Apache Sqoop是用来实现结构型数据（如关系数据库）和Hadoop之间进行数据迁移的工具。它充分利用了MapReduce的并行特点以批处理的方式加快数据的传输，同时也借助MapReduce实现了容错。

项目地址： ***[http://sqoop.apache.org/](http://sqoop.apache.org/" \t "http://www.tuicool.com/articles/_blank)***

目前为止，已经演化出了2个版本：sqoop1和sqoop2。

sqoop1的最新版本是1.4.5，sqoop2的最新版本是1.99.3；1.99.3和1.4.5是不兼容的，并且功能尚未开发完成，还不适合在生产环境部署。

sqoop支持的数据库：

|  |  |  |  |
| --- | --- | --- | --- |
| **Database** | **version** | **--direct support?** | **connect string matches** |
| HSQLDB | 1.8.0+ | No | ***[jdbc:hsqldb:\*//](http://jdbchsqldb*/" \t "http://www.tuicool.com/articles/_blank)*** |
| MySQL | 5.0+ | Yes | ***jdbc:mysql://*** |
| Oracle | 10.2.0+ | No | ***[jdbc:oracle:\*//](http://jdbcoracle*/" \t "http://www.tuicool.com/articles/_blank)*** |
| PostgreSQL | 8.3+ | Yes (import only) | ***jdbc:postgresql:/*** |

guojian@localtest:~/work$ sudo apt-get install sqoop

guojian@localtest:~/work$ sqoop help

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

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.

import是将关系数据库迁移到HDFS上

guojian@localtest:~/work$ sqoop import --connectjdbc:mysql://192.168.81.176/hivemeta2db --username root -password passwd --table sds

guojian@localtest:~/work$ hadoop fs -ls /user/guojian/sds

Found 5 items

-rw-r--r-- 3 guojian cug\_test 0 2014-09-11 16:04 /user/guojian/sds/\_SUCCESS

-rw-r--r-- 3 guojian cug\_test 483 2014-09-11 16:03 /user/guojian/sds/part-m-00000.snappy

-rw-r--r-- 3 guojian cug\_test 504 2014-09-11 16:04 /user/guojian/sds/part-m-00001.snappy

-rw-r--r-- 3 guojian cug\_test 1001 2014-09-11 16:03 /user/guojian/sds/part-m-00002.snappy

-rw-r--r-- 3 guojian cug\_test 952 2014-09-11 16:03 /user/guojian/sds/part-m-00003.snappy

可以通过--m设置并行数据，即map的数据，决定文件的个数。

默认目录是/user/${user.name}/${tablename}，可以通过--target-dir设置hdfs上的目标目录。

如果想要将整个数据库中的表全部导入到hdfs上，可以使用import-all-tables命令。

sqoop import-all-tables –connect jdbc:mysql://192.168.81.176/hivemeta2db --username root -password passwd

如果想要指定所需的列，使用如下：

sqoop import --connect jdbc:mysql://192.168.81.176/hivemeta2db --username root -password passwd --table sds --columns "SD\_ID,CD\_ID,LOCATION"

指定导出文件为SequenceFiles，并且将生成的类文件命名为com.ctrip.sds：

sqoop **import** --connect ***jdbc:mysql://192.168.81.176/hivemeta2db*** --username root -password passwd --table sds --**class**-name com.ctrip.sds --**as**-sequencefile

导入文本时可以指定分隔符：

sqoop import --connect jdbc:mysql://192.168.81.176/hivemeta2db --username root -password passwd --table sds --fields-terminated-by '\t' --lines-terminated-by '\n' --optionally-enclosed-by '\"'

可以指定过滤条件：

sqoop import --connect jdbc:mysql://192.168.81.176/hivemeta2db --username root -password passwd --table sds --where "sd\_id > 100"

export是import的反向过程，将hdfs上的数据导入到关系数据库中

sqoop export --connect jdbc:mysql://192.168.81.176/sqoop --username root -password passwd --table sds --export-dir /user/guojian/sds

上例中sqoop数据中的sds表需要先把表结构创建出来，否则export操作会直接失败。

由于sqoop是通过map完成数据的导入，各个map过程是独立的，没有事物的概念，可能会有部分map数据导入失败的情况。为了解决这一问题，sqoop中有一个折中的办法，即是指定中间 staging 表，成功后再由中间表导入到结果表。

这一功能是通过 --staging-table <staging-table-name> 指定，同时staging表结构也是需要提前创建出来的:

sqoop export --connect jdbc:mysql://192.168.81.176/sqoop --username root -password passwd --table sds --export-dir /user/guojian/sds --staging-table sds\_tmp

需要说明的是，在使用 --direct ， --update-key 或者--call存储过程的选项时，staging中间表是不可用的。

验证结果：

（1）数据会首先写到sds\_tmp表，导入操作成功后，再由sds\_tmp表导入到sds结果表中，同时会清除sds\_tmp表。

（2）如果有map失败，则成功的map会将数据写入tmp表，export任务失败，同时tmp表的数据会被保留。

（3）如果tmp中已有数据，则此export操作会直接失败，可以使用 --clear-staging-table 指定在执行前清除中间表。

export选项：

|  |  |
| --- | --- |
| --direct | 直接使用 mysqlimport 工具导入mysql |
| --export-dir <dir> | 需要export的hdfs数据路径 |
| -m,--num-mappers <n> | 并行export的map个数n |
| --table <table-name> | 导出到的目标表 |
| --call <stored-proc-name> | 调用存储过程 |
| --update-key <col-name> | 指定需要更新的列名，可以将数据库中已经存在的数据进行更新 |
| --update-mode <mode> | 更新模式，包括 updateonly (默认）和allowinsert |
| 前者只允许更新，后者允许新的列数据写入 |  |
| --input-null-string <null-string> | The string to be interpreted as null for string columns |
| --input-null-non-string <null-string> | The string to be interpreted as null for non-string columns |
| --staging-table <staging-table-name> | 指定中间staging表 |
| --clear-staging-table | 执行export前将中间staging表数据清除 |
| --batch | Use batch mode for underlying statement execution. |
| **Argument** | **Description** |

create-hive-table将关系数据库表导入到hive表中

|  |  |
| --- | --- |
| 参数 | 说明 |
| –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 |

sqoop **create**-hive-**table** --**connect** jdbc:mysql://192.168.81.176/sqoop --username root -password passwd --**table** sds --hive-**table** sds\_bak

默认sds\_bak是在default数据库的。

这一步需要依赖HCatalog，需要先安装HCatalog，否则报如下错误：

Hive history file=/tmp/guojian/hive\_job\_log\_cfbe2de9-a358-4130-945c-b97c0add649d\_1628102887.txt

FAILED: ParseException line 1:44 mismatched input ')' expecting Identifier near '(' in column specification

list-databases列出一台server上可用的数据库

sqoop **list**-databases --connect jdbc:mysql:*//192.168.81.176/ --username root -password passwd*

list-tables列出一个数据库中的表

sqoop **list**-tables --connect jdbc:mysql:*//192.168.81.176/sqoop --username root -password passwd*

codegen：将关系数据库表映射为一个java文件、java class类、以及相关的jar包

sqoop codegen --connect jdbc:mysql:*//192.168.81.176/sqoop --username root -password passwd --table sds*

Note: /tmp/sqoop-guojian/compile/d58f607b046a061593ba708ec5f3d608/sds.java uses or overrides a deprecated API.

Note: Recompile with -Xlint:deprecation **for** details.

guojian@localtest:~/work$ ll /tmp/sqoop-guojian/compile/d58f607b046a061593ba708ec5f3d608/

total 48

drwxrwxr-x 2 guojian guojian 4096 9月 12 14:15 ./

drwxrwxr-x 11 guojian guojian 4096 9月 12 14:15 ../

-rw-rw-r-- 1 guojian guojian 11978 9月 12 14:15 sds.**class**

-rw-rw-r-- 1 guojian guojian 4811 9月 12 14:15 sds.jar

-rw-rw-r-- 1 guojian guojian 17525 9月 12 14:15 sds.java

merge是将两个数据集合并的工具，对于相同的key会覆盖老的值。

|  |  |
| --- | --- |
| --class-name <class> | 指定merge job使用的类名称 |
| --jar-file <file> | 合并时引入的jar包，该jar包是通过Codegen工具生成的jar包 |
| --merge-key <col> | 指定作为merge key的列名 |
| --new-data <path> | 指定newer数据目录 |
| --onto <path> | 指定older数据目录 |
| --target-dir <path> | 指定目标输出目录 |
| **参数** | **说明** |

sqoop merge --**new**-data /user/guojian/sds --onto /user/guojian/sqoop --target-dir /user/guojian/sds\_new --jar-file sds.jar --**class**-name sds --merge-key SD\_ID

值得注意的是，--target-dir如果设置为已经存在的目录，sqoop会报错退出。

eval用户可以很快的使用sql语句对数据库进行操作。这使得用户在执行import操作之前检查sql语句是否正确。

sqoop eval --connect jdbc:mysql://192.168.81.176/sqoop --username root -password passwd --query "SELECT SD\_ID,CD\_ID,LOCATION FROM sds LIMIT 10"

job用来生成sqoop任务。

|  |  |
| --- | --- |
| --create <job-id> | 创业一个新的sqoop作业. |
| --delete <job-id> | 删除一个sqoop job |
| --exec <job-id> | 执行一个 --create 保存的作业 |
| --show <job-id> | 显示一个作业的参数 |
| --list | 显示所有创建的sqoop作业 |
| **参数** | **说明** |

例子：

sqoop job --create myimportjob -- import --connectjdbc:mysql://192.168.81.176/hivemeta2db --username root -password passwd --table TBLS

guojian@localtest:~/work$ sqoop job --listAvailable jobs:

myimportjob

guojian@localtest:~/work$ sqoop job *--show myimportjob*

Job: myimportjob

Tool: import

Options:*----------------------------*

verbose = false

db.connect.string = ***jdbc:mysql://192.168.81.176/hivemeta2db***

codegen.output.delimiters.escape = 0

codegen.output.delimiters.enclose.required = false

codegen.input.delimiters.field = 0

hbase.**create**.**table** = **false**

db.require.password = **true**

hdfs.append.dir = **false**

db.**table** = TBLS

import.**fetch**.**size** = **null**

accumulo.**create**.**table** = **false**

codegen.**input**.delimiters.**escape** = 0

codegen.**input**.delimiters.enclose.required = **false**

db.username = root

codegen.**output**.delimiters.record = 10

import.max.inline.lob.**size** = 16777216

hbase.bulk.**load**.enabled = **false**

hcatalog.**create**.**table** = **false**

db.clear.staging.**table** = **false**

codegen.**input**.delimiters.record = 0

enable.compression = **false**

hive.overwrite.**table** = **false**

hive.import = **false**

codegen.**input**.delimiters.enclose = 0

accumulo.batch.**size** = 10240000

hive.**drop**.delims = **false**

codegen.**output**.delimiters.enclose = 0

hdfs.**delete**-target.dir = **false**

codegen.**output**.dir = .

codegen.auto.compile.dir = **true**

mapreduce.num.mappers = 4

accumulo.max.latency = 5000

import.direct.split.**size** = 0

codegen.**output**.delimiters.field = 44

export.new.**update** = UpdateOnly

incremental.mode = None

hdfs.file.format = TextFile

codegen.compile.dir = /tmp/sqoop-guojian/compile/bd9c7f7b651276067b3f7b341b7fa4cb

direct.import = **false**

hive.fail.**table**.**exists** = **false**

db.batch = **false**

执行：

sqoop job -exec myimportjob

metastore 配置sqoop job的共享元数据信息，这样多个用户定义和执行sqoop job在这一 metastore中。默认存储在~/.sqoop

启动：sqoop metastore

关闭：sqoop metastore --shutdown

metastore文件的存储位置是在 conf/sqoop-site.xml中sqoop.metastore.server.location 配置，指向本地文件。

metastore可以通过TCP/IP访问，端口号可以通过sqoop.metastore.server.port 配置，默认是16000。

客户端可以通过 指定 sqoop.metastore.client.autoconnect.url 或使用 --meta-connect ，配置为 ***jdbc:hsqldb:hsql://***<server-name>:<port>/sqoop，例如 ***jdbc:hsqldb:hsql://metaserver.example.com:16000/sqoop*** 。

更多说明见 ***[http://sqoop.apache.org/docs/1.4.5/SqoopUserGuide.html](http://sqoop.apache.org/docs/1.4.5/SqoopUserGuide.html" \t "http://www.tuicool.com/articles/_blank)***