Apache Sqoop

BAS Academy

Agenda

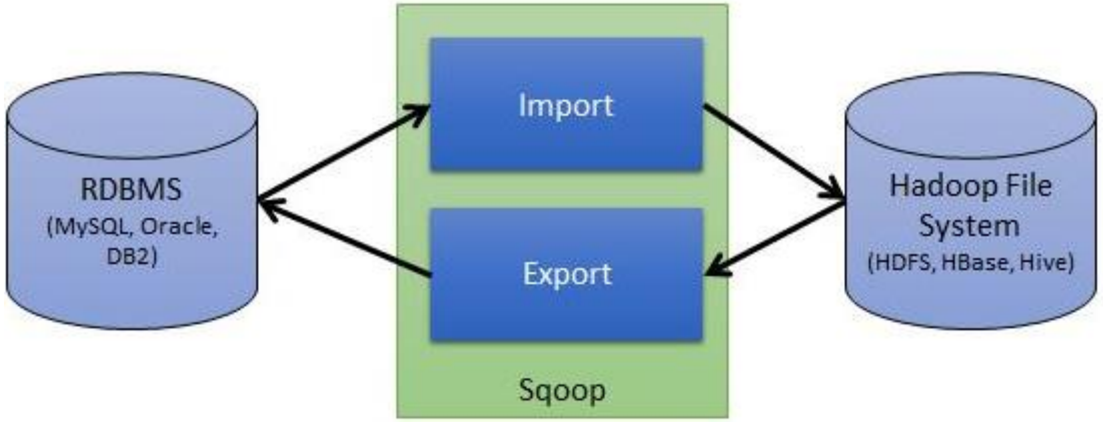
- About Scoop
- Sqoop Import
- Incremental Import
- Sqoop Commands and Arguments
- Sqoop Export
- Integration with Ecosystem
- Hands On

About Sqoop

What is Sqoop

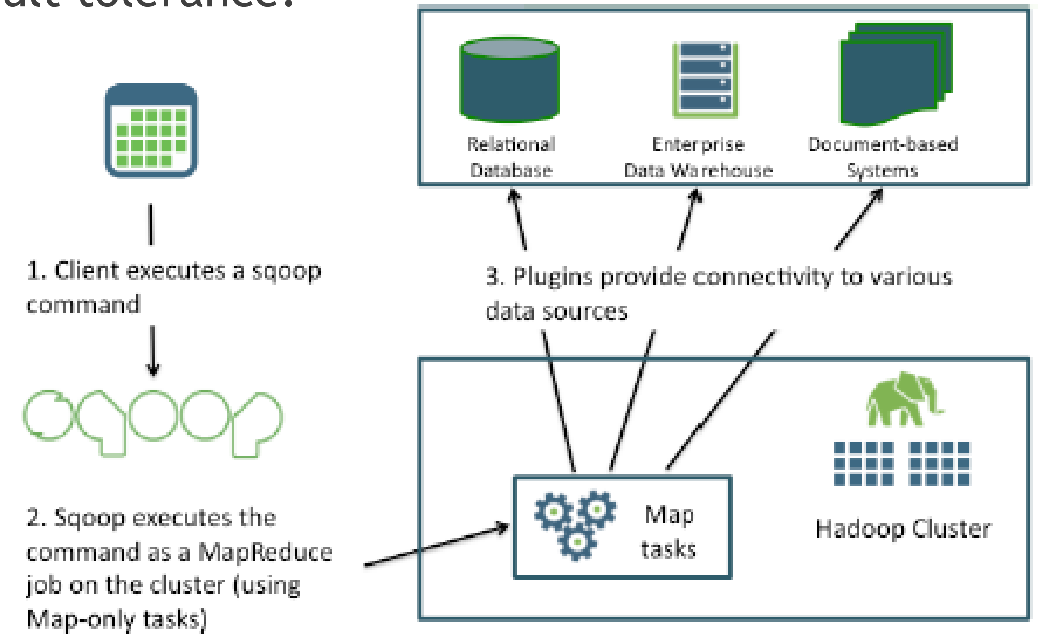
- **Sqoop:** “SQL to Hadoop and Hadoop to SQL”

- Sqoop is a tool designed to transfer data between Hadoop and relational database servers.
- It is used to import data from relational databases such as MySQL, Oracle to Hadoop HDFS, and export from Hadoop file system to relational databases.
- It is provided by the Apache Software Foundation.



Sqoop Architecture

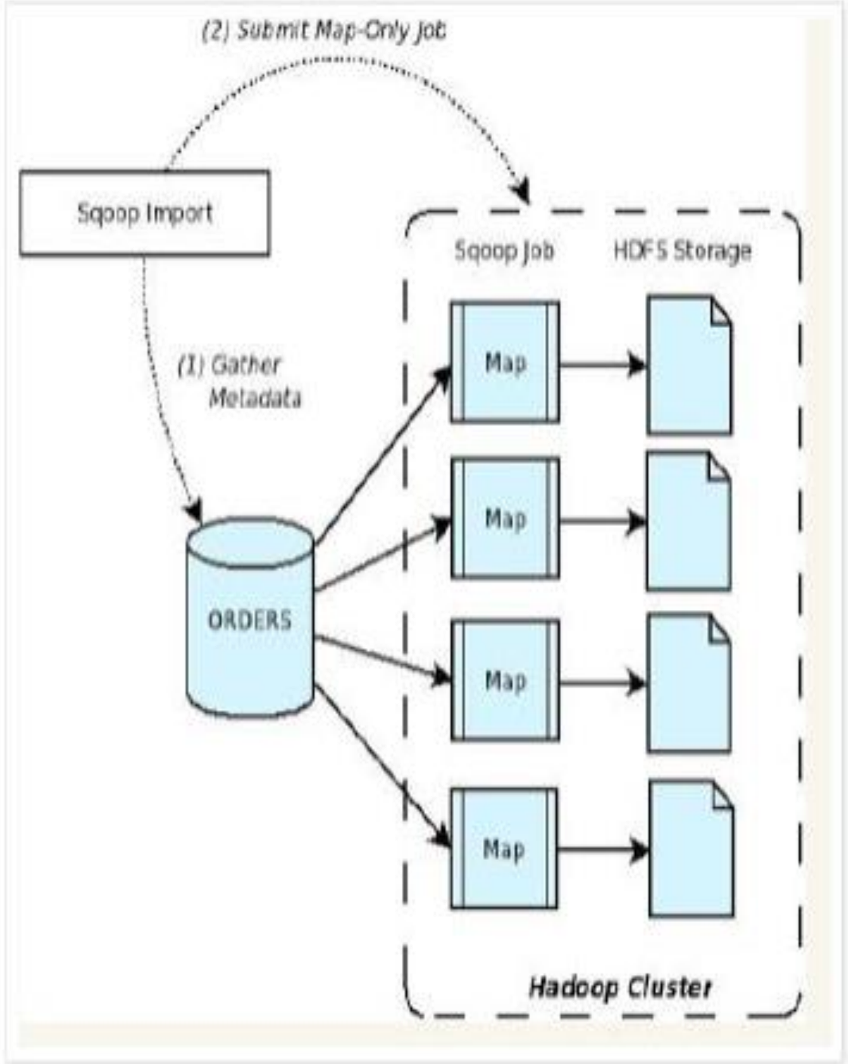
- Sqoop uses a connector-based architecture that supports plugins that provide connectivity to additional external systems.
- Sqoop uses MapReduce to distribute its work across the Hadoop cluster
- Using MapReduce to perform Sqoop commands provides parallel operation as well as fault tolerance.


Relational Database
Enterprise Data Warehouse
Document-based Systems
1. Client executes a sqoop command
3. Plugins provide connectivity to various data sources
2. Sqoop executes the command as a MapReduce job on the cluster (using Map-only tasks)
Map tasks
Hadoop Cluster

Sqoop Import

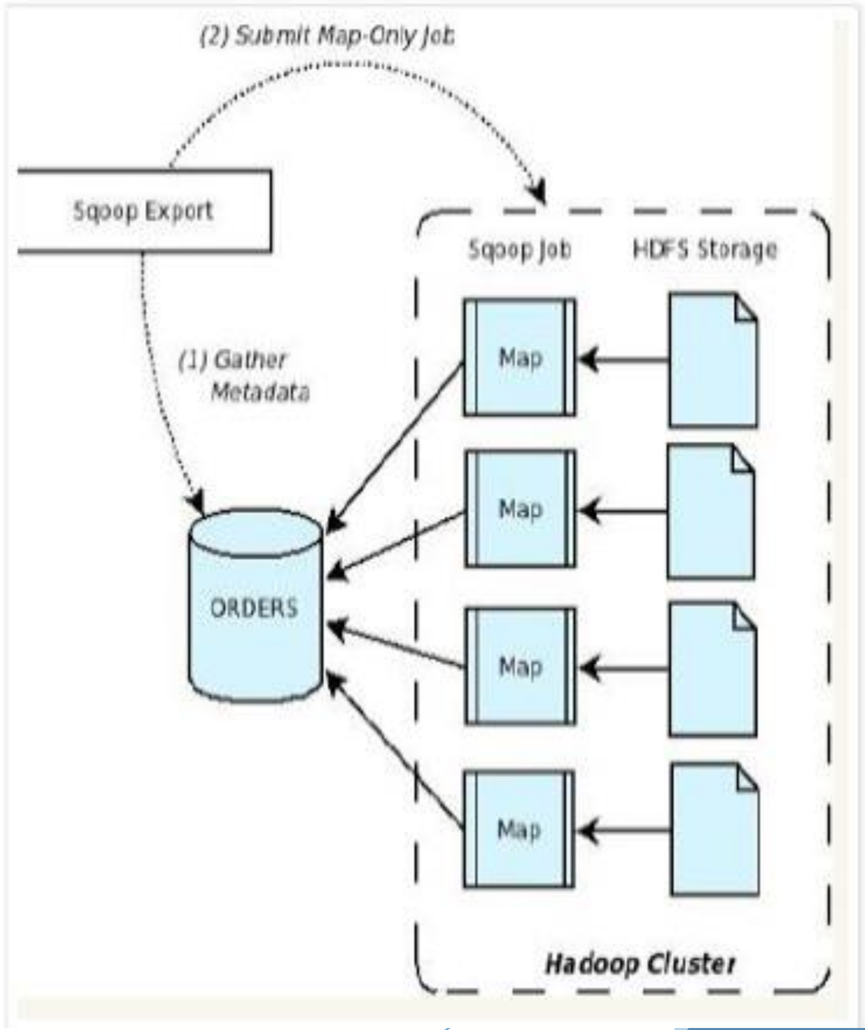
- The input to the import process is a database table
- Sqoop will read the table row-by-row into HDFS.
- The output of this import process is a set of files containing a copy of the imported table.

- The import process is performed in parallel. For this reason, the output will be in multiple files.
- These files may be delimited text files (for example, with commas or tabs separating each field) or binary Avro or SequenceFiles



Sqoop Export

- After manipulating the imported records (for example, with MapReduce or Hive) you may have a result data set which you can then export back to the relational database.
- The export tool exports a set of files from HDFS back to an RDBMS.
- The files given as input to Sqoop contain records, which are called as rows in table.
- Those are read in parallel and parsed into a set of records and inserted them as new rows in a target database table



Sqoop Import

Importing Tables

The import command has the following requirements:

- Must specify a connect string using the -- connect argument
- Can include credentials in the connect string, using the --username and –password arguments
- Must specify either a table to import using --table or the result of an SQL query using –query
- The default number of map tasks for Sqoop is four, so the result of default import will be in four files

```
sqoop import
 --connect jdbc:mysql://host/nyse
 --table StockPrices
 --target-dir /data/stockprice/
 --as-textfile
```

File Format

Sqoop supports Text and Binary files.

- Text

Import into text file using the --as-textfile parameter:

- Binary formats are a natural fit for storing binary values like images or PDF documents
- To access the binary data, you need to implement extra functionality or load special libraries in your application

Binary Files:

- Avro

Import into Avro file by specifying the --as-avrodatafile parameter

- SequenceFile

Import into SequenceFile using the --as-sequencefile parameter:

Import All Tables

- Sqoop has the feature to import all the tables at once
- Tables will be imported in sequential order

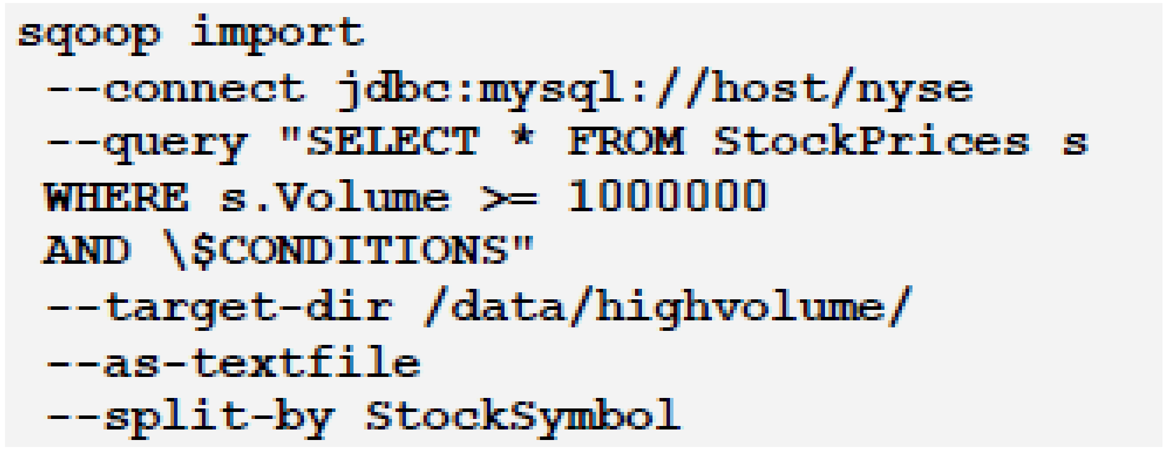
```
sqoop import-all-tables \
  --connect jdbc:mysql://mysql.example.com/sqoop \
  --username sqoop \
  --password sqoop
```

- Option to exclude few tables

```
sqoop import-all-tables \
  --connect jdbc:mysql://mysql.example.com/sqoop \
  --username sqoop \
  --password sqoop \
  --exclude-tables cities,countries
```

Freeform SQL

- Instead of using table import, use free-form query import.
- Use the --query argument to specify which rows to select from a table.
- Sqoop will not use the database catalog to fetch the metadata



```
sqoop import
 --connect jdbc:mysql://host/nyse
 --query "SELECT * FROM StockPrices s
 WHERE s.Volume >= 1000000
 AND \$CONDITIONS"
 --target-dir /data/highvolume/
 --as-textfile
 --split-by StockSymbol
```

- --split-by parameter with the column is used for slicing the data into multiple parallel tasks.
- Using --query is limited to simple queries

Incremental Import

Incremental Import

- Sqoop provides an incremental import mode which can be used to retrieve only rows newer than some previously-imported set of rows.

Sqoop supports two types of incremental imports:

- append

For importing the newly created rows to the existing data

- lastmodified

Used when existing rows needs to be updated

- --incremental argument is used for incremental import

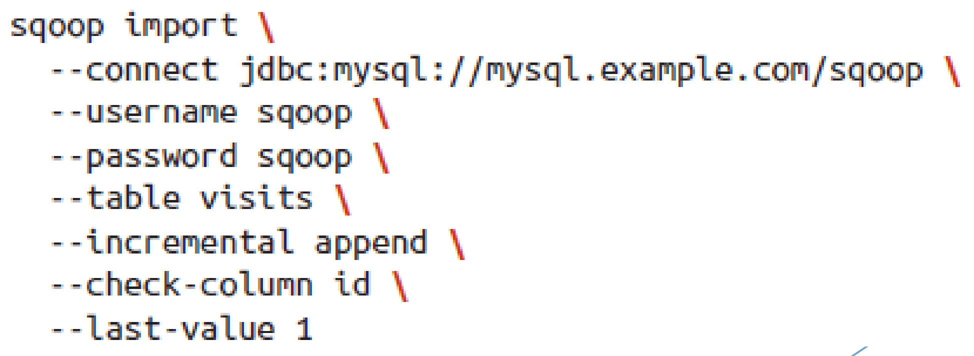
Incremental Import – Append

- Incremental import in append mode will allow you to transfer only the newly created rows.

Incremental import also requires two additional parameters:

- --check-column indicates a column name that should be checked for newly appended data
- -last-value contains the last value that successfully imported into Hadoop.

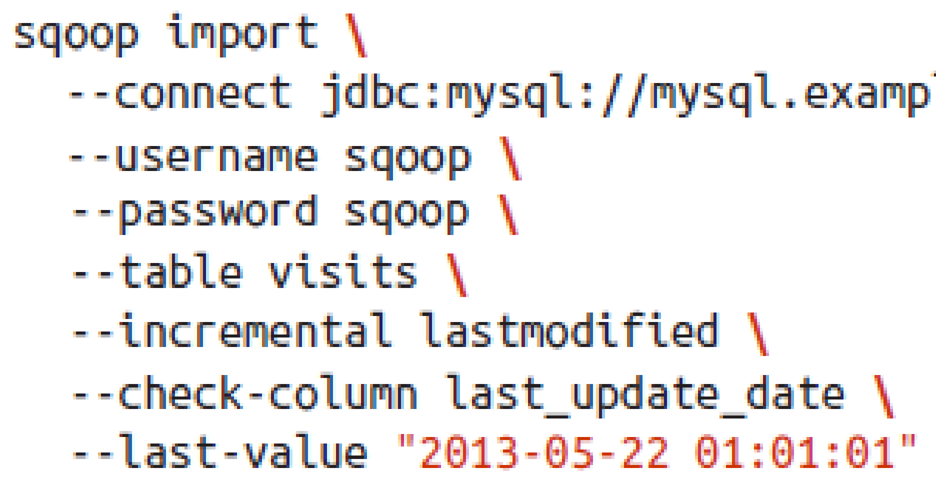
- Sqoop when running in incremental mode, always prints out the value of the last imported row.



```
sqoop import \
  --connect jdbc:mysql://mysql.example.com/sqoop \
  --username sqoop \
  --password sqoop \
  --table visits \
  --incremental append \
  --check-column id \
  --last-value 1
```

Incremental Import – Lastmodified

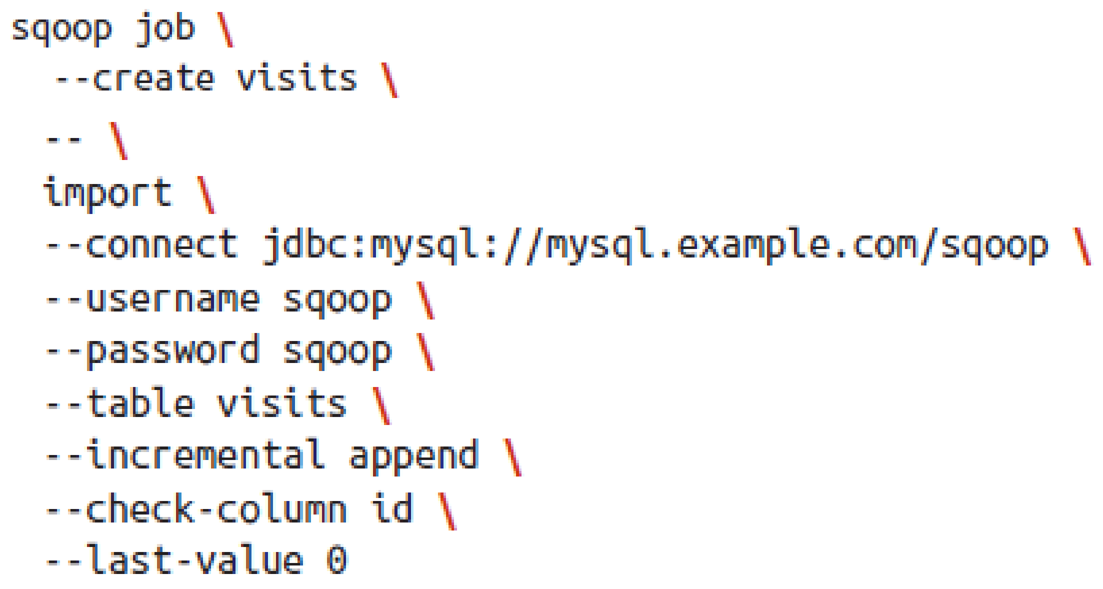
- This is used for mutable data. The data which is getting changed
- Use the lastmodified mode instead of the append mode.
- The incremental mode lastmodified requires a column holding a date value (suitable types are date, time, datetime, and timestamp) containing information as to when each row was last updated.



```
sqoop import \
  --connect jdbc:mysql://mysql.example.com/sqoop \
  --username sqoop \
  --password sqoop \
  --table visits \
  --incremental lastmodified \
  --check-column last_update_date \
  --last-value "2013-05-22 01:01:01"
```

Sqoop Jobs

- The Sqoop metastore allows you to retain your job definitions and to easily run them anytime.
- Each saved job has a logical name that is used for referencing.
- To create a sqoop job:



```
sqoop job \
  --create visits \
  -- \
  import \
  --connect jdbc:mysql://mysql.example.com/sqoop \
  --username sqoop \
  --password sqoop \
  --table visits \
  --incremental append \
  --check-column id \
  --last-value 0
```

- List all retained jobs using the --list parameter

sqoop job –list

- View content of the saved job definitions using the –show parameter

sqoop job --show visits

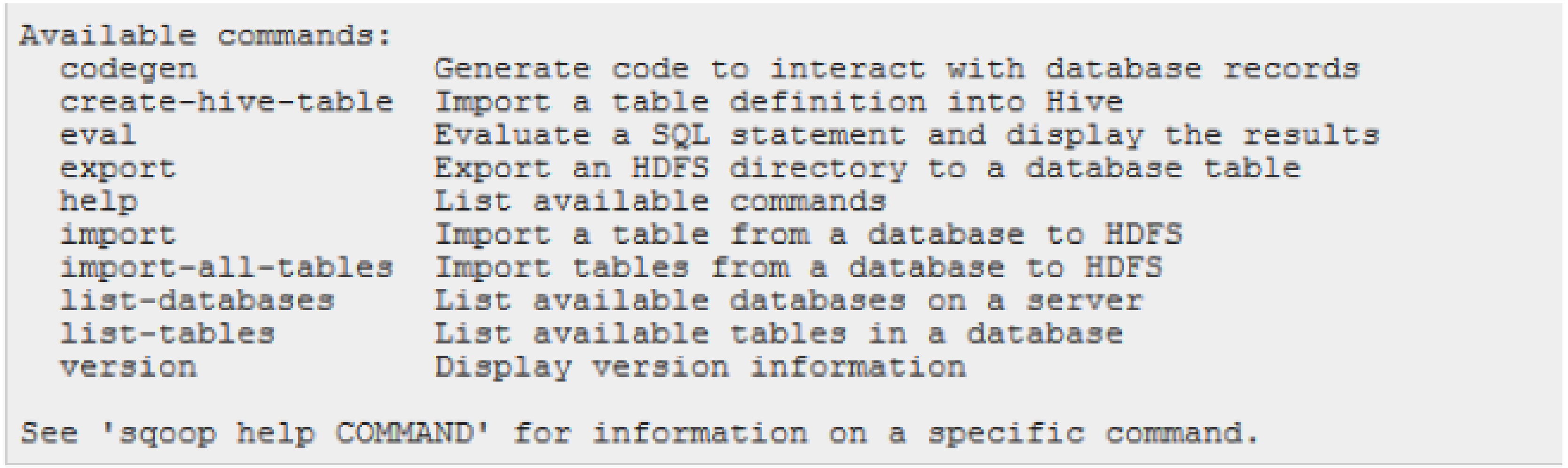
- Remove the old job definitions that are no longer needed with the -delete parameter

sqoop job --delete visits

Sqoop Commands and Arguments

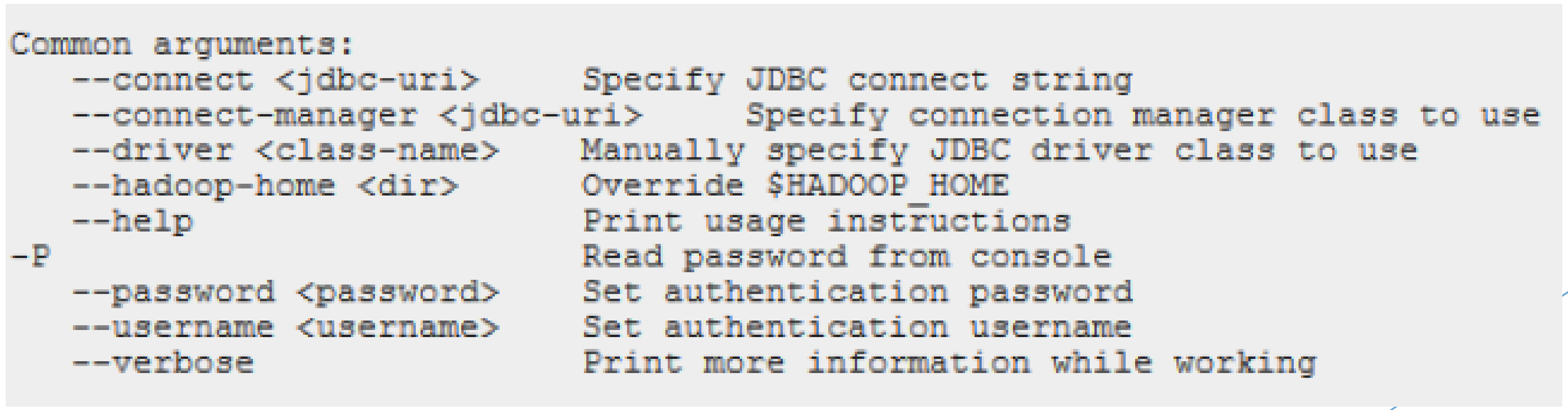
Sqoop Commands

Tool specific arguments start with two dashes (--)



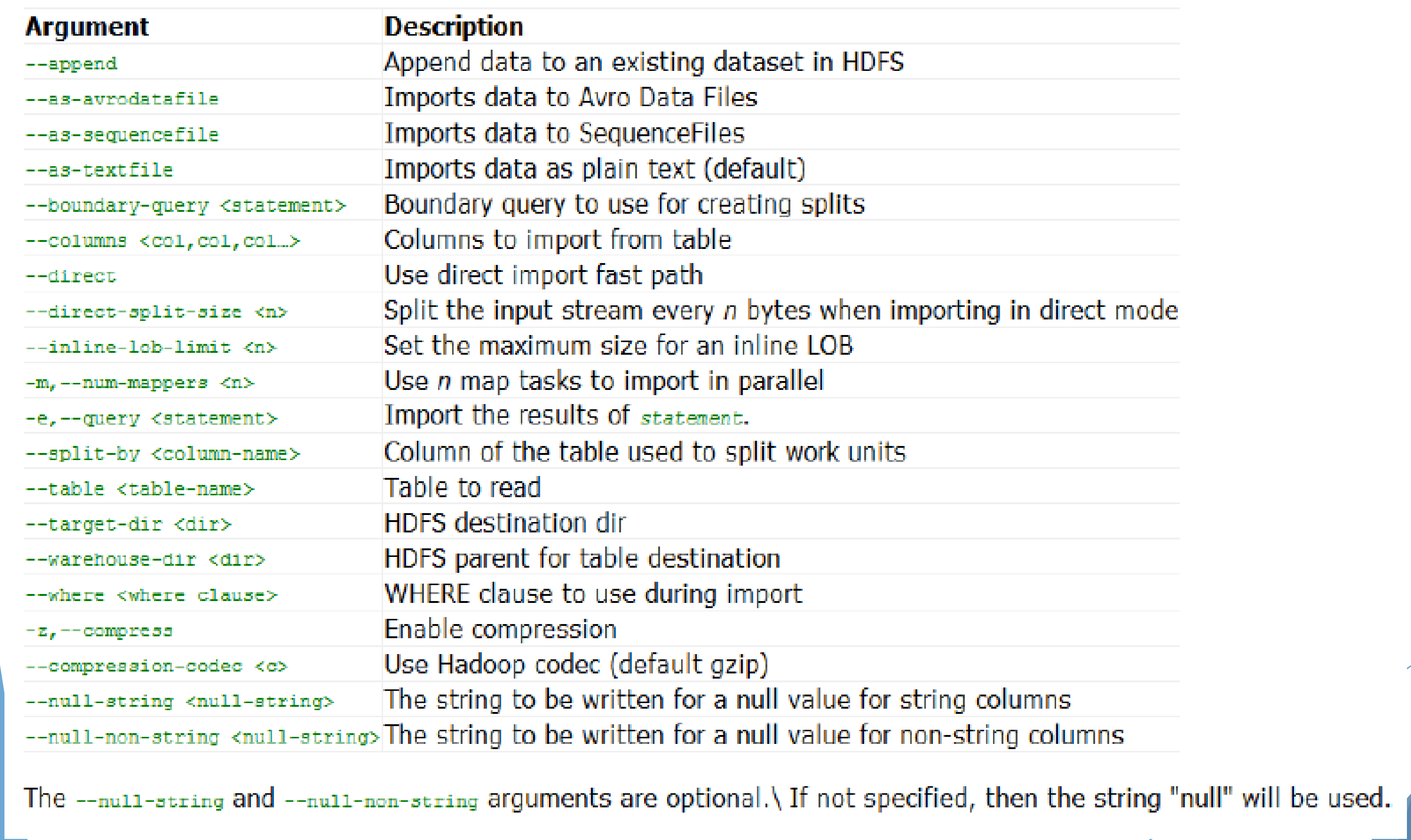
```
Available commands:
  codegen            Generate code to interact with database records
  create-hive-table  Import a table definition into Hive
  eval               Evaluate a SQL statement and display the results
  export             Export an HDFS directory to a database table
  help               List available commands
  import             Import a table from a database to HDFS
  import-all-tables  Import tables from a database to HDFS
  list-databases     List available databases on a server
  list-tables        List available tables in a database
  version            Display version information

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

Hadoop specific arguments are preceded by single dash character (-)



```
Common arguments:
   --connect <jdbc-uri>     Specify JDBC connect string
   --connect-manager <jdbc-uri>     Specify connection manager class to use
   --driver <class-name>    Manually specify JDBC driver class to use
   --hadoop-home <dir>      Override $HADOOP_HOME
   --help                   Print usage instructions
-P                          Read password from console
   --password <password>    Set authentication password
   --username <username>    Set authentication username
   --verbose                Print more information while working
```

Sqoop Import Control Args



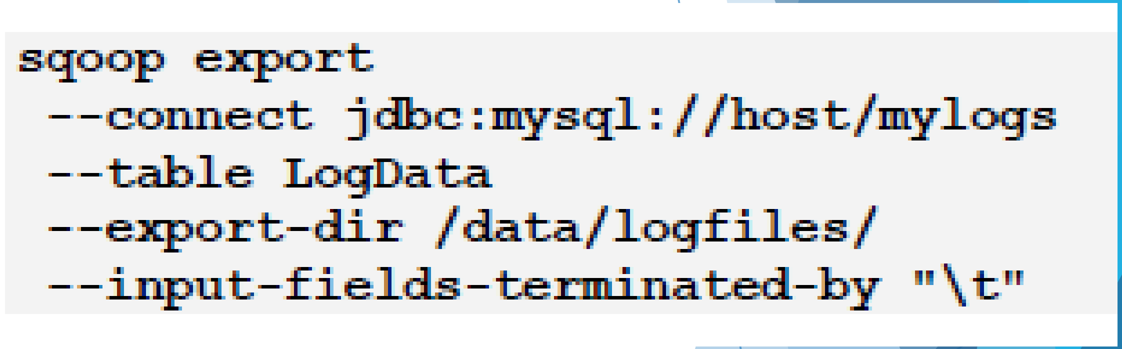
Argument	Description
`--append`	Append data to an existing dataset in HDFS
`--as-avrodatafile`	Imports data to Avro Data Files
`--as-sequencefile`	Imports data to SequenceFiles
`--as-textfile`	Imports data as plain text (default)
`--boundary-query <statement>`	Boundary query to use for creating splits
`--columns <col,col,col…>`	Columns to import from table
`--direct`	Use direct import fast path
`--direct-split-size <n>`	Split the input stream every *n* bytes when importing in direct mode
`--inline-lob-limit <n>`	Set the maximum size for an inline LOB
`-m,--num-mappers <n>`	Use *n* map tasks to import in parallel
`-e,--query <statement>`	Import the results of `statement`.
`--split-by <column-name>`	Column of the table used to split work units
`--table <table-name>`	Table to read
`--target-dir <dir>`	HDFS destination dir
`--warehouse-dir <dir>`	HDFS parent for table destination
`--where <where clause>`	WHERE clause to use during import
`-z,--compress`	Enable compression
`--compression-codec <c>`	Use Hadoop codec (default gzip)
`--null-string <null-string>`	The string to be written for a null value for string columns
`--null-non-string <null-string>`	The string to be written for a null value for non-string columns

The `--null-string` and `--null-non-string` arguments are optional.\ If not specified, then the string "null" will be used.

Sqoop Export

Sqoop Export

- Sqoop's export process will read a set of delimited text files from HDFS in parallel, parse them into records, and insert them as new rows in a target database table.
- Data can be exported from Hadoop to the database on an iterative basis.
- The only requirement is that there not be any constraint violations when performing the INSERT statements
- With insert mode, records exported by Sqoop are appended to the end of the target table
- With Update mode, it works as if exists update else insert
- With Call mode, it makes stored procedure call

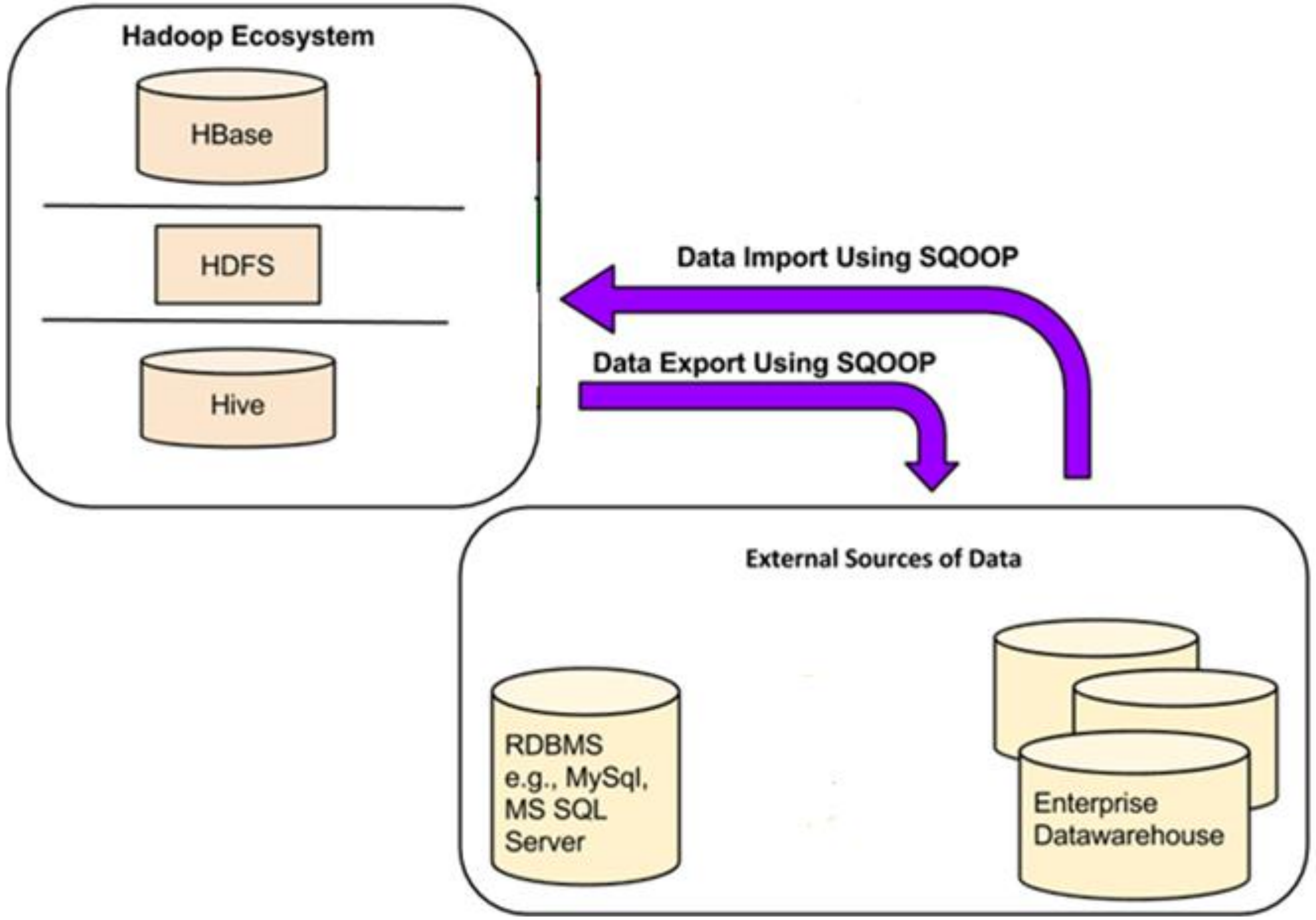


```
sqoop export
 --connect jdbc:mysql://host/mylogs
 --table LogData
 --export-dir /data/logfiles/
 --input-fields-terminated-by "\t"
```

Integration with Ecosystem

Integration with Ecosystem

- Sqoop can be integrated with the rest of the Hadoop Ecosystem

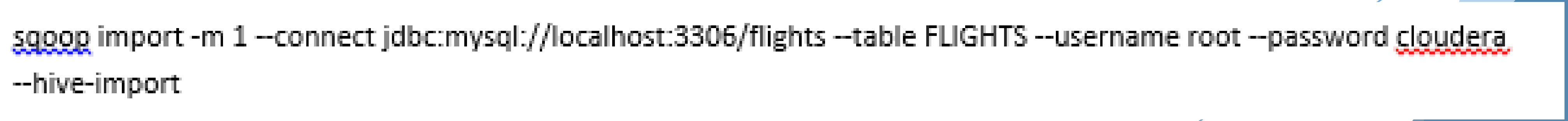


Sqoop with Hive

- Sqoop can import your data directly into Hive.
- Add the parameter --hive-import to your command to enable it

- Example

```
$ sqoop create-hive-table
   --connect jdbc:mysql://localhost:3306/flights
   --table Flights
   --username root
   --password cloudera
```



```
sqoop import -m 1 --connect jdbc:mysql://localhost:3306/flights --table FLIGHTS --username root --password cloudera
--hive-import
```

Sqoop with HBase

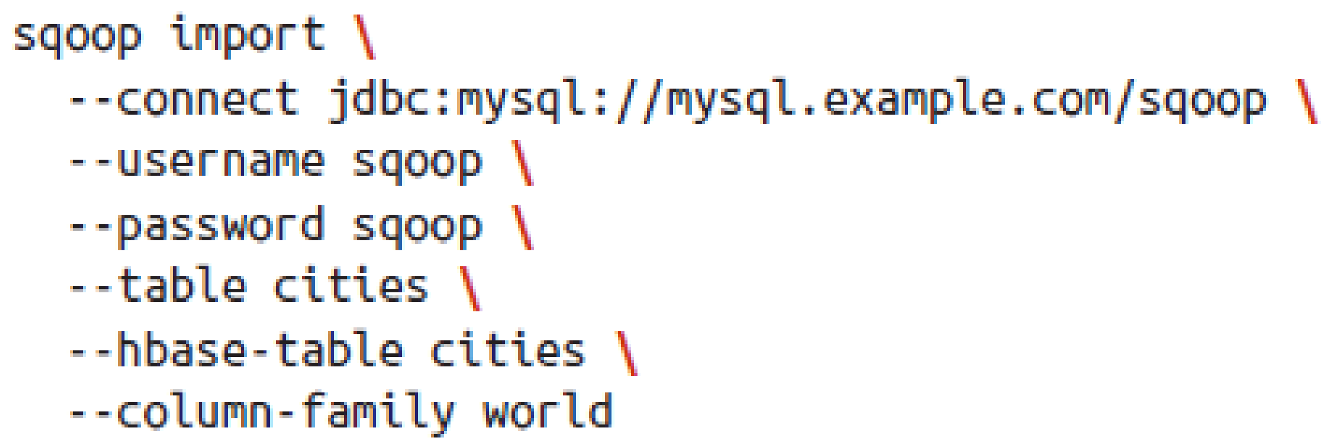
To enable import into HBase, there are two additional parameters:

- **--hbase-table**

Specifies the name of the table in HBase to which you want to import your data.

- **--column-family**

Specifies into which column family Sqoop will import your table's data.



```
sqoop import \
  --connect jdbc:mysql://mysql.example.com/sqoop \
  --username sqoop \
  --password sqoop \
  --table cities \
  --hbase-table cities \
  --column-family world
```

Sqoop with Oozie

- Sqoop jobs can be scheduled in Oozie
- Oozie includes special Sqoop actions that you can use to call Sqoop in your workflow.

```
<workflow-app name="sqoop-workflow" xmlns="uri:oozie:workflow:0.1">
    ...
    <action name="sqoop-action">
        <sqoop xmlns="uri:oozie:sqoop-action:0.2">
```

- <command> to list all the parameters,

```
<command>import --table cities --username sqoop --password sqoop ...</command>
```

Hands On

Sqoop Hands On

1. Import all columns, filter rows using where clause
2. Use Split by option the query
3. Write a sqoop command to perform Incremental imports
4. Use eval function to display the sqoop import result on the console

Thank You

Keerthiga Barathan