To see list of commands use

sqoop help

It would look like this:

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

See 'sqoop help COMMAND' for information on a specific command.

**Import from SQL to HDFS**

**Check mysql table**

Connect to mysql

mysql -h 10.142.1.2 -u sqoopuser -pNHkkP876rp

On mysql prompt, run the following:

use sqoopex

select \* from widgets

**Import**

If widgets is existing in hdfs, please delete it:

hadoop fs -rmr widgets

Run the import command:

sqoop import --connect jdbc:mysql://10.142.1.2/sqoopex --table widgets -m 2 --username sqoopuser --password NHkkP876rp --split-by id

**mysql to hive**

sqoop import --connect jdbc:mysql://10.142.1.2/sqoopex --table widgets -m 2 --hive-import --username sqoopuser --hive-database sandeepgiri9034 --split-by id --password NHkkP876rp

**mysql to HBase**

sqoop import --connect jdbc:mysql://ip-172-31-20-247/sqoopex --table widgets --hbase-table 'sgiri:widgets' --column-family cf2 --username sqoopuser --hbase-create-table --columns id,widget\_name --hbase-row-key id -m 1 --password NHkkP876rp

sqoop import --connect jdbc:mysql://ip-172-31-20-247/sqoopex --table widgets --hbase-table 'sgiri:widgets' --column-family cf1 --username sqoopuser --hbase-create-table --columns id,lastUpdated --hbase-row-key id -m 1 --password NHkkP876rp

**Hive to MySQL**

Copy sales.log locally

hadoop fs -cp /data/hive/sales.log .

Launch hive using command: hive

Create Hive Table:

use sg;

CREATE TABLE sales\_test(widget\_id INT, qty INT,

street STRING, city STRING, state STRING,

zip INT, sale\_date STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

Find the location of your table using:

describe formatted sales\_test

Load Data:

LOAD DATA INPATH "sales.log" INTO TABLE sales\_test;

Select rows to see data:

select \* from sales\_test;

**Documentation**

To see help documentation of import use:

sqoop help import

This looks like the following:

[sandeepgiri9034@cxln4 ~]$ sqoop help import

SLF4J: Class path contains multiple SLF4J bindings.

SLF4J: Found binding in [jar:file:/usr/hdp/2.6.2.0-205/hadoop/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: Found binding in [jar:file:/usr/hdp/2.6.2.0-205/accumulo/lib/slf4j-log4j12.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: See http://www.slf4j.org/codes.html#multiple\_bindings for an explanation.

SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]

19/09/12 04:19:56 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6.2.6.2.0-205

usage: sqoop import [GENERIC-ARGS] [TOOL-ARGS]

Common arguments:

--connect Specify JDBC

connect

string

--connection-manager Specify

connection

manager

class name

--connection-param-file Specify

connection

parameters

file

--driver Manually

specify JDBC

driver class

to use

--hadoop-home Override

$HADOOP\_MAPR

ED\_HOME\_ARG

--hadoop-mapred-home Override

$HADOOP\_MAPR

ED\_HOME\_ARG

--help Print usage

instructions

--metadata-transaction-isolation-level Defines the

transaction

isolation

level for

metadata

queries. For

more details

check

java.sql.Con

nection

javadoc or

the JDBC

specificaito

n

-P Read

password

from console

--password Set

authenticati

on password

--password-alias Credential

provider

password

alias

--password-file Set

authenticati

on password

file path

--relaxed-isolation Use

read-uncommi

tted

isolation

for imports

--skip-dist-cache Skip copying

jars to

distributed

cache

--temporary-rootdir Defines the

temporary

root

directory

for the

import

--username Set

authenticati

on username

--verbose Print more

information

while

working

Import control arguments:

--append Imports data

in append

mode

--as-avrodatafile Imports data

to Avro data

files

--as-parquetfile Imports data

to Parquet

files

--as-sequencefile Imports data

to

SequenceFile

s

--as-textfile Imports data

as plain

text

(default)

--autoreset-to-one-mapper Reset the

number of

mappers to

one mapper

if no split

key

available

--boundary-query Set boundary

query for

retrieving

max and min

value of the

primary key

--columns Columns to

import from

table

--compression-codec Compression

codec to use

for import

--delete-target-dir Imports data

in delete

mode

--direct Use direct

import fast

path

--direct-split-size Split the

input stream

every 'n'

bytes when

importing in

direct mode

-e,--query Import

results of

SQL

'statement'

--fetch-size Set number

'n' of rows

to fetch

from the

database

when more

rows are

needed

--inline-lob-limit Set the

maximum size

for an

inline LOB

-m,--num-mappers Use 'n' map

tasks to

import in

parallel

--mapreduce-job-name Set name for

generated

mapreduce

job

--merge-key Key column

to use to

join results

--split-by Column of

the table

used to

split work

units

--split-limit Upper Limit

of rows per

split for

split

columns of

Date/Time/Ti

mestamp and

integer

types. For

date or

timestamp

fields it is

calculated

in seconds.

split-limit

should be

greater than

0

--table Table to

read

--target-dir HDFS plain

table

destination

--validate Validate the

copy using

the

configured

validator

--validation-failurehandler Fully

qualified

class name

for

ValidationFa

ilureHandler

--validation-threshold Fully

qualified

class name

for

ValidationTh

reshold

--validator Fully

qualified

class name

for the

Validator

--warehouse-dir HDFS parent

for table

destination

--where WHERE clause

to use

during

import

-z,--compress Enable

compression

Incremental import arguments:

--check-column Source column to check for incremental

change

--incremental Define an incremental import of type

'append' or 'lastmodified'

--last-value Last imported value in the incremental

check column

Output line formatting arguments:

--enclosed-by Sets a required field enclosing

character

--escaped-by Sets the escape character

--fields-terminated-by Sets the field separator character

--lines-terminated-by Sets the end-of-line character

--mysql-delimiters Uses MySQL's default delimiter set:

fields: , lines: \n escaped-by: \

optionally-enclosed-by: '

--optionally-enclosed-by Sets a field enclosing character

Input parsing arguments:

--input-enclosed-by Sets a required field encloser

--input-escaped-by Sets the input escape

character

--input-fields-terminated-by Sets the input field separator

--input-lines-terminated-by Sets the input end-of-line

char

--input-optionally-enclosed-by Sets a field enclosing

character

Hive arguments:

--create-hive-table Fail if the target hive

table exists

--hive-compute-stats Overwrite existing data in

the Hive table

--hive-database Sets the database name to

use when importing to hive

--hive-delims-replacement Replace Hive record \0x01

and row delimiters (\n\r)

from imported string fields

with user-defined string

--hive-drop-import-delims Drop Hive record \0x01 and

row delimiters (\n\r) from

imported string fields

--hive-home Override $HIVE\_HOME

--hive-import Import tables into Hive

(Uses Hive's default

delimiters if none are

set.)

--hive-overwrite Overwrite existing data in

the Hive table

--hive-partition-key Sets the partition key to

use when importing to hive

--hive-partition-value Sets the partition value to

use when importing to hive

--hive-table Sets the table name to use

when importing to hive

--map-column-hive Override mapping for

specific column to hive

types.

HBase arguments:

--column-family Sets the target column family for the

import

--hbase-bulkload Enables HBase bulk loading

--hbase-create-table If specified, create missing HBase tables

--hbase-row-key Specifies which input column to use as the

row key

--hbase-table Import to

|  |
| --- |
|  |

in HBase

HCatalog arguments:

--hcatalog-database HCatalog database name

--hcatalog-home Override $HCAT\_HOME

--hcatalog-partition-keys Sets the partition

keys to use when

importing to hive

--hcatalog-partition-values Sets the partition

values to use when

importing to hive

--hcatalog-table HCatalog table name

--hive-home Override $HIVE\_HOME

--hive-partition-key Sets the partition key

to use when importing

to hive

--hive-partition-value Sets the partition

value to use when

importing to hive

--map-column-hive Override mapping for

specific column to

hive types.

HCatalog import specific options:

--create-hcatalog-table Create HCatalog before import

--drop-and-create-hcatalog-table Drop and Create HCatalog before

import

--hcatalog-storage-stanza HCatalog storage stanza for table

creation

Accumulo arguments:

--accumulo-batch-size Batch size in bytes

--accumulo-column-family Sets the target column family for

the import

--accumulo-create-table If specified, create missing

Accumulo tables

--accumulo-instance Accumulo instance name.

--accumulo-max-latency Max write latency in milliseconds

--accumulo-password Accumulo password.

--accumulo-row-key Specifies which input column to

use as the row key

--accumulo-table

|  |
| --- |
|  |

Import to

|  |
| --- |
|  |

in Accumulo

--accumulo-user Accumulo user name.

--accumulo-visibility Visibility token to be applied to

all rows imported

--accumulo-zookeepers Comma-separated list of

zookeepers (host:port)

Code generation arguments:

--bindir Output directory for compiled

objects

--class-name Sets the generated class name.

This overrides --package-name.

When combined with --jar-file,

sets the input class.

--input-null-non-string Input null non-string

representation

--input-null-string Input null string representation

--jar-file Disable code generation; use

specified jar

--map-column-java Override mapping for specific

columns to java types

--null-non-string Null non-string representation

--null-string Null string representation

--outdir Output directory for generated

code

--package-name Put auto-generated classes in

this package

Generic Hadoop command-line arguments:

(must preceed any tool-specific arguments)

Generic options supported are

-conf specify an application configuration file

-D use value for given property

-fs specify a namenode

-jt specify a ResourceManager

-files specify comma separated files to be copied to the map reduce cluster

-libjars specify comma separated jar files to include in the classpath.

-archives specify comma separated archives to be unarchived on the compute machines.

The general command line syntax is

bin/hadoop command [genericOptions] [commandOptions]

At minimum, you must specify --connect and --table

Arguments to mysqldump and other subprograms may be supplied

after a '--' on the command line.