Sqoop

Sqoop Help

```
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
  import-mainframe     Import datasets from a mainframe server 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
```

#Basic Syntax

sqoop [command] \

--connect

[JDBC(Java Data Base Connector) connector to different databases]

[Hostname (type hostname on terminal)]

--username

--password

--target-dir or --warehouse-dir

(location on HDFS where the data needs to be stored)

--table

(specific table that needs to be imported. Hence, usually used with import command)

--query

(For specific SQL query usually used with eval command)

-m

(number of mappers default is 4 in VM)

--append

(used to append data)

--delete-target-dir

(used to overwrite data)

--Split-by [column name]

--query

--hive-import

--hive-database

--create-hive-table

--delete-target-dir

Example 1

```
#Import-all-tables command
sqoop import-all-tables \
--connect jdbc:mysql://quickstart.cloudera:3306/retail_db \
--username retail_dba \
-P \
--warehouse-dir sqoop_import/retail_db
# validate
hadoop fs -ls sqoop_import/retail_db/
```

Example 2

```
sqoop import-all-tables
--connect jdbc:mysql://quickstart.cloudera:3306/classic
--username classic_dba
-P
--warehouse-dir sqoop_import/classic
-m 1
#Validate
hadoop fs -tail sqoop_import/classic/products/part-m-00000
```

Example 3

```
sqoop import
--connect jdbc:mysql://quickstart.cloudera:3306/retail_db
--username retail_dba
-P
--table orders
--split-by order_status
--target-dir sqoop_import/orderbysplit
-m 1
# validate
hadoop fs -tail sqoop_import/orderbysplit1/part-m-00000
```

Example 4 : query (alternative to table – can’t be used together)

```
#Using eval command
sqoop eval
--connect jdbc:mysql://quickstart.cloudera:3306/classic
--username classic_dba
-P
--query "select * from orders limit 10"
```

Or

```
sqoop eval
--connect jdbc:mysql://quickstart.cloudera:3306/classic
--username classic_dba
-P
--query "describe orders"
```

```
#Example 5
#Different format
```

```
#Sequence files
#This will generate meta data of sequence file
sqoop import
--connect jdbc:mysql://quickstart.cloudera:3306/retail_db
--username retail_dba
-P
--table orders
--split-by order_status
--target-dir sqoop_import/orderbysplit
--delete-target-dir
--as-sequencefile
```

Validate

```
hadoop fs -tail sqoop_import/orderbysplit/part-m-00000
```

Example 6

Store the table order_items stored as avrofiles

```
sqoop import
--connect jdbc:mysql://quickstart.cloudera:3306/retail_db
--username retail_dba
-P
--table order_items
--target-dir sqoop_import/order_items_avro
--as-avrodatafile
```

Example 7: only get few columns

```
sqoop import

--connect jdbc:mysql://quickstart.cloudera:3306/retail_db

--username retail_dba

-P

--table orders

--columns "order_id,order_status"

--target-dir sqoop_import/order_items_order_status

# validate

hadoop fs -tail sqoop_import/order_items_order_status/part-m-00000
```

Example 8: Using the where clause

```
sqoop import

--connect jdbc:mysql://quickstart.cloudera:3306/retail_db

--username retail_dba

-P

--query "select * from orders where \$CONDITIONS and order_status = 'COMPLETE'"

--target-dir sqoop_import/query

--delete-target-dir

-m 1

#validate

hadoop fs -tail sqoop_import/query/part-m-00000
```

Example 9: importing as avro files

```
sqoop import-all-tables

--connect jdbc:mysql://quickstart.cloudera:3306/retail_db
```

```
--username retail_dba
-P
--as-avrodatafile
--warehouse-dir sqoop_import/avrofiles
# validate
hadoop fs -tail sqoop_import/avrofiles/products/part-m-00003.avro
```

#Next List of commands

#Output line formatting arguments

#Default delimiters in sqoop “,”

Override can be done with

```
--fields-terminated-by
--myql-delimiters
--lines-terminated-by
```

and others

Changing delimiters

Example 10:

```
sqoop import
--connect jdbc:mysql://quickstart.cloudera/classic
--username classic_dba
--password classic
--table payments
--target-dir practice/payments
--delete-target-dir
--fields-terminated-by "|"
#validate
hadoop fs -tail practice/payments/part-m-00000
```

Example 11: import classic.orders fields terminated by char X and lines terminated by tab

```
sqoop import
--connect jdbc:mysql://quickstart.cloudera/classic
--username classic_dba
--password classic
--table orders
--target-dir practice/orders
--delete-target-dir
--fields-terminated-by X
--lines-terminated-by "\t"
```

```
# validate
hadoop fs -tail practice/orders/part-m-00003
```

-compression

Example 12: import classic.orderdetails fields terminated by char Z and lines terminated by “\n” and compress and store a text compressed file -snappy compression

```
sqoop import
--connect jdbc:mysql://quickstart.cloudera/classic
--username classic_dba
--password classic
--table orderdetails
--target-dir practice/orderdetails
--delete-target-dir
--fields-terminated-by Z
--compress
--compression-codec=snappy
```

Example 13: import classic.customers and compress and store as **one** avro file compressed file -snappy compression

```
sqoop import

--connect jdbc:mysql://quickstart.cloudera/classic

--username classic_dba

--password classic

--table customers

--target-dir practice/customers

--delete-target-dir

--compress

--compression-codec=snappy

--as-avrodatafile

-m 1

#validate

hadoop fs -ls practice/customers
```

Task 3

Hive related import

Create a hive database

```
hive -e "create database sqoopdatabase"


sqoop import
--connect jdbc:mysql://quickstart.cloudera:3306/classic
--username classic_dba
-P
--table orders
--hive-import
--hive-database sqoopdatabase
```

Or (if –hive-database does not work)

```
sqoop import
--connect jdbc:mysql://quickstart.cloudera:3306/classic
--username classic_dba
-P
--table orders
--hive-import
--hive-table sqoopdatabase.orders
# validate
hadoop fs -ls /user/hive/warehouse/sqoopdatabase.db/orders
or
# using hive
--hive-overwrite
# if table already exist
```

#Some avro tools

```
avro-tools
```

you will get a list of avro commands

avro is a binary JSON format

#Get one file to the local file

```
hadoop fs -get sqoop_import/avrofiles/products/part-m-00003.avro .
```

1) Lets get the schema of the file

```
avro-tools getschema part-m-00003.avro
```

2) converting into a JSON file

```
avro-tools tojson part-m-00003.avro
```

3) store that into a file

```
avro-tools tojson part-m-00003.avro >> part-m-00003.json
view part-m-00003.json
```