HIVE CHALLENGE 3

Scenario Based questions:

Q. Will the reducer work or not if you use “Limit 1” in any HiveQL query?

-Reducer will not work in this HiveQL query

Q. Suppose I have installed Apache Hive on top of my Hadoop cluster using default metastore configuration. Then, what will happen if we have multiple clients trying to access Hive at the same time?

- The default metastore configuration allows only one Hive session to be opened at a time for accessing the metastore. Therefore, if multiple clients try to access the metastore at the same time, they will get an error. One has to use a local or remote metastore configuration in Apache Hive for allowing access to multiple clients concurrently.

Q. Suppose, I create a table that contains details of all the transactions done by the customers: CREATE TABLE transaction\_details (cust\_id INT, amount FLOAT, month STRING, country STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’ ; Now, after inserting 50,000 records in this table, I want to know the total revenue generated for each month. But, Hive is taking too much time in processing this query. How will you solve this problem and list the steps that I will be taking in order to do so?

-In this case, I will use dynamic partition on the month column to optimize the performance of the query

Step 1: Create a table without partition to insert this table’s data into the partitioned table

>CREATE TABLE transaction\_details\_without\_partition (cust\_id INT, amount FLOAT, month STRING, country STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’ ;

Step 2: Load data into table without partitioning

>load data local inpath ‘pathtocsvdata’ into table transaction\_details\_without\_partition

Step 3: set this property for dynamic portioning

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

Step 4: Create a partitioned table

>CREATE TABLE transaction\_details (cust\_id INT, amount FLOAT, country STRING) partitioned by( month ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’ ;

Step 5: Inserting data into partitioned table

>insert override table transaction\_details(month) select cust\_id, amount ,country, month from transaction\_details\_without\_partition;

Q. How can you add a new partition for the month December in the above partitioned table?

>Alter Table transaction\_details Add Partition (month=’dec’);

>insert into table partition(month=’dec’) select \* from transaction\_details where month=’dec’;

Q. I am inserting data into a table based on partitions dynamically. But, I received an error – FAILED ERROR IN SEMANTIC ANALYSIS: Dynamic partition strict mode requires at least one static partition column. How will you remove this error?

This happens because the dynamic partition strict mode is active by default, so to turn it on we need to explicitly turn it off.

set hive.exec.dynamic.partition = true;

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

Q. Suppose, I have a CSV file – ‘sample.csv’ present in ‘/temp’ directory with the following entries:

id first\_name last\_name email gender ip\_address

How will you consume this CSV file into the Hive warehouse using built-in SerDe?

>create table student\_data

( id int, first\_name string, last\_name string, email varchar(30), gender string, ip\_address longint)

row format serde ‘org.apache.hadoop.hive.serde2.OpenCSVSerde’

with serdeproperties ("separatorChar" = ",", "quoteChar" = "`", "escapeChar" = "\\" )

stored as textfile tblproperties (“skip.header.line.count”=’1’);

Q. Suppose, I have a lot of small CSV files present in the input directory in HDFS and I want to create a single Hive table corresponding to these files. The data in these files are in the format: {id, name, e-mail, country}. Now, as we know, Hadoop performance degrades when we use lots of small files.

So, how will you solve this problem where we want to create a single Hive table for lots of small files without degrading the performance of the system?

-CREATE TABLE temp\_smalltable (id INT, name STRING, email STRING, country STRING) nive ROW FORMAT DELIMITED FIELDS TERMINATED BY STORED AS TEXTFILE;

-hdfs dfs-copyFromLocal small\*.csv /user/bigdata/filesdata;

-hadoop fs-Is /user/bigdata/filesdata;

- LOAD DATA INPATH '/user/bigdata/filesdata/small\*.csv' INTO TABLE temp\_smalltable;

- select from temp\_smalltable;

- CREATE TABLE small\_seqfile (id INT, name STRING, email STRING, country STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY", STORED AS SEQUENCEFILE;

- INSERT INTO small seqfile SELECT \* FROM temp\_smalltable;

Q. LOAD DATA LOCAL INPATH ‘Home/country/state/’ OVERWRITE INTO TABLE address

The following statement failed to execute. What can be the cause?

* There might be one of the two reason
* 1. The load data local inpath ‘’ might not work because it might have been executed previously and the file is not available in the path. This happens because when we execute this command for a internal table the csv file is moved from the base location into hive hdfs warehouse location
* 2. The file u are using to load data does not have right permission’s of access

Q. Is it possible to add 100 nodes when we already have 100 nodes in Hive? If yes, how?

Yes, we can add the nodes by following the below steps:

Step 1: Take a new system create a new username and password

Step 2: Install SSH and with the master node setup SSH connections

Step 3: Add ssh public\_rsa id key to the authorized keys file

Step 4: Add the new DataNode hostname, IP address, and other details in /etc/hosts slaves file:

192.168.1.102 slave3.in slave3

Step 5: Start the DataNode on a new node

Step 6: Login to the new node

Step 7: Start HDFS of the newly added slave node by using the following command:

./bin/hadoop-daemon.sh start data node

Step 8: Check the output of the jps command on the new node

Hive Practical questions:

Hive Join operations

Create a table named CUSTOMERS(ID | NAME | AGE | ADDRESS | SALARY)

Create a Second table ORDER(OID | DATE | CUSTOMER\_ID | AMOUNT

)

Now perform different joins operations on top of these tables

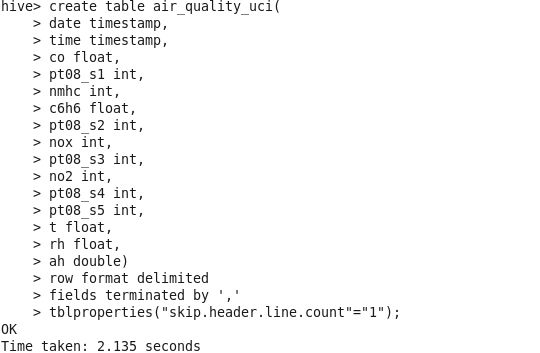
(Inner JOIN, LEFT OUTER JOIN ,RIGHT OUTER JOIN ,FULL OUTER JOIN)

BUILD A DATA PIPELINE WITH HIVE

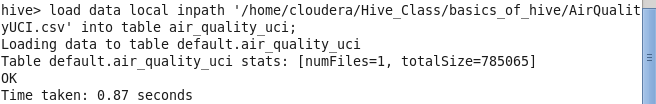
Download a data from the given location -

https://archive.ics.uci.edu/ml/machine-learning-databases/00360/

1. Create a hive table as per given schema in your dataset



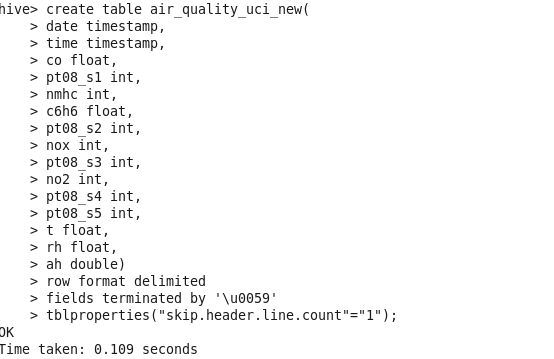
2. try to place a data into table location



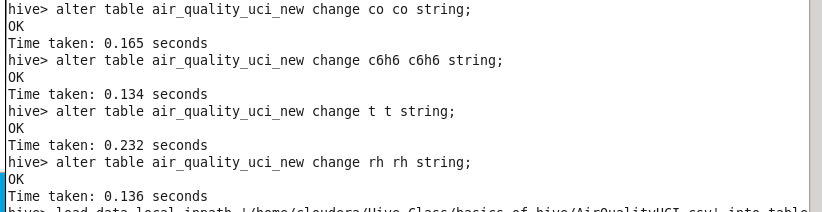
This approach does’nt work as the filed in the file are terminated by ; and the decimal place are separated by ,

So, we need transformation on this table

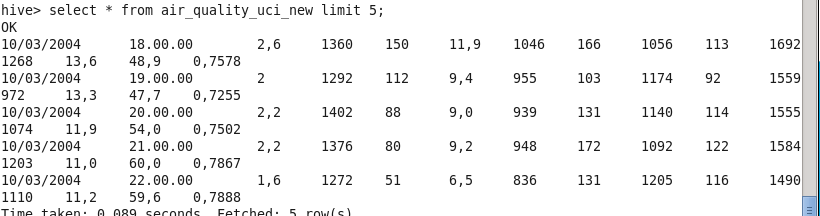
Step 1: take datatime as string in the table and decimal values as string in the table



Using alter table command to change the datatype of the columns

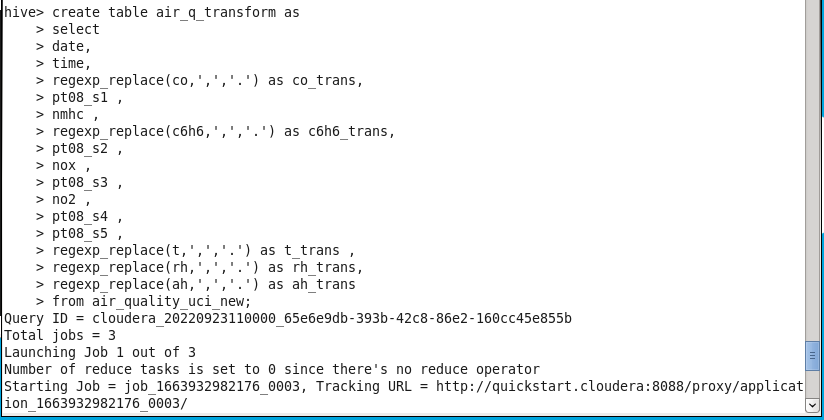


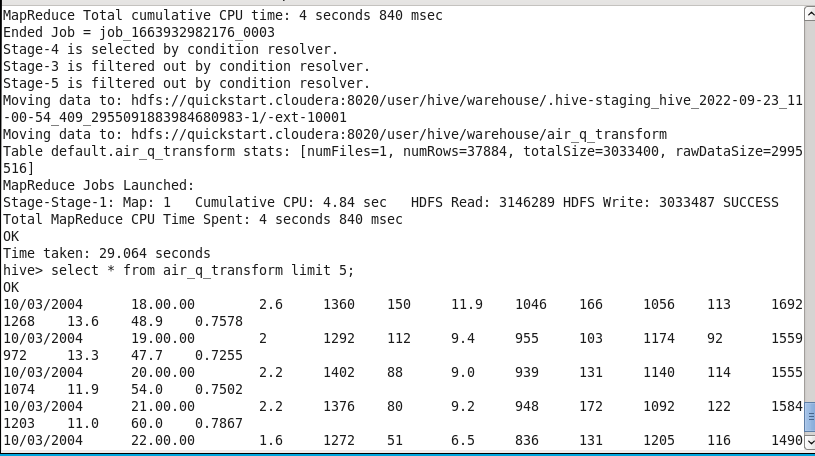
Loading Data in the table



As we can see the decimal values are stored as comma separated values ,so we need to transform this

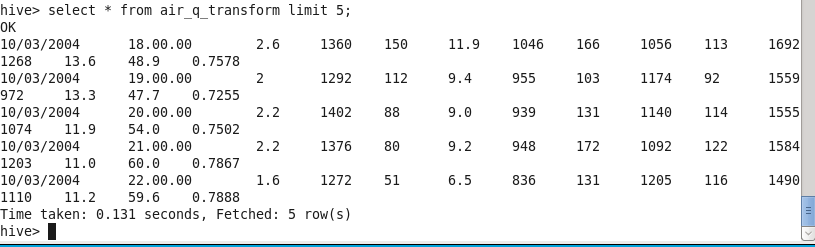
Step 2: transform the column of decimal values using regex\_replace and store data into other table





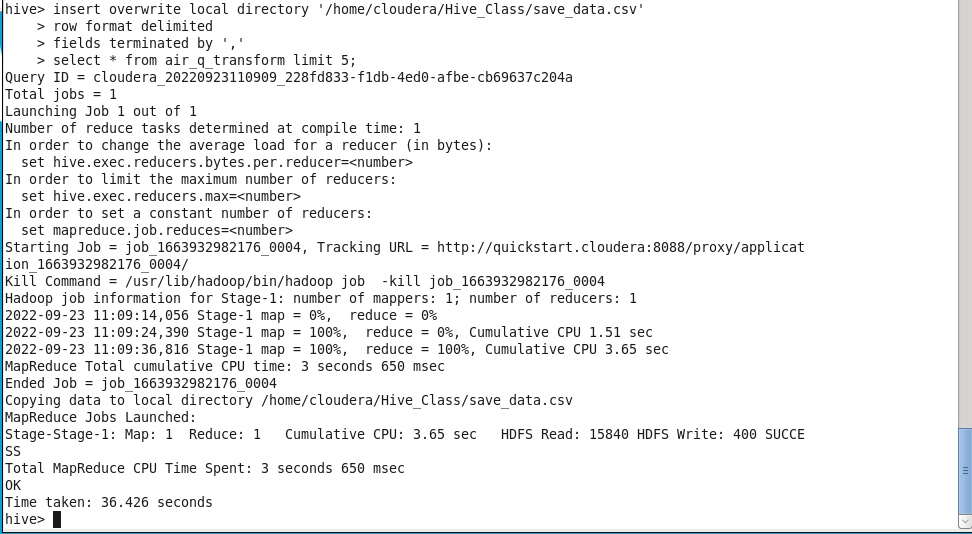
As u can see the comma separated decimal data got transformed into point separated data

3. Perform a select operation.

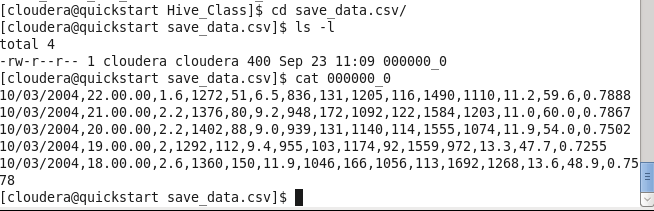


4. Fetch the result of the select operation in your local as a csv file .

Using insert overwrite local directory command I save the result of the query in the local system



Proof of data being stored in the local system in the form of partitioned file

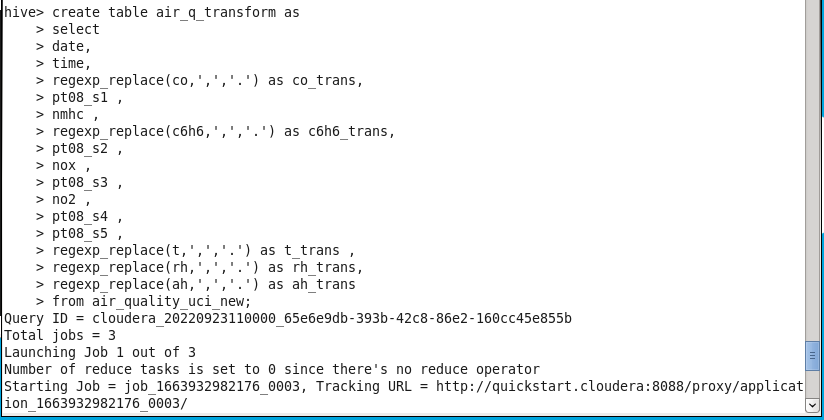


5. Perform group by operation .

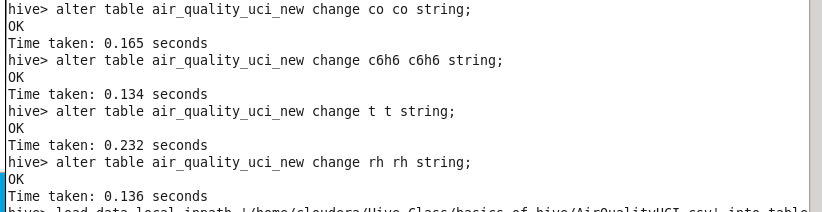


7. Perform filter operation at least 5 kinds of filter examples.

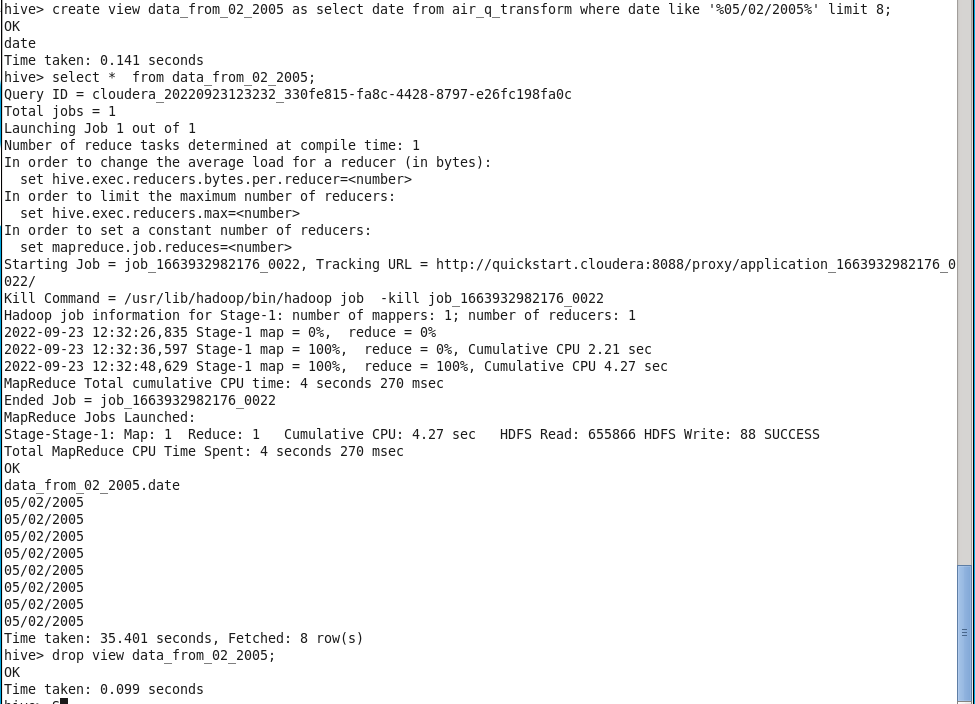
8. show and example of regex operation



9. alter table operation



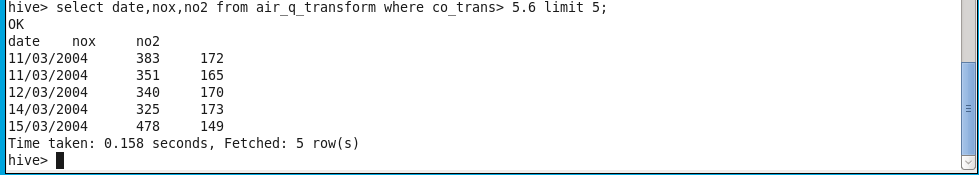
10 . drop table operation



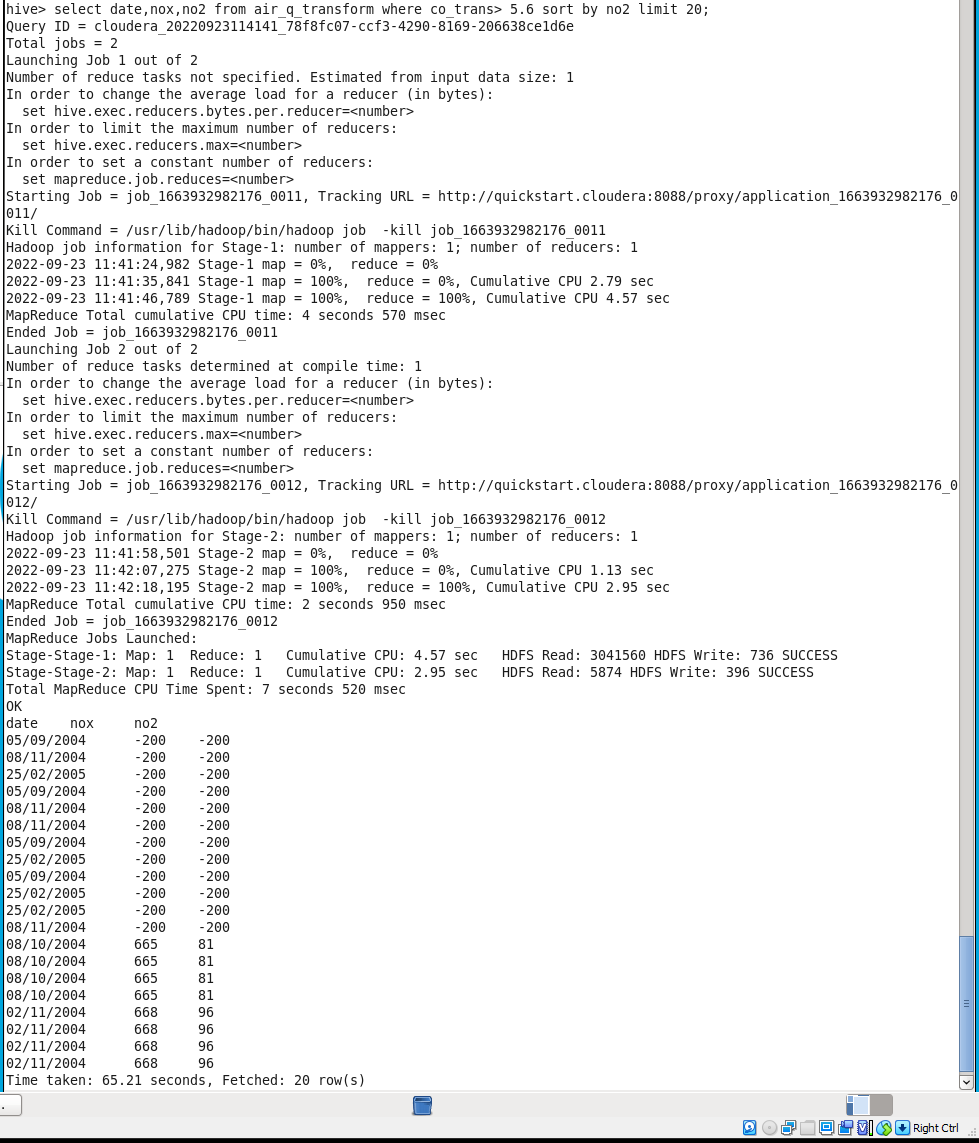
12 . order by operation .

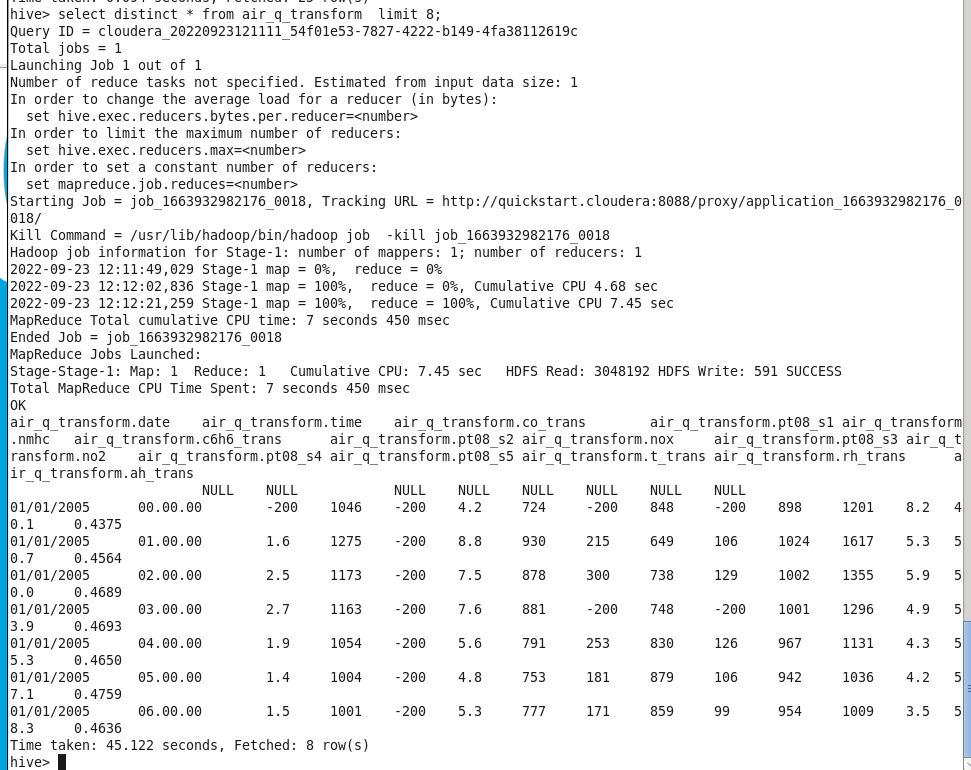


13 . where clause operations you have to perform .

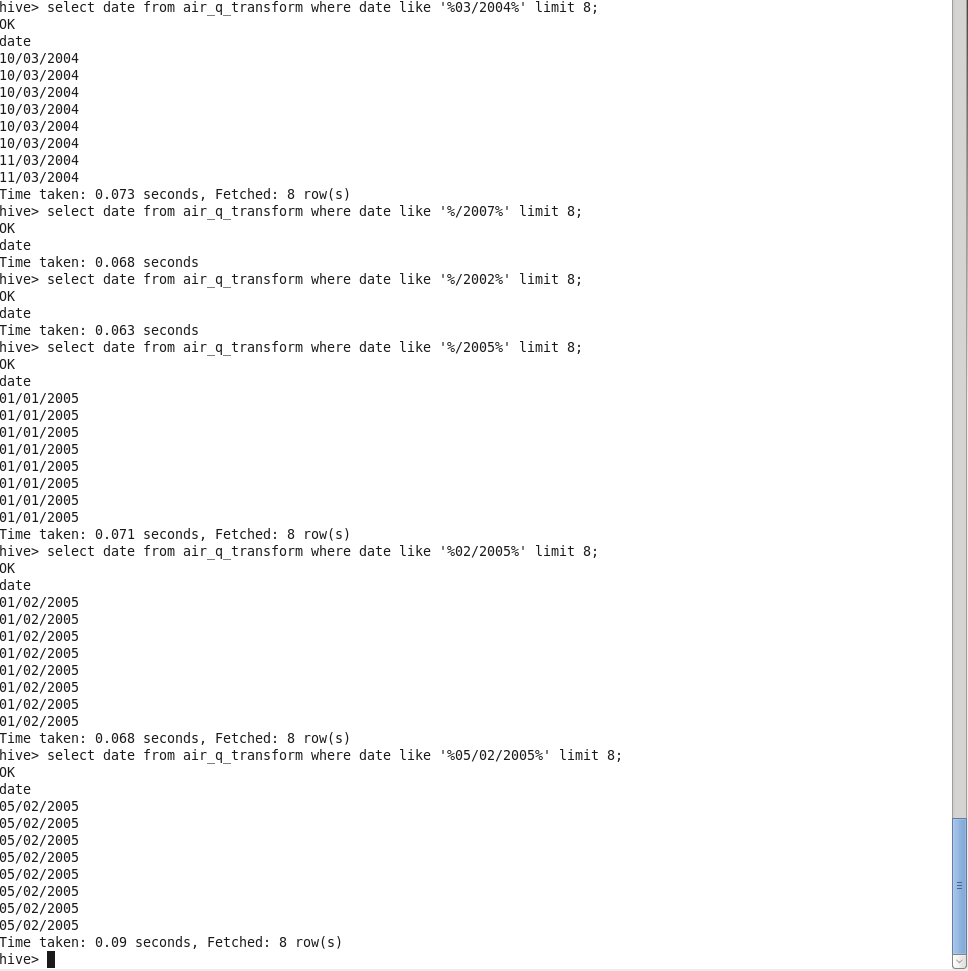


14 . sorting operation you have to perform .

15 . distinct operation you have to perform .



16 . like an operation you have to perform .

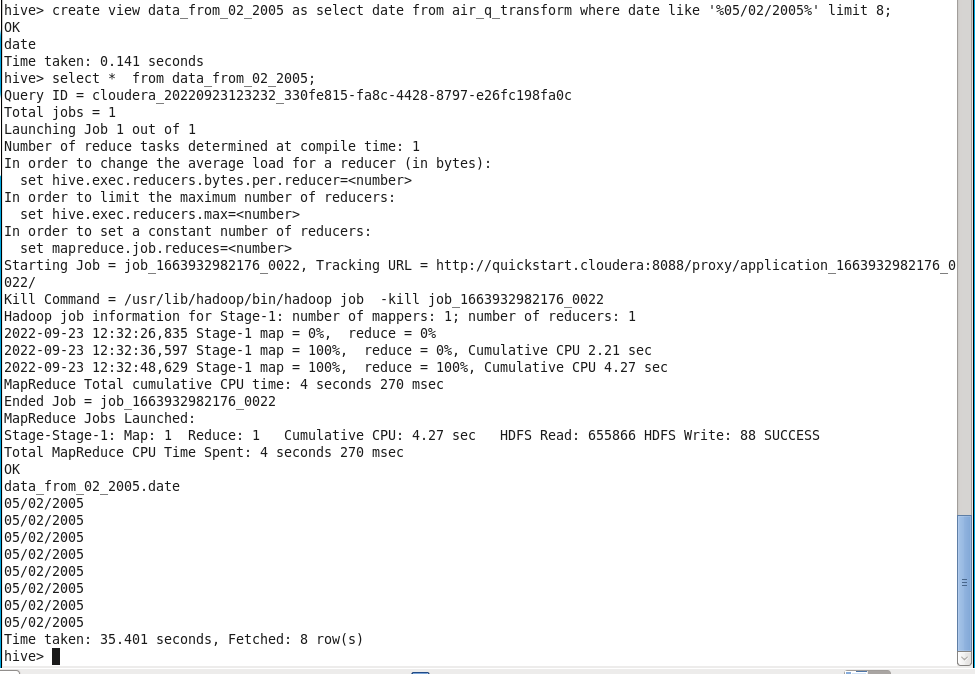


17 . union operation you have to perform .





18 . table view operation you have to perform .



hive operation with python

Create a python application that connects to the Hive database for extracting data, creating sub tables for data processing, drops temporary tables. Fetch rows to python itself into a list of tuples and mimic the join or filter operations