## 大数据技术之 **Sqoop**

一、**Sqoop** 简介

Apache Sqoop(TM)是一种旨在有效地在 Apache Hadoop 和诸如关系数据库等结构化数据存储之间传输大量数据的工具。

Sqoop 于 2012 年 3 月孵化出来，现在是一个顶级的 Apache 项目。

最新的稳定版本是 1.4.7。Sqoop2 的最新版本是 1.99.7。请注意，1.99.7 与 1.4.7不兼容，且没有特征不完整，它并不打算用于生产部署。

二、**Sqoop** 原理

将导入或导出命令翻译成 mapreduce 程序来实现。

在翻译出的 mapreduce 中主要是对 inputformat 和 outputformat 进行定制。

三、**Sqoop** 安装

安装 Sqoop 的前提是已经具备 Java 和 Hadoop 的环境。

# 、下载并解压

1. 最新版下载地址：[http://mirrors.hust.edu.cn/apache/sqoop/1.4.7/](http://mirrors.hust.edu.cn/apache/sqoop/1.4.6/)
2. 上传安装包 sqoop-1.4.7.bin\_\_hadoop-2.6.0.tar.gz 到虚拟机中，如我的上传目录是：

/home/admin/softwares/installation

1. 解压 sqoop 安装包到指定目录，如：

$ tar -zxf sqoop-1.4.7.bin\_\_hadoop-2.6.0.tar.gz -C ~/opt/

# 、修改配置文件

Sqoop 的配置文件与大多数大数据框架类似，在 sqoop 根目录下的 conf 目录中。

1. 重命名配置文件

$ mv sqoop-env-template.sh sqoop-env.sh

$ mv sqoop-site-template.xml sqoop-site.xml

1. 修改配置文件

sqoop-env.sh

* 1. 、拷贝 **JDBC** 驱动

export HADOOP\_COMMON\_HOME=/opt/hadoop-2.7.2 export HADOOP\_MAPRED\_HOME=/opt/hadoop-2.7.2 export HIVE\_HOME=/opt/apache-hive-1.2.2-bin

export ZOOKEEPER\_HOME=/opt/zookeeper-3.4.5

export ZOOCFGDIR=/opt/zookeeper-3.4.5

拷贝 jdbc 驱动到 sqoop 的 lib 目录下，如：

$ cp -a mysql-connector-java-5.1.27-bin.jar ~/opt/sqoop-1.4.7.bin hadoop-2.6.0/lib/

* 1. 、验证 **Sqoop**

我们可以通过某一个 command 来验证 sqoop 配置是否正确：

Run a standalone Sqoop metastore

metastore

Merge results of incremental imports

merge

List available tables in a database

list-tables

List available databases on a server

list-databases

Work with saved jobs

job

Import datasets from a mainframe server to HDFS

import-mainframe

Import tables from a database to HDFS

import-all-tables

Import a table from a database to HDFS

import

List available commands

help

Export an HDFS directory to a database table

export

Evaluate a SQL statement and display the results

eval

Import a table definition into Hive

create-hive-table

Generate code to interact with database records

codegen

$ bin/sqoop help

出现一些 Warning 警告（警告信息已省略），并伴随着帮助命令的输出：

Available commands:

* 1. 、测试 **Sqoop** 是否能够成功连接数据库

$ bin/sqoop list-databases --connect jdbc:mysql://linux01:3306/ --username root --password 123456

出现如下输出：

information\_schema metastore

mysql

performance\_schema

version Display version information

四、**Sqoop** 的简单使用案例

# 、导入数据

在 Sqoop 中，“导入”概念指：从非大数据集群（RDBMS）向大数据集群（HDFS，HIVE，

HBASE）中传输数据，叫做：导入，即使用 import 关键字。

* + 1. 、**RDBMS** 到 **HDFS**

1. 确定 Mysql 服务开启正常
2. 在 Mysql 中新建一张表并插入一些数据

$ mysql -uroot -p123456

mysql> create database company;

mysql> create table company.staff(id int(4) primary key not null auto\_increment, name varchar(255), sex varchar(255));

mysql> insert into company.staff(name, sex) values('Thomas', 'Male');

mysql> insert into company.staff(name, sex) values('Catalina', 'FeMale');

1. 导入数据

（**1**）全部导入

$ bin/sqoop import \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff \

--target-dir /user/company \

--delete-target-dir \

--num-mappers 1 \

--fields-terminated-by "\t"

（**2**）查询导入

$ bin/sqoop import \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--target-dir /user/company \

--delete-target-dir \

--num-mappers 1 \

--fields-terminated-by "\t" \

--query 'select name,sex from staff where id <=1 and $CONDITIONS;'

提示：must contain '$CONDITIONS' in WHERE clause.

提示：如果 query 后使用的是双引号，则$CONDITIONS 前必须加转移符，防止 shell

识别为自己的变量。

提示：--query 选项，不能同时与--table 选项使用

（**3**）导入指定列

$ bin/sqoop import \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--target-dir /user/company \

--delete-target-dir \

--num-mappers 1 \

--fields-terminated-by "\t" \

--columns id,sex \

--table staff

提示：columns 中如果涉及到多列，用逗号分隔，分隔时不要添加空格

（**4**）使用 **sqoop** 关键字筛选查询导入数据

$ bin/sqoop import \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--target-dir /user/company \

--delete-target-dir \

--num-mappers 1 \

--fields-terminated-by "\t" \

--table staff \

--where "id=1"

提示：在 Sqoop 中可以使用 sqoop import -D property.name=property.value 这样的方式加

入执行任务的参数，多个参数用空格隔开。

* + 1. 、**RDBMS** 到 **Hive**

$ bin/sqoop import \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff \

--num-mappers 1 \

--hive-import \

--fields-terminated-by "\t" \

--hive-overwrite \

--hive-table staff\_hive

提示：该过程分为两步，第一步将数据导入到 HDFS，第二步将导入到 HDFS 的数据迁

移到 Hive 仓库

提示：第一步默认的临时目录是/user/admin/表名

**4.2**、导出数据

在 Sqoop 中，“导出”概念指：从大数据集群（HDFS，HIVE，HBASE）向非大数据集群

（RDBMS）中传输数据，叫做：导出，即使用 export 关键字。

**4.2.1**、**HIVE/HDFS** 到 **RDBMS**

$ bin/sqoop export \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff \

--num-mappers 1 \

--export-dir /user/hive/warehouse/staff\_hive \

--input-fields-terminated-by "\t"

提示：Mysql 中如果表不存在，不会自动创建

思考：数据是覆盖还是追加

**4.3**、脚本打包

使用 opt 格式的文件打包 sqoop 命令，然后执行

1. 创建一个**.opt** 文件

$ mkdir opt

1. 编写 **sqoop** 脚本

$ touch opt/job\_HDFS2RDBMS.opt

$ vi opt/job\_HDFS2RDBMS.opt

export

--connect jdbc:mysql://linux01:3306/company

--username root

--password 123456

--table staff

--num-mappers 1

--export-dir

/user/hive/warehouse/staff\_hive

--input-fields-terminated-by "\t"

1. 执行该脚本

$ bin/sqoop --options-file opt/job\_HDFS2RDBMS.opt

五、**Sqoop** 一些常用命令及参数

# 、常用命令列举

列出一部分 Sqoop 操作时的常用参数，以供参考，需要深入学习的可以参看对应类的源代码。

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 命令 | 类 | 说明 |
| 1 | import | ImportTool | 将数据导入到集群 |
| 2 | export | ExportTool | 将集群数据导出 |
| 3 | codegen | CodeGenTool | 获取数据库中某张表  数据生成Java 并打包  Jar |
| 4 | create-hive-table | CreateHiveTableTool | 创建 Hive 表 |
| 5 | eval | EvalSqlTool | 查看 SQL 执行结果 |
| 6 | import-all-tables | ImportAllTablesTool | 导入某个数据库下所  有表到 HDFS 中 |
| 7 | job | JobTool | 用来生成一个 sqoop  的任务，生成后，该任务并不执行，除非使用命令执行该任务。 |
| 8 | list-databases | ListDatabasesTool | 列出所有数据库名 |
| 9 | list-tables | ListTablesTool | 列出某个数据库下所  有表 |
| 10 | merge | MergeTool | 将 HDFS 中不同目录  下面的数据合在一起，并存放在指定的目录中 |
| 11 | metastore | MetastoreTool | 记 录 sqoop job 的 元数据信息，如果不启动 metastore 实例，则默认的元数据存储目录为：~/.sqoop，如果 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | 要更改存储目录，可以 在 配 置 文 件sqoop-site.xml 中进行更改。 |
| 12 | help | HelpTool | 打印 sqoop 帮助信息 |
| 13 | version | VersionTool | 打印 sqoop 版本信息 |

# 、命令**&**参数详解

刚才列举了一些 Sqoop 的常用命令，对于不同的命令，有不同的参数，让我们来一一列举说明。

首先来我们来介绍一下公用的参数，所谓公用参数，就是大多数命令都支持的参数。

# 、公用参数：数据库连接

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --connect | 连接关系型数据库的 URL |
| 2 | --connection-manager | 指定要使用的连接管理类 |
| 3 | --driver | JDBC 的 driver class |
| 4 | --help | 打印帮助信息 |
| 5 | --password | 连接数据库的密码 |
| 6 | --username | 连接数据库的用户名 |
| 7 | --verbose | 在控制台打印出详细信息 |

* + 1. 、公用参数：**import**

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --enclosed-by <char> | 给字段值前后加上指定的字  符 |
| 2 | --escaped-by <char> | 对字段中的双引号加转义符 |
| 3 | --fields-terminated-by <char> | 设定每个字段是以什么符号  作为结束，默认为逗号 |

|  |  |  |
| --- | --- | --- |
| 4 | --lines-terminated-by <char> | 设定每行记录之间的分隔符，  默认是\n |
| 5 | --mysql-delimiters | Mysql 默认的分隔符设置，字段之间以逗号分隔，行之间以  \n 分隔，默认转义符是\，字  段值以单引号包裹。 |
| 6 | --optionally-enclosed-by  <char> | 给带有双引号或单引号的字  段值前后加上指定字符。 |

* + 1. 、公用参数：**export**

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --input-enclosed-by <char> | 对字段值前后加上指定字符 |
| 2 | --input-escaped-by <char> | 对含有转移符的字段做转义  处理 |
| 3 | --input-fields-terminated-by  <char> | 字段之间的分隔符 |
| 4 | --input-lines-terminated-by  <char> | 行之间的分隔符 |
| 5 | --input-optionally-enclosed-by  <char> | 给带有双引号或单引号的字  段前后加上指定字符 |

* + 1. 、公用参数：**hive**

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --hive-delims-replacement  <arg> | 用自定义的字符串替换掉数  据中的\r\n 和\013 \010 等字符 |
| 2 | --hive-drop-import-delims | 在导入数据到 hive 时，去掉  数据中的\r\n\013\010 这样的字符 |
| 3 | --map-column-hive <map> | 生成 hive 表时，可以更改生 |

|  |  |  |
| --- | --- | --- |
|  |  | 成字段的数据类型 |
| 4 | --hive-partition-key | 创建分区，后面直接跟分区名，分区字段的默认类型为  string |
| 5 | --hive-partition-value <v> | 导入数据时，指定某个分区的  值 |
| 6 | --hive-home <dir> | hive 的安装目录，可以通过该参数覆盖之前默认配置的目  录 |
| 7 | --hive-import | 将数据从关系数据库中导入  到 hive 表中 |
| 8 | --hive-overwrite | 覆盖掉在 hive 表中已经存在  的数据 |
| 9 | --create-hive-table | 默认是 false，即，如果目标表已经存在了，那么创建任务  失败。 |
| 10 | --hive-table | 后面接要创建的 hive 表,默认  使用 MySQL 的表名 |
| 11 | --table | 指定关系数据库的表名 |

公用参数介绍完之后，我们来按照命令介绍命令对应的特有参数。

* + 1. 、命令**&**参数：**import**

将关系型数据库中的数据导入到 HDFS（包括 Hive，HBase）中，如果导入的是 Hive，那么当 Hive 中没有对应表时，则自动创建。

1. 命令：

如：导入数据到 hive 中

$ bin/sqoop import \

如：增量导入数据到 hive 中，mode=append

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff \

--hive-import

append 导入：

$ bin/sqoop import \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff \

--num-mappers 1 \

--fields-terminated-by "\t" \

--target-dir /user/hive/warehouse/staff\_hive \

--check-column id \

--incremental append \

--last-value 3

提示：append 不能与--hive-等参数同时使用（Append mode for hive imports is not yet

supported. Please remove the parameter --append-mode） 如：增量导入数据到 hdfs 中，mode=lastmodified

先在 mysql 中建表并插入几条数据：

mysql> create table company.staff\_timestamp(id int(4), name varchar(255), sex varchar(255), last\_modified timestamp DEFAULT CURRENT\_TIMESTAMP ON UPDATE

CURRENT\_TIMESTAMP);

mysql> insert into company.staff\_timestamp (id, name, sex) values(1, 'AAA', 'female'); mysql> insert into company.staff\_timestamp (id, name, sex) values(2, 'BBB', 'female');

先导入一部分数据：

$ bin/sqoop import \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff\_timestamp \

--delete-target-dir \

--m 1

再增量导入一部分数据：

mysql> insert into company.staff\_timestamp (id, name, sex) values(3, 'CCC', 'female');

$ bin/sqoop import \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff\_timestamp \

--check-column last\_modified \

--incremental lastmodified \

--last-value "2017-09-28 22:20:38" \

--m 1 \

--append

提示：使用 lastmodified 方式导入数据要指定增量数据是要--append（追加）还是要

--merge-key（合并）

提示：last-value 指定的值是会包含于增量导入的数据中

1. 参数：

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |

|  |  |  |
| --- | --- | --- |
| 1 | --append | 将数据追加到 HDFS 中已经存在的 DataSet 中，如果使用该参数，sqoop 会把数据先导  入到临时文件目录，再合并。 |
| 2 | --as-avrodatafile | 将数据导入到一个 Avro 数据  文件中 |
| 3 | --as-sequencefile | 将数据导入到一个 sequence  文件中 |
| 4 | --as-textfile | 将数据导入到一个普通文本  文件中 |
| 5 | --boundary-query <statement> | 边界查询，导入的数据为该参数的值（一条 sql 语句）所执  行的结果区间内的数据。 |
| 6 | --columns <col1, col2, col3> | 指定要导入的字段 |
| 7 | --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， 以便加快导入导出过程。 |
| 8 | --direct-split-size | 在使用上面 direct 直接导入的基础上，对导入的流按字节分块，即达到该阈值就产生一  个新的文件 |
| 9 | --inline-lob-limit | 设定大对象数据类型的最大  值 |
| 10 | --m 或–num-mappers | 启动 N 个 map 来并行导入数  据，默认 4 个。 |
| 11 | --query 或--e <statement> | 将查询结果的数据导入，使用  时必须伴随参--target-dir， |

|  |  |  |
| --- | --- | --- |
|  |  | --hive-table，如果查询中有  where 条件，则条件后必须加  上$CONDITIONS 关键字 |
| 12 | --split-by <column-name> | 按照某一列来切分表的工作单元，不能与  --autoreset-to-one-mapper 连  用（请参考官方文档） |
| 13 | --table <table-name> | 关系数据库的表名 |
| 14 | --target-dir <dir> | 指定 HDFS 路径 |
| 15 | --warehouse-dir <dir> | 与 14 参数不能同时使用，导  入数据到 HDFS 时指定的目录 |
| 16 | --where | 从关系数据库导入数据时的  查询条件 |
| 17 | --z 或--compress | 允许压缩 |
| 18 | --compression-codec | 指定 hadoop 压缩编码类，默  认为 gzip(Use Hadoop codec default gzip) |
| 19 | --null-string <null-string> | string 类型的列如果 null，替  换为指定字符串 |
| 20 | --null-non-string <null-string> | 非 string 类型的列如果 null，  替换为指定字符串 |
| 21 | --check-column <col> | 作为增量导入判断的列名 |
| 22 | --incremental <mode> | mode：append 或 lastmodified |
| 23 | --last-value <value> | 指定某一个值，用于标记增量  导入的位置 |

* + 1. 、命令**&**参数：**export**

从 HDFS（包括 Hive 和 HBase）中奖数据导出到关系型数据库中。

1. 命令： 如：

$ bin/sqoop export \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff \

--export-dir /user/company \

--input-fields-terminated-by "\t" \

--num-mappers 1

1. 参数：

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --direct | 利用数据库自带的导入导出  工具，以便于提高效率 |
| 2 | --export-dir <dir> | 存放数据的 HDFS 的源目录 |
| 3 | -m 或--num-mappers <n> | 启动 N 个 map 来并行导入数  据，默认 4 个 |
| 4 | --table <table-name> | 指定导出到哪个 RDBMS 中  的表 |
| 5 | --update-key <col-name> | 对某一列的字段进行更新操  作 |
| 6 | --update-mode <mode> | updateonly  allowinsert(默认) |
| 7 | --input-null-string | 请参考 import 该类似参数说 |

|  |  |  |
| --- | --- | --- |
|  | <null-string> | 明 |
| 8 | --input-null-non-string  <null-string> | 请参考 import 该类似参数说  明 |
| 9 | --staging-table  <staging-table-name> | 创建一张临时表，用于存放所有事务的结果，然后将所有事务结果一次性导入到目标表  中，防止错误。 |
| 10 | --clear-staging-table | 如果第 9 个参数非空，则可以  在导出操作执行前，清空临时事务结果表 |

* + 1. 、命令**&**参数：**codegen**

将关系型数据库中的表映射为一个 Java 类，在该类中有各列对应的各个字段。如：

$ bin/sqoop codegen \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff \

--bindir /home/admin/Desktop/staff \

--class-name Staff \

--fields-terminated-by "\t"

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --bindir <dir> | 指定生成的 Java 文件、编译成的 class 文件及将生成文件  打包为 jar 的文件输出路径 |

|  |  |  |
| --- | --- | --- |
| 2 | --class-name <name> | 设定生成的 Java 文件指定的  名称 |
| 3 | --outdir <dir> | 生成 Java 文件存放的路径 |
| 4 | --package-name <name> | 包名，如 com.z，就会生成 com  和 z 两级目录 |
| 5 | --input-null-non-string  <null-str> | 在生成的 Java 文件中，可以将 null 字符串或者不存在的字符串设置为想要设定的值  （例如空字符串） |
| 6 | --input-null-string <null-str> | 将 null 字符串替换成想要替  换的值（一般与 5 同时使用） |
| 7 | --map-column-java <arg> | 数据库字段在生成的 Java 文件中会映射成各种属性，且默认的数据类型与数据库类型保持对应关系。该参数可以改变默认类型，例如：  --map-column-java id=long,  name=String |
| 8 | --null-non-string <null-str> | 在生成 Java 文件时，可以将  不存在或者 null 的字符串设置为其他值 |
| 9 | --null-string <null-str> | 在生成 Java 文件时，将 null  字符串设置为其他值（一般与  8 同时使用） |
| 10 | --table <table-name> | 对应关系数据库中的表名生  成的 Java 文件中的各个属性与该表的各个字段一一对应 |

* + 1. 、命令**&**参数：**create-hive-table**

生成与关系数据库表结构对应的 hive 表结构。命令：

如：

$ bin/sqoop create-hive-table \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff \

--hive-table hive\_staff

参数：

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --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 |
| 2 | --hive-overwrite | 覆盖掉在 Hive 表中已经存在  的数据 |
| 3 | --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 |
| 4 | --hive-table | 后面接要创建的 hive 表 |
| 5 | --table | 指定关系数据库的表名 |

* + 1. 、命令**&**参数：**eval**

可以快速的使用 SQL 语句对关系型数据库进行操作，经常用于在 import 数据之前，了解一下 SQL 语句是否正确，数据是否正常，并可以将结果显示在控制台。

命令： 如：

$ bin/sqoop eval \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--query "SELECT \* FROM staff"

参数：

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --query 或--e | 后跟查询的 SQL 语句 |

* + 1. 、命令**&**参数：**import-all-tables**

可以将 RDBMS 中的所有表导入到 HDFS 中，每一个表都对应一个 HDFS 目录命令：

如：

$ bin/sqoop import-all-tables \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--warehouse-dir /all\_tables

参数：

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --as-avrodatafile | 这些参数的含义均和 import  对应的含义一致 |
| 2 | --as-sequencefile |
| 3 | --as-textfile |
| 4 | --direct |
| 5 | --direct-split-size <n> |
| 6 | --inline-lob-limit <n> |

|  |  |  |
| --- | --- | --- |
| 7 | --m 或—num-mappers <n> |  |
| 8 | --warehouse-dir <dir> |
| 9 | -z 或--compress |
| 10 | --compression-codec |

* + 1. 、命令**&**参数：**job**

用来生成一个 sqoop 任务，生成后不会立即执行，需要手动执行。命令：

如：

$ bin/sqoop job \

--create myjob -- import-all-tables \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456

$ bin/sqoop job \

--list

$ bin/sqoop job \

--exec myjob

提示：注意 import-all-tables 和它左边的--之间有一个空格

提示：如果需要连接 metastore，则--meta-connect jdbc:hsqldb:hsql://linux01:16000/sqoop

参数：

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --create <job-id> | 创建 job 参数 |
| 2 | --delete <job-id> | 删除一个 job |
| 3 | --exec <job-id> | 执行一个 job |
| 4 | --help | 显示 job 帮助 |
| 5 | --list | 显示 job 列表 |

|  |  |  |
| --- | --- | --- |
| 6 | --meta-connect <jdbc-uri> | 用来连接 metastore 服务 |
| 7 | --show <job-id> | 显示一个 job 的信息 |
| 8 | --verbose | 打印命令运行时的详细信息 |

提示：在执行一个 job 时，如果需要手动输入数据库密码，可以做如下优化

<property>

<name>sqoop.metastore.client.record.password</name>

<value>true</value>

<description>If true, allow saved passwords in the metastore.</description>

</property>

* + 1. 、命令**&**参数：**list-databases**

命令： 如：

$ bin/sqoop list-databases \

--connect jdbc:mysql://linux01:3306/ \

--username root \

--password 123456

参数：与公用参数一样

* + 1. 、命令**&**参数：**list-tables**

命令： 如：

$ bin/sqoop list-tables \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456

参数：与公用参数一样

* + 1. 、命令**&**参数：**merge**

将 HDFS 中不同目录下面的数据合并在一起并放入指定目录中数据环境：

new\_staff

1 AAA

male

2

BBB

male

3

CCC

male

4

DDD

male

old\_staff

1 AAA

female

2

CCC

female

3

BBB

female

6

DDD

female

提示：上边数据的列之间的分隔符应该为\t，行与行之间的分割符为\n，如果直接复制，

请检查之。命令：

如：

创建 JavaBean：

$ bin/sqoop codegen \

--connect jdbc:mysql://linux01:3306/company \

--username root \

--password 123456 \

--table staff \

--bindir /home/admin/Desktop/staff \

--class-name Staff \

--fields-terminated-by "\t"

4 DDD MALE

6 DDD FEMALE

MALE

3 CCC

MALE

2 BBB

开始合并：

$ bin/sqoop merge \

--new-data /test/new/ \

--onto /test/old/ \

--target-dir /test/merged \

--jar-file /home/admin/Desktop/staff/Staff.jar \

--class-name Staff \

--merge-key id

结果：

1 AAA MALE

参数：

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --new-data <path> | HDFS 待合并的数据目录，  合并后在新的数据集中保留 |
| 2 | --onto <path> | HDFS 合并后，重复的部分在  新的数据集中被覆盖 |
| 3 | --merge-key <col> | 合并键，一般是主键 ID |
| 4 | --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 |
| 5 | --class-name <class> | 对应的表名或对象名， 该  class 类是包含在 jar 包中的 |

|  |  |  |
| --- | --- | --- |
| 6 | --target-dir <path> | 合并后的数据在 HDFS 里存  放的目录 |

* + 1. 、命令**&**参数：**metastore**

记录了 Sqoop job 的元数据信息，如果不启动该服务，那么默认 job 元数据的存储目录为

~/.sqoop，可在 sqoop-site.xml 中修改。命令：

如：启动 sqoop 的 metastore 服务

$ bin/sqoop metastore

参数：

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 说明 |
| 1 | --shutdown | 关闭 metastore |