

**SQOOP WORKOUTS**

**MYSQL (Preparation of Source):**

mysql -u credo;

Select the practicals database:

use practicals;

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,'2015-09-20',100000);

insert into customer values(2,'srini','vasan','chennai',33,'2015-09-21',10000);

insert into customer values(3,'vasu','devan','banglore',39,'2015-09-23',90000);

insert into customer values(4,'mohamed','imran','hyderabad',33,'2015-09-24',1000);

insert into customer values(5,'arun','basker','chennai',23,'2015-09-20',200000);

insert into customer values(6,'ramesh','babu','manglore',39,'2015-09-21',100000);

**SQOOP WORKOUTS (Open a separate Linux terminal)**

* To List Databases which are in MySQL

sqoop list-databases --connect jdbc:mysql://localhost:3306/ --username credo;

* To List Tables from test database

sqoop list-tables --connect jdbc:mysql://localhost:3306/practicals --username credo ;

Import Table from SQL to HDFS:

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo -table customer -m 1 ;

hadoop fs -rm -r /user/credo/customer

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo -table customer -m 1 --direct;

hadoop fs -rm -r /user/credo/customer

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo -table customer -m 1 --target-dir sqoop\_import;

**Check whether the below import works?**

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo -table customer -m 2;

**Import with --split-by option**

hadoop fs -rm -r /user/credo/customer

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo -table customer -m 10 --split-by custid;

hadoop fs -rm -r /user/credo/customer

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo -table customer -m 3 –splitby city;

**Using Where condition:**

hadoop fs -rm -r /user/credo/customer

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer -m 1 --where "city ='chennai'" --target-dir filtered

**Incremental import:**

Execute the below insert in mysql

insert into customer values(7,'md','irfan1','hyderabad',33,'2015-09-28',10000);

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo -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:3306/practicals --username credo -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:3306/practicals --username credo --table customer --target-dir savedjob -m 1

list saved jobs:

sqoop job -list

Execute

sqoop job --exec myjob1

password:credo

**Incremental Saved jobs:**

sqoop job --create myjob2 -- import --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer --target-dir savedjob1 --m 1 --incremental append --check-column custid --last-value 0

sqoop job --exec myjob2

Passwod : credo

insert into customer values (8,'md','irfan1','pune',33,'2015-09-28',10000);

sqoop job --delete myjob2

sqoop job --create myjob2 -- import --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer --target-dir savedjob1 --m 1 --incremental append --check-column custid --last-value 7

sqoop job --exec myjob2

Export from HDFS to SQL: (Create table in MYSQL before running this command)

CREATE TABLE customer1 (custid INT,firstname VARCHAR(20),lastname VARCHAR(20),city varchar(50),age

int,createdt date,transactamt int );

sqoop export --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer1 –exportdir savedjob1

Fields terminated by & Lines terminated by

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo -table customer -m 1 --target-dir imp\_del --fields-terminated-by '~' --lines-terminated-by '\n';

**Which one the below export works?**

sqoop export --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer1 –exportdir imp\_del;

sqoop export --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer1 –exportdir imp\_del --fields-terminated-by '~' --lines-terminated-by '\n';

Incremental export - insert else update

delete from customer1;

insert into customer1 select \* from customer;

cd ~

hadoop fs -get imp\_del/part-m-00000 .

hadoop fs -rm imp\_del/part-m-00000

vi ~/part-m-00000

7~r~yogi~hyderabad~33~2015-09-28~10000

8~r~yogi~Calcutta~33~2015-09-28~10000

9~y~arun~chennai~38~2016-08-08~13000

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

sqoop export --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer1 –exportdir imp\_del --fields-terminated-by '~' --lines-terminated-by '\n' --update-key custid --update-mode updateonly;

select \* from customer1 ORDER BY 1;

ALTER TABLE customer1 ADD PRIMARY KEY (custid);

hadoop fs -rm imp\_del/part-m-00000

vi ~/part-m-00000

8~r~yogi~Mumbai~33~2015-09-28~10000

10~a~naveen~chennai~33~2016-08-08~13000

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

sqoop export --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer1 –exportdir imp\_del --fields-terminated-by '~' --lines-terminated-by '\n' --update-key custid --update-mode allowinsert;

**SQOOP Best Practices and Performance tuning**

**Import:**

1. Definite number of mappers: - m

a. 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.

b. 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.

2. Use Direct mode for all available DBs.

a. Rather than using the JDBC interface for transferring data, the direct mode delegates the job of transferring data to the native utilities provided by the database vendor. For E.g. In the case of MySQL, the mysql dump and mysql import will be used for retrieving data from the database server or moving data back.

b. Escape characters, type mapping, column and row delimiters may not be supported. Binary format doesn’t work.

**3. Splitting Data --split-by: Boundary Queries --boundary-query**

a. 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

b. 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

E.g.

sqoop import \

--connect 'jdbc:mysql://.../...' \

--direct \

--username uname --password pword \

--hive-import \

--hive-table query\_import \

--boundary-query 'SELECT 0, MAX(id) FROM a' \

--query 'SELECT a.id, a.name, b.id, b.name FROM a, b WHERE a.id = b.id AND $CONDITIONS'\

--num-mappers 3

--split-by a.id \

--target-dir /data/import \

- - split-by

Select min(a.id) , max(a.id) from (SELECT a.id, a.name, b.id, b.name FROM a, b WHERE a.id = b.id AND

$CONDITIONS) as temp

-- boundary-query

SELECT 0, MAX(id) FROM a

**4. Using $CONDITIONS**

a. $CONDITIONS is used by Sqoop process, it will replace with a unique condition expression internally to get the data-set. If you run a parallel import, the map tasks will execute your query with different values substituted in for $CONDITIONS.

For Eg. Above query will execute parallel like this.

SELECT a.id, a.name, b.id, b.name FROM a, b WHERE a.id = b.id AND a.id BETWEEN 0 AND 10;

SELECT a.id, a.name, b.id, b.name FROM a, b WHERE a.id = b.id AND a.id BETWEEN 11 AND 20;

SELECT a.id, a.name, b.id, b.name FROM a, b WHERE a.id = b.id AND a.id BETWEEN 21 AND 30;

**Export:**

5. 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.

6. BATCH mode --batch

a. Sqoop performs export row by row if we don’t leverage batch mode option.

b. Enabling batch mode will export more than one row at a time as batch of rows.

7. Specify the number of records to export -Dsqoop.export.records.per.statement=10

a. The above option will define how many number of rows should be used in each insert statements.

8. 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. INSERT INTO xxx VALUES (), (), (), ...

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

9. 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 Use Cases:**

**Usecase 1:**

**Import:**

1. Import all columns of customer and customer\_details data by joining custid between the 2 tables.

customer\_details can have columns as given.

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);

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

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

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

ALTER TABLE customers ADD PRIMARY KEY(custid);

insert into customers values(1,'naveen','raj','chennai',5,'2016-02-01');

insert into customers values(2,'arun','yakambaram','chennai',25,'2016-01-30');

insert into customers values(3,'augustin','daniel','hyderabad',25,'2016-02-03');

insert into customers values(4,'bala','krishnamurthi','bangalore',35,'2016-02-02');

insert into customer\_details values(1,'naveen','5’, third street, mylapore,Chennai','household','2016-02-01',4000,'2016-02-01');

insert into customer\_details values(1,''naveen','5’, third street, mylapore,Chennai 44','Automobile','2016-02-02',6000,'2016-02-02');

insert into customer\_details values(1,'karthik','5’, third street, mylapore, Chennai 44','Foods','2016-02-02',3000,'2016-02-02');

insert into customer\_details values(1,'naveen',’5’, third street, mylapore, Chennai 44',null,'2016-02-02',1000,'2016-02-03');

insert into customer\_details values(2,'arun','17, fourth street, triplicane, Chennai 01','tools','2016-02-02',11000,'2016-02-03');

insert into customer\_details values(2,'arun','17, fourth street, triplicane, Chennai 01','electronics','2016-02-02',15000,'2016-02-04');

insert into customer\_details values(3,'augustin','8, thiruvalluvar nagar, adyar, Chennai 33','clothes','2016-02-02',15000,'2016-02-04');

update customer\_details set category=null where category='ornaments';

Select a.custid master\_custid,a.firstname,b.custid detail\_custid,a.createdt

from customers a join customer\_details b

on a.custid=b.custid;

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.

4. Insert null values in category column and import as NA 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.

**Export:**

1. Export the content of the HDFS into customer\_exp table that has same data structure as customer table.

2. Use batch mode for fast export.

3. Use staging table.

4. Export only subset of columns (for eg. create a file with only 2 columns such as custid and firstname and export

to the customer\_exp table specifying only these 2 columns).

**Script:**

**Import:**

sqoop import --connect jdbc:mysql://localhost:3306/practicals --username credo -P --boundary-query "select

min(custid), max(custid) from customers" --query 'Select a.custid master\_custid,a.firstname,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 '999' --null-string 'NA' --compress --direct --num-mappers 3;

**Export:**

sqoop export --connect jdbc:mysql://localhost:3306/practicals --username credo –table customer\_exp --export-dir cust\_exp --batch --staging-table customer\_stage --columns custid,firstname

**Usecase 2:** Exporting multiple tables using stored procedure

1) Create file and type data into it

vi empinfo

101,yogi,dept1,Accounts

102,arun,dept2,Finanace

103,naveen,dept3,IT

104,bala,dept4,Marketing

2) Create folder and copy file into hadoop

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

hadoop fs -copyFromLocal empinfo /user/sqoop/spexport

3) In MySQL create tables and stored procedure

use test;

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 ;

4) 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:3306/practicals --username credo --call

sp\_insert\_empdeptinfo --export-dir /user/sqoop/spexport -m 1

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

mysql> use test;

mysql> select \* from empinfo;

mysql> select \* from deptinfo;

**More Sqoop Options:**

**Sqoop evaluation:**

sqoop eval --connect jdbc:mysql://localhost:3306/practicals --username credo --query "select \* from customer"

**Batch export:**

sqoop export -Dsqoop.export.statements.per.transaction=10 --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer1 --export-dir savedjob1 –batch

**Codegen:**

sqoop codegen --connect jdbc:mysql://localhost:3306/practicals --username credo --table customer