Sqoop1、Sqoop2功能差异及使用示例

目录

[一、功能差异- *sqoop1* 和*sqoop2* 1](#_Toc428540427)

[二、Sqoop1 2](#_Toc428540428)

[2.1 介绍 2](#_Toc428540429)

[三、安装 Sqoop 2](#_Toc428540430)

[四、使用 3](#_Toc428540431)

[4.1 命令说明 3](#_Toc428540432)

[4.2 MySQL导入数据到Hdfs 3](#_Toc428540433)

[4.3 将数据从hdfs导出到mysql中 5](#_Toc428540434)

[4.4 创建 hive 表 5](#_Toc428540435)

[4.5 导入数据到hive 6](#_Toc428540436)

[五、sqoop2 6](#_Toc428540437)

[5.1 介绍 6](#_Toc428540438)

[六、安装 Sqoop2 7](#_Toc428540439)

[七、sqoop2 使用 7](#_Toc428540440)

[7.1 命令说明 7](#_Toc428540441)

[7.2 MySQL导入数据到Hdfs 8](#_Toc428540442)

[7.2.1创建Job 对象 9](#_Toc428540443)

[7.2.2启动job 11](#_Toc428540444)

[7.2.3查看数据是否导入 11](#_Toc428540445)

[7.3 Hdfs导出数据到MySQL 11](#_Toc428540446)

# 一、功能差异- *sqoop1* 和*sqoop2*

|  |  |  |
| --- | --- | --- |
| **功能** | **Sqoop 1** | **Sqoop 2** |
| 用于所有主要 RDBMS 的连接器 | 支持 | 不支持  **解决办法：**使用已在以下数据库上执行测试的通用 JDBC 连接器：Microsoft SQL Server、PostgreSQL、MySQL 和 Oracle。  此连接器应在任何其它符合 JDBC 要求的数据库上运行。但是，性能可能无法与 Sqoop 中的专用连接器相比 |
| Kerberos 安全集成 | 支持 | 不支持 |
| 数据从 RDBMS 传输至 Hive 或 HBase | 支持 | 不支持  **解决办法：**按照此两步方法操作。   1. 将数据从 RDBMS 导入 HDFS 2. 在 Hive 中使用相应的工具和命令（例如 LOAD DATA 语句），手动将数据载入 Hive 或 HBase |
| 数据从 Hive 或 HBase 传输至 RDBMS | 不支持  **解决办法：**按照此两步方法操作。   1. 从 Hive 或 HBase 将数据提取至 HDFS（作为文本或 Avro 文件） 2. 使用 Sqoop 将上一步的输出导出至 RDBMS | 不支持  按照与 Sqoop 1 相同的解决方法操作 |

# 二、Sqoop1

## 2.1 介绍

sqoop架构非常简单，其整合了Hive、Hbase和Oozie，通过map-reduce任务来传输数据，从而提供并发特性和容错。

sqoop主要通过JDBC和关系数据库进行交互。理论上支持JDBC的database都可以使用sqoop和hdfs进行数据交互

# 三、安装 Sqoop

使用rpm安装即可

yum install sqoop sqoop-metastore -y

安装完之后需要下载 mysql jar 包到 sqoop 的 lib 目录。

mysql-connector-java-5.1.32-bin.jar

如果使用默认的mysql jar会出现以下错误

[ERROR Streaming result set com.mysql.jdbc.RowDataDynamic@1d5a7f6 is still active. No statements may](http://blog.csdn.net/wind520/article/details/39127843)

# 四、sqoop1使用

## 4.1 命令说明

# sqoop help

Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.

Please set $ACCUMULO\_HOME to the root of your Accumulo installation.

15/08/27 15:58:24 INFO sqoop.Sqoop: Running Sqoop version: 1.4.5-cdh5.4.3

usage: sqoop COMMAND [ARGS]

Available commands:

codegen Generate code to interact with database records

create-hive-table Import a table definition into Hive

eval Evaluate a SQL statement and display the results

export Export an HDFS directory to a database table

help List available commands

import Import a table from a database to HDFS

import-all-tables Import tables from a database to HDFS

import-mainframe Import datasets from a mainframe server to HDFS

job Work with saved jobs

list-databases List available databases on a server

list-tables List available tables in a database

merge Merge results of incremental imports

metastore Run a standalone Sqoop metastore

version Display version information

See 'sqoop help COMMAND' for information on a specific command.

## 4.2 MySQL导入数据到Hdfs

测试sqoop连接mysql

# /usr/lib/sqoop/bin/sqoop-list-databases --connect jdbc:mysql://20.12.20.12:3306/ --username root

Warning: /usr/lib/sqoop/bin/../../accumulo does not exist! Accumulo imports will fail.

Please set $ACCUMULO\_HOME to the root of your Accumulo installation.

15/08/27 16:06:35 INFO sqoop.Sqoop: Running Sqoop version: 1.4.5-cdh5.4.3

15/08/27 16:06:35 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.

information\_schema

mysql

sqoop

test

看到了mysql，test库，则连接没问题

使用 sqoop import 命令可以从关系数据库导入数据到 hdfs

sqoop import --connect jdbc:mysql://20.12.20.12:3306/sqoop --username root --table test -m 1 --target-dir /user/hive/result

note：mysql jdbc url 请使用 ip 地址

如果重复执行，会提示目录已经存在，可以手动删除

如果不指定 --target-dir，导入到用户家目录下的 test 目录

你还可以指定其他的参数：

--append 将数据追加到hdfs中已经存在的dataset中已经存在的dataset中。使用该参数，sqoop将把数据先导入到一个临时目录中，然后重新给文件命 名到一个正式的目录中，以避免和该目录中已存在的文件重名。

--as-avrodatafile 将数据导入到一个Avro数据文件中

--as-sequencefile 将数据导入到一个sequence文件中

--as-textfile 将数据导入到一个普通文本文件中，生成该文本文件后，可以在hive中通过sql语句查询出结果。

--boundary-query <statement>边界查询，也就是在导入前先通过SQL查询得到一个结果集，然后导入的数据就是该结果集内的数据，格式如：--boundary-query 'select id,no from t where id = 3'，表示导入的数据为id=3的记录，或者 select min(<split-by>), max(<split-by>) from <table name>，注意查询的字段中不能有数据类型为字符串的字段，否则会报错

--columns<col,col> 指定要导入的字段值，格式如：--columns id,username

--direct 直接导入模式，使用的是关系数据库自带的导入导出工具。官网上是说这样导入会更快

--direct-split-size 在使用上面direct直接导入的基础上，对导入的流按字节数分块，特别是使用直连模式从PostgreSQL导入数据的时候，可以将一个到达设定大小的文件分为几个独立的文件。

--inline-lob-limit 设定大对象数据类型的最大值

-m,--num-mappers 启动N个map来并行导入数据，默认是4个，最好不要将数字设置为高于集群的节点数

--query，-e <sql> 从查询结果中导入数据，该参数使用时必须指定–target-dir、–hive-table，在查询语句中一定要有where条件且在where条件中需要包含 \$CONDITIONS，示例：--query 'select \* from t where \$CONDITIONS ' --target-dir /tmp/t –hive-table t

--split-by <column> 表的列名，用来切分工作单元，一般后面跟主键ID

--target-dir <dir> 指定hdfs路径

--delete-target-dir 删除目标目录

--warehouse-dir <dir> 与 --target-dir 不能同时使用，指定数据导入的存放目录，适用于hdfs导入，不适合导入hive目录

--where 从关系数据库导入数据时的查询条件，示例：--where "id = 2"

-z,--compress 默认情况下数据是没被压缩的，通过该参数可以使用gzip压缩算法对数据进行压缩，适用于SequenceFile, text文本文件, 和Avro文件

--compression-codec Hadoop压缩编码，默认是gzip

--table <table-name> 关系数据库表名，数据从该表中获取

eg：

sqoop import --connect jdbc:mysql://20.12.20.11:3306/hive --username root --password 123456 --table TBLS --columns "tbl\_id,create\_time" --where "tbl\_id > 1" --target-dir /user/hive/result

查看hdfs中数据

# hadoop fs -cat /user/hive/result/part-m-00000

2,1440639469

## 4.3 将数据从hdfs导出到mysql中

hdfs中的数据

# hadoop fs -cat /user/root/test.txt

1,aaaaaa,bbbb

2,cccccc,dddd

3,eeeeee,ffff

执行下面的命令会将 hdfs 中的数据导入到 mysql 中

#sqoop export --connect jdbc:mysql://20.12.20.12:3306/sqoop --username root --table yonyou --export-dir '/user/root/test.txt' --fields-terminated-by ','

查看结果

mysql> select \* from yonyou;

+------+--------+------+

| id | va01 | va02 |

+------+--------+------+

| 1 | aaaaaa | bbbb |

| 2 | cccccc | dddd |

| 3 | eeeeee | ffff |

+------+--------+------+

3 rows in set (0.00 sec)

## 4.4 创建 hive 表

生成与关系数据库表的表结构对应的HIVE表

#sqoop create-hive-table --connect jdbc:mysql://20.12.20.12:3306/sqoop --username root --table yonyou

note：

--hive-home <dir> Hive的安装目录，可以通过该参数覆盖掉默认的hive目录

--hive-overwrite 覆盖掉在hive表中已经存在的数据

--create-hive-table 默认是false，如果目标表已经存在了，那么创建任务会失败

--hive-table 后面接要创建的hive表

--table 指定关系数据库表名

## 4.5 导入数据到hive

执行下面的命令会将 mysql 中的数据导入到 hdfs 中，然后创建一个hive 表，最后再将 hdfs 上的文件移动到 hive 表的目录下面。

#sqoop import --connect jdbc:mysql://20.12.20.11:3306/hive --username root --password 123456 --table TBLS --fields-terminated-by "\t" --lines-terminated-by "\n" --hive-import --hive-overwrite --create-hive-table --hive-table dw\_srclog.TBLS --delete-target-dir

notice:

可以在 hive 的表名前面指定数据库名称

可以通过 --create-hive-table 创建表，如果表已经存在则会执行失败

接下来可以查看 hive 中的数据：

# hive -e 'select \* from dw\_srclog.tbls'

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties

OK

2 1440639469 1 0 hue 0 2 test MANAGED\_TABLE null null NULL

3 1440664128 1 0 hue 0 3 test\_yonyou MANAGED\_TABLE null null NULL

4 1440664159 1 0 hue 0 4 test\_you MANAGED\_TABLE null null NULL

Time taken: 3.02 seconds, Fetched: 3 row(s)

直接查看文件内容：

# hadoop fs -cat /user/hive/warehouse/dw\_srclog.db/tbls/part-m-0000\*

2 1440639469 1 0 hue 0 2 test MANAGED\_TABLE null null null

3 1440664128 1 0 hue 0 3 test\_yonyou MANAGED\_TABLE null null null

4 1440664159 1 0 hue 0 4 test\_you MANAGED\_TABLE null null null

# 五、sqoop2

## 5.1 介绍

在架构上，sqoop2引入了sqoop server（具体服务器为tomcat），对connector实现了集中的管理。其访问方式也变得多样化了，其可以通过REST API、JAVA API、WEB UI以及CLI控制台方式进行访问。另外，其在安全性能方面也有一定的改善，在sqoop1中我们经常用脚本的方式将HDFS中的数据导入到mysql中，或者反过来将mysql数据导入到HDFS中，其中在脚本里边都要显示指定mysql数据库的用户名和密码的，安全性做的不是太完善。在sqoop2中，如果是通过CLI方式访问的话，会有一个交互过程界面，你输入的密码信息不被看到。

# 六、安装 Sqoop2

Sqoop 2 被分布为两个单独的软件包；一个客户端软件包 (sqoop2-client) 和一个服务器软件包 (sqoop2-server)。在集群中的一个节点上安装服务器包；因为 Sqoop2 Server 充当 MapReduce 客户端，所以此节点必须已安装和配置 Hadoop。

在将充当客户端的每个节点上安装客户端包。Sqoop2 客户端将始终连接至 Sqoop2 Server，以执行任何操作，因此，Hadoop 无需安装在客户端节点上。

这里客户端和服务器安装在同一节点上

# yum install sqoop2-server sqoop2-client

note：

Sqoop 2 包不能安装在与 Sqoop1 包相同的机器上。但是，您可以通过在不同节点上安装 Sqoop1 和 Sqoop 2，在同一 Hadoop 集群中使用两个版本。

# 七、sqoop2 使用

## 7.1 命令说明

服务端的启动

sqoop2-server start

使用以下命令，以交互式模式启动客户端：

# sqoop2

sqoop:000> help

For information about Sqoop, visit: http://sqoop.apache.org/

Available commands:

exit (\x ) Exit the shell

history (\H ) Display, manage and recall edit-line history

help (\h ) Display this help message

set (\st ) Configure various client options and settings

show (\sh ) Display various objects and configuration options

create (\cr ) Create new object in Sqoop repository

delete (\d ) Delete existing object in Sqoop repository

update (\up ) Update objects in Sqoop repository

clone (\cl ) Create new object based on existing one

start (\sta) Start job

stop (\stp) Stop job

status (\stu) Display status of a job

enable (\en ) Enable object in Sqoop repository

disable (\di ) Disable object in Sqoop repository

For help on a specific command type: help command

## 7.2 MySQL导入数据到Hdfs

我们在使用的过程中可能会遇到错误，使用以下命令来使错误信息显示出来

sqoop:000> set option --name verbose --value true

连接Sqoop服务端:

sqoop:000> set server --host udh-yf-15.yonyou.com

Server is set successfully

6.2.1 创建Link 对象（连接mysql或hdfs等）

检查Sqoop服务(server)已经注册的 connectors:

sqoop:000> show connector

+----+------------------------+-----------------+------------------------------------------------------+----------------------+

| Id | Name | Version | Class | Supported Directions |

+----+------------------------+-----------------+------------------------------------------------------+----------------------+

| 1 | kite-connector | 1.99.5-cdh5.4.3 | org.apache.sqoop.connector.kite.KiteConnector | FROM/TO |

| 2 | kafka-connector | 1.99.5-cdh5.4.3 | org.apache.sqoop.connector.kafka.KafkaConnector | TO |

| 3 | hdfs-connector | 1.99.5-cdh5.4.3 | org.apache.sqoop.connector.hdfs.HdfsConnector | FROM/TO |

| 4 | generic-jdbc-connector | 1.99.5-cdh5.4.3 | org.apache.sqoop.connector.jdbc.GenericJdbcConnector | FROM/TO |

+----+------------------------+-----------------+------------------------------------------------------+----------------------+

这里我们利用Generic JDBC Connector 的id来为connector创建新的link对象，下面创建的是连接JDBC的link：

sqoop:000> create link -c 4 #note：这边的4是connector的id，表明创建的是一个generic jdbc connector

Creating link for connector with id 4

Please fill following values to create new link object

Name: mysql-link #note：Name是唯一的

Link configuration

JDBC Driver Class: com.mysql.jdbc.Driver

JDBC Connection String: jdbc:mysql://20.12.20.14:3306/hive #note：jdbc:mysql://主机名(ip):端口/数据库名

Username: root

Password: \*\*\*\*\*\*

JDBC Connection Properties:

There are currently 0 values in the map:

entry# protocol=tcp

There are currently 1 values in the map:

protocol = tcp

New link was successfully created with validation status OK and persistent id 1

sqoop:000> show link

+----+------------+--------------+------------------------+---------+

| Id | Name | Connector Id | Connector Name | Enabled |

+----+------------+--------------+------------------------+---------+

| 1 | mysql-link | 4 | generic-jdbc-connector | true |

+----+------------+--------------+------------------------+---------+

上面，我们成功创建了一个id为1的jdbc link对象。

接下来我们为hdfs connector创建一个link：

sqoop:000> create link -c 3

Creating link for connector with id 3

Please fill following values to create new link object

Name: hdfs-link

Link configuration

HDFS URI: hdfs://udh-yf-13.yonyou.com:8020

New link was successfully created with validation status OK and persistent id 2

已经成功创建了id为2的hdfs link 对象。

### 7.2.1创建Job 对象

Connectors 的From 用于读取数据，To用于写入数据。使用上面的show connector -all命令可以显示出Generic JDBC Connector对From和To都是支持的。也就是说我们既可以从数据库中读取数据，也可以往数据库中写入数据。为了创建一个Job，我们需要指定Job的From和To部分，From和To部分可以使用link Id 来表示。

最后，我们可以使用这两个link Id来关联job的From和To部分。说的通俗一点，就是我们需要从哪里(From)读取数据,把这些数据导入(To)到哪里。

#从MySQL读数据导入到hdfs

sqoop:000> create job -f 1 -t 2

Creating job for links with from id 1 and to id 2

Please fill following values to create new job object

Name: mysql-hdfs #Name必须唯一

From database configuration

Schema name: hive #必填，数据库名称

Table name: TBLS #必填，表名

Table SQL statement: #可选

Table column names: #可选

Partition column name: #可选

Null value allowed for the partition column: #可选

Boundary query: #可选

ToJob configuration

Override null value: #可选

Null value:

Output format:

0 : TEXT\_FILE

1 : SEQUENCE\_FILE

Choose: 0 #必选

Compression format:

0 : NONE

1 : DEFAULT

2 : DEFLATE

3 : GZIP

4 : BZIP2

5 : LZO

6 : LZ4

7 : SNAPPY

8 : CUSTOM

Choose: 0 #必选

Custom compression format:

Output directory: /tmp/tbls #必填

Throttling resources

Extractors: 2 #可选，对应mapreduce的job中的map的数量

Loaders: 1 #可选，对应mapreduce的job中的reduce的数量

New job was successfully created with validation status OK and persistent id 1

这样就建立了一个新的job，他的id是1

### 7.2.2启动job

可以使用以下命令来执行job:

#这边-j后面的1代表上面创建的job的id，可以使用show job来查看已经创建的job

sqoop:000> start job -j 1 -s

### 7.2.3查看数据是否导入

# hadoop fs -cat /tmp/tbls/33ccf4ab-8531-45cf-bf45-b046afdb9ff6.txt

3,1440743238,1,0,'hue',0,3,'test','MANAGED\_TABLE',NULL,NULL,NULL

4,1440743248,1,0,'hue',0,4,'test01','MANAGED\_TABLE',NULL,NULL,NULL

5,1440743252,1,0,'hue',0,5,'test02','MANAGED\_TABLE',NULL,NULL,NULL

## 7.3 Hdfs导出数据到MySQL

Hdfs数据

1,aaaaaaa,bbbbbb

2,ccccccc,dddddd

3,eeeeeee,ffffff

4,df,df

同样我们可以根据上面已创建的两个Link 对象将hdfs上数据导出到MySQL的表中，只需重新创建一个job即可

sqoop:000> create job -f 2 -t 1

Creating job for links with from id 2 and to id 1

Please fill following values to create new job object

Name: hdfs-mysql

From Job configuration

Input directory: /tmp/test.txt

Override null value:

Null value:

To database configuration

Schema name: hive

Table name: test1

Table SQL statement:

Table column names:

Stage table name:

Should clear stage table:

Throttling resources

Extractors: 2

Loaders: 1

New job was successfully created with validation status OK and persistent id 2

这样就建立了一个新的job，他的id是2

启动该Job

sqoop:000> start job -j 2 -s

查看结果

mysql> select \* from test1;

+------+---------+--------+

| id | va01 | va02 |

+------+---------+--------+

| 1 | aaaaaaa | bbbbbb |

| 2 | ccccccc | dddddd |

| 3 | eeeeeee | ffffff |

| 4 | df | df |

+------+---------+--------+

4 rows in set (0.00 sec)