# 实验四 **数据质量Apache griffin安装与集成**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | B10楼 | 303房； | **实验台号：** |  |
| **实验日期与时间：** | 2023/4/25 | | **评 分：** |  |
| **预习检查纪录：** |  | | **实验教师：** | 蔡文学 |

1. **实验目的**

随着业务发展和数据量的增加，大数据应用开发已成为部门应用开发常用的开发方式，由于部门业务特点的关系，spark和hive应用开发在部门内部较为常见。当处理的数据量达到一定量级和系统的复杂度上升时，数据的唯一性、完整性、一致性等等校验就开始受到关注，而通常做法是根据业务特点，额外开发job如报表或者检查任务，这样会比较费时费力。

目前遇到的表大部分在几亿到几十亿的数据量之间，并且报表数量在不断增加，在这种情况下，一个可配置、可视化、可监控的数据质量工具就显得尤为重要了。Griffin 数据质量监控工具正是可以解决前面描述的数据质量问题的开源解决方案。

1. **实验平台**

(1).操作系统：ubuntu

(2).java版本：1.8.0

(3).zookeeper版本:3.5.7

(4).mysql版本:5.7.28

(5).hive版本:3.1.2

(6).scala版本:2.11.8

(7).spark版本:2.4.5

(8).Livy版本:0.5.0

(9).griffin版本:0.5.0

(10).Kibana版本:6.3.1

(11).elasticsearch版本:6.3.1

(12).hadoop版本:3.1.3

1. **实验步骤（包括实验结果/截图）**

由于java已经安装过了，所以这里就不赘述了。

安装ELasticSearch

1.解压安装包

tar -zxvf elasticsearch-6.3.1.tar.gz -C /opt/module

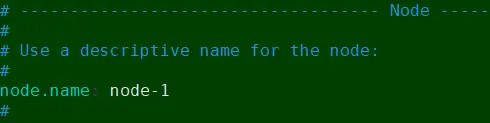
2.修改配置文件config/elasticsearch.yml，添加以下内容：

#集群名，主机名，端口号，关闭bootstrap程序：



elasticsearch.yml

当前节点名称：



elasticsearch.yml

集群节点：

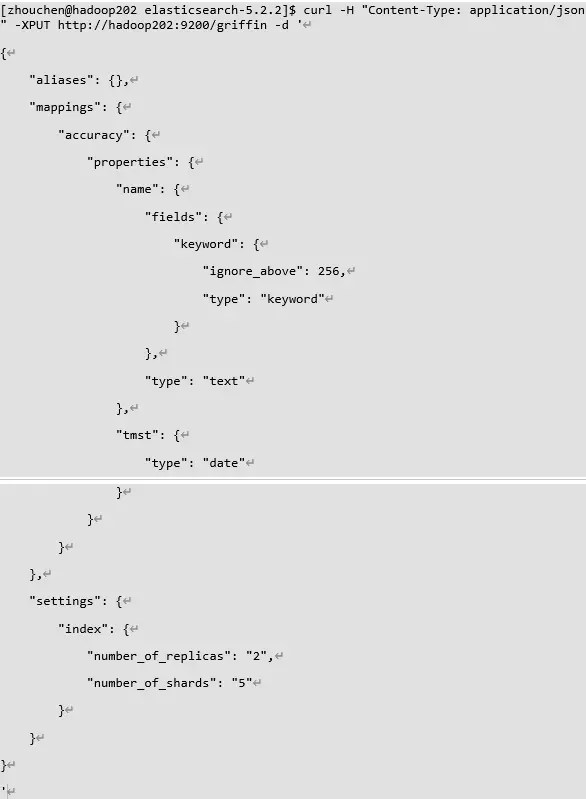


elasticsearch.yml

启动ES

$bin/elasticsearch

创建Griffin索引



ES创建Griffin索引

2.3.2 修改默认Linux配置

1. 修改默认打开的最大文件数

sudo vim /etc/security/limits.conf

#添加如下内容

#ElesticSearch添加的内容

\* soft nofile 65536

\* hard nofile 131072

\* soft nproc 2048

\* hard nproc 65536

2. 用户允许的最大进程数量

sudo vim /etc/security/limits.d/20-nproc.conf

\* soft nproc 4096

3. 虚拟内存区域的数量

sudo vim /etc/sysctl.conf

#ElesticSearch添加的内容

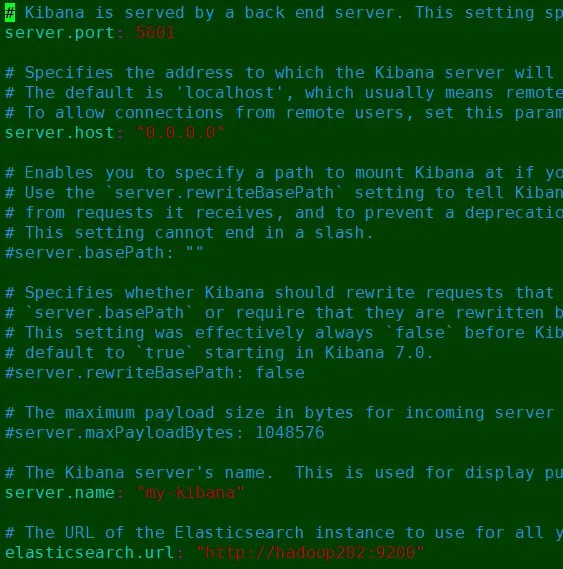
vm.max\_map\_count=655360

安装kibana

解压安装包

tar -zxvf kibana-6.3.1-linux-x86\_64.tar.gz -C /opt/module

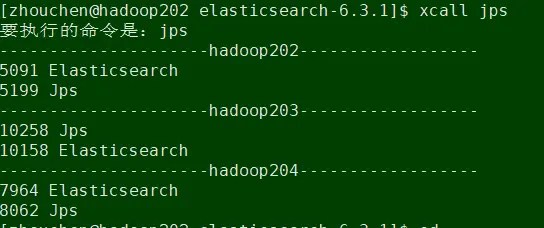
2.配置conf/kibana.yml



kibana.yml

ES/Kibana安装检查

1.查看进程



ES 的进程

2.elasticsearch界面访问

http://localhost:9200/\_cat/nodes?v



ES Web界面

3.kibana界面访问

http://localhost:5601/



kibana Web界面

安装spark

安装spark之前，默认已经安装好scala环境(这里我安装的版本是2.11.8)。

解压安装包

$ tar -zxvf spark-2.4.5-bin-hadoop2.7.tgz -C /opt/module/

$ mv spark-2.4.5-bin-hadoop2.7/ spark-2.4.5

修改配置spark-default.conf

$ mvspark-defaults.conf.template?spark-defaults.conf

$ vim spark-defaults.conf

#添加如下配置

spark.eventLog.enabledrue

spark.eventLog.dir hdfs://localhost:9820/spark\_directory

spark.sql.autoBroadcastJoinThreshold?? 1

Hadoop集群上提前创建spark\_directory

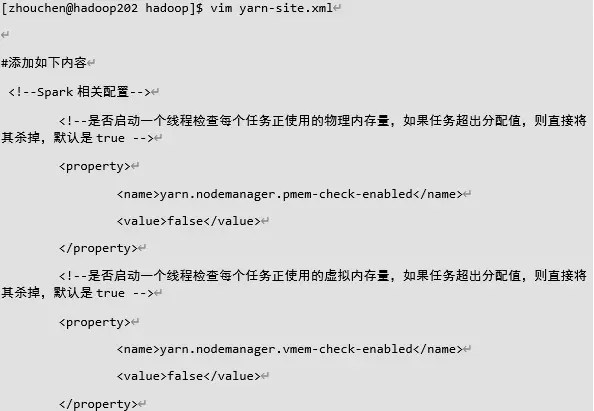
$ hadoop fs -mkdir /spark\_directory

修改配置spark-env.xml



spark-env.xml

修改yarn-site.xml



yarnnn-site.xml

拷贝jar包

将Hive中/opt/module/hive-2.3.6/lib/datanucleus-\*.jar包拷贝到Spark的/opt/module/spark/jars路径

$ cp /opt/module/hive/lib/datanucleus-\*.jar /opt/module/spark-2.4.5/jars/

拷贝配置文件

将Hive中/opt/module/hive-2.3.6/conf/hive-site.xml包拷贝到Spark的/opt/module/spark/conf路径

$ cp /opt/module/hive/conf/hive-site.xml? /opt/module/spark-2.4.5/conf/

修改slaves

$ mv slaves.template slaves

$ vim slaves

#修改内容如下：

hadoop203

hadoop204

分发spark安装目录到其他节点

$ xsync /opt/module/spark-2.4.5/

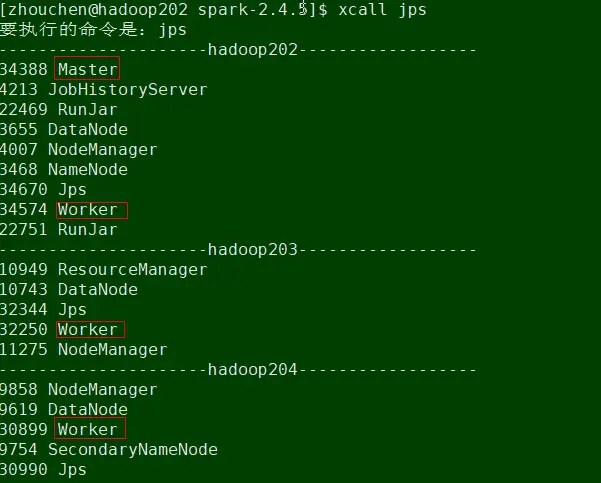
安装检查

1.启动spark集群模式：

$ bin/start-all.sh

$ xcall jps

2.检查进程



spark各进程

3. 运行示例：

$ bin/run-example SparkPi 2>&1 | grep "Pi is roughly"

4.检查计算结果



spark计算Pi的结果

安装Livy

1.解压

[ software]$ unzip incubator-livy-0.5.0-incubating.zip -d /opt/module/

2.修改配置文件

[ conf]$ mv livy.conf.template livy.conf

[ conf]$ mv livy-client.conf.template livy-client.conf

[ conf]$ mv livy-env.sh.template livy-env.sh

[ conf]$ mv spark-blacklist.conf.template spark-blacklist.conf

[ conf]$ mv log4j.properties.template log4j.properties

[ conf]$ vim livy-env.sh

#添加的内容如下

export JAVA\_HOME=/opt/module/jdk1.8

export SPARK\_HOME=/opt/module/spark-2.4.5

export SPARK\_CONF\_DIR=$SPARK\_HOME/conf

export HADOOP\_HOME=/opt/module/hadoop-3.1.3

export HADOOP\_CONF\_DIR=$HADOOP\_HOME/etc/hadoop

[ conf]$ vim livy.conf

#添加的内容如下

livy.server.host = localhost

livy.server.port = 8998

livy.spark.master = yarn

livy.repl.enableHiveContext = true

livy.spark.deploy-mode = client

3.手动创建logs目录

mkdir /opt/module/livy-0.5.0/logs

4.启动(脚本不带start操作会打印前台进程，不保存pid无法关闭，建议带start操作)

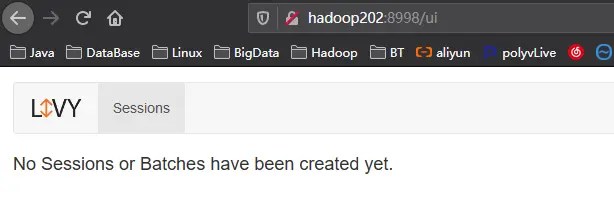
bin/livy-server



livy启动

4.检查安装

http://localhost:8998



Livy Web界面

由于maven已经安装过了，所以这里就不赘述了。

2.7 Shell 脚本

编写各个集群启动/停止脚本 /home/usr/bin

Hadoop启动脚本

vi hadoop.sh



hadoop.sh

Hive启动脚本

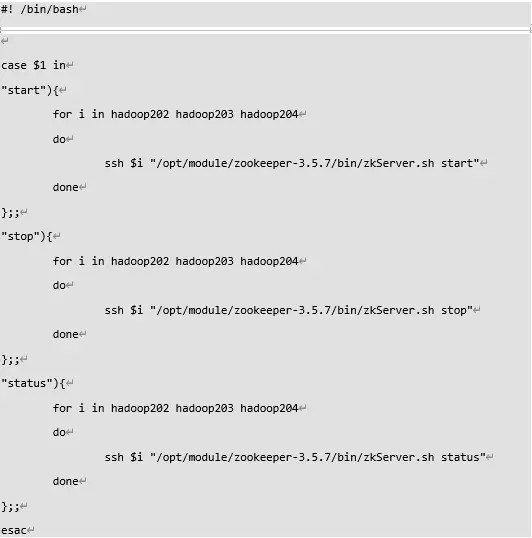
vi hiveservices.sh



hiveservices.sh

Zookeeper启动脚本

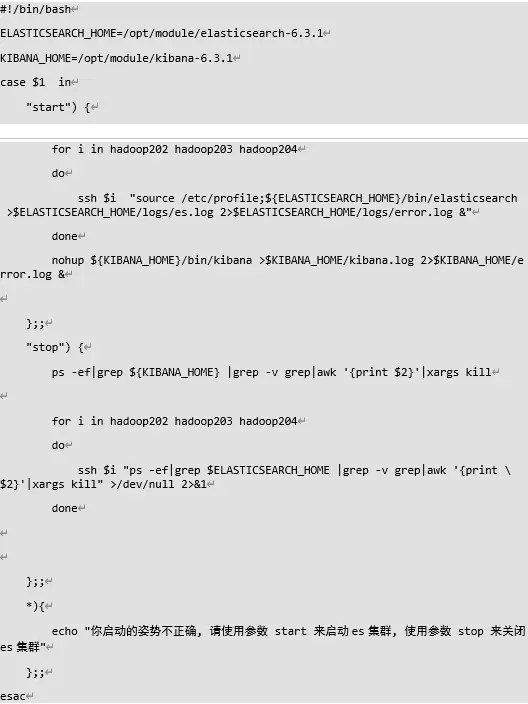
vi zk.sh



zk.sh

ElasticSearch/Kibane启动脚本

vi es.sh



es.sh

Livy启动脚本

vi livy.sh



livy.sh

编译Griffin

解压安装包

unzip griffin-master.zip -d /opt/module/

初始化mysql中的quartz

1.创建quartz库

mysql -u root -e "create database quartz" -p

2. 使用使用解压包中的脚本初始化数据库

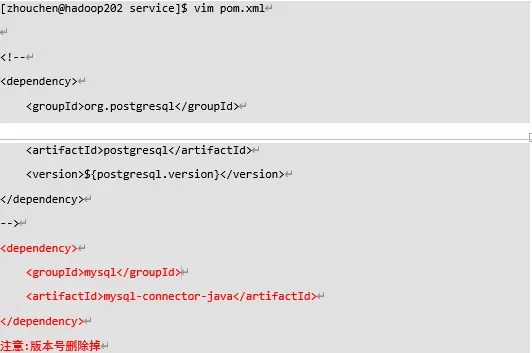
service/src/main/resources/Init\_quartz\_mysql\_innodb.sql

mysql -u root -p quartz < service/src/main/resources/Init\_quartz\_mysql\_innodb.sql

修改配置文件

1.修改/opt/module/griffin-master/service/pom.xml文件

注释掉org.postgresql，添加mysql依赖。



pom.xml

2.修改/opt/module/griffin-master/service/src/main/resources/application.properties文件

vim /opt/module/griffin-master/service/src/main/resources/application.properties



application.properties

3.修改/opt/module/griffin-master/service/src/main/resources/sparkProperties.json文件

vim /opt/module/griffin-master/service/src/main/resources/sparkProperties.json



sparkProperties.json

4.修改/opt/module/griffin-master/service/src/main/resources/env/env\_batch.json文件

vim /opt/module/griffin-master/service/src/main/resources/env/env\_batch.json



env\_batch.json

5. /opt/module/griffin-master/service/src/main/resources/env/env\_streaming.json文件

vim /opt/module/griffin-master/service/src/main/resources/env/env\_streaming.json



env\_streaming.json

执行编译

1.启动hadoop

bash hadoop.sh start

2.启动zookeeper

bash zk.sh start

3.启动hive

bash hiveservices.sh start

4.启动ElasticSearch

bash es.sh start

5.启动livy

bash livy.sh start

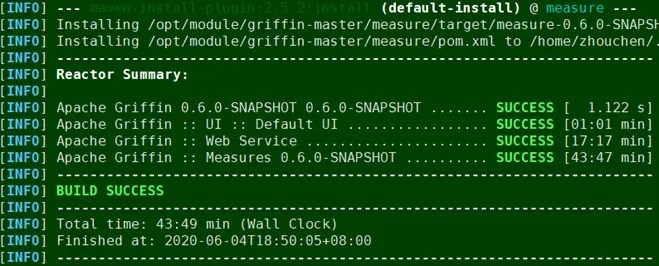
6.启动Spark

/opt/module/spark-2.4.5/sbin/start-all.sh

7.在/opt/module/griffin-master路径执行maven命令，开始编译Griffin源码

mvn -Dmaven.test.skip=true clean install

8.编译完成



build success

修改名称并上传hdfs

命令执行完成后，会在Service和Measure模块的target目录下分别看到service-0.6.0.jar和measure-0.6.0.jar两个jar包

1.修改/opt/module/griffin-master/measure/target/measure-0.6.0-SNAPSHOT.jar名称

mv measure-0.6.0-SNAPSHOT.jar griffin-measure.jar

2.上传griffin-measure.jar到HDFS文件目录里

hadoop fs -mkdir /griffin/

hadoop fs -put griffin-measure.jar /griffin/

这样做的目的主要是因为Spark在YARN集群上执行任务时，需要到HDFS的/griffin目录下加载griffin-measure.jar，避免发生类org.apache.griffin.measure.Application找不到的错误。

3.上传hive-site.xml文件到HDFS的/home/spark\_conf/路径

hadoop fs -mkdir -p /home/spark\_conf/

hadoop fs -put /opt/module/hive-2.3.6/conf/hive-site.xml /home/spark\_conf/

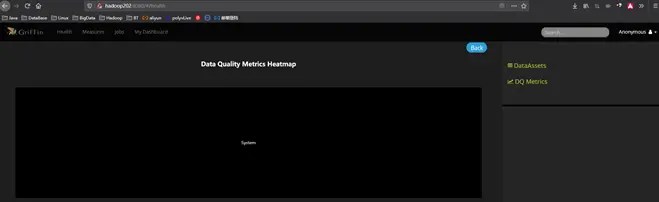
4.进入到/opt/module/griffin-master/service/target/路径，运行service-0.6.0-SNAPSHOT.jar

控制台启动：控制台打印信息$ java -jar service-0.6.0-SNAPSHOT.jar

5.后台启动：启动后台并把日志归写倒service.out

$ nohup java -jar service-0.6.0-SNAPSHOT.jar>service.out 2>&1 &

最后访问http://localhost:8080 默认账户和密码都是无



Griffin的 Web页面

1. **思考讨论题或体会或对改进实验的建议**

缺点：

1.整体功能比较单一，只有数据监控功能。

2.依赖组件太多，安装繁琐。

3.缺少预警功能。

优点：

1.配置简单，上手难度较低。

1. **实验源代码**