# 问题描述

**需求：**订单服务器产生数据，要求统计每1分钟的交易量

1、统计总的交易额

2、审计前面的金额。

**思路：**

0.产生交易数据

0.1 使用SheetGeneratorServer类产生交易数据，每1s产生一个记录；

0.2 产生的数据使用log4j写入到日志文件中，按照每128MB产生一个文件；

0.3 产生的数据同时写入到kafka中，topic自定义；

1.通过实时进行统计汇总，这个与前面做的内容思路类似；

1.1 使用storm消费kafka中的数据，代码单独建立一个工程去实现；

1.2 在redis中存储的是实时查询的数据；

1.3 在MySQL中存储每一分钟的数据；

create table trade\_minutes(

id bigint primary key auto\_increment,

value decimal(10,2), --表示每分钟的交易额

logtime datetime --表示每分钟的时间戳

);

2.通过离线进行统计汇总，与实时的数据做对照；

2.1 使用Flume NG监控日志文件中的数据变化，实时写入到HDFS中，HDFS的路径自定义；

2.2 创建hive的外部表管理导入的数据；

3.审计

3.1 使用java代码写一个方法

//形参是201612070949格式

boolean audit(String minute){

//1.查询MySQL

//2.查询Hive的时候，范围是[20161207094900, 20161207095000)

//3.判断两个结果是否相等

}

# 集群启动

## 关闭防火墙

service iptables stop

## Redis

redis-server /opt/redis/etc/redis.conf

## Mysql

service mysqld start

## Zookeeper

在每个节点上执行bin/zkServer.sh start

启动之后，一定要查看文件zookeeper.out。

并且使用ps -ef |grep zookeeper查看进程是否存在

## Kafka

### 启动命令

nohup bin/kafka-server-start.sh config/server.properties>nohup.out 2>&1 &

### 命令行操作

创建主题

bin/kafka-topics.sh --create --zookeeper 192.168.1.100:2181,192.168.1.101:2181,192.168.1.102:2181 --replication-factor 1 --partitions 1 --topic test

描述主题

bin/kafka-topics.sh --describe --zookeeper 192.168.1.100:2181,192.168.1.101:2181,192.168.1.102:2181 --topic test

删除主题

bin/kafka-topics.sh --delete --zookeeper 192.168.1.100:2181,192.168.1.101:2181,192.168.1.102:2181 --topic test

生产者

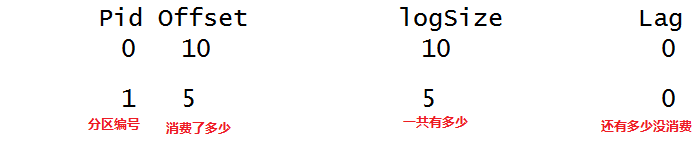
bin/kafka-console-producer.sh --broker-list 192.168.1.100:9092,192.168.1.101:9092,192.168.1.02:9092 --topic test

消费者

bin/kafka-console-consumer.sh --zookeeper 192.168.1.100:2181,192.168.1.101:2181,192.168.1.102:2181 --topic test --from-beginning

查看消费进度

bin/kafka-run-class.sh kafka.tools.ConsumerOffsetChecker --zookeeper 192.168.1.100:2181 --topic test -group group1



## Storm

### 启动命令

启动nimbus

nohup bin/storm nimbus >storm.out 2>&1 &

启动supervisor

nohup bin/storm supervisor >storm.out 2>&1 &

启动ui

nohup bin/storm ui >storm.out 2>&1 &

### 命令行操作

提交代码到storm集群运行。

把代码打成jar包，执行storm jar xxxx.jar xxxxxTopology

## Hdfs

sbin/start-dfs.sh

## Hive

在hive中运行以下代码，打开对jdbc的监听

nohup hive --service hiveserver -p 10000 >/dev/null 2>/dev/null &

外部表

create external table trade(time string, ordernum string,goods string,price string,amount string) row format delimited fields terminated by '\t' location '/imxlee/out';

## Flume

nohup flume-ng agent -n agent1 -f spoll2hdfs.agent -Dflume.root.logger=DEBUG,console &

# 项目代码

## 产生随机订单

## 写Log4j日志

以下是log4j的配置文件

log4j.rootLogger=ERROR, stdout, R  
log4j.appender.stdout=org.apache.log4j.ConsoleAppender  
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout  
log4j.appender.stdout.layout.ConversionPattern=%m%n  
  
log4j.appender.R=org.apache.log4j.RollingFileAppender  
log4j.appender.R.File=./logs/example.log  
log4j.appender.R.MaxFileSize=12KB  
log4j.appender.R.layout=org.apache.log4j.PatternLayout  
log4j.appender.R.layout.ConversionPattern=%m%n  
log4j.logger.cn.crxy=INFO

这里设定的输出文件大小是12KB，方便测试。

特别注意ConversionPattern的格式，目的是只输出msg，不输出其他。

运行稳定后，这里的stdout可以删除。目前存在的意义是为了测试方便。

## 写Kafka

创建主题语句

/opt/kafka/bin/kafka-topics.sh --create --zookeeper 192.168.1.100:2181,192.168.1.101:2181,192.168.1.102:2181 --replication-factor 1 --partitions 1 --topic sheet\_topic

## 打jar包部署

使用maven-assembly-plugin插件打包

<plugin>  
 <artifactId>maven-assembly-plugin</artifactId>  
 <configuration>  
 <descriptorRefs>  
 <descriptorRef>jar-with-dependencies</descriptorRef>  
 </descriptorRefs>  
 <archive>  
 <manifest>  
 <mainClass>xxxx.SheetGeneratorServer</mainClass>  
 </manifest>  
 </archive>  
 </configuration>  
 <executions>  
 <execution>  
 <id>make-assembly</id>  
 <phase>package</phase>  
 <goals>  
 <goal>single</goal>  
 </goals>  
 </execution>  
 </executions>  
 </plugin>

执行命令mvn clean package就能打包，结果在target目录中。

打完包后，使用winrar打开，检查一下里面的内容是否全面。特别是检查log4j.properties和price900文件是否存在。

## 运行jar

上传到linux后，执行java -jar xxxxx.jar就可以运行。

注意观察输出的log4j日志内容。观察文件的变化。

## 使用flume NG收集到HDFS

agent的配置文件如下所示

*#起名字*agent1.sources=s1  
agent1.sinks=k1  
agent1.channels=c1  
  
*#配置source*agent1.sources.s1.type=exec  
agent1.sources.s1.command=tail -F /home/wuchao/logs/example.log  
agent1.sources.s1.channels=c1  
  
*#配置sink*agent1.sinks.k1.type = hdfs  
agent1.sinks.k1.channel = c1  
agent1.sinks.k1.hdfs.path = hdfs://192.168.1.100:9000/user/out  
agent1.sinks.k1.hdfs.fileType = DataStream  
agent1.sinks.k1.hdfs.rollInterval=0  
agent1.sinks.k1.hdfs.rollSize=10240  
agent1.sinks.k1.hdfs.rollCount=0  
agent1.sinks.k1.hdfs.idleTimeout=0  
  
*#配置channel*agent1.channels.c1.type = SPILLABLEMEMORY

注意这里hdfs的配置。默认hdfs大小是10KB，方便测试。

执行命令

nohup flume-ng agent -n agent1 -f exec2hdfs.agent -Dflume.root.logger=DEBUG,console & 就可以启动flume

## 创建Hive外部表管理数据

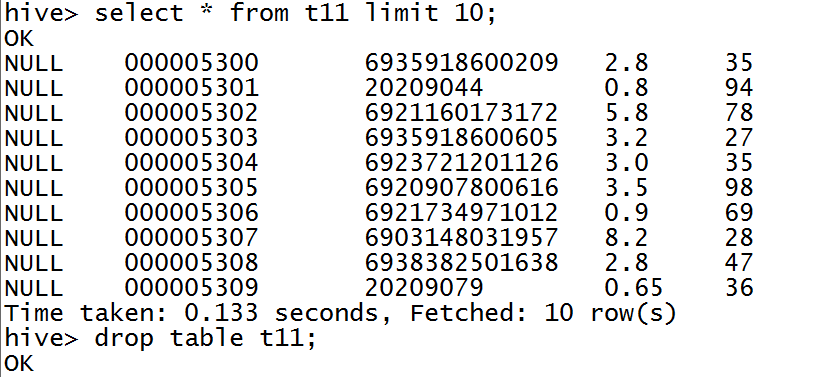
建表语句如下：

create external table t11(logtime int, sid string, pid string, price double, amount int)

row format delimited fields terminated by '\t'

location '/user/out';

显示结构如下



（请忽略左侧的NULL，着急，没调试）

## 使用Storm消费Kafka数据