

# 尚硅谷大数据技术之 Griffin

(作者：尚硅谷大数据研发部)

版本：V2.0

## 第 1 章 Griffin 入门

### 1.1 Griffin 概述

Apache Griffin 是一个开源的大数据数据质量解决方案，它支持批处理和流模式两种数据质量检测方式，可以从不同维度度量数据资产，从而提升数据的准确度和可信度。例如：离线任务执行完毕后检查源端和目标端的数据数量是否一致，源表的数据空值等。

### 1.2 Griffin 架构原理

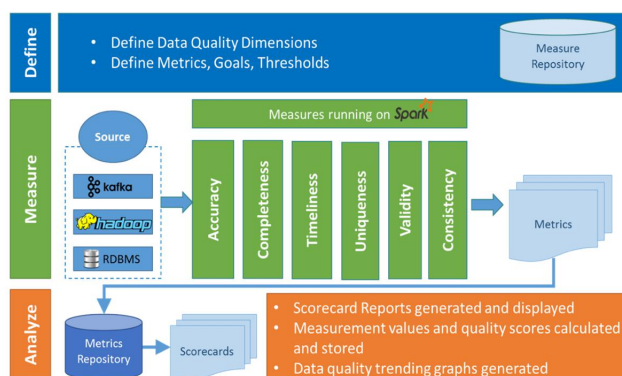
#### Griffin架构原理



**Define:** 主要负责定义数据质量统计的维度，比如数据质量统计的时间跨度，统计的目标（源端和目标端的数据数量是否一致，数据源里某一字段的非空的数，不重复值的数量，最大值，最小值，TOP5的值数量等）。

**Measure:** 主要负责执行统计任务，生成统计结果。

**Analyze:** 主要负责保存与展示统计结果。



让天下没有难学的技术

## 第 2 章 Griffin 安装及使用

### 2.1 安装前环境准备

#### 2.1.1 安装 ES5.2

1) 上传 elasticsearch-5.2.2.tar.gz 到 hadoop102 的 /opt/software 目录，并解压到 /opt/module 目录

```
[atguigu@hadoop102 software]$ tar -zxvf elasticsearch-5.2.2.tar.gz -C /opt/module/
```

2) 修改 /opt/module/elasticsearch-5.2.2/config/elasticsearch.yml 配置文件

更多 Java - 大数据 - 前端 - python 人工智能资料下载，可百度访问：尚硅谷官网

```
[atguigu@hadoop102 config]$ vim elasticsearch.yml

network.host: hadoop102
http.port: 9200
http.cors.enabled: true
http.cors.allow-origin: "*"
bootstrap.memory_lock: false
bootstrap.system_call_filter: false
```

### 3) 修改 Linux 系统配置文件/etc/security/limits.conf

```
[atguigu@hadoop102 elasticsearch-5.2.2]$ sudo vim
/etc/security/limits.conf

#添加如下内容
* soft nproc 65536
* hard nproc 65536
* soft nofile 65536
* hard nofile 65536
```

```
[atguigu@hadoop102 elasticsearch-5.2.2]$ sudo vim
/etc/sysctl.conf

#添加
vm.max_map_count=655360
```

```
[atguigu@hadoop102 elasticsearch-5.2.2]$ sudo vim
/etc/security/limits.d/90-nproc.conf

#修改配置
*          soft      nproc      2048
```

```
[atguigu@hadoop102 elasticsearch-5.2.2]$ sudo sysctl -p
```

### 4) 需要重新启动虚拟机

```
[atguigu@hadoop102 elasticsearch-5.2.2]$ su root
root@hadoop102 elasticsearch-5.2.2]# reboot
```

### 5) 在/opt/module/elasticsearch-5.2.2 路径上, 启动 ES

```
[atguigu@hadoop102 elasticsearch-5.2.2]$ bin/elasticsearch
```

### 6) 在 ES 里创建 griffin 索引

```
[atguigu@hadoop102 ~]$ curl -XPUT http://hadoop102:9200/griffin
-d '{
  {
    "aliases": {},
    "mappings": {
      "accuracy": {
        "properties": {
          "name": {
            "fields": {
              "keyword": {
                "ignore_above": 256,
                "type": "keyword"
              }
            }
          }
        }
      }
    }
  },
```

```
        "type": "text"
      },
      "tmst": {
        "type": "date"
      }
    }
  },
  "settings": {
    "index": {
      "number_of_replicas": "2",
      "number_of_shards": "5"
    }
  }
}
```

## 2.1.2 安装 JDK8、Hadoop2.7.2

注意：JDK 版本至少 1.8 及以上，Hadoop 版本至少 2.6.0 及以上

### 1) 安装 Hadoop 集群



## 尚硅谷大数据技术 之Hadoop (入门)

### 2) 启动 Hadoop 集群

```
[atguigu@hadoop102 hadoop-2.7.2]$ sbin/start-dfs.sh
[atguigu@hadoop103 hadoop-2.7.2]$ sbin/start-yarn.sh
```

## 2.1.3 安装 Hive2.3

注意：Hive 版本至少 2.2 及以上

### 1) 上传 apache-hive-2.3.0-bin.tar.gz 到/opt/software 目录下，并解压到/opt/module

```
[atguigu@hadoop102 software]$ tar -zxvf
apache-hive-2.3.6-bin.tar.gz -C /opt/module/
```

### 2) 修改 apache-hive-2.3.6-bin 名称为 hive-2.3.6

```
[atguigu@hadoop102 module]$ mv apache-hive-2.3.6-bin hive-2.3.6
```

### 3) 将 Mysql 的 mysql-connector-java-5.1.27-bin.jar 拷贝到/opt/module/hive-2.3.6/lib/

```
[atguigu@hadoop102 module]$ cp
/opt/software/mysql-libs/mysql-connector-java-5.1.27/mysql-con
nector-java-5.1.27-bin.jar /opt/module/hive-2.3.6/lib/
```

### 4) 在/opt/module/hive-2.3.6/conf 路径上，创建 hive-site.xml 文件

```
[atguigu@hadoop102 conf]$ vim hive-site.xml
```

#添加如下内容

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
```

```
<configuration>
  <property>
    <name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:mysql://hadoop102:3306/metastore?createDatabaseIfNotExist=true</value>
    <description>JDBC connect string for a JDBC metastore</description>
  </property>

  <property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
    <description>Driver class name for a JDBC metastore</description>
  </property>

  <property>
    <name>javax.jdo.option.ConnectionUserName</name>
    <value>root</value>
    <description>username to use against metastore database</description>
  </property>

  <property>
    <name>javax.jdo.option.ConnectionPassword</name>
    <value>000000</value>
    <description>password to use against metastore database</description>
  </property>

  <property>
    <name>hive.metastore.warehouse.dir</name>
    <value>/user/hive/warehouse</value>
    <description>location of default database for the warehouse</description>
  </property>

  <property>
    <name>hive.cli.print.header</name>
    <value>true</value>
  </property>

  <property>
    <name>hive.cli.print.current.db</name>
    <value>true</value>
  </property>

  <property>
    <name>hive.metastore.schema.validation</name>
    <value>false</value>
  </property>

  <property>
    <name>datanucleus.schema.autoCreateAll</name>
    <value>true</value>
  </property>

  <property>
    <name>hive.metastore.uris</name>
    <value>thrift://hadoop102:9083</value>
  </property>
</configuration>
```

### 3) 启动服务

更多 Java - 大数据 - 前端 - python 人工智能资料下载，可百度访问：[尚硅谷官网](http://www.shangguigu.com)

```
[atguigu@hadoop102 hive-2.3.6]$ nohup bin/hive --service metastore &
[atguigu@hadoop102 hive-2.3.6]$ nohup bin/hive --service hiveserver2 &
```

注意: hive2.x 版本需要启动两个服务 metastore 和 hiveserver2, 否则会报错 Exception in thread "main" java.lang.RuntimeException: org.apache.hadoop.hive.ql.metadata.HiveException: java.lang.RuntimeException: Unable to instantiate org.apache.hadoop.hive.ql.metadata.SessionHiveMetaStoreClient

- 4) 服务启动完毕后在启动 Hive

```
[atguigu@hadoop102 hive-2.3.6]$ bin/hive
```

## 2.1.4 安装 Spark2.4.3

注意: Spark 版本至少 2.2.1 及以上

- 1) 把 spark-2.4.3-bin-hadoop2.7.tgz 上传到/opt/software 目录, 并解压到/opt/module

```
[atguigu@hadoop102 software]$ tar -zxvf spark-2.4.3-bin-hadoop2.7.tgz -C /opt/module/
```

- 2) 修改名称/opt/module/spark-2.4.3-bin-hadoop2.7 名称为 spark

```
[atguigu@hadoop102 module]$ mv spark-2.4.3-bin-hadoop2.7/ spark
```

- 3) 修改/opt/module/spark/conf/spark-defaults.conf.template 名称为 spark-defaults.conf

```
[atguigu@hadoop102 conf]$ mv spark-defaults.conf.template spark-defaults.conf
```

- 4) 在 hadoop 集群上提前创建 spark\_directory 日志路径

```
[atguigu@hadoop102 spark]$ hadoop fs -mkdir /spark_directory
```

- 5) 在 spark-default.conf 文件中配置 Spark 日志路径

```
[atguigu@hadoop102 conf]$ vim spark-defaults.conf
```

```
#添加如下配置
spark.eventLog.enabled true
spark.eventLog.dir
hdfs://hadoop102:9000/spark_directory
```

- 6) 修改/opt/module/spark/conf/spark-env.sh.template 名称为 spark-env.sh

```
[atguigu@hadoop102 conf]$ mv spark-env.sh.template spark-env.sh
```

- 7) 在/opt/module/spark/conf/spark-env.sh 文件中配置 YARN 配置文件路径、配置历史服务器

相关参数

```
[atguigu@hadoop102 conf]$ vim spark-env.sh
```

```
#添加如下参数
YARN_CONF_DIR=/opt/module/hadoop-2.7.2/etc/hadoop
export SPARK_HISTORY_OPTS="-Dspark.history.ui.port=18080
-Dspark.history.retainedApplications=30
-Dspark.history.fs.logDirectory=hdfs://hadoop102:9000/spark_directory"
```

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

```
[atguigu@hadoop102 lib]$ cp /opt/module/hive-2.3.6/lib/datanucleus-*.jar /opt/module/spark/jars/
```

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

```
[atguigu@hadoop102 conf]$ cp /opt/module/hive-2.3.6/conf/hive-site.xml /opt/module/spark/conf/
```

- 10) 测试环境

```
[atguigu@hadoop102 spark]$ bin/spark-shell  
  
scala>spark.sql("show databases").show
```

### 2.1.5 安装 Livy0.3

- 1) 上传 livy-server-0.3.0.zip 到 hadoop102 的 /opt/software 目录下，并解压到 /opt/module

```
[atguigu@hadoop102 software]$ unzip livy-server-0.3.0.zip -d /opt/module/
```

- 2) 修改 /opt/module/livy-server-0.3.0 文件名称为 livy

```
[atguigu@hadoop102 module]$ mv livy-server-0.3.0/ livy
```

- 3) 修改 /opt/module/livy/conf/livy.conf 文件，配置 livy 与 spark 相关参数

```
livy.server.host = Hadoop102  
livy.spark.master = yarn  
livy.spark.deployMode = client  
livy.repl.enableHiveContext = true  
livy.server.port = 8998
```

- 4) 配置需要的环境变量

```
[atguigu@hadoop102 conf]$ sudo vim /etc/profile  
  
#SPARK_HOME  
export SPARK_HOME=/opt/module/spark  
export PATH=$PATH:$SPARK_HOME/bin  
  
[atguigu@hadoop102 conf]$ source /etc/profile
```

- 5) 在 /opt/module/livy/ 路径上，启动 livy 服务

```
[atguigu@hadoop102 livy]$ bin/livy-server start
```

### 2.1.6 安装 MySQL5.6

- 1) 上传 Init\_quartz\_mysql\_innodb.sql 到 hadoop102 的 /opt/software 目录

- 2) 使用 mysql 创建 quartz 库，执行脚本初始化表信息

```
[atguigu@hadoop102 ~]$ mysql -uroot -p000000  
mysql> create database quartz;  
mysql> use quartz;  
mysql> source /opt/software/Init_quartz_mysql_innodb.sql  
mysql> show tables;
```

## 2.1.7 安装 Maven

1) Maven 下载: <https://maven.apache.org/download.cgi>

2) 把 apache-maven-3.6.1-bin.tar.gz 上传到 linux 的 /opt/software 目录下

3) 解压 apache-maven-3.6.1-bin.tar.gz 到 /opt/module/ 目录下

```
[atguigu@hadoop102 software]$ tar -zxvf apache-maven-3.6.1-bin.tar.gz -C /opt/module/
```

4) 修改 apache-maven-3.6.1 的名称为 maven

```
[atguigu@hadoop102 module]$ mv apache-maven-3.6.1/ maven
```

5) 添加环境变量到 /etc/profile 中

```
[atguigu@hadoop102 module]$ sudo vim /etc/profile
#MAVEN_HOME
export MAVEN_HOME=/opt/module/maven
export PATH=$PATH:$MAVEN_HOME/bin
```

6) 测试安装结果

```
[atguigu@hadoop102 module]$ source /etc/profile
[atguigu@hadoop102 module]$ mvn -v
```

7) 修改 setting.xml, 指定为阿里云

```
[atguigu@hadoop102 maven]$ cd conf
[atguigu@hadoop102 maven]$ vim settings.xml
```

```
<!-- 添加阿里云镜像 -->
<mirror>
  <id>nexus-aliyun</id>
  <mirrorOf>central</mirrorOf>
  <name>Nexus aliyun</name>
  <url>http://maven.aliyun.com/nexus/content/groups/public</url>
</mirror>
<mirror>
  <id>UK</id>
  <name>UK Central</name>
  <url>http://uk.maven.org/maven2</url>
  <mirrorOf>central</mirrorOf>
</mirror>
<mirror>
  <id>repo1</id>
  <mirrorOf>central</mirrorOf>
  <name>Human Readable Name for this Mirror.</name>
  <url>http://repo1.maven.org/maven2</url>
</mirror>
<mirror>
  <id>repo2</id>
  <mirrorOf>central</mirrorOf>
  <name>Human Readable Name for this Mirror.</name>
  <url>http://repo2.maven.org/maven2</url>
</mirror>
```

8) 在 /home/atguigu 目录下创建 .m2 文件夹

```
[atguigu@hadoop102 ~]$ mkdir .m2
```

## 2.2 编译 Griffin0.4.0

### 2.2.1 修改配置文件:

- 1) 上传 griffin-master.zip 到 hadoop102 的 /opt/software 目录，并解压 tar.gz 包到 /opt/module

```
[atguigu@hadoop102 software]$ unzip griffin-master.zip -d /opt/module/
```

- 2) 修改 /opt/module/griffin-master/service/pom.xml 文件，注释掉 org.postgresql，添加 mysql 依赖。

```
[atguigu@hadoop102 service]$ vim pom.xml

<!--
<dependency>
  <groupId>org.postgresql</groupId>
  <artifactId>postgresql</artifactId>
  <version>${postgresql.version}</version>
</dependency>
-->
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
</dependency>
注意:版本号删除掉
```

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

```
[atguigu@hadoop102 service]$ vim /opt/module/griffin-master/service/src/main/resources/application.properties

# Apache Griffin 应用名称
spring.application.name=griffin_service
# MySQL 数据库配置信息
spring.datasource.url=jdbc:mysql://hadoop102:3306/quartz?autoReconnect=true&useSSL=false
spring.datasource.username=root
spring.datasource.password=000000
spring.jpa.generate-ddl=true
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.jpa.show-sql=true
# Hive metastore 配置信息
hive.metastore.uris=thrift://hadoop102:9083
hive.metastore.dbname=default
hive.hmsHandler.retry.attempts=15
hive.hmsHandler.retry.interval=2000ms
# Hive cache time
cache.evict.hive.fixedRate.in.milliseconds=900000
# Kafka schema registry 按需配置
kafka.schema.registry.url=http://hadoop102:8081
# Update job instance state at regular intervals
jobInstance.fixedDelay.in.milliseconds=60000
# Expired time of job instance which is 7 days that is 604800000 milliseconds. Time unit only supports milliseconds
jobInstance.expired.milliseconds=604800000
```



```
# schedule predicate job every 5 minutes and repeat 12 times at most
#interval time unit s:second m:minute h:hour d:day,only support these four units
predicate.job.interval=5m
predicate.job.repeat.count=12
# external properties directory location
external.config.location=
# external BATCH or STREAMING env
external.env.location=
# login strategy ("default" or "ldap")
login.strategy=default
# ldap
ldap.url=ldap://hostname:port
ldap.email=@example.com
ldap.searchBase=DC=org,DC=example
ldap.searchPattern=(SAMAccountName={0})
# hdfs default name
fs.defaultFS=
# elasticsearch
elasticsearch.host=hadoop102
elasticsearch.port=9200
elasticsearch.scheme=http
# elasticsearch.user = user
# elasticsearch.password = password
# livy
livy.uri=http://hadoop102:8998/batches
# yarn url
yarn.uri=http://hadoop103:8088
# griffin event listener
internal.event.listeners=GriffinJobEventHook
```

#### 4) 修改/opt/module/griffin-master/service/src/main/resources/sparkProperties.json 文件

```
[atguigu@hadoop102 service]$ vim /opt/module/griffin-master/service/src/main/resources/sparkProperties.json

{
  "file": "hdfs://hadoop102:9000/griffin/griffin-measure.jar",
  "className": "org.apache.griffin.measure.Application",
  "name": "griffin",
  "queue": "default",
  "numExecutors": 2,
  "executorCores": 1,
  "driverMemory": "1g",
  "executorMemory": "1g",
  "conf": {
    "spark.yarn.dist.files":
    "hdfs://hadoop102:9000/home/spark_conf/hive-site.xml"
  },
  "files": [
  ]
}
```

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

```
[atguigu@hadoop102 service]$ vim /opt/module/griffin-master/service/src/main/resources/env/env_batch.json
```

```
{
  "spark": {
    "log.level": "INFO"
  },
  "sinks": [
    {
      "type": "CONSOLE",
      "config": {
        "max.log.lines": 10
      }
    },
    {
      "type": "HDFS",
      "config": {
        "path": "hdfs://hadoop102:9000/griffin/persist",
        "max.persist.lines": 10000,
        "max.lines.per.file": 10000
      }
    },
    {
      "type": "ELASTICSEARCH",
      "config": {
        "method": "post",
        "api": "http://hadoop102:9200/griffin/accuracy",
        "connection.timeout": "1m",
        "retry": 10
      }
    }
  ],
  "griffin.checkpoint": []
}
```

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

```
[atguigu@hadoop102 service]$ vim
/opt/module/griffin-master/service/src/main/resources/env/env_
streaming.json

{
  "spark": {
    "log.level": "WARN",
    "checkpoint.dir": "hdfs:///griffin/checkpoint/${JOB_NAME}",
    "init.clear": true,
    "batch.interval": "1m",
    "process.interval": "5m",
    "config": {
      "spark.default.parallelism": 4,
      "spark.task.maxFailures": 5,
      "spark.streaming.kafkaMaxRatePerPartition": 1000,
      "spark.streaming.concurrentJobs": 4,
      "spark.yarn.maxAppAttempts": 5,
      "spark.yarn.am.attemptFailuresValidityInterval": "1h",
      "spark.yarn.max.executor.failures": 120,
      "spark.yarn.executor.failuresValidityInterval": "1h",
      "spark.hadoop.fs.hdfs.impl.disable.cache": true
    }
  },
  "sinks": [
```

```
{
  "type": "CONSOLE",
  "config": {
    "max.log.lines": 100
  }
},
{
  "type": "HDFS",
  "config": {
    "path": "hdfs://hadoop102:9000/griffin/persist",
    "max.persist.lines": 10000,
    "max.lines.per.file": 10000
  }
},
{
  "type": "ELASTICSEARCH",
  "config": {
    "method": "post",
    "api": "http://hadoop102:9200/griffin/accuracy"
  }
}
],
"griffin.checkpoint": [
  {
    "type": "zk",
    "config": {
      "hosts": "zk:2181",
      "namespace": "griffin/infocache",
      "lock.path": "lock",
      "mode": "persist",
      "init.clear": true,
      "close.clear": false
    }
  }
]
}
```

7) 修改/opt/module/griffin-master/service/src/main/resources/quartz.properties 文件

```
[atguigu@hadoop102 service]$ vim
/opt/module/griffin-master/service/src/main/resources/quartz.p
roperties

org.quartz.scheduler.instanceName=spring-boot-quartz
org.quartz.scheduler.instanceId=AUTO
org.quartz.threadPool.threadCount=5
org.quartz.jobStore.class=org.quartz.impl.jdbcjobstore.JobStor
eTX
# If you use postgresql as your database,set this property value
to org.quartz.impl.jdbcjobstore.PostgreSQLDelegate
# If you use mysql as your database,set this property value to
org.quartz.impl.jdbcjobstore.StdJDBCDelegate
# If you use h2 as your database, it's ok to set this property value
to StdJDBCDelegate, PostgreSQLDelegate or others
org.quartz.jobStore.driverDelegateClass=org.quartz.impl.jdbcjo
bstore.StdJDBCDelegate
org.quartz.jobStore.useProperties=true
org.quartz.jobStore.misfireThreshold=60000
org.quartz.jobStore.tablePrefix=QRTZ_
```

```
org.quartz.jobStore.isClustered=true
org.quartz.jobStore.clusterCheckinInterval=20000
```

## 2.2.2 执行编译

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

```
[atguigu@hadoop102 griffin-master]$ mvn -Dmaven.test.skip=true
clean install
```

- 2) 见到如下页面，表示编译成功。（大约需要 1 个小时左右）

```
[INFO] -----
[INFO] Reactor Summary for Apache Griffin 0.6.0-SNAPSHOT 0.6.0-SNAPSHOT:
[INFO] Apache Griffin 0.6.0-SNAPSHOT ..... SUCCESS [01:15 min]
[INFO] Apache Griffin :: UI :: Default UI ..... SUCCESS [57:27 min]
[INFO] Apache Griffin :: Web Service ..... SUCCESS [09:03 min]
[INFO] Apache Griffin :: Measures ..... SUCCESS [05:07 min]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 01:13 h
[INFO] Finished at: 2019-09-19T15:58:48+08:00
[INFO] -----
[atguigu@hadoop102 griffin-master]$ █
```

## 2.2.3 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

```
[atguigu@hadoop102 measure]$ mv measure-0.6.0-SNAPSHOT.jar
griffin-measure.jar
```

- 2) 上传 griffin-measure.jar 到 HDFS 文件目录里

```
[atguigu@hadoop102 measure]$ hadoop fs -mkdir /griffin/
[atguigu@hadoop102 measure]$ hadoop fs -put griffin-measure.jar
/griffin/
```

注意：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

```
[atguigu@hadoop102 ~]$ hadoop fs -mkdir -p /home/spark_conf/
[atguigu@hadoop102 ~]$ 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

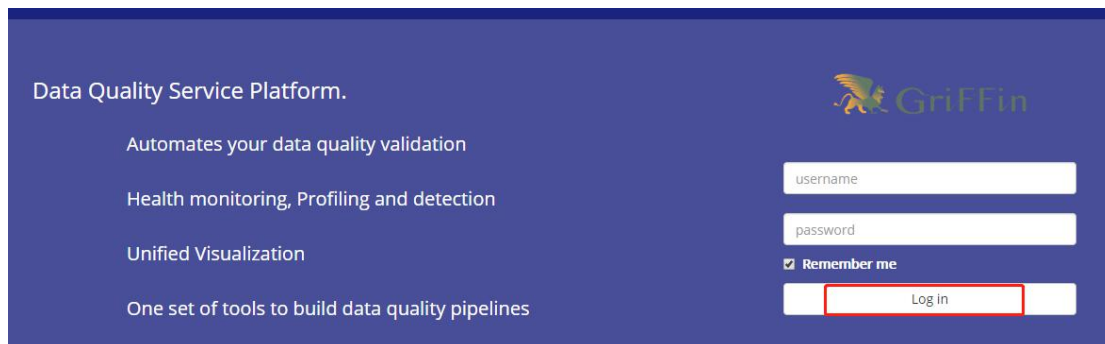
```
[atguigu@hadoop102 target]$ java -jar service-0.6.0-SNAPSHOT.jar
```

后台启动：启动后台并把日志归写倒 service.out

```
[atguigu@hadoop102 ~]$ nohup java -jar
service-0.6.0-SNAPSHOT.jar>service.out 2>&1 &
```

## 2.2.4 浏览器访问

http://hadoop102:8080 默认账户和密码都是无



## 第3章 案例实操

### 3.1 生产测试数据

获取官网测试数据。在/opt/module/目录下创建 data 文件夹，并下载相关测试数据

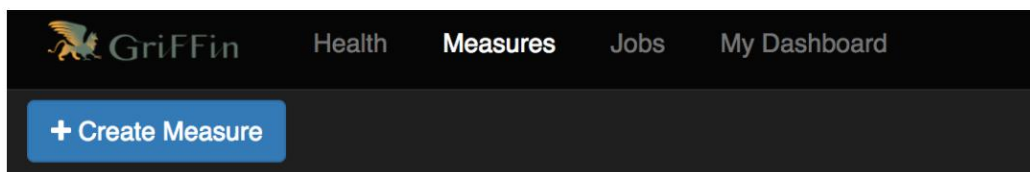
```
[atguigu@hadoop102 moudle]$ mkdir data
[atguigu@hadoop102 data]$
wget http://griffin.apache.org/data/batch/gen_demo_data.sh
wget http://griffin.apache.org/data/batch/gen_delta_src.sh
wget http://griffin.apache.org/data/batch/demo_basic
wget http://griffin.apache.org/data/batch/delta_tgt
wget
http://griffin.apache.org/data/batch/insert-data.hql.template
wget http://griffin.apache.org/data/batch/gen-hive-data.sh
wget http://griffin.apache.org/data/batch/create-table.hql
wget http://griffin.apache.org/data/batch/delta\_src
wget http://griffin.apache.org/data/batch/delta\_tgt

[atguigu@hadoop102 data]$ chmod 777 ../data -R
#生成临时文件
[atguigu@hadoop102 data]$ ./gen_demo_data.sh
#生产测试数据
[atguigu@hadoop102 data]$ ./gen-hive-data.sh
```

### 3.2 UI 创建 Measure

注意根据官网描述，目前 UI 创建 Measure 只支持 Accuracy 的 Measure，UI 界面上虽然有其他选项但是无法运行 job。

By clicking "Measures", and then choose "Create Measure". You can use the measure to process data and get the result you want.

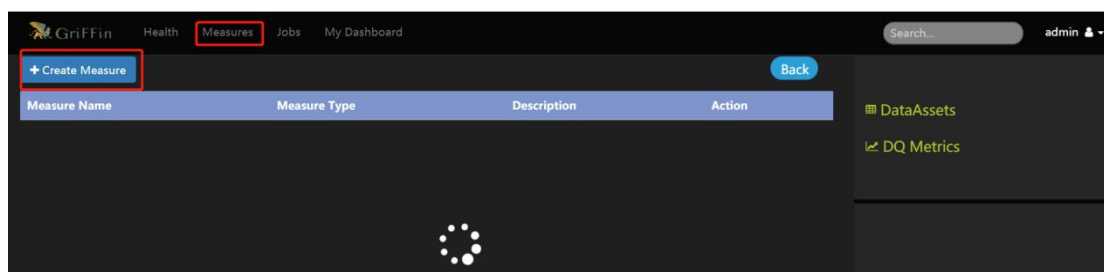


There are mainly four kinds of measures for you to choose, which are:

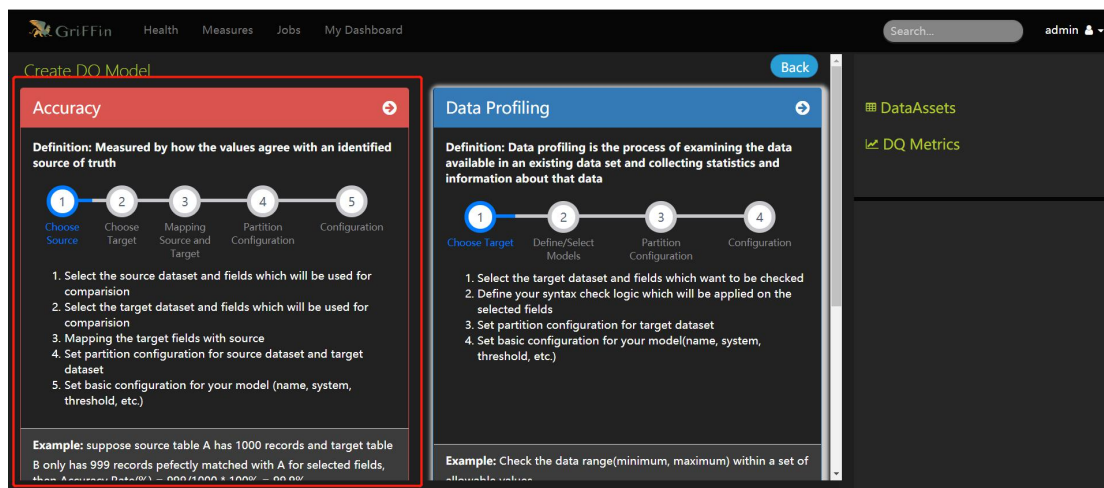
1. if you want to measure the match rate between source and target, choose accuracy.
2. if you want to check the specific value of the data(such as: null column count), choose profiling.

At current we only support accuracy measure creation from UI.

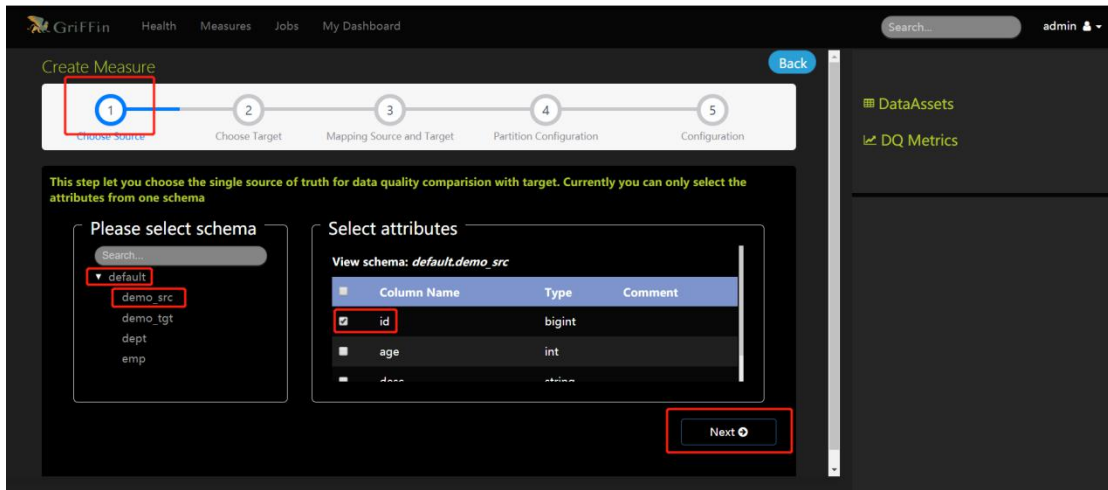
### 3.2.1 添加一个新的 Measure



### 3.2.2 选择准确度 Accuracy

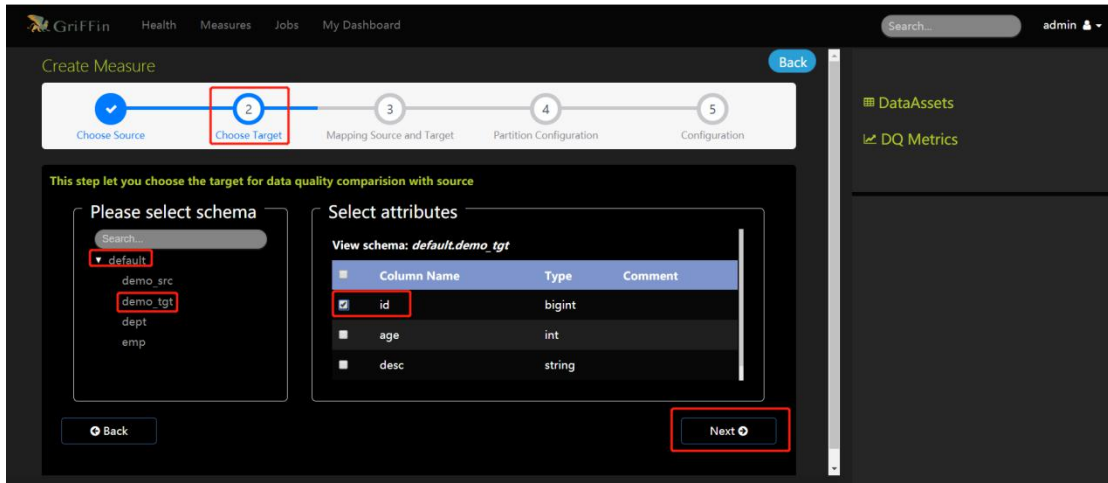


### 3.2.3 选择数据源的字段



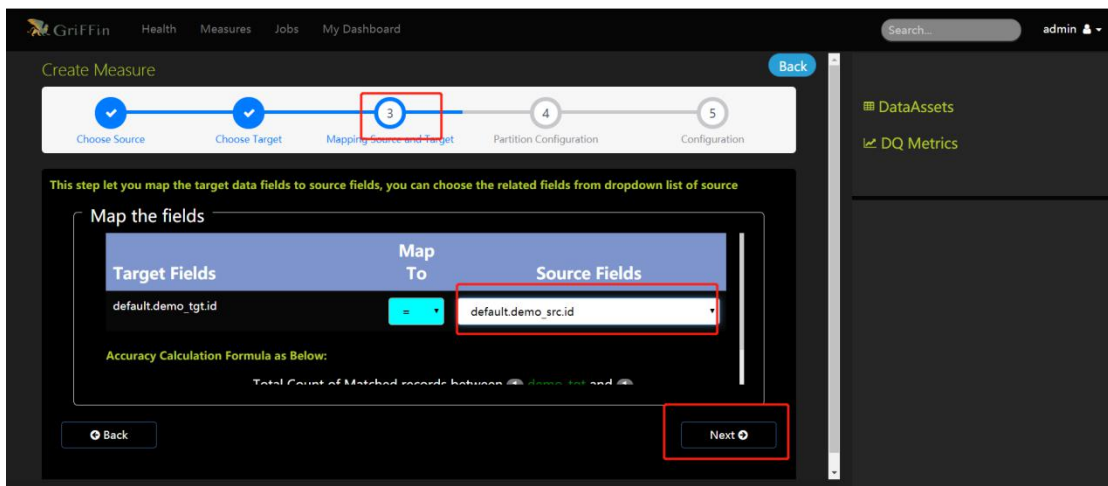
The screenshot shows the 'Create Measure' wizard in the Griffin interface. The first step, 'Choose Source', is highlighted with a red box. The progress bar at the top shows five steps: 1. Choose Source, 2. Choose Target, 3. Mapping Source and Target, 4. Partition Configuration, and 5. Configuration. The main content area has two sections: 'Please select schema' and 'Select attributes'. In 'Please select schema', a dropdown menu is open showing 'default' and 'demo\_src', with 'demo\_src' selected. In 'Select attributes', a table titled 'View schema: default.demo\_src' lists columns: 'id' (bigint), 'age' (int), and 'desc' (string). The 'id' column is selected with a checkbox. A 'Next' button is at the bottom right.

### 3.2.4 选择目标表的字段



The screenshot shows the 'Create Measure' wizard in the Griffin interface. The second step, 'Choose Target', is highlighted with a red box. The progress bar at the top shows five steps: 1. Choose Source, 2. Choose Target, 3. Mapping Source and Target, 4. Partition Configuration, and 5. Configuration. The main content area has two sections: 'Please select schema' and 'Select attributes'. In 'Please select schema', a dropdown menu is open showing 'default' and 'demo\_tgt', with 'demo\_tgt' selected. In 'Select attributes', a table titled 'View schema: default.demo\_tgt' lists columns: 'id' (bigint), 'age' (int), and 'desc' (string). The 'id' column is selected with a checkbox. A 'Next' button is at the bottom right.

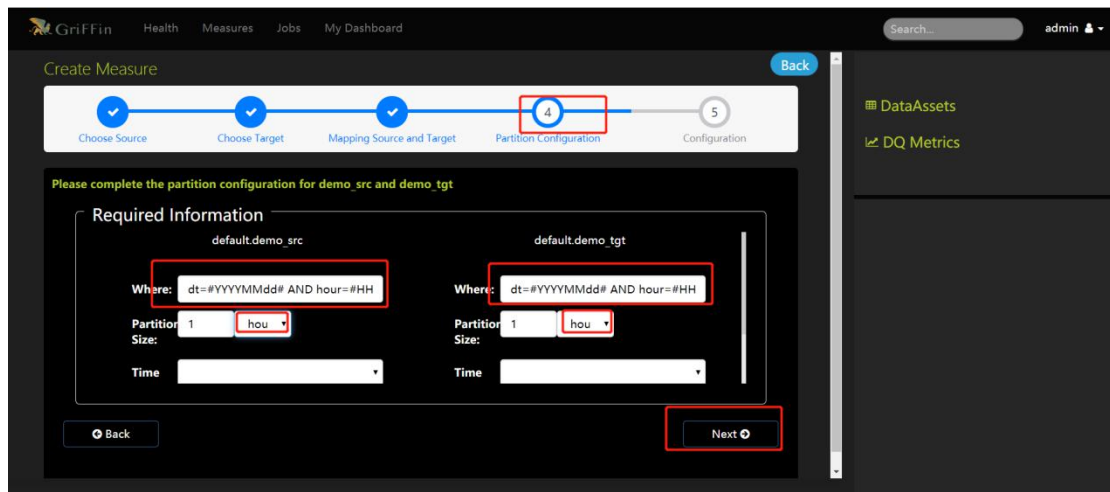
### 3.2.5 选择条件



The screenshot shows the 'Create Measure' wizard in the Griffin interface. The third step, 'Mapping Source and Target', is highlighted with a red box. The progress bar at the top shows five steps: 1. Choose Source, 2. Choose Target, 3. Mapping Source and Target, 4. Partition Configuration, and 5. Configuration. The main content area has a section titled 'Map the fields'. It contains a table with two columns: 'Target Fields' and 'Source Fields'. The 'Target Fields' column contains 'default.demo\_tgt.id'. The 'Source Fields' column contains 'default.demo\_src.id'. A 'Map To' button is between them. Below the table, there is an 'Accuracy Calculation Formula as Below:' section. A 'Next' button is at the bottom right.

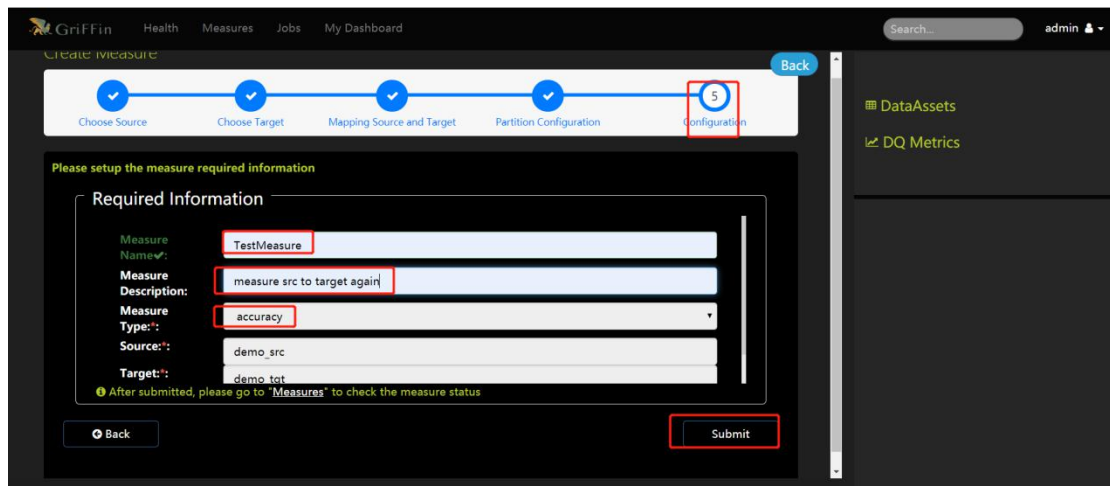


### 3.2.6 选择时间格式和分区尺度



The screenshot shows the 'Partition Configuration' step (4) of the 'Create Measure' wizard. The wizard progress bar at the top indicates the current step. The main form area is titled 'Please complete the partition configuration for demo\_src and demo\_tgt'. It contains two columns of configuration for 'default.demo\_src' and 'default.demo\_tgt'. Each column has a 'Where' clause, 'Partition Size', and 'Time' field. The 'Where' clause for both is 'dt=#YYYYMMdd# AND hour=#HH'. The 'Partition Size' is set to '1' and the unit is 'hou'. The 'Time' field is empty. There are 'Back' and 'Next' buttons at the bottom.

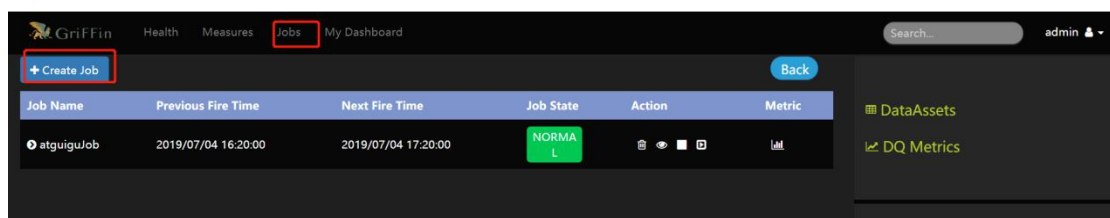
### 3.2.7 添加 Measure 名称和描述






The screenshot shows the 'Configuration' step (5) of the 'Create Measure' wizard. The main form area is titled 'Please setup the measure required information'. It contains a 'Required Information' section with fields for 'Measure Name', 'Measure Description', 'Measure Type', 'Source', and 'Target'. The 'Measure Name' is 'TestMeasure', 'Measure Description' is 'measure src to target again', 'Measure Type' is 'accuracy', 'Source' is 'demo\_src', and 'Target' is 'demo\_tgt'. There are 'Back' and 'Submit' buttons at the bottom. A note at the bottom says 'After submitted, please go to Measures to check the measure status'.

## 3.3 UI 创建 Job

### 3.3.1 新建一个 Job

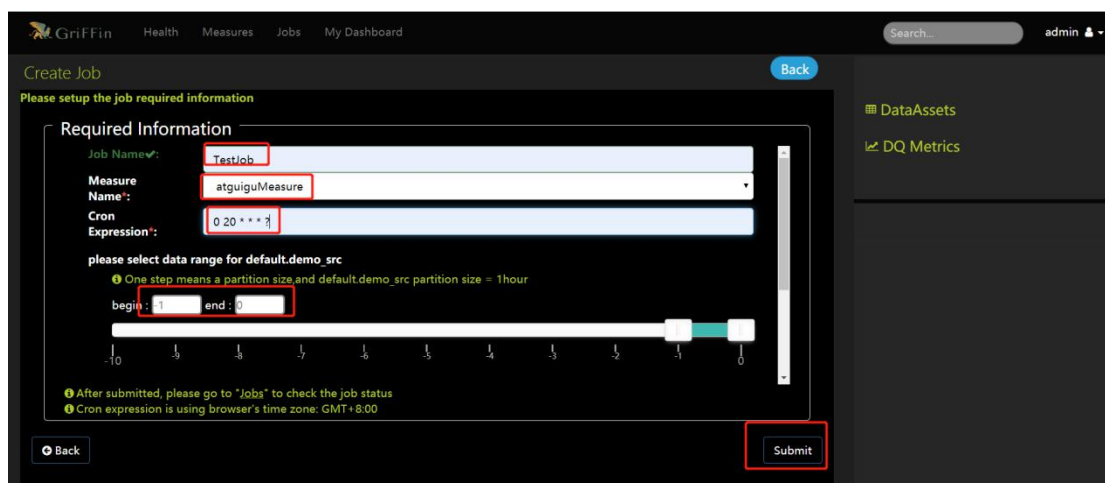


The screenshot shows the 'Jobs' page in the Griffin UI. The 'Jobs' tab is selected in the top navigation bar. A '+ Create Job' button is visible. Below it is a table listing jobs.

Job Name	Previous Fire Time	Next Fire Time	Job State	Action	Metric
atguiguJob	2019/07/04 16:20:00	2019/07/04 17:20:00	NORMAL	   	

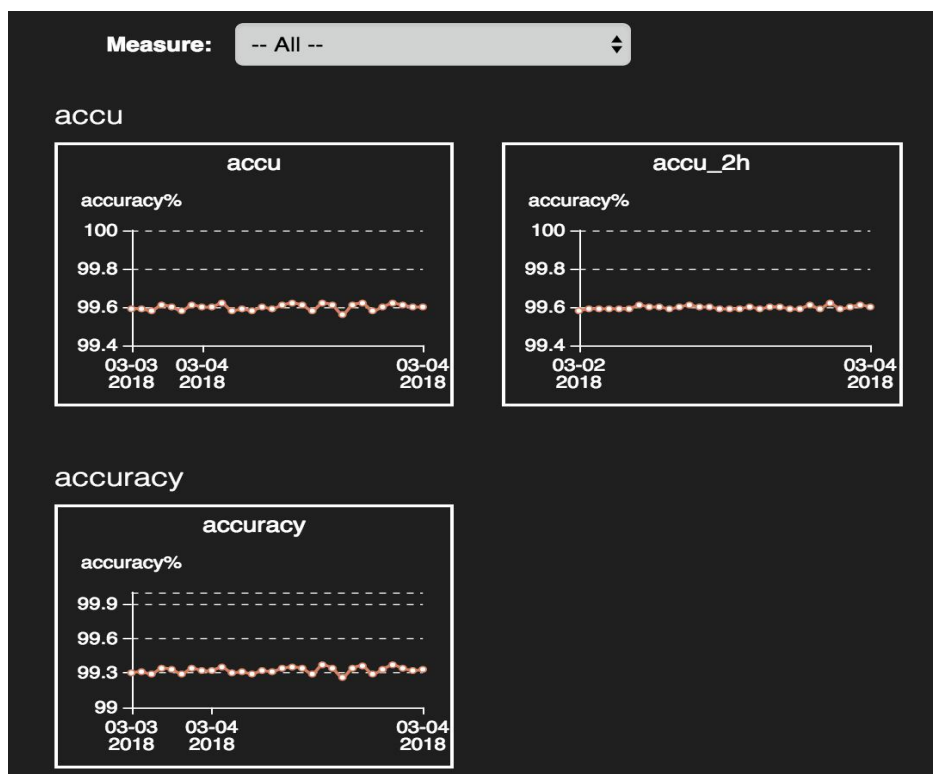


### 3.3.2 与 Measure 结合并调度任务执行

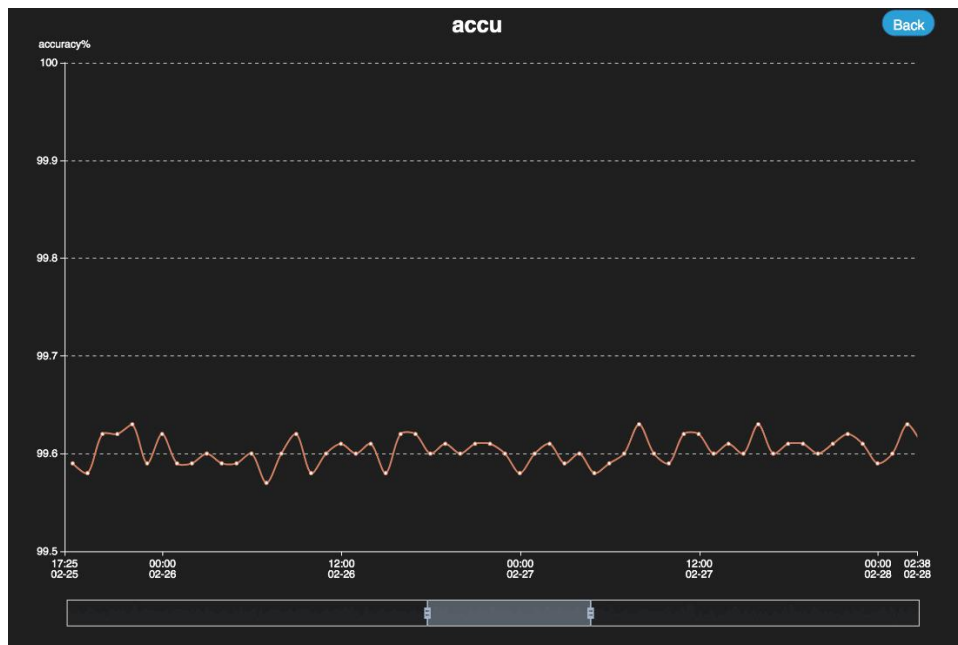


The screenshot shows the 'Create Job' interface in the Griffin dashboard. The 'Required Information' section includes fields for 'Job Name' (TestJob), 'Measure Name' (atguiguMeasure), and 'Cron Expression' (0 20 \* \* \* ?). Below these, there is a section for selecting a data range for 'default.demo.src', with 'begin' and 'end' fields set to 1 and 0 respectively. A 'Submit' button is visible at the bottom right.

### 3.3.3 查看运行结果单击“DQ 指标”



单击放大图片



### 3.4 手动定制规则运行 Job

#### 3.4.1 准确度度量 Accuracy Measure

1) 创建 json 文件制定相应规则

在/opt/module/griffin-master/measure/target 目录下创建文件 dq.json

vim dq.json

```
{
  "name": "batch_accu",
  "process.type": "batch",
  "data.sources": [
    {
      "name": "src",
      "baseline": true,
      "connectors": [
        {
          "type": "hive",
          "version": "2.3",
          "config": {
            "database": "gmall",
            "table.name": "dwd_order_info"
          }
        }
      ]
    }
  ], {
    "name": "tgt",
    "connectors": [
      {
        "type": "hive",
        "version": "2.3",
        "config": {
          "database": "gmall",
```

```
        "table.name": "dws_user_action"
    }
}
],
"evaluate.rule": {
    "rules": [
        {
            "dsl.type": "griffin-dsl",
            "dq.type": "accuracy",
            "out.dataframe.name": "accu",
            "rule": "src.id = tgt.user_id ",
            "details": {
                "source": "src",
                "target": "tgt",
                "miss": "miss_count",
                "total": "total_count",
                "matched": "matched_count"
            },
            "out": [
                {
                    "type": "metric",
                    "name": "accu"
                },
                {
                    "type": "record",
                    "name": "missRecords"
                }
            ]
        }
    ]
},
"sinks": ["CONSOLE", "HDFS"]
}
```

选项: "rule": "src.id = tgt.id AND src.age = tgt.age AND src.desc = tgt.desc"

rule 制定匹配规则如 src 表的 id 字段对应 tgt 表 id 字段

当前规则是 id age desc 三者字段相同，这个规则就是接下来对应 job 所需监控的对应关系

2) 在/opt/module/griffin-master/measure/target 目录下创建文件 env.json

vim env.json

```
{
  "spark": {
    "log.level": "WARN"
  },
  "sinks": [
    {
      "type": "console"
    },
    {
      "type": "hdfs",
      "config": {
        "path": "hdfs://hadoop102:9000/griffin/persist"
      }
    },
    {
      "type": "elasticsearch",
      "config": {
```

```

    "method": "post",
    "api": "http://hadoop102:9200/griffin/accuracy"
  }
}
]
}

```

分别指定 hdfs 结果存储路径和 elasticsearch 路径

两个 json 文件编写完毕后 指定 /opt/module/griffin-0.4.0/measure/target 该路径下的 griffin-measure.jar 运行 Spark 任务

### (3) Spark 提交脚本命令

```

[atguigu@hadoop102 target]$ spark-submit --class org.apache.griffin.measure.Application
--master yarn --deploy-mode client --driver-memory 1g --executor-memory 1g
--num-executors 1 griffin-measure.jar env.json dq.json

```

```

[1566808142747] batch_accu start: application_1566800981025_0011
batch_accu [1566808142747] metrics:
{"name": "batch_accu", "tmst": 1566808142747, "value": {"total_count": 250000, "miss_count": 980, "matched_count": 249020, "matchedFraction": 0.99608}}
[1566808142747] 1566808167917: process using time: 25170 ms
[1566808142747] batch_accu finish

```

跑完任务之后，可以看到对比结果总条数 250000 未匹配上的数有 980 条，匹配上的数据有 249020 条，匹配分数为 0.99608。

也可以去 HDFS 上查看结果

### Browse Directory

/griffin/persist/batch_accu/1566808142747								Go!
Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
-rw-r--r--	atguigu	supergroup	0 B	2019/8/26 下午4:29:28	1	128 MB	._FINISH	
-rw-r--r--	atguigu	supergroup	137 B	2019/8/26 下午4:29:27	1	128 MB	._LOG	
-rw-r--r--	atguigu	supergroup	139 B	2019/8/26 下午4:29:27	1	128 MB	._METRICS	
-rw-r--r--	atguigu	supergroup	30 B	2019/8/26 下午4:29:09	1	128 MB	._START	
-rw-r--r--	atguigu	supergroup	83.26 KB	2019/8/26 下午4:29:27	1	128 MB	missRecords	

Hadoop, 2015.

根据任务 id 号，下载 missRecords 文件，里面保存的所有没匹配上的数据，方便分析问题。

```
missRecords - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
{"id":124,"age":1302,"desc":"1302","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1302,"desc":"1302","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1935,"desc":"1935","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1935,"desc":"1935","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1065,"desc":"1065","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1065,"desc":"1065","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1798,"desc":"1798","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1798,"desc":"1798","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1404,"desc":"1404","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1404,"desc":"1404","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1585,"desc":"1585","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1585,"desc":"1585","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1585,"desc":"1585","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1585,"desc":"1585","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1585,"desc":"1585","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1619,"desc":"1619","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1619,"desc":"1619","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1714,"desc":"1714","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1714,"desc":"1714","dt":"20190820","hour":"14","__tmst":1566808142747}
{"id":124,"age":1714,"desc":"1714","dt":"20190826","hour":"15","__tmst":1566808142747}
{"id":124,"age":1714,"desc":"1714","dt":"20190826","hour":"15","__tmst":1566808142747}
```

### 3.4.2 性能分析度量 Profiling Measure

如果想查看最大值、总个数、空值个数、最大长度等值那就得创建 profiling measure

#### 1) 创建 json 文件定制规则

vim dq2.json

```
{
  "name": "batch_prof",
  "process.type": "batch",
  "data.sources": [
    {
      "name": "src",
      "baseline": true,
      "connectors": [
        {
          "type": "hive",
          "version": "2.2",
          "config": {
            "database": "tmp",
            "table.name": "demo_tgt"
          }
        }
      ]
    }
  ],
  "evaluate.rule": {
    "rules": [
      {
        "dsl.type": "griffin-dsl",
        "dq.type": "profiling",
        "out.dataframe.name": "prof",
        "rule": "src.id.count() AS id_count, src.age.max() AS age_max, src.desc.length().max() AS desc_length_max",
        "out": [
```

```
{
  {
    "type": "metric",
    "name": "prof"
  }
]
},
"sinks": ["CONSOLE", "HDFS"]
}
```

同样 rule 选项编写规则，如当前规则统计 id 总个数、age 最大值和 desc 最大长度。

具体 rule 规则可查看官网

<https://github.com/apache/griffin/blob/master/griffin-doc/measure/measure-batch-sample.md>

## 2) 编辑 env.json

vim env.json

```
{
  "spark": {
    "log.level": "WARN"
  },
  "sinks": [
    {
      "type": "console"
    },
    {
      "type": "hdfs",
      "config": {
        "path": "hdfs://hadoop102:9000/griffin/persist"
      }
    },
    {
      "type": "elasticsearch",
      "config": {
        "method": "post",
        "api": "http://hadoop102:9200/griffin/accuracy"
      }
    }
  ]
}
```

分别指定 hdfs 结果存储路径和 elasticsearch 路径

两个 json 文件编写完毕后 指定 /opt/module/griffin-0.4.0/measure/target 该路径下的 griffin-measure.jar 运行 spark 任务

## 3) Spark 提交脚本命令

```
spark-submit --class org.apache.griffin.measure.Application --master yarn --deploy-mode client --driver-memory 1g --executor-memory 1g --num-executors 1 griffin-measure.jar env.json dq2.json
```

```
[1566809453110] batch_prof start: application_1566800981025_0012
batch_prof [1566809453110] metrics:
{"name":"batch_prof","tmst":1566809453110,"value":{"id_count":250000,"age_max":1000,"desc_length_max":4}}
```

跑完任务后，显示结果 id 个数一共 250000 条，age 最大值 1000，desc 长度最大值 4，对应 hdfs 路径，env.json 路径加时间戳。

Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities

### Browse Directory

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	atguigu	supergroup	0 B	2019/8/26 下午4:51:21	1	128 MB	<a href="#">_FINISH</a>
-rw-r--r--	atguigu	supergroup	137 B	2019/8/26 下午4:51:21	1	128 MB	<a href="#">_LOG</a>
-rw-r--r--	atguigu	supergroup	105 B	2019/8/26 下午4:51:21	1	128 MB	<a href="#">_METRICS</a>
-rw-r--r--	atguigu	supergroup	30 B	2019/8/26 下午4:51:08	1	128 MB	<a href="#">_START</a>

Hadoop, 2015.



\_LOG - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
===== log of Mon Aug 26 16:50:53 CST 2019 =====
```

```
--- Mon Aug 26 16:51:21 CST 2019 ---
```

```
process using time: 28173 ms
```



\_METRICS - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
{"name":"batch_prof","tmst":1566809453110,"value":{"id_count":250000,"age_max":1000,"desc_length_max":4}}
```



\_START - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
application_1566800981025_0012
```



## 3.5 Rule 规则编写

### 3.5.1 空值个数查询

```

"name": "batch_prof",
"process.type": "batch",
"data.sources": [
  {
    "name": "src",
    "baseline": true,
    "connectors": [
      {
        "type": "hive",
        "version": "2.2",
        "config": {
          "database": "tmp",
          "table.name": "demo_tgt"
        }
      }
    ]
  }
],
"evaluate.rule": {
  "rules": [
    {
      "dsl.type": "griffin-dsl",
      "dq.type": "profiling",
      "out_dataframe_name": "prof",
      "rule": "select count(id) from src where id is null",
      "out": [
        {
          "type": "metric",
          "name": "prof"
        }
      ]
    }
  ]
},
"sinks": ["CONSOLE", "HDFS"]

```

在 rule 里可以写 sql 比如当前 rule 中统计 id 为空的个数，注意查询时，不是查询表名而是查询 name 名称

```

batch_prof [1566810731958] metrics:
{"name":"batch_prof","tmst":1566810731958,"value":{"count":0}}

```



### 3.5.2 去重个数统计

```
{
  "name": "batch_prof",
  "process.type": "batch",
  "data.sources": [
    {
      "name": "src",
      "baseline": true,
      "connectors": [
        {
          "type": "hive",
          "version": "2.2",
          "config": {
            "database": "tmp",
            "table.name": "demo_tgt"
          }
        }
      ]
    }
  ],
  "evaluate.rule": {
    "rules": [
      {
        "dsl.type": "griffin-dsl",
        "dq.type": "profiling",
        "out_dataframe_name": "prof",
        "rule": "select distinct count(id) from src ",
        "out": [
          {
            "type": "metric",
            "name": "prof"
          }
        ]
      }
    ]
  },
  "sinks": ["CONSOLE", "HDFS"]
}
```

```
batch_prof [1566811098646] metrics:
{"name":"batch_prof","tmst":1566811098646,"value":{"count":250000}}
```

### 3.5.3 最大值、最小值

```
{
  "name": "batch_prof",
  "process.type": "batch",
  "data.sources": [
    {
      "name": "src",
      "baseline": true,
      "connectors": [
        {
          "type": "hive",
          "version": "2.2",
          "config": {
            "database": "tmp",
            "table.name": "demo_tgt"
          }
        }
      ]
    }
  ],
  "evaluate.rule": {
    "rules": [
      {
        "dsl.type": "griffin-dsl",
        "dq.type": "profiling",
        "out.dataframe.name": "prof",
        "rule": "select max(age),min(age) from src ",
        "out": [
          {
            "type": "metric",
            "name": "prof"
          }
        ]
      }
    ]
  },
  "sinks": ["CONSOLE", "HDFS"]
},
{"name": "batch_prof", "tmst": 1566815829848, "value": {"max": 1000, "min": 1}}
```

### 3.5.4 排序

```
{
  "name": "batch_prof",
  "process.type": "batch",
  "data.sources": [
    {
      "name": "src",
      "baseline": true,
      "connectors": [
        {
          "type": "hive",
          "version": "2.2",
          "config": {
            "database": "tmp",
            "table.name": "demo_tgt"
          }
        }
      ]
    }
  ],
  "evaluate.rule": {
    "rules": [
      {
        "dsl.type": "griffin-dsl",
        "dq.type": "profiling",
        "out.dataframe.name": "prof",
        "rule": "select age from src order by age desc limit 10 ",
        "out": [
          {
            "type": "metric",
            "name": "prof"
          }
        ]
      }
    ]
  },
  "sinks": ["CONSOLE", "HDFS"]
}
```

```
batch_prof [1568816648178] metrics:  
{"name":"batch_prof","task":"1568816648178","value":{"prof":[{"age":1000}, {"age":1000}, {"age":1000}, {"age":1000}, {"age":1000}, {"age":1000}, {"age":1000}, {"age":1000}, {"age":1000}, {"age":1000}],  
1568816648178.15696166742402.nprocess.usine.time: 26664 ms
```

### 3.5.5 范围查询

```
{
  "name": "batch_prof",
  "process.type": "batch",
  "data.sources": [
    {
      "name": "src",
      "baseline": true,
      "connectors": [
        {
          "type": "hive",
          "version": "2.2",
          "config": {
            "database": "tmp",
            "table.name": "demo_tgt"
          }
        }
      ]
    }
  ],
  "evaluate.rule": {
    "rules": [
      {
        "dsl.type": "griffin-dsl",
        "dq.type": "profiling",
        "out.dataframe.name": "prof",
        "rule": "select count(*) from src where age between 0 and 50 ",
        "out": [
          {
            "type": "metric",
            "name": "prof"
          }
        ]
      }
    ]
  },
  "sinks": ["CONSOLE", "HDFS"]
}
```

```
batch_prof [1566820801985] metrics:
{"name":"batch_prof","tmst":1566820801985,"value":{"count":12500}}
```