## 模块A：大数据平台搭建（容器环境）

### 任务一：Hadoop HA安装部署

本环节需要使用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、分别将Master节点Hadoop，Zookeeper,JDK安装包解压到/opt/module路径中(若路径不存在，则需新建)，其中将JDK，Hadoop解压命令复制并粘贴至对应报告中;

2、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，修改容器中/etc/profile文件，设置JDK环境变量并使其生效，分发jdk至slave1，slave2中，均配置完毕后在Master节点分别执行“java -version”和“javac”命令，将命令行执行结果分别截图并粘贴至对应报告中;

3、配置好zookeeper，其中zookeeper使用集群模式，分别在master，slave1，slave2作为其集群的节点，使用zkServer.sh status获取zookeeper服务端状态，将命令和结果截图粘贴至对应报告中（注：只截取三个节点中zookeeper server角色模式为leader的节点）；

4、配置好Hadoop HA，请将dfs.ha.namenodes.hadoopcluster设置为nn1,nn2，同时yarn.resourcemanager.ha.rm-ids设置为rm1,rm2，并在master启动nn1与rm1，在salve1启动nn2与rm2，将master，slave1，slave2均作为datanode，分发hadoop至slave1，slave2中。启动yarn与hdfs的HA集群（Hadoop HA集群），并在mster节点上使用命令分别查看服务nn2与rm2进程状态，并将查看命令及结果截图粘贴至对应报告中；

5、Hadoop HA配置并启动完毕后，使用jps在slave1节点查看服务进程，将查看命令及结果截图粘贴至对应报告中。

### 任务二：Kafka安装配置

本环节需要使用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. 将Master节点Zookeeper，kafka安装包解压到/opt/module目录下，将kafka解压命令复制并粘贴至对应报告中；
2. 配置好zookeeper，其中zookeeper使用集群模式，分别在master，slave1，slave2作为其节点（若zookpeer已安装配置好，则无需再次配置），配置好kafka的环境变量，使用kafka-server-start.sh --version查看kafka的版本内容，并将命令和结果截图粘贴至对应报告中；
3. 完善其他配置并分发kafka文件到slave1,slave2中，并在每个节点启动Kafka，创建topic，其中toipic名称为installtopic，分区数为2，副本数为2，将创建命令和创建成果截图粘贴到对应报告中；

### 任务三：Hive安装配置

本环节需要使用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. 将Master节点Hive安装包解压到/opt/module目录下，将命令复制并粘贴至对应报告中；
2. 设置Hive环境变量，并使环境变量生效，执行命令hive --version并将命令与结果截图容复制并粘贴至对应报告中；

3、完成相关配置并添加所依赖包，将MySQL数据库作为Hive元数据库。初始化Hive元数据，并通过schematool相关命令执行初始化，将初始化结果截图（范围为命令执行结束的最后10行）复制粘贴至对应报告中。

### 任务四：Spark on Yarn安装配置

本环节需要使用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、将Spark包解压到路径/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对应报告中;；

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对应报告中;

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对应报告中（截取Pi结果的前后各5行）。

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

### 任务五：Flume安装配置

本环节需要使用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. 将Master节点Flume安装包解压到/opt/module目录下，将解压命令复制并粘贴至对应报告中；
2. 完善相关配置设置，Hive环境变量，并使环境变量生效，执行命令flume-ng version并将命令与结果截图粘贴至对应报告中；
3. 启动Flume传输Hadoop日志（namenode或datanode日志），查看HDFS中/tmp/flume目录下生成的内容，将查看命令及结果（至少5条结果）截图并粘贴至对应报告中。

## 模块B：离线数据处理

### 任务一：数据抽取

使用Scala编写spark工程代码，将MySQL的ds\_db01库中表customer\_inf、customer\_inf、order\_detail、order\_master、product\_info的数据增量抽取到Hive的ods库（需自建）中对应表customer\_inf、order\_detail、order\_master、product\_info中。

1. 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.user\_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，且值为当前日期的前一天日期（分区字段格式为yyyyMMdd）。使用hive cli执行show partitions ods.customer\_inf命令；
2. 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，且值为当前日期的前一天日期（分区字段格式为yyyyMMdd）。使用hive cli执行show partitions ods.product\_info命令；
3. 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，且值为当前日期的前一天日期（分区字段格式为yyyyMMdd）。使用hive cli执行show partitions ods.order\_master命令；
4. 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。

### 任务二：数据清洗

**提示**：若完成了实时计算的任务二中的第3小题后，可以将HBase中备份的数据与ods中的离线数据进行合并，抽取到dwd（需自建）的对应表中。

编写Hive SQL或者Spark Sql代码，将ods库中相应表数据（经过数据抽取得数据）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，若与日期有关的数据，必须转为timestamp，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customer\_inf表中昨天的分区（任务一生成的分区）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执行show partitions dwd.dim\_customer\_inf命令；
2. 抽取ods库中product\_info表中昨天的分区（任务一生成的分区）数据，并结合dim\_product\_info最新分区现有的数据，根据product\_id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\_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\_product\_info命令；
3. 将ods库中order\_master表昨天的分区（任务一生成的分区）数据抽取到dwd库中fact\_order\_master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\_modify\_time均填写当前操作时间，并进行数据类型转换，需要过滤掉city字段长度大于8。使用hive cli执行show partitions dwd.fact\_order\_info命令；

**提示**：若完成实时任务二中的第3小题后，可将hbase中的数据合并后在进行清洗，清洗后数据添加至DWD中

1. 将ods库中order\_detail表昨天的分区（任务一生成的分区）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\_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\_detail命令；

**提示**：若完成实时任务二中的第3小题后，可将hbase中的数据合并后在进行清洗，清洗后数据添加至DWD中

### 任务三：指标计算

**注：**与订单金额计算相关使用order\_money字段，同一个订单无需多次重复计算，需要考虑退款或者取消的订单

1. 根据dwd或者dws层表统计每人每天下单的数量和下单的总金额，存入dws层（需自建）的user\_consumption\_day\_aggr表中（表结构如下），然后使用hive cli按照客户主键、订单总金额均为降序排序，查询出前5条；

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| customer\_id | int | 客户主键 | customer\_id |
| customer\_name | string | 客户名称 | customer\_name |
| total\_amount | double | 订单总金额 | 当天订单总金额 |
| total\_count | int | 订单总数 | 当天订单总数 |
| year | int | 年 | 订单产生的年,为动态分区字段 |
| month | int | 月 | 订单产生的月,为动态分区字段 |
| day | int | 日 | 订单产生的日,为动态分区字段 |

2、根据dwd或者dws层表统计每个城市每月下单的数量和下单的总金额（以order\_master中的地址为判断依据），并按照province\_name，year，month进行分组,按照total\_amount逆序排序，形成sequence值，将计算结果存入Hive的dws数据库city\_consumption\_day\_aggr表中（表结构如下），然后使用hive cli根据订单总数、订单总金额均为降序排序，查询出前5条，在查询时对于订单总金额字段将其转为bigint类型（避免用科学计数法展示）；

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| city\_name | string | 城市名称 |  |
| province\_name | string | 省份名称 |  |
| total\_amount | double | 订单总金额 | 当月订单总金额 |
| total\_count | int | 订单总数 | 当月订单总数 |
| sequence | int | 次序 | 即当月中该城市消费额在该省中的排名 |
| year | int | 年 | 订单产生的年,为动态分区字段 |
| month | int | 月 | 订单产生的月,为动态分区字段 |

3、请根据dwd或者dws层表计算出每个城市每个月平均订单金额和该城市所在省份平均订单金额相比较结果（“高/低/相同”）,存入ClickHouse数据库shtd\_result的cityavgcmpprovince表中（表结构如下），然后在Linux的ClickHouse命令行中根据城市平均订单金额、省份平均订单金额均为降序排序，查询出前5条；

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| cityname | text | 城市份名称 |  |
| cityavgconsumption | double | 该城市平均订单金额 |  |
| provincename | text | 省份名称 |  |
| provinceavgconsumption | double | 该省平均订单金额 |  |
| comparison | text | 比较结果 | 城市平均订单金额和该省平均订单金额比较结果，值为：高/低/相同 |

4、请根据dwd或者dws层表计算出每个城市每个月平均订单金额和该城市所在省份订单金额中位数相比较结果（“高/低/相同”）,存入ClickHouse数据库shtd\_result的citymidcmpprovince表中（表结构如下），然后在Linux的ClickHouse命令行中根据城市平均订单金额、省份平均订单金额均为降序排序，查询出前5条；

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| cityname | text | 城市份名称 |  |
| citymidconsumption | double | 该城市订单金额中位数 |  |
| provincename | text | 省份名称 |  |
| provincemidconsumption | double | 该省订单金额中位数 |  |
| comparison | text | 比较结果 | 城市订单金额中位数和该省订单金额中位数比较结果，值为：高/低/相同 |

5、请根据dwd或者dws层表来计算每个省份2022年订单金额前3省份，依次存入ClickHouse数据库shtd\_result的regiontopthree表中（表结构如下），然后在Linux的ClickHouse命令行中根据省份升序排序，查询出前5条；

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| provincename | text | 省份名称 |  |
| citynames | text | 城市名称 | 用,分割显示前三城市的name |
| cityamount | text | 省份名称 | 用,分割显示前三城市的订单金额（需要去除小数部分，使用四舍五入） |

例如：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | 华东地区 | 21,27,11 | 上海市,江苏省,浙江省 | 100000,100,10 |

6、请根据dwd或者dws层的相关表，计算销售量前10的商品，销售额前10的商品，存入ClickHouse数据库shtd\_result的topten表中（表结构如下），然后在Linux的ClickHouse命令行中根据排名升序排序，查询出前5条;

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| topquantityid | int | 商品id | 销售量前10的商品 |
| topquantityname | text | 商品名称 | 销售量前10的商品 |
| topquantity | int | 该商品销售量 | 销售量前10的商品 |
| toppriceid | text | 商品id | 销售额前10的商品 |
| toppricename | text | 商品名称 | 销售额前10的商品 |
| topprice | decimal | 该商品销售额 | 销售额前10的商品 |
| sequence | int | 排名 | 所属排名 |

7、请根据dwd或者dws层的数据，请计算连续两天下单的用户与已下单用户的占比，将结果存入ClickHouse数据库shtd\_result的userrepurchasedrate表中(表结构如下)，然后在Linux的ClickHouse命令行中查询结果数据；

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| purchaseduser | int | 下单人数 | 已下单人数 |
| repurchaseduser | int | 连续下单人数 | 连续两天下单的人数 |
| repurchaserate | text | 百占比 | 连续两天下单人数/已下单人数百分比（保留1位小数，四舍五入，不足的补0）例如21.1%，或者32.0% |

8、根据dwd或者dws层的数据，请计算每个省份累计订单量，然后根据每个省份订单量从高到低排列，将结果打印到控制台（使用spark中的show算子，同时需要显示列名）；

例如：可以考虑首先生成类似的临时表A：

|  |  |
| --- | --- |
| province\_name | Amount（订单量） |
| A省 | 10122 |
| B省 | 301 |
| C省 | 2333333 |

然后生成结果类似如下：其中C省销量最高，排在第一列，A省次之，以此类推。

|  |  |  |
| --- | --- | --- |
| C省 | A省 | B省 |
| 23333331 | 10122 | 301 |

9、根据dwd或者dws层的相关表，请计算2022年4月26日凌晨0点0分0秒到早上9点59分59秒为止，该时间段每小时的新增订单金额与当天订单总金额累加值，存入ClickHouse数据库shtd\_result的accumulateconsumption表中，然后在Linux的ClickHouse命令行中根据订单时间段升序排序，查询出前5条;

假如数据为：

|  |  |  |
| --- | --- | --- |
| 用户 | 订单时间 | 订单金额 |
| 张三1号 | 2020-04-26 00:00:10 | 10 |

|  |  |  |
| --- | --- | --- |
| 李四1号 | 2020-04-26 00:20:10 | 5 |
| 李四2号 | 2020-04-26 01:21:10 | 10 |
| 王五1号 | 2020-04-26 03:20:10 | 50 |

计算结果则为：

|  |  |  |
| --- | --- | --- |
| 订单时间段 | 订单新增金额 | 累加总金额 |
| 2020-04-26 00 | 15 | 15 |
| 2020-04-26 01 | 10 | 25 |

|  |  |  |
| --- | --- | --- |
| 2020-04-26 02 | 0 | 25 |

|  |  |  |
| --- | --- | --- |
| 2020-04-26 03 | 50 | 75 |

accumulateconsumption表结构如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| consumptiontime | varchar | 消费时间段 |  |
| consumptionadd | double | 订单新增金额 |  |
| consumptionacc | double | 累加总金额 |  |

10、根据dwd层或dws层的相关表，请计算2022年4月26日凌晨0点0分0秒到早上9点59分59秒为止的数据，以5个小时为时间窗口，滑动的步长为1小时，做滑动窗口计算该窗口内订单总金额和订单总量，时间不满5小时不触发计算（即从凌晨5点0分0秒开始触发计算），存入ClickHouse数据库shtd\_result的slidewindowconsumption表中，然后在Linux的ClickHouse命令行中根据订单时间段升序排序，查询出前5条，将核心业务代码中的开窗相关代码与MySQL查询结果展示出来。

假如数据为：

|  |  |  |
| --- | --- | --- |
| 用户 | 订单时间 | 订单金额 |
| 张三1号 | 2020-04-26 00:00:10 | 10 |

|  |  |  |
| --- | --- | --- |
| 李四1号 | 2020-04-26 00:20:10 | 25 |
| 李四2号 | 2020-04-26 01:21:10 | 10 |
| 李四2号 | 2020-04-26 02:21:10 | 5 |
| 王五1号 | 2020-04-26 03:20:10 | 20 |
| 李四2号 | 2020-04-26 04:20:10 | 10 |
| 王五2号 | 2020-04-26 05:10:10 | 10 |
| 李四2号 | 2020-04-26 06:20:10 | 10 |
| 赵六2号 | 2020-04-26 07:10:10 | 10 |
| 赵六2号 | 2020-04-26 08:10:10 | 10 |
| 王五2号 | 2020-04-26 09:11:10 | 10 |
| 王五4号 | 2020-04-26 09:32:10 | 30 |

计算结果则为：

|  |  |  |  |
| --- | --- | --- | --- |
| 订单时间段 | 该窗口内订单金额 | 订单总量 | 平均每单价格 |
| 2020-04-26 04 | 80 | 6 | 13.33 |
| 2020-04-26 05 | 55 | 5 | 11 |
| 2020-04-26 06 | 55 | 5 | 11 |
| 2020-04-26 07 | 60 | 5 | 12 |
| 2020-04-26 08 | 50 | 5 | 10 |
| 2020-04-26 09 | 80 | 6 | 13.33 |

slidewindowconsumption表结构如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| consumptiontime | varchar | 订单时间段 |  |
| consumptionsum | double | 该窗口内的订单总金额 |  |
| consumptioncount | double | 订单总数量 |  |
| consumptionavg | double | 平均每单价格 | 上面两个字段相除，四舍五入保留两位小数 |

## 模块C：数据挖掘

### 任务一：特征工程（一）

剔除订单主表与订单详情表中用户id与商品id不存在现有的维表中的记录，同时建议多利用缓存并充分考虑并行度来优化代码，达到更快的计算效果。

1. 根据Hive的dwd库中相关表或MySQL中ds\_db01中相关表，计算出与用户customer\_id为5811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customer\_id进行输出，输出格式如下；

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

1. 根据Hive的dwd库中相关表或MySQL中ds\_db01中相关商品表（sku\_info），获取id、brand\_id、price、weight、height、length、width、three\_category\_id 字段并进行数据预处理，对数值类型进行规范化(StandardScaler)处理，对类别类型进行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）。

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| id | double | 主键 |  |
| price | double | 价格 |  |
| weight | double | 重量 |  |
| 。。。 | 。。。 | 。。。 |  |
| brand\_id#1 | double | brand\_id 1 | 若属于该品牌，则内容为1否则为0 |
| brand\_id#2 | double | brand\_id 2 | 若属于该品牌，则内容为1否则为0 |
| ..... | double |  |  |

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

### 任务三：特征工程（二）

1. 据Hive的dwd库中相关表或MySQL数据库ds\_db01中订单相关表，对用户购买过的商品进行去重，将其转换为以下格式：第一列为用户id mapping，第二列为用户购买过的商品id mapping，按照customer\_id与product\_id进行升序排序，输出前5行；

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| customer\_id | int | 用户id的mapping对应键 |  |
| product\_id | int | 商品id的mapping对应键 |  |

提示：

Mapping操作：例如用户id：1、4、7、8、9，则做完mapping操作转为字典类型，键0对应用户id 1，键1对应用户id 4，以此类推

结果格式如下：

------- customer\_id\_mapping与product\_id\_mapping数据前5条如下：-------

0:0

0:89

1:1

1:2

1:3

1. 根据第1小题的结果，对其进行聚合，其中对product\_id进行one-hot转换，将其转换为以下格式矩阵：第一列为用户id，其余列名为商品id，按照用户id进行升序排序，展示矩阵第一行前5列数据。

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| customer\_id | double | 客户id |  |
| product\_id0 | double | 用户是否购买过商品1 | 若用户购买过该商品，则值为1，否则为0 |
| product\_id1 | double | 用户是否购买过商品2 | 若用户购买过该商品，则值为1，否则为0 |
| product\_id2 | double | 用户是否购买过商品3 | 若用户购买过该商品，则值为1，否则为0 |
| ..... |  |  |  |

结果格式如下：

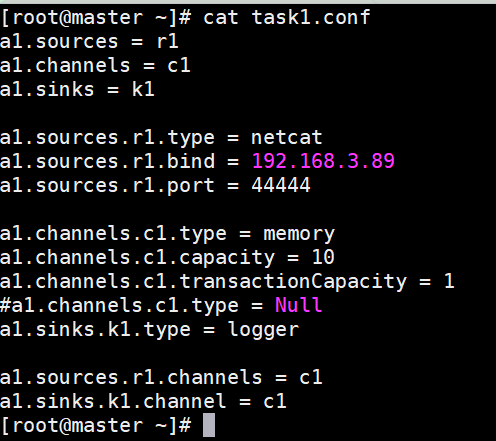
---------------第一行前5列结果展示为---------------

0.0,1.0,0.0,0.0,0.0

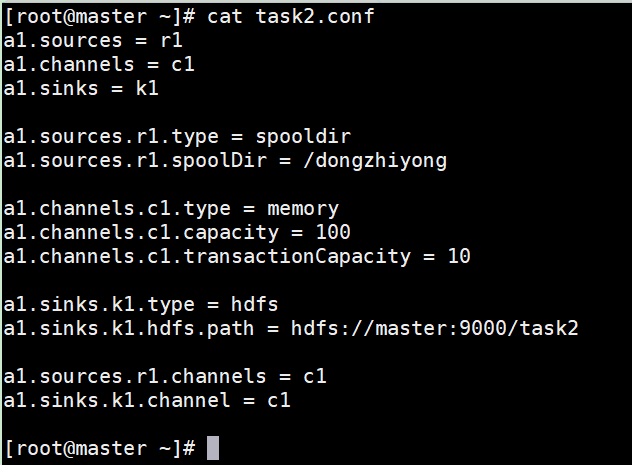
## 模块D：数据采集与实时计算

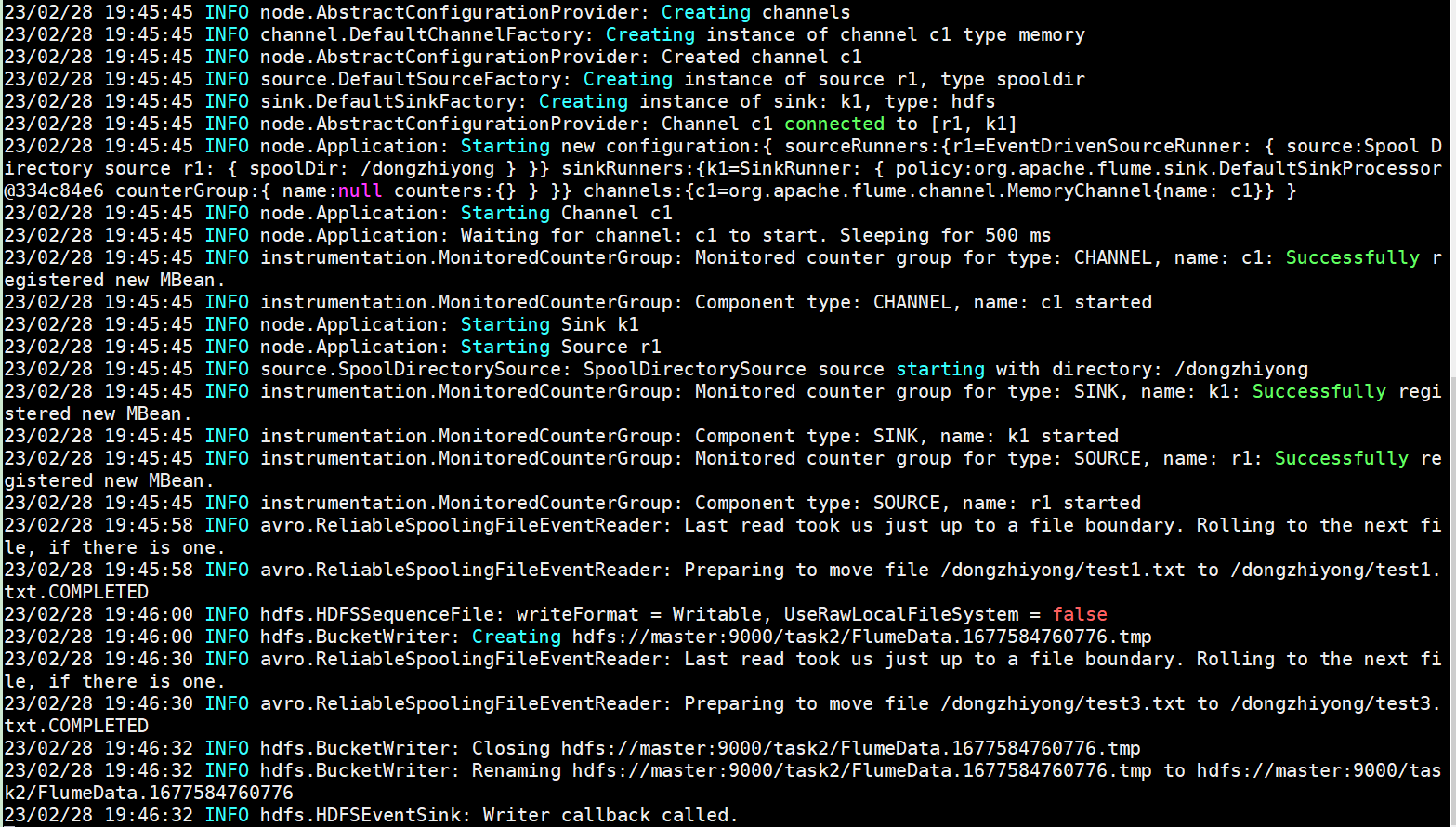
### 任务一：实时数据采集

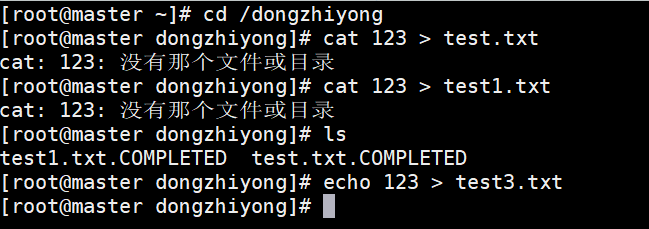
1. 在Master节点上使用telent命令发送网络数据，通过flume采集telnet发送的数据，并在控制台显示接收到的数据。将Flume的配置截图和运行截图粘贴至对应报告中

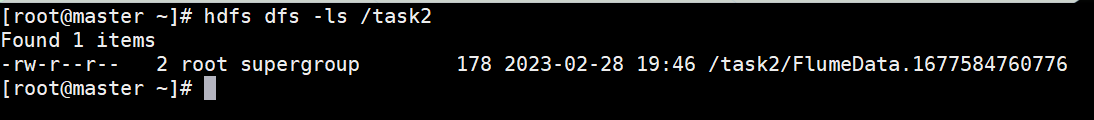


1. 使用flume监听特定目录/dongzhiyong，当该目录下产生新的文件，则将文件上传到hdfs中。将Flume的配置截图和运行截图粘贴至对应报告中。







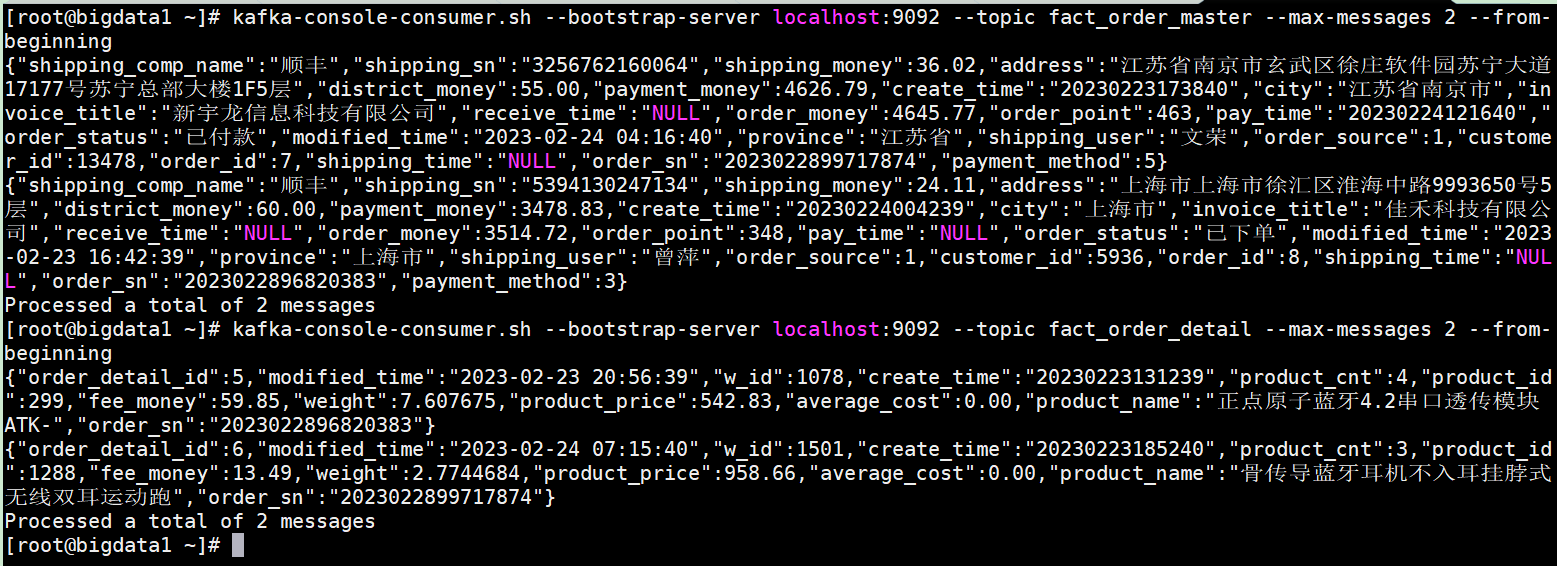


### 任务二：使用Flink处理Kafka中的数据

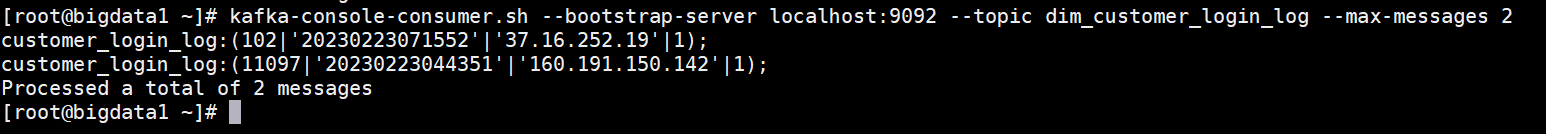
**注：**与订单金额计算相关使用order\_money字段，同一个订单无需多次重复计算，需要考虑退款或者取消的订单。

编写Scala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ProcessTime）。

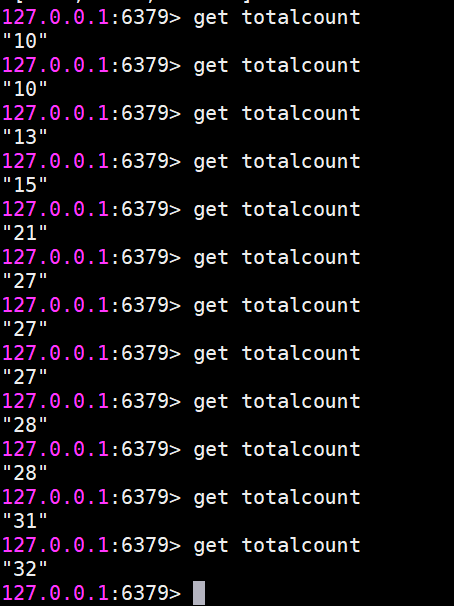
1. 使用Flink消费Kafka中topic为ods\_mall\_data的数据，根据数据中不同的表将数据分别分发至kafka的DWD层的fact\_order\_master、fact\_order\_detail的Topic中（只获取data的内容，具体的内容格式考生请自查），其他的表则无需处理，其中请对进入到fact\_order\_master的内容进行排序，同时迟到容忍度设置为2分钟；



1. 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，将数据分别分发至kafka的DWD层的dim\_customer\_login\_log的Topic中，其他的表则无需处理；



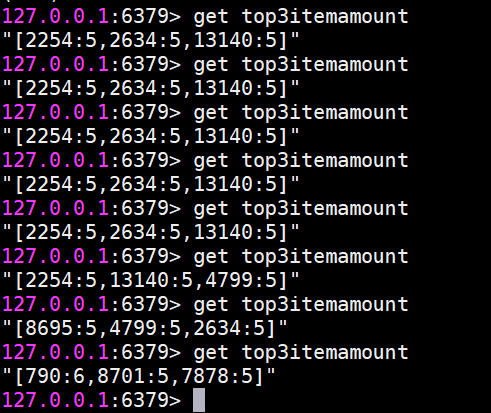
1. 在任务1，2进行的同时，需要将order\_master、order\_detail、customer\_login\_log备份至Hbase中，同时建立Hive外表(表结构与离线数据ods层一致，timestamp格式可用string或者long代替，rowkey使用随机数（0-9）+yyyyMMddHHmmssSSS，其中对于customer\_login\_log缺失主键，请用随机数（0-9）+用户id+登陆时间代替)，同时在Hive中查询每个表前5条数据；
2. 使用Flink消费Kafka的dwd层数据，统计商城实时订单数量，将key设置成totalcount存入Redis中。使用redis cli以get key方式获取totalcount值；



1. 使用Flink消费Kafka的dwd层数据，实时统计商城中销售量前3的商品，将key设置成top3itemamount存入Redis中（value使用String数据格式，value为前3的商品信息并且外层用[]包裹，其中按排序依次存放商品id:销售量，并用逗号分割）。使用redis cli以get key方式获取top3itemamount值；

示例如下：

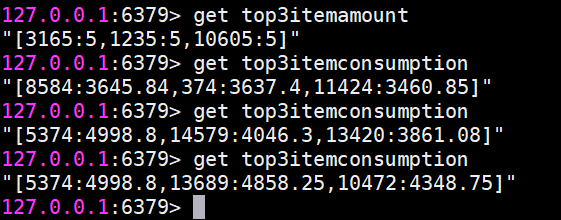
top3itemamount：[1:700,42:500,41:100]



1. 使用Flink消费Kafka的dwd层数据，实时统计商城中销售额前3的商品（销售额为单价\*数量），将key设置成top3itemconsumption存入Redis中（value使用String数据格式，value为前3的商品信息并且外层用[]包裹，其中按排序依次存放商品id:销售额，并用逗号分割）。使用redis cli以get key方式获取top3itemconsumption值。

示例如下：

top3itemconsumption：[1:10020.2,42:4540.0,12:540]



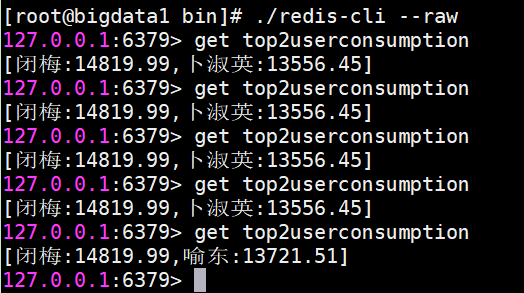
1. 使用Flink消费Kafka的dwd层数据，监控order\_status字段为已退款的数据,将数据存入ClickHouse数据库shtd\_result的order\_master表中，然后在Linux的ClickHouse命令行中根查询出前5条。



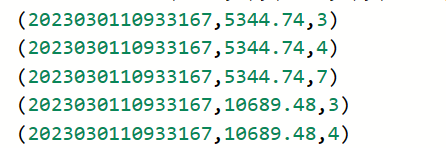
1. 使用Flink消费Kafka中的数据，实时统计商城中消费额前2的用户（，将key设置成top2userconsumption存入Redis中（value使用String数据格式，value为前2的用户信息并且外层用[]包裹，其中按排序依次存放为该用户id:用户名称:消费总额，用逗号分割，其中用户名称可从HBase或者Hive或者Kafka中的客户表中获取）。使用redis cli以get key方式获取top2userconsumption值；

示例如下：

top2userconsumption：[1:张三:10020,42:李四:4540]



1. 采用双流JOIN的方式（本系统稳定，无需担心数据迟到与丢失的问题,建议使用滚动窗口），结合订单信息和订单详细信息，拼接成如下表所示格式，其中包含订单id、订单总金额、商品数，将数据存入ClickHouse数据库shtd\_result的orderpostiveaggr表中（表结构如下），然后在Linux的ClickHouse命令行中根据sn降序排序，查询出前5条。



|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| sn | int | 订单sn |  |
| orderprice | double | 订单总金额 | 统计订单主表中 order\_money字段 |
| orderdetailcount | int | 商品数量总计 | 累加订单详细信息中 product\_cnt字段 |

数据示例：

Ordermaster:

{"order\_id":16845,"order\_sn":"2023021196454794","customer\_id":9969,"shipping\_user":"申楠","province":"江苏省","city":"江苏省常州市","address":"江苏省常州市新北区通江中路883567号一层L1001商铺23层","order\_source":1,"payment\_method":5,"order\_money":10909.54,"district\_money":85.00,"shipping\_money":35.47,"payment\_money":10860.01,"shipping\_comp\_name":"顺丰","shipping\_sn":"2799925324640","create\_time":"20230206043446","shipping\_time":"NULL","pay\_time":"NULL","receive\_time":"NULL","order\_status":"已下单","order\_point":1086,"invoice\_title":"飞利信网络有限公司","modified\_time":"2023-02-05 20:34:46"}

Orderdetail:

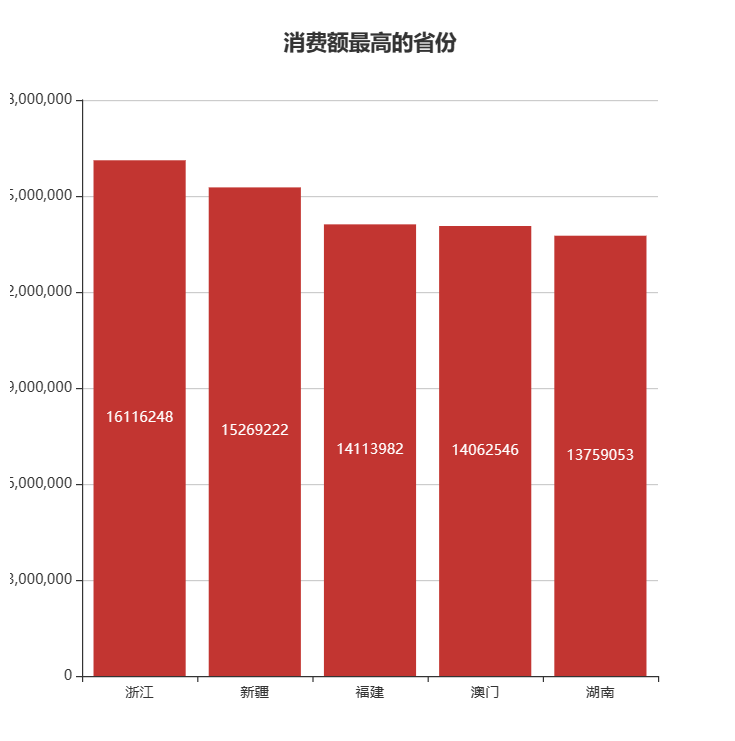
{"order\_detail\_id":31060,"order\_sn":"2023021179437449","product\_id":12201,"product\_name":"蓝牙耳机无线双耳vivo适用苹果入耳式5.0","product\_cnt":1,"product\_price":962.37,"average\_cost":0.00,"weight":2.1962569,"fee\_money":89.39,"w\_id":419,"create\_time":"20230206031734","modified\_time":"2023-02-06 03:32:34"}

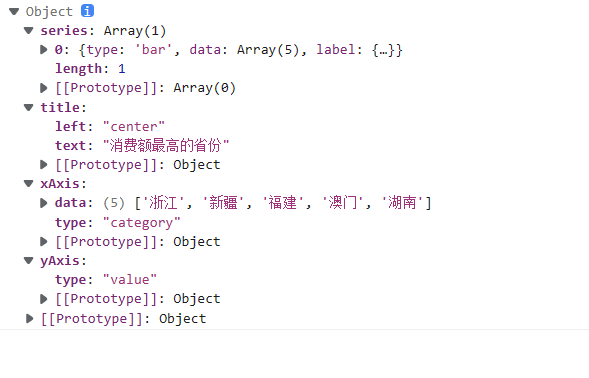
## 模块E：数据可视化

### 任务一、电商可视化项目

### 1、用柱状图展示消费额最高的省份

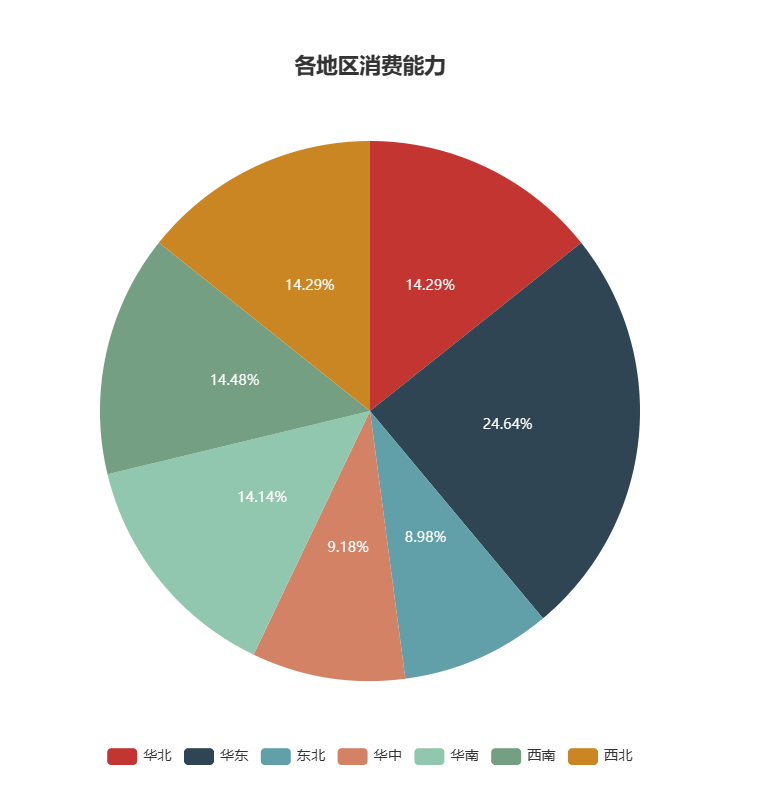
编写Vue工程代码，根据接口，用柱状图展示2020年消费额最高的5个省份，同时将用于图表展示的数据结构在浏览器的console中进行打印输出，将图表可视化结果和浏览器console打印结果分别截图并粘贴至对应报告中。

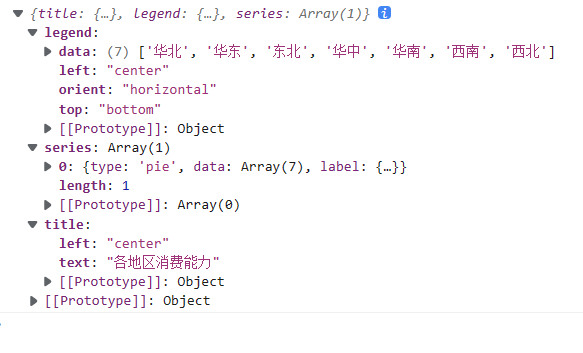




### 2、用饼状图展示各地区消费能力

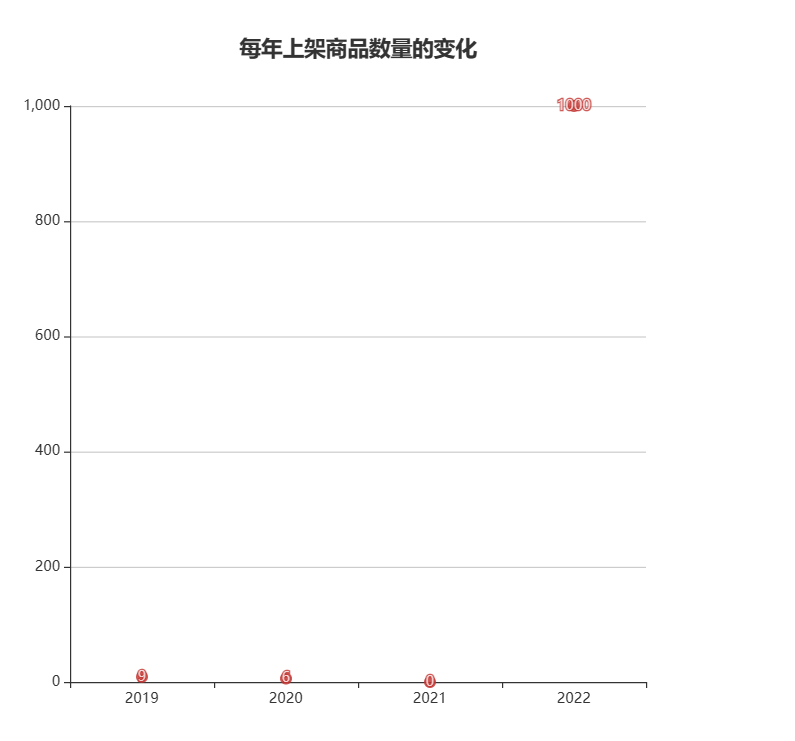
编写Vue工程代码，根据接口，用饼状图展示2020年各地区的消费总额占比，同时将用于图表展示的数据结构在浏览器的console中进行打印输出，将图表可视化结果和浏览器console打印结果分别截图并粘贴至对应报告中。

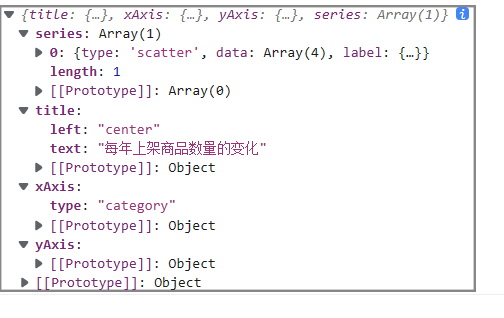




### 3、用散点图展示每年上架商品数量的变化

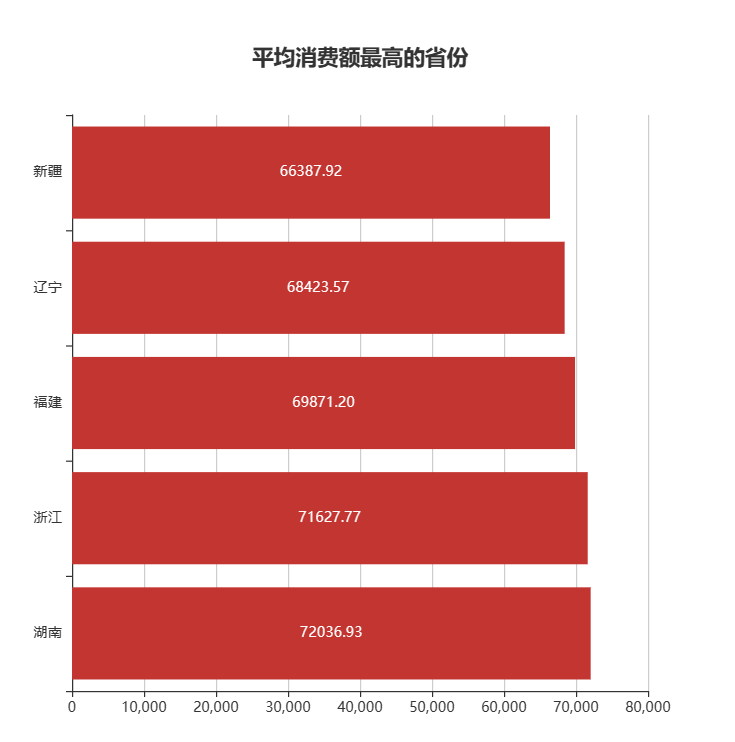
编写Vue工程代码，根据接口，用基础散点图展示每年上架商品数量的变化情况，同时将用于图表展示的数据结构在浏览器的console中进行打印输出，将图表可视化结果和浏览器console打印结果分别截图并粘贴至对应报告中。

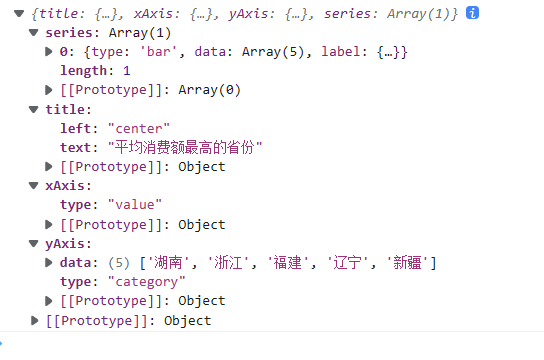




### 4、用条形图展示平均消费额最高的省份

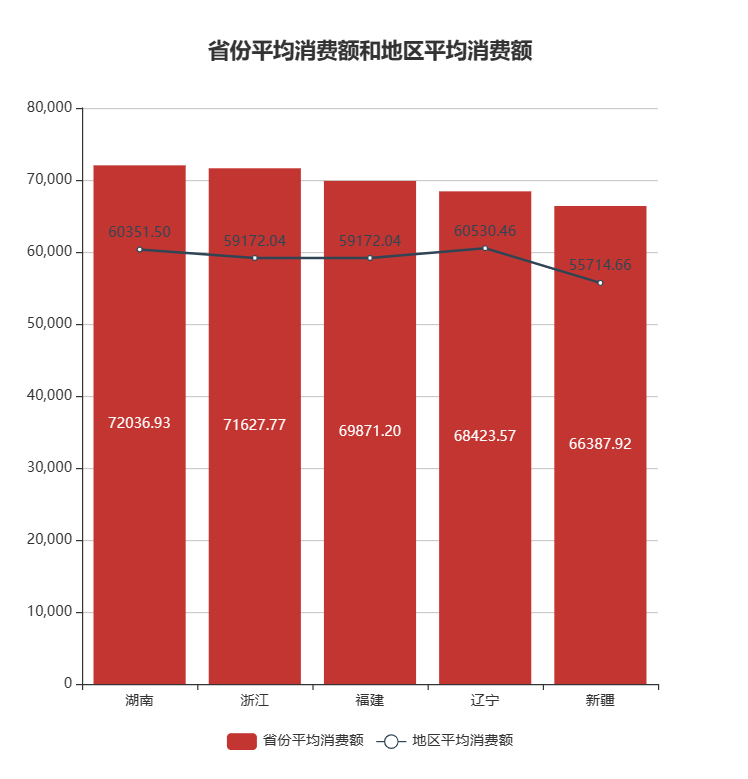
编写Vue工程代码，根据接口，用条形图展示2020年平均消费额（四舍五入保留两位小数）最高的5个省份，同时将用于图表展示的数据结构在浏览器的console中进行打印输出，将图表可视化结果和浏览器console打印结果分别截图并粘贴至对应报告中。





### 5、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对应报告中。





### 任务二、订单可视化项目

在mysql数据库中建立表order，字段如下：

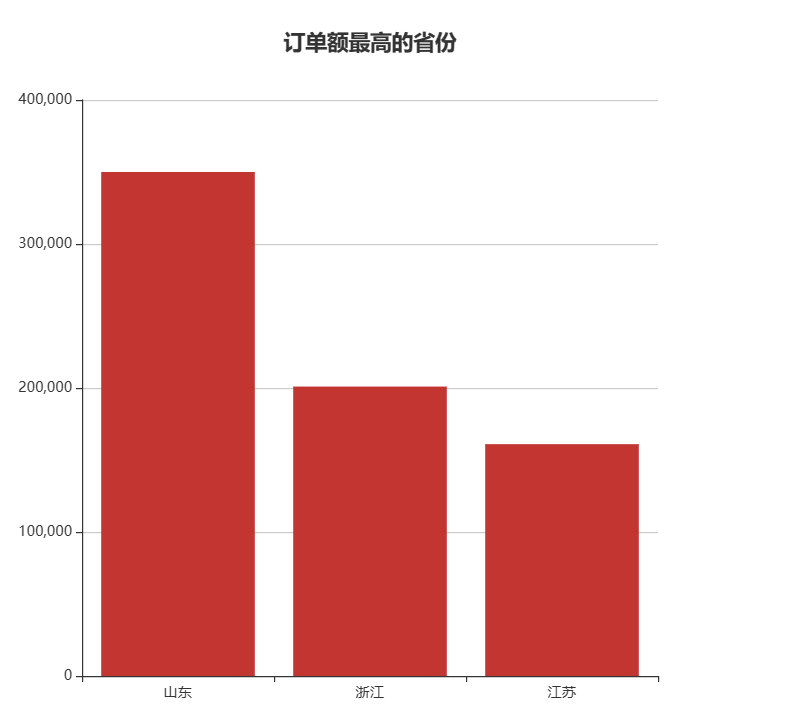
|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 中文含义 | 备注 |
| provincename | text | 省份名称 |  |
| regionname | text | 地区名称 |  |
| totalconsumption | double | 订单总金额 | 当月订单总金额 |
| totalorder | int | 订单总数 | 当月订单总数 |
| year | int | 年 | 订单产生的年 |
| month | int | 月 | 订单产生的月 |

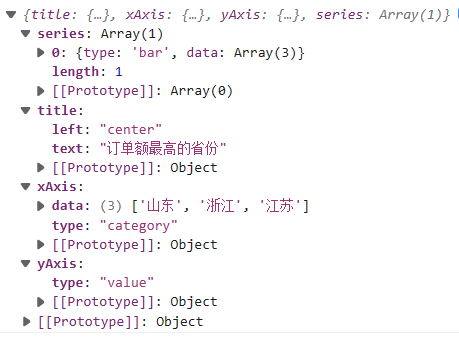
Order表的数据如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| provincename | regionname | totalconsumption | totalorder | year | month |
| 江苏 | 南京 | 10000 | 30 | 2020 | 1 |
| 江苏 | 南京 | 20000 | 40 | 2020 | 2 |
| 江苏 | 南京 | 30000 | 50 | 2020 | 3 |
| 江苏 | 南京 | 40000 | 60 | 2020 | 4 |
| 江苏 | 南京 | 50000 | 70 | 2020 | 5 |
| 江苏 | 常州 | 5000 | 25 | 2020 | 1 |
| 江苏 | 常州 | 6000 | 35 | 2020 | 2 |
| 安徽 | 合肥 | 7000 | 50 | 2020 | 1 |
| 安徽 | 合肥 | 8000 | 30 | 2020 | 2 |
| 安徽 | 芜湖 | 9500 | 40 | 2020 | 1 |
| 安徽 | 芜湖 | 10500 | 50 | 2020 | 2 |
| 浙江 | 杭州 | 55000 | 80 | 2020 | 3 |
| 浙江 | 杭州 | 66000 | 85 | 2020 | 4 |
| 浙江 | 绍兴 | 45000 | 40 | 2020 | 3 |
| 浙江 | 绍兴 | 35000 | 35 | 2020 | 4 |
| 山东 | 青岛 | 80000 | 90 | 2020 | 4 |
| 山东 | 青岛 | 85000 | 92 | 2020 | 5 |
| 山东 | 济南 | 90000 | 80 | 2020 | 4 |
| 山东 | 青岛 | 95000 | 85 | 2020 | 5 |

### 1、用柱状图展示订单额最高的省份

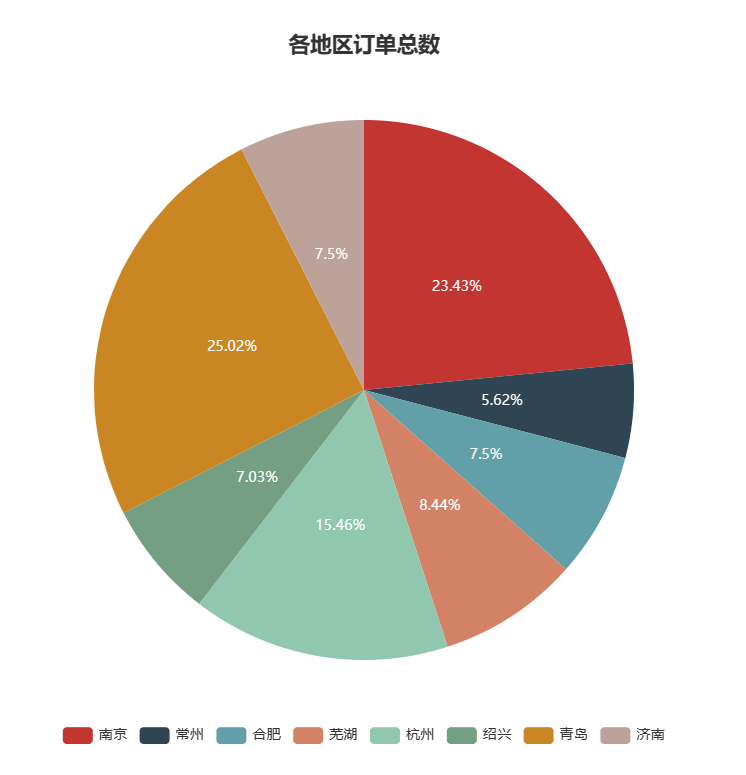
编写Vue工程代码，根据接口，用柱状图展示2020年全年订单额最高的3个省份，同时将用于图表展示的数据结构在浏览器的console中进行打印输出，将图表可视化结果和浏览器console打印结果分别截图并粘贴至对应报告中。





### 2、用饼状图展示各地区订单总数

编写Vue工程代码，根据接口，用饼状图展示2020年全年各省份的订单总数占比，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对应报告中。





## 模块F：综合分析

### 任务一：对于Hadoop HA的理解

什么是高可靠性（HA）？Hadoop HA的搭建和Hadoop 全分布式的搭建有什么区别？Hadoop HA是如何体现高可靠性的？

### 任务二、Hive和MySQL有什么区别？

Hive和MySQL都可以对结构化数据进行处理，两者在使用场景上有什么区别？

### 任务三、离线计算和实时计算有什么区别？

什么是离线计算？什么是实时计算？为什么说Hadoop是离线计算的工具，而spark和flink是实时计算的工具？实时计算和离线计算的区别是什么？

### 任务四、对于数据库仓库DW的理解

什么是数据仓库？数据仓库的应用场景是什么？数据仓库通常需要要对数据进行分层处理，这样做的优点是什么？数据明细层DWD（Data Warehouse Detail）和数据服务层DWS（Data WareHouse Servce）的作用是什么？

### 任务五、对Hive的静态分区和动态分区的理解

Hive分区的目的是什么？静态分区和动态分区有什么区别？

### 任务六、对Hadaoop组件的理解

Hadoop NameNode和DataNodes的主要功能是什么？两者有什么区别？

Hadoop 全分布式模式和伪分布式模式有什么区别？请说明它们的应用场景。