

Sqoop for Data Ingestion

Transferring data between relational databases and Hadoop

Dr Amin Karami

a.karami@uel.ac.uk

www.aminkarami.com



CN7022 – Week 4
25 October 2019

Outline

- What is Apache Sqoop?
- Sqoop MetaStore and Sqoop Merge
- How to use Sqoop for transferring data from relational databases to HDFS
- Practice Sqoop commands



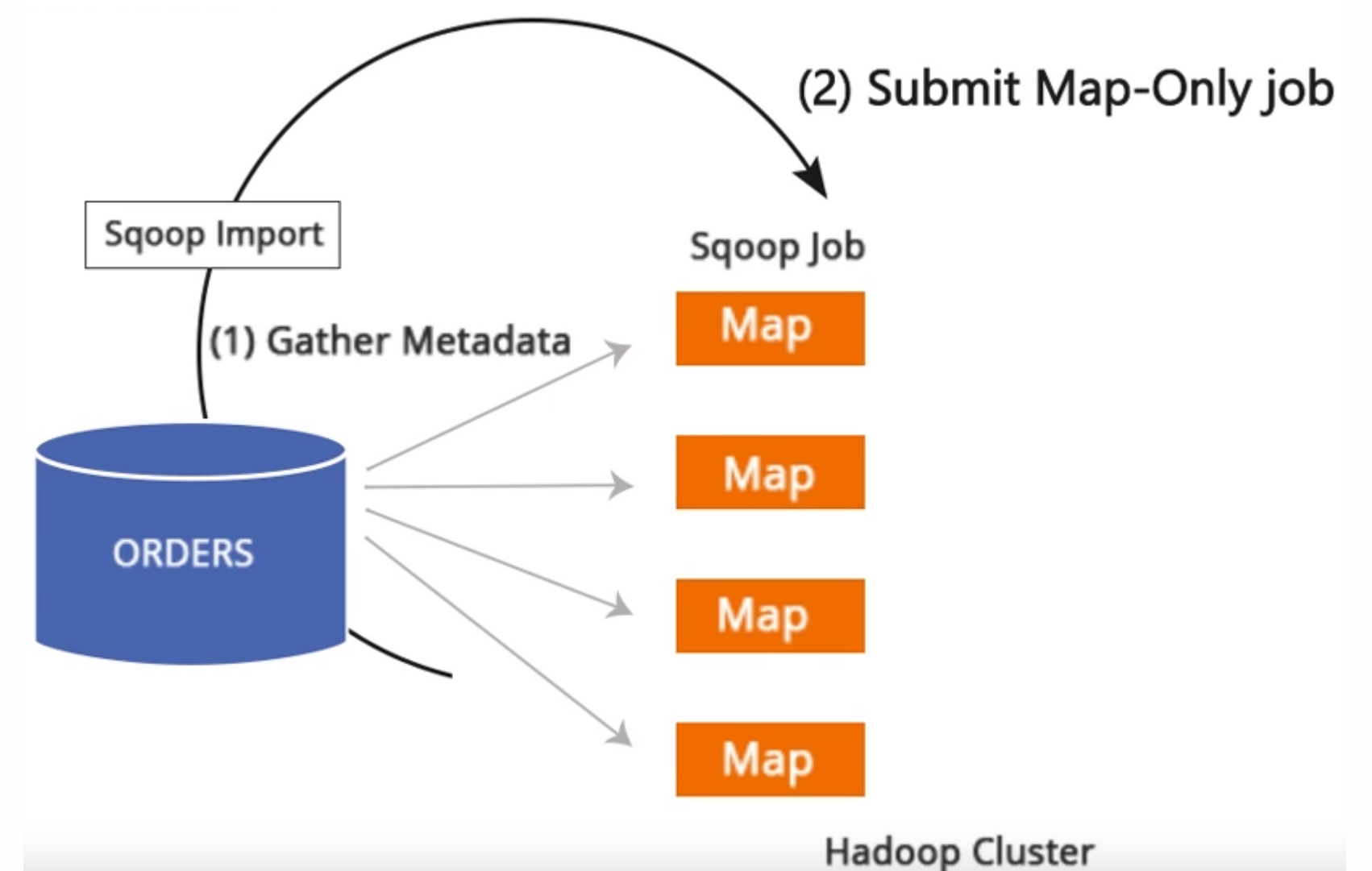
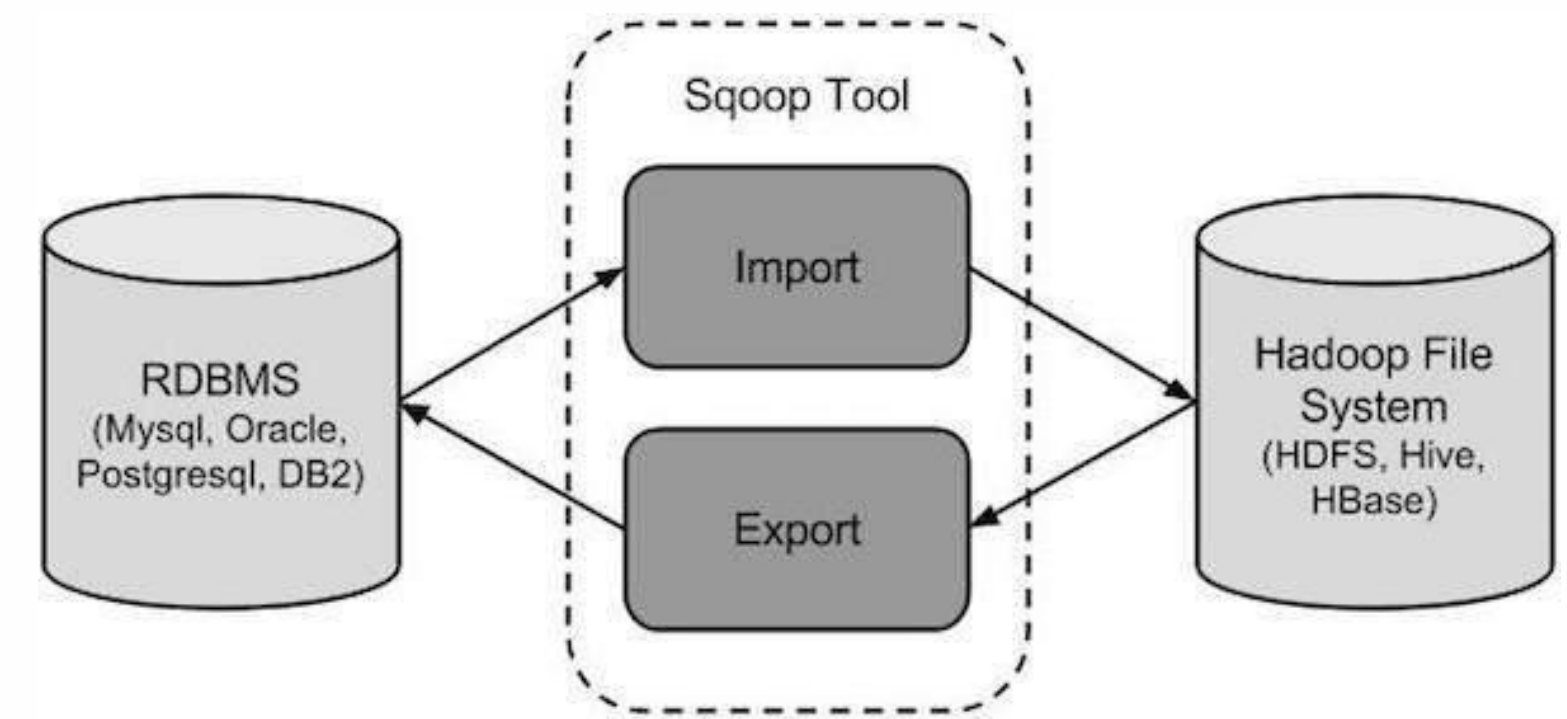
Learning Outcomes

- To be able to explain Sqoop
- Understand Sqoop's engine
- Get familiar with advanced Sqoop components: MetaStore and Merge
- To be able to write Sqoop commands



What is Sqoop?

- SQOOP (SQL-to-Hadoop) is a **MapReduce** based tool designed to transfer data between Hadoop and relational databases.
- Sqoop is written in Java and Open Source
- Map (only), Command line (only)
- Not Secure
- No Client-Server



Sqoop user guide:

<https://sqoop.apache.org/docs/1.4.6/SqoopUserGuide.html>

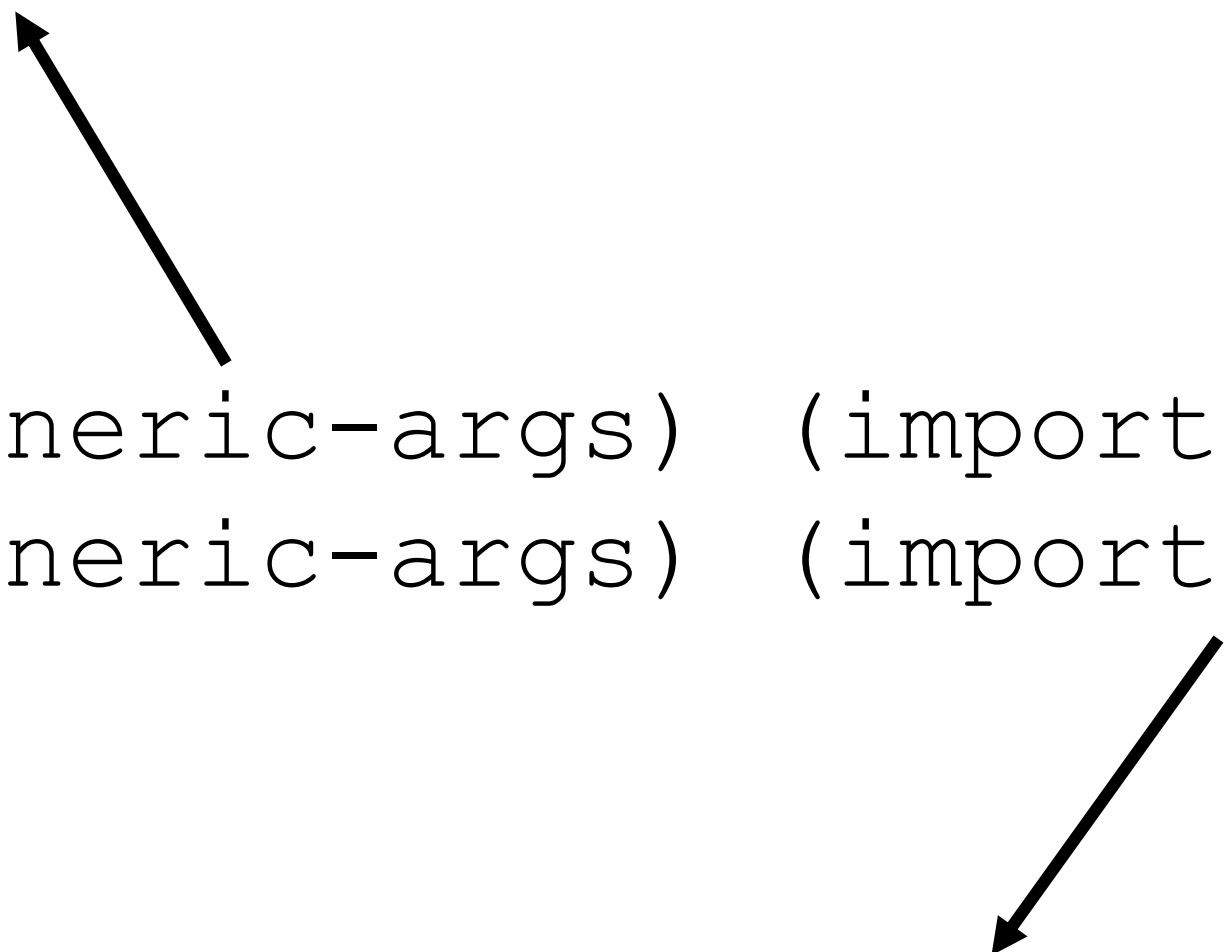


University of
East London

Sqoop Syntax

- With the generic arguments, you point to your database and provide the necessary **login information**.

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



- In the import arguments, you have the ability to specify **what you want to import** and **how** you want the import to be performed.

The Common Arguments

Argument	Description
<code>--connect <jdbc-uri></code>	Specify JDBC connect string
<code>--connection-manager <class-name></code>	Specify connection manager class to use
<code>--driver <class-name></code>	Manually specify JDBC driver class to use
<code>--hadoop-home <dir></code>	Override \$HADOOP_HOME
<code>--help</code>	Print usage instructions
<code>-P</code>	Read password from console
<code>--password <password></code>	Set authentication password
<code>--username <username></code>	Set authentication username
<code>--verbose</code>	Print more information while working
<code>--connection-param-file <filename></code>	Optional properties file that provides connection parameters

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



Example: Sqoop Import

emp:

id	name	deg	salary	dept
1201	gopal	manager	50,000	TP
1202	manisha	Proof reader	50,000	TP
1203	khalil	php dev	30,000	AC
1204	prasanth	php dev	30,000	AC
1204	kranthi	admin	20,000	TP

emp_add:

id	hno	street	city
1201	288A	vgiri	jublee
1202	108I	aoc	sec-bad
1203	144Z	pgutta	hyd
1204	78B	old city	sec-bad
1205	720X	hitec	sec-bad

emp_contact:

id	phno	email
1201	2356742	gopal@tp.com
1202	1661663	manisha@tp.com
1203	8887776	khalil@ac.com
1204	9988774	prasanth@ac.com
1205	1231231	kranthi@tp.com

- Let us take an example of three tables named as *emp*, *emp_add*, and *emp_contact*, which are in a database called *userdb* in a MySQL database.

Example: Sqoop Import

```
sqoop import \  
--table emp -m 2 \  
--where 'city="sec-bad"' \  
--connect jdbc:mysql://<IP Address>/userdb \  
--username=root \  
--password=cloudera \  
--compression-codec=snappy \  
--as-parquetfile \  
--warehouse-dir=/user/hive/warehouse \  
--hive-import
```

- By default, Sqoop would use 4 map tasks.

- If you have a *Hive* metastore associated with your HDFS cluster, you can add `--hive-import` at the end.

- Parquet is an open source file format for Hadoop (Hive, Hbase, MapReduce, Pig, Spark). Parquet stores nested data structures in a flat columnar format. Compared to a traditional approach where data is stored in row-oriented approach, parquet is more efficient in terms of storage and performance.

Import All table into HDFS using Sqoop

- Import all the tables from the RDBMS database server to the HDFS. Each table data is stored in a separate directory and the directory name is same as the table name.

```
$ sqoop import-all-tables (generic-args) (import-args)
$ sqoop-import-all-tables (generic-args) (import-args)
```

```
sqoop import-all-tables \
-m 2 \
--connect jdbc:mysql://quickstart.cloudera:3306/userdb \
--username=root \
--password=cloudera \
--compression-codec=snappy \
--as-parquetfile \
--warehouse-dir=/user/hive/warehouse \
--hive-import
```

Incremental Import

- Incremental import is a technique that imports only the newly added rows in a table. It is required to add 'incremental', 'check-column', and 'last-value' options to perform the incremental import.

```
--incremental <mode>  
--check-column <column name>  
--last value <last check column value>
```

Example: Let us assume the newly added data into *emp* table is as follows:

```
1206, Mike T., admin, 20000, GR
```

Incremental Import

- The incremental import command would be:

```
$ sqoop import \  
--connect jdbc:mysql://localhost/userdb \  
--username root \  
--table emp \  
--m 1 \  
--incremental append \  
--check-column id \  
--last value 1205
```



Verify Import Data in HDFS

- The following command is used to verify the imported data from *emp* table to HDFS *emp/* directory.

```
$ hdfs dfs -cat /emp/part-m-*
```

- The following command is used to see the modified or newly added rows from the *emp* table.

```
$ hdfs dfs -cat /emp/part-m-*1
```

- Or, easily see all the imported data in DataNode using web UI.

Sqoop Export

```
$ sqoop export (generic-args) (export-args)
$ sqoop-export (generic-args) (export-args)
```

- The target table must exist in the target database.
- The default operation is to insert all the record from the input files to the database table using the **INSERT** statement.
- In the update mode, Sqoop generates the **UPDATE** statement that replaces the existing record into the database.
- It only understands HDFS directories not Hive or Hcatalog.

```
sqoop export \  
--table employee \  
--connect jdbc:mysql://quickstart.cloudera:3306/userdb \  
--username=root \  
--password=cloudera \  
--export-dir /emp/emp_data
```



Sqoop MetaStore tool

- A *Sqoop metastore* is used to store Sqoop job information in a central place. This helps the **collaboration between Sqoop users and developers**.
- It is a shared metadata repository, in which remote users and/or multiple users can execute and define saved jobs in the metastore.
- For example, a user can create a job to load some specific data. Then any other user can access from any node in the cluster the same job and just run it again. This is very convenient when using Sqoop in Oozie workflows.

```
$ sqoop metastore (generic-args) (metastore-args)  
$ sqoop-metastore (generic-args) (metastore-args)
```


How to Execute Sqoop MetaStore?

- Running `sqoop-metastore` launches a shared HSQLDB database instance on the current machine. Clients can connect to this metastore and create jobs which can be shared between users for execution.
- The location of the metastore's files on disk is controlled by the `sqoop.metastore.server.location` property in `conf/sqoop-site.xml`. This should point to a directory on the local filesystem.
- The metastore is available over TCP/IP. The port is controlled by the `sqoop.metastore.server.port` configuration parameter, and defaults to 16000.
- Clients should connect to the metastore by specifying `sqoop.metastore.client.autoconnect.url` or `--meta-connect` with the value `jdbc:hsqldb:hsqldb://<server-name>:<port>/sqoop`.
- This metastore may be hosted on a machine within the Hadoop cluster, or elsewhere on the network.

Sqoop Merge

- *Sqoop merge* tool enables you to combine two data sets (flatten two data sets into one), whereby entries in one data set overwrite entries in an older data set. It is a good tool for incremental import, where we have multiple mappers (blocks) in HDFS.

```
$ sqoop merge (generic-args) (merge-args)
$ sqoop-merge (generic-args) (merge-args)
```

Argument	Description
<code>--class-name <class></code>	Specify the name of the record-specific class to use during the merge job.
<code>--jar-file <file></code>	Specify the name of the jar to load the record class from.
<code>--merge-key <col></code>	Specify the name of a column to use as the merge key.
<code>--new-data <path></code>	Specify the path of the newer dataset.
<code>--onto <path></code>	Specify the path of the older dataset.
<code>--target-dir <path></code>	Specify the target path for the output of the merge job.

Example:

```
$ sqoop merge -new-data newer -onto older -target-dir HDFS path
```

Other useful Sqoop tools

- *sqoop-list-databases*: List databases present on a server.
- *sqoop-list-tables*: List tables present in a database.
- *sqoop-job*: The job tool allows you to create and work with saved jobs. Saved jobs remember the parameters used to specify a job, so they can be re-executed by invoking the job by its handle.
- *sqoop-create-hive-table*: To create a hive warehouse.

Source: <https://sqoop.apache.org/docs/1.4.2/>



University of
East London

Summary

- Introduced Apache Sqoop
- Discussed Sqoop import and export commands
- Practiced Sqoop commands
- Advanced Sqoop components: MetaStore and Merge