1.sqoop概述

1.1sqoop产生背景

基于传统关系型数据库的稳定性，还是有很多企业将数据存储在关系型数据库中；早期由于工具的缺乏，Hadoop与传统数据库之间的数据传输非常困难。基于前两个方面的考虑，需要一个在传统关系型数据库和Hadoop之间进行数据传输的项目，Sqoop应运而生。

1.2sqoop是什么

Sqoop是一个用于Hadoop和结构化数据存储（如关系型数据库）之间进行高效传输大批量数据的工具。它包括以下两个方面：

可以使用Sqoop将数据从关系型数据库管理系统(如MySQL)导入到Hadoop系统(如HDFS、Hive、HBase)中

将数据从Hadoop系统中抽取并导出到关系型数据库(如MySQL)

Sqoop的核心设计思想是利用MapReduce加快数据传输速度。也就是说Sqoop的导入和导出功能是通过基于Map Task（只有map）的MapReduce作业实现的。所以它是一种批处理方式进行数据传输，难以实现实时的数据进行导入和导出。

官网介绍：

```
Apache Sqoop(TM) is a tool designed for efficiently transferring bulk
data between Apache Hadoop and structured datastores such as relational databases.
```

1.3sqoop的特点

优点：它可以将跨平台的数据进行整合。

缺点：它不是很灵活。

```
mysql <--- > hdfs
mysql ---> hive
mysql ---> hbase

sqoop的重要的几个关键词？？
import ：  从关系型数据库到hadoop
export ：  从hadoop到关系型数据库。
```

2.sqoop的安装部署

```
1、解压配置环境变量
tar -zxvf /home/sqoop... -C /usr/local/sqoop...
vi /etc/profile

2、将sqoop/conf下面的sqoop-env-template.sh改名为sqoop-env.sh
mv ./conf/sqoop-env-template.sh ./conf/sqoop-env.sh
```

```
3.配置sqoop-env.sh
//根据自己的路径来
export HADOOP_COMMON_HOME=/usr/local/hadoop-2.7.1/
export HADOOP_MAPRED_HOME=/usr/local/hadoop-2.7.1/
export HIVE_HOME=/usr/local/hive-1.2.1/
export ZOOCFGDIR=/usr/local/zookeeper-3.4.7/

4、将mysql的驱动包导入到sqoop安装目录下的lib包下面
cp /home/mysql-connector-java-5.1.18.jar ./lib/

5、启动测试:
sqoop version
sqoop help
```

3.sqoop使用案例

sqoop官方文档

```
//参考文档
http://sqoop.apache.org/docs/1.4.6/SqoopUserGuide.html#_literal_sqoop_list_databases_literal
```

3.1查看mysql的数据库信息

```
sqoop list-databases -connect jdbc:mysql://mini1:3306 \
--username root --password root \
;
```

3.2(mysql--->hdfs)

```
//aaa 是表名字  -m是用m个map task去并行执行
//--target-dir 指定导入到hdfs的路径
sqoop import -connect jdbc:mysql://mini1:3306/test \
--driver com.mysql.jdbc.Driver \
--username root --password root \
--table aa -m 1 \
--fields-terminated-by '\t' --lines-terminated-by '\n' \
--null-string '\\N' --null-non-string '\\N' \
--target-dir /sqo/01 \
;
-----------------------------------------------------------------------------
//加上--columns可以指定某些列的导入
...
--table aa -m 1 \
--columns 'name' \
...
;
-----------------------------------------------------------------------------
//where和columns语句结合
```

```
...
--columns 'id,name' \
--where 'id>2' \
...
-------------------------------------------------------------------------
//使用query语句
//query替换table、column、where条件
//conditions结束符号
//必须要包含conditions
//--query和--table 不能同时存在
//--query 后面 双引号和单引号的区别（和shell差不多）：双引号里面$CONDITIONS 要写成
\$CONDITIONS 转义出来。
-m  默认是4个 task去跑
sqoop import -connect jdbc:mysql://mini1:3306/test \
--driver com.mysql.jdbc.Driver \
--username root --password root -m 1 \
--query 'select id,name from aa where id>2 and $CONDITIONS' \
--fields-terminated-by '\t' --lines-terminated-by '\n' \
--null-string '\\N' --null-non-string '\\N' \
--target-dir /sqo/04 --delete-target-dir \
;
-------------------------------------------------------------------------
//split-by语句
//用于指定用什么字段来进行分割数据
sqoop import -connect jdbc:mysql://mini1:3306/test \
--driver com.mysql.jdbc.Driver \
--username root --password root -m 2 \
--table aa \
--split-by id \
--fields-terminated-by '\t' --lines-terminated-by '\n' \
--null-string '\\N' --null-non-string '\\N' \
--target-dir /sqo/05 --delete-target-dir \
;
```

3.2.1文件存储格式为parquet

```
sqoop import \
--connect jdbc:mysql://mini1:3306/test \
--driver com.mysql.jdbc.Driver \
--username root \
--password root \
--table aa \
--target-dir /sqoop/import/user_parquet \
--delete-target-dir \
--m 1 \
--as-parquetfile

创建hive表

create table test_par1(
id int,
name string
```

```
)
stored as parquet
location '/sqoop/import/user_parquet/'
;
```

3.3mysql到hive

```
sqoop import --connect jdbc:mysql://localhost:3306/test \
--driver com.mysql.jdbc.Driver \
--username root --password root \
--table aa \
-m 1 \
--fields-terminated-by '\t' \
--lines-terminated-by '\n' \
--null-string '\\N' \
--null-non-string '\\N' \
--create-hive-table \
--hive-import \
--hive-overwrite \
--hive-table default.sqo \
--delete-target-dir \
;
```

3.3.1导入到hive的分区表

```
create table if not exists part2(
id int,
name string
)
partitioned by (dt string)
row format delimited fields terminated by ' '
;

alter table part2 add partition(dt='2019')

sqoop import -connect jdbc:mysql://mini1:3306/test \
--driver com.mysql.jdbc.Driver \
--username root \
--password root \
--table aa \
--hive-import \
--hive-overwrite \
--hive-table bg24.part2 \
--hive-partition-key  DT \
--hive-partition-value 2019 \
--fields-terminated-by ' ' \
;
```

```
//查看分区表信息

select * from part2;
```

3.4使用压缩导入(mysql--->hdfs)

```
--compress  默认是.gz
--compression-codec com.hadoop.compresion.lzo.lzoCode
```

3.5增量导入(mysql--->hdfs)

增量分为两种append和lastmodified（使用时间戳）

1.append导入

```
sqoop import --connect jdbc:mysql://mini1:3306/test \
--driver com.mysql.jdbc.Driver \
--username root --password roor \
--table stu \
-m 1 \
--incremental append \
--check-column id \
--last-value 0 \
--targer-dir /testsqo
```

2.lastmodified导入

```
create table if not exists part3(
id int,
name string,
time string
)
partitioned by (dt string)
row format delimited fields terminated by '\t'
;

--分区可以不用自己建

sqoop import \
--connect "jdbc:mysql://mini1:3306/test?useUnicode=true&characterEncoding=utf8" \
--username root \
--password root \
--table user \
--incremental lastmodified \
--check-column time \
--last-value '2019-09-20 18:52:20' \
--merge-key name \
```

```
--hive-import \
--hive-drop-import-delims \
--hive-database bg24 \
--hive-table part3 \
--hive-partition-key dt \
--hive-partition-value '2019' \
--fields-terminated-by '\t' \
-m 1
```

3.6sqoop的job

```
sqoop的job:
   --create <job-id>            Create a new saved job
   --delete <job-id>            Delete a saved job
   --exec <job-id>              Run a saved job
   --help                       Print usage instructions
   --list                       List saved jobs
   --meta-connect <jdbc-uri>    Specify JDBC connect string for the
                                metastore
   --show <job-id>              Show the parameters for a saved job
   --verbose                    Print more information while working

//查看sqoop的job信息
sqoop job --list

//创建job
sqoop job --create myjob -- import --connect jdbc:mysql://mini1:3306/test \
--driver com.mysql.jdbc.Driver \
--username root --password root \
--table stu \
-m 1 \
--incremental append \
--check-column id \
--last-value 0 \
--target-dir hdfs://mini1:9000/sq24/10

//执行job
sqoop job --exec myjob
```

3.7sqoop导出（hdfs--->mysql）

```
//update-mode 更新模式 updateonly | allowinsert
sqoop export -connect jdbc:mysql://mini1:3306/test \
--driver com.mysql.jdbc.Driver \
--username root
--password root
--table aa2 -m 1 \
--export-dir '/sqo/02' \
--input-fields-terminated-by '\t' \
```

```
--input-lines-terminated-by '\n' \
--input-null-string '\\N' \
--input-null-non-string '\\N' \
;

如下两个是用于，有主键的表的数据的更新（替换原来相同主键的信息）
--update-key
--update-mode allowinsert
可以将目标数据库中原来不存在的数据也导入到数据库表中。
即将存在的数据更新，不存在数据插入。
```

注意点：

```
mysql表的编码格式做为utf8，hdfs文件中的列数和mysql表中的字段数一样
导出暂不能由hive表导出mysql关系型数据库中
从hdfs到mysql时注意数据类型
--export-dir是一个hdfs中的目录，它不识别_SUCCESS文件
```