# Big Data

Data Ingestion

# Argomenti della lezione

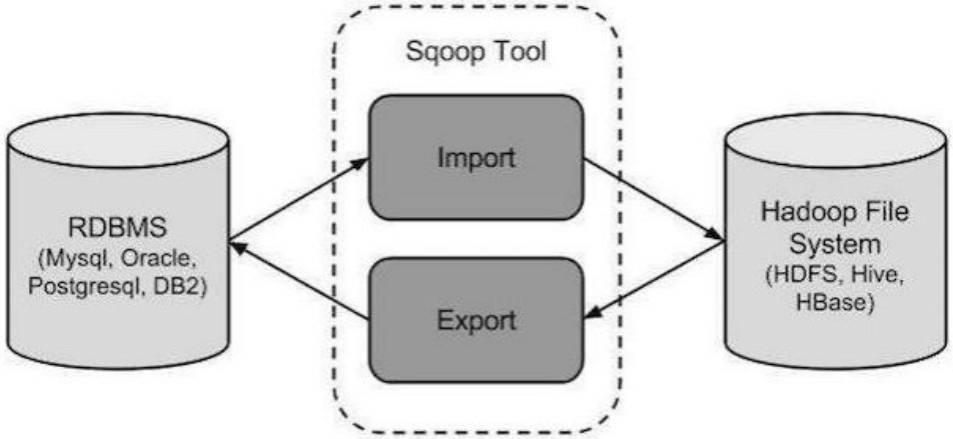
- Cos’è sqoop
- import
- export
- sqoop job

# Cos’è sqoop

Sqoop è un tool, accessibile da riga di comando, pensato per trasferire in maniera efficiente grandi qualità di dati (bulk) tra datastore strutturati, come un database relazionale, e HDFS (import) e viceversa (export) .

La funzionalità di import permette di salvare ogni riga di una tabella come record separato in HDFS. I record possono essere memorizzati come file di testo o in rappresentazione binaria come file Avro o Sequence.

La funzionalità di export consente, a partire da una serie di files, di popolare le tabelle di un database relazionale. I files passati in input, naturalmente, conterranno la rappresentazione dei records da inserire nelle tabelle ed i vari valori saranno delimitati con un delimitatore scelto dall’utente.




Sqoop supporta il caricamento incrementale di una singola tabella o query SQL, o job salvati che possono essere eseguiti più volte per importare aggiornamenti fatti alla base di dati rispetto all'ultimo import.

# Import

Per effettuare l’ingest dei dati su HDFS da RDBMS sqoop mette a disposizione i seguenti comandi:

**$ sqoop import [generic-args] [import-args]**

**$ sqoop-import [generic-args] [import-args]**

Prendiamo, come esempio, il database costituito dalle 4 tabelle Impiegato, dipartimento, progetto e partecipazione di cui è possibile visualizzare i dati nell’immagine seguente

impiegato

| Matricola | Cognome | Stipendio | Dipartimento |
|---|---|---|---|
| 101 | Sili | 60 | NO |
| 102 | Rossi | 40 | NO |
| 103 | Neri | 40 | NO |
| 201 | Neri | 40 | SU |
| 202 | Verdi | 50 | SU |
| 301 | Bisi | 70 | IS |

dipartimento

| Codice | Nome | Sede | Direttore |
|---|---|---|---|
| NO | Nord | Milano | 101 |
| SU | Sud | Napoli | 201 |
| IS | Isole | Palermo | 301 |

progetto

| Sigla | Nome | Bilancio | Responsabile |
|---|---|---|---|
| Alpha | Vendite | 30 | 202 |
| Beta | Inventario | 50 | 301 |
| Gamma | Distribuzione | 18 | 301 |

partecipazione

| Impiegato | Progetto |
|---|---|
| 101 | Alpha |
| 101 | Beta |
| 103 | Alpha |
| 103 | Beta |
| 201 | Beta |
| 202 | Beta |

Per importare i dati da una tabella in HDFS come file di testo o binario dovremmo lanciare il seguente comando:

```
$ sqoop import \
--connect jdbc:mysql://localhost/userdb \
--username root \
--password pass \
--table impiegato --m 1
```

dove, tralasciando i parametri che hanno un significato ovvio, il parametro -m determina il numero di task map paralleli da utilizzare per effettuare l'import.

Come visibile dall'output viene effettivamente lanciato un job MapReduce per effettuare l'operazione di import.

Una volta terminata l'operazione sarà possibile visualizzare i dati importati in HDFS con il comando:

**$ hdfs dfs -cat /impiegato/part-m-***

questo comando mostrerà il contenuto dei file HDFS frutto dell'import. I dati della tabella impiegato mostrati utilizzando una virgola come separatore tra i valori:

**101, Sili, 60, NO**
**102, Rossi, 40, NO**
**103, Neri, 40, NO**

E’ possibile specificare, tramite il parametro --target-dir, la directory di destinazione dell’import.

**$ sqoop import **
**--connect jdbc:mysql://localhost/userdb **
**--username root **
**--username pass **
**--table dipartimento **
**--m 1 **
**--target-dir /result_folder**

Naturalmente in questo caso per visualizzare i valori importati dalla tabella dipartimento sarà necessario eseguire il seguente comando:

**$ hdfs dfs -cat /result_folder/part-m-***

La cartella di destinazione può essere già presente, se non presente sarà creata.

E’ anche possibile effettuare l’import solo dei dati di una tabella che rispettano una determinata condizione aggiungendo l’attributo **--where <condizione>**

```
$ sqoop import \
--connect jdbc:mysql://localhost/userdb \
--username root \
--password pass \
--table impiegato \
--m 1 \
--where “dipartimento =’NO’” \
--target-dir /wherequery
```

Infine, un’altra modalità di import consentita su una tabella è “l’import incrementale”. Questa tecnica ci consente di importare soltanto i nuovi records aggiunti in una tabella. Per effettuare un import incrementale abbiamo bisogno di aggiungere i parametri **incremental, check-column e last-value**

```
$ sqoop import \
--connect jdbc:mysql://localhost/userdb \
--username root \
--password pass \
--table impiegato \
--m 1 \
--incremental append \
--check-column id \
--last value 202
```

in questo caso è possibile visualizzare tutte le informazioni importate da una determinata tabella con:

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

oppure solo gli eventuali records aggiunti con

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

Altre possibilità di import messe a disposizione da sqoop per l'import di dati sono il recupero di tutte le tabelle di un database con un'unica istruzione e la possibilità di utilizzare una query SQL per definire le informazioni da importare.

```
$ sqoop import-all-tables \
--connect jdbc:mysql://localhost/userdb \
--username root
--password pass
```

```
$ sqoop import \
--query 'SELECT a.*, b.* FROM a JOIN b on (a.id == b.id) WHERE $CONDITIONS' \
-m 1 --target-dir /user/foo/joinresults
```

In questo ultimo caso è possibile determinare il numero di map task in base alla metodologia di split con il seguente comando:

```
$ sqoop import \
--query 'SELECT a.*, b.* FROM a JOIN b on (a.id == b.id) WHERE $CONDITIONS'
--split-by a.id --target-dir /user/foo/joinresults
```

# Export

Per effettuare l’export dei dati su RDBMS da HDFS sqoop mette a disposizione i seguenti comandi:

**$ sqoop export [generic-args] [import-args]**

**$ sqoop-export [generic-args] [import-args]**

Prendiamo, come esempio, il database costituito dalle 4 tabelle Impiegato, dipartimento, progetto e partecipazione di cui è possibile visualizzare i dati nell’immagine seguente

impiegato

| Matricola | Cognome | Stipendio | Dipartimento |
|---|---|---|---|
| 101 | Sili | 60 | NO |
| 102 | Rossi | 40 | NO |
| 103 | Neri | 40 | NO |
| 201 | Neri | 40 | SU |
| 202 | Verdi | 50 | SU |
| 301 | Bisi | 70 | IS |

dipartimento

| Codice | Nome | Sede | Direttore |
|---|---|---|---|
| NO | Nord | Milano | 101 |
| SU | Sud | Napoli | 201 |
| IS | Isole | Palermo | 301 |

progetto

| Sigla | Nome | Bilancio | Responsabile |
|---|---|---|---|
| Alpha | Vendite | 30 | 202 |
| Beta | Inventario | 50 | 301 |
| Gamma | Distribuzione | 18 | 301 |

partecipazione

| Impiegato | Progetto |
|---|---|
| 101 | Alpha |
| 101 | Beta |
| 103 | Alpha |
| 103 | Beta |
| 201 | Beta |
| 202 | Beta |

e consideriamo di avere all'interno del path /impiegato/ di HDFS i dati dati dei dipendenti nel seguente formato:

**101, Sili, 60, NO**
**102, Rossi, 40, NO**
**103, Neri, 40, NO**

Per effettuare l’export di questi dati da HDFS verso un RDBMS possiamo lanciare il seguente comando:

```
$ sqoop export \
--connect jdbc:mysql://localhost/db \
--username root \
--password pass \
--table impiegato \
--export-dir /impiegato
```

# Job

I comandi visti fino ad ora possono essere utilizzati per creare dei Job, Un Job, di fatto, è un comando di import o export identificato dal nome del job che può essere rieseguito e/o schedulato. Generlamente i job sono utilizzati per la gestione degli import incrementali.

**$ sqoop job (generic-args) (job-args)**
**[-- [subtool-name] (subtool-args)]**

**$ sqoop-job (generic-args) (job-args)**
**[-- [subtool-name] (subtool-args)]**

Il comando seguente crea un job, denominato job_impiegati, che importa le informazioni della tabella impiegati all'interno di HDFS:

```
$ sqoop job --create job_impiegati \
--import \
--connect jdbc:mysql://localhost/db \
--username root \
--password pass \
--table impiegati --m 1
```

per visualizzare la lista di job presenti:

```
$ sqoop job --list
Available jobs:
  job_impiegati
```

Per eseguire un determinato job viene utilizzato il seguente comando:

**$ sqoop job --exec job_impiegato**

mentre, se volessimo ispezionarne i dettagli potremmo utilizzare il comando:

**$ sqoop job --show myjob**