |
| 订单表明细表 | itcast\_fdm\_order\_desc |
| 用户app端view表 | itcast\_fdm\_user\_app\_pageview |
| 用户pc端view表 | itcast\_fdm\_user\_pc\_pageview |
| GDM层数据表  (通用数据模型层) | 客户基本属性表 | itcast\_gdm\_user\_basic |
| 客户消费订单表 | itcast\_gdm\_user\_consume\_order |
| 订单模型表 | itcast\_gdm\_order |
| 客户购买类目表 | itcast\_gdm\_user\_buy\_category |
| 客户访问信息表 | itcast\_gdm\_user\_visit |

# 电商用户画像开发

## 8.1用户画像--数据开发的步骤

* 数据开发前置依赖

-需求确定

-建模确定表结构

-实现方案确定

* 数据开发过程

-表落地

-写sql语句实现业务逻辑

-部署代码

-数据测试

-试运行与上线

在接下来的客户基本属性表开发中演示开发的流程。

## 8.2 用户画像开发--客户基本属性表

|  |
| --- |
| --用户画像-客户基本属性模型表  create database if not exists gdm;  create table if not exists gdm.itcast\_gdm\_user\_basic(  user\_id string ,--用户ID  user\_name string ,--用户登陆名  user\_sex string ,--用户性别  user\_birthday string ,--用户生日  user\_age bigint ,--用户年龄  constellation string ,--用户星座  province string ,--省份  city string ,--城市  city\_level string ,--城市等级  hex\_mail string ,--邮箱  op\_mail string ,--邮箱运营商  hex\_phone string ,--手机号  fore\_phone string ,--手机前3位  op\_phone string ,--手机运营商  add\_time timestamp ,--注册时间  login\_ip string ,--登陆ip地址  login\_source string ,--登陆来源  request\_user string ,--邀请人  total\_mark bigint ,--会员积分  used\_mark bigint ,--已使用积分  level\_name string ,--会员等级名称  blacklist bigint ,--用户黑名单  is\_married bigint ,--婚姻状况  education string ,--学历  monthly\_money double ,--收入  profession string ,--职业  sex\_model bigint ,--性别模型  is\_pregnant\_woman bigint ,--是否孕妇  is\_have\_children bigint ,--是否有小孩  children\_sex\_rate double ,--孩子性别概率  children\_age\_rate double ,--孩子年龄概率  is\_have\_car bigint ,--是否有车  potential\_car\_user\_rate double ,--潜在汽车用户概率  phone\_brand string ,--使用手机品牌  phone\_brand\_level string ,--使用手机品牌档次  phone\_cnt bigint ,--使用多少种不同的手机  change\_phone\_rate bigint ,--更换手机频率  majia\_flag string ,--马甲标志  majie\_account\_cnt bigint ,--马甲账号数量  loyal\_model bigint ,--用户忠诚度  shopping\_type\_model bigint ,--用户购物类型  figure\_model bigint ,--身材  stature\_model bigint ,--身高  dw\_date timestamp  ) partitioned by (dt string); |

该模型表其基本信息主要来源于用户表、用户调查表。有静态信息和动态信息、后面的一些是数据挖掘模型（数据挖掘模型比较多，逻辑比较复杂，在机器学习课程中给大家介绍）。

|  |
| --- |
| #\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  --客户基本属性模型表BDM层  create database if not exists bdm;  create external table if not exists bdm.itcast\_bdm\_user(  user\_id string ,--用户ID  user\_name string ,--用户登陆名  user\_sex string ,--用户性别  user\_birthday string ,--用户生日  user\_age bigint ,--用户年龄  constellation string ,--用户星座  province string ,--省份  city string ,--城市  city\_level string ,--城市等级  hex\_mail string ,--邮箱  op\_mail string ,--邮箱运营商  hex\_phone string ,--手机号  fore\_phone string ,--手机前3位  op\_phone string ,--手机运营商  add\_time string ,--注册时间  login\_ip string ,--登陆ip地址  login\_source string ,--登陆来源  request\_user string ,--邀请人  total\_mark bigint ,--会员积分  used\_mark bigint ,--已使用积分  level\_name string ,--会员等级名称  blacklist bigint ,--用户黑名单  is\_married bigint ,--婚姻状况  education string ,--学历  monthly\_money double ,--收入  profession string --职业  ) partitioned by (dt string)  row format delimited fields terminated by ',';  alter table itcast\_bdm\_user add partition (dt='2017-01-01') location '/business/itcast\_bdm\_user/2017-01-01';  --客户基本属性表FDM层  create database if not exists fdm;  create table if not exists fdm.itcast\_fdm\_user\_wide(  user\_id string ,--用户ID  user\_name string ,--用户登陆名  user\_sex string ,--用户性别  user\_birthday string ,--用户生日  user\_age bigint ,--用户年龄  constellation string ,--用户星座  province string ,--省份  city string ,--城市  city\_level string ,--城市等级  hex\_mail string ,--邮箱  op\_mail string ,--邮箱运营商  hex\_phone string ,--手机号  fore\_phone string ,--手机前3位  op\_phone string ,--手机运营商  add\_time string ,--注册时间  login\_ip string ,--登陆ip地址  login\_source string ,--登陆来源  request\_user string ,--邀请人  total\_mark bigint ,--会员积分  used\_mark bigint ,--已使用积分  level\_name string ,--会员等级名称  blacklist bigint ,--用户黑名单  is\_married bigint ,--婚姻状况  education string ,--学历  monthly\_money double ,--收入  profession string ,--职业  dw\_date timestamp  ) partitioned by (dt string);  --加载数据  insert overwrite table fdm.itcast\_fdm\_user\_wide partition(dt='2017-01-01')  select  t.user\_id,  t.user\_name,  t.user\_sex,  t.user\_birthday,  t.user\_age,  t.constellation,  t.province,  t.city,  t.city\_level,  t.hex\_mail,  t.op\_mail,  t.hex\_phone,  t.fore\_phone,  t.op\_phone,  t.add\_time,  t.login\_ip,  t.login\_source,  t.request\_user,  t.total\_mark,  t.used\_mark,  t.level\_name,  t.blacklist,  t.is\_married,  t.education,  t.monthly\_money,  t.profession,  from\_unixtime(unix\_timestamp()) dw\_date  from bdm.itcast\_bdm\_user t where dt='2017-01-01';  --用户画像-客户基本属性模型表GDM层  create database if not exists gdm;  create table if not exists gdm.itcast\_gdm\_user\_basic(  user\_id string ,--用户ID  user\_name string ,--用户登陆名  user\_sex string ,--用户性别  user\_birthday string ,--用户生日  user\_age bigint ,--用户年龄  constellation string ,--用户星座  province string ,--省份  city string ,--城市  city\_level string ,--城市等级  hex\_mail string ,--邮箱  op\_mail string ,--邮箱运营商  hex\_phone string ,--手机号  fore\_phone string ,--手机前3位  op\_phone string ,--手机运营商  add\_time string ,--注册时间  login\_ip string ,--登陆ip地址  login\_source string ,--登陆来源  request\_user string ,--邀请人  total\_mark bigint ,--会员积分  used\_mark bigint ,--已使用积分  level\_name string ,--会员等级名称  blacklist bigint ,--用户黑名单  is\_married bigint ,--婚姻状况  education string ,--学历  monthly\_money double ,--收入  profession string ,--职业  sex\_model bigint ,--性别模型  is\_pregnant\_woman bigint ,--是否孕妇  is\_have\_children bigint ,--是否有小孩  children\_sex\_rate double ,--孩子性别概率  children\_age\_rate double ,--孩子年龄概率  is\_have\_car bigint ,--是否有车  potential\_car\_user\_rate double ,--潜在汽车用户概率  phone\_brand string ,--使用手机品牌  phone\_brand\_level string ,--使用手机品牌档次  phone\_cnt bigint ,--使用多少种不同的手机  change\_phone\_rate bigint ,--更换手机频率  majia\_flag string ,--马甲标志  majie\_account\_cnt bigint ,--马甲账号数量  loyal\_model bigint ,--用户忠诚度  shopping\_type\_model bigint ,--用户购物类型  figure\_model bigint ,--身材  stature\_model bigint ,--身高  dw\_date timestamp  ) partitioned by (dt string);  --加载数据  insert overwrite table gdm.itcast\_gdm\_user\_basic partition(dt='2017-01-01')  select  t.user\_id,  t.user\_name,  t.user\_sex,  t.user\_birthday,  t.user\_age,  t.constellation,  t.province,  t.city,  t.city\_level,  t.hex\_mail,  t.op\_mail,  t.hex\_phone,  t.fore\_phone,  t.op\_phone,  t.add\_time,  t.login\_ip,  t.login\_source,  t.request\_user,  t.total\_mark,  t.used\_mark,  t.level\_name,  t.blacklist,  t.is\_married,  t.education,  t.monthly\_money,  t.profession,  null sex\_model,--数据挖掘模型-开始  null is\_pregnant\_woman,  null is\_have\_children,  null children\_sex\_rate,  null children\_age\_rate,  null is\_have\_car,  null potential\_car\_user\_rate,  null phone\_brand,  null phone\_brand\_level,  null phone\_cnt,  null change\_phone\_rate,  null majia\_flag,  null majie\_account\_cnt,  null loyal\_model,  null shopping\_type\_model,  null figure\_model,  null stature\_model,--数据挖掘模型-结束  from\_unixtime(unix\_timestamp()) dw\_date  from (select \* from fdm.itcast\_fdm\_user\_wide where dt='2017-01-01') t; |
| 演示模型表开发脚本：  ######################  #名称：客户基本属性模型表  # itcast\_gdm\_user\_basic.sh  ######################  #!/bin/sh  yesterday=`date -d '-1 day' "+%Y-%m-%d"`  if [ $1 ];then  yesterday=$1  fi  SPARK\_SUBMIT\_INFO="/export/servers/spark/bin/spark-sql --master spark://node1:7077 --executor-memory 1g --total-executor-cores 2 --conf spark.sql.warehouse.dir=hdfs://node1:9000/user/hive/warehouse"  SOURCE\_DATA="/root/source\_data"  SQL\_BDM="create database if not exists bdm;  create external table if not exists bdm.itcast\_bdm\_user(  user\_id string ,--用户ID  user\_name string ,--用户登陆名  user\_sex string ,--用户性别  user\_birthday string ,--用户生日  user\_age bigint ,--用户年龄  constellation string ,--用户星座  province string ,--省份  city string ,--城市  city\_level string ,--城市等级  hex\_mail string ,--邮箱  op\_mail string ,--邮箱运营商  hex\_phone string ,--手机号  fore\_phone string ,--手机前3位  op\_phone string ,--手机运营商  add\_time string ,--注册时间  login\_ip string ,--登陆ip地址  login\_source string ,--登陆来源  request\_user string ,--邀请人  total\_mark bigint ,--会员积分  used\_mark bigint ,--已使用积分  level\_name string ,--会员等级名称  blacklist bigint ,--用户黑名单  is\_married bigint ,--婚姻状况  education string ,--学历  monthly\_money double ,--收入  profession string --职业  ) partitioned by (dt string)  row format delimited fields terminated by ','  location '/business/bdm/itcast\_bdm\_user' ;  alter table bdm.itcast\_bdm\_user add partition (dt='$yesterday');"  SQL\_FDM="create database if not exists fdm;  create table if not exists fdm.itcast\_fdm\_user\_wide(  user\_id string ,--用户ID  user\_name string ,--用户登陆名  user\_sex string ,--用户性别  user\_birthday string ,--用户生日  user\_age bigint ,--用户年龄  constellation string ,--用户星座  province string ,--省份  city string ,--城市  city\_level string ,--城市等级  hex\_mail string ,--邮箱  op\_mail string ,--邮箱运营商  hex\_phone string ,--手机号  fore\_phone string ,--手机前3位  op\_phone string ,--手机运营商  add\_time string ,--注册时间  login\_ip string ,--登陆ip地址  login\_source string ,--登陆来源  request\_user string ,--邀请人  total\_mark bigint ,--会员积分  used\_mark bigint ,--已使用积分  level\_name string ,--会员等级名称  blacklist bigint ,--用户黑名单  is\_married bigint ,--婚姻状况  education string ,--学历  monthly\_money double ,--收入  profession string ,--职业  dw\_date timestamp  ) partitioned by (dt string);"  ##加载数据  LOAD\_FDM="  insert overwrite table fdm.itcast\_fdm\_user\_wide partition(dt='$yesterday')  select  t.user\_id,  t.user\_name,  t.user\_sex,  t.user\_birthday,  t.user\_age,  t.constellation,  t.province,  t.city,  t.city\_level,  t.hex\_mail,  t.op\_mail,  t.hex\_phone,  t.fore\_phone,  t.op\_phone,  t.add\_time,  t.login\_ip,  t.login\_source,  t.request\_user,  t.total\_mark,  t.used\_mark,  t.level\_name,  t.blacklist,  t.is\_married,  t.education,  t.monthly\_money,  t.profession,  from\_unixtime(unix\_timestamp()) dw\_date  from bdm.itcast\_bdm\_user t where dt='$yesterday';"  SQL\_GDM="create database if not exists gdm;  create table if not exists gdm.itcast\_gdm\_user\_basic(  user\_id string ,--用户ID  user\_name string ,--用户登陆名  user\_sex string ,--用户性别  user\_birthday string ,--用户生日  user\_age bigint ,--用户年龄  constellation string ,--用户星座  province string ,--省份  city string ,--城市  city\_level string ,--城市等级  hex\_mail string ,--邮箱  op\_mail string ,--邮箱运营商  hex\_phone string ,--手机号  fore\_phone string ,--手机前3位  op\_phone string ,--手机运营商  add\_time string ,--注册时间  login\_ip string ,--登陆ip地址  login\_source string ,--登陆来源  request\_user string ,--邀请人  total\_mark bigint ,--会员积分  used\_mark bigint ,--已使用积分  level\_name string ,--会员等级名称  blacklist bigint ,--用户黑名单  is\_married bigint ,--婚姻状况  education string ,--学历  monthly\_money double ,--收入  profession string ,--职业  sex\_model bigint ,--性别模型  is\_pregnant\_woman bigint ,--是否孕妇  is\_have\_children bigint ,--是否有小孩  children\_sex\_rate double ,--孩子性别概率  children\_age\_rate double ,--孩子年龄概率  is\_have\_car bigint ,--是否有车  potential\_car\_user\_rate double,--潜在汽车用户概率  phone\_brand string ,--使用手机品牌  phone\_brand\_level string ,--使用手机品牌档次  phone\_cnt bigint ,--使用多少种不同的手机  change\_phone\_rate bigint ,--更换手机频率  majia\_flag string ,--马甲标志  majie\_account\_cnt bigint ,--马甲账号数量  loyal\_model bigint ,--用户忠诚度  shopping\_type\_model bigint ,--用户购物类型  figure\_model bigint ,--身材  stature\_model bigint ,--身高  dw\_date timestamp  ) partitioned by (dt string);"  ##加载数据到GDM  LOAD\_GDM="insert overwrite table gdm.itcast\_gdm\_user\_basic partition(dt='$yesterday')  select  t.user\_id,  t.user\_name,  t.user\_sex,  t.user\_birthday,  t.user\_age,  t.constellation,  t.province,  t.city,  t.city\_level,  t.hex\_mail,  t.op\_mail,  t.hex\_phone,  t.fore\_phone,  t.op\_phone,  t.add\_time,  t.login\_ip,  t.login\_source,  t.request\_user,  t.total\_mark,  t.used\_mark,  t.level\_name,  t.blacklist,  t.is\_married,  t.education,  t.monthly\_money,  t.profession,  null sex\_model,--数据挖掘模型-开始  null is\_pregnant\_woman,  null is\_have\_children,  null children\_sex\_rate,  null children\_age\_rate,  null is\_have\_car,  null potential\_car\_user\_rate,  null phone\_brand,  null phone\_brand\_level,  null phone\_cnt,  null change\_phone\_rate,  null majia\_flag,  null majie\_account\_cnt,  null loyal\_model,  null shopping\_type\_model,  null figure\_model,  null stature\_model,--数据挖掘模型-结束  from\_unixtime(unix\_timestamp()) dw\_date  from (select \* from fdm.itcast\_fdm\_user\_wide where dt='$yesterday') t;"  ##创建BDM层表  echo "${SQL\_BDM}"  $SPARK\_SUBMIT\_INFO -e "${SQL\_BDM}"  ##添加数据到BDM  hdfs dfs -put $SOURCE\_DATA/itcast\_bdm\_user.txt /business/bdm/itcast\_bdm\_user/"dt=$yesterday"  ##创建FDM层表  echo "${SQL\_FDM}"  $SPARK\_SUBMIT\_INFO -e "${SQL\_FDM}"  ##导入数据到FDM  echo "${LOAD\_FDM}"  $SPARK\_SUBMIT\_INFO -e "${LOAD\_FDM}"  ##创建GDM层表  echo "${SQL\_GDM}"  $SPARK\_SUBMIT\_INFO -e "${SQL\_GDM}"  ##导入GDM数据  echo "${LOAD\_GDM}"  $SPARK\_SUBMIT\_INFO -e "${LOAD\_GDM}" |

## 8.3 用户画像开发--订单表宽表

|  |
| --- |
| --订单宽表模型  create database if not exists gdm;  create table if not exists gdm.itcast\_gdm\_order(  order\_id string,--订单ID  order\_no string,--订单号  order\_date string,--订单日期  user\_id string,--用户ID  user\_name string,--登录名  order\_money double,--订单金额  order\_type string,--订单类型  order\_status string,--订单状态  pay\_status string,--支付状态  pay\_type string,--支付方式 1、在线支付，2、货到付款  order\_source string,--订单来源  consignee string,--收货人姓名  area\_id string,--收货人地址ID  area\_name string,--地址ID对应的地址段（粒度到县）  address string,--收货人地址（手工填写的地址）  mobile string,--收货人手机号  telphone string,--收货人电话  coupon\_id bigint,--使用代金券ID  coupon\_money double,--使用代金券金额  carriage\_money double,--运费  create\_time timestamp,--创建时间  update\_time timestamp,--更新时间  dw\_date timestamp  ) partitioned by (dt string); |

|  |
| --- |
| --订单主要信息表BDM层  create database if not exists bdm;  create external table if not exists bdm.itcast\_bdm\_order(  order\_id string, --订单ID  order\_no string, --订单号  order\_date string, --订单日期  user\_id string, --用户ID  user\_name string, --登录名  order\_money double, --订单金额  order\_type string, --订单类型  order\_status string, --订单状态  pay\_status string, --支付状态  pay\_type string, --支付方式 1、在线支付，2、货到付款  order\_source string, --订单来源  update\_time timestamp, --订单更新时间  dw\_date timestamp  ) partitioned by (dt string)  row format delimited fields terminated by ','  lines terminated by '\n';  alter table bdm.itcast\_bdm\_order add partition (dt='2017-01-01') location '/business/itcast\_bdm\_order/2017-01-01';  hdfs dfs -put fdm\_order.txt /business/itcast\_bdm\_order/2017-01-01 |

|  |
| --- |
| --订单主要信息表FDM层  create database if not exists fdm;  create table if not exists fdm.itcast\_fdm\_order(  order\_id string, --订单ID  order\_no string, --订单号  order\_date string, --订单日期  user\_id string, --用户ID  user\_name string, --登录名  order\_money double, --订单金额  order\_type string, --订单类型  order\_status string, --订单状态  pay\_status string, --支付状态  pay\_type string, --支付方式 1、在线支付，2、货到付款  order\_source string, --订单来源  update\_time timestamp, --订单更新时间  dw\_date timestamp  ) partitioned by (dt string);  --加载数据  insert overwrite table fdm.itcast\_fdm\_order partition(dt='2017-01-01')  select  t.order\_id, --订单ID  t.order\_no, --订单号  t.order\_date, --订单日期  t.user\_id, --用户ID  t.user\_name, --登录名  t.order\_money, --订单金额  t.order\_type, --订单类型  t.order\_status, --订单状态  t.pay\_status, --支付状态  t.pay\_type, --支付方式  t.order\_source, --订单来源  t.update\_time timestamp,--订单更新时间  from\_unixtime(unix\_timestamp()) dw\_date  from bdm.itcast\_bdm\_order t where dt='2017-01-01'; |

|  |
| --- |
| -------订单详细信息表BDM层----------------  create database if not exists bdm;  create external table if not exists bdm.itcast\_bdm\_order\_desc(  order\_id string, --订单ID  order\_no string, --订单号  consignee string, --收货人姓名  area\_id string, --收货人地址ID  area\_name string, --地址ID对应的地址段  address string, --收货人地址  mobile string, --收货人手机号  telphone string, --收货人电话  coupon\_id bigint, --使用代金券ID  coupon\_money double, --使用代金券金额  carriage\_money double, --运费  create\_time timestamp, --创建时间  update\_time timestamp, --更新时间  dw\_date timestamp  )partitioned by (dt string)  row format delimited fields terminated by ',';  alter table bdm.itcast\_bdm\_order\_desc add partition (dt='2017-01-01') location '/business/itcast\_bdm\_order\_desc/2017-01-01';  hdfs dfs -put itcast\_bdm\_order\_desc.txt /business/itcast\_bdm\_order\_desc/2017-01-01 |

|  |
| --- |
| -----订单主要信息表FDM层  create database if not exists fdm;  create table if not exists fdm.itcast\_fdm\_order\_desc(  order\_id string, --订单ID  order\_no string, --订单号  consignee string, --收货人姓名  area\_id string, --收货人地址ID  area\_name string, --地址ID对应的地址段  address string, --收货人地址  mobile string, --收货人手机号  telphone string, --收货人电话  coupon\_id bigint, --使用代金券ID  coupon\_money double, --使用代金券金额  carriage\_money double, --运费  create\_time timestamp, --创建时间  update\_time timestamp, --更新时间  dw\_date timestamp  ) partitioned by (dt string);  ------加载数据  insert overwrite table fdm.itcast\_fdm\_order\_desc partition(dt='2017-01-01')  select  t.order\_id, --订单ID  t.order\_no, --订单号  t.consignee, --收货人姓名  t.area\_id, --收货人地址ID  t.area\_name, --地址ID对应的地址段  t.address, --收货人地址  t.mobile, --收货人手机号  t.telphone, --收货人电话  t.coupon\_id, --使用代金券ID  t.coupon\_money, --使用代金券金额  t.carriage\_money, --运费  t.create\_time, --创建时间  t.update\_time, --更新时间  from\_unixtime(unix\_timestamp()) dw\_date  from bdm.itcast\_bdm\_order\_desc t where dt='2017-01-01'; |

|  |
| --- |
| --------订单宽表模型表GDM  create database if not exists gdm;  create external table if not exists gdm.itcast\_gdm\_order(  order\_id string, --订单ID  order\_no string, --订单号  order\_date string, --订单日期  user\_id string, --用户ID  user\_name string, --登录名  order\_money double, --订单金额  order\_type string, --订单类型  order\_status string,--订单状态  pay\_status string, --支付状态  pay\_type string, --支付方式 1、在线支付，2、货到付款  order\_source string,--订单来源  consignee string, --收货人姓名  area\_id string, --收货人地址ID  area\_name string, --地址ID对应的地址段（粒度到县）  address string, --收货人地址（手工填写的地址）  mobile string, --收货人手机号  telphone string, --收货人电话  coupon\_id bigint, --使用代金券ID  coupon\_money double,--使用代金券金额  carriage\_money double,--运费  create\_time timestamp,--创建时间  update\_time timestamp,--更新时间  dw\_date timestamp  ) partitioned by (dt string);  ---加载数据生成订单宽表  insert overwrite table gdm.itcast\_gdm\_order partition(dt='2017-01-01')  select  a.order\_id, --订单ID  a.order\_no, --订单号  a.order\_date, --订单日期  a.user\_id, --用户ID  a.user\_name, --用户名  a.order\_money, --订单金额  a.order\_type, --订单类型  a.order\_status, --订单状态  a.pay\_status, --支付类型  a.pay\_type, --支付方式  a.order\_source, --订单来源  b.consignee, --收货人姓名  b.area\_id, --收货人地址ID  b.area\_name, --地址ID对应的地址段  b.address, --收货人地址  b.mobile, --收货人手机号  b.telphone, --收货人电话  b.coupon\_id, --使用代金券ID  b.coupon\_money, --使用代金券金额  b.carriage\_money, --运费  b.create\_time, --创建时间  b.update\_time, --更新时间  from\_unixtime(unix\_timestamp()) dw\_date  from (select \* from fdm.itcast\_fdm\_order where dt='2017-01-01') a  join (select \* from fdm.itcast\_fdm\_order\_desc where dt='2017-01-01') b on a.order\_id=b.order\_id; |

## 8.4 用户画像开发--客户消费订单表

|  |
| --- |
| --用户画像 客户消费订单表  create database if not exists gdm;  create table if not exists gdm.itcast\_gdm\_user\_consume\_order(  user\_id string, --客户ID  first\_order\_time timestamp, --第一次消费时间  last\_order\_time timestamp, --最近一次消费时间  first\_order\_ago bigint, --首单距今时间  last\_order\_ago bigint, --尾单距今时间  month1\_hg\_order\_cnt bigint, --近30天购买次数（不含退拒）  month1\_hg\_order\_amt double, --近30天购买金额（不含退拒）  month2\_hg\_order\_cnt bigint, --近60天购买次数（不含退拒）  month2\_hg\_order\_amt double, --近60天购买金额（不含退拒）  month3\_hg\_order\_cnt bigint, --近90天购买次数（不含退拒）  month3\_hg\_order\_amt double, --近90天购买金额（不含退拒）  month1\_order\_cnt bigint, --近30天购买次数（含退拒）  month1\_order\_amt double, --近30天购买金额（含退拒）  month2\_order\_cnt bigint, --近60天购买次数（含退拒）  month2\_order\_amt double, --近60天购买金额（含退拒）  month3\_order\_cnt bigint, --近90天购买次数（含退拒）  month3\_order\_amt double, --近90天购买金额（含退拒）  max\_order\_amt double, --最大消费金额  min\_order\_amt double, --最小消费金额  total\_order\_cnt bigint, --累计消费次数（不含退拒）  total\_order\_amt double, --累计消费金额（不含退拒）  user\_avg\_amt double, --客单价（含退拒）  month3\_user\_avg\_amt double, --近90天的客单价  common\_address string, --常用收货地址  common\_paytype string, --常用支付方式  month1\_cart\_cnt bigint, --近30天购物车的次数  month1\_cart\_goods\_cnt bigint, --近30天购物车商品件数  month1\_cart\_submit\_cnt bigint, --近30天购物车提交商品件数  month1\_cart\_rate double, --近30天购物车成功率  month1\_cart\_cancle\_cnt double, --近30天购物车放弃件数  return\_cnt bigint, --退货商品数量  return\_amt double, --退货商品金额  reject\_cnt bigint, --拒收商品数量  reject\_amt double, --拒收商品金额  last\_return\_time timestamp, --最近一次退货时间  school\_order\_cnt bigint, --学校下单总数  company\_order\_cnt bigint, --单位下单总数  home\_order\_cnt bigint, --家里下单总数  forenoon\_order\_cnt bigint, --上午下单总数  afternoon\_order\_cnt bigint, --下午下单总数  night\_order\_cnt bigint, --晚上下单总数  morning\_order\_cnt bigint, --凌晨下单总数  dw\_date timestamp  ) partitioned by (dt string); |

|  |
| --- |
| ---客户消费订单模型表-临时表01  drop table if exists gdm.itcast\_gdm\_user\_consume\_order\_temp\_01;  CREATE TABLE gdm.itcast\_gdm\_user\_consume\_order\_temp\_01 AS  SELECT  t.user\_id,  MIN(order\_date) first\_order\_time,--第一次消费时间  MAX(order\_date) last\_order\_time,--最近一次消费时间  DATEDIFF(MIN(order\_date), '2017-01-01') first\_order\_ago,--首单距今时间  DATEDIFF(MAX(order\_date), '2017-01-01') last\_order\_ago,--尾单距今时间  SUM(  CASE  WHEN t.dat\_30 = 1  AND t.order\_flag = 0  THEN 1  END  ) month1\_hg\_order\_cnt,--近30天购买次数（不含退拒）  SUM(  CASE  WHEN t.dat\_30 = 1  AND t.order\_flag = 0  THEN t.order\_money  END  ) month1\_hg\_order\_amt,--近30天购买金额（不含退拒）  SUM(  CASE  WHEN t.dat\_60 = 1  AND t.order\_flag = 0  THEN 1  END  ) month2\_hg\_order\_cnt,--近60天购买次数（不含退拒）  SUM(  CASE  WHEN t.dat\_60 = 1  AND t.order\_flag = 0  THEN t.order\_money  END  ) month2\_hg\_order\_amt,--近60天购买金额（不含退拒）  SUM(  CASE  WHEN t.dat\_90 = 1  AND t.order\_flag = 0  THEN 1  END  ) month3\_hg\_order\_cnt,--近90天购买次数（不含退拒）  SUM(  CASE  WHEN t.dat\_90 = 1  AND t.order\_flag = 0  THEN t.order\_money  END  ) month3\_hg\_order\_amt,--近90天购买金额（不含退拒）  SUM(dat\_30) month1\_order\_cnt,--近30天购买次数（含退拒）  SUM(  CASE  WHEN t.dat\_30 = 1  THEN t.order\_money  END  ) month1\_order\_amt,--近30天购买金额（含退拒）  SUM(dat\_60) month2\_order\_cnt,--近60天购买次数（含退拒）  SUM(  CASE  WHEN t.dat\_60 = 1  THEN t.order\_money  END  ) month2\_order\_amt,--近60天购买金额（含退拒）  SUM(dat\_90) month3\_order\_cnt,--近90天购买次数（含退拒）  SUM(  CASE  WHEN t.dat\_90 = 1  THEN t.order\_money  END  ) month3\_order\_amt,--近90天购买金额（含退拒）  MAX(t.order\_money) max\_order\_amt,--最大消费金额  MIN(t.order\_money) min\_order\_amt,--最小消费金额  SUM(  CASE  WHEN t.order\_flag = 0  THEN 1  END  ) total\_order\_cnt,--累计消费次数（不含退拒）  SUM(  CASE  WHEN t.order\_flag = 0  THEN t.order\_money  END  ) total\_order\_amt,--累计消费金额（不含退拒）  SUM(coupon\_money) total\_coupon\_amt,--累计使用代金券金额  SUM(t.order\_money) / COUNT(1) user\_avg\_amt,--客单价（含退拒）  0 month3\_user\_avg\_amt,--近90天的客单价（含退拒）  0 common\_address,--常用收获地址  0 common\_paytype,--常用支付方式  0 month1\_cart\_cnt,--最近30天购物车次数  0 month1\_cart\_goods\_cnt,--最近30天购物车商品件数  0 month1\_cart\_submit\_cnt,--最近30天购物车提交商品件数  0 month1\_order\_rate,--最近30天购物车成功率  0 month1\_cart\_cancle\_cnt,--最近30天购物车放弃件数  SUM(  CASE  WHEN t.order\_status = 3  THEN t1.goods\_amount  END  ) return\_cnt,--退货商品数量  SUM(  CASE  WHEN t.order\_status = 3  THEN t.order\_money  END  ) return\_amt,--退货商品金额  SUM(  CASE  WHEN t.order\_status = 4  THEN t1.goods\_amount  END  ) reject\_cnt,--拒收商品数量  SUM(  CASE  WHEN t.order\_status = 4  THEN t.order\_money  END  ) reject\_amt,--拒收商品金额  MAX(  CASE  WHEN t.order\_status = 3  THEN t.order\_date  END  ) last\_return\_time,--最近一次退货时间  SUM(  CASE  WHEN t2.order\_addr = 1  THEN 1  END  ) school\_order\_cnt,--学校下单总数  SUM(  CASE  WHEN t2.order\_addr = 2  THEN 1  END  ) company\_order\_cnt,--单位下单总数  SUM(  CASE  WHEN t2.order\_addr = 3  THEN 1  END  ) home\_order\_cnt,--家里下单总数  SUM(  CASE  WHEN t.order\_hour >= 8  AND t.order\_hour <= 11  THEN 1  END  ) forenoon\_order\_cnt,--上午下单总数  SUM(  CASE  WHEN t.order\_hour >= 12  AND t.order\_hour <= 18  THEN 1  END  ) afternoon\_order\_cnt,--下午下单总数  SUM(  CASE  WHEN t.order\_hour >= 19  AND t.order\_hour <= 22  THEN 1  END  ) night\_order\_cnt,--晚上下单总数  SUM(  CASE  WHEN t.order\_hour >= 23  AND t.order\_hour <= 7  THEN 1  END  ) morning\_order\_cnt --凌晨下单总数  FROM  (SELECT  a.\*,  (  CASE  WHEN order\_date >= DATE\_SUB('2017-01-01', 29)  AND order\_date <= '2017-01-01'  THEN 1  END  ) dat\_30,  (  CASE  WHEN order\_date >= DATE\_SUB('2017-01-01', 59)  AND order\_date <= '2017-01-01'  THEN 1  END  ) dat\_60,  (  CASE  WHEN order\_date >= DATE\_SUB('2017-01-01', 89)  AND order\_date <= '2017-01-01'  THEN 1  END  ) dat\_90,  (  CASE  WHEN a.order\_status IN (3, 4)  THEN 1  ELSE 0  END  ) order\_flag,--退货与拒收标示  HOUR(order\_date) order\_hour  FROM  gdm.itcast\_gdm\_order a  WHERE dt = '2017-01-01') t  LEFT JOIN  (SELECT  order\_id,  goods\_amount  FROM  fdm.itcast\_fdm\_order\_goods) t1  ON (t.order\_id = t1.order\_id)  LEFT JOIN  (SELECT  user\_id,  order\_addr  FROM  gdm.itcast\_user\_order\_addr\_model) t2  ON (t.user\_id = t2.user\_id)  GROUP BY t.user\_id ; |

|  |
| --- |
| ---购物车临时模型表--临时表02  DROP TABLE IF EXISTS gdm.itcast\_gdm\_user\_consume\_order\_temp\_02;  CREATE TABLE gdm.itcast\_gdm\_user\_consume\_order\_temp\_02 AS  SELECT  user\_id,  COUNT(1) month1\_cart\_cnt,--最近30天购物车次数  SUM(goods\_num) month1\_cart\_goods\_cnt,--最近30天购物车商品件数  SUM(  CASE  WHEN sumbit\_time <> ''  THEN goods\_num  ELSE 0  END  ) month1\_cart\_submit\_cnt,--最近30天购物车提交商品件数  '' month1\_cart\_rate,--最近30天购物车成功率  SUM(  CASE  WHEN cancle\_time <> ''  THEN goods\_num  ELSE 0  END  ) month1\_cart\_cancle\_cnt --最近30天购物车放弃件数  FROM  fdm.itcast\_fdm\_order\_cart  WHERE dt = '2017-01-01'  AND to\_date (add\_time) >= DATE\_SUB('2017-01-01', 29)  AND to\_date (add\_time) <= '2017-01-01'  GROUP BY user\_id ; |

|  |
| --- |
| ---购物车临时模型表---常用地址和常用支付方式-临时表03  drop table if exists gdm.itcast\_gdm\_user\_consume\_order\_temp\_03;  create table gdm.gdm\_user\_consume\_order\_temp\_03 as  select  t.user\_id,  t.con,  t.type,  t.cnt  from  (select  b.user\_id,  b.con,  b.type,  b.cnt,  row\_number() over(distribute by b.user\_id,  b.type sort by b.cnt,  b.type desc) rn  from  (select  a.user\_id,concat(  coalesce(area\_name, ''),  coalesce(address, '')) con,  'address' type,  count(1) cnt  from  gdm.itcast\_gdm\_order a where dt = '2017-01-01'  group by a.user\_id,  concat(  coalesce(area\_name, ''),  coalesce(address, '')  )  union  all  select  a.user\_id,  a.pay\_type con,  'pay\_type' type,  count(1) cnt  from  gdm.itcast\_gdm\_order a  where dt = '2017-01-01'  group by a.user\_id,  a.pay\_type) b) t  where t.rn = 1 ; |

|  |
| --- |
| --购物车表和订单表整合  drop table if exists gdm.itcast\_gdm\_user\_consume\_order\_temp\_100;  create table gdm.gdm\_user\_consume\_order\_temp\_100 as  select  a.user\_id  from  (select  user\_id  from  gdm.itcast\_gdm\_user\_consume\_order\_temp\_01  union  all  select  user\_id  from  gdm.itcast\_gdm\_user\_consume\_order\_temp\_02) a  group by a.user\_id ; |

|  |
| --- |
| --生成最终的客户消费订单表  INSERT overwrite TABLE gdm.itcast\_gdm\_user\_consume\_order PARTITION (dt = '2017-01-01')  SELECT  t.user\_id, --客户ID  t1.first\_order\_time, --常用地址和常用支付方式次消费时间  t1.last\_order\_time, --最近一次消费时间  t1.first\_order\_ago, --首单距今时间  t1.last\_order\_ago, --尾单距今时间  t1.month1\_hg\_order\_cnt,--近30天购买次数（不含退拒）  t1.month1\_hg\_order\_amt,--近30天购买金额（不含退拒）  t1.month2\_hg\_order\_cnt,--近60天购买次数（不含退拒）  t1.month2\_hg\_order\_amt,--近60天购买金额（不含退拒）  t1.month3\_hg\_order\_cnt,--近90天购买次数（不含退拒）  t1.month3\_hg\_order\_amt,--近90天购买金额（不含退拒）  t1.month1\_order\_cnt, --近30天购买次数（含退拒）  t1.month1\_order\_amt, --近30天购买金额（含退拒）  t1.month2\_order\_cnt, --近60天购买次数（含退拒）  t1.month2\_order\_amt, --近60天购买金额（含退拒）  t1.month3\_order\_cnt, --近90天购买次数（含退拒）  t1.month3\_order\_amt, --近90天购买金额（含退拒）  t1.max\_order\_amt, --最大消费金额  t1.min\_order\_amt, --最小消费金额  t1.total\_order\_cnt, --累计消费次数（不含退拒）  t1.total\_order\_amt, --累计消费金额（不含退拒）  t1.user\_avg\_amt, --客单价（含退拒）  (  CASE  WHEN t1.month3\_order\_cnt <> 0  THEN t1.month3\_order\_amt / t1.month3\_order\_cnt  ELSE 0  END  ) month3\_user\_avg\_amt, --近90天的客单价(含退拒)  t3.common\_address, --常用收货地址  t3.common\_paytype, --常用支付方式  t2.month1\_cart\_cnt, --近30天购物车的次数  t2.month1\_cart\_goods\_cnt, --近30天购物车商品件数  t2.month1\_cart\_submit\_cnt, --近30天购物车提交商品件数  (  CASE  WHEN t1.month1\_order\_cnt <> 0  THEN t2.month1\_cart\_submit\_cnt / t2.month1\_cart\_goods\_cnt  ELSE 0  END  ) month1\_cart\_rate, --近30天购物车成功率  t2.month1\_cart\_cancle\_cnt, --近30天购物车放弃件数  t1.return\_cnt, --退货商品数量  t1.return\_amt, --退货商品金额  t1.reject\_cnt, --拒收商品数量  t1.reject\_amt, --拒收商品金额  t1.last\_return\_time, --最近一次退货时间  t1.school\_order\_cnt, --学校下单总数  t1.company\_order\_cnt, --单位下单总数  t1.home\_order\_cnt, --家里下单总数  t1.forenoon\_order\_cnt, --上午下单总数  t1.afternoon\_order\_cnt, --下午下单总数  t1.night\_order\_cnt, --晚上下单总数  t1.morning\_order\_cnt, --凌晨下单总数  FROM\_UNIXTIME(UNIX\_TIMESTAMP()) dw\_date  FROM  gdm.itcast\_gdm\_user\_consume\_order\_temp\_100 t  LEFT JOIN gdm.itcast\_gdm\_user\_consume\_order\_temp\_01 t1  ON (t.user\_id = t1.user\_id)  LEFT JOIN gdm.itcast\_gdm\_user\_consume\_order\_temp\_02 t2  ON (t.user\_id = t2.user\_id)  LEFT JOIN  (SELECT  user\_id,  MAX(  CASE  WHEN type = 'address'  THEN con  END  ) common\_address,  MAX(  CASE  WHEN type = 'pay\_type'  THEN con  END  ) common\_paytype  FROM  gdm.itcast\_gdm\_user\_consume\_order\_temp\_03  GROUP BY user\_id) t3  ON (t.user\_id = t3.user\_id); |

## 8.5 用户画像开发--客户购买类目表

|  |
| --- |
| --用户画像 客户购买类目表  create database if not exists gdm;  create external table if not exists gdm.itcast\_gdm\_user\_buy\_category(  user\_id bigint , --客户ID  first\_cat bigint , --一级分类ID  first\_cat\_name string , --一级分类名称  second\_cat bigint , --二分类ID  second\_cat\_name string , --二级分类名称  third\_cat bigint , --三级分类ID  third\_cat\_name string , --三级分类名称  month1\_cat\_cnt bigint , --近30天购物类目次数  month1\_cat\_amt string , --近30天购物类目金额  month3\_cat\_cnt bigint , --近90天购物类目次数  month3\_cat\_amt string , --近90天购物类目金额  month6\_cat\_cnt bigint , --近180天购物类目次数  month6\_cat\_amt string , --近180天购物类目金额  total\_cat\_cnt bigint ，--累计购物类目次数  total\_cat\_amt string , --累计购物类目次数  month1\_cart\_cat\_cnt bigint , --近30天购物车类目次数  month3\_cart\_cat\_cnt bigint , --近90天购物车类目次数  month6\_cart\_cat\_cnt bigint , --近180天购物车类目次数  total\_cart\_cat\_cnt bigint , --累计购物车类目次数  last\_cat\_time timestamp , --最后一次购买类目时间  last\_cat\_ago bigint , --最后一次购买类目距今天数  dw\_date timestamp  )partitioned by (dt string); |

|  |
| --- |
| --计算订单中客户购买情况  drop table if exists gdm.itcast\_gdm\_user\_buy\_category\_temp;  CREATE TABLE gdm.itcast\_gdm\_user\_buy\_category\_temp AS  SELECT  a.user\_id,  '' first\_category\_id,  '' first\_category\_name,  '' second\_category\_id,  '' second\_catery\_name,  b.third\_cart third\_category\_id,  b.third\_cat\_name third\_category\_name,  SUM(  CASE  WHEN a.dat\_30 = 1  THEN b.goods\_amount  END  ) month1\_category\_cnt,  SUM(  CASE  WHEN a.dat\_30 = 1  THEN COALESCE(b.curr\_price,0)\* COALESCE(b.goods\_amount,0)  END  ) month1\_category\_amt,  SUM(  CASE  WHEN a.dat\_90 = 1  THEN b.goods\_amount  END  ) month3\_category\_cnt,  SUM(  CASE  WHEN a.dat\_90 = 1  THEN COALESCE(b.curr\_price, 0)\* COALESCE(b.goods\_amount, 0)  END  ) month3\_category\_amt,  SUM(  CASE  WHEN a.dat\_180 = 1  THEN b.goods\_amount  END  ) month6\_category\_cnt,  SUM(  CASE  WHEN a.dat\_180 = 1  THEN COALESCE(b.curr\_price, 0) \* COALESCE(b.goods\_amount, 0)  END  ) month6\_category\_amt,  SUM(b.goods\_amount) total\_category\_cnt,  SUM(  COALESCE(b.curr\_price, 0) \* COALESCE(b.goods\_amount, 0)  ) total\_category\_amt,  MAX(a.order\_date) last\_category\_time,  DATEDIFF(MIN(a.order\_date), '2017-01-01') last\_category\_ago,  FROM\_UNIXTIME(UNIX\_TIMESTAMP()) dw\_date  FROM  (SELECT  a.\*,  (  CASE  WHEN order\_date >= DATE\_SUB('2017-01-01', 29)  AND order\_date <= '2017-01-01'  THEN  1  END  ) dat\_30,  (  CASE  WHEN order\_date >= DATE\_SUB('2017-01-01', 89)  AND order\_date <= '2017-01-01'  THEN  1  END  ) dat\_90,  (  CASE  WHEN order\_date >= DATE\_SUB('2017-01-01', 179)  AND order\_date <= '2017-01-01'  THEN  1  END  ) dat\_180  FROM  fdm.itcast\_fdm\_order a  WHERE dt = '2017-01-01') a  JOIN  (SELECT  \*  FROM  fdm.itcast\_fdm\_order\_goods  WHERE dt = '2017-01-01') b  ON (a.user\_id = b.user\_id)  GROUP BY a.user\_id,b.third\_cart,b.third\_cat\_name; |

|  |
| --- |
| --购物车中类目情况  --购物车中类目情况  drop table if exists gdm.itcast\_gdm\_user\_cart\_category\_temp;  create table gdm.itcast\_gdm\_user\_cart\_category\_temp as  select  a.user\_id,  b.third\_cart,  sum(  case  when to\_date (add\_time) >= date\_sub('2017-01-01', 29)  and to\_date (add\_time) <= '2017-01-01'  then 1  end  ) month1\_category\_cnt,  sum(  case  when to\_date (add\_time) >= date\_sub('2017-01-01', 59)  and to\_date (add\_time) <= '2017-01-01'  then 1  end  ) month3\_category\_cnt,  sum(  case  when to\_date (add\_time) >= date\_sub('2017-01-01', 179)  and to\_date (add\_time) <= '2017-01-01'  then 1  end  ) month6\_category\_cnt,  count(1) total\_category\_cnt  from  (select  \*  from  fdm.itcast\_fdm\_order\_cart  where dt = '2017-01-01'  and to\_date (add\_time) >= date\_sub('2017-01-01', 179)  and to\_date (add\_time) <= '2017-01-01') a  left join  (select  goods\_id,  third\_cart  from  fdm.itcast\_fdm\_order\_goods  where dt = '2017-01-01'  group by goods\_id,  third\_cart) b  on (a.goods\_id = b.goods\_id)  group by user\_id,  b.third\_cart ; |

|  |
| --- |
| --整合  drop table if exists gdm.itcast\_gdm\_user\_category\_total;  create table gdm.itcast\_gdm\_user\_category\_total as  select  a.user\_id,  b.first\_category\_id,  b.first\_category\_name,  b.second\_category\_id,  b.second\_catery\_name,  a.third\_category\_id,  b.third\_category\_name  from  (select  user\_id,  third\_category\_id  from  gdm.itcast\_gdm\_user\_buy\_category\_temp  union  all  select  user\_id,  third\_cart  from  gdm.itcast\_gdm\_user\_cart\_category\_temp) a  left join gdm.gdm\_category\_code b  on (  a.third\_category\_id = b.third\_category\_id  )  group by a.user\_id,  b.first\_category\_id,  b.first\_category\_name,  b.second\_category\_id,  b.second\_catery\_name,  a.third\_category\_id,  b.third\_category\_name ; |

|  |
| --- |
| --生成最终客户购买类目表  INSERT overwrite TABLE gdm.itcast\_gdm\_user\_buy\_category PARTITION (dt = '2017-01-01')  SELECT  t.user\_id,  t.first\_category\_id,  t.first\_category\_name,  t.second\_category\_id,  t.second\_catery\_name,  t.third\_category\_id,  t.third\_category\_name,  t1.month1\_category\_cnt,  t1.month1\_category\_amt,  t1.month3\_category\_cnt,  t1.month3\_category\_amt,  t1.month6\_category\_cnt,  t1.month6\_category\_amt,  t1.total\_category\_cnt,  t1.total\_category\_amt,  t2.month1\_category\_cnt,  t2.month3\_category\_cnt,  t2.month6\_category\_cnt,  t2.total\_category\_cnt,  t1.last\_category\_time,  t1.last\_category\_ago,  FROM\_UNIXTIME(UNIX\_TIMESTAMP()) dw\_date  FROM  gdm.itcast\_gdm\_user\_category\_total t  LEFT JOIN gdm.itcast\_gdm\_user\_buy\_category\_temp t1  ON (  t.user\_id = t1.user\_id  AND t.third\_category\_id = t1.third\_category\_id  )  LEFT JOIN gdm.itcast\_gdm\_user\_cart\_category\_temp t2  ON (t.user\_id = t2.user\_id  AND t.third\_category\_id = t2.third\_cart) ; |

## 8.6 用户画像开发--客户访问信息表

|  |
| --- |
| --用户画像-客户访问信息表  drop table if exists gdm.itcast\_gdm\_user\_visit;  create external table gdm.itcast\_gdm\_user\_visit(  user\_id string ,--客户ID  latest\_pc\_visit\_date string ,--最近一次PC端访问日期  latest\_app\_visit\_date string ,--最近一次APP端访问日期  latest\_pc\_visit\_session string ,--最近一次PC端访问的session  latest\_pc\_cookies string ,--最近一次PC端访问的cookies  latest\_pc\_pv string ,--最近一次PC端访问的PV  latest\_pc\_browser\_name string ,--最近一次PC端访问使用的游览器  latest\_pc\_visit\_os string ,--最近一次PC端访问使用的操作系统  latest\_app\_name string ,--最近一次APP端访问app名称  latest\_app\_visit\_os string ,--最近一次APP端访问使用的操作系统  latest\_visit\_ip string ,--最近一次访问IP(不分APP与PC)  latest\_city string ,--最近一次访问城市(不分APP与PC)  latest\_province string ,--最近一次访问省份(不分APP与PC)  first\_pc\_visit\_date string ,--第一次PC端访问日期  first\_app\_visit\_date string ,--第一次APP端访问日期  first\_pc\_visit\_session string ,--第一次PC端访问的session  first\_pc\_cookies string ,--第一次PC端访问的cookies  first\_pc\_pv string ,--第一次PC端访问的PV  first\_pc\_browser\_name string ,--第一次PC端访问使用的游览器  first\_pc\_visit\_os string ,--第一次PC端访问使用的操作系统  first\_app\_name string ,--第一次APP端访问app名称  first\_app\_visit\_os string ,--第一次APP端访问使用的操作系统  first\_visit\_ip string ,--第一次访问IP(不分APP与PC)  first\_city string ,--第一次访问城市(不分APP与PC)  first\_province string ,--第一次访问省份(不分APP与PC)  day7\_app\_cnt bigint ,--近7天APP端访问次数  day15\_app\_cnt bigint ,--近15天APP端访问次数  month1\_app\_cnt bigint ,--近30天APP端访问次数  month2\_app\_cnt bigint ,--近60天APP端访问次数  month3\_app\_cnt bigint ,--近90天APP端访问次数    day7\_pc\_cnt bigint ,--近7天PC端访问次数  day15\_pc\_cnt bigint ,--近15天PC端访问次数  month1\_pc\_cnt bigint ,--近30天PC端访问次数  month2\_pc\_cnt bigint ,--近60天PC端访问次数  month3\_pc\_cnt bigint ,--近90天PC端访问次数    month1\_pc\_days bigint ,--近30天PC端访问天数  month1\_pc\_buy\_cnt bigint ,--近30天PC端访问购买次数  month1\_pc\_pv bigint ,--近30天PC端访问PV  month1\_pc\_avg\_pv bigint ,--近30天PC端访问平均PV  month1\_pc\_diff\_ip\_cnt bigint ,--近30天PC端访问不同ip数  month1\_pc\_common\_ip bigint ,--近30天PC端访问最常用ip  month1\_pc\_diff\_cookie\_cnt bigint ,--近30天PC端访问不同的cookie数  month1\_pc\_common\_cookie bigint ,--近30天PC端访问最常用的cookie  month1\_pc\_common\_browser\_name bigint ,--近30天PC端访问最常用游览器  month1\_pc\_common\_os bigint ,--近30天PC端访问最常用的操作系统  month1\_hour025\_cnt bigint ,--近30天0-5点访问次数(不分PC与APP)  month1\_hour627\_cnt bigint ,--近30天6-7点访问次数(不分PC与APP)  month1\_hour829\_cnt bigint ,--近30天8-9点访问次数(不分PC与APP)  month1\_hour10212\_cnt bigint ,--近30天10-12点访问次数(不分PC与APP)  month1\_hour13214\_cnt bigint ,--近30天13-14点访问次数(不分PC与APP)  month1\_hour15217\_cnt bigint ,--近30天15-17点访问次数(不分PC与APP)  month1\_hour18219\_cnt bigint ,--近30天18-19点访问次数(不分PC与APP)  month1\_hour20221\_cnt bigint ,--近30天20-21点访问次数(不分PC与APP)  month1\_hour22223\_cnt bigint ,--近30天22-23点访问次数(不分PC与APP)  dw\_date timestamp  ) partitioned by (dt string); |

|  |
| --- |
| ------用户上网轨迹表BDM层--PC端  DROP TABLE IF EXISTS bdm.itcast\_bdm\_user\_pc\_click\_log ;  CREATE external TABLE bdm.itcast\_bdm\_user\_pc\_click\_log (  session\_id STRING, --sessionID  cookie\_id STRING, --cookieID  visit\_time STRING, --访问时间  user\_id STRING, --用户ID  visit\_url STRING, --访问的URL  visit\_os STRING, --操作系统  browser\_name STRING,--游览器名称  visit\_ip STRING, --访问ip  province STRING, --省份  city STRING, --城市  page\_id STRING, --页面ID  goods\_id STRING, --商品ID  shop\_id STRING --商店ID  ) partitioned by (dt string)  row format delimited fields terminated by ','  lines terminated by '\n';  alter table bdm.itcast\_bdm\_user\_pc\_click\_log add partition (dt='2017-01-01') location '/business/itcast\_bdm\_user\_pc\_click\_log/2017-01-01';  hdfs dfs -put itcast\_bdm\_user\_pc\_click\_log.txt /business/itcast\_bdm\_user\_pc\_click\_log/2017-01-01 |

|  |
| --- |
| ------用户上网轨迹表FDM层--PC端  DROP TABLE IF EXISTS fdm.itcast\_fdm\_user\_pc\_pageview ;  CREATE external TABLE fdm.itcast\_fdm\_user\_pc\_pageview (  session\_id STRING, --sessionID  cookie\_id STRING, --cookieID  user\_id STRING, --用户ID  in\_time STRING, --访问进入时间  out\_time STRING, --访问离开时间  stay\_time STRING, --访问停留时间  pv BIGINT, --PV  visit\_os STRING, --操作系统  browser\_name STRING,--游览器名称  visit\_ip STRING, --访问ip  province STRING, --省份  city STRING --城市  ) partitioned BY (dt STRING);  -------------加载数据---  INSERT overwrite TABLE fdm.itcast\_fdm\_user\_pc\_pageview PARTITION (dt = '2017-01-01')  SELECT  t.session\_id,  t.cookie\_id,  t.user\_id,  MIN(visit\_time) in\_time,  MAX(visit\_time) out\_time,  (  case  WHEN MIN(visit\_time) = MAX(visit\_time)  then 5  else unix\_timestamp(MAX(visit\_time)) - unix\_timestamp(MIN(visit\_time))  end  ) stay\_time,  COUNT(1) pv,  t.visit\_os,  t.browser\_name,  t.visit\_ip,  t.province,  t.city  FROM  bdm.itcast\_bdm\_user\_pc\_click\_log t  WHERE dt = '2017-01-01'  GROUP BY t.session\_id,  t.cookie\_id,  t.user\_id,  t.visit\_os,  t.browser\_name,  t.visit\_ip,  t.province,  t.city ; |

|  |
| --- |
| ------用户上网轨迹表BDM层--APP端  drop table if exists bdm.itcast\_bdm\_user\_app\_click\_log;  create external table bdm.itcast\_bdm\_user\_app\_click\_log(  user\_id string, --用户ID  log\_time string, --访问时间  phone\_id string, --手机ID，唯一标识一台设备  visit\_os string, --操作系统 android、ios、wp  os\_version string, --操作系统版本  app\_name string, --APP的名称  app\_version string, --APP的版本  device\_token string, --PUSH码，消息推送的  visit\_ip string, --访问ip  province string, --省份  city string --城市  ) partitioned by (dt string)  row format delimited fields terminated by ','  lines terminated by '\n';  alter table bdm.itcast\_bdm\_user\_app\_click\_log add partition (dt='2017-01-01') location '/business/itcast\_bdm\_user\_app\_click\_log/2017-01-01';  hdfs dfs -put itcast\_bdm\_user\_app\_click\_log.txt /business/itcast\_bdm\_user\_app\_click\_log/2017-01-01  --Push指运营人员通过自己的产品或第三方工具对用户移动设备进行的主动消息推送。用户可以在移动设备锁定屏幕和通知栏看到push消息通知，通知栏点击可唤起APP并去往相应页面。我们平时在锁屏上看到的微信消息等等都属于APP消息推送行列。 |

|  |
| --- |
| ------用户上网轨迹表FDM层--app端  DROP TABLE IF EXISTS fdm.itcast\_fdm\_user\_app\_pageview ;  CREATE external TABLE fdm.itcast\_fdm\_user\_app\_pageview (  user\_id string, --用户ID  log\_time string, --访问时间  log\_hour string, --访问时间的小时数  phone\_id string, --手机ID，唯一标识一台设备  visit\_os string, --操作系统 android、ios、wp  os\_version string, --操作系统版本  app\_name string, --APP的名称  app\_version string, --APP的版本  device\_token string, --PUSH码，消息推送的  visit\_ip string, --访问ip  province string, --省份  city string --城市  ) partitioned BY (dt STRING);  -------------加载数据---  INSERT overwrite TABLE fdm.itcast\_fdm\_user\_app\_pageview PARTITION (dt = '2017-01-01')  SELECT  t.user\_id,  t.log\_time,  HOUR(t.log\_time) log\_hour,  t.phone\_id,  t.visit\_os,  t.os\_version,  t.app\_name,  t.app\_version,  t.device\_token,  t.visit\_ip,  t.province,  t.city  FROM  bdm.itcast\_bdm\_user\_app\_click\_log t  WHERE dt = '2017-01-01' ; |

|  |
| --- |
| --生成临时表-统计出用户近30天访问的ip/cookie/游览器信息  drop table if exists gdm.itcast\_gdm\_user\_visit\_temp\_01;  create table gdm.itcast\_gdm\_user\_visit\_temp\_01 as  select  t.user\_id,  t.type,  t.con,  t.cnt,  t.rn  from  (select  b.user\_id,  b.con,  b.type,  b.cnt,  row\_number () over (  distribute by b.user\_id,  b.type sort by b.cnt desc  ) rn  from  (select  a.user\_id,  a.visit\_ip con,  'visit\_ip' type,  count(1) cnt  from  fdm.itcast\_fdm\_user\_pc\_pageview a  where dt >= date\_add('2017-01-01', -29)  group by a.user\_id,  a.visit\_ip  union  all  select  a.user\_id,  a.cookie\_id con,  'cookie\_id' type,  count(1) cnt  from  fdm.itcast\_fdm\_user\_pc\_pageview a  where dt >= date\_add('2017-01-01', -29)  group by a.user\_id,  a.cookie\_id  union  all  select  a.user\_id,  a.browser\_name con,  'browser\_name' type,  count(1) cnt  from  fdm.itcast\_fdm\_user\_pc\_pageview a  where dt >= date\_add('2017-01-01', -29)  group by a.user\_id,  a.browser\_name  union  all  select  a.user\_id,  a.visit\_os con,  'visit\_os' type,  count(1) cnt  from  fdm.itcast\_fdm\_user\_pc\_pageview a  where dt >= date\_add('2017-01-01', -29)  group by a.user\_id,  a.visit\_os) b) t ; |

|  |
| --- |
| -------生成客户访问模型表  -------------加载数据---  INSERT overwrite TABLE fdm.itcast\_fdm\_user\_app\_pageview PARTITION (dt = '2017-01-01')  SELECT  t.user\_id,  t.log\_time,  HOUR(t.log\_time) log\_hour,  t.phone\_id,  t.visit\_os,  t.os\_version,  t.app\_name,  t.app\_version,  t.device\_token,  t.visit\_ip,  t.province,  t.city  FROM  bdm.itcast\_bdm\_user\_app\_click\_log t  WHERE dt = '2017-01-01' ;  ------近30天PC端访问最常用的指标  drop table if exists gdm.itcast\_gdm\_user\_visit\_temp\_01;  create table gdm.itcast\_gdm\_user\_visit\_temp\_01 as  select  t.user\_id,  t.type,  t.con,  t.cnt,  t.rn  from  (select  b.user\_id,  b.con,  b.type,  b.cnt,  row\_number () over (  distribute by b.user\_id,  b.type sort by b.cnt desc  ) rn  from  (select  a.user\_id,  a.visit\_ip con,  'visit\_ip' type,  count(1) cnt  from  fdm.itcast\_fdm\_user\_pc\_pageview a  where dt >= date\_add('2017-01-01', -29)  group by a.user\_id,  a.visit\_ip  union  all  select  a.user\_id,  a.cookie\_id con,  'cookie\_id' type,  count(1) cnt  from  fdm.itcast\_fdm\_user\_pc\_pageview a  where dt >= date\_add('2017-01-01', -29)  group by a.user\_id,  a.cookie\_id  union  all  select  a.user\_id,  a.browser\_name con,  'browser\_name' type,  count(1) cnt  from  fdm.itcast\_fdm\_user\_pc\_pageview a  where dt >= date\_add('2017-01-01', -29)  group by a.user\_id,  a.browser\_name  union  all  select  a.user\_id,  a.visit\_os con,  'visit\_os' type,  count(1) cnt  from  fdm.itcast\_fdm\_user\_pc\_pageview a  where dt >= date\_add('2017-01-01', -29)  group by a.user\_id,  a.visit\_os) b) t ;  -------生成客户访问模型表  INSERT overwrite TABLE gdm.itcast\_gdm\_user\_visit PARTITION (dt = '2017-01-01')  SELECT  t.user\_id,--客户ID  MAX(  CASE  WHEN pc.rn\_desc = 1  THEN pc.in\_time  END  ) latest\_pc\_visit\_date,--最近一次PC端访问日期  MAX(  CASE  WHEN app.rn\_desc = 1  THEN app.log\_time  END  ) latest\_app\_visit\_date,--最近一次APP端访问日期  MAX(  CASE  WHEN pc.rn\_desc = 1  THEN pc.session\_id  END  ) latest\_pc\_visit\_session,--最近一次PC端访问的session  MAX(  CASE  WHEN pc.rn\_desc = 1  THEN pc.cookie\_id  END  ) latest\_pc\_cookies,--最近一次PC端访问的cookies  MAX(  CASE  WHEN pc.rn\_desc = 1  THEN pc.pv  END) latest\_pc\_pv,--最近一次PC端访问的PV  MAX(  CASE  WHEN pc.rn\_desc = 1  THEN pc.browser\_name  END  ) latest\_pc\_browser\_name,--最近一次PC端访问使用的游览器  MAX(  CASE  WHEN pc.rn\_desc = 1  THEN pc.visit\_os  END  ) latest\_pc\_visit\_os,--最近一次PC端访问使用的操作系统  MAX(  CASE  WHEN app.rn\_desc = 1  THEN app.app\_name  END  ) latest\_app\_name,--最近一次APP端访问app名称  MAX(  CASE  WHEN app.rn\_desc = 1  THEN app.visit\_os  END  ) latest\_app\_visit\_os,--最近一次APP端访问使用的操作系统  MAX(  CASE  WHEN pc.rn\_desc = 1  AND app.rn\_desc = 1  AND pc.in\_time >= app.log\_time  THEN pc.visit\_ip  WHEN pc.rn\_desc = 1  AND app.rn\_desc = 1  AND pc.in\_time < app.log\_time  THEN app.visit\_ip  END  ) latest\_visit\_ip,--最近一次访问IP(不分APP与PC)  MAX(  CASE  WHEN pc.rn\_desc = 1  AND app.rn\_desc = 1  AND pc.in\_time >= app.log\_time  THEN pc.city  WHEN pc.rn\_desc = 1  AND app.rn\_desc = 1  AND pc.in\_time < app.log\_time  THEN app.city  END  ) latest\_city,--最近一次访问城市(不分APP与PC)  MAX(  CASE  WHEN pc.rn\_desc = 1  AND app.rn\_desc = 1  AND pc.in\_time >= app.log\_time  THEN pc.province  WHEN pc.rn\_desc = 1  AND app.rn\_desc = 1  AND pc.in\_time < app.log\_time  THEN app.province  END  ) latest\_province,--最近一次访问省份(不分APP与PC)  MAX(  CASE  WHEN pc.rn\_asc = 1  THEN pc.in\_time  END  ) first\_pc\_visit\_date,--第一次PC端访问日期  MAX(  CASE  WHEN app.rn\_asc = 1  THEN app.log\_time  END  ) first\_app\_visit\_date,--第一次APP端访问日期  MAX(  CASE  WHEN pc.rn\_asc = 1  THEN pc.session\_id  END  ) first\_pc\_visit\_session,--第一次PC端访问的session  MAX(  CASE  WHEN pc.rn\_asc = 1  THEN pc.cookie\_id  END  ) first\_pc\_cookies,--第一次PC端访问的cookies  MAX(  CASE  WHEN pc.rn\_asc = 1  THEN pc.pv  END) first\_pc\_pv,--第一次PC端访问的PV  MAX(  CASE  WHEN pc.rn\_asc = 1  THEN pc.browser\_name  END  ) first\_pc\_browser\_name,--第一次PC端访问使用的游览器  MAX(  CASE  WHEN pc.rn\_asc = 1  THEN pc.visit\_os  END  ) first\_pc\_visit\_os,--第一次PC端访问使用的操作系统  MAX(  CASE  WHEN app.rn\_asc = 1  THEN app.app\_name  END  ) first\_app\_name,--第一次APP端访问app名称  MAX(  CASE  WHEN app.rn\_asc = 1  THEN app.visit\_os  END  ) first\_app\_visit\_os,--第一次APP端访问使用的操作系统  MAX(  CASE  WHEN pc.rn\_asc = 1  AND app.rn\_asc = 1  AND pc.in\_time <= app.log\_time  THEN pc.visit\_ip  WHEN pc.rn\_asc = 1  AND app.rn\_asc = 1  AND pc.in\_time > app.log\_time  THEN app.visit\_ip  END  ) first\_visit\_ip,--第一次访问IP(不分APP与PC)  MAX(  CASE  WHEN pc.rn\_asc = 1  AND app.rn\_asc = 1  AND pc.in\_time <= app.log\_time  THEN pc.city  WHEN pc.rn\_asc = 1  AND app.rn\_asc = 1  AND pc.in\_time > app.log\_time  THEN app.city  END  ) first\_city,--第一次访问城市(不分APP与PC)  MAX(  CASE  WHEN pc.rn\_asc = 1  AND app.rn\_asc = 1  AND pc.in\_time <= app.log\_time  THEN pc.province  WHEN pc.rn\_asc = 1  AND app.rn\_asc = 1  AND pc.in\_time > app.log\_time  THEN app.province  END  ) first\_province,--第一次访问省份(不分APP与PC)  SUM(  CASE  WHEN app.dat\_7 = 1  THEN 1  END) day7\_app\_cnt,--近7天APP端访问次数  SUM(  CASE  WHEN app.dat\_15 = 1  THEN 1  END) day15\_app\_cnt,--近15天APP端访问次数  SUM(  CASE  WHEN app.dat\_30 = 1  THEN 1  END) month1\_app\_cnt,--近30天APP端访问次数  SUM(  CASE  WHEN app.dat\_60 = 1  THEN 1  END) month2\_app\_cnt,--近60天APP端访问次数  SUM(  CASE  WHEN app.dat\_90 = 1  THEN 1  END) month3\_app\_cnt,--近90天APP端访问次数  COUNT(  CASE  WHEN pc.dat\_7 = 1  THEN pc.session\_id  END  ) day7\_pc\_cnt,--近7天PC端访问次数  COUNT(  CASE  WHEN pc.dat\_15 = 1  THEN pc.session\_id  END  ) day15\_pc\_cnt,--近15天PC端访问次数  COUNT(  CASE  WHEN pc.dat\_30 = 1  THEN pc.session\_id  END  ) month1\_pc\_cnt,--近30天PC端访问次数  COUNT(  CASE  WHEN pc.dat\_60 = 1  THEN pc.session\_id  END  ) month2\_pc\_cnt,--近60天PC端访问次数  COUNT(  CASE  WHEN pc.dat\_90 = 1  THEN pc.session\_id  END  ) month3\_pc\_cnt,--近90天PC端访问次数  COUNT(DISTINCT substr(pc.in\_time,0,10)) month1\_pc\_days,--近30天PC端访问天数  SUM(  CASE  WHEN pc.dat\_30 = 1  THEN pc.pv  END) month1\_pc\_pv,--近30天PC端访问PV  SUM(  CASE  WHEN pc.dat\_30 = 1  THEN pc.pv  END) / COUNT(DISTINCT substr(pc.in\_time,0,10)) month1\_pc\_avg\_pv,--近30天PC端访问平均PV  MAX(b.month1\_pc\_diff\_ip\_cnt),--近30天PC端访问不同ip数  MAX(b.month1\_pc\_diff\_cookie\_cnt),--近30天PC端访问不同的cookie数  MAX(b.month1\_pc\_common\_ip),--近30天PC端访问最常用ip  MAX(b.month1\_pc\_common\_cookie),--近30天PC端访问最常用的cookie  MAX(b.month1\_pc\_common\_browser\_name),--近30天PC端访问最常用游览器  MAX(b.month1\_pc\_common\_os),--近30天PC端访问最常用的操作系统  COUNT(  CASE  WHEN pc.visit\_hour >= 0  AND pc.visit\_hour <= 5  THEN pc.session\_id  END  ) month1\_hour025\_cnt,--近30天PC端0-5点访问次数  COUNT(  CASE  WHEN pc.visit\_hour >= 6  AND pc.visit\_hour <= 7  THEN pc.session\_id  END  ) month1\_hour627\_cnt,--近30天PC端6-7点访问次数  COUNT(  CASE  WHEN pc.visit\_hour >= 8  AND pc.visit\_hour <= 9  THEN pc.session\_id  END  ) month1\_hour829\_cnt,--近30天PC端8-9点访问次数  COUNT(  CASE  WHEN pc.visit\_hour >= 10  AND pc.visit\_hour <= 12  THEN pc.session\_id  END  ) month1\_hour10212\_cnt,--近30天PC端10-12点访问次数  COUNT(  CASE  WHEN pc.visit\_hour >= 13  AND pc.visit\_hour <= 14  THEN pc.session\_id  END  ) month1\_hour13214\_cnt,--近30天PC端13-14点访问次数  COUNT(  CASE  WHEN pc.visit\_hour >= 15  AND pc.visit\_hour <= 17  THEN pc.session\_id  END  ) month1\_hour15217\_cnt,--近30天PC端15-17点访问次数  COUNT(  CASE  WHEN pc.visit\_hour >= 18  AND pc.visit\_hour <= 19  THEN pc.session\_id  END  ) month1\_hour18219\_cnt,--近30天PC端18-19点访问次数  COUNT(  CASE  WHEN pc.visit\_hour >= 20  AND pc.visit\_hour <= 21  THEN pc.session\_id  END  ) month1\_hour20221\_cnt,--近30天PC端20-21点访问次数  COUNT(  CASE  WHEN pc.visit\_hour >= 22  AND pc.visit\_hour <= 23  THEN pc.session\_id  END  ) month1\_hour22223\_cnt,--近30天PC端22-23点访问次数  FROM\_UNIXTIME(UNIX\_TIMESTAMP()) dw\_date  FROM  (SELECT  user\_id  FROM  fdm.itcast\_fdm\_user\_wide  WHERE dt = '2017-01-01') t  LEFT JOIN  (SELECT  a.\*,  (  CASE  WHEN in\_time >= DATE\_SUB('2017-01-01', 6)  AND in\_time <= '2017-01-01'  THEN 1  END  ) dat\_7,  (  CASE  WHEN in\_time >= DATE\_SUB('2017-01-01', 14)  AND in\_time <= '2017-01-01'  THEN 1  END  ) dat\_15,  (  CASE  WHEN in\_time >= DATE\_SUB('2017-01-01', 29)  AND in\_time <= '2017-01-01'  THEN 1  END  ) dat\_30,  (  CASE  WHEN in\_time >= DATE\_SUB('2017-01-01', 59)  AND in\_time <= '2017-01-01'  THEN 1  END  ) dat\_60,  (  CASE  WHEN in\_time >= DATE\_SUB('2017-01-01', 89)  AND in\_time <= '2017-01-01'  THEN 1  END  ) dat\_90,  hour(in\_time) visit\_hour,  row\_number () over (  distribute BY a.user\_id sort BY a.in\_time DESC  ) rn\_desc,  row\_number () over (  distribute BY a.user\_id sort BY a.in\_time ASC  ) rn\_asc  FROM  fdm.itcast\_fdm\_user\_pc\_pageview a  WHERE in\_time >= DATE\_ADD('2017-01-01', -89)  AND in\_time <= '2017-01-01') pc  ON (t.user\_id = pc.user\_id)  LEFT JOIN  (SELECT  user\_id,  sum(  CASE  WHEN TYPE = 'visit\_ip'  THEN cnt  END  ) month1\_pc\_diff\_ip\_cnt,  MAX(  CASE  WHEN TYPE = 'visit\_ip' and rn= 1  THEN con  END  ) month1\_pc\_common\_ip,  sum(  CASE  WHEN TYPE = 'cookie\_id'  THEN cnt  END  ) month1\_pc\_diff\_cookie\_cnt,  MAX(  CASE  WHEN TYPE = 'cookie\_id' and rn = 1  THEN con  END  ) month1\_pc\_common\_cookie,  MAX(  CASE  WHEN TYPE = 'browser\_name' and rn = 1  THEN con  END  ) month1\_pc\_common\_browser\_name,  MAX(  CASE  WHEN TYPE = 'visit\_os' and rn = 1  THEN con  END  ) month1\_pc\_common\_os  FROM  gdm.itcast\_gdm\_user\_visit\_temp\_01  GROUP BY user\_id) b  ON (t.user\_id = b.user\_id)  LEFT JOIN  (SELECT  a.\*,  (  CASE  WHEN log\_time >= DATE\_SUB('2017-01-01', 6)  AND log\_time <= '2017-01-01'  THEN 1  END  ) dat\_7,  (  CASE  WHEN log\_time >= DATE\_SUB('2017-01-01', 14)  AND log\_time <= '2017-01-01'  THEN 1  END  ) dat\_15,  (  CASE  WHEN log\_time >= DATE\_SUB('2017-01-01', 29)  AND log\_time <= '2017-01-01'  THEN 1  END  ) dat\_30,  (  CASE  WHEN log\_time >= DATE\_SUB('2017-01-01', 59)  AND log\_time <= '2017-01-01'  THEN 1  END  ) dat\_60,  (  CASE  WHEN log\_time >= DATE\_SUB('2017-01-01', 89)  AND log\_time <= '2017-01-01'  THEN 1  END  ) dat\_90,  row\_number () over (  distribute BY a.user\_id sort BY a.log\_time DESC  ) rn\_desc,  row\_number () over (  distribute BY a.user\_id sort BY a.log\_time ASC  ) rn\_asc  FROM  fdm.itcast\_fdm\_user\_app\_pageview a  WHERE log\_time >= DATE\_ADD('2017-01-01', -89)  AND log\_time <= '2017-01-01') app  ON (t.user\_id = app.user\_id)  GROUP BY t.user\_id ; |

## 8.7 用户画像开发--用户宽表汇总模型表

|  |
| --- |
| --用户画像宽表模型  create database if not exists adm;  create table if not exists adm.itcast\_adm\_personas(  user\_id string ,--用户ID  user\_name string ,--用户登陆名  user\_sex string ,--用户性别  user\_birthday string ,--用户生日  user\_age bigint ,--用户年龄  constellation string ,--用户星座  province string ,--省份  city string ,--城市  city\_level string ,--城市等级  hex\_mail string ,--邮箱  op\_mail string ,--邮箱运营商  hex\_phone string ,--手机号  fore\_phone string ,--手机前3位  op\_phone string ,--手机运营商  add\_time timestamp ,--注册时间  login\_ip string ,--登陆ip地址  login\_source string ,--登陆来源  request\_user string ,--邀请人  total\_mark bigint ,--会员积分  used\_mark bigint ,--已使用积分  level\_name string ,--会员等级名称  blacklist bigint ,--用户黑名单  is\_married bigint ,--婚姻状况  education string ,--学历  monthly\_money double ,--收入  profession string ,--职业  sex\_model bigint ,--性别模型  is\_pregnant\_woman bigint ,--是否孕妇  is\_have\_children bigint ,--是否有小孩  children\_sex\_rate double , --孩子性别概率  children\_age\_rate double ,--孩子年龄概率  is\_have\_car bigint ,--是否有车  potential\_car\_user\_rate double ,--潜在汽车用户概率  phone\_brand string ,--使用手机品牌  phone\_brand\_level string ,--使用手机品牌档次  phone\_cnt bigint ,--使用多少种不同的手机  change\_phone\_rate bigint ,--更换手机频率  majia\_flag string ,--马甲标志  majie\_account\_cnt bigint ,--马甲账号数量  loyal\_model bigint ,--用户忠诚度  shopping\_type\_model bigint ,--用户购物类型  figure\_model bigint ,--身材  stature\_model bigint ,--身高  first\_order\_time timestamp, --第一次消费时间  last\_order\_time timestamp, --最近一次消费时间  first\_order\_ago bigint, --首单距今时间  last\_order\_ago bigint, --尾单距今时间  month1\_hg\_order\_cnt bigint, --近30天购买次数（不含退拒）  month1\_hg\_order\_amt double, --近30天购买金额（不含退拒）  month2\_hg\_order\_cnt bigint, --近60天购买次数（不含退拒）  month2\_hg\_order\_amt double, --近60天购买金额（不含退拒）  month3\_hg\_order\_cnt bigint, --近90天购买次数（不含退拒）  month3\_hg\_order\_amt double, --近90天购买金额（不含退拒）  month1\_order\_cnt bigint, --近30天购买次数（含退拒）  month1\_order\_amt double, --近30天购买金额（含退拒）  month2\_order\_cnt bigint, --近60天购买次数（含退拒）  month2\_order\_amt double, --近60天购买金额（含退拒）  month3\_order\_cnt bigint, --近90天购买次数（含退拒）  month3\_order\_amt double, --近90天购买金额（含退拒）  max\_order\_amt double, --最大消费金额  min\_order\_amt double, --最小消费金额  total\_order\_cnt bigint, --累计消费次数（不含退拒）  total\_order\_amt double, --累计消费金额（不含退拒）  user\_avg\_amt double, --客单价（含退拒）  month3\_user\_avg\_amt double, --近90天的客单价  common\_address string, --常用收货地址  common\_paytype string, --常用支付方式  month1\_cart\_cnt bigint, --近30天购物车的次数  month1\_cart\_goods\_cnt bigint, --近30天购物车商品件数  month1\_cart\_submit\_cnt bigint, --近30天购物车提交商品件数  month1\_cart\_rate double, --近30天购物车成功率  month1\_cart\_cancle\_cnt double, --近30天购物车放弃件数  return\_cnt bigint, --退货商品数量  return\_amt double, --退货商品金额  reject\_cnt bigint, --拒收商品数量  reject\_amt double, --拒收商品金额  last\_return\_time timestamp, --最近一次退货时间  school\_order\_cnt bigint, --学校下单总数  company\_order\_cnt bigint, --单位下单总数  home\_order\_cnt bigint, --家里下单总数  forenoon\_order\_cnt bigint, --上午下单总数  afternoon\_order\_cnt bigint, --下午下单总数  night\_order\_cnt bigint, --晚上下单总数  morning\_order\_cnt bigint, --凌晨下单总数  first\_category\_id BIGINT, --一级分类ID  first\_category\_name STRING, --一级分类名称  second\_category\_id BIGINT, --二分类ID  second\_catery\_name STRING, --二级分类名称  third\_category\_id BIGINT, --三级分类ID  third\_category\_name STRING, --三级分类名称  month1\_category\_cnt BIGINT, --近30天购物类目次数  month1\_category\_amt STRING, --近30天购物类目金额  month3\_category\_cnt BIGINT, --近90天购物类目次数  month3\_category\_amt STRING, --近90天购物类目金额  month6\_category\_cnt BIGINT, --近180天购物类目次数  month6\_category\_amt STRING, --近180天购物类目金额  total\_category\_cnt BIGINT, --累计购物类目次数  total\_category\_amt STRING, --累计购物类目次数  month1\_cart\_category\_cnt BIGINT, --近30天购物车类目次数  month3\_cart\_category\_cnt BIGINT, --近90天购物车类目次数  month6\_cart\_category\_cnt BIGINT, --近180天购物车类目次数  total\_cart\_category\_cnt BIGINT, --累计购物车类目次数  last\_category\_time TIMESTAMP, --最后一次购买类目时间  last\_category\_ago BIGINT, --最后一次购买类目距今天数  latest\_pc\_visit\_date string, --最近一次PC端访问日期  latest\_app\_visit\_date string, --最近一次APP端访问日期  latest\_pc\_visit\_session string, --最近一次PC端访问的session  latest\_pc\_cookies string, --最近一次PC端访问的cookies  latest\_pc\_pv string, --最近一次PC端访问的PV  latest\_pc\_browser\_name string, --最近一次PC端访问使用的游览器  latest\_pc\_visit\_os string, --最近一次PC端访问使用的操作系统  latest\_app\_name string, --最近一次APP端访问app名称  latest\_app\_visit\_os string, --最近一次APP端访问使用的操作系统  latest\_visit\_ip string, --最近一次访问IP(不分APP与PC)  latest\_city string, --最近一次访问城市(不分APP与PC)  latest\_province string, --最近一次访问省份(不分APP与PC)  first\_pc\_visit\_date string, --第一次PC端访问日期  first\_app\_visit\_date string, --第一次APP端访问日期  first\_pc\_visit\_session string, --第一次PC端访问的session  first\_pc\_cookies string, --第一次PC端访问的cookies  first\_pc\_pv string, --第一次PC端访问的PV  first\_pc\_browser\_name string, --第一次PC端访问使用的游览器  first\_pc\_visit\_os string, --第一次PC端访问使用的操作系统  first\_app\_name string, --第一次APP端访问app名称  first\_app\_visit\_os string, --第一次APP端访问使用的操作系统  first\_visit\_ip string, --第一次访问IP(不分APP与PC)  first\_city string, --第一次访问城市(不分APP与PC)  first\_province string, --第一次访问省份(不分APP与PC)  day7\_app\_cnt bigint, --近7天APP端访问次数  day15\_app\_cnt bigint, --近15天APP端访问次数  month1\_app\_cnt bigint, --近30天APP端访问次数  month2\_app\_cnt bigint, --近60天APP端访问次数  month3\_app\_cnt bigint, --近90天APP端访问次数  day7\_pc\_cnt bigint, --近7天PC端访问次数  day15\_pc\_cnt bigint, --近15天PC端访问次数  month1\_pc\_cnt bigint, --近30天PC端访问次数  month2\_pc\_cnt bigint, --近60天PC端访问次数  month3\_pc\_cnt bigint, --近90天PC端访问次数  month1\_pc\_days bigint, --近30天PC端访问天数  month1\_pc\_pv bigint, --近30天PC端访问PV  month1\_pc\_avg\_pv bigint, --近30天PC端访问平均PV  month1\_pc\_diff\_ip\_cnt bigint, --近30天PC端访问不同ip数  month1\_pc\_diff\_cookie\_cnt bigint, --近30天PC端访问不同的cookie数  month1\_pc\_common\_ip string, --近30天PC端访问最常用ip  month1\_pc\_common\_cookie string, --近30天PC端访问最常用的cookie  month1\_pc\_common\_browser\_name string, --近30天PC端访问最常用游览器  month1\_pc\_common\_os string, --近30天PC端访问最常用的操作系统  month1\_hour025\_cnt bigint, --近30天PC端0-5点访问次数  month1\_hour627\_cnt bigint, --近30天PC端6-7点访问次数  month1\_hour829\_cnt bigint, --近30天PC端8-9点访问次数  month1\_hour10212\_cnt bigint, --近30天PC端10-12点访问次数  month1\_hour13214\_cnt bigint, --近30天PC端13-14点访问次数  month1\_hour15217\_cnt bigint, --近30天PC端15-17点访问次数  month1\_hour18219\_cnt bigint, --近30天PC端18-19点访问次数  month1\_hour20221\_cnt bigint, --近30天PC端20-21点访问次数  month1\_hour22223\_cnt bigint --近30天PC端22-23点访问次数  );  -----加载数据  insert overwrite table adm.itcast\_adm\_personas  select  a.user\_id,  a.user\_name,  a.user\_sex,  a.user\_birthday,  a.user\_age,  a.constellation,  a.province,  a.city,  a.city\_level,  a.hex\_mail,  a.op\_mail,  a.hex\_phone,  a.fore\_phone,  a.op\_phone,  a.add\_time,  a.login\_ip,  a.login\_source,  a.request\_user,  a.total\_mark,  a.used\_mark,  a.level\_name,  a.blacklist,  a.is\_married,  a.education,  a.monthly\_money,  a.profession,  a.sex\_model,  a.is\_pregnant\_woman,  a.is\_have\_children,  a.children\_sex\_rate,  a.children\_age\_rate,  a.is\_have\_car,  a.potential\_car\_user\_rate,  a.phone\_brand,  a.phone\_brand\_level,  a.phone\_cnt,  a.change\_phone\_rate,  a.majia\_flag,  a.majie\_account\_cnt,  a.loyal\_model,  a.shopping\_type\_model,  a.figure\_model,  a.stature\_model,  b.first\_order\_time,  b.last\_order\_time,  b.first\_order\_ago,  b.last\_order\_ago,  b.month1\_hg\_order\_cnt,  b.month1\_hg\_order\_amt,  b.month2\_hg\_order\_cnt,  b.month2\_hg\_order\_amt,  b.month3\_hg\_order\_cnt,  b.month3\_hg\_order\_amt,  b.month1\_order\_cnt,  b.month1\_order\_amt,  b.month2\_order\_cnt,  b.month2\_order\_amt,  b.month3\_order\_cnt,  b.month3\_order\_amt,  b.max\_order\_amt,  b.min\_order\_amt,  b.total\_order\_cnt,  b.total\_order\_amt,  b.user\_avg\_amt,  b.month3\_user\_avg\_amt,  b.common\_address,  b.common\_paytype,  b.month1\_cart\_cnt,  b.month1\_cart\_goods\_cnt,  b.month1\_cart\_submit\_cnt,  b.month1\_cart\_rate,  b.month1\_cart\_cancle\_cnt,  b.return\_cnt,  b.return\_amt,  b.reject\_cnt,  b.reject\_amt,  b.last\_return\_time,  b.school\_order\_cnt,  b.company\_order\_cnt,  b.home\_order\_cnt,  b.forenoon\_order\_cnt,  b.afternoon\_order\_cnt,  b.night\_order\_cnt,  b.morning\_order\_cnt,  c.first\_category\_id,  c.first\_category\_name,  c.second\_category\_id,  c.second\_catery\_name,  c.third\_category\_id,  c.third\_category\_name,  c.month1\_category\_cnt,  c.month1\_category\_amt,  c.month3\_category\_cnt,  c.month3\_category\_amt,  c.month6\_category\_cnt,  c.month6\_category\_amt,  c.total\_category\_cnt,  c.total\_category\_amt,  c.month1\_category\_cnt,  c.month3\_category\_cnt,  c.month6\_category\_cnt,  c.total\_category\_cnt,  c.last\_category\_time,  c.last\_category\_ago,  d.latest\_pc\_visit\_date,  d.latest\_app\_visit\_date,  d.latest\_pc\_visit\_session,  d.latest\_pc\_cookies,  d.latest\_pc\_pv,  d.latest\_pc\_browser\_name,  d.latest\_pc\_visit\_os,  d.latest\_app\_name,  d.latest\_app\_visit\_os,  d.latest\_visit\_ip,  d.latest\_city,  d.latest\_province,  d.first\_pc\_visit\_date,  d.first\_app\_visit\_date,  d.first\_pc\_visit\_session,  d.first\_pc\_cookies,  d.first\_pc\_pv,  d.first\_pc\_browser\_name,  d.first\_pc\_visit\_os,  d.first\_app\_name,  d.first\_app\_visit\_os,  d.first\_visit\_ip,  d.first\_city,  d.first\_province,  d.day7\_app\_cnt,  d.day15\_app\_cnt,  d.month1\_app\_cnt,  d.month2\_app\_cnt,  d.month3\_app\_cnt,  d.day7\_pc\_cnt,  d.day15\_pc\_cnt,  d.month1\_pc\_cnt,  d.month2\_pc\_cnt,  d.month3\_pc\_cnt,  d.month1\_pc\_days,  d.month1\_pc\_pv,  d.month1\_pc\_avg\_pv,  d.month1\_pc\_diff\_ip\_cnt,  d.month1\_pc\_diff\_cookie\_cnt,  d.month1\_pc\_common\_ip,  d.month1\_pc\_common\_cookie,  d.month1\_pc\_common\_browser\_name,  d.month1\_pc\_common\_os,  d.month1\_hour025\_cnt,  d.month1\_hour627\_cnt,  d.month1\_hour829\_cnt,  d.month1\_hour10212\_cnt,  d.month1\_hour13214\_cnt,  d.month1\_hour15217\_cnt,  d.month1\_hour18219\_cnt,  d.month1\_hour20221\_cnt,  d.month1\_hour22223\_cnt  from gdm.itcast\_gdm\_user\_basic a  left join gdm.itcast\_gdm\_user\_consume\_order b on a.user\_id=b.user\_id  left join gdm.itcast\_gdm\_user\_buy\_category c on a.user\_id=c.user\_id  left join gdm.itcast\_gdm\_user\_visit d on a.user\_id=d.user\_id;  ----------建立hive/hbase关联表（利用hive/hbase之间互相映射的特性，将存在hive数据仓库中的数据导入到hbase中）  CREATE TABLE adm.itcast\_adm\_personas\_hbase (  user\_id STRING,  user\_name STRING,  user\_sex STRING,  user\_birthday STRING,  user\_age BIGINT,  constellation STRING,  province STRING,  city STRING,  city\_level STRING,  hex\_mail STRING,  op\_mail STRING,  hex\_phone STRING,  fore\_phone STRING,  op\_phone STRING,  add\_time TIMESTAMP,  login\_ip STRING,  login\_source STRING,  request\_user STRING,  total\_mark BIGINT,  used\_mark BIGINT,  level\_name STRING,  blacklist BIGINT,  is\_married BIGINT,  education STRING,  monthly\_money DOUBLE,  profession STRING,  sex\_model BIGINT,  is\_pregnant\_woman BIGINT,  is\_have\_children BIGINT,  children\_sex\_rate DOUBLE,  children\_age\_rate DOUBLE,  is\_have\_car BIGINT,  potential\_car\_user\_rate DOUBLE,  phone\_brand STRING,  phone\_brand\_level STRING,  phone\_cnt BIGINT,  change\_phone\_rate BIGINT,  majia\_flag STRING,  majie\_account\_cnt BIGINT,  loyal\_model BIGINT,  shopping\_type\_model BIGINT,  figure\_model BIGINT,  stature\_model BIGINT,  first\_order\_time TIMESTAMP,  last\_order\_time TIMESTAMP,  first\_order\_ago BIGINT,  last\_order\_ago BIGINT,  month1\_hg\_order\_cnt BIGINT,  month1\_hg\_order\_amt DOUBLE,  month2\_hg\_order\_cnt BIGINT,  month2\_hg\_order\_amt DOUBLE,  month3\_hg\_order\_cnt BIGINT,  month3\_hg\_order\_amt DOUBLE,  month1\_order\_cnt BIGINT,  month1\_order\_amt DOUBLE,  month2\_order\_cnt BIGINT,  month2\_order\_amt DOUBLE,  month3\_order\_cnt BIGINT,  month3\_order\_amt DOUBLE,  max\_order\_amt DOUBLE,  min\_order\_amt DOUBLE,  total\_order\_cnt BIGINT,  total\_order\_amt DOUBLE,  user\_avg\_amt DOUBLE,  month3\_user\_avg\_amt DOUBLE,  common\_address STRING,  common\_paytype STRING,  month1\_cart\_cnt BIGINT,  month1\_cart\_goods\_cnt BIGINT,  month1\_cart\_submit\_cnt BIGINT,  month1\_cart\_rate DOUBLE,  month1\_cart\_cancle\_cnt DOUBLE,  return\_cnt BIGINT,  return\_amt DOUBLE,  reject\_cnt BIGINT,  reject\_amt DOUBLE,  last\_return\_time TIMESTAMP,  school\_order\_cnt BIGINT,  company\_order\_cnt BIGINT,  home\_order\_cnt BIGINT,  forenoon\_order\_cnt BIGINT,  afternoon\_order\_cnt BIGINT,  night\_order\_cnt BIGINT,  morning\_order\_cnt BIGINT,  first\_category\_id BIGINT,  first\_category\_name STRING,  second\_category\_id BIGINT,  second\_catery\_name STRING,  third\_category\_id BIGINT,  third\_category\_name STRING,  month1\_category\_cnt BIGINT,  month1\_category\_amt STRING,  month3\_category\_cnt BIGINT,  month3\_category\_amt STRING,  month6\_category\_cnt BIGINT,  month6\_category\_amt STRING,  total\_category\_cnt BIGINT,  total\_category\_amt STRING,  month1\_cart\_category\_cnt BIGINT,  month3\_cart\_category\_cnt BIGINT,  month6\_cart\_category\_cnt BIGINT,  total\_cart\_category\_cnt BIGINT,  last\_category\_time TIMESTAMP,  last\_category\_ago BIGINT,  latest\_pc\_visit\_date STRING,  latest\_app\_visit\_date STRING,  latest\_pc\_visit\_session STRING,  latest\_pc\_cookies STRING,  latest\_pc\_pv STRING,  latest\_pc\_browser\_name STRING,  latest\_pc\_visit\_os STRING,  latest\_app\_name STRING,  latest\_app\_visit\_os STRING,  latest\_visit\_ip STRING,  latest\_city STRING,  latest\_province STRING,  first\_pc\_visit\_date STRING,  first\_app\_visit\_date STRING,  first\_pc\_visit\_session STRING,  first\_pc\_cookies STRING,  first\_pc\_pv STRING,  first\_pc\_browser\_name STRING,  first\_pc\_visit\_os STRING,  first\_app\_name STRING,  first\_app\_visit\_os STRING,  first\_visit\_ip STRING,  first\_city STRING,  first\_province STRING,  day7\_app\_cnt BIGINT,  day15\_app\_cnt BIGINT,  month1\_app\_cnt BIGINT,  month2\_app\_cnt BIGINT,  month3\_app\_cnt BIGINT,  day7\_pc\_cnt BIGINT,  day15\_pc\_cnt BIGINT,  month1\_pc\_cnt BIGINT,  month2\_pc\_cnt BIGINT,  month3\_pc\_cnt BIGINT,  month1\_pc\_days BIGINT,  month1\_pc\_pv BIGINT,  month1\_pc\_avg\_pv BIGINT,  month1\_pc\_diff\_ip\_cnt BIGINT,  month1\_pc\_diff\_cookie\_cnt BIGINT,  month1\_pc\_common\_ip string,  month1\_pc\_common\_cookie string,  month1\_pc\_common\_browser\_name string,  month1\_pc\_common\_os string,  month1\_hour025\_cnt BIGINT,  month1\_hour627\_cnt BIGINT,  month1\_hour829\_cnt BIGINT,  month1\_hour10212\_cnt BIGINT,  month1\_hour13214\_cnt BIGINT,  month1\_hour15217\_cnt BIGINT,  month1\_hour18219\_cnt BIGINT,  month1\_hour20221\_cnt BIGINT,  month1\_hour22223\_cnt BIGINT  ) STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH SERDEPROPERTIES (  "hbase.columns.mapping" = ":key,  basicInfo:user\_name,  basicInfo:user\_sex,  basicInfo:user\_birthday,  basicInfo:user\_age,  basicInfo:constellation,  basicInfo:province,  basicInfo:city,  basicInfo:city\_level,  basicInfo:hex\_mail,  basicInfo:op\_mail,  basicInfo:hex\_phone,  basicInfo:fore\_phone,  basicInfo:op\_phone,  basicInfo:add\_time,  basicInfo:login\_ip,  basicInfo:login\_source,  basicInfo:request\_user,  basicInfo:total\_mark,  basicInfo:used\_mark,  basicInfo:level\_name,  basicInfo:blacklist,  basicInfo:is\_married,  basicInfo:education,  basicInfo:monthly\_money,  basicInfo:profession,  basicInfo:sex\_model,  basicInfo:is\_pregnant\_woman,  basicInfo:is\_have\_children,  basicInfo:children\_sex\_rate,  basicInfo:children\_age\_rate,  basicInfo:is\_have\_car,  basicInfo:potential\_car\_user\_rate,  basicInfo:phone\_brand,  basicInfo:phone\_brand\_level,  basicInfo:phone\_cnt,  basicInfo:change\_phone\_rate,  basicInfo:majia\_flag,  basicInfo:majie\_account\_cnt,  basicInfo:loyal\_model,  basicInfo:shopping\_type\_model,  basicInfo:figure\_model,  basicInfo:stature\_model,  order:first\_order\_time,  order:last\_order\_time,  order:first\_order\_ago,  order:last\_order\_ago,  order:month1\_hg\_order\_cnt,  order:month1\_hg\_order\_amt,  order:month2\_hg\_order\_cnt,  order:month2\_hg\_order\_amt,  order:month3\_hg\_order\_cnt,  order:month3\_hg\_order\_amt,  order:month1\_order\_cnt,  order:month1\_order\_amt,  order:month2\_order\_cnt,  order:month2\_order\_amt,  order:month3\_order\_cnt,  order:month3\_order\_amt,  order:max\_order\_amt,  order:min\_order\_amt,  order:total\_order\_cnt,  order:total\_order\_amt,  order:user\_avg\_amt,  order:month3\_user\_avg\_amt,  order:common\_address,  order:common\_paytype,  order:month1\_cart\_cnt,  order:month1\_cart\_goods\_cnt,  order:month1\_cart\_submit\_cnt,  order:month1\_cart\_rate,  order:month1\_cart\_cancle\_cnt,  order:return\_cnt,  order:return\_amt,  order:reject\_cnt,  order:reject\_amt,  order:last\_return\_time,  order:school\_order\_cnt,  order:company\_order\_cnt,  order:home\_order\_cnt,  order:forenoon\_order\_cnt,  order:afternoon\_order\_cnt,  order:night\_order\_cnt,  order:morning\_order\_cnt,  category:first\_category\_id,  category:first\_category\_name,  category:second\_category\_id,  category:second\_catery\_name,  category:third\_category\_id,  category:third\_category\_name,  category:month1\_category\_cnt,  category:month1\_category\_amt,  category:month3\_category\_cnt,  category:month3\_category\_amt,  category:month6\_category\_cnt,  category:month6\_category\_amt,  category:total\_category\_cnt,  category:total\_category\_amt,  category:month1\_cart\_category\_cnt,  category:month3\_cart\_category\_cnt,  category:month6\_cart\_category\_cnt,  category:total\_cart\_category\_cnt,  category:last\_category\_time,  category:last\_category\_ago,  visit:latest\_pc\_visit\_date,  visit:latest\_app\_visit\_date,  visit:latest\_pc\_visit\_session,  visit:latest\_pc\_cookies,  visit:latest\_pc\_pv,  visit:latest\_pc\_browser\_name,  visit:latest\_pc\_visit\_os,  visit:latest\_app\_name,  visit:latest\_app\_visit\_os,  visit:latest\_visit\_ip,  visit:latest\_city,  visit:latest\_province,  visit:first\_pc\_visit\_date,  visit:first\_app\_visit\_date,  visit:first\_pc\_visit\_session,  visit:first\_pc\_cookies,  visit:first\_pc\_pv,  visit:first\_pc\_browser\_name,  visit:first\_pc\_visit\_os,  visit:first\_app\_name,  visit:first\_app\_visit\_os,  visit:first\_visit\_ip,  visit:first\_city,  visit:first\_province,  visit:day7\_app\_cnt,  visit:day15\_app\_cnt,  visit:month1\_app\_cnt,  visit:month2\_app\_cnt,  visit:month3\_app\_cnt,  visit:day7\_pc\_cnt,  visit:day15\_pc\_cnt,  visit:month1\_pc\_cnt,  visit:month2\_pc\_cnt,  visit:month3\_pc\_cnt,  visit:month1\_pc\_days,  visit:month1\_pc\_pv,  visit:month1\_pc\_avg\_pv,  visit:month1\_pc\_diff\_ip\_cnt,  visit:month1\_pc\_diff\_cookie\_cnt,  visit:month1\_pc\_common\_ip,  visit:month1\_pc\_common\_cookie,  visit:month1\_pc\_common\_browser\_name,  visit:month1\_pc\_common\_os,  visit:month1\_hour025\_cnt,  visit:month1\_hour627\_cnt,  visit:month1\_hour829\_cnt,  visit:month1\_hour10212\_cnt,  visit:month1\_hour13214\_cnt,  visit:month1\_hour15217\_cnt,  visit:month1\_hour18219\_cnt,  visit:month1\_hour20221\_cnt,  visit:month1\_hour22223\_cnt"  ) TBLPROPERTIES (  "hbase.table.name" = "itcast\_adm\_personas\_hbase\_2017\_01\_01",  "hbase.mapred.output.outputtable"="itcast\_adm\_personas\_hbase\_2017\_01\_01"  ) ;  ----向映射表中加载数据  insert overwrite table adm.itcast\_adm\_personas\_hbase select \* from adm.itcast\_adm\_personas;  ----加载成功后查看表中是否有数据存在  hive中的表adm.itcast\_adm\_personas\_hbase  hive映射到hbase中的表itcast\_adm\_personas\_hbase\_2017\_01\_01 |

# 电商用户画像数据可视化

## 9.1 数据可视化方案

技术框架：maven+spring+phoenix

核心点：采用phoenix与hbase整合，通过我们熟知的sql语句来操作NoSql数据库。

## 9.2 Phoenix建立hbase表的映射表

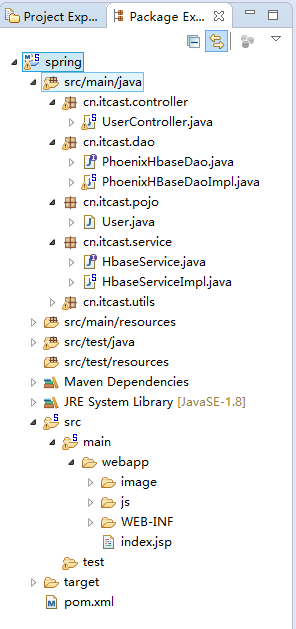
|  |
| --- |
| CREATE TABLE IF NOT EXISTS "itcast\_adm\_personas\_hbase\_2017\_01\_01"(  rw varchar(100) primary key,  "basicInfo"."user\_name" VARCHAR(100),  "basicInfo"."user\_sex" VARCHAR(100),  "basicInfo"."user\_birthday" VARCHAR(100),  "basicInfo"."user\_age" VARCHAR(100),  "basicInfo"."constellation" VARCHAR(100),  "basicInfo"."province" VARCHAR(100),  "basicInfo"."city" VARCHAR(100),  "basicInfo"."city\_level" VARCHAR(100),  "basicInfo"."hex\_mail" VARCHAR(100),  "basicInfo"."op\_mail" VARCHAR(100),  "basicInfo"."hex\_phone" VARCHAR(100),  "basicInfo"."fore\_phone" VARCHAR(100),  "basicInfo"."op\_phone" VARCHAR(100),  "basicInfo"."add\_time" VARCHAR(100),  "basicInfo"."login\_ip" VARCHAR(100),  "basicInfo"."login\_source" VARCHAR(100),  "basicInfo"."request\_user" VARCHAR(100),  "basicInfo"."total\_mark" VARCHAR(100),  "basicInfo"."used\_mark" VARCHAR(100),  "basicInfo"."level\_name" VARCHAR(100),  "basicInfo"."blacklist" VARCHAR(100),  "basicInfo"."is\_married" VARCHAR(100),  "basicInfo"."education" VARCHAR(100),  "basicInfo"."monthly\_money" VARCHAR(100),  "basicInfo"."profession" VARCHAR(100),  "basicInfo"."sex\_model" VARCHAR(100),  "basicInfo"."is\_pregnant\_woman" VARCHAR(100),  "basicInfo"."is\_have\_children" VARCHAR(100),  "basicInfo"."children\_sex\_rate" VARCHAR(100),  "basicInfo"."children\_age\_rate" VARCHAR(100),  "basicInfo"."is\_have\_car" VARCHAR(100),  "basicInfo"."potential\_car\_user\_rate" VARCHAR(100),  "basicInfo"."phone\_brand" VARCHAR(100),  "basicInfo"."phone\_brand\_level" VARCHAR(100),  "basicInfo"."phone\_cnt" VARCHAR(100),  "basicInfo"."change\_phone\_rate" VARCHAR(100),  "basicInfo"."majia\_flag" VARCHAR(100),  "basicInfo"."majie\_account\_cnt" VARCHAR(100),  "basicInfo"."loyal\_model" VARCHAR(100),  "basicInfo"."shopping\_type\_model" VARCHAR(100),  "basicInfo"."figure\_model" VARCHAR(100),  "basicInfo"."stature\_model" VARCHAR(100),  "order"."first\_order\_time" VARCHAR(100),  "order"."last\_order\_time" VARCHAR(100),  "order"."first\_order\_ago" VARCHAR(100),  "order"."last\_order\_ago" VARCHAR(100),  "order"."month1\_hg\_order\_cnt" VARCHAR(100),  "order"."month1\_hg\_order\_amt" VARCHAR(100),  "order"."month2\_hg\_order\_cnt" VARCHAR(100),  "order"."month2\_hg\_order\_amt" VARCHAR(100),  "order"."month3\_hg\_order\_cnt" VARCHAR(100),  "order"."month3\_hg\_order\_amt" VARCHAR(100),  "order"."month1\_order\_cnt" VARCHAR(100),  "order"."month1\_order\_amt" VARCHAR(100),  "order"."month2\_order\_cnt" VARCHAR(100),  "order"."month2\_order\_amt" VARCHAR(100),  "order"."month3\_order\_cnt" VARCHAR(100),  "order"."month3\_order\_amt" VARCHAR(100),  "order"."max\_order\_amt" VARCHAR(100),  "order"."min\_order\_amt" VARCHAR(100),  "order"."total\_order\_cnt" VARCHAR(100),  "order"."total\_order\_amt" VARCHAR(100),  "order"."user\_avg\_amt" VARCHAR(100),  "order"."month3\_user\_avg\_amt" VARCHAR(100),  "order"."common\_address" VARCHAR(100),  "order"."common\_paytype" VARCHAR(100),  "order"."month1\_cart\_cnt" VARCHAR(100),  "order"."month1\_cart\_goods\_cnt" VARCHAR(100),  "order"."month1\_cart\_submit\_cnt" VARCHAR(100),  "order"."month1\_cart\_rate" VARCHAR(100),  "order"."month1\_cart\_cancle\_cnt" VARCHAR(100),  "order"."return\_cnt" VARCHAR(100),  "order"."return\_amt" VARCHAR(100),  "order"."reject\_cnt" VARCHAR(100),  "order"."reject\_amt" VARCHAR(100),  "order"."last\_return\_time" VARCHAR(100),  "order"."school\_order\_cnt" VARCHAR(100),  "order"."company\_order\_cnt" VARCHAR(100),  "order"."home\_order\_cnt" VARCHAR(100),  "order"."forenoon\_order\_cnt" VARCHAR(100),  "order"."afternoon\_order\_cnt" VARCHAR(100),  "order"."night\_order\_cnt" VARCHAR(100),  "order"."morning\_order\_cnt" VARCHAR(100),  "category"."first\_category\_id" VARCHAR(100),  "category"."first\_category\_name" VARCHAR(100),  "category"."second\_category\_id" VARCHAR(100),  "category"."second\_catery\_name" VARCHAR(100),  "category"."third\_category\_id" VARCHAR(100),  "category"."third\_category\_name" VARCHAR(100),  "category"."month1\_category\_cnt" VARCHAR(100),  "category"."month1\_category\_amt" VARCHAR(100),  "category"."month3\_category\_cnt" VARCHAR(100),  "category"."month3\_category\_amt" VARCHAR(100),  "category"."month6\_category\_cnt" VARCHAR(100),  "category"."month6\_category\_amt" VARCHAR(100),  "category"."total\_category\_cnt" VARCHAR(100),  "category"."total\_category\_amt" VARCHAR(100),  "category"."month1\_cart\_category\_cnt" VARCHAR(100),  "category"."month3\_cart\_category\_cnt" VARCHAR(100),  "category"."month6\_cart\_category\_cnt" VARCHAR(100),  "category"."total\_cart\_category\_cnt" VARCHAR(100),  "category"."last\_category\_time" VARCHAR(100),  "category"."last\_category\_ago" VARCHAR(100),  "visit"."latest\_pc\_visit\_date" VARCHAR(100),  "visit"."latest\_app\_visit\_date" VARCHAR(100),  "visit"."latest\_pc\_visit\_session" VARCHAR(100),  "visit"."latest\_pc\_cookies" VARCHAR(100),  "visit"."latest\_pc\_pv" VARCHAR(100),  "visit"."latest\_pc\_browser\_name" VARCHAR(100),  "visit"."latest\_pc\_visit\_os" VARCHAR(100),  "visit"."latest\_app\_name" VARCHAR(100),  "visit"."latest\_app\_visit\_os" VARCHAR(100),  "visit"."latest\_visit\_ip" VARCHAR(100),  "visit"."latest\_city" VARCHAR(100),  "visit"."latest\_province" VARCHAR(100),  "visit"."first\_pc\_visit\_date" VARCHAR(100),  "visit"."first\_app\_visit\_date" VARCHAR(100),  "visit"."first\_pc\_visit\_session" VARCHAR(100),  "visit"."first\_pc\_cookies" VARCHAR(100),  "visit"."first\_pc\_pv" VARCHAR(100),  "visit"."first\_pc\_browser\_name" VARCHAR(100),  "visit"."first\_pc\_visit\_os" VARCHAR(100),  "visit"."first\_app\_name" VARCHAR(100),  "visit"."first\_app\_visit\_os" VARCHAR(100),  "visit"."first\_visit\_ip" VARCHAR(100),  "visit"."first\_city" VARCHAR(100),  "visit"."first\_province" VARCHAR(100),  "visit"."day7\_app\_cnt" VARCHAR(100),  "visit"."day15\_app\_cnt" VARCHAR(100),  "visit"."month1\_app\_cnt" VARCHAR(100),  "visit"."month2\_app\_cnt" VARCHAR(100),  "visit"."month3\_app\_cnt" VARCHAR(100),  "visit"."day7\_pc\_cnt" VARCHAR(100),  "visit"."day15\_pc\_cnt" VARCHAR(100),  "visit"."month1\_pc\_cnt" VARCHAR(100),  "visit"."month2\_pc\_cnt" VARCHAR(100),  "visit"."month3\_pc\_cnt" VARCHAR(100),  "visit"."month1\_pc\_days" VARCHAR(100),  "visit"."month1\_pc\_pv" VARCHAR(100),  "visit"."month1\_pc\_avg\_pv" VARCHAR(100),  "visit"."month1\_pc\_diff\_ip\_cnt" VARCHAR(100),  "visit"."month1\_pc\_diff\_cookie\_cnt" VARCHAR(100),  "visit"."month1\_pc\_common\_ip" VARCHAR(100),  "visit"."month1\_pc\_common\_cookie" VARCHAR(100),  "visit"."month1\_pc\_common\_browser\_name" VARCHAR(100),  "visit"."month1\_pc\_common\_os" VARCHAR(100),  "visit"."month1\_hour025\_cnt" VARCHAR(100),  "visit"."month1\_hour627\_cnt" VARCHAR(100),  "visit"."month1\_hour829\_cnt" VARCHAR(100),  "visit"."month1\_hour10212\_cnt" VARCHAR(100),  "visit"."month1\_hour13214\_cnt" VARCHAR(100),  "visit"."month1\_hour15217\_cnt" VARCHAR(100),  "visit"."month1\_hour18219\_cnt" VARCHAR(100),  "visit"."month1\_hour20221\_cnt" VARCHAR(100),  "visit"."month1\_hour22223\_cnt" VARCHAR(100)  ); |

这个语句有几个注意点

在**建立映射表**之前要说明的是，Phoenix是大小写敏感的，并且所有命令都是大写，如果你建的表名没有用双引号括起来，那么无论你输入的是大写还是小写，建立出来的表名都是大写的，如果你需要建立出同时包含大写和小写的表名和字段名，请把表名或者字段名用双引号括起来。  
你可以建立读写的表或者只读的表，他们的区别如下：

* 读写表：如果你定义的列簇不存在，会被自动建立出来，并且赋以空值
* 只读表：你定义的列簇必须事先存在
* IF NOT EXISTS可以保证如果已经有建立过这个表，配置不会被覆盖
* 作为rowkey的字段用 PRIMARY KEY标定
* 列簇用 columnFamily.columnName 来表示

## 9.3 构建maven工程



## 9.4 用户画像查询展现

根据不同的维度进行组合查询，筛选出满足条件的用户。

