# Sqoop

## Sqoop基本概念

### （★）è¿éåå¾çæè¿°Sqoop架构

sqoop架构非常简单，其底层就是map-reduce,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。

参考网址：<https://blog.csdn.net/py_123456/article/details/80761446>

### （★）Sqoop的基本概念和使用方式

Sqoop是一种用于在结构化，半结构化和非结构化数据源之间有效传输数据的工具。关系数据库是结构化数据源的示例，具有为其存储的数据定义良好的模式。Cassandra，Hbase是半结构化数据源的示例，HDFS是Sqoop可以支持的非结构化数据源的示例。

使用方式：<http://www.cnblogs.com/piperck/p/9984236.html>

## Sqoop常见问题

### （★）sqoop在导入数据到mysql中，如何让数据不重复导入？

#### 增量导出（insert模式）

**应用场景：**将Hive表中的增量记录以及有修改的记录同步到目标表中。

**使用限制：**

update-key可以是多个字段，但这些字段的记录都应该是未被更新过的，若该参数指定的字段有更新，则对应记录的更新不会被同步到目标表中。

**参数：**

源表、目标表、筛选字段及其取值范围、导出字段（select的字段）、映射关系（--column后的参数）、更新的参考字段（--update-key后的参数）

**适用的数据库：**Oracle、SQL Server、MySQL

**SQOOP脚本示例：**

Sqoop export \

--connect jdbc:mysql://localhost:3306/wht \

（此句是配置mysql路径,linux下一行未完结要加上反斜杠）

--username root \ （配置mysql的帐号）

--password cloudera \ （配置mysql的密码）

--table <targetTable> \ （配置mysql的table表名）

--fields-terminated-by ‘,’ \ （指定列分隔符是 ‘,’）

--columns F1,F2,F3 \ （配置table抽取的字段）

--update-key F4 \ （通过哪个字段来判断增量）

--update-mode allowinsert \ （插入模式，默认是只更新，这里设置为allowinsert. ）

--export-dir /user/root/export/test （导出文件的HDFS路径）

#### 更新导出（update模式）

**应用场景：**将Hive表中的有更新的记录同步到目标表。

**使用限制：**

update-key可以是多个字段，但这些字段的记录都应该是未被更新过的，若该参数指定的字段有更新，则对应记录的更新不会被同步到目标表中。

**参数：**

源表、目标表、筛选字段及其取值范围、导出字段（select的字段）、映射关系（--column后的参数）、更新的参考字段（--update-key后的参数）

**适用的数据库：**Oracle、DB2、SQL Server、PG、MySQL

**SQOOP脚本示例：**

Sqoop export \

--connect jdbc:mysql://localhost:3306/wht \

（此句是配置mysql路径,linux下一行未完结要加上反斜杠）

--username root \ （配置mysql的帐号）

--password cloudera \ （配置mysql的密码）

--table <target Table> \ （配置mysql的table表名）

--fields-terminated-by ‘,’ \ （指定列分隔符是 ‘,’）

--columns F1,F2,F3 \ （配置table抽取的字段）

--update-key F4 \ （通过哪个字段来判断更新）

--update-mode updateonly \ （插入模式，默认是只更新，可以设置为allowinsert. ）

--export-dir /user/root/export/test （导出文件的HDFS路径）

### （★★）你们数据库怎么导入Hive的,有没有出现问题

使用sqoop导入

#### 遇到的问题1：

我们公司的数据库中设计了有blob（二进制字段）或者text字段，导致导入的时候出现了缓存不够的情况，开始解决起来感觉很棘手，后来查看了sqoop的文档，加上了limit属性。

##### 解决方案：

text：缓存配置量增大

sqoop import

--connect jdbc:[MySQL](http://lib.csdn.net/base/mysql" \t "_blank" \o "MySQL知识库)://192.168.56.204:3306/sqoop \（配置mysql路径）

--username [Hive](http://lib.csdn.net/base/hive" \t "_blank" \o "Hive知识库) \ （配置mysql的帐号）

--password hive \ （配置mysql的密码）

--table jobinfo \ （配置mysql的table表名）

--target-dir /sqoop/test5 \ （指定HDFS路径）

--inline-lob-limit 16777216 \ （设定大对象数据类型的最大值）

--fields-terminated-by '\t' \ （指定列分隔符是 ‘,’）

-m 1 （启动N个map来并行导入数据，默认4个）

#### 遇到的问题2：

在导入hive的时候，如果数据库中有blob或者text字段，会报错。在将数据由MYSQL数据库导入到Hive时，发现带有blob字段的表的数据会错乱，出现一些字段全为NULL的空行。

##### 解决方案：

**1.字段没用实际用途，无需保留**

由于在项目中BLOB字段没有实际的分析用途，因此考虑将BLOB字段去掉；同时，为了防止BLOB字段产生一些问题，因此将HIVE中BLOB字段禁用。

禁用的方式如下：

[Hadoop@mastersqoop-1.4.5]$cd$SQOOP\_HOME/conf

[hadoop@masterconf]$vi oraoop-site.xml

将以下属性的注释去掉，并且将value改为true

<property>

<name>oraoop.import.omit.lobs.and.long</name>

<value>true</value>

<description>Iftrue,OraOopwillomitBLOB,BLOB,NBLOBandLONGcolumnsduringanImport.</description>

</property>

**2.字段需要保留**

有些表中虽然有blob字段，但是不能排除掉，因为其他字段我们需要使用，因此在导入的时候采用指定字段（--columns配置table抽取的字段）的方式来进行导入

**SQOOP脚本示例：**

sqoop import \

--hive-import \ （将数据从关系数据库中导入到 hive 表中）

--hive-database test \ （hive中使用的数据库名，默认为default库）

--hive-table hive\_table\_name \ （要创建的 hive 表,默认使用 MySQL 的表名）

--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）

--connect jdbc:mysql://localhost:3306/wht \ （配置mysql路径）

--username user \ （配置mysql的帐号）

--password user \ （配置mysql的密码）

--bindir //scratch \ （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）

--outdir /Java \ （生成Java文件存放的路径）

--table aaa \ （配置mysql的table表名）

--columns "ID,NAME" \ （配置table抽取的字段）

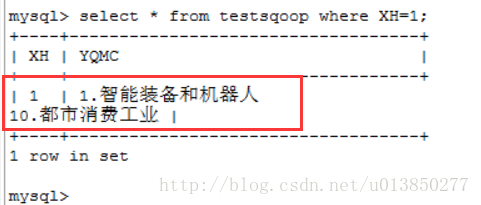
-m 1 \ （启动N个map来并行导入数据，默认4个）

--null-string '\\N' \ （string类型的列如果null，替换为指定字符串）

--null-non-string '\\N' （非string类型的列如果null，替换为指定字符串）

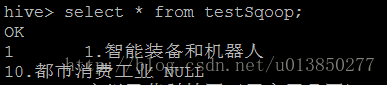
### （★★）有关sqoop 分割符的问题

在将mysql 中的数据导入到hive中，mysql 中的数据如下；



如XH=1在mysql中这是一条数据，但是数据对应的某一列数据有换行符。

在进行sqoop import 数据时，如果不加其他参数，导入的数据默认的列分隔符是’\001’，默认的行分隔符是’\n’。也就像下面的数据那样，在导入时出现换行符时hive以为这条数据已经结束，并将其后面输入的数据当做另一条数据。

因而hive 默认会解析成两条数据，这样就造成导入数据时出现了数据跟原表不一致的问题。如下图所示：

sqoop import

--connect jdbc:mysql://xxx:3306/db / （配置mysql路径）

--username xxx / （配置mysql的帐号）

--password xxx / （配置mysql的密码）

--table testSqoop / （配置mysql的table表名）

--hive-import / （将数据从关系数据库中导入到 hive 表中）

--hive-table testSqoop / （要创建的 hive 表,默认使用 MySQL 的表名）

-m 1 / （启动N个map来并行导入数据，默认4个）

**解决方法：**

加上参数–hive-drop-import-delims来把导入数据中包含的hive默认的分隔符去掉

**命令如下所示：**

sqoop import

--connect jdbc:mysql://xxx:3306/db / （配置mysql路径）

--username xxx / （配置mysql的帐号）

--password xxx / （配置mysql的密码）

--table testSqoop / （配置mysql的table表名）

--hive-import / （将数据从关系数据库中导入到 hive 表中）

--hive-table testSqoop / （要创建的 hive 表,默认使用 MySQL 的表名）

-m 1 / （启动N个map来并行导入数据，默认4个）

--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）

### （★）sqoop命令

**常用命令列举：**

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

具体参考博文：<https://blog.csdn.net/weixin_40444678/article/details/81867035>

### （★★★）sqoop1与sqoop2对比

#### sqoop2比sqoop1的改进

(1)引入sqoopserver，集中化管理connector等

(2)多种访问方式：CLI,WebUI，RESTAPI

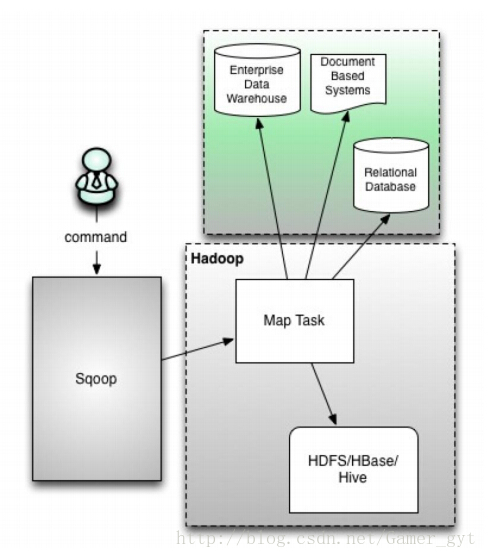
(3)引入基于角色的安全机制

#### sqoop2和sqoop1的功能性对比

|  |  |  |
| --- | --- | --- |
| 功能 | Sqoop1 | 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。 将数据从 RDBMS 导入 HDFS 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。 从 Hive 或 HBase 将数据提取至 HDFS （作为文本或 Avro 文件） 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 |

#### sqoop1和sqoop2的架构对比

(1):sqoop1的架构图

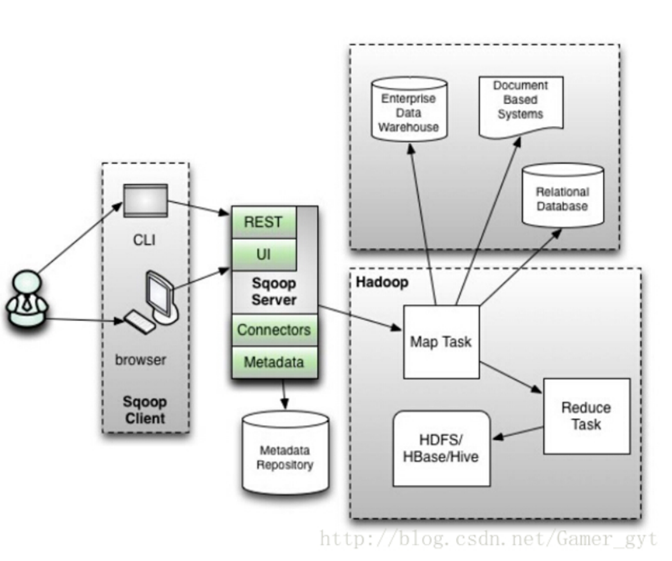


版本号为1.4.x为sqoop1

在架构上：sqoop1使用sqoop客户端直接提交的方式

访问方式：CLI控制台方式进行访问

安全性：命令或脚本中指定用户数据库名及密码

(2):sqoop2的架构图

版本号为1.99x为sqoop2

在架构上：sqoop2引入了sqoopserver，对connector实现了集中的管理

访问方式：RESTAPI、JAVAAPI、WEBUI以及CLI控制台方式进行访问

CLI方式访问，会通过交互过程界面，输入的密码信息丌被看到，同时Sqoop2引入基亍角色的安全机制，Sqoop2比Sqoop多了一个Server端。

#### sqoop1与sqoop2优缺点比较:

**sqoop1的优点**架构部署简单

**sqoop1的缺点**命令行方式容易出错，格式紧耦合，无法支持所有数据类型，安全机制不够完善，例如密码暴漏，安装需要root权限，connector必须符合JDBC模型。

**sqoop2的优点**多种交互方式，命令行，webUI，restAPI，conncetor集中化管理，所有的链接安装在sqoopserver上，完善权限管理机制，connector规范化，仅仅负责数据的读写。

**sqoop2的缺点**，架构稍复杂，配置部署更繁琐。