# 1 安装sqoop2

## 1.1 下载sqoop：

官网下载地址：<http://www.apache.org/dyn/closer.lua/sqoop/1.99.6>

解压： 上传到/opt/

tar –xvf sqoop-1.99.6-bin-hadoop200.tar.gz

## 1.2 环境变量

Vim /etc/profile

export SQOOP\_HOME=/opt/sqoop-1.99.6-bin-hadoop200

export CATALINA\_BASE=$SQOOP\_HOME/server

export PATH=.:$SQOOP\_HOME/bin:$PATH

export LOGDIR=$SQOOP\_HOME/logs

保存配置：source /etc/profile

## 1.3 驱动jar

把mysql的驱动包mysql-connector-java-5.1.10-bin.jar放到${SQOOP\_HOME}/server/lib/里面

避免JAR包版本冲突

下载地址  
  
去http://archive.apache.org/dist/上下载db-derby-10.11.1.1-bin.tar.gz

1. cd /usr/sqooserver/webapps/sqoop/WEB-INF/lib/
2. mv derby-10.8.2.2.jar derby-10.8.2.2.jar.bak
3. mv derbyclient-10.8.2.2.jar derbyclient-10.8.2.2.jar.bak
5. cp /home/\*\*\*\*/Downloads/db-derby-10.11.1.1-bin/lib/derby.jar /usr/sqoop/server/webapps/sqoop/WEB-INF/lib/
6. cp /home/\*\*\*\*/Downloads/db-derby-10.11.1.1-bin/lib/derbyclient.jar /usr/sqoop/server/webapps/sqoop/WEB-INF/lib/

## 1.4 配置catalina.properties

配置${SQOOP\_HOME}/server/conf/catalina.properties 文件

common.loader=${catalina.base}/lib,${catalina.base}/lib/\*.jar,${catalina.home}/lib,${catalina.home}/lib/\*.jar,${catalina.home}/../lib/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/common/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/common/lib/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/hdfs/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/hdfs/lib/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/mapreduce/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/mapreduce/lib/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/tools/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/tools/lib/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/yarn/\*.jar,

/opt/hadoop-2.7.3/share/hadoop/yarn/lib/\*.jar,

/opt/apache-hive-1.2.1-bin/lib/\*.jar

## 1.5 配置sqoop.properties

配置${SQOOP\_HOME}/server/conf/sqoop.properties 文件

org.apache.sqoop.submission.engine.mapreduce.configuration.directory=/opt/hadoop-2.7.3/etc/hadoop/

org.apache.sqoop.repository.jdbc.url=jdbc:derby:@BASEDIR@/repository/**sqoop**;create=true

  这里的sqoop是事先在**[MySQL](http://lib.csdn.net/base/mysql" \o "MySQL知识库" \t "http://blog.csdn.net/huyangshu87/article/details/_blank)**这边创建的**[数据库](http://lib.csdn.net/base/mysql" \o "MySQL知识库" \t "http://blog.csdn.net/huyangshu87/article/details/_blank)**，并赋予了权限：

1. create database sqoop ;
2. create user sqoop identified by '123456';
3. grant all privileges on sqoop.\* to sqoop;
4. flush privileges;

grant all privileges on \*.\* to 'root'@'%' identified by 'root' with grant option;

flush privileges;

## 1.6 修改hadoop的yarn-site.xml文件

1. <property>
2. <name>yarn.nodemanager.aux-services</name>
3. <value>mapreduce\_shuffle</value>
4. </property>
5. <property>
6. <name>yarn.nodemanager.aux-services.mapreduce\_shuffle.class</name>
7. <value>org.apache.hadoop.mapred.ShuffleHandler</value>
8. </property>

## 1.7 启动sqoop

首先启动hadoop集群，再启动sqoop

sqoop.sh server start

停止：

   Sqoop2-server  stop

检查sqoop2配置是否正确

Jps

sqoop2-tool verify  
    注意：先启动服务后再去执行sqoop2-tool verify    否则会报错：说什么连接不上

启动sqoop客户端：  
sqoop2-shell

Client连接到服务端：set server --host 127.0.0.1 --port 12000 --webapp sqoop

可用浏览器在12000端口查看

# 2 使用sqoop 1.99.6

## 2.1 常用命令

### 2.1.1show相关

查看服务器配置：

sqoop:000> show version --all

查看sqoop的所有连接：

sqoop:000> show connector

### 2.1.2 link相关：

查看 当前的所有link，在向hdfs导入导出数据时，需要依赖以上四个连接创建link：

sqoop:000> show link

修改link：update link -lid 1

删除link：

sqoop:000> delete link --lid 1

### 2.1.2 job相关：

创建job( 从link 2 向link 1导入数据)：

sqoop:001> create job --f 2 --t 1

 查看job：

sqoop:000> show job

修改link：update job jid 2

启动job：

sqoop:000> start job --jid 1

查看导入状态：

sqoop:000> status job --jid 1

删除job：

sqoop:000> delete job --jid 1

### 2.1.3 Set相关

为客户端配置服务器：

sqoop:000> set server --host node3--port 12000 --webapp sqoop

客户端属性设置：

set option --name poll-timeout --value 20000 --超时时间

## 2.2 使用实例：

### 2.2.1 创建hdfs连接

1. sqoop:000> create link --cid 3
2. Creating link **for** connector with id 1
3. Please fill following values to create **new** link object
4. Name: hdfs\_link --设置连接名称
5. Link configuration
6. HDFS URI: hdfs://192.168.18.107:9000 --HDFS访问地址
7. New link was successfully created with validation status OK and persistent id 2

### 2.2.2 创建mysql链接

1. sqoop:000> create link --cid 4
2. Creating link **for** connector with id 2
3. Please fill following values to create **new** link object
4. Name: mysq\_link  --设置连接名称
5. Link configuration
6. JDBC Driver Class: com.mysql.jdbc.Driver --驱动
7. JDBC Connection String: jdbc:mysql://hadoop000:3306/hive   --JDBC连接
8. Username: root --数据库用户
9. Password: root --数据库密码
10. JDBC Connection Properties:
11. There are currently 0 values in the map:
12. entry#protocol=tcp --之后按回车
13. New link was successfully created with validation status OK and persistent id 1

### 2.2.3 创建job

1. sqoop:000> create job -f 2 -t 1
2. Creating job **for** links with from id 1 and to id 6
3. Please fill following values to create **new** job object
4. Name: mysql\_openfire --设置 任务名称
5. FromJob configuration
6. Schema name:(Required)sqoop --库名：必填
7. Table name:(Required)sqoop --表名：必填
8. Table SQL statement:(Optional) --选填
9. Table column names:(Optional) --选填
10. Partition column name:(Optional) id --选填
11. Null value allowed **for** the partition column:(Optional) --选填
12. Boundary query:(Optional) --选填
13. ToJob configuration
14. Output format:
15. 0 : TEXT\_FILE
16. 1 : SEQUENCE\_FILE
17. Output format:
18. 0 : TEXT\_FILE
19. 1 : SEQUENCE\_FILE
20. Choose: 0 --选择文件压缩格式
21. Compression format:
22. 0 : NONE
23. 1 : DEFAULT
24. 2 : DEFLATE
25. 3 : GZIP
26. 4 : BZIP2
27. 5 : LZO
28. 6 : LZ4
29. 7 : SNAPPY
30. 8 : CUSTOM
31. Choose: 0 --选择压缩类型
32. Custom compression format:(Optional) --选填
33. Output directory:hdfs:/ns1/sqoop --HDFS存储目录（目的地）目录名必须是不存在的，hdfs会自动创建
34. Driver Config
35. Extractors: 2 --提取器
36. Loaders: 2 --加载器
37. New job was successfully created with validation status OK and persistent id 1

（Schema name，Table name）和Table SQL statement 不能同时设置，这是为了防止表名不同一造成的问题。

Table Columns name可以设置要抽取的列名。基本格式为：COL1,COL2，COL3  .因为Table Columns name的长度有一个变态的设置，就是这个字符串长度不能超过50.所以基本上设置不了几列。在这里，我通过查找源码发现了Table SQL statement的用法，其实可以说他的存在就是为了解决上面的问题。首先我们可以看到源码里sql的size为2000.所以有足够的长度。但是之前试了好多种sql的写法，发现总是不成功。于是对sqoop各种吐槽。。。

后来终于发现它的基本写法是这样子滴：SELECT ID,COL1,COL2,COL3 FROM SCHEMA.TABLE WHERE ${CONDITIONS}

那个${CONDITIONS}只是个占位符，没有什么用处，它在源码里会被替换成1=1

这样子就解决了COLUMNS设置的问题。

再来讲解一下

Partition column name 一般可设置为ID，它的作用就是为了并行处理做准备的。sqoop通过获取Partition column的最大值和最小值对数据库表进行切割，分块读取数据（这样很有可能导致任务之间的不平衡）。

最后一个问题，sqoop现在列之间的分隔符用的是“，”而不是“\t”，而且在外部已经不能手动去改了，它在代码里写死了。要改的话可以通过重新编译源代码的方式进行修改。

那个类是org.apache.sqoop.connector.hdfs.HdfsConstants

## 2.3 常见错误

sqoop 1.99.6遇到的错误

### 2.3.1、create job 时：

There are issues with entered data, please revise your input:

说明配置有错，请重新检查各输入项

### 2.3.2、create job 时：

Last value is required during incremental read

是因为**Check column:** 输入了值，所以要求输入 Last value

### 2.3.3、输入Last value后：

Size of input exceeds allowance for this input field. Maximal allowed size is -1

怎么都没办法输入正确，一个字符也不行，重新配置时把Check column 项清空就好了

### 2.3.4、start job -j 1时:

Exception has occurred during processing command  
Exception: org.apache.sqoop.common.SqoopException Message: CLIENT\_0001:Server has returned exception

根本不知道这个提示说什么，通过修改设置：查看job详情

**set option --name verbose --value true**

再次执行 start job -j 1，显示出有用的出错信息：Caused by: Exception: java.net.ConnectException Message: 拒绝连接

 在$HADOOP\_HOME/etc/hadoop/core-site.xml 检查默认端口为 9000，重新修改连到HDFS的Link配置，重新执行job，问题解决。

# 3 sqoop1

## 安装

下载：

Apache 提供的工具

安装：

要求必须有jdk 和 hadoop的支持，并且有版本要求。

上传到linux中，进行解压

sqoop可以通过JAVA\_HOME找到jdk 可以通过HADOOP\_HOME找到hadoop所以不需要做任何配置就可以工作。

需要将要连接的数据库的驱动包加入sqoop的lib目录下 1

## 2 sqoop命令

Sqoop大约有13种命令,和几种通用的参数(都支持这13种命令)，这里先列出这13种命令。

接着列出Sqoop的各种通用参数,然后针对以上13个命令列出他们自己的参数。Sqoop通用参数又分Common arguments,Incremental import arguments,Output line formatting arguments,Input parsing arguments,Hive arguments,HBase arguments,Generic Hadoop command-line arguments,下面一一说明:

sqoop 沟通hdfs和关系型数据库的桥梁，可以从hdfs导出数据到关系型数据库，也可以从关系型数据库导入数据到hdfs

### 2.1从数据库导入到hdfs:

sqoop import --connect jdbc:mysql://192.168.1.10:3306/tedu --username root --password 123 --table trade\_detail --columns 'id, account, income, expenses'

指定输出路径、指定数据分隔符

sqoop import --connect jdbc:mysql://192.168.1.10:3306/tedu --username root --password 123 --table trade\_detail --target-dir '/sqoop/td' --fields-terminated-by '\t'

指定Map数量 -m

sqoop import --connect jdbc:mysql://192.168.1.10:3306/tedu --username root --password 123 --table trade\_detail --target-dir '/sqoop/td1' --fields-terminated-by '\t' -m 2

增加where条件, 注意：条件必须用引号引起来

sqoop import --connect jdbc:mysql://192.168.1.10:3306/tedu --username root --password 123 --table trade\_detail --where 'id>3' --target-dir '/sqoop/td2'

增加query语句(使用 \ 将语句换行)

sqoop import --connect jdbc:mysql://192.168.1.10:3306/tedu --username root --password 123 --query 'SELECT \* FROM trade\_detail where id > 2 AND $CONDITIONS' --split-by trade\_detail.id --target-dir '/sqoop/td3'

注意：如果使用--query这个命令的时候，需要注意的是where后面的参数，AND $CONDITIONS这个参数必须加上

而且存在单引号与双引号的区别，如果--query后面使用的是双引号，那么需要在$CONDITIONS前加上\即\$CONDITIONS

如果设置map数量为1个时即-m 1，不用加上--split-by ${tablename.column}，否则需要加上

从数据库增量导入表数据到hdfs中

sqoop import –connect jdbc:mysql://10.28.168.109:3306/compression

–username=hadoop –password=123456 –table HADOOP\_USER\_INFO -m 1

–target-dir /user/test  –check-column id –incremental append

–last-value 3

### 2.2从hdfs导出到数据库:

sqoop export --connect jdbc:mysql://192.168.8.120:3306/tedu --username root --password 123 --export-dir '/td3' --table td\_bak -m 1 --fields-terminated-by ','

### 2.3从数据库导入到hive：

sqoop import --connect jdbc:mysql://192.168.1.10:3306/tedu --username root --password 123 --table trade\_detail --hive-import -hive-table hivetab1 -m 1

从hive导出数据到关系型数据库

就是将hive在hdfs文件夹下的文件导出到mysql中，参考从hdfs到处数据到关系型数据库的过程。

====================================================================

sqoop export --connect jdbc:mysql://192.168.242.1:3306/zebra --username root --password root --export-dir '/user/hive/warehouse/zebra.db/d\_h\_http\_apptype/000000\_0' --table d\_h\_http\_apptype -m 1 --fields-terminated-by '|'

### 2.4 列出mysql数据库中的所有数据库

 sqoop list-databases --connect jdbc:**[MySQL](http://lib.csdn.net/base/mysql" \o "MySQL知识库" \t "http://blog.csdn.net/duan19056/article/details/_blank)**://localhost:3306/ -username dyh -password 000000

连接mysql并列出数据库中的表

sqoop list-tables --connect jdbc:mysql://localhost:3306/test --username dyh --password 000000

### 2.5将关系型数据的表结构复制到hive中

sqoop create-**[Hive](http://lib.csdn.net/base/hive" \o "Hive知识库" \t "http://blog.csdn.net/duan19056/article/details/_blank)**-table --connect jdbc:mysql://localhost:3306/test --table users --username dyh --password 000000 --hive-table users  --fields-terminated-by "\0001"  --lines-terminated-by "\n";

## 3 Sqoop原理

Sqoop在import时，需要制定split-by参数。Sqoop根据不同的split-by参数值来进行切分,然后将切分出来的区域分配到不同map中。每个map中再处理数据库中获取的一行一行的值，写入到HDFS中。同时split-by根据不同的参数类型有不同的切分方法，如比较简单的int型，Sqoop会取最大和最小split-by字段值，然后根据传入的num-mappers来确定划分几个区域。 比如select max(split\_by),min(split-by) from得到的max(split-by)和min(split-by)分别为1000和1，而num-mappers为2的话，则会分成两个区域(1,500)和(501-100),同时也会分成2个sql给2个map去进行导入操作，分别为select XXX from table where split-by>=1 and split-by<500和select XXX from table where split-by>=501 and split-by<=1000。最后每个map各自获取各自SQL中的数据进行导入工作。