Apache Sqoop

序言

1. 内容介绍

本章详细介绍Apache Sqoop数据迁移工具的使用方法。

2. 理论目标

- 掌握Sqoop安装与配置
- 掌握数据的导入导出方法

3. 实践目标

- 无

4. 实践案例

无

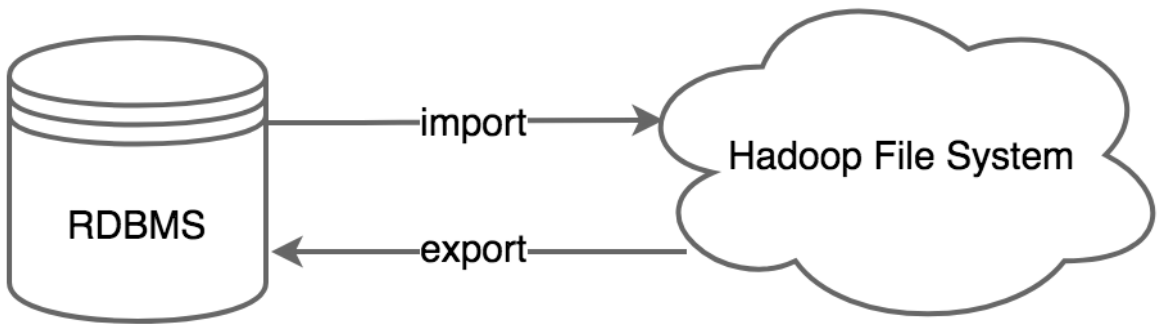
5. 内容目录

第01节 sqoop安装

1. 介绍

sqoop是apache旗下，用于关系型数据库和hadoop之间传输数据的工具，sqoop可以用在离线分析中，将保存在mysql的业务数据传输到hive数仓，数仓分析完得到结果，再通过sqoop传输到mysql，最后通过web+echart来进行图表展示，更加直观的展示数据指标。

如下图所示，sqoop中有导入和导出的概念，参照物都是hadoop文件系统，其中关系型数据库可以是mysql、oracle和db2，hadoop文件系统中可以是hdfs、hive和hbase等。执行sqoop导入和导出，其本质都是转化成了mr任务去执行。



2. sqoop安装

安装sqoop的前提是已经具备java和hadoop的环境。

1. 上传sqoop安装包至服务端　sqoop1.xxx sqoop1.99xx
2. 解压

```
tar -zxvf sqoop-1.4.7.bin__hadoop-2.6.0.tar.gz -C /opt/server/
```

3. 编辑配置文件

```
cd /opt/server/sqoop-1.4.7.bin__hadoop-2.6.0/conf
cp sqoop-env-template.sh sqoop-env.sh
vim sqoop-env.sh
# 加入以下内容
export HADOOP_COMMON_HOME=/opt/server/hadoop-3.1.0
export HADOOP_MAPRED_HOME=/opt/server/hadoop-3.1.0
export HIVE_HOME=/opt/server/apache-hive-3.1.2-bin
```

4. 加入mysql的jdbc驱动包

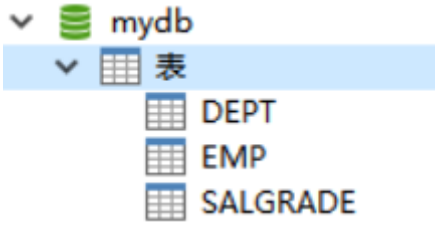
```
cd /opt/server/sqoop-1.4.7.bin__hadoop-2.6.0/lib
# mysql-connector-java-5.1.38.jar
```

第02节 导入数据

下面的语法用于将数据导入HDFS：

```
sqoop import (generic-args) (import-args)
```

在mysql中创建数据库mydb，执行mydb.sql脚本创建表。



1. 全量导入

全量导入mysql表数据到HDFS

下面的命令用于从MySQL数据库服务器中的emp表导入HDFS：

```
./sqoop import \
--connect jdbc:mysql://192.168.80.100:3306/mydb \
--username root \
--password root \
--delete-target-dir \
--target-dir /sqoopresult \
--table emp --m 1
```

其中--target-dir可以用来指定导出数据存放至HDFS的目录；

查看导入的数据：

```
cd /root/server/hadoop-3.1.0/bin
hadoop dfs -cat /sqoopresult/part-m-00000
```

使用where导入数据子集

--where参数可以指定从关系数据库导入数据时的查询条件，根据条件将数据库结果抽取到HDFS中。

```
bin/sqoop import \
--connect jdbc:mysql://node-1:3306/sqoopdb \
--username root \
--password hadoop \
--where "city ='sec-bad'" \
--target-dir /wherequery \
--table emp_add
--m 1
```

使用select导入数据子集

```
bin/sqoop import \
--connect jdbc:mysql://node-1:3306/userdb \
--username root \
--password hadoop \
--target-dir /wherequery12 \
--query 'select id,name,deg from emp WHERE  id>1203 and $CONDITIONS' \
--split-by id \
--fields-terminated-by '\t' \
--m 2
```

--split-by id通常配合-m参数配合使用，用于指定根据哪个字段进行划分并启动多少个maptask，上述语句使用时有以下注意事项：

- 使用query sql语句时不能使用参数 --table
- 语句中必须有where条件
- where条件后面必须有$CONDITIONS 字符串
- 查询字符串必须使用单引号

2. 增量导入

在实际工作当中，数据的导入往往只需要导入新增数据即，sqoop支持根据某些字段进行增量数据导入：

--check-column (col)

用于指定增量导入数据时所依赖的列，一般指定自增字段或时间戳，同时支持指定多列。

--incremental (mode)

用于指定根据何种模式导入增量数据：

- append：配合--last-value参数使用，对大于last-value的值进行增量导入
- lastmodified：配合--last-value参数使用，追加last-value指定的日期之后的记录

--last-value (value)

指定自从上次导入数据后列的最大值，此值在Sqoop作业中会自动进行更新

Append模式增量导入

增量导入empno为7934后的数据：

```
./sqoop import \
--connect jdbc:mysql://192.168.80.100:3306/mydb \
--username root  --password root \
--table emp --m 1 \
--target-dir /appendresult \
--incremental append \
--check-column empno \
--last-value  7934
```

Lastmodified模式增量导入

创建示例表：

```
create table test(
    id int,
    name varchar(20),
    last_mod timestamp default current_timestamp on update current_timestamp
);
-- last_mod为时间戳类型，并设置为在数据更新时都会记录当前时间，同时默认值为当前时间
```

使用incremental的方式进行增量的导入：

```
bin/sqoop import \
--connect jdbc:mysql://node-1:3306/userdb \
--username root \
--password hadoop \
--table customertest \
--target-dir /lastmodifiedresult \
--check-column last_mod \
--incremental lastmodified \
--last-value "2021-09-28 18:55:12" \
--m 1 \
--append
```

注意：采用lastmodified模式去处理增量时，会将大于等于last-value值的数据当做增量插入

使用lastmodified模式进行增量导入时，需要指定增量数据是以***append***模式(附加)还是***merge-key***(合并)模式导入

```
bin/sqoop import \
--connect jdbc:mysql://node-1:3306/userdb \
--username root \
--password hadoop \
--table customertest \
--target-dir /lastmodifiedresult \
--check-column last_mod \
--incremental lastmodified \
--last-value "2021-09-28 18:55:12" \
--m 1 \
--merge-key id
```

merge-key，如果之前旧的数据发生了变化，则不会再以追加形式导入，而是会以更新的形式导入

第03节 导出数据

将数据从Hadoop生态体系导出到数据库时，**数据库表必须已经创建。**

默认的导出操作是使用INSERT语句将数据插入到目标表中，也可以指定为更新模式，sqoop则会使用update语句更新目标数据。

1. 默认方式导出

因为默认是使用insert语句导出到目标表，如果数据库中的表具有约束条件，例如主键不能重复，如果违反了违反了约束条件，则会导致导出失败。

导出时，可以指定全部字段或部分字段导出到目标表。

```
./sqoop export \
--connect jdbc:mysql://server:3306/userdb \
--username root \
--password root \
--table employee \
--export-dir /emp/emp_data
# 将HDFS/emp/emp_data文件中的数据导出到mysql的employee表中
```

导出时，还可以指定以下参数；

--input-fields-terminated-by '\t'：指定HDFS文件的分隔符

--columns：如果不使用 column 参数，就要默认Hive 表中的字段顺序和个数和MySQL表保持一致，如果字段顺序或个数不一致可以加 columns 参数进行导出控制，没有被包含在–columns后面列名或字段要么具备默认值，要么就允许插入空值。否则数据库会拒绝接受sqoop导出的数据，导致Sqoop作业失败、

--export-dir：导出目录，在执行导出的时候，必须指定此参数

2. updateonly模式

-- update-key，更新标识，即指定根据某个字段进行更新，例如id，可以指定多个更新标识的字段，多个字段之间用逗号分隔。

-- updatemod，指定updateonly（默认模式），指仅仅更新已存在的数据记录，不会插入新纪录。

```
bin/sqoop export \
--connect jdbc:mysql://node-1:3306/userdb \
--username root --password hadoop \
--table updateonly \
--export-dir /updateonly_2/ \
--update-key id \
--update-mode updateonly
# 根据id对已经存在的数据进行更新操作，新增的数据会被忽略
```

3. allowinsert模式

--updatemod，指定allowinsert，更新已存在的数据记录，同时插入新纪录。实质上是一个insert & update的操作。

```
bin/sqoop export \
--connect jdbc:mysql://node-1:3306/userdb \
--username root --password hadoop \
--table allowinsert \
--export-dir /allowinsert_2/ \
--update-key id \
--update-mode allowinsert
# 根据id对已经存在的数据进行更新操作，同时导出新增的数据
```

第04节 Sqoop job作业

创建作业(--create):

在这里，我们创建一个名为myjob，这可以从RDBMS表的数据导入到HDFS作业。下面的命令用于创建一个从DB数据库的emp表导入到HDFS文件的作业。

```
./sqoop job --create job_test1 \
-- import \
--connect jdbc:mysql://192.168.80.100:3306/mydb \
--username root \
--password root \
--target-dir /sqoopresult333 \
--table emp --m 1
# import前要有空格
```

验证job

--list参数是用来查看保存的作业：

```
bin/sqoop job --list
bin/sqoop job --delete  job_test6
```

执行job

--exec 选项用于执行保存的作业:

```
bin/sqoop job --exec myjob
```