Moving anyfile from your machine to sandbox local file system(RAM Baand)

scp -P 2222 hive-serde-1.0-sources.jar root@127.0.0.1:

start mysql on the command line cd /usr/local/mysql/bin

after entering bin directory we have ./mysql –u root –p

password : password

load data local infile '/Users/Addytheriot/Desktop/sqldatafile.txt' into table datafile;

Mysql :

1. show databases; To see the database in Mysql. On SandBox.
2. use database name; eg: use hive; # changes database and goes into hive table
3. show tables; Will show all the tables in the database selected.
4. Create database books1; # creates a new database.
5. CREATE TABLE authors ( id INT, name VRACHAR(20),email VRACHAR(20));
6. INSERT INTO authors (id,name,email) VALUES(1,”Aditya”,”[vermaaaditya038@gmail.com](mailto:vermaaaditya038@gmail.com)”);
7. Select \* from authors;
8. DROP TABLE authors; drops the whole table in the MYSQL database.
9. UPDATE authors SET id = 4 WHERE name = “ mannua”; # updates a particular column in the given table.
10. DELETE from table where column name = ` some initialized value.

11 UPDATE Aditya SET name = CASE id WHEN 1 THEN " addy" WHEN 2 THEN "hello" WHEN 3 THEN “theriot”

END

-> WHERE id IN ( 1,2,3);

12 update tablename set columnname1 = value , columnname2 = value , where id = 1;

13 Alter table adi add column id; 🡪this will add the column id to the last column of the existing table , using select id , name , address from adi; 🡪 this will bring the id as the first column as output.

1. Select count(\*) from tablename;
2. Time stamp mysql command: create table Time\_stamp( id INT primary key,name VARCHAR(20),salary VARCHAR(20), revision\_time timestamp default current\_timestamp on update current\_timestamp);

Input will be : insert into time\_stamp values(1,’Aditya’,10000,current\_time) or (1,’Aditya’,10000,’2015-10-09 12:10:09’),(2,’mannu’,9000,’2015-11-09 10:11:09’)

Sandbox sqoop commands: Revised

To connect to Mysql database:

No mapper job needed . statement is correc

Sqoop list-databases --connect jdbc:mysql://127.0.0.1:3306/books2 --username root

Sqoop list-tables --connect jdbc:mysql://127.0.0.1:3306/books2 --username root

To import data from mysql to hdfs :

Sqoop import --connect jdbc:mysql://127.0.0.1:3306/books2--username root --table authors2 –target-dir /mysql/sqoop/ -m 1

To import data from mysql to hdfs if primary key is missing . use split by option of want to give more than one mappers .

Sqoop import --connect jdbc:mysql://127.0.0.1:3306/books2--username root --table authors2

--split-by id –target-dir /mysql/sqoop/ -m 2

when we want the changed row only not the entire table only the modified value after a certain value or id

sqoop import --connect jdbc:mysql://127.0.0.1:3306/books1 --username root --table author12 --split-by id --target-dir /mysqoop/program2/ -m 1 --incremental append --check-column id --last-value 2

This command is used to update the last modified value plus we need to use the append command also so that we can use the same directory and the mapreduce doesn’t show the directory does not exists.

sqoop import --connect jdbc:mysql://127.0.0.1:3306/books1 –-username root –-table tablename –-incremental lastmodified –-check-column revision\_time –-last-value ‘2016-10-11 12:10:16’ –-target-dir /user/timestamp/ --append –m 1

this statement automatically gives the last modified values unlike in the above one where you have to mention

sqoop import --connect jdbc:mysql://127.0.0.1:3306/books1 --username root --table time\_stamp --incremental lastmodified --check-column revision\_time --last-value 0 --target-dir /mysql/timecheck/ --append -m 1

Create a job in sqoop command. SO that can be used for latter reference.(soace between dashes and import is mandatory to create the job)

Sqoop job --create mysqljob -- import --connect jdbc:mysql://127.0.0.1:3306/books1 –-username root –-table authors –target-dir /mysqoop/sqljob/ -m 1

**Too see if the sqoop job is created**

Sqoop job --list

**To see whats in the job .**

Gives the meta data password = try Hadoop if doesn’t work that give the empty string.

Sqoop job –-show mysqljob

Sqoop job –-exec mysqljob

If password press enter for the empty for the root doesn’t work then use this command

use mysql;

select host,user,password from mysql.user;

update mysql.user set password = PASSWORD('123456') where password = '';

flush privileges;

if mysql doesn’t enter

mysql –u root –p

password=123456

sqoop command for combining two or more tables having he same id and name. But the other table has an extra column called salary: In This we are taking id and name from the first column and salary from the other column respective of the id.

create a sqoop job here.(space between import and dash)

Sqoop job –-create mysqoopjob -- import –-connect jdbc:mysql://127.0.0.1:3306://books1 --username root --query

‘select authors.id,authors.name,salary.salary from authors join salary USING(id) where $CONDITIONS’ –-target-dir /mysql/anyname –m 1

sqoop import --connect jdbc:mysql://127.0.0.1:3306/books1 --username root --query 'select Aditya.name,Aditya1.newname from Aditya inner join Aditya1 ON Aditya.ID = Aditya1.ID where $CONDITIONS' --target-dir /mysql/new -m 1

if you want to give a clause or condition

sqoop job –-create newsqljob -- import --connect jdbc:mysql://127.0.0.1:3306/books1 --username root --query 'select \* from try WHERE (id > 2) AND $CONDITIONS' --target-dir /mysql/sqoop2/ -m 1

sqoop job to export data from a table which has rows to an empty table which we just created with no rows and columns.

IN this we export data from hdfs to mysql table which is empty by giving the path of already imported table which is already in hdfs that is hue.

Sqoop export –-connect jdbc:mysql://127.0.0.1:3306/books1 –-username root –-table NameOfTheEmptyTableWhereYouHaveToSendData –-export-dir /mysqoop/program7/part-m-00000 –m 1

Export directory is the directory where our processed data is already stored here /mysqoop/program7/part-m-00000 is the directory where everything is stored and being sent to the empty table.

If the empty table is in other database suppose books2 and the other table is in books1 database so import it to hdfs and then export ot the empty one.

Sqoop export –-connect jdbc:mysql://127.0.0.1:3306/books2 –-username root –-table NameOfTheEmptyTableWhereYouHaveToSendData –-export-dir /mysqoop/program7/part-m-00000 –m 1

If you want to import only some columns of the table

sqoop import --connect jdbc:mysql://127.0.0.1:3306/books1 --username root --table Aditya --columns "name,address" --target-dir /mysql/sqoopcolumn -m 1

create hive tables from sqoop: this will create a table emps with the same schema

sqoop create-hive-table --connect jdbc:mysql://127.0.0.1:3306/books1 --username root --table Aditya --hive-table emps

HIVE :

show databases;

when ever we create a table we need to mention the delimeters

that’s means how are your lines are terminating and how your fleilds are separated.

Make sure you separate them by single tab other format issues will be there.

create table if not exists employee(eid int, name String ,salary float, designation String)

* ROW FORMAT DELIMITED #Tells the format by which
* FIELDS TERMINATED BY ‘\t’ files are seperated
* LINES TERMINATED BY ‘\n’
* STORED AS TEXTFILE;

Show tables;

Now after loading text data into hdfs we have to put in hive.

LOAD DATA INPATH ‘/user/hue/salary.txt’ INTO TABLE employee;

When you load data from local file system into hive

LOAD DATA LOCAL INPATH ‘FILENAME.TXT’ into table tablename;

To see the data in the tables ;

Select \* from employee;

IN order you want to give the path at a particular location instead of apps where generally output of hive is stored by default. Then write this query.

create table if not exists employee(eid int, name String ,salary float, designation String)

* ROW FORMAT DELIMITED #Tells the format by which
* FIELDS TERMINATED BY ‘\t’ files are seperated
* LINES TERMINATED BY ‘\n’
* STORED AS TEXTFILE LOCATION ‘/user/emp/’; #your desired location.

LOAD DATA INPATH’/user/hue/salary.txt’ INTO TABLE employee;

If we want to drop table but not loose the data then we can use EXTERNAL command.

IN this we have to store data into some place other than its default place which is apps in hue. So /user/emp is the new place

Create EXTERNAL table if not exists employee1(id int,name string,salary float,designation string)

* ROW FORMAT DELIMITED
* FIELDS TERMINATED BY ‘\t’
* LINES TERMINATED BY ‘\n’
* STORED AS TEXTFILE LOCATION ‘/user/emp’;

LOAD DATA INPATH’/user/hue/salary.txt’ INTO TABLE employee1

This is also an empty table in which we will load data by giving reference from some other table

Create EXTERNAL table if not exists employee1(id int,name string,salary float,designation string)

* ROW FORMAT DELIMITED
* FIELDS TERMINATED BY ‘\t’
* LINES TERMINATED BY ‘\n’
* STORED AS TEXTFILE LOCATION ‘/user/emp’;

INSERT INTO TABLE employe1

* Select \* from employee(this is table where our data is);

In this we have created a directory mysample in hue

INSERT OVERWRITE DIRECTORY ‘/user/emp/mysample’

* Select \* from sample\_07; # this table already had data which is now being transferred to mysample directory.

Now we have to give intials where we already have an existing table but no numbers are initialized .

We can also copy the format from an existing table into a new one using command:

Create table sample\_10 like sample\_07;

Copies the schema

describe sample\_10; # shows you the schema of the table

the jar used for allocating number or sequence generator for existing table .

To know the version of your Hadoop for calling the jar command.

Hadoop version

add jar /usr/hdp/2.2.4.2-2/hive/lib/hive-contrib.jar;

This is the class where we have our logic to generate row sequence nubmer. That is the sequence generator for already existing table.

CREATE TEMPORARY FUNCTION row\_sequence as 'org.apache.hadoop.hive.contrib.udf.UDFRowSequence';

So now the tables which we create after this we will have an id beacause in the sample table which is already their did not had any id number in the sample. So now create a new table with addition of id in the table

Create table if not exists sample\_11(id int ,code string, description string,total\_emp string,salary string);

insert into table sample\_11 select row\_sequence(),code,description,total\_emp,salary from tablenameWhereTheactuallTableis

now for the partitioning and bucketing we have to make hive dynamic which is not that generally by defaultt

You have to enter this everytime you have to use hive for dynamic partition .

set hive.exec.dynamic.partition.mode = nonstrict;

set hive.exec.dynamic.partition = true;

so after this create a new table for bucketing and partitioning .

create table newbucket(id int , name string , salary int) partitioned by (country string)

* row delimited by
* fields terminated by ‘,’
* lines terminated by’\n’ ;

insert overwrite table newbucket partition (country) select \* from bucket;

if the rows are to be aligned according to a particular column which is not last in the table , wrong output can be generated , in this case use the statement below to get the desired output irrespective of the positon of the column

insert overwrite table exp1 partition(country) select id,name,salary,country from expbucket;

The one column you are partitioning should be always on the end in the select statement irrespective of the column and their order.

1 kiran usa 21000

2 Aditya usa 21005

3 sahil india 29009

4 rohit india 28292

5 kgsdgsg china 21003

6 Adzdsd china 21000

7 sadcd cali 29007

8 reds cali 28299

here the name is in the second column but to call it form partition which just names it in the end in the select statement which calls the partition on name , can be seen in hue

create table newbucket(id int , country string , salary int) partitioned by (name string)

* row delimited by
* fields terminated by ‘,’
* lines terminated by’\n’ ;

insert overwrite table exp2 partition(name) select id,country,salary,name from expbucket;

then use where clause to get the data

select \* from exp1 where country = “usa”;

for bucketing we use **clustered by**  and the column name .

create table bucket\_emp(id int, name string ,salary int,country string)clustered by (salary)into 4 buckets

> row format delimited

> fields terminated by ','

> lines terminated by'\n';

insert overwrite table bucket\_emp select\*from newbucket1;

this gives distinct value on multiple columns in hive.

select distinct yearid,lgid from part where yearid > 1978 and yearid < 2010 order by yearid;

This statement executes both partition and and bucket.

create table partbucket(playerid string,yearid int,gamenum int,gteamid string,gp int,startingpos int)partitioned by ( gameid string,lgid string) clustered by(yearid) into 4 buckets

> row format delimited

> fields terminated by ','

> lines terminated by '\n';

using this statement we get the bucketing which is inside the partitioning .!

insert overwrite table partbucket partition(gameid,lgid) select playerid,yearid,gamenum,teamid,gp,startingpos,gameid,lgid from allstarfull;

remove the header from the

add this on the last line while creating table

* tblproperties (“skip.header.line.count”=”1”);

the orc.compress used at the end will help you to compress very large file into significat small size and from there you can perform fetching operation which will be faster to retrieve from the compressed orc table.

create table trip\_data\_orc(medallion string,hack\_license string,vendor\_id string,rate\_code int,store\_and\_fwd\_flag string,pickup\_datetime timestamp,dropoff\_datetime timestamp,passenger\_count int,trip\_time\_in\_secs int,trip\_distance float,pickup\_longitude float,pickup\_latitude float,dropoff\_longitude float,dropoff\_latitude float) clustered by (pickup\_datetime) into 10 buckets stored as orc tblproperties("orc.compress"="SNAPPY");

hive serde: Mainly used for key and value file. Just like

{“key1”:”value1”,”key2”:100,“key3”:”value3”,”key4”:272.92}

{“key5”:”value5”,”key6”:121,“key7”:”value7”,”key8”:299.982}

{“key9”:”value9”,”key10”:1332,“key11”:”value11”,”key12”:9829.2}

{“key13”:”value13”,”key14”:123,“key14”:”value14”,”key15”:9829}

Add jar for serde to use hive serde.

add jar json-serde-1.3.6-SNAPSHOT-jar-with-dependencies.jar;

HIVE serde is in the other document .

PIG (taking an input data see that it is seprated by tab)

Make a table first

With the tab separate three columns

1950 2 1

1951 -3 2

1950 2 1

1951 -3 2

1950 2 1

1951 -3 2

save this file as a text document.

load this data into hdfs using hue

So this data doesn’t have a schema

1. load it without schema

records = load ‘/user/hue/pig.txt’ using PigStorage(‘\t’);

describe records ; output will be no schema .

1. mention some schema without datatypes and load files

records = load ‘/user/hue/pig.txt’ using PigStorage(‘\t’) AS (year,temp,quality);

describe records;

will show the schema but all data types will be bytearray, that’s default which happened when there is not data type mention along with the column name

1. mention schema with respective data types

records = load ‘/user/hue/pig.txt’ using PigStorage AS ( year:int,temp:int,quality:int);

describe records : shows the datatype along with the column names

dump records; runs The MR job, output generated is a csv tuples .

1.so when there is no schema for the table for projecting the selected values we use

records1 = load ‘/user/hue/pig.txt’ using PigStorage (‘\t’);

using foreach loop for records1

proj\_records = foreach records1 generate $0,$1;

dump proj\_records;

will run MR job and will give the index 0 and 1 record.

2. so now with the schema. Instead of mentioning the index numbers you can actually mention the columns names;

records1 = load ‘/user/hue/pig.txt’ using PigStorage AS ( year:int,temp:int,quality:int);

proj\_records = foreach record1 generate year, temp;

dump proj\_records;

data for all records starting from index 0

proj\_records = foreach record1 generate $0..;

all records from 6th column;

proj\_records = foreach record1 generate $5..;

all the record from 1st column to 8th column

proj\_records = foreach record1 generate ..$7;

fetch data between 5th column to 8th column

proj\_records = foreach record1 generate $4..$7;

records

DUMP

If we need to split data into different tables and dump them separately.

split records into subrecords if column1 < condition, records2 if(column2>somevalue or column2<somevalue), records3 if column3 == somevalue;

dump subrecords

dump records2

dump records3

split records into x if f1<4, y if f2 == 5,z if (f3>6 or f1<4);

dump x

(1,2,3)

dump y

(4,5,6)

dump z

(1,2,3)🡪 for f1 < 4

(7,8,9)🡪 for f2 > 6

now in records if we have null values then we can separate it.

(1960,11,12,3)

(1960,11,12,2)

(1960,11,,2)

(1960,11,12,)

split records into good\_records IF year is not null ,bad\_records IF year is null;

dump good\_records;

dump bad\_records:

good\_records = FILTER records BY num1 is not null ;

dump good\_records ;

now to cut the records into somepart like 30 percent.

records = load ‘/filedirectory/textfile’ using PigStorage(‘,’) AS(fi:int,f2:chararray,f3:chararray);

newrecords = sample records 0.3; sample is a key word and the record is divided into 30 percent of the total length of the record!. the records can be random

bill = foreach record generate $0 AS somenewcolumnName,$1,$2;

this statement will replace the column name at that index even if it is already mentioned .

to add a new column .

finalbill = foreach bill generate $0,$1,$2,$1\*$2 as new\_column\_final\_bill;

describe finalbill : the schema will show you the new column added.

Dump finalbill will show you the multiplied value under the fourth column.

Price = FILTER records by Salary >somevalue; This can be given by looking at your data!

Price = DISTINCT records;🡪 only stores the unique value in price from the table records

Dump price;

Storing Data into HDFS:

STORE price INTO ‘/user/hue/pricefile’ using PigStorage(‘,’); 🡪 in this we have ti mention the name of the directory without making one in hdfs, so here “pricefile” directory will automatically get created in hdfs with the data in stored in the variable price

F 17 41000.0 95103

M 19 76000.0 95102

F 22 95000.0 95103

F 19 60000.0 95105

M 19 14000.0 95102

M 17 35000.0 95103

F 17 41000.0 95103

M 19 76000.0 95102

F 22 95000.0 95103

F 19 60000.0 95105

M 19 14000.0 95102

M 17 35000.0 95103

Salaries = load ‘/user/hue/salaries.txt’ using PigStorage(‘\t’) AS (gender:chararray,age:int,salary:double,zip:int);

filterit = filter salaries by (gender == 'M') and (NOT(salary >90000));

try:

filterit = filter salaries by (gender == 'M') OR (NOT(salary >90000));

dump filterit;

filterit = foreach salaries generate (int)age + 1; 🡪 adds one to each age:

SalariesBygroup = GROUP salaries by age;

Dump salariesBygroup.

Group all the people with same age group irrespective of gender

Mygroup = GROUP salaries by (gender,age);

Dump mygroup;

Group all the people with same age and gender: so all the males with same age group will have their data summed up in on e tupple.

Salariesbyage = Group salaries by age;

Sumofsalaries = Foreach salariesbyage generate group,SUM(salary); This group here is for age as it is mentioned in the above statement that grouping is my age.

Salariesbygender = group salaries by gender;

Sumofsalaries = foreach salariesbygender generate group,sum(salaries.salary);

Dump sumofsalaries;

Sumofsalaries = foreach salariesbygender generate group,MAX(salaries.salary);

Sumofsalaries = foreach salariesbygender generate group,MIN(salaries.salary);

Dump sumofsalaries; This will dump the maximum and minimum salary in a particulatr age group as it is grouped by age;

Salaries = group salaries ALL; gives the output with all age with all the information is tuple form .

Orderofthesalaries = order salaries by age asc;

order salaries by age asc, salary desc; In this among the age in an ascending order the salaries which will be highest in the same age will come first respective to that age. Try taking same ages multiple times with deifferent salaries attached to them.

Using switch case in pig language .

grunt> bonuses = foreach salaries generate salary,(

>> CASE

>> WHEN salary >= 70000 THEN salary\*.10

>> WHEN salary < 70000 AND salary >=35000 THEN salary\*.5

>> WHEN salary == 14000 THEN 0

>> END

>> );

this is to store if the data is in bags and tuple form .

Rich remote {(SD),(CA)}

uff onsite {(CA)}

Tom remote {(OH),(NY)}

barry remote {(NV),(NY)}

places = load ‘/user/hue/locations.txt’ using PigStorage(‘\t’) AS (name:chararray,

* location:chararray,
* states:bag{t:tuple(state:chararray)});

innerjoin = JOIN joindemo BY name ,joindemo1 BY name;

join two tables with same name and gives all the data.

Outerjoin = JOIN joindemo BY name LEFT OUTER , joindemo1 BY name;

Gives all the matched and unmatched data of the left table deosnt matter if they match

(OH,Tom,Tom,marketing)

(CA,ulf,ulf,managment)

(SD,rich,,)

(NV,berry,,)

(co,george,,)

Outerjoin = JOIN joindemo BY name RIGHT OUTER, joindemo1 BY name;

Returns all the data of the right table doesn’t matter if it matches or doesn’t not.

(OH,Tom,Tom,marketing)

(CA,ulf,ulf,managment)

(,,Rich,marketing)

(,,Rich,sales)s

(,,barry,sales)

cogrouping = COGROUP joindemo by name, joindemo1 by name;

(Tom,{(OH,Tom)},{(Tom,marketing)})

(ulf,{(CA,ulf)},{(ulf,managment)})

(Rich,{(SD,Rich)},{(Rich,marketing),(Rich,sales)})

(barry,{},{(barry,sales)})

(berry,{(NV,berry)},{})

(george,{(co,george)},{})

gives all the information related to that name in the table: in this rich name is in two times on the right hand side means rich comes under two departments.

To know the which names falls under how many department we can use count statement .

Counter = FOREACH cogrouping generate group, COUNT(name),COUNT(name);

(Tom,1,1)

(ulf,1,1)

(Rich,1,2)

(barry,0,1)

(berry,1,0)

(george,1,0)

before opeing pig use this

pig –useHCatalog

then use pig grunt shell

A = LOAD 'default.sample\_08' USING org.apache.hive.hcatalog.pig.HCatLoader();

Dump A

Create table allstarlike like allstarfull; this is in hive

Then

STORE A into 'default.allstarlike' USING org.apache.hive.hcatalog.pig.HCatLoader();

Dump default.allstarlike;

Lines = LOAD ‘/wordcount/wordcounts.txt’ using PigStorage() AS (shabd : chararray);

Describe Lines ;

Explain Lines;

Words = FOREACH Lines GENERATE TOKENIZE(shabd)

Output : ({(this),(is),(hello),(hope),(you),(are),(doing),(fine)})

Not we have to flatten them

Words = FOREACH Lines GENERATE FLATTEN(TOKENIZE(shabd)) AS newflat;

Output:

(this)

(is)

(hello)

(hope)

(you)

(are)

(doing)

(fine)

grouped = Group words BY newflat;

dump grouped;

i,{(i)})

(be,{(be)})

(hi,{(hi)})

(on,{(on)})

(and,{(and)})

(are,{(are),(are),(are)})

(how,{(how)})

(you,{(you)})

(care,{(care)})

(fine,{(fine),(fine),(fine)})

(hope,{(hope)})

(keep,{(keep)})

(they,{(they)})

(will,{(will)})

(things,{(things),(things)})

(venkat,{(venkat)})

(don’t,{(don’t)})

(laughing,{(laughing)})

wordcount = FOREACH Grouped GENERATE group,COUNT(words);

dump words;

(i,1)

(be,1)

(hi,1)

(on,1)

(and,1)

(are,3)

(how,1)

(you,1)

(care,1)

(fine,3)

(hope,1)

(keep,1)

(they,1)

(will,1)

(things,2)

(venkat,1)

(don’t,1)

(laughing,1)