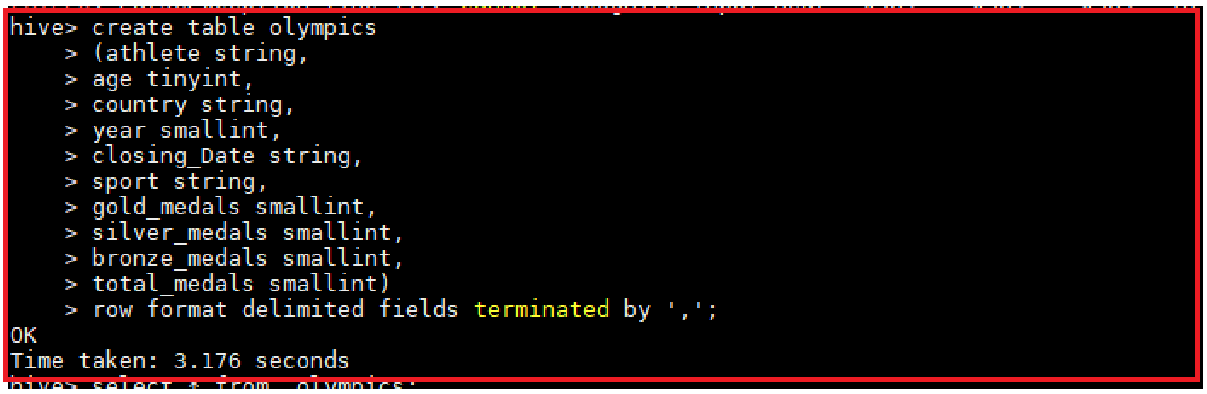
Assignment 9.1
Advance Hive

Task 1

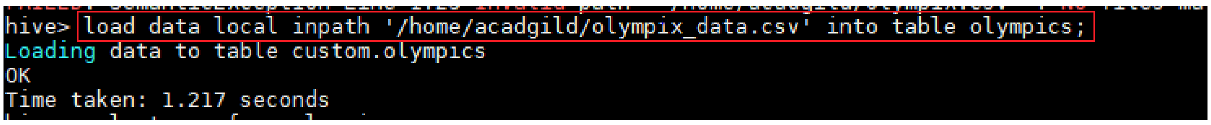
1. **Write a Hive program to find the number of medals won by each country in swimming.**

We have created **olympics** table with all required fields as columns and as we are using csv file **'olympix_data'** to load data into this table, we are using ',' as delimiter to separate column values:

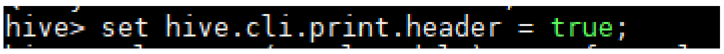


```
hive> create table olympics
    > (athlete string,
    > age tinyint,
    > country string,
    > year smallint,
    > closing_Date string,
    > sport string,
    > gold_medals smallint,
    > silver_medals smallint,
    > bronze_medals smallint,
    > total_medals smallint)
    > row format delimited fields terminated by ',';
OK
Time taken: 3.176 seconds
```

Then we have loaded data from **'olympix_data.csv'** file present in local system as shown in below screenshot.



```
hive> load data local inpath '/home/acadgild/olympix_data.csv' into table olympics;
Loading data to table custom.olympics
OK
Time taken: 1.217 seconds
```



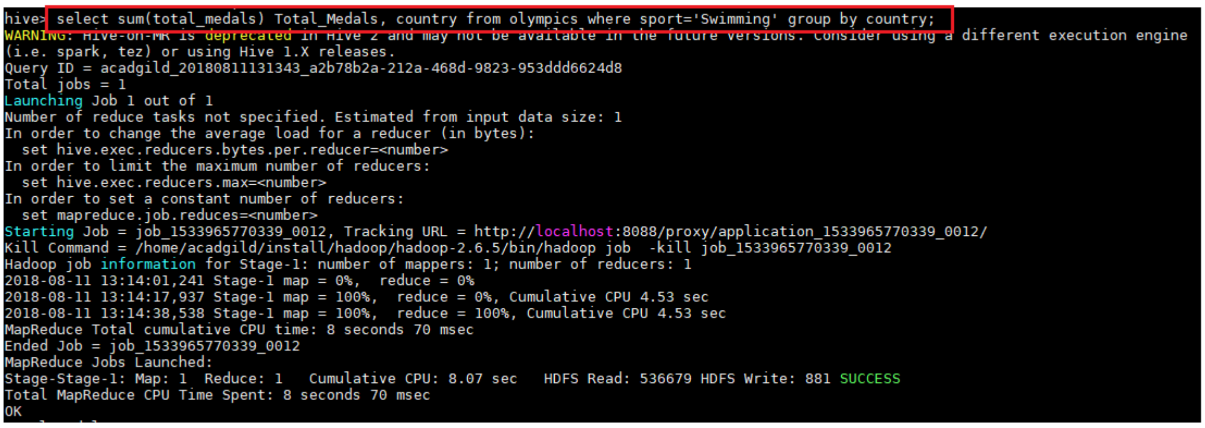
```
hive> set hive.cli.print.header = true;
```

Hive Query 1:

Here we are using **GROUP BY** clause to group all the records by using column 'country' and **sum** function to calculate total number of medals for a country in Swimming sport.

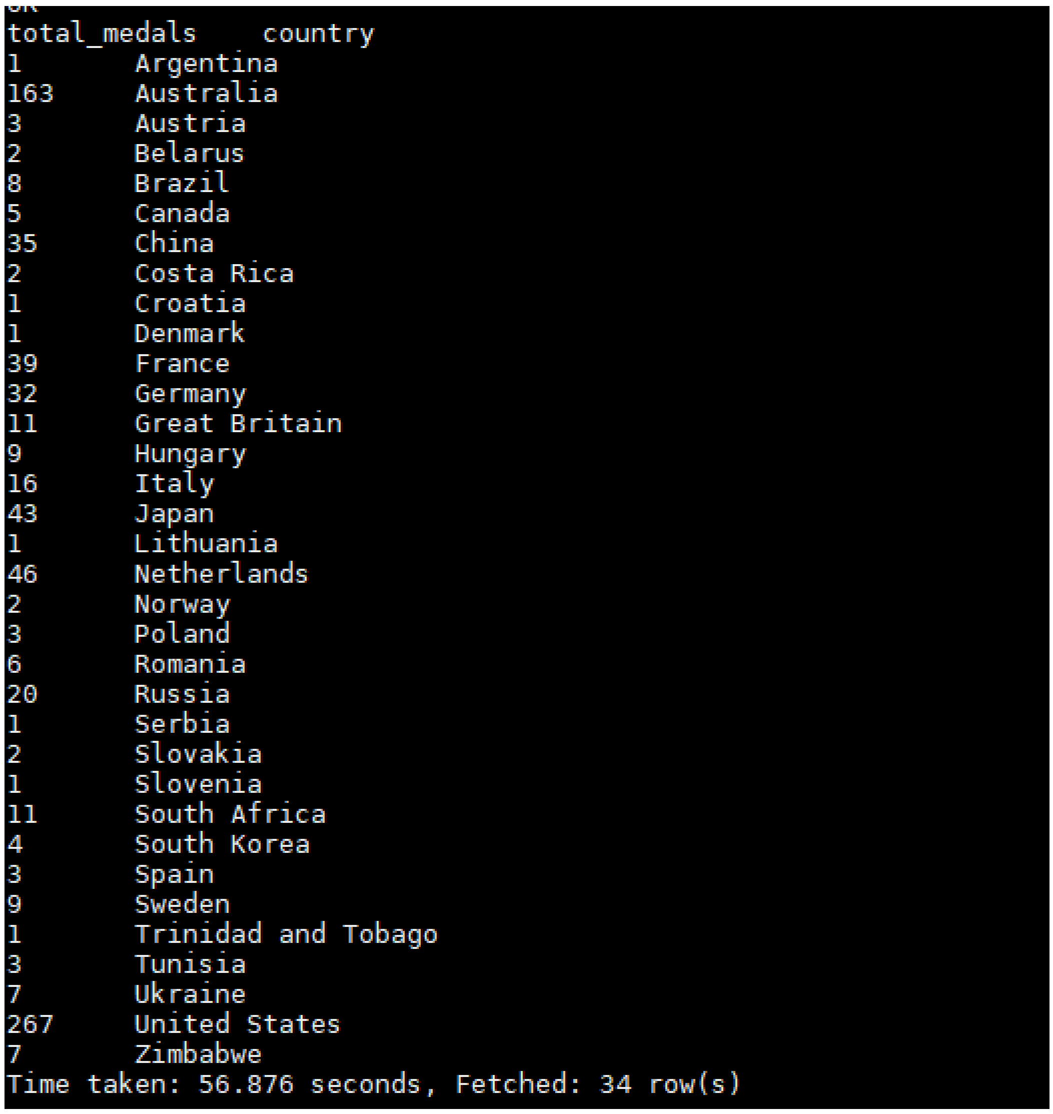
Assignment 9.1
Advance Hive

select sum(total_medals) Total_Medals, country from olympics where sport='Swimming' group by country;



```
hive> select sum(total_medals) Total_Medals, country from olympics where sport='Swimming' group by country;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine
(i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180811131343_a2b78b2a-212a-468d-9823-953ddd6624d8
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1533965770339_0012, Tracking URL = http://localhost:8088/proxy/application_1533965770339_0012/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1533965770339_0012
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-08-11 13:14:01,241 Stage-1 map = 0%,  reduce = 0%
2018-08-11 13:14:17,937 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.53 sec
2018-08-11 13:14:38,538 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.53 sec
MapReduce Total cumulative CPU time: 8 seconds 70 msec
Ended Job = job_1533965770339_0012
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 8.07 sec   HDFS Read: 536679 HDFS Write: 881 SUCCESS
Total MapReduce CPU Time Spent: 8 seconds 70 msec
OK
```

Output of Query 1:



```
total_medals    country
1       Argentina
163     Australia
3       Austria
2       Belarus
8       Brazil
5       Canada
35      China
2       Costa Rica
1       Croatia
1       Denmark
39      France
32      Germany
11      Great Britain
9       Hungary
16      Italy
43      Japan
1       Lithuania
46      Netherlands
2       Norway
3       Poland
6       Romania
20      Russia
1       Serbia
2       Slovakia
1       Slovenia
11      South Africa
4       South Korea
3       Spain
9       Sweden
1       Trinidad and Tobago
3       Tunisia
7       Ukraine
267     United States
7       Zimbabwe
Time taken: 56.876 seconds, Fetched: 34 row(s)
```

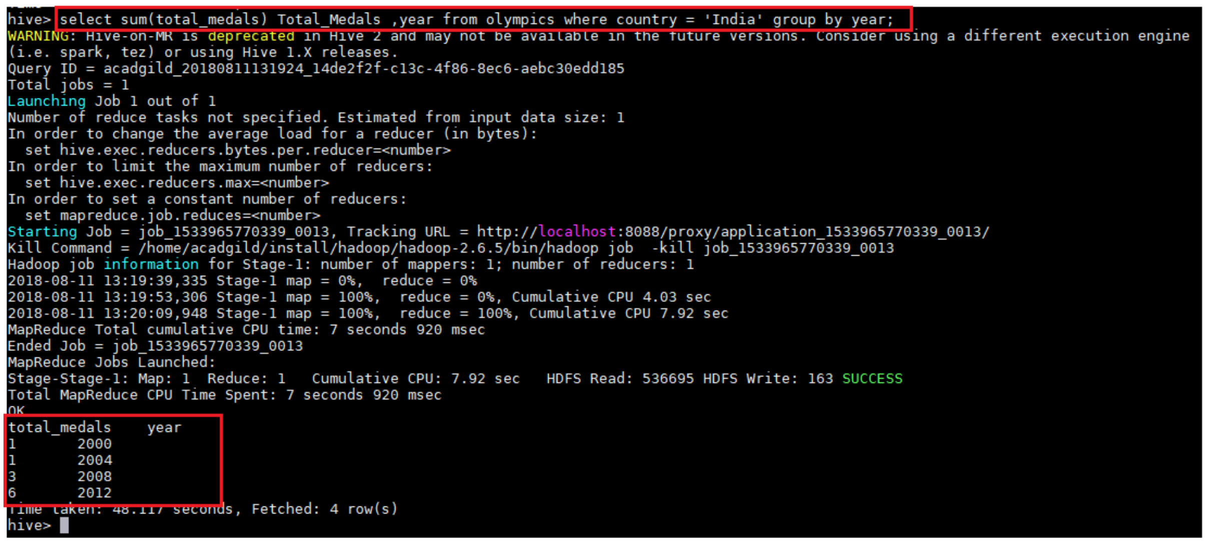
Assignment 9.1
Advance Hive

2. Write a Hive program to find the number of medals that India won year wise.

Here we are using **GROUP BY** clause to group all the records by using column **'year'** and **sum** function to calculate total number of medals for a year in **India** country.

Hive Query 2:

select sum(total_medals) Total_Medals ,year from olympics where country = 'India' group by year;



```
hive> select sum(total_medals) Total_Medals ,year from olympics where country = 'India' group by year;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine
(i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180811131924_14de2f2f-c13c-4f86-8ec6-aebc30edd185
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1533965770339_0013, Tracking URL = http://localhost:8088/proxy/application_1533965770339_0013/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1533965770339_0013
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-08-11 13:19:39,335 Stage-1 map = 0%,  reduce = 0%
2018-08-11 13:19:53,306 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.03 sec
2018-08-11 13:20:09,948 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 7.92 sec
MapReduce Total cumulative CPU time: 7 seconds 920 msec
Ended Job = job_1533965770339_0013
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 7.92 sec   HDFS Read: 536695 HDFS Write: 163 SUCCESS
Total MapReduce CPU Time Spent: 7 seconds 920 msec
OK
total_medals    year
1       2000
1       2004
3       2008
6       2012
Time taken: 48.117 seconds, Fetched: 4 row(s)
hive>
```

Output :

total_medals	**year**
1	**2000**
1	**2004**
3	**2008**
6	**2012**

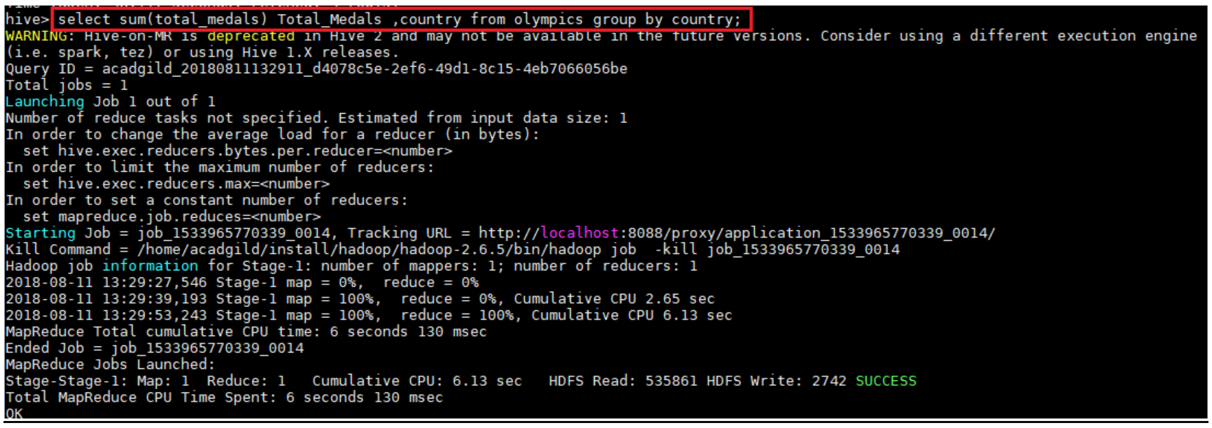
Assignment 9.1
Advance Hive

3. Write a Hive Program to find the total number of medals each country won.

Here we are using **GROUP BY** clause to group all the records by using column **'country'** and **sum** function to calculate total number of medals for a country.

<u>Hive Query 3:</u>

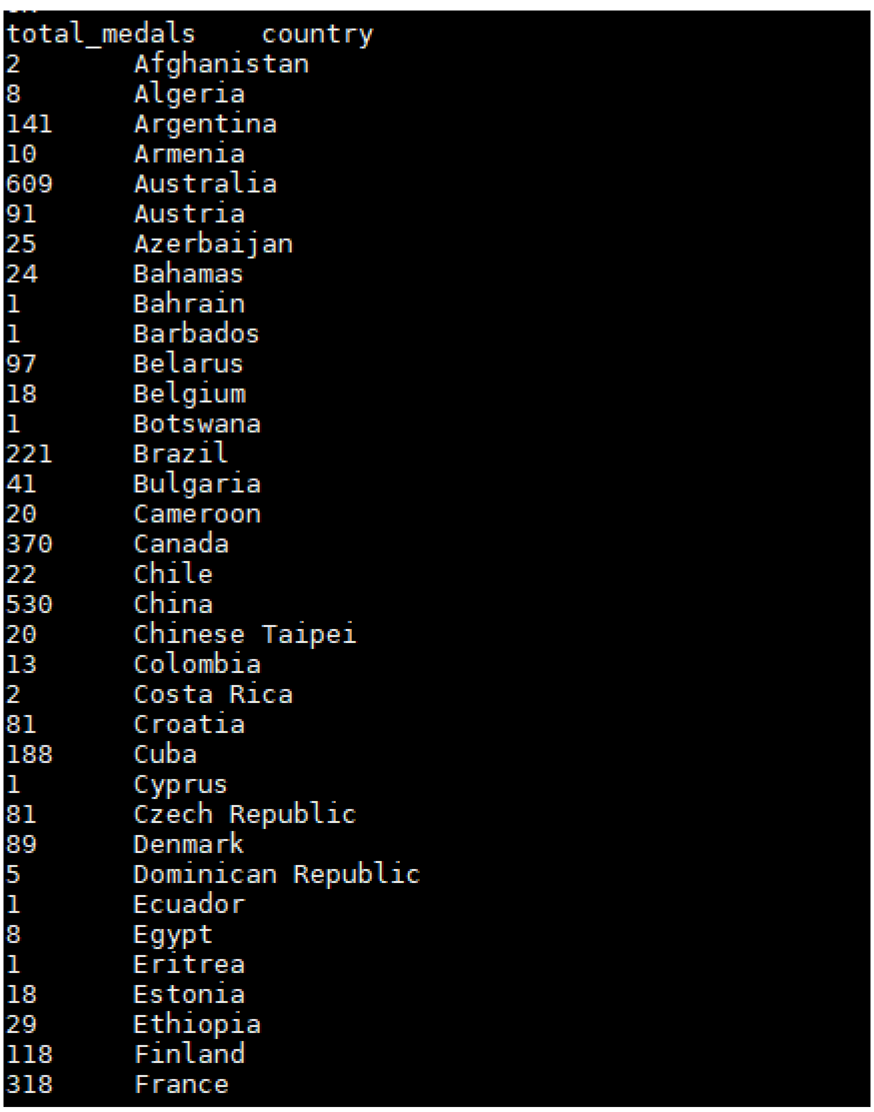
select sum(total_medals) Total_Medals ,country from olympics group by country;



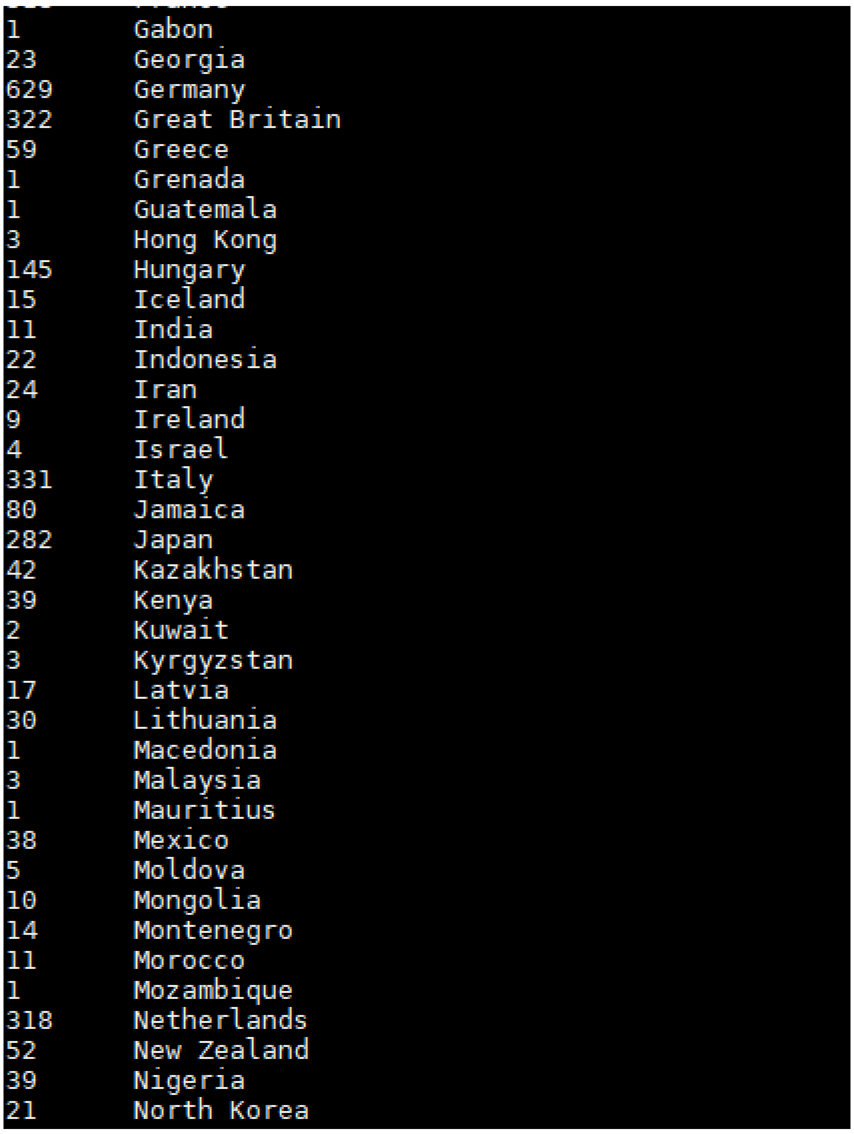
```
hive> select sum(total_medals) Total_Medals ,country from olympics group by country;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine
(i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180811132911_d4078c5e-2ef6-49d1-8c15-4eb7066056be
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1533965770339_0014, Tracking URL = http://localhost:8088/proxy/application_1533965770339_0014/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1533965770339_0014
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-08-11 13:29:27,546 Stage-1 map = 0%,  reduce = 0%
2018-08-11 13:29:39,193 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.65 sec
2018-08-11 13:29:53,243 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 6.13 sec
MapReduce Total cumulative CPU time: 6 seconds 130 msec
Ended Job = job_1533965770339_0014
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 6.13 sec   HDFS Read: 535861 HDFS Write: 2742 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 130 msec
OK
```

<u>Output :</u>

Assignment 9.1
Advance Hive

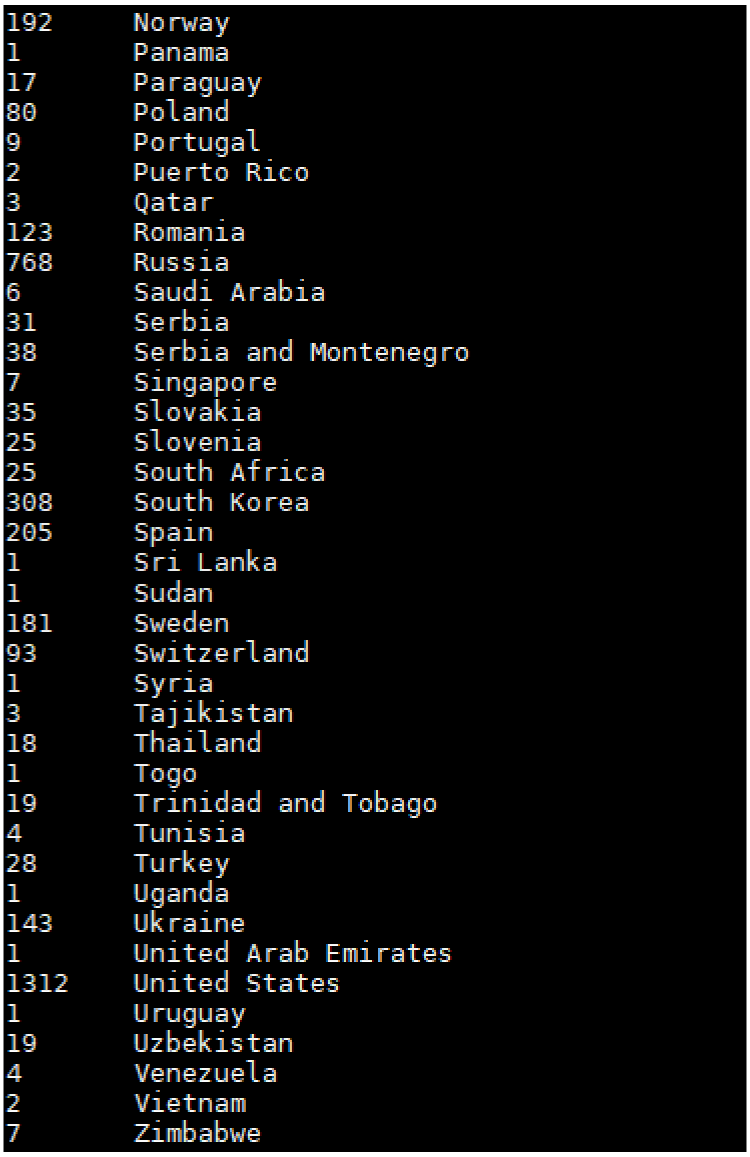


```
total_medals    country
2       Afghanistan
8       Algeria
141     Argentina
10      Armenia
609     Australia
91      Austria
25      Azerbaijan
24      Bahamas
1       Bahrain
1       Barbados
97      Belarus
18      Belgium
1       Botswana
221     Brazil
41      Bulgaria
20      Cameroon
370     Canada
22      Chile
530     China
20      Chinese Taipei
13      Colombia
2       Costa Rica
81      Croatia
188     Cuba
1       Cyprus
81      Czech Republic
89      Denmark
5       Dominican Republic
1       Ecuador
8       Egypt
1       Eritrea
18      Estonia
29      Ethiopia
118     Finland
318     France
```



```
1       Gabon
23      Georgia
629     Germany
322     Great Britain
59      Greece
1       Grenada
1       Guatemala
3       Hong Kong
145     Hungary
15      Iceland
11      India
22      Indonesia
24      Iran
9       Ireland
4       Israel
331     Italy
80      Jamaica
282     Japan
42      Kazakhstan
39      Kenya
2       Kuwait
3       Kyrgyzstan
17      Latvia
30      Lithuania
1       Macedonia
3       Malaysia
1       Mauritius
38      Mexico
5       Moldova
10      Mongolia
14      Montenegro
11      Morocco
1       Mozambique
318     Netherlands
52      New Zealand
39      Nigeria
21      North Korea
```

Assignment 9.1
Advance Hive

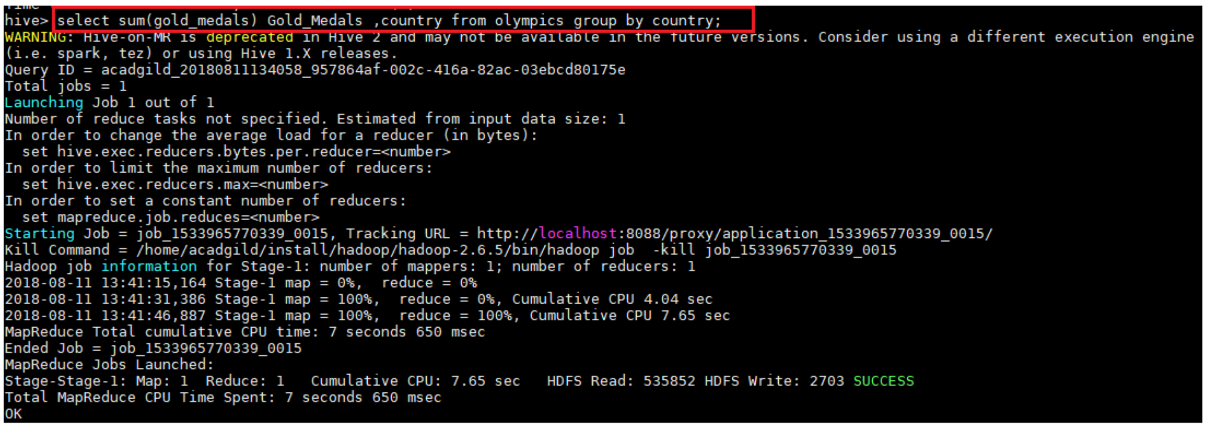

```
192     Norway
1       Panama
17      Paraguay
80      Poland
9       Portugal
2       Puerto Rico
3       Qatar
123     Romania
768     Russia
6       Saudi Arabia
31      Serbia
38      Serbia and Montenegro
7       Singapore
35      Slovakia
25      Slovenia
25      South Africa
308     South Korea
205     Spain
1       Sri Lanka
1       Sudan
181     Sweden
93      Switzerland
1       Syria
3       Tajikistan
18      Thailand
1       Togo
19      Trinidad and Tobago
4       Tunisia
28      Turkey
1       Uganda
143     Ukraine
1       United Arab Emirates
1312    United States
1       Uruguay
19      Uzbekistan
4       Venezuela
2       Vietnam
7       Zimbabwe
```

4. **Write a Hive program to find the number of gold medals each country won.**

   Here we are using **GROUP BY** clause to group all the records by using column **'country'** and **sum** function to calculate total number of Gold medals for a country.

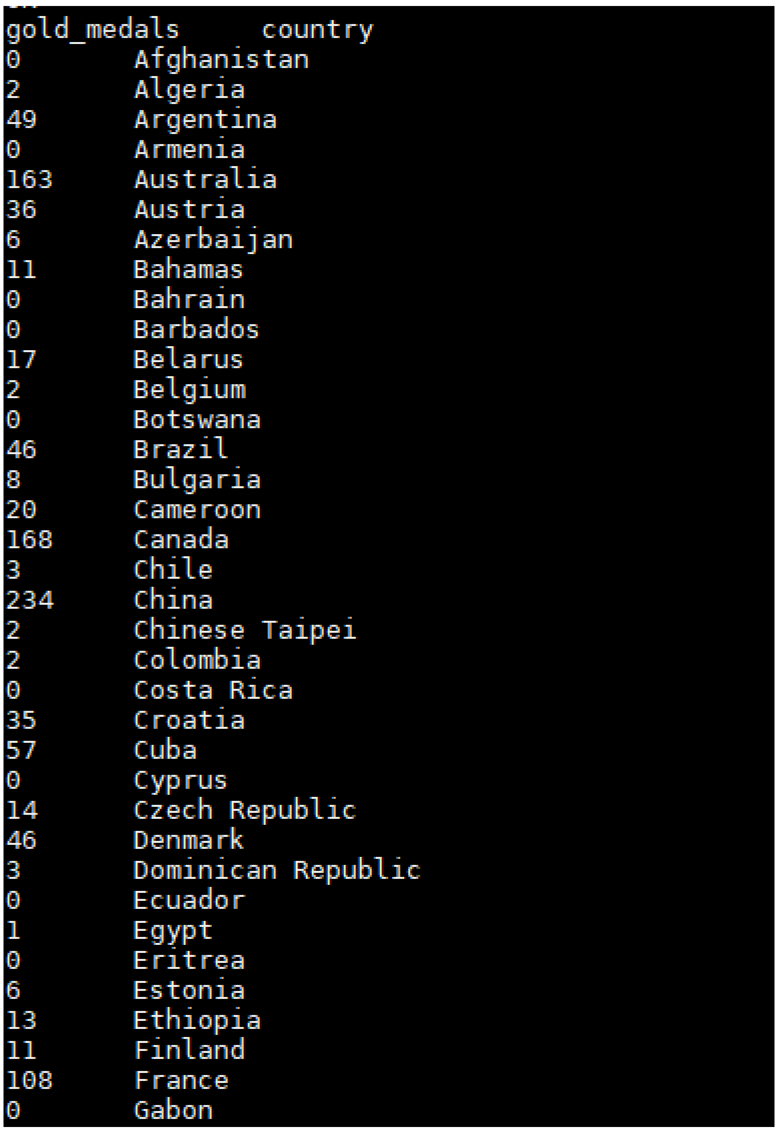
<u>Hive Query 4 :</u>

select sum(gold_medals) Gold_Medals ,country from olympics group by country;



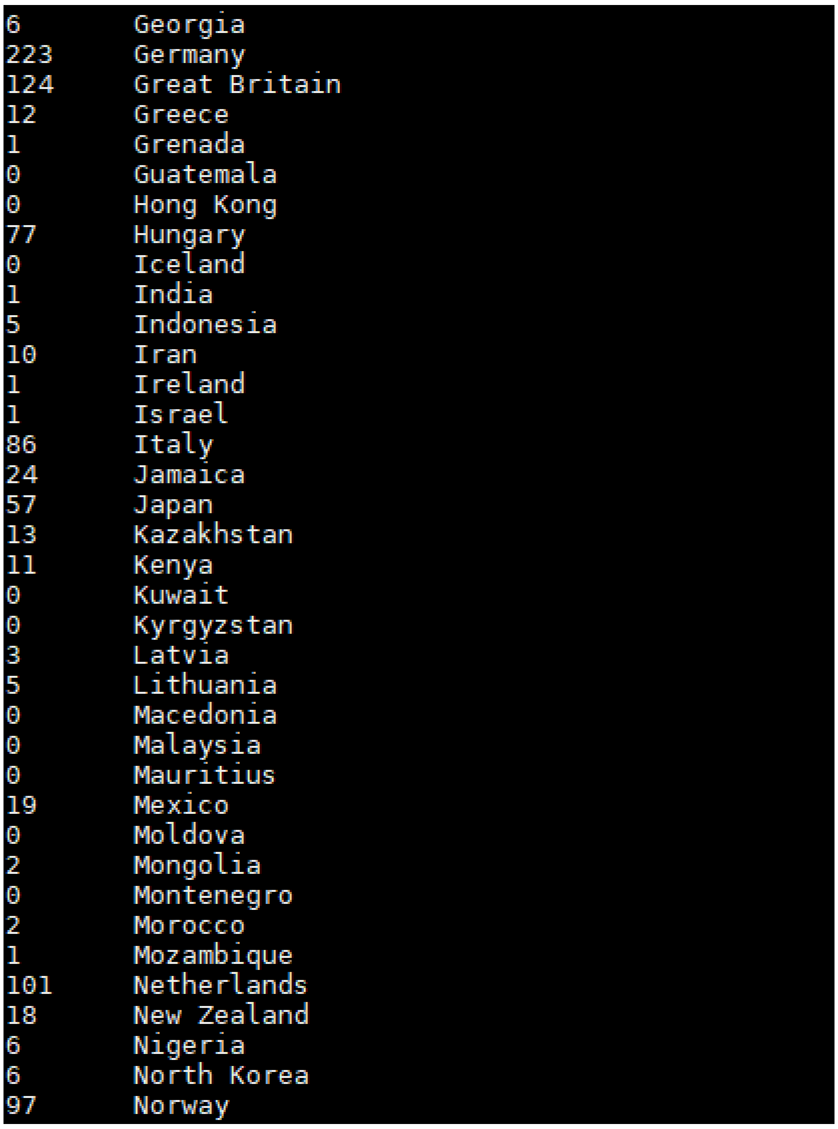
```
hive> select sum(gold_medals) Gold_Medals ,country from olympics group by country;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine
(i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180811134058_957864af-002c-416a-82ac-03ebcd80175e
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1533965770339_0015, Tracking URL = http://localhost:8088/proxy/application_1533965770339_0015/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1533965770339_0015
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-08-11 13:41:15,164 Stage-1 map = 0%,  reduce = 0%
2018-08-11 13:41:31,386 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.04 sec
2018-08-11 13:41:46,887 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 7.65 sec
MapReduce Total cumulative CPU time: 7 seconds 650 msec
Ended Job = job_1533965770339_0015
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 7.65 sec   HDFS Read: 535852 HDFS Write: 2703 SUCCESS
Total MapReduce CPU Time Spent: 7 seconds 650 msec
OK
```

Assignment 9.1
Advance Hive



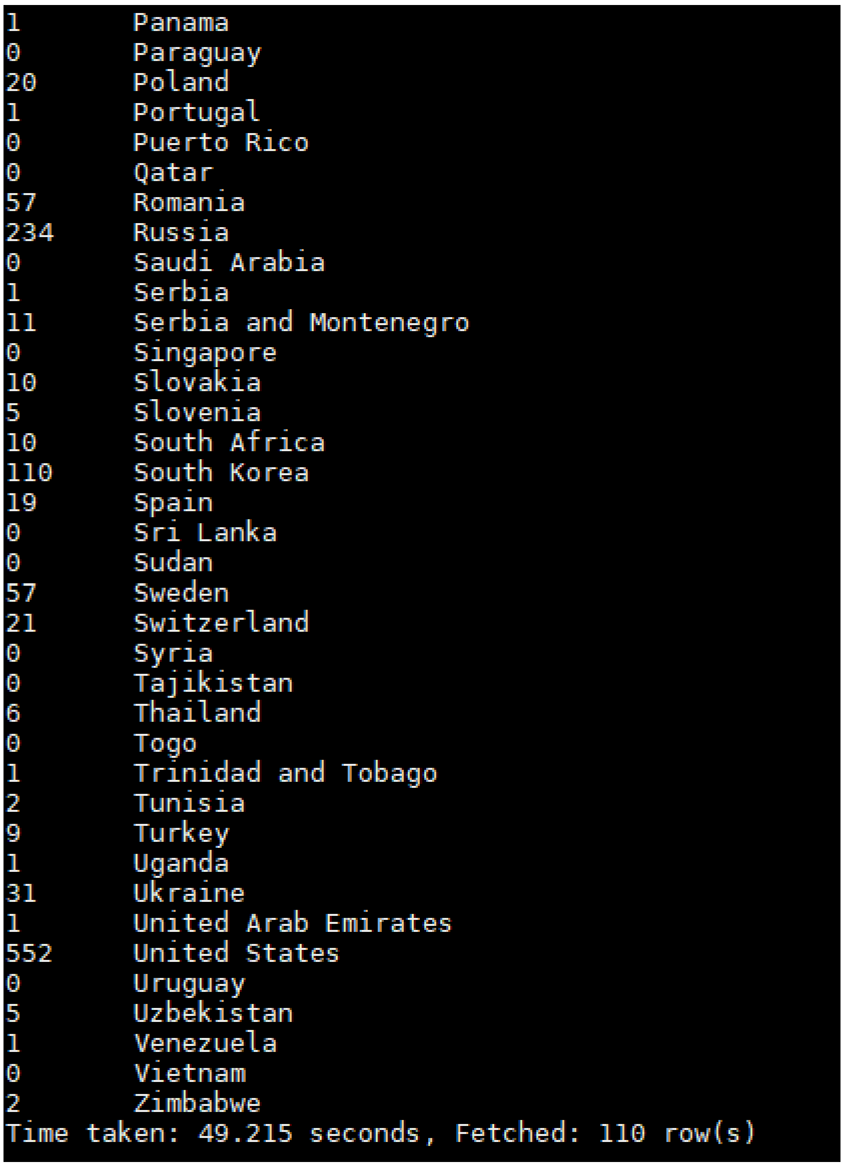
```
gold_medals     country
0       Afghanistan
2       Algeria
49      Argentina
0       Armenia
163     Australia
36      Austria
6       Azerbaijan
11      Bahamas
0       Bahrain
0       Barbados
17      Belarus
2       Belgium
0       Botswana
46      Brazil
8       Bulgaria
20      Cameroon
168     Canada
3       Chile
234     China
2       Chinese Taipei
2       Colombia
0       Costa Rica
35      Croatia
57      Cuba
0       Cyprus
14      Czech Republic
46      Denmark
3       Dominican Republic
0       Ecuador
1       Egypt
0       Eritrea
6       Estonia
13      Ethiopia
11      Finland
108     France
0       Gabon
```

Assignment 9.1
Advance Hive



```
6       Georgia
223     Germany
124     Great Britain
12      Greece
1       Grenada
0       Guatemala
0       Hong Kong
77      Hungary
0       Iceland
1       India
5       Indonesia
10      Iran
1       Ireland
1       Israel
86      Italy
24      Jamaica
57      Japan
13      Kazakhstan
11      Kenya
0       Kuwait
0       Kyrgyzstan
3       Latvia
5       Lithuania
0       Macedonia
0       Malaysia
0       Mauritius
19      Mexico
0       Moldova
2       Mongolia
0       Montenegro
2       Morocco
1       Mozambique
101     Netherlands
18      New Zealand
6       Nigeria
6       North Korea
97      Norway
```

Assignment 9.1
Advance Hive



```
1       Panama
0       Paraguay
20      Poland
1       Portugal
0       Puerto Rico
0       Qatar
57      Romania
234     Russia
0       Saudi Arabia
1       Serbia
11      Serbia and Montenegro
0       Singapore
10      Slovakia
5       Slovenia
10      South Africa
110     South Korea
19      Spain
0       Sri Lanka
0       Sudan
57      Sweden
21      Switzerland
0       Syria
0       Tajikistan
6       Thailand
0       Togo
1       Trinidad and Tobago
2       Tunisia
9       Turkey
1       Uganda
31      Ukraine
1       United Arab Emirates
552     United States
0       Uruguay
5       Uzbekistan
1       Venezuela
0       Vietnam
2       Zimbabwe
Time taken: 49.215 seconds, Fetched: 110 row(s)
```

Assignment 9.1
Advance Hive

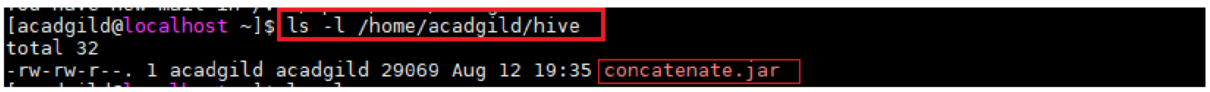
Task 2 :

Write a hive UDF that implements functionality of string concat_ws(string SEP, array<string>).
This UDF will accept two arguments, one string and one array of string.
It will return a single string where all the elements of the array are separated by the SE

We have written below java program **'concatenate.java'** .
Evaluate function takes delimiter SEP and Array of Strings (List, in Java) as input.
Then returns **word** as concatenated string like output of **concat_ws** function.

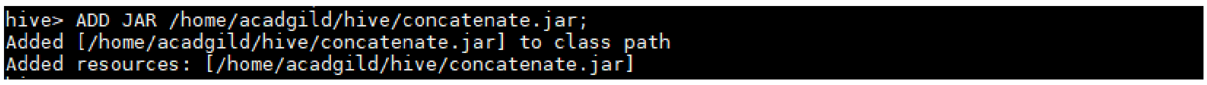
```java
import org.apache.hadoop.hive.ql.exec.UDF;
import org.apache.hadoop.io.Text;
import java.util.List;

public class concatenate extends UDF {


    public Text evaluate(Text SEP,List<String> arr) {
             Text to_value = new Text("");
             if (arr != null) {

                    String word = "";
                         for (int i=0; i<arr.size(); i++) {
                                if(i==0)
                                       word =word + arr.get(i);
                                else
                                       word = word+ SEP + arr.get(i);
                         }


                    to_value.set(word);
              }
             return to_value;
    }

}
```

Then from this java code, we have exported JAR file as **'concatenate.jar'**.
After this we have added this JAR file into our VM in path location: **/home/acadgild/hive**

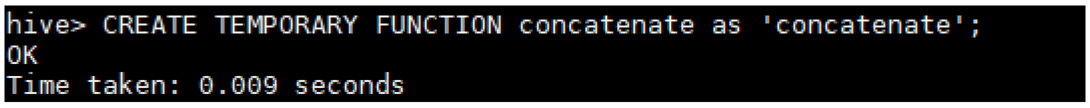


```
[acadgild@localhost ~]$ ls -l /home/acadgild/hive
total 32
-rw-rw-r--. 1 acadgild acadgild 29069 Aug 12 19:35 concatenate.jar
```

Then we have created a temporary function **concatenate** below :



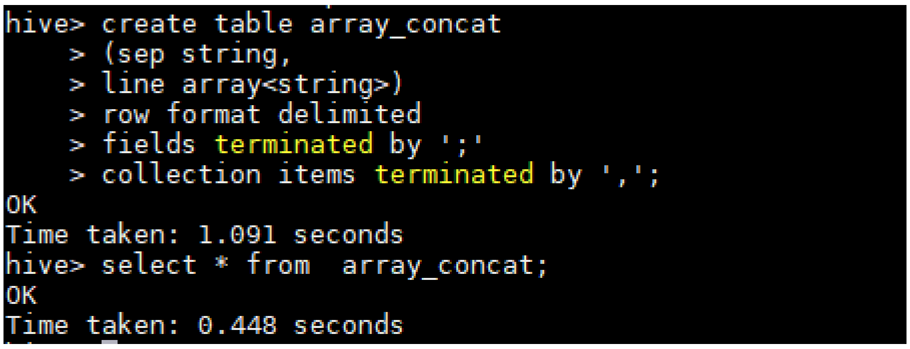
```
hive> ADD JAR /home/acadgild/hive/concatenate.jar;
Added [/home/acadgild/hive/concatenate.jar] to class path
Added resources: [/home/acadgild/hive/concatenate.jar]
```



```
hive> CREATE TEMPORARY FUNCTION concatenate as 'concatenate';
OK
Time taken: 0.009 seconds
```

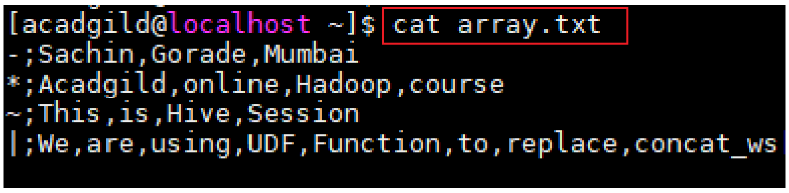
Assignment 9.1
Advance Hive

We have created table **array_concat** with columns as sep as string and line as array of strings.



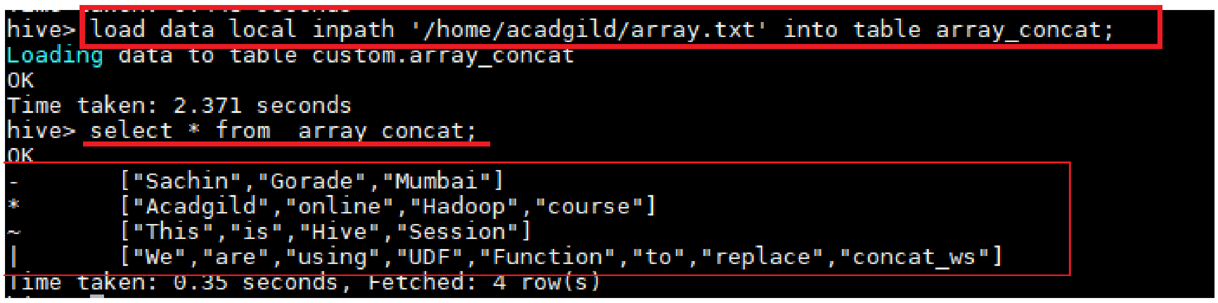
```
hive> create table array_concat
    > (sep string,
    > line array<string>)
    > row format delimited
    > fields terminated by ';'
    > collection items terminated by ',';
OK
Time taken: 1.091 seconds
hive> select * from  array_concat;
OK
Time taken: 0.448 seconds
```

Below is content in array.txt with two fields:



```
[acadgild@localhost ~]$ cat array.txt
-;Sachin,Gorade,Mumbai
*;Acadgild,online,Hadoop,course
~;This,is,Hive,Session
|;We,are,using,UDF,Function,to,replace,concat_ws
```

Then we have loaded data from array.txt file into **array_concat** table.

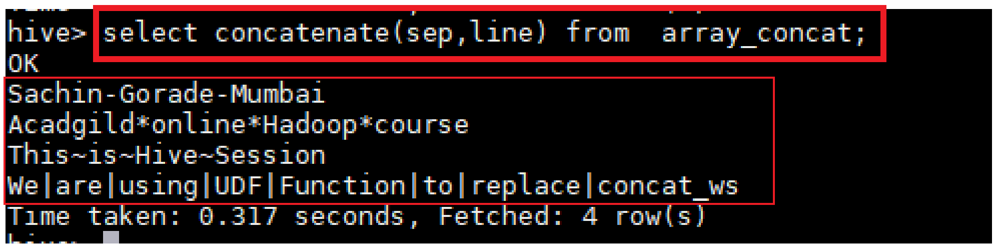


```
hive> load data local inpath '/home/acadgild/array.txt' into table array_concat;
Loading data to table custom.array_concat
OK
Time taken: 2.371 seconds
hive> select * from  array_concat;
OK
-       ["Sachin","Gorade","Mumbai"]
*       ["Acadgild","online","Hadoop","course"]
~       ["This","is","Hive","Session"]
|       ["We","are","using","UDF","Function","to","replace","concat_ws"]
Time taken: 0.35 seconds, Fetched: 4 row(s)
```

This is the output. You could see below that we have used **concatenate** function and used sep and line columns as input from **array_concat** table.

e.g. in first row, '-' is delimiter and strings in array are "Sachin","Gorade" and "Mumbai".

So concatenated output is **Sachin-Gorade-Mumbai.**



```
hive> select concatenate(sep,line) from  array_concat;
OK
Sachin-Gorade-Mumbai
Acadgild*online*Hadoop*course
This~is~Hive~Session
We|are|using|UDF|Function|to|replace|concat_ws
Time taken: 0.317 seconds, Fetched: 4 row(s)
```

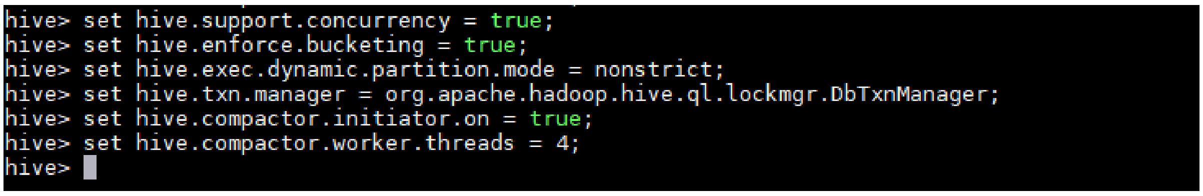
Assignment 9.1
Advance Hive

Task 3

Link: https://acadgild.com/blog/transactions-in-hive/

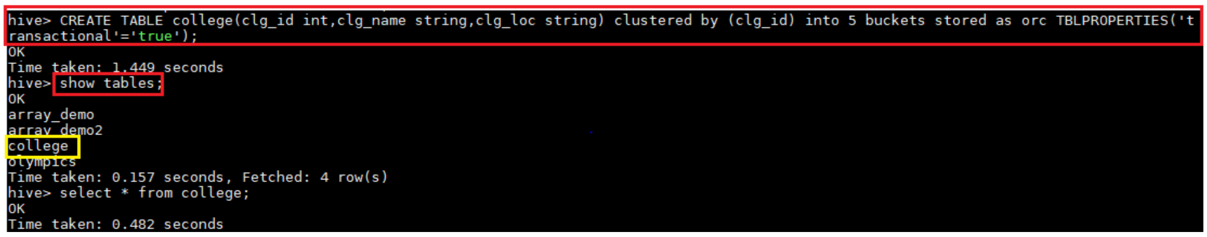
Refer the above given link for transactions in Hive and implement the operations given in the blog using your own sample data set and send us the screenshot.

We are setting below properties in Hive.Because without setting these properties 'Update' and 'Delete' will not work and we will receive errors.



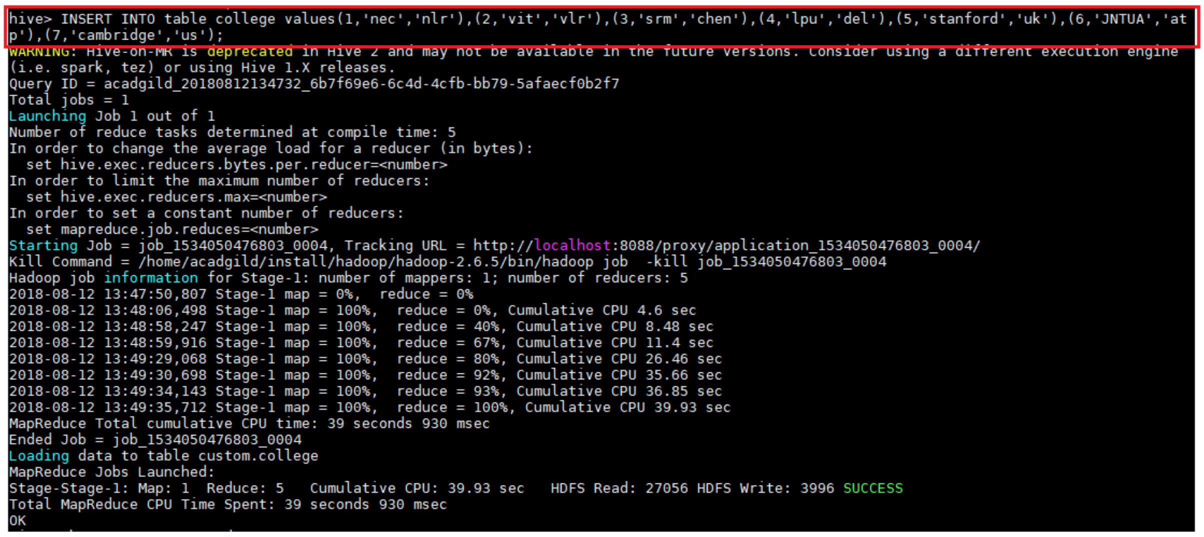
```
hive> set hive.support.concurrency = true;
hive> set hive.enforce.bucketing = true;
hive> set hive.exec.dynamic.partition.mode = nonstrict;
hive> set hive.txn.manager = org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;
hive> set hive.compactor.initiator.on = true;
hive> set hive.compactor.worker.threads = 4;
hive> 
```

We have created a table with name **'college'** and its columns are clg_id, clg_name, clg_loc.
We are bucketing this table by **clg_id** column and using ORC file format.



```
hive> CREATE TABLE college(clg_id int,clg_name string,clg_loc string) clustered by (clg_id) into 5 buckets stored as orc TBLPROPERTIES('t
ransactional'='true');
OK
Time taken: 1.449 seconds
hive> show tables;
OK
array_demo
array_demo2
college
olympics
Time taken: 0.157 seconds, Fetched: 4 row(s)
hive> select * from college;
OK
Time taken: 0.482 seconds
```

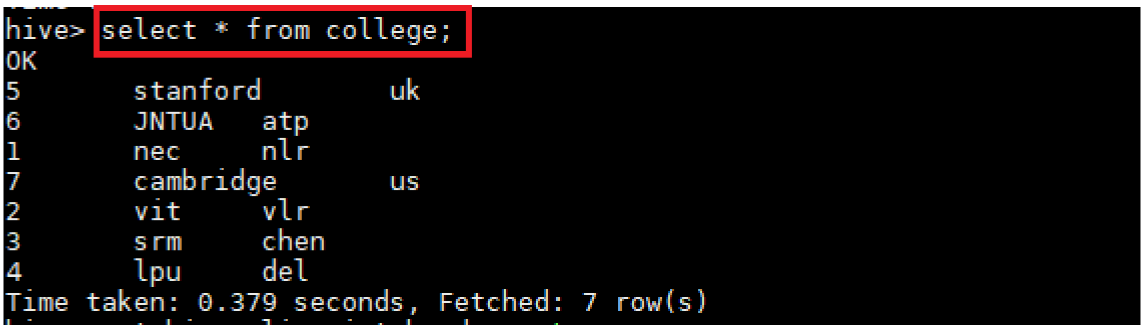
Then we have inserted data into this college table with below insert command:



```
hive> INSERT INTO table college values(1,'nec','nlr'),(2,'vit','vlr'),(3,'srm','chen'),(4,'lpu','del'),(5,'stanford','uk'),(6,'JNTUA','at
p'),(7,'cambridge','us');
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine
(i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180812134732_6b7f69e6-6c4d-4cfb-bb79-5afaecf0b2f7
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 5
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1534050476803_0004, Tracking URL = http://localhost:8088/proxy/application_1534050476803_0004/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1534050476803_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 5
2018-08-12 13:47:50,807 Stage-1 map = 0%,  reduce = 0%
2018-08-12 13:48:06,498 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.6 sec
2018-08-12 13:48:58,247 Stage-1 map = 100%,  reduce = 40%, Cumulative CPU 8.48 sec
2018-08-12 13:48:59,916 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 11.4 sec
2018-08-12 13:49:29,068 Stage-1 map = 100%,  reduce = 80%, Cumulative CPU 26.46 sec
2018-08-12 13:49:30,698 Stage-1 map = 100%,  reduce = 92%, Cumulative CPU 35.66 sec
2018-08-12 13:49:34,143 Stage-1 map = 100%,  reduce = 93%, Cumulative CPU 36.85 sec
2018-08-12 13:49:35,712 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 39.93 sec
MapReduce Total cumulative CPU time: 39 seconds 930 msec
Ended Job = job_1534050476803_0004
Loading data to table custom.college
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 5   Cumulative CPU: 39.93 sec   HDFS Read: 27056 HDFS Write: 3996 SUCCESS
Total MapReduce CPU Time Spent: 39 seconds 930 msec
OK
```

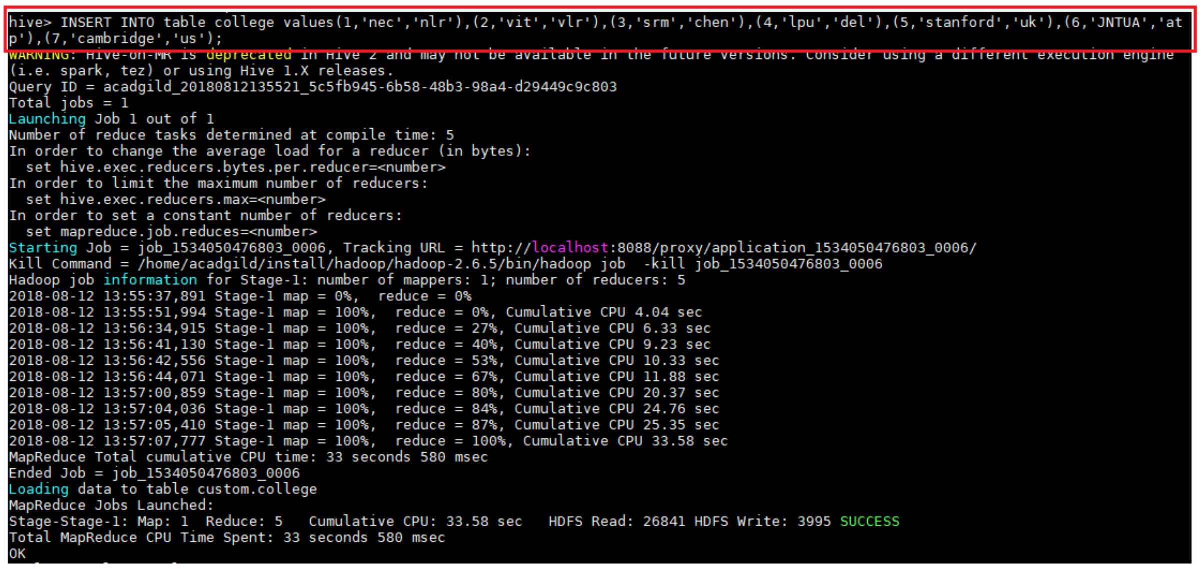
Assignment 9.1
Advance Hive

We could see below that 7 rows have been inserted into college table successfully.



```
hive> select * from college;
OK
5       stanford        uk
6       JNTUA   atp
1       nec     nlr
7       cambridge       us
2       vit     vlr
3       srm     chen
4       lpu     del
Time taken: 0.379 seconds, Fetched: 7 row(s)
```

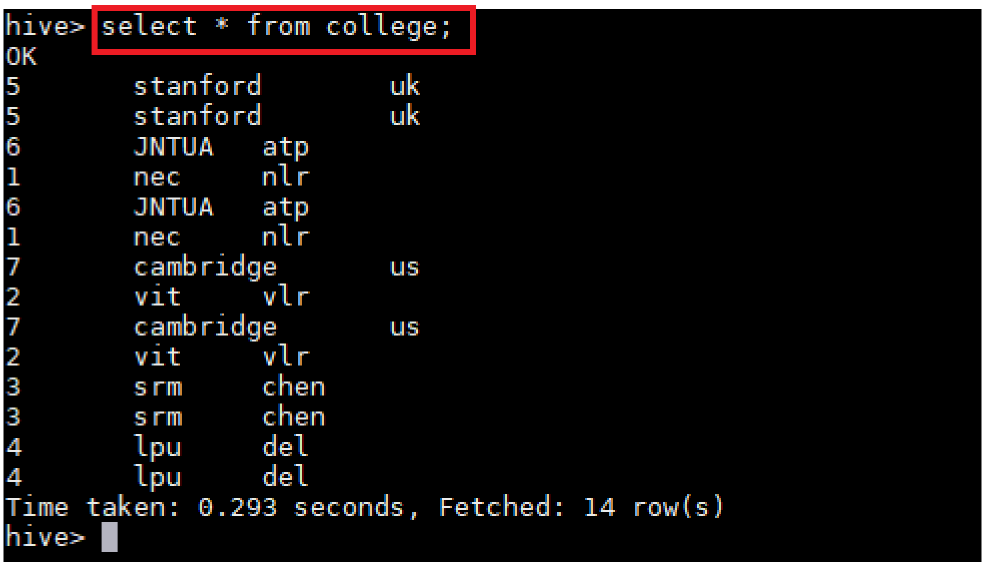
Now we are inserting same records again into college table and these rows will be appended.



```
hive> INSERT INTO table college values(1,'nec','nlr'),(2,'vit','vlr'),(3,'srm','chen'),(4,'lpu','del'),(5,'stanford','uk'),(6,'JNTUA','at
p'),(7,'cambridge','us');
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine
(i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180812135521_5c5fb945-6b58-48b3-98a4-d29449c9c803
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 5
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1534050476803_0006, Tracking URL = http://localhost:8088/proxy/application_1534050476803_0006/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1534050476803_0006
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 5
2018-08-12 13:55:37,891 Stage-1 map = 0%,  reduce = 0%
2018-08-12 13:55:51,994 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.04 sec
2018-08-12 13:56:34,915 Stage-1 map = 100%,  reduce = 27%, Cumulative CPU 6.33 sec
2018-08-12 13:56:41,130 Stage-1 map = 100%,  reduce = 40%, Cumulative CPU 9.23 sec
2018-08-12 13:56:42,556 Stage-1 map = 100%,  reduce = 53%, Cumulative CPU 10.33 sec
2018-08-12 13:56:44,071 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 11.88 sec
2018-08-12 13:57:00,859 Stage-1 map = 100%,  reduce = 80%, Cumulative CPU 20.37 sec
2018-08-12 13:57:04,036 Stage-1 map = 100%,  reduce = 84%, Cumulative CPU 24.76 sec
2018-08-12 13:57:05,410 Stage-1 map = 100%,  reduce = 87%, Cumulative CPU 25.35 sec
2018-08-12 13:57:07,777 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 33.58 sec
MapReduce Total cumulative CPU time: 33 seconds 580 msec
Ended Job = job_1534050476803_0006
Loading data to table custom.college
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 5   Cumulative CPU: 33.58 sec   HDFS Read: 26841 HDFS Write: 3995 SUCCESS
Total MapReduce CPU Time Spent: 33 seconds 580 msec
OK
```

Assignment 9.1
Advance Hive

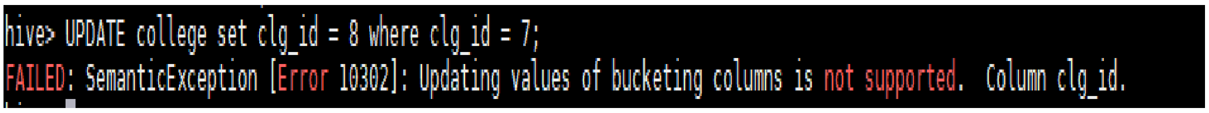
We could see data in college table below :



```
hive> select * from college;
OK
5       stanford        uk
5       stanford        uk
6       JNTUA   atp
1       nec     nlr
6       JNTUA   atp
1       nec     nlr
7       cambridge       us
2       vit     vlr
7       cambridge       us
2       vit     vlr
3       srm     chen
3       srm     chen
4       lpu     del
4       lpu     del
Time taken: 0.293 seconds, Fetched: 14 row(s)
hive>
```

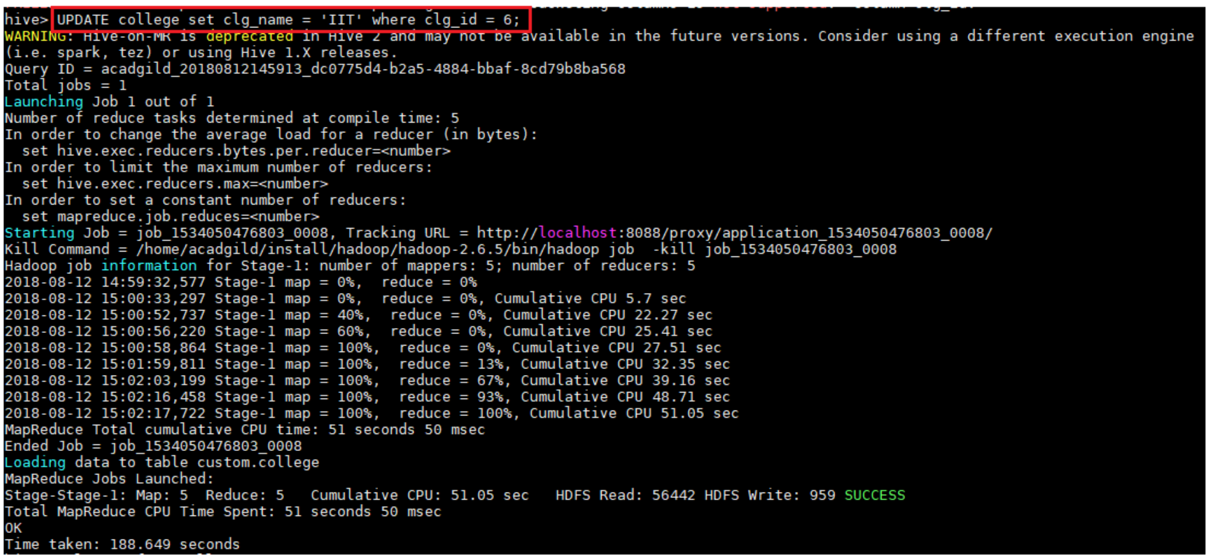
Below we are trying to update bucketed column ‘clg_id’. But we have received error.

So it means that we cannot update bucketed column.



```
hive> UPDATE college set clg_id = 8 where clg_id = 7;
FAILED: SemanticException [Error 10302]: Updating values of bucketing columns is not supported.  Column clg_id.
```

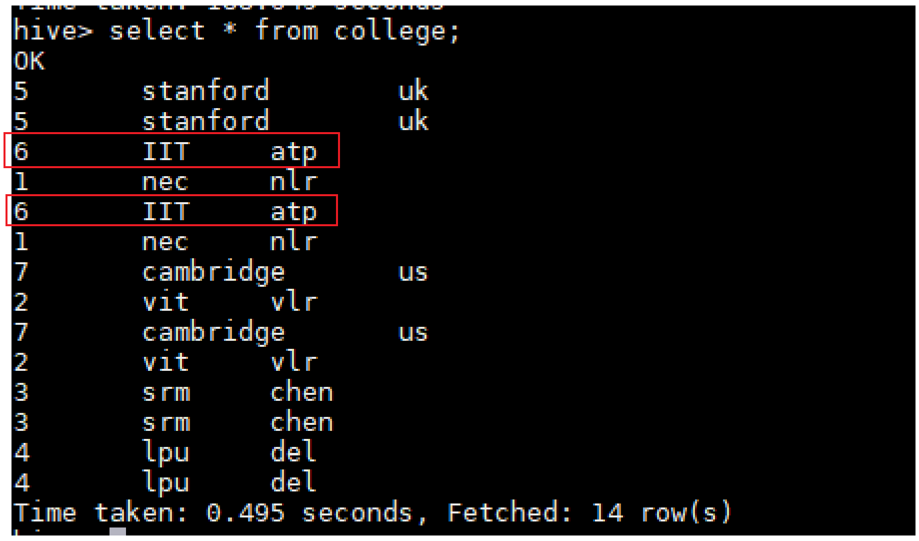
Below we have performed update on non-bucketed column ‘clg_name’ and it has been updated successfully.
This means that we can update non-bucketed column.



```
hive> UPDATE college set clg_name = 'IIT' where clg_id = 6;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine
(i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180812145913_dc0775d4-b2a5-4884-bbaf-8cd79b8ba568
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 5
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1534050476803_0008, Tracking URL = http://localhost:8088/proxy/application_1534050476803_0008/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1534050476803_0008
Hadoop job information for Stage-1: number of mappers: 5; number of reducers: 5
2018-08-12 14:59:32,577 Stage-1 map = 0%,  reduce = 0%
2018-08-12 15:00:33,297 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 5.7 sec
2018-08-12 15:00:52,737 Stage-1 map = 40%,  reduce = 0%, Cumulative CPU 22.27 sec
2018-08-12 15:00:56,220 Stage-1 map = 60%,  reduce = 0%, Cumulative CPU 25.41 sec
2018-08-12 15:00:58,864 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 27.51 sec
2018-08-12 15:01:59,811 Stage-1 map = 100%,  reduce = 13%, Cumulative CPU 32.35 sec
2018-08-12 15:02:03,199 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 39.16 sec
2018-08-12 15:02:16,458 Stage-1 map = 100%,  reduce = 93%, Cumulative CPU 48.71 sec
2018-08-12 15:02:17,722 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 51.05 sec
MapReduce Total cumulative CPU time: 51 seconds 50 msec
Ended Job = job_1534050476803_0008
Loading data to table custom.college
MapReduce Jobs Launched:
Stage-Stage-1: Map: 5  Reduce: 5   Cumulative CPU: 51.05 sec   HDFS Read: 56442 HDFS Write: 959 SUCCESS
Total MapReduce CPU Time Spent: 51 seconds 50 msec
OK
Time taken: 188.649 seconds
```

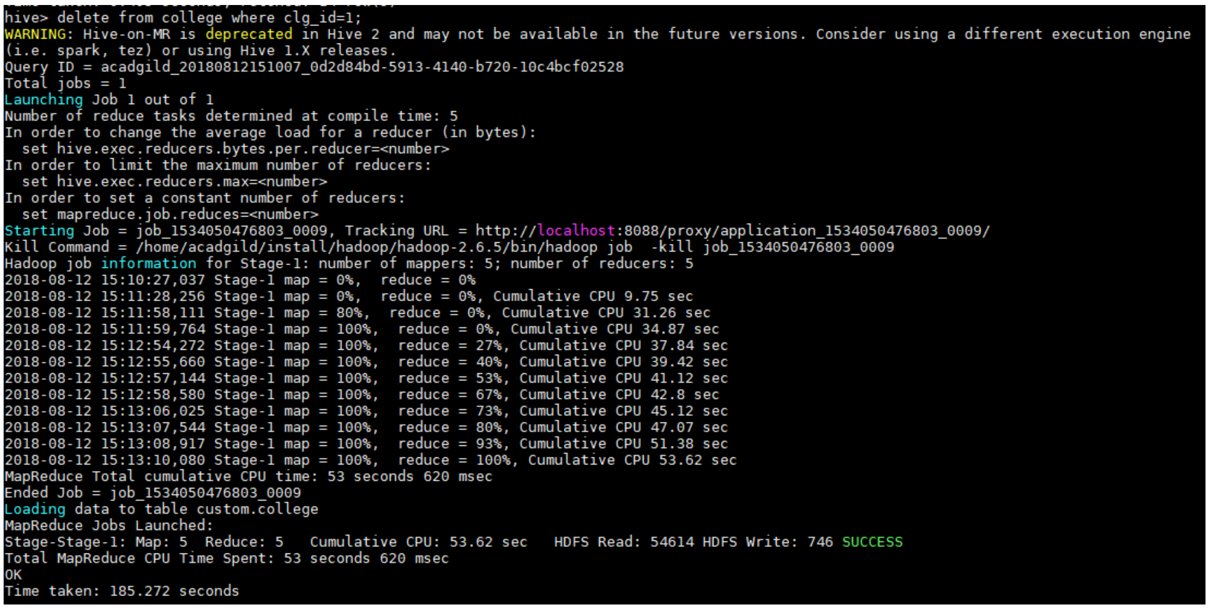
Assignment 9.1
Advance Hive

Below you could see that clg_name has been changed to **IIT** for clg_id =6



```
hive> select * from college;
OK
5       stanford        uk
5       stanford        uk
6       IIT     atp
1       nec     nlr
6       IIT     atp
1       nec     nlr
7       cambridge       us
2       vit     vlr
7       cambridge       us
2       vit     vlr
3       srm     chen
3       srm     chen
4       lpu     del
4       lpu     del
Time taken: 0.495 seconds, Fetched: 14 row(s)
```

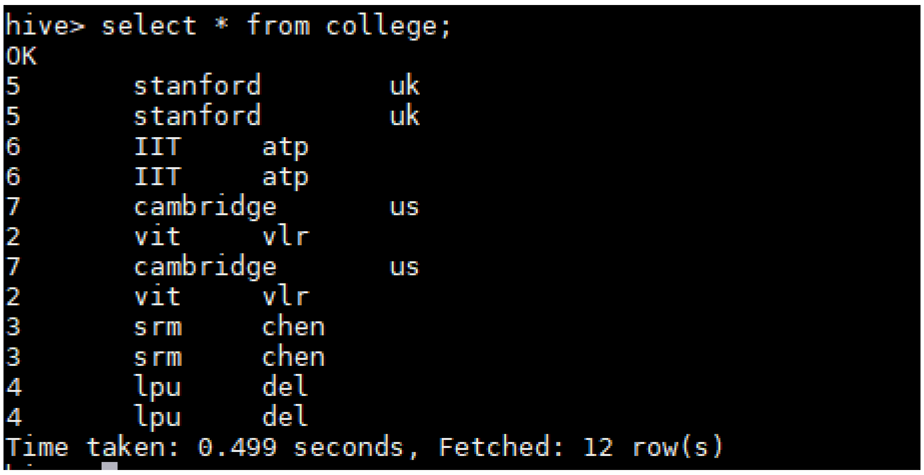
Below we have deleted data having clg_id = 1.



```
hive> delete from college where clg_id=1;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine
(i.e. spark, tez) or using Hive 1.X releases.
Query ID = acadgild_20180812151007_0d2d84bd-5913-4140-b720-10c4bcf02528
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 5
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1534050476803_0009, Tracking URL = http://localhost:8088/proxy/application_1534050476803_0009/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1534050476803_0009
Hadoop job information for Stage-1: number of mappers: 5; number of reducers: 5
2018-08-12 15:10:27,037 Stage-1 map = 0%,  reduce = 0%
2018-08-12 15:11:28,256 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 9.75 sec
2018-08-12 15:11:58,111 Stage-1 map = 80%,  reduce = 0%, Cumulative CPU 31.26 sec
2018-08-12 15:11:59,764 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 34.87 sec
2018-08-12 15:12:54,272 Stage-1 map = 100%,  reduce = 27%, Cumulative CPU 37.84 sec
2018-08-12 15:12:55,660 Stage-1 map = 100%,  reduce = 40%, Cumulative CPU 39.42 sec
2018-08-12 15:12:57,144 Stage-1 map = 100%,  reduce = 53%, Cumulative CPU 41.12 sec
2018-08-12 15:12:58,580 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 42.8 sec
2018-08-12 15:13:06,025 Stage-1 map = 100%,  reduce = 73%, Cumulative CPU 45.12 sec
2018-08-12 15:13:07,544 Stage-1 map = 100%,  reduce = 80%, Cumulative CPU 47.07 sec
2018-08-12 15:13:08,917 Stage-1 map = 100%,  reduce = 93%, Cumulative CPU 51.38 sec
2018-08-12 15:13:10,080 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 53.62 sec
MapReduce Total cumulative CPU time: 53 seconds 620 msec
Ended Job = job_1534050476803_0009
Loading data to table custom.college
MapReduce Jobs Launched:
Stage-Stage-1: Map: 5  Reduce: 5   Cumulative CPU: 53.62 sec   HDFS Read: 54614 HDFS Write: 746 SUCCESS
Total MapReduce CPU Time Spent: 53 seconds 620 msec
OK
Time taken: 185.272 seconds
```

Assignment 9.1
Advance Hive

We could see that clg_id having value as 1 has been deleted successfully from college table.



```
hive> select * from college;
OK
5       stanford        uk
5       stanford        uk
6       IIT     atp
6       IIT     atp
7       cambridge       us
2       vit     vlr
7       cambridge       us
2       vit     vlr
3       srm     chen
3       srm     chen
4       lpu     del
4       lpu     del
Time taken: 0.499 seconds, Fetched: 12 row(s)
```