**附件一、竞赛样题**

背景描述

大数据时代背景下，电商经营模式发生很大改变。在传统运营模式中，缺乏数据积累，人们在做出一些决策行为过程中，更多是凭借个人经验和直觉，发展路径比较自我封闭。而大数据时代，为人们提供一种全新的思路，通过大量的数据分析得出的结果将更加现实和准 确。商家可以对客户的消费行为信息数据进行收集和整理，比如消费 者购买产品的花费、选择产品的渠道、偏好产品的类型、产品回购周期、购买产品的目的、消费者家庭背景、工作和生活环境、个人消费观和价值观等。通过数据追踪，知道顾客从哪儿来，是看了某网站投放的广告还是通过朋友推荐链接，是新访客还是老用户，喜欢浏览什么产品，购物车有无商品，是否清空，还有每一笔交易记录，精准锁定一定年龄、收入、对产品有兴趣的顾客，对顾客进行分组、标签化，通过不同标签组合运用，获得不同目标群体，以此开展精准推送。

因数据驱动的零售新时代已经到来，没有大数据，我们无法为消费者提供这些体验，为完成电商的大数据分析工作，你所在的小组将应用大数据技术，以 Scala 作为整个项目的基础开发语言，基于大数据平台综合利用 Hive、Spark、Flink、Vue.js 等技术，对数据进行处理、分析及可视化呈现，你们作为该小组的技术人员，请按照下面任务完成本次工作。

任务A：大数据平台搭建(容器环境)（15分)

环境说明：

|  |
| --- |
| 服务端登录地址详见各任务服务端说明。补充说明：宿主机及各容器节点可通过 Asbru 工具或 SSH 客户端进行SSH访问。 |

子任务一：Hadoop 完全分布式安装配置

本任务需要使用 root 用户完成相关配置，安装 Hadoop 需要配置前置环境。命令中要求使用绝对路径，具体要求如下:

1. 从宿主机/opt目录下将文件 hadoop-3.1.3.tar.gz、jdk-8u212- linux-x64.tar.gz 复制到容器Master 中的/opt/software 路径中 （若路径不存在，则需新建），将 Master 节点 JDK 安装包解压到 /opt/module 路径中(若路径不存在，则需新建)，将 JDK 解压命 令复制并粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对应的任务序号下；
2. 修改容器中/etc/profile 文件，设置 JDK 环境变量并使其生效， 配置完毕后在 Master 节点分别执行 “ java -version”和 “ java c ”命令，将命令行执行结果分别截图并粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对应的任务序号下；
3. 请完成 host 相关配置，将三个节点分别命名为 master、slave1、 slave2，并做免密登录，用 scp 命令并使用绝对路径从 Master 复 制 JDK 解压后的安装文件到 slave1、slave2 节点（若路径不存在， 则需新建），并配置 slave1、slave2 相关环境变量，将全部 scp 复制 JDK 的命令复制并粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对应的任务序号下；
4. 在容器 Master 将 Hadoop 解压到/opt/module (若路径不存在，则需新建)目录下，并将解压包分发至 slave1、slave2 中，其中 master、slave1、slave2 节点均作为 datanode，配置好相关环境， 初始化 Hadoop 环境 namenode，将初始化命令及初始化结果截图 （截取初始化结果日志最后 20 行即可）粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对应的任务序号下；
5. 启动 Hadoop 集群（包括 hdfs 和 yarn），使用 jps 命令查看 Mas ter 节点与 slave1 节点的 Java 进程，将 jps 命令与结果截图粘 贴至客户端桌面【Release\任务 A 提交结果.docx】中对应的任务序号下。

子任务二：Spark on Yarn 安装配置

本任务需要使用 root 用户完成相关配置， 已安装 Hadoop 及需要配置前置环境，具体要求如下：

1. 从宿主机/opt目录下将文件 spark-3.1.1-bin-hadoop3.2.tgz 复制到容器 Master中的/opt/software（若路径不存在，则需新建）中，将 Spark 包解压到/opt/module 路径中(若路径不存在，则需新建)，将完整解压命令复制粘贴至客户端桌面【Release\任务 A提交结果.docx】中对应的任务序号下；
2. 修改容器中/etc/profile文件，设置 Spark 环境变量并使环境变 量生效，在/opt 目录下运行命令 spark-submit --version，将命令与结果截图粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对应的任务序号下；
3. 完成 on yarn 相关配置，使用 spark on yarn 的模式提交$SPARK\_HOME/examples/jars/spark-examples\_2.12-3.1.1.jar 运行的 主类为 org.apache.spark.examples.SparkPi，将运行结果截图 粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对应的任务序号下（截取 Pi 结果的前后各 5 行）。

（运行命令为：spark-submit --master yarn --class org.ap ache.spark.examples.SparkPi $SPARK\_HOME/examples/jars/spark-example\_2.12-3.1.1.jar）

子任务三：HBase 分布式安装配置

本任务需要使用 root用户完成相关配置，安装HBase 需要配置 Hadoop 和 ZooKeeper 等前置环境。命令中要求使用绝对路径，具体要求如下:

1. 从宿主机/opt 目录下将文件 apache-zookeeper-3.5.7-bin.tar.gz、hbase-2.2.3-bin.tar.gz 复制到容器 Master 中的/opt/software 路径中 (若路径不存在，则需新建) ，将 zookeeper、hbase 安装包解压到/opt/module 目录下，将 HBase 的解压命令复制并 粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对应的任务序号下；
2. 完成 ZooKeeper 相关部署，用 scp 命令并使用绝对路径从容器ma ster 复制 HBase 解压后的包分发至 slave1、slave2 中，并修改相关配置，配置好环境变量，在容器Master 节点中运行命令hbase version，将全部复制命令复制并将 hbase version 命令的结果截图粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对 应的任务序号下;
3. 启动 HBase 后在三个节点分别使用 jps 命令查看，并将结果分别 截图粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对应 的任务序号下；正常启动后在 hbase shell 中查看命名空间，将 查看命名空间的结果截图粘贴至客户端桌面【Release\任务 A 提交结果.docx】中对应的任务序号下。

任务B：离线数据处理(25分)

环境说明：

|  |
| --- |
| 服务端登录地址详见各任务服务端说明。补充说明：各节点可通过 Asbru 工具或 SSH 客户端进行 SSH 访问；主节点 MySQL 数据库用户名/密码：root/123456（已配置远程连接）；Hive 的配置文件位于/opt/apache-hive-2.3.4-bin/conf/Spark 任务在 Yarn 上用 Client 运行，方便观察日志。 |

子任务一：数据抽取

编写 Scala 代码，使用 Spark 将 MySQL 的 shtd\_store 库中表 user\_info、sku\_info、base\_province、base\_region、order\_info、o r der\_detail 的数据增量抽取到 Hive 的 ods 库中对应表 user\_info、 sku\_info、base\_province、base\_region、order\_info、order\_detail 中。

1. 抽取 shtd\_store 库中 user\_info 的增量数据进入 Hive 的 ods 库中表 user\_info。根据 ods.user\_info 表中 operate\_time 或 create\_time 作为增量字段(即 MySQL 中每条数据取这两个时间中较 大的那个时间作为增量字段去和 ods 里的这两个字段中较大的时 间进行比较)，只将新增的数据抽入，字段名称、类型不变，同时 添加静态分区，分区字段为 etl\_date，类型为 String，且值为当 前比赛日的前一天日期（分区字段格式为 yyyyMMdd）。使用 hive cli 执行 show partitions ods.user\_info 命令，将结果截图 粘贴至客户端桌面【Release\任务B提交结果.docx】中对应的任务序号下；
2. 抽取 shtd\_store 库中 sku\_info 的增量数据进入 Hive 的 ods 库中表 sku\_info。根据 od s.sku\_info 表中 create\_time 作为增量字段，只将新增的数据抽入，字段名称、类型不变，同时添加静态分区，分区字段为 etl\_date，类型为 String，且值为当前比赛日的前一天日期（分区字段格式为 yyyyMMdd）。使用 hive cli 执行 show partition s ods.sku\_info 命令，将结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；
3. 抽取 shtd\_store 库中 base\_province 的增量数据进入 Hive 的 od s 库中表 base\_province。根据 ods.base\_province 表中 id 作为 增量字段，只将新增的数据抽入，字段名称、类型不变并添加字 段 create\_time 取当前时间，同时添加静态分区，分区字段为 etl\_date，类型为 String，且值为当前比赛日的前一天日期（分区字段格式为 yyyyMMdd）。使用 hive cli 执行 show partitions ods.base\_province 命令，将结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；
4. 抽取 shtd\_store 库中 base\_region 的增量数据进入 Hive 的 ods 库中表 base\_region。根据 ods.base\_region 表中 id 作为增量字段，只将新增的数据抽入，字段名称、类型不变并添加字段 crea te\_time 取当前时间，同时添加静态分区，分区字段为 etl\_date， 类型为 String，且值为当前比赛日的前一天日期（分区字段格式 为 yyyyMMdd）。使用 hive cli 执行 show partitions ods.base\_region 命令，将结果截图粘贴至客户端桌面【Release\任务 B提交结果.docx】中对应的任务序号下；
5. 抽取 shtd\_store 库中 order\_info 的增量数据进入 Hive 的 ods 库中表 order\_info，根据 ods.order\_info 表中 operate\_time 或 create\_time 作为增量字段(即 MySQL 中每条数据取这两个时间中 较大的那个时间作为增量字段去和 ods 里的这两个字段中较大的时间进行比较)，只将新增的数据抽入，字段名称、类型不变，同时添加静态分区，分区字段为 etl\_date，类型为 String，且值为 当前比赛日的前一天日期（分区字段格式为 yyyyMMdd）。使用 hive cli 执行 show partitions ods.order\_info 命令，将结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；
6. 抽取 shtd\_store 库中 order\_detail 的增量数据进入 Hive 的 ods库中表 order\_detail，根据 ods.order\_detail 表中 create\_time 作为增量字段，只将新增的数据抽入，字段名称、类型不变，同时添加静态分区，分区字段为 etl\_date，类型为 String，且值 为当前比赛日的前一天日期（分区字段格式为yyyyMMdd）。使用 hive cli 执行 show partitions ods.order\_detail 命令，将结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；

子任务二：数据清洗

编写 Scala 代码，使用 Spark 将 ods 库中相应表数据全量抽取到 Hive 的 dwd 库中对应表中。表中有涉及到 timestamp 类型的，均要求按照 yyyy-MM-dd HH:mm:ss，不记录毫秒数，若原数据中只有年月日，则在时分秒的位置添加 00:00:00，添加之后使其符合 yyyy-MM-dd HH:mm:ss。

1. 抽取 ods 库中 user\_info 表中昨天的分区（子任务一生成的分区） 数据，并结合 dim\_user\_info 最新分区现有的数据，根据 id 合并 数据到 dwd 库中 dim\_user\_info 的分区表（合并是指对 dwd 层数 据进行插入或修改，需修改的数据以 id 为合并字段，根据 operate\_time 排序取最新的一条），分区字段为 etl\_date 且值与 ods 库的相对应表该值相等，同时若 operate\_time 为空，则用 create\_time 填充，并添加 dwd\_insert\_user、dwd\_insert\_time、dwd\_modify\_user、dwd\_modify\_time 四列,其中 dwd\_insert\_user、dwd\_modify\_user 均填写“user1”。若该条记录第一次进入数仓 d wd 层则 dwd\_insert\_time、dwd\_modify\_time 均存当前操作时间， 并进行数据类型转换。若该数据在进入 dwd 层时发生了合并修改， 则 dwd\_insert\_time 时间不变，dwd\_modify\_time 存当前操作时 间，其余列存最新的值。使用 hive cli 执行 show partitions dwd.dim\_user\_info 命令，将结果截图粘贴至客户端桌面【Re lease\任务 B 提交结果.docx】中对应的任务序号下；
2. 抽取 ods 库 sku\_info 表中昨天的分区（子任务一生成的分区）数 据，并结合 dim\_sku\_info 最新分区现有的数据，根据 id 合并数据到 dwd 库中 dim\_sku\_info 的分区表（合并是指对 dwd 层数据进 行插入或修改，需修改的数据以 id 为合并字段，根据 create\_time 排序取最新的一条），分区字段为 etl\_date 且值与 ods 库的 相对应表该值相等，并添加 dwd\_insert\_user、dwd\_insert\_time、 dwd\_modify\_user、dwd\_modify\_time 四列,其中 dwd\_insert\_use r、dwd\_modify\_user 均填写 “user1 ”。若该条数据第一次进入 数仓 dwd 层则 dwd\_insert\_time、dwd\_modify\_time 均填写当前操 作时间，并进行数据类型转换。若该数据在进入 dwd 层时发生了 合并修改，则 dwd\_insert\_time 时间不变，dwd\_modify\_time 存 当前操作时间，其余列存最新的值。使用 hive cli 查询表 dim\_s ku\_info 的字段 id、sku\_desc、dwd\_insert\_user、dwd\_modify\_ time、etl\_date，条件为最新分区的数据，id 大于等于 15 且小于等于 20，并且按照 id 升序排序，将结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；
3. 抽取 ods 库 base\_province 表中昨天的分区（子任务一生成的分区）数据，并结合 dim\_province 最新分区现有的数据，根据 id 合并数据到 dwd 库中 dim\_province 的分区表（合并是指对 dwd 层 数据进行插入或修改，需修改的数据以 id 为合并字段，根据 create\_time 排序取最新的一条），分区字段为 etl\_date 且值与 ods 库的相对应表该值相等，并添加 dwd\_insert\_user、dwd\_in ser t\_time、dwd\_modify\_user、dwd\_modify\_time 四列,其中 dwd\_insert\_user、dwd\_modify\_user 均填写 “user1 ”。若该条数据第 一次进入数仓 dwd 层则 dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间，并进行数据类型转换。若该数据在进入 dwd 层时发生了合并修改，则 dwd\_insert\_time 时间不变，dwd\_modify\_time 存当前操作时间，其余列存最新的值。使用 hive cli 在表 dwd.dim\_province 最新分区中，查询该分区中数据的条数，将结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；
4. 抽取 ods 库 base\_region 表中昨天的分区（子任务一生成的分区） 数据，并结合 dim\_region 最新分区现有的数据，根据 id 合并数 据到 dwd 库中 dim\_region 的分区表（合并是指对 dwd 层数据进行 插入或修改，需修改的数据以 id 为合并字段，根据 create\_time 排序取最新的一条），分区字段为 etl\_date 且值与 ods 库的相对应表该值相等，并添加 dwd\_insert\_user、dwd\_insert\_time、dwd\_modify\_user、dwd\_modify\_time 四列,其中 dwd\_insert\_user、 dwd\_modify\_user 均填写“user1”。若该条数据第一次进入数仓 dwd 层则 dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时 间，并进行数据类型转换。若该数据在进入 dwd 层时发生了合并 修改，则 dwd\_insert\_time 时间不变，dwd\_modify\_time 存当前 操作时间，其余列存最新的值。使用 hive cli 在表 dwd.dim\_reg ion 最新分区中，查询该分区中数据的条数，将结果截图粘贴至 客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；
5. 将 ods 库中 order\_info 表昨天的分区（子任务一生成的分区）数据抽取到 dwd 库中 fact\_order\_info 的动态分区表，分区字段为 etl\_date，类型为 String，取 create\_time 值并将格式转换为 yyyyMMdd，同时若 operate\_time 为空，则用 create\_time 填充，并添加 dwd\_insert\_user、dwd\_insert\_time、dwd\_modify\_user、dwd\_modify\_time 四列，其中 dwd\_insert\_user、dwd\_modify\_us er 均填写 “user1”，dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间，并进行数据类型转换。使用 hive cli 执行 show partitions dwd.fact\_order\_info 命令，将结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；
6. 将 ods库中 order\_detail表昨天的分区（子任务一中生成的分区） 数据抽取到 dwd 库中 fact\_order\_detail 的动态分区表，分区字段为 etl\_date，类型为 String，取 create\_time 值并将格式转换为 yyyyMMdd，并添加 dwd\_insert\_user、dwd\_insert\_time、dwd \_modify\_user、dwd\_modify\_time 四列，其中 dwd\_insert\_user、 dwd\_modify\_user 均填写 “user1”，dwd\_insert\_time、dwd\_mod ify\_time 均填写当前操作时间，并进行数据类型转换。使用 hiv e cli 执行 show partitions dwd.fact\_order\_detail 命令，将 结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下。

子任务三：指标计算

编写 Scala 代码，使用 Spark 计算相关指标。

1.本任务基于以下 2、3、4 小题完成，使用 Azkaban 完成第 2、3、4 题任务代码的调度。工作流要求，使用 shell 输出 “开始 ”作为工作流的第一个 job（job1），2、3、4 题任务为串行任务且它 们依赖 job1 的完成（命名为 job2、job3、job4），job2、job3、job4 完成之后使用 shell 输出 “结束”作为工作流的最后一个 job（endjob），endjob 依赖 job2、job3、job4，并将最终任务调 度完成后的工作流截图，将截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| provinceid | in t | 省份表主键 |  |
| provincename | text | 省份名称 |  |
| region id | in t | 地区表主键 |  |
| regionname | text | 地区名称 |  |
| totalconsumption | double | 订单总金额 | 当月订单总金额 |
| totalorder | in t | 订单总数 | 当月订单总数 |
| year | in t | 年 | 订单产生的年 |
| month | in t | 月 | 订单产生的月 |

2.根据 dwd 层表统计每个省份、每个地区、每个月下单的数量和下 单的总金额，存入 MySQL 数据库 shtd\_result 的 provinceeverymon th 表中（表结构如下），然后在 Linux 的 MySQL 命令行中根据 订单总数、订单总金额、省份表主键均为降序排序，查询出前 5 条，将 SQL 语句复制粘贴至客户端桌面【Release\任务 B 提交结 果.docx】中对应的任务序号下，将执行结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下；

3.请根据 dwd 层表计算出 2020 年 4 月每个省份的平均订单金额和所有省份平均订单金额相比较结果（ “高/低/相同”）,存入 MySQL 数据库 shtd\_result 的 provinceavgcmp 表（表结构如下）中，然后在 Linux 的 MySQL 命令行中根据省份表主键、该省平均订单 金额均为降序排序，查询出前 5 条，将 SQL 语句复制粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下，将执行结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下;

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| province id | in t | 省份表主键 |  |
| provincename | text | 省份名称 |  |
| provinceavgconsumption | double | 该省平均订单金额 |  |
| allprovinceavgconsumption | double | 所有省平均订单金额 |  |
| comparison | text | 比较结果 | 该省平均订单金额和所有省平均订单金额比较结果，值为：高/低/相同 |

4.根据 dwd 层表统计在两天内连续下单并且下单金额保持增长的用户，存入 MySQL 数据库 shtd\_result 的 usercontinueorder 表(表结构如下)中，然后在 Linux 的 MySQL 命令行中根据订单总数、订单总金额、客户主键均为降序排序，查询出前 5 条，将 SQL 语句 复制粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应 的任务序号下，将执行结果截图粘贴至客户端桌面【Release\任务 B 提交结果.docx】中对应的任务序号下。

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| userid | in t | 客户主键 |  |
| username | text | 客户名称 |  |
| day | text | 日 | 记录下单日的时间，格式为yyyyMMdd\_yyyyMMdd例如： 20220101\_20220102 |
| totalconsumption | double | 订单总金额 | 连续两天的订单总金额 |
| totalorder | in t | 订单总数 | 连续两天的订单总数 |

任务 C：数据挖掘（10 分）

环境说明：

|  |
| --- |
| 服务端登录地址详见各任务服务端说明。补充说明：各节点可通过 Asbru 工具或 SSH 客户端进行 SSH 访问； 主节点 MySQL 数据库用户名/密码：root/123456（已配置远程连接）；Hive 的配置文件位于/opt/apache-hive-2.3.4-bin/conf/Spark 任务在 Yarn 上用 Client 运行，方便观察日志。该任务均使用 Scala 编写，利用 Spark 相关库完成。 |

子任务一：特征工程

1.根据 Hive 的 dwd 库中相关表或 MySQL 中 shtd\_store 中相关表（o r der\_detail、sku\_info），计算出与用户 id 为 6708 的用户所购 买相同商品种类最多的前 10 位用户（只考虑他俩购买过多少种相 同的商品，不考虑相同的商品买了多少次），将 10 位用户 id 进 行输出，若与多个用户购买的商品种类相同，则输出结果按照用户 id 升序排序，输出格式如下，将结果截图粘贴至客户端桌面【Release\任务 C 提交结果.docx】中对应的任务序号下；结果格式如下：

-----------------相同种类前 10 的 id 结果展示为：------------------

1,2,901,4,5,21,32,91,14,52

2.根据 Hive 的 dwd 库中相关表或 MySQL中 shtd\_store 中相关商品表（sku\_info），获取 id、spu\_id、price、weight、tm\_id、category3\_id 这六个字段并进行数据预处理，对 price、weight 进 行规范化(StandardScaler)处理，对 spu\_id、tm\_id、category3\_id 进行 one-hot 编码处理（若该商品属于该品牌则置为 1，否则置为 0）,并按照 id 进行升序排序，在集群中输出第一条数据前 10 列（无需展示字段名），将结果截图粘贴至客户端桌面【Re lease\任务 C 提交结果.docx】中对应的任务序号下。

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| id | double | 主键 |  |
| price | double | 价格 |  |
| weight | double | 重量 |  |
| spu\_id#1 | double | spu\_id 1 | 若属于该 spu\_id，则内容为 1 否则为 0 |
| spu\_id#2 | double | spu\_id 2 | 若属于该 spu\_id，则内容为 1 否则为 0 |
| ..... | double |  |  |
| tm\_id#1 | double | 品牌 1 | 若属于该品牌，则内容为 1 否则为 0 |
| tm\_id#2 | double | 品牌 2 | 若属于该品牌，则内容为 1 否则为 0 |
| …… | double |  |  |
| category3\_id#1 | double | 分类级别 31 | 若属于该分类级别 3，则内容为 1 否则为 0 |
| category3\_id#2 | double | 分类级别 32 | 若属于该分类级别 3，则内容为 1 否则为 0 |
| …… |  |  |  |

结果格式如下：

------------第一条数据前 10 列结果展示为：----------

1.0,0.892346,1.72568,0.0,0.0,0.0,0.0,1.0,0.0,0.0

子任务二：推荐系统

1.根据子任务一的结果，计算出与用户 id 为 6708 的用户所购买相 同商品种类最多的前 10 位用户 id（只考虑他俩购买过多少种相 同的商品，不考虑相同的商品买了多少次），并根据 Hive 的 dwd 库中相关表或 MySQL 数据库 shtd\_store 中相关表，获取到这 10 位用户已购买过的商品，并剔除用户 6708已购买的商品，通过计 算这 10 位用户已购买的商品（剔除用户 6708 已购买的商品）与用户 6708已购买的商品数据集中商品的余弦相似度累加再求均值，输出均值前 5 商品 id 作为推荐使用，将执行结果截图粘贴至客户端桌面【Release\任务 C 提交结果.docx】中对应的任务序号下。

结果格式如下：

----------------推荐 Top5 结果如下-----------------

相似度 top1(商品 id：1，平均相似度：0.983456)

相似度 top2(商品 id：71，平均相似度：0.782672)

相似度 top3(商品 id：22，平均相似度：0.7635246)

相似度 top4(商品 id：351，平均相似度：0.7335748)

相似度 top5(商品 id：14，平均相似度：0.522356)

**任务 D：数据采集与实时计算（20 分）**

环境说明：

|  |
| --- |
| 服务端登录地址详见各任务服务端说明。补充说明：各节点可通过 Asbru 工具或 SSH 客户端进行 SSH 访问； Flink 任务在 Yarn 上用 per job 模式（即 Job 分离模式，不采用 Session 模式），方便 Yarn 回收资源。 |

子任务一：实时数据采集

1.在主节点使用 Flume 采集实时数据生成器 10050 端口的 socket 数据，将数据存入到 Kafka 的 Topic 中（Topic 名称为 order，分区数为 4），使用 Kafka自带的消费者消费 order（Topic）中的数 据，将前 2 条数据的结果截图粘贴至客户端桌面【Release\任务D 提交结果.docx】中对应的任务序号下；

2.采用多路复用模式，Flume 接收数据注入 kafka的同时，将数据 备份到 HDFS 目录/user/test/flumebackup 下，将查看备份目录下的第一个文件的前 2 条数据的命令与结果截图粘贴至客户端桌面【Release\任务 D 提交结果.docx】中对应的任务序号下。

子任务二：使用 Flink 处理 Kafka 中的数据

编写 Scala 代码，使用 Flink 消费 Kafka中 Topic 为 order 的数 据并进行相应的数据统计计算（订单信息对应表结构 order\_info,订单详细信息对应表结构 order\_detail，同时计算中使用 order\_info 或 order\_detail 表中 create\_time 或 operate\_time 取两者中值较大 者作为 EventTime，若 operate\_time 为空值或无此列，则使用 create\_time 填充，允许数据延迟 5s）。

1. 使用 Flink 消费 Kafka中的数据，统计商城实时订单实收金额， 将 key 设置成 totalprice 存入 Redis中。使用 redis cli以 get key 方式获取 totalprice 值，将结果截图粘贴至客户端桌面【R e lease\任务 D 提交结果.docx】中对应的任务序号下，需两次截图，第一次截图和第二次截图间隔 1 分钟以上，第一次截图放前面，第二次截图放后面；
2. 在任务 1 进行的同时，使用侧边流，监控若发现 order\_status 字段为退回完成, 将 key 设置成 totalrefundordercount 存入 Redis中，value 存放用户退款消费额。使用 redis cli以 get key 方式获取 totalrefundordercount 值，将结果截图粘贴至客户端桌面【Release\任务 D 提交结果.docx】中对应的任务序号下，需两次截图，第一次截图和第二次截图间隔 1 分钟以上，第一次截图放前面，第二次截图放后面；
3. 在任务 1 进行的同时，使用侧边流，监控若发现 order\_status 字段为取消订单,将数据存入 MySQL 数据库 shtd\_result 的 order\_info 表中，然后在 Linux 的 MySQL 命令行中根据 id 降序排序， 查询列 id、consignee、consignee\_tel、final\_total\_amount、 feight\_fee，查询出前 5 条，将 SQL 语句复制粘贴至客户端桌面 【Release\任务 D 提交结果.docx】中对应的任务序号下，将执行 结果截图粘贴至客户端桌面【Release\任务 D 提交结果.docx】中对应的任务序号下。

**任务 E：数据可视化（15 分）**

环境说明：

|  |
| --- |
| 数据接口地址及接口描述详见各任务服务端说明。 |

子任务一：用柱状图展示消费额最高的省份

编写 Vue 工程代码，根据接口，用柱状图展示 2020 年消费额最高的5 个省份，同时将用于图表展示的数据结构在浏览器的 console 中进行打印输出，将图表可视化结果和浏览器 console 打印结果分别截图 并粘贴至客户端桌面【Release\任务 E 提交结果.docx】中对应的任务序号下。

子任务二：用柱状图展示消费额最低的省份

编写 Vue 工程代码，根据接口，用柱状图展示 2020 年消费额最低的 5 个省份，同时将用于图表展示的数据结构在浏览器的 console 中进行打印输出，将图表可视化结果和浏览器 console 打印结果分别截图 并粘贴至客户端桌面【Release\任务 E 提交结果.docx】中对应的任务序号下。

子任务三：用折线图展示每年上架商品数量变化

编写 Vue 工程代码，根据接口，用折线图展示每年上架商品数量的变化情况，同时将用于图表展示的数据结构在浏览器的 console中进行打印输出，将图表可视化结果和浏览器 console 打印结果分别截图并 粘贴至客户端桌面【Release\任务 E 提交结果.docx】中对应的任务序号下。

子任务四：用条形图展示平均消费额最高的省份

编写 Vue 工程代码，根据接口，用条形图展示 2020 年平均消费额（四舍五入保留两位小数）最高的 5 个省份，同时将用于图表展示的数据结构在浏览器的 console 中进行打印输出，将图表可视化结果和浏览 器 console 打印结果分别截图并粘贴至客户端桌面【Release\任务 E提交结果.docx】中对应的任务序号下。

子任务五：用折柱混合图展示省份平均消费额和地区平均消费额

编写 Vue工程代码，根据接口，用折柱混合图展示 2020 年各省份平 均消费额（四舍五入保留两位小数）和地区平均消费额（四舍五入保留两位小数）的对比情况，柱状图展示平均消费额最高的 5 个省份，折线图展示这 5 个省所在的地区的平均消费额变化，同时将用于图表展示的数据结构在浏览器的 console中进行打印输出，将图表可视化 结果和浏览器 console 打印结果分别截图并粘贴至客户端桌面【Release\任务 E 提交结果.docx】中对应的任务序号下。

**任务 F：综合分析（10 分）**

子任务一：F link 有哪些重启策略？各个重启策略如何配置？

在任务 D中使用到了 Flink，Flink 在运行 job 时可能会出现各种问 题，从而会导致其失败或者重启，对于类似于网络波动造成的运行失 败可以采取相对应重启策略来重试，请问 Flink 有几种重启策略（中文）？分别怎么配置这些重启策略？将内容编写至客户端桌面【Release\任务 F 提交结果.docx】中对应的任务序号下。

子任务二：Hadoop 有哪些类型的调度器？简要说明其工作方法。

简要描述 Hadoop 有哪些类型的调度器并简要说明其工作方法，将内 容编写至客户端桌面【Release\任务 F 提交结果.docx】中对应的任务序号下。

子任务三：分析下一年度的建仓目的地。

根据任务E的图表，分析各省份的经济现状，公司决定挑选 3 个省份进行仓储建设，请问应该在哪些省份建设？将内容编写至客户端桌面【Release\任务 F 提交结果.docx】中对应的任务序号下。