# Step 1: Installing Sqoop

SQOOP WORKOUTS

Untar the sqoop 1.4.6 tar package and move to the common directory and give the respective permissions.

**cd ~/install**

**tar xvzf sqoop-1.4.6.bin hadoop-2.0.4-alpha.tar.gz**

**sudo mv sqoop-1.4.6.bin hadoop-2.0.4-alpha /usr/local/sqoop**

# Step 2: Configuring Sqoop

To configure Sqoop with Hadoop, you need to edit the sqoop-env.sh file, which is placed in the

$SQOOP\_HOME/conf directory. First of all, Redirect to Sqoop config directory and copy the template file using the following command:

**cd $SQOOP\_HOME/conf**

**mv sqoop-env-template.sh sqoop-env.sh**

## echo 'export HADOOP\_COMMON\_HOME=/usr/local/hadoop' >> $SQOOP\_HOME/conf/sqoop-env.sh echo 'export HADOOP\_MAPRED\_HOME=/usr/local/hadoop' >> $SQOOP\_HOME/conf/sqoop-env.sh

**Step 3: Copy mysql-connector-java**

**cp -p ~/install/mysql-connector-java.jar /usr/local/sqoop/**

**Step 4: Verifying Sqoop**

The following command is used to verify the Sqoop version.

## sqoop-version

**MYSQL (Preparation of Source) : Start the MYSQL Service :**

**sudo service mysqld start mysql -u root -p Password: root**

**Select the custdb database:**

**create database custdb; use custdb;**

**CREATE TABLE customer (custid INT,firstname VARCHAR(20),lastname VARCHAR(20),city varchar(50),age int,createdt date,transactamt int );**

insert into customer values(1,'Arun','Kumar','chennai',33,'2017-09-20',100000); insert into customer values(2,'srini','vasan','chennai',33,'2017-09-21',10000); insert into customer values(3,'vasu','devan','banglore',39,'2017-09-23',90000);

insert into customer values(4,'mohamed','imran','hyderabad',33,'2017-09-24',1000); insert into customer values(5,'arun','basker','chennai',23,'2017-09-20',200000); insert into customer values(6,'ramesh','babu','manglore',39,'2017-09-21',100000); create table customer\_bkp as select \* from customer;

create table customer\_bkp1 as select \* from customer; select \* from customer;

# SQOOP WORKOUTS (Open a separate linux terminal)

## To List Databases which are in MySql

sqoop list-databases --connect jdbc:mysql://localhost --username root --password root;

## To List Tables from custdb database

sqoop list-tables --connect jdbc:mysql://localhost/custdb --username root --password root;

## Import Table from SQL to HDFS with 1 mapper:

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 1 -- delete-target-dir ;

**Check whether the below import works?**

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 2;

sqoop import --connect jdbc:mysql://localhost/custdb --username root -P -table customer -m 3 \

--split-by custid --target-dir sqoop\_import --delete-target-dir --direct;

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 3 \

--split-by city --append --fetch-size 100;

**Fields terminated by & Lines terminated by**

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 1 -- target-dir imp\_del --fields-terminated-by '~' --lines-terminated-by '\n' --delete-target-dir;

**Controlling Import:**

**Using Where condition:**

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root --table customer -m 1 -- where "city ='banglore' or age>33" --target-dir filtered --delete-target-dir;

**Using free form query:**

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root --query " select custid,age,transactamt from customer where (city ='banglore' or age>33) and \$CONDITIONS " --target-dir filtered --delete-target-dir -m 2 --split-by custid;

**Incremental import:**

**Execute the below insert in mysql**

insert into customer values(7,'inceptez','tech','Chennai',3,'2017-09-28',10000);

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 1 -- target-dir incrimport --incremental append --check-column custid --last-value 6

**Whether the below import works?**

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root -table customer -m 1 -- target-dir incrimport --incremental append --check-column city --last-value 'pune'

**Working with Saved Jobs:**

sqoop job --create myjob1 -- import --connect jdbc:mysql://localhost/custdb --username root --password root -- table customer --target-dir savedjob --delete-target-dir -m 1

**list saved jobs:**

sqoop job -list

**sqoop job --exec myjob1**

**Incremental Saved Jobs:**

**sqoop job --create myjob2 -- import --connect jdbc:mysql://localhost/custdb --username root --password root -- table customer --target-dir savedjob1 --m 1 --incremental append --check-column custid --last-value 0**

**sqoop job --exec myjob2 Passwod : root**

**Insert the below row into mysql:**

**insert into customer values (8,'Iz','tech','pune',4,'2017-09-28',10000);**

**Execute once again the same job, now only newly added custid 8 will be imported:**

**sqoop job --exec myjob2 sqoop job --delete myjob2 Update the job:**

**sqoop job --create myjob2 -- import --connect jdbc:mysql://localhost/custdb --username root --password root -- table customer --target-dir savedjob1 --m 1 --incremental append --check-column custid --last-value 6**

**Execute once again the job, now custid from 7 will be imported:**

**sqoop job --exec myjob2**

**Export from HDFS to SQL:**

**Create table in MYSQL before running this command**

CREATE TABLE customer\_hdfs (custid INT,firstname VARCHAR(20),lastname VARCHAR(20),city varchar(50),age int,createdt date,transactamt int );

sqoop export --connect jdbc:mysql://localhost/custdb --username root --password root --table customer\_hdfs -- export-dir savedjob1

Which one of the below export works?

sqoop export --connect jdbc:mysql://localhost/custdb --username root --password root --table customer\_hdfs -- export-dir imp\_del;

sqoop export --connect jdbc:mysql://localhost/custdb --username root --password root --table customer\_hdfs -- export-dir imp\_del --fields-terminated-by '~' --lines-terminated-by '\n';

Incremental export

Update only mode:

delete from customer\_hdfs;

insert into customer\_hdfs select \* from customer;

vi ~/part-m-00000 7~inceptez~tech~Chennai~3~2017-09-28~10000 8~iz~tech~Calcutta~5~2015-09-28~10000 9~a~srini~chennai~38~2016-08-08~13000

**Note:** Delete the last blank line in the above file

hadoop fs -put -f part-m-00000 imp\_del/

sqoop export --connect jdbc:mysql://localhost/custdb --username root --password root --table customer\_hdfs -- export-dir imp\_del --fields-terminated-by '~' --lines-terminated-by '\n' --update-key custid --update-mode updateonly;

select \* from customer\_hdfs ORDER BY 1; Allow insert mode:

ALTER TABLE customer\_hdfs ADD PRIMARY KEY (custid);

vi ~/part-m-00000 8~iz~tech~Mumbai~33~2017-09-28~10000

**10~inceptez~technologies~chennai~33~2017-09-28~13000 Note:** Delete the last blank line in the above file

hadoop fs -put -f part-m-00000 imp\_del/

sqoop export --connect jdbc:mysql://localhost/custdb --username root --password root --table customer\_hdfs -- export-dir imp\_del --fields-terminated-by '~' --lines-terminated-by '\n' --update-key custid --update-mode allowinsert;

**SQOOP Best Practices and Performance tuning**

**Import:**

1. **Definate number of mappers: - m**
2. More mappers can lead to faster jobs, but only up to a saturation point. This varies per table, job parameters, time of day and server availability.
3. Too many mappers will increase the number of parallel sessions on the database, hence affect source DB performance affecting the regular workload of the DB.
4. **Use Direct mode for all available DBs.**
5. Rather than using the JDBC interface for transferring data, the direct mode delegatesthe job of transferring data to the native utilities provided by the database vendor. For Eg. In the case of MySQL, the mysqldump and mysqlimport will be used for retrieving data from the database server or moving data back.
6. Binary format don’t work in direct mode.
7. **Splitting Data --split-by: Boundary Queries --boundary-query**
8. By default, the primary key is used. Prior to starting the transfer, Sqoop will retrieve the min/max values for this column. Changed column with the --split-by parameter
9. Boundary Queries - What if your split-by column is skewed, table is not indexed or can be retrieved from another table?

If --split-by is not giving you the optimal performance you can use this to improve the performance further to Use a boundary query to create the splits using the option --boundary-query

**Eg.**

sqoop import --connect jdbc:mysql://localhost/custdb --username root -P \

--boundary-query "select min(custid), max(custid) from customers"

--query 'Select a.custid master\_custid,a.firstname,a.age,a.city,b.custid detail\_custid,a.createdt,b.fulladdress,category,transactiondt,transactamt from customers a join customer\_details b

on a.custid=b.custid WHERE $CONDITIONS'

--split-by a.custid

Select min(temp.id) , max(temp.id) from (Select a.custid master\_custid,a.firstname,a.age,a.city,b.custid detail\_custid,a.createdt,b.fulladdress,category,transactiondt,transactamt

from customers a join customer\_details b on a.custid=b.custid) as temp

-- boundary-query

--boundary-query "select min(custid), max(custid) from customers"

$CONDITIONS

For Eg. Above query will execute parallel like this.

SELECT a.id, a.name, b.id, b.name FROM customers a join customer\_details b on a.id = b.id where a.id BETWEEN 0 AND 10;

SELECT a.id, a.name, b.id, b.name FROM customers a join customer\_details b on a.id = b.id where a.id BETWEEN 11 AND 20;

SELECT a.id, a.name, b.id, b.name FROM customers a join customer\_details b on a.id = b.id where a.id BETWEEN 21 AND 30;

1. **Compression**

Compress or z When you configure the compress or z argument, you can compress the data approximately by 60% and reduce the amount of disk space required in the target. You can configure compression when the target storage is limited.

1. **fetch-size**

Specifies the number of entries that Sqoop can import at a time. Use the following syntax: --fetch-size 100

Where 100 represents the number of entries that Sqoop must fetch at a time. Default is 1000.

**Export:**

1. **Defining mappers --num-mapper**

a. Number of simultaneous connections that will be opened against database. Sqoop will use that many processes to export data (each process will export slice of the data). Here you have to take care about the max open connections to your RDBMS, since this can overwhelm the RDBMS easily.

1. **BATCH mode --batch**
   1. Sqoop performs export row by row if we don’t leverage batch mode option.
   2. Enabling batch mode will export more than one row at a time as batch of rows.
2. **Specify the number of records to export -Dsqoop.export.records.per.statement=10**
   1. The above option will define how many number of rows should be used in each insert statements.

**e.g. INSERT INTO xxx VALUES (), (), (), ...**

1. **Specify the number of records per transaction - -Dsqoop.export.statements.per.transaction=10**

The above option will define how much number of statements should be used in each transaction.

**e.g BEGIN; INSERT, INSERT, COMMIT**

1. **Data Consistency --staging-table**

a. In order to provide the consistent data access for the users in end database, using a staging table, Sqoop will first export all data into this staging table instead of the main table that is present in the parameter --table. Sqoop opens a new transaction to move data from the staging table to the final destination, if and only if all parallel tasks successfully transfer data.

## Sqoop Additional commands and Use Cases

**Common Usecase 1:**

**Import All tables from a DB :**

sqoop import-all-tables --connect jdbc:mysql://localhost/custdb --username root --password root --warehouse- dir '/user/hduser/sqoop/testtables' -m 1

## Import All tables other than excluded tables from a DB :

sqoop import-all-tables --connect jdbc:mysql://localhost/custdb --username root --password root --warehouse- dir '/user/hduser/sqoop/testtables' --exclude-tables customer\_bkp1 -m 1

## Sqoop evaluation:

sqoop eval --connect jdbc:mysql://localhost/custdb --username root --password root --query "select \* from customer "

## Batch export:

sqoop export -Dsqoop.export.statements.per.transaction=10 --connect jdbc:mysql://localhost/custdb -- username root --password root --table customer1 --export-dir savedjob1 --batch

## Usecase 2 :

**Create the below Tables MYSQL for Import:**

use custdb;

CREATE TABLE customers (custid INT,firstname VARCHAR(20),lastname VARCHAR(20),city varchar(50),age int,createdt date );

CREATE TABLE customer\_details (custid INT,firstname VARCHAR(20),fulladdress VARCHAR(200),category varchar(50),transactiondt date,transactamt int,createdt date);

ALTER TABLE customers ADD PRIMARY KEY(custid);

insert into customers values(1,'karthik','vijay','chennai',5,'2018-02-01'); insert into customers values(2,'arun','kumar','chennai',25,'2018-01-30'); insert into customers values(3,'vishwa','ajit','hyderabad',null,'2018-02-03'); insert into customers values(4,'bala','palani','bangalore',30,'2018-02-02');

insert into customer\_details values(1,'karthik','3/2, jeyaram street, chrompet,Chennai','household','2018-02- 01',4000,'2018-02-01');

insert into customer\_details values(1,'karthik','3/2, jeyaram street, chrompet,Chennai 44','Automobile','2018- 02-02',6000,'2018-02-02');

insert into customer\_details values(1,'karthik','3/2, jeyaram street, chrompet,Chennai 44','Foods','2018-02- 02',3000,'2018-02-02');

insert into customer\_details values(1,'karthik','3/2, jeyaram street, chrompet,Chennai 44',null,'2018-02- 02',1000,'2018-02-03');

insert into customer\_details values(2,'arun','11, palayam blvd, broadway,Chennai 01','tools','2018-02- 02',11000,'2018-02-03');

insert into customer\_details values(2,'arun','11, palayam blvd, broadway,Chennai 01','electronics','2018-02- 02',15000,'2018-02-04');

insert into customer\_details values(3,'vishwa','1A, Elango nagar, Vadapalani,Chennai 33','clothes','2018-02- 02',15000,'2018-02-04');

Select a.custid master\_custid,a.firstname,b.custid detail\_custid,a.createdt,a.age,category,transactamt from customers a join customer\_details b

on a.custid=b.custid;

## Tables for Export:

CREATE TABLE customer\_stage (custid INT,fullname VARCHAR(40), city varchar(50),age int,createdt date ); CREATE TABLE customer\_exp (custid INT,fullname VARCHAR(40),city varchar(50),age int,createdt date ); ALTER TABLE customer\_exp ADD PRIMARY KEY(custid);

## Import Approach:

1. Import all columns of customer and customer\_details data by joining custid between the 2 tables. customer\_details can have columns as given.
2. Both custid should be named as master\_custid and detail\_custid from customer and customer\_details respectively.
3. Use column boundary queries using customer.custid column, split using custid
4. Insert null values in category column and age columns import as NA and age a 0 respectively in the hdfs.
5. Store the output in cust\_details hdfs directory.
6. Compress the imported data.
7. Use direct mode to transfer the entire content.
8. Define number of mappers as 3.
9. Use fetch size 100.
10. Once the sqoop is executed view the content in hdfs using the command: hadoop fs -text cust\_details/part-m-0000\*.gz

## Import Solution Command:

sqoop import --connect jdbc:mysql://localhost/custdb --username root -P --boundary-query "select min(custid), max(custid) from customers" --query 'Select a.custid master\_custid,a.firstname,a.age,a.city,b.custid detail\_custid,a.createdt,b.fulladdress,category,transactiondt,transactamt from customers a join customer\_details b on a.custid=b.custid WHERE $CONDITIONS' --split-by a.custid --target-dir cust\_details --null- non-string '0' --null-string 'NA' --compress --direct --num-mappers 3 --fetch-size 100 --delete-target-dir;

## Export Approach:

1. Import only the subset of columns from the customer table to the HDFS (custid, concatenation of firstname and lastname,age).
2. Export only subset of the above imported columns to the customer\_exp table specifying only these 3 columns.
3. Use batch mode for fast export with the sqoop.export.records.per.statement=5.
4. Use staging table to provide consistent load with the clear staging table option to clean the table before each load.

## Export Solution Command:

sqoop import --connect jdbc:mysql://localhost/custdb --username root --password root --query " select custid,concat(firstname,' ',lastname),age from customers where \$CONDITIONS " --target-dir cust\_exp -- delete-target-dir -m 1;

sqoop export -Dsqoop.export.records.per.statement=5 --connect jdbc:mysql://localhost/custdb --username root

--password root --table customer\_exp --export-dir cust\_exp --batch --staging-table customer\_stage --clear- staging-table --columns custid,fullname,age

## Usecase 3 : Exporting multiple tables using stored procedure

1. **Create file and type data into it**

vi empinfo 101,raja,dept1,Accounts 102,vinay,dept2,Finanace 103,karthik,dept3,IT 104,bala,dept4,Marketing

## Note: Delete the last blank line in the above file

1. **Create folder and copy file into hadoop**

hadoop fs -mkdir -p /user/sqoop/spexport

hadoop fs -copyFromLocal -f empinfo /user/sqoop/spexport

## In MySQL create tables and stored procedure

mysql -u root -p password: root

use custdb;

create table empinfo(empid int, empname varchar(20));

create table deptinfo(deptid varchar(10),deptname varchar(20)); delimiter //

create procedure sp\_insert\_empdeptinfo (IN pid int, IN pname varchar(20), IN pdeptid varchar(10), IN pdeptname varchar(20))

BEGIN

INSERT INTO empinfo(empid, empname) VALUES(pid, pname);

INSERT INTO deptinfo(deptid,deptname) values(pdeptid, pdeptname);

END // delimiter ;

## Export the data from the file to the mysql stored procedure to load data into two tables using SQOOP

sqoop export --connect jdbc:mysql://localhost/custdb --username root --password root --call sp\_insert\_empdeptinfo --export-dir /user/sqoop/spexport -m 1

## In mysql,select both the table empinfo and deptinfo to see the records

select \* from empinfo; select \* from deptinfo;