

2024 年河南省职业院校  
技能大赛高职组

“大数据分析与应用”  
赛项

任  
务  
书  
5

参赛队编号：\_\_\_\_\_

## 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 Java、Scala、JavaScript 作为整个项目的基础开发语言，基于大数据平台综合利用 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。

## 模块 A：数据采集（15 分）

### 环境说明：

服务端登录地址详见各模块服务端说明。

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 的配置文件位于主节点/opt/module/hive-3.1.2/conf/  
Spark 任务在 Yarn 上用 Client 运行，方便观察日志；  
建议使用 gson 解析 json 数据。

### 任务一：离线数据采集

编写 Scala 工程代码，将 MySQL 的 ds\_db01 库中表 order\_master、order\_detail、coupon\_info、coupon\_use、product\_browse、product\_info、customer\_inf、customer\_login\_log、order\_cart、customer\_level\_inf、customer\_addr 的数据增量抽取到 Hive 的 ods 库中对应表 order\_master、order\_detail、coupon\_info、coupon\_use、product\_browse、product\_info、customer\_inf、customer\_login\_log、order\_cart、customer\_level\_inf、customer\_addr 中(ods 库中部分表没有数据，正常抽取即可)。

1、抽取 ds\_db01 库中 order\_master 的增量数据进入 Hive 的 ods 库中表 order\_master。根据 ods.order\_master 表中 modified\_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\_master 命令，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；

2、抽取 ds\_db01 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 modified\_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，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；

3、抽取 ds\_db01 库中 coupon\_info 的增量数据进入 Hive 的 ods 库中表 coupon\_info，根据 ods.coupon\_info 表中 modified\_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.coupon\_info 命令，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；

4、抽取 ds\_db01 库中 coupon\_use 的增量数据进入 Hive 的 ods 库中表 coupon\_use，增量字段取 ods.coupon\_use 表中 get\_time、used\_time、pay\_time 中的最大者，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 查询最新分区数据总条数，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；

5、抽取 ds\_db01 库中 product\_browse 的增量数据进入 Hive 的 ods 库中表 product\_browse，根据 ods.product\_browse 表中 modified\_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.product\_browse 命令，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下。

- 6、抽取 ds\_db01 库中 product\_info 的增量数据进入 Hive 的 ods 库中表 product\_info，根据 ods.product\_info 表中 modified\_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.product\_info 命令，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；
- 7、抽取 ds\_db01 库中 customer\_inf 的增量数据进入 Hive 的 ods 库中表 customer\_inf，根据 ods.customer\_inf 表中 modified\_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.customer\_inf 命令，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；
- 8、抽取 ds\_db01 库中 customer\_login\_log 的增量数据进入 Hive 的 ods 库中表 customer\_login\_log，根据 ods.customer\_login\_log 表中 login\_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.customer\_login\_log 命令，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；
- 9、抽取 ds\_db01 库中 order\_cart 的增量数据进入 Hive 的 ods 库中表 order\_cart，根据 ods.order\_cart 表中 modified\_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\_cart 命令，

将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；

10、抽取 ds\_db01 库中 customer\_addr 的增量数据进入 Hive 的 ods 库中表 customer\_addr，根据 ods.customer\_addr 表中 modified\_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.customer\_addr 命令，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；

11、抽取 ds\_db01 库中 customer\_level\_inf 的增量数据进入 Hive 的 ods 库中表 customer\_level\_inf，根据 ods.customer\_level\_inf 表中 modified\_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.customer\_level\_inf 命令，将执行结果截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下。

## 任务二：实时数据采集

1、在主节点使用 Flume 采集实时数据生成器 25001 端口的 socket 数据（实时数据生成器脚本为主节点/data\_log 目录下的 gen\_ds\_data\_to\_socket 脚本，该脚本为 Master 节点本地部署且使用 socket 传输），将数据存入到 Kafka 的 Topic 中（Topic 名称为 ods\_mall\_log，分区数为 2，ZK 关于 Kafka 的信息在其/kafka 节点），使用 Kafka 自带的消费者消费 ods\_mall\_log (Topic) 中的数据，查看 Topic 中的前 1 条数据的结果，将查看命令与结果完整的截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下；

注：需先启动已配置好的 Flume 再启动脚本，否则脚本将无法成功启动，启动

方式为进入/data\_log 目录执行./gen\_ds\_data\_to\_socket（如果没有权限，请执行授权命令 chmod 777 /data\_log/gen\_ds\_data\_to\_socket）

2、实时脚本启动后，在主节点进入到 maxwell-1.29.0 的解压后目录下（在 /opt/module 下），配置相关文件并启动，读取主节点 MySQL 数据的 binlog 日志（MySQL 的 binlog 相关配置已完毕，只需要关注 ds\_realtime\_db 数据库的表）到 Kafka 的 Topic 中（Topic 名称为 ods\_mall\_data，分区数为 2，ZK 关于 Kafka 的信息在其/kafka 节点）。使用 Kafka 自带的消费者消费 ods\_mall\_data（Topic）中的数据，查看 Topic 中的前 1 条数据的结果，将查看命令与结果完整的截图粘贴至客户端桌面【Release\模块 A 提交结果.docx】中对应的任务序号下。

## 模块 B：实时数据处理（25 分）

### 环境说明：

服务端登录地址详见各模块服务端说明。

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)；  
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；  
建议使用 gson 解析 json 数据。

### 任务一：实时数据清洗

编写 Java 代码，使用 Flink 消费 Kafka 中 Topic 为 ods\_mall\_log 和 ods\_mall\_data 的数据并进行相应的数据统计计算(使用 Processing Time)。

- 1、使用 Flink 消费 Kafka 中 topic 为 ods\_mall\_data 的数据，根据数据中不同的表将数据分别分发至 kafka 的 DWD 层的 fact\_order\_master、fact\_order\_detail 的 Topic 中（只获取 data 的内容，具体的内容格式请自查，其分区数均为 2），其他的表则无需处理。使用 Kafka 自带的消费者消费 fact\_order\_master (Topic) 的前 1 条数据，将结果截图粘贴至客户端桌面【Release\模块 B 提交结果.docx】中对应的任务序号下；

fact\_order\_master 表结构，存储位置：Kafka，存储格式：json

字段	类型	中文含义（和 MySQL 中相同）	备注
order_id	int		
order_sn	string		
customer_id	int		
shipping_user	string		



province	string		
city	string		
address	string		
order_source	int		
payment_method	int		
order_money	double		
district_money	double		
shipping_money	double		
payment_money	double		
shipping_comp_name	string		
shipping_sn	string		
create_time	timestamp		
shipping_time	timestamp		
pay_time	timestamp		
receive_time	timestamp		
order_status	string		
order_point	int		
invoice_title	string		
modified_time	timestamp		

fact\_order\_detail 表结构，存储位置：Kafka，存储格式：json

字段	类型	中文含义（和 MySQL 中相同）	备注
order_detail_id	int		
order_sn	string		
product_id	int		
product_name	string		
product_cnt	int		
product_price	double		

average_cost	double		
weight	double		
fee_money	double		
w_id	int		
create_time	timestamp		
modified_time	timestamp		

2、使用 Flink 消费 Kafka 中 topic 为 ods\_mall\_log 的数据，根据数据中不同的表前缀区分，过滤出 product\_browse 的数据，将数据分别分发至 kafka 的 DWD 层 log\_product\_browse 的 Topic 中，其分区数为 2，其他的表则无需处理。使用 Kafka 自带的消费者消费 log\_product\_browse (Topic) 的前 1 条数据，将结果截图粘贴至客户端桌面【Release\模块 B 提交结果.docx】中对应的任务序号下。

log\_product\_browse 表结构，存储位置：Kafka，存储格式：json

字段	类型	中文含义 (和 MySQL 中相同)	备注
log_id	long	自增长 id	可以使用随机数(0-9)+MMddHHmmssSSS 代替
product_id	string		
customer_id	int		
gen_order	int		
order_sn	string		
modified_time	timestamp		

3、在任务 1、2 进行的同时，需要将 order\_master、order\_detail、product\_browse 备份至 HBase 中（若 Int 类型长度不够，可使用 BigInt 或

Long 类型代替），使用 HBase Shell 查看 ods:order\_master 表的任意 2 条数据，查看字段为 row\_key 与 shipping\_user、查看 ods:order\_detail 表的任意 2 条数据，查看字段为 row\_key 与 product\_name、查看 ods:product\_browse 表的任意 2 条数据，查看字段为 row\_key 与 order\_sn。将结果分别截图粘贴至客户端桌面【Release\模块 B 提交结果.docx】中对应的任务序号下（截图中不能有乱码）。

三个 HBase 中的数据结构为：

ods:order\_master 数据结构如下：

字段	类型	中文含义 (和 MySQL 中相同)	备注
rowkey	string	rowkey	可以使用随机数 (0-9)+yyyyMMddHHmmssSSS (date 的格式) 代替
Info		列族名	
order_id	int		
order_sn	string		
customer_id	int		
shipping_user	string		
province	string		
city	string		
address	string		
order_source	int		
payment_method	int		
order_money	double		
district_money	double		
shipping_money	double		
payment_money	double		

shipping_comp_name	string		
shipping_sn	string		
create_time	string		
shipping_time	string		
pay_time	string		
receive_time	string		
order_status	string		
order_point	int		
invoice_title	string		
modified_time	string		

ods:order\_detail 数据结构如下:

字段	类型	中文含义 (和 MySQL 中 相同)	备注
rowkey	string	rowkey	可以使用随机数 (0-9) +yyyyMMddHHmmssSSS (date 的格式) 代替
Info		列族名	
order_detail_id	int		
order_sn	string		
product_id	int		
product_name	string		
product_cnt	int		
product_price	double		
average_cost	double		
weight	double		

fee_money	double		
w_id	int		
create_time	string		
modified_time	string		

ods:product\_browse 数据结构如下:

字段	类型	中文含义 (和 MySQL 中 相同)	备注
rowkey	string	rowkey	该字段使用 logid 进行拆分, 将 log_id 拆分为随机数和 MMddHHmmssSSS 两块,在其 中插入 yyyy (date 的格式) 最终格式为: 随机数 (0-9) +yyyy +MMddHHmmssSSS
Info		列族名	
log_id	int		该字段缺失,使用随机数 (0-9) + MMddHHmmssSSS (date 的格式)
order_sn	string		
product_id	int		
customer_id	string		
gen_order	int		
modified_time	double		

## 模块 C：离线数据处理（30 分）

### 环境说明：

服务端登录地址详见各模块服务端说明。

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 的配置文件位于主节点/opt/module/hive-3.1.2/conf/  
Spark 任务在 Yarn 上用 Client 运行，方便观察日志；  
ClickHouse 的 jdbc 连接端口 8123,用户名/密码：default/123456,  
命令行客户端（tcp）端口 9001；  
建议使用 gson 解析 json 数据。

### 任务一：离线数据清洗

编写 Scala 工程代码，将 ods 库中表 order\_master、order\_detail、coupon\_info、coupon\_use、product\_browse、product\_info、customer\_inf、customer\_login\_log、order\_cart、customer\_level\_inf、customer\_addr 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 库中表 customer\_inf 最新分区数据，并结合 dim\_customer\_inf 最新分区现有的数据，根据 customer\_id 合并数据到 dwd 库中 dim\_customer\_inf 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 customer\_id 为合并字段，根据 modified\_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

记录第一次进入数仓 dwd 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 查询 modified\_time 为 2022 年 10 月 01 日当天的数据，查询字段为 customer\_id、customer\_email、modified\_time、dwd\_insert\_time、dwd\_modify\_time，并按照 customer\_id 进行升序排序，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

- 2、抽取 ods 库中表 coupon\_info 最新分区数据，并结合 dim\_coupon\_info 最新分区现有的数据，根据 coupon\_id 合并数据到 dwd 库中 dim\_coupon\_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 coupon\_id 为合并字段，根据 modified\_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”。若该条记录第一次进入数仓 dwd 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\_coupon\_info 命令，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

- 3、抽取 ods 库中表 product\_info 最新分区的数据，并结合 dim\_product\_info 最新分区现有的数据，根据 product\_core 合并数据到 dwd 库中 dim\_product\_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 product\_core 为合并字段，根据 modified\_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

记录第一次进入数仓 dwd 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\_product\_info 命令，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

- 4、抽取 ods 库中表 order\_master 最新分区的数据，并结合 HBase 中 order\_master\_offline 表中的数据合并抽取到 dwd 库中 fact\_order\_master 的分区表，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”，dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间（年月日必须是今天，时分秒只需在比赛时间范围内即可），抽取 HBase 中的数据时，只抽取 2022 年 10 月 01 日的数据（以 rowkey 为准），并进行数据类型转换。使用 hive cli 查询 modified\_time 为 2022 年 10 月 01 日当天的数据，查询字段为 order\_id、order\_sn、shipping\_user、create\_time、shipping\_time，并按照 order\_id 进行升序排序，将结果截图复制粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

ods: order\_master\_offline 数据结构如下：

字段	类型	中文含义	备注
rowkey	string	rowkey	随机数（0-9） +yyyyMMddHHmmssSSS（date 的格式）
Info		列族名	
order_id	int		
order_sn	string		
customer_id	int		
shipping_user	string		



province	string		
city	string		
address	string		
order_source	int		
payment_method	int		
order_money	double		
district_money	double		
shipping_money	double		
payment_money	double		
shipping_comp_name	string		
shipping_sn	string		
create_time	string		
shipping_time	string		
pay_time	string		
receive_time	string		
order_status	string		
order_point	int		
invoice_title	string		
modified_time	string		

5、抽取 ods 库中表 order\_detail 表最新分区的数据，并结合 HBase 中 order\_detail\_offline 表中的数据合并抽取到 dwd 库中 fact\_order\_detail 的分区表，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”，dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间（年月日必须是今天，时分秒只需在比赛时间范围内即可），抽取 HBase 中的数据时，只抽取 2022 年 10 月 01 日的数据（以 rowkey 为准），并进行数据类型转换。使用 hive cli 查询 modified\_time 为 2022 年 10 月 01 日当天的数据，查询字段

为 order\_detail\_id、order\_sn、product\_name、create\_time，并按照 order\_detail\_id 进行升序排序，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

ods:order\_detail\_offline 数据结构如下：

字段	类型	中文含义	备注
rowkey	string	rowkey	随机数（0-9） +yyyyMMddHHmmssSSS（date 的格式）
Info		列族名	
order_detail_id	int		
order_sn	string		
product_id	int		
product_name	string		
product_cnt	int		
product_price	double		
average_cost	double		
weight	double		
fee_money	double		
w_id	int		
create_time	string		
modified_time	string		

6、抽取 ods 库中表 coupon\_use 最新分区的数据到 dwd 库中 fact\_coupon\_use 的分区表，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”，dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间（年月日必须是今天，时分秒只需在比赛时间范围内即可），并进行数据类型转换。使用 hive cli 执行 show partitions dwd.fact\_coupon\_use 命令，将结果截图粘贴至

客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

- 7、抽取 ods 库中表 customer\_login\_log 最新分区的数据到 dwd 库中 log\_customer\_login 的分区表，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”，dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间（年月日必须是今天，时分秒只需在比赛时间范围内即可），并进行数据类型转换。使用 hive cli 执行 show partitions dwd.log\_customer\_login 命令，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；
- 8、抽取 ods 库中表 order\_cart 最新分区的数据到 dwd 库中 fact\_order\_cart 的分区表，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”，dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间（年月日必须是今天，时分秒只需在比赛时间范围内即可），并进行数据类型转换。使用 hive cli 执行 show partitions dwd.fact\_order\_cart 命令，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；
- 9、抽取 ods 库中表 product\_browse 最新分区的数据，并结合 HBase 中 product\_browse\_offline 表中的数据合并抽取到 dwd 库中 log\_product\_browse 的分区表，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”，dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间（年月日必须是今天，时分秒只需在比赛时间范围内即可），抽取 HBase 中的数据时，只抽取 2022 年 10 月 01 日的数据（以 rowkey

为准），并进行数据类型转换。使用 hive cli 查询 modified\_time 为 2022 年 10 月 01 日当天的数据，查询字段为 log\_id、product\_id、order\_sn、modified\_time，并按照 log\_id 进行升序排序，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

ods:product\_browse\_offline 数据结构如下：

字段	类型	中文含义	备注
rowkey	string	rowkey	随机数（0-9） +MMddHHmmssSSS
Info		列族名	
log_id	int		
product_id	int		
customer_id	string		
gen_order	int		
order_sn	string		
modified_time	double		

10、抽取 ods 库中表 customer\_level\_inf 最新分区的数据到 dwd 库中 dim\_customer\_level\_inf 的分区表，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”，dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间（年月日必须是今天，时分秒只需在比赛时间范围内即可），并进行数据类型转换。使用 hive cli 执行 show partitions dwd.dim\_customer\_level\_inf 命令，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

11、抽取 ods 库中表 customer\_addr 最新分区的数据到 dwd 库中 dim\_customer\_addr 的分区表，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”，dwd\_insert\_time、dwd\_modify\_time 均填写当前操作时间（年月日必须是今天，时分秒只需在比赛时间范围内即可），并进行数据类型转换。使用 hive cli 执行 show partitions dwd.dim\_customer\_addr 命令，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

12、 将 dwd 库中 dim\_customer\_inf 、 dim\_customer\_addr 、 dim\_customer\_level\_inf 表的数据关联到 dws 库中 customer\_addr\_level\_aggr 的分区表,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），并添加 dws\_insert\_user、dws\_insert\_time、dws\_modify\_user、dws\_modify\_time 四列，其中 dws\_insert\_user、dws\_modify\_user 均填写“user1”，dws\_insert\_time、dws\_modify\_time 均填写当前操作时间（年月日必须是今天，时分秒只需在比赛时间范围内即可），并进行数据类型转换。使用 hive cli 统计最新分区中得数据总量，将结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下。

## 任务二：离线指标计算

1、 编写 Scala 工程代码，根据 dwd 的订单表 dwd.fact\_order\_master，求各省份下单时间为 2022 年的支付转化率，并将计算结果按照下述表结构写入 clickhouse 的 ds\_result 库的 payment\_cvr 表。在 Linux 的 clickhouse 命令行中根据 ranking 字段查询出转化率前三的省份，将 SQL 语句与执行结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

注:支付转化率 = 完成支付的订单数 / 已下单数。

payment\_cvr 表结构：

字段	类型	中文含义	备注
province	string	省份名	
creat_order	int	已下单数	

payment	int	已支付的订单数	
payCVR	float64	支付转化率	四舍五入保留三位小数
ranking	int	转化率排名	

2、编写 Scala 工程代码，根据 dwd 的 fact\_order\_master 表最新分区关联 fact\_order\_detail 表，计算所有订单中各商品所有订单（若该订单存在“已退款”状态则该订单不做计算，其余情况都参与计算）总销售金额（购买商品单价\*购买商品数量）排名，并将计算结果按照下述表结构写入 clickhouse 的 ds\_result 库的 sales\_amount\_rank 表。然后在 Linux 的 clickhouse 命令行中根据 sales\_rank 升序查询前 5 行，将 SQL 语句与执行结果截图粘贴至客户端桌面【Release\模块 C 提交结果.docx】中对应的任务序号下；

sales\_amount\_rank 表结构：

字段	类型	中文含义	备注
product_id	int	订单商品 ID	
sales_amount	float64	商品总销售金额	四舍五入保留两位小数
product_totalcnt	int	商品销售总数	
sales_rank	int	销售金额排名	

## 模块 D：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模块服务端说明。

### 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（不用考虑数据是否合规，直接聚合省份消费额），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\模块 D 提交结果.docx】中对应的任务序号下。

### 任务二：用饼状图展示各地区消费能力

编写 Vue 工程代码，根据接口，用饼状图展示 2020 年各地区的消费总额占比（不用考虑数据是否合规，直接聚合地区消费额），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\模块 D 提交结果.docx】中对应的任务序号下。

## 模块 E：综合分析（10 分）

### 任务一：Spark 中的 RDD 是如何实现容错性的？

解释 Spark 中的 RDD 是如何实现容错性的,以及在分布式计算中的具体应用场景,将内容编写至客户端桌面【Release\模块 E 提交结果.docx】中对应的任务序号下。

### 任务二：请简述 Flink 资源管理中 Task Slot 的概念。

请简述你对 Task Slot 的理解,将内容编写至客户端桌面【Release\模块 EF 提交结果.docx】中对应的任务序号下。

### 任务三：Flink 中的状态管理是如何实现的？

Flink 中的状态管理是如何实现的,具体描述在有状态的流处理中如何有效地处理和维护状态?将内容编写至客户端桌面【Release\模块 E 提交结果.docx】中对应的任务序号下。



## 附录：补充说明

### 命令行截图样例（保证清晰）

```
21/06/17 14:24:59 INFO ShuffleBlockFetcherIterator: Getting 1 non-empty blocks out of 4 blocks
21/06/17 14:24:59 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
21/06/17 14:24:59 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks out of 4 blocks
21/06/17 14:24:59 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
21/06/17 14:24:59 INFO Executor: Finished task 2.0 in stage 64.0 (TID 223). 2399 bytes result sent to driver
21/06/17 14:24:59 INFO Executor: Running task 5.0 in stage 64.0 (TID 220)
21/06/17 14:24:59 INFO Executor: Finished task 4.0 in stage 64.0 (TID 225). 2439 bytes result sent to driver
21/06/17 14:24:59 INFO TaskSetManager: Finished task 2.0 in stage 64.0 (TID 223) in 12 ms on localhost (2/6)
21/06/17 14:24:59 INFO TaskSetManager: Finished task 4.0 in stage 64.0 (TID 225) in 6 ms on localhost (3/6)
21/06/17 14:24:59 INFO Executor: Running task 3.0 in stage 64.0 (TID 224)
21/06/17 14:24:59 INFO ShuffleBlockFetcherIterator: Getting 0 non-empty blocks out of 4 blocks
21/06/17 14:24:59 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
21/06/17 14:24:59 INFO Executor: Finished task 5.0 in stage 64.0 (TID 220). 2280 bytes result sent to driver
21/06/17 14:24:59 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks out of 4 blocks
21/06/17 14:24:59 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
21/06/17 14:24:59 INFO TaskSetManager: Finished task 1.0 in stage 64.0 (TID 222) in 17 ms on localhost (4/6)
21/06/17 14:24:59 INFO Executor: Finished task 3.0 in stage 64.0 (TID 224). 2439 bytes result sent to driver
21/06/17 14:24:59 INFO TaskSetManager: Finished task 5.0 in stage 64.0 (TID 220) in 19 ms on localhost (5/6)
21/06/17 14:24:59 INFO TaskSetManager: Finished task 3.0 in stage 64.0 (TID 224) in 13 ms on localhost (6/6)
21/06/17 14:24:59 INFO TaskSchedulerImpl: Removed TaskSet 64.0, whose tasks have all completed, from pool
21/06/17 14:24:59 INFO DAGScheduler: ResultStage 64 (collect at rol11.scala:188) finished in 0.023 s
21/06/17 14:24:59 INFO DAGScheduler: Job 33 finished: collect at rol11.scala:188, took 0.028396 s
```

### 表结构说明

MySQL 数据库中已自带注释，自行连接使用工具查看。

### 若 IDEA 运行代码时候出现

scalac: No 'scala-library\*.jar' in Scala compiler classpath in Scala SDK Maven:  
org.scala-lang:scala-library:2.12.0

则需要先在 File->Project Structure->Project Settings->Libraries->添加 scala  
的包（2.12 大版本一致即可）。