Assignment 9

Task 1:

This Data set is about Olympics. You can download the data set from the below link:

https://drive.google.com/open?id=0ByJLBTmJojjzV1czX3Nha0R3bTQ

Create Table:

CREATE TABLE olympic

```
acadgild@localhost:~/hive
hive> CREATE TABLE olympic
    > (
    > athelete STRING,
    > age INT,
    > country STRING,
    > year STRING,
    > closing STRING,
    > sport STRING,
    > gold INT,
    > silver INT,
    > bronze INT,
    > total INT
    > )
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY '\t';
OK
Time taken: 3.33 seconds
```

Load the data into table:

LOAD DATA LOCAL INPATH '/home/acadgild/hive/olympix_data.csv'

INTO TABLE olympic;

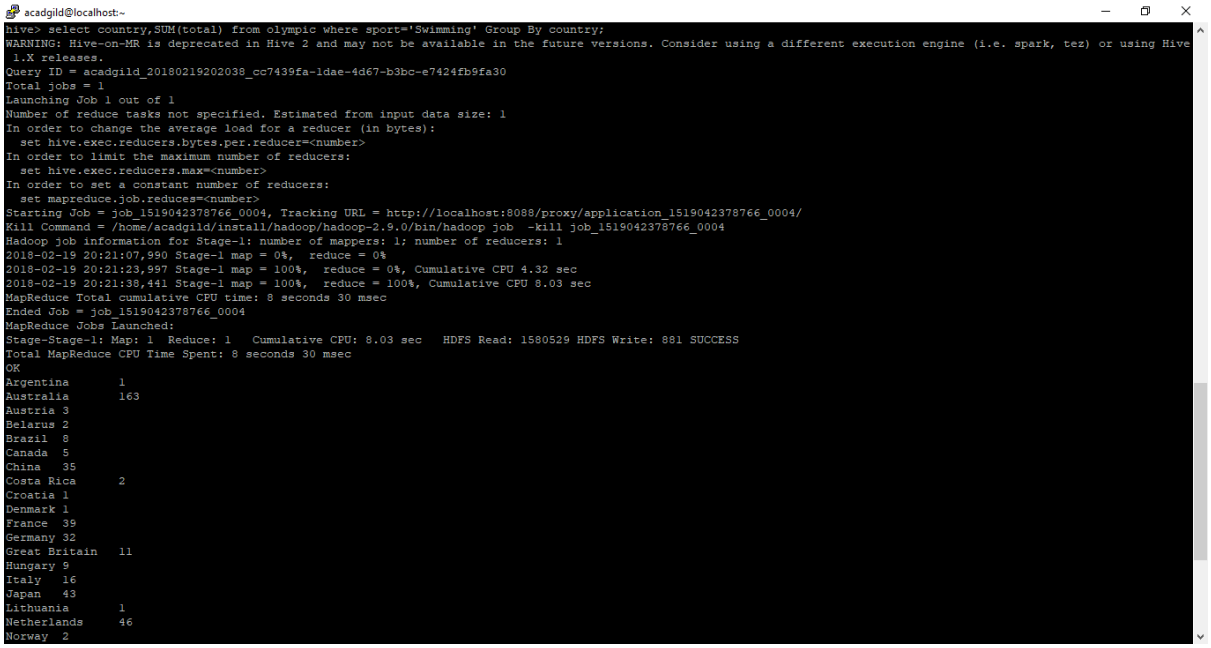
```
acadgild@localhost:~
hive> LOAD DATA LOCAL INPATH '/home/acadgild/hive/olympix_data.csv'
    > INTO TABLE olympic;
Loading data to table default.olympic
OK
Time taken: 3.876 seconds
hive>
```

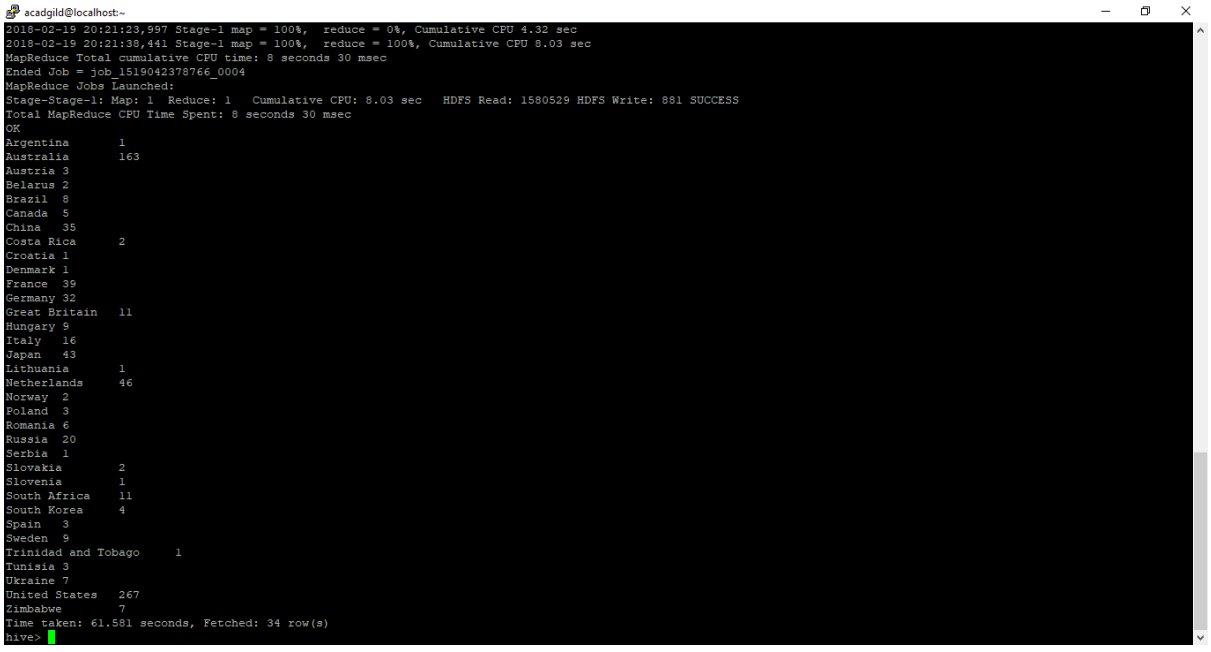
1.Write a Hive program to find the number of medals won by each country in swimming.

Query:

select country,SUM(total) from olympic where sport='Swimming' Group By country;

Output:



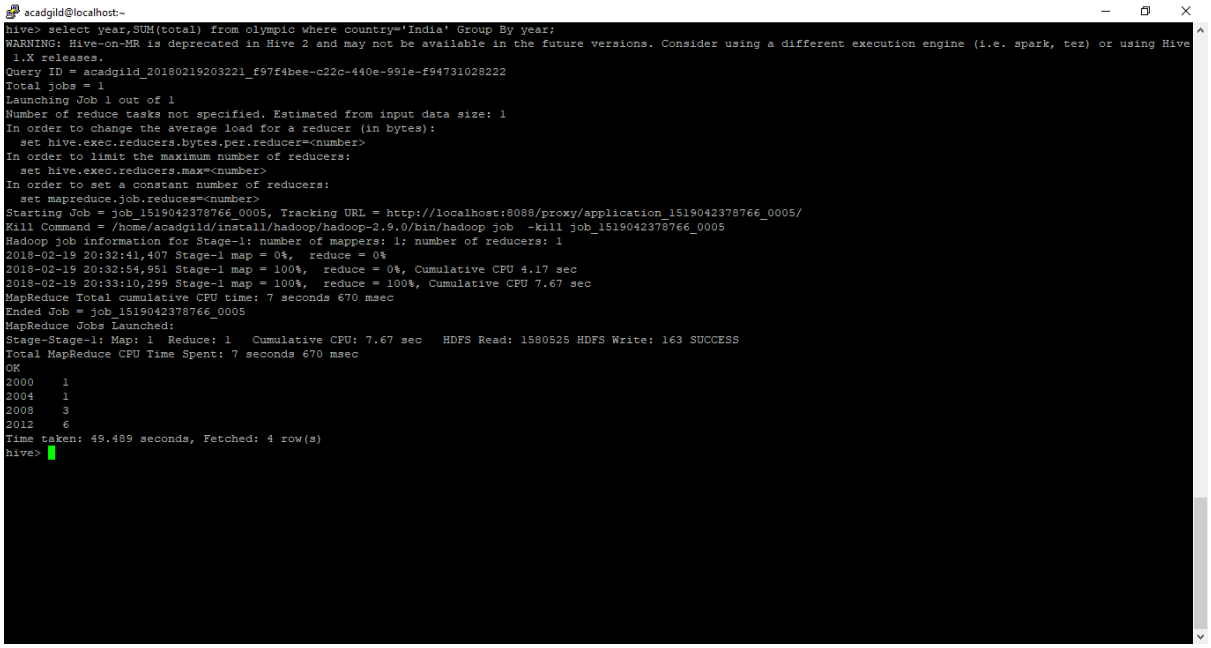


2. Write a Hive program to find the number of medals that India won year wise.

Query:

select year,SUM(total) from olympic where country='India' Group By year;

Output:

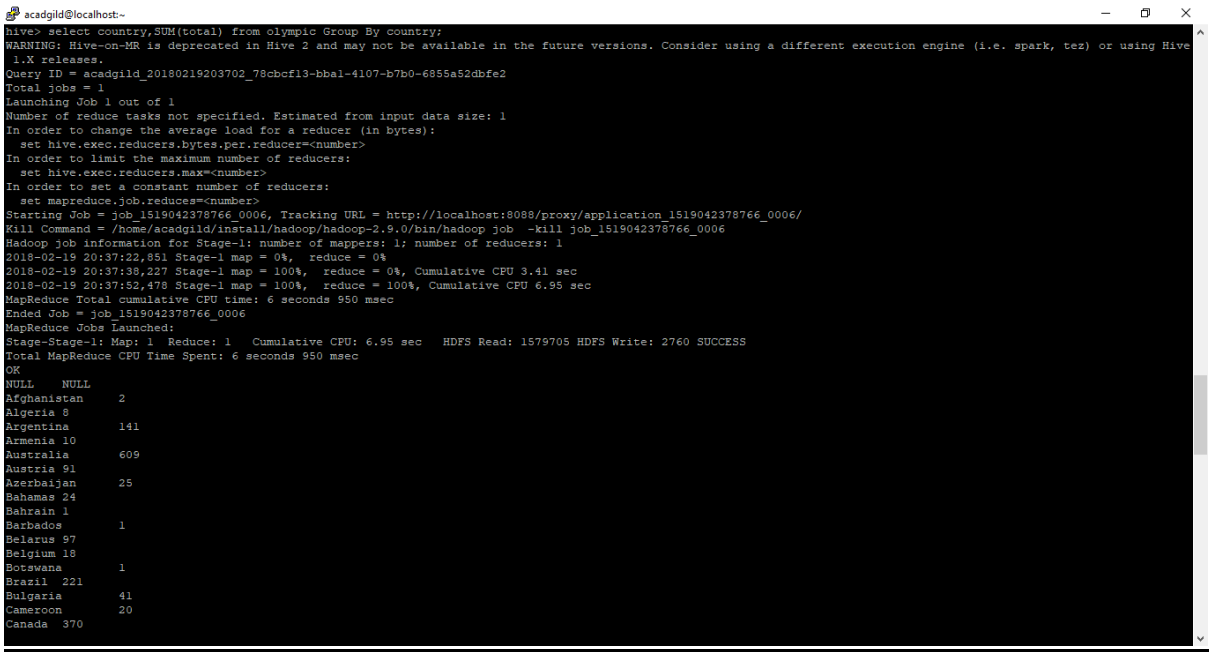


3. Write a Hive Program to find the total number of medals each country won.

Query:

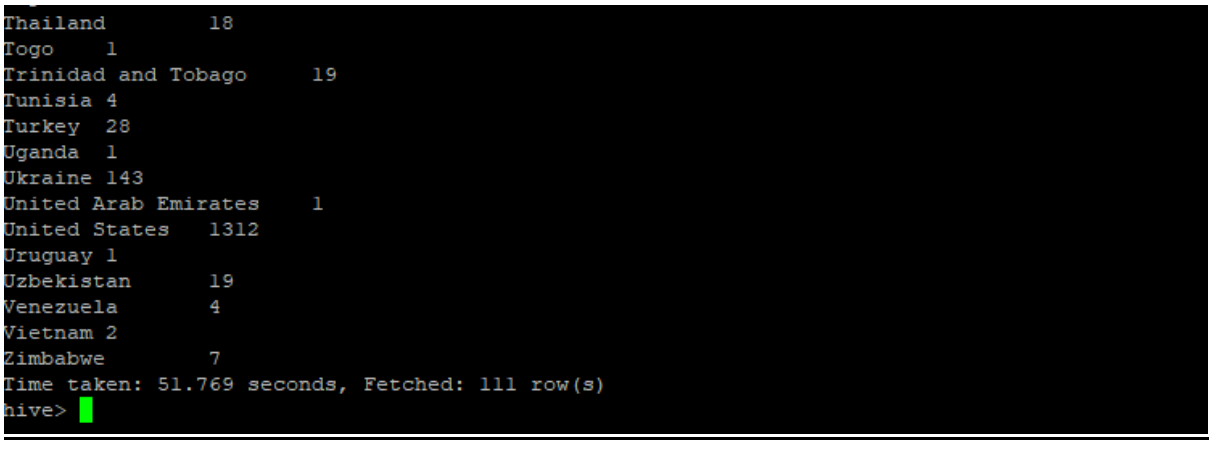
select country,SUM(total) from olympic Group By country;

Output:




acadgild@localhost:~
Canada 370
Chile 22
China 530
Chinese Taipei 20
Colombia 13
Costa Rica 2
Croatia 81
Cuba 188
Cyprus 1
Czech Republic 81
Denmark 89
Dominican Republic 5
Ecuador 1
Egypt 8
Eritrea 1
Estonia 18
Ethiopia 29
Finland 118
France 318
Gabon 1
Georgia 23
Germany 629
Great Britain 322
Greece 59
Grenada 1
Guatemala 1
Hong Kong 3
Hungary 145
Iceland 15
India 11
Indonesia 22
Iran 24
Ireland 9
Israel 4
Italy 331
Jamaica 80
Japan 282
Kazakhstan 42
Kenya 39
Kuwait 2
Kyrgyzstan 3
Latvia 17
Lithuania 30


acadgild@localhost:~
Latvia 17
Lithuania 30
Macedonia 1
Malaysia 3
Mauritius 1
Mexico 38
Moldova 5
Mongolia 10
Montenegro 14
Morocco 11
Mozambique 1
Netherlands 318
New Zealand 52
Nigeria 39
North Korea 21
Norway 192
Panama 1
Paraguay 17
Poland 80
Portugal 9
Puerto Rico 2
Qatar 3
Romania 123
Russia 768
Saudi Arabia 6
Serbia 31
Serbia and Montenegro 38
Singapore 7
Slovakia 35
Slovenia 25
South Africa 25
South Korea 308
Spain 205
Sri Lanka 1
Sudan 1
Sweden 181
Switzerland 93
Syria 1
Tajikistan 3
Thailand 18
Togo 1
Trinidad and Tobago 19
Tunisia 4

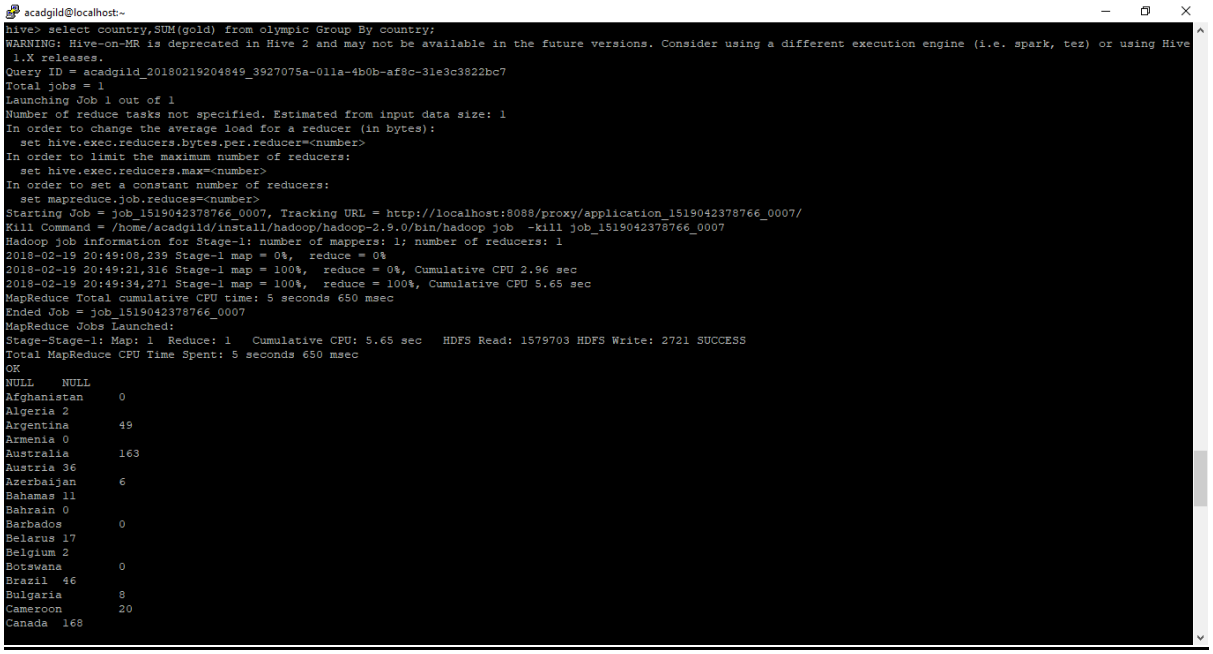

Thailand 18
Togo 1
Trinidad and Tobago 19
Tunisia 4
Turkey 28
Uganda 1
Ukraine 143
United Arab Emirates 1
United States 1312
Uruguay 1
Uzbekistan 19
Venezuela 4
Vietnam 2
Zimbabwe 7
Time taken: 51.769 seconds, Fetched: 111 row(s)
hive>

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

Query:

select country,SUM(gold) from olympic Group By country;

Output:



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 SEP.

Solution:

We have companies list and its company website URL, but the ‘www’ and the remaining domain are separated. In our output we try to achieve the output as below,

<u>Dataset:</u>

Cat > comp.txt

acadgild@localhost:~

```
login as: acadgild
acadgild@192.168.85.129's password:
Last login: Sun Feb 18 22:16:37 2018 from 192.168.85.1
[acadgild@localhost ~]$ cat > comp.txt
1,wipro,www,wipro.com
2,infosys,www,infosys.com
3,google,www,google.com
4,apple,www,apple.com
5,walmart,www,walmart.com
^C
[acadgild@localhost ~]$ cat comp.txt
1,wipro,www,wipro.com
2,infosys,www,infosys.com
3,google,www,google.com
4,apple,www,apple.com
5,walmart,www,walmart.com
[acadgild@localhost ~]$
```

<u>Create Database:</u>

Create database company;

acadgild@localhost:~

```
hive> create database company;
Sun Feb 18 22:24:57 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+
 requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertif
icate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate ve
rification.
Sun Feb 18 22:24:58 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+
 requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertif
icate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate ve
rification.
Sun Feb 18 22:24:59 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+
 requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertif
icate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate ve
rification.
Sun Feb 18 22:24:59 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+
 requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertif
icate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate ve
rification.
Sun Feb 18 22:25:05 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+
 requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertif
icate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate ve
rification.
Sun Feb 18 22:25:05 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+
 requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertif
icate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate ve
rification.
Sun Feb 18 22:25:05 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+
 requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertif
icate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate ve
rification.
Sun Feb 18 22:25:05 IST 2018 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+
 requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SSL the verifyServerCertif
icate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate ve
rification.
OK
Time taken: 14.75 seconds
hive>
```

Show databases;

Use company;

acadgild@localhost:~

```
hive> show databases;
OK
advanced
company
custom
default
fortune
fortunel
olympics
retail
Time taken: 0.354 seconds, Fetched: 8 row(s)
hive> use company;
OK
Time taken: 0.073 seconds
hive>
```

Create Table and Load the data:

-> CREATE TABLE company1(rank int,company_name string,website string,protocal string)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’;

-> LOAD DATA LOCAL INPATH ‘/home/acadgild/comp.txt’

INTO TABLE company1;

acadgild@localhost:~

```
hive> CREATE TABLE company1(rank int,company_name string,website string,protocal string)
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
OK
Time taken: 1.585 seconds
hive> LOAD DATA LOCAL INPATH '/home/acadgild/comp.txt'
    > INTO TABLE company1;
Loading data to table company.company1
OK
Time taken: 2.029 seconds
hive>
```

View the data in the table company1:

Select * from company1;

acadgild@localhost:~

```
hive> SELECT * FROM company1;
OK
1       wipro   www     wipro.com
2       infosys www     infosys.com
3       google  www     google.com
4       apple   www     apple.com
5       walmart www     walmart.com
Time taken: 2.926 seconds, Fetched: 5 row(s)
hive>
```

HIVE UDF java code:

```
package concatws;

import org.apache.hadoop.hive.ql.exec.UDF;
import org.apache.hadoop.hive.ql.exec.Description;
@Description(name = "concatws", value = "_FUNC_(string SEP, array<string>) -
RETURN_TYPE(STRING)\n" + "Description: Concatenate two strings, separated by the
seperator",
extended = "Example:\n"
             + "  > SELECT CONCAT_WS (website,'.',protocal) FROM src;\n"
             + "www.walmart.com")

public class hiveUDF extends UDF

{
       public String evaluate(String param1, String[] param2)

{
       String Output = "";
       if(param1==null && param2==null)
       {
              return null;
       }
       for(int i = 0; i < param2.length; i++)
       {
              Output+= param2[i];
       }
       return(param1.concat(Output));
}
}
```

After that we are adding JAR created from the JAVA class which is defining the UDF using below syntax-

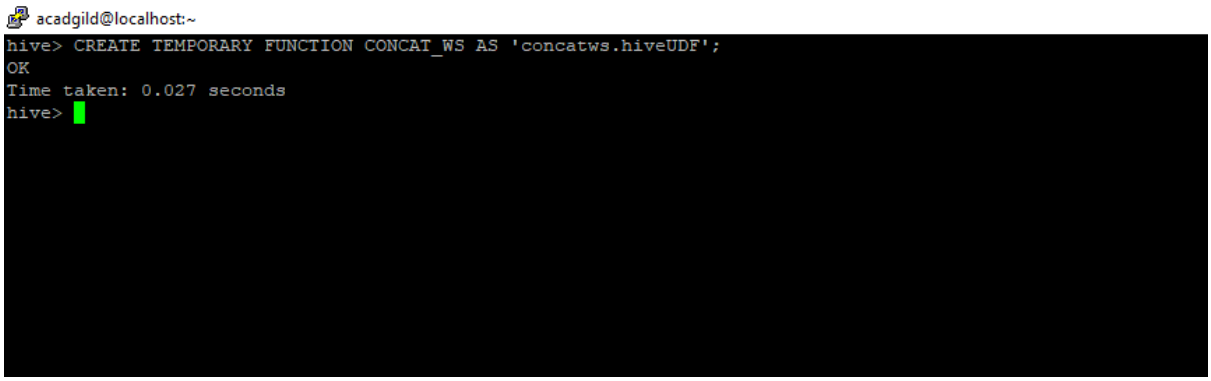
Adding the jar function:

Add jar /home/acadgild/udf.jar;

```
hive> add jar /home/acadgild/udf.jar;
Added [/home/acadgild/udf.jar] to class path
Added resources: [/home/acadgild/udf.jar]
hive>
```

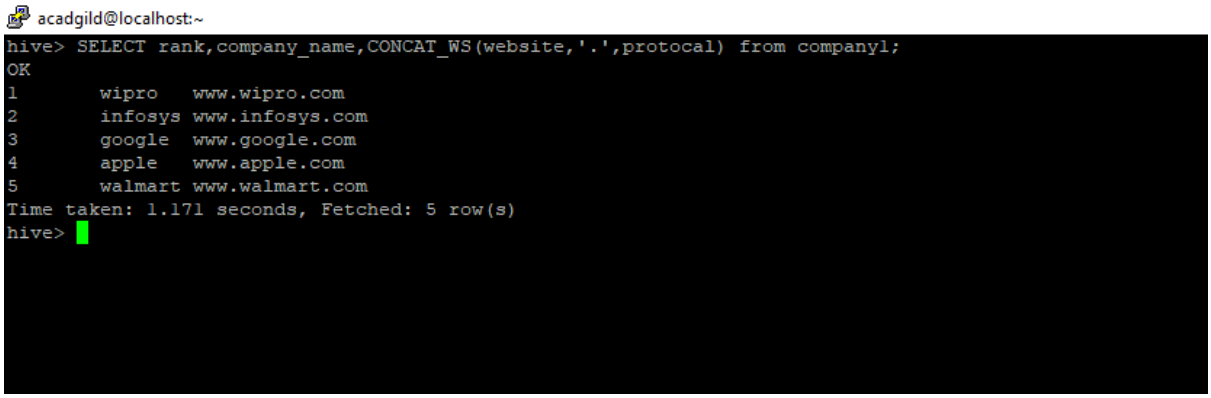
creating a temporary function “CONCAT_WS”

CREATE TEMPORARY FUNCTION CONCAT_WS AS 'concatws.hiveUDF ';



After that we run below query to take one column (company_name) input as String and another array(website,'.',protocal) as Array of Strings and concatenate them,

Select rank,company_name,CONCAT_WS(website,'.',protocal) from company1;



Task3:

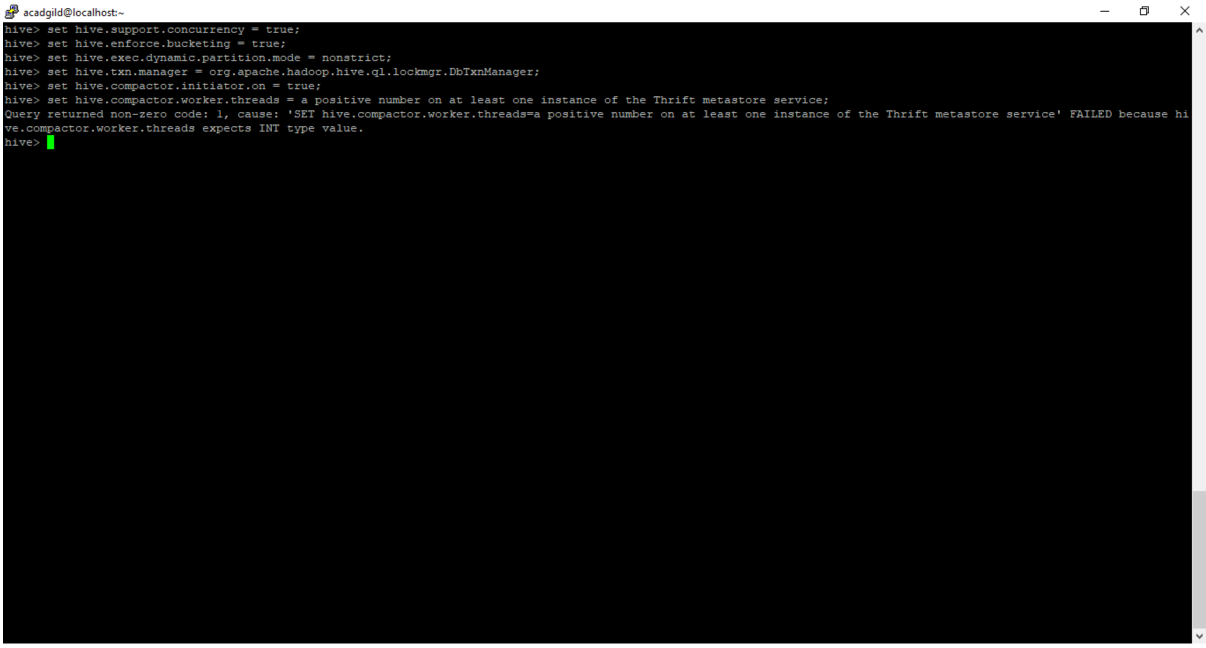
Link: https://acadgild.com/blog/transactions-in-hive/

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.

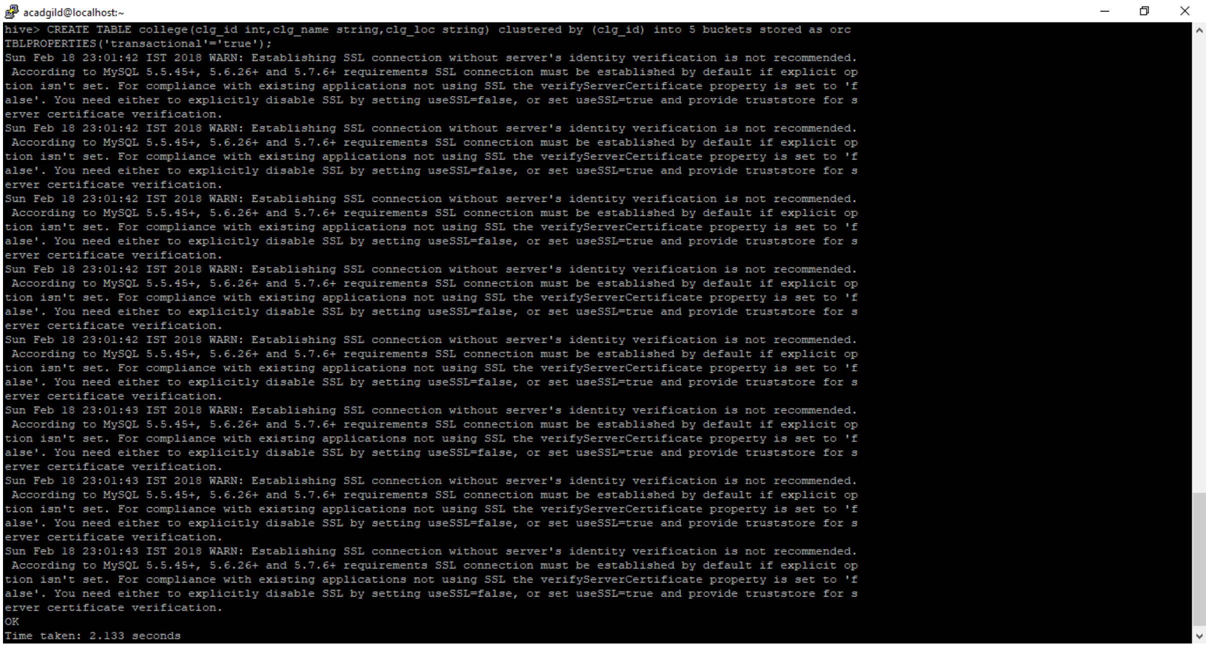
Transactions are provided at the row-level in Hive 0.14. The different row-level transactions available in Hive 0.14 are as follows:

1. Insert
2. Delete
3. Update



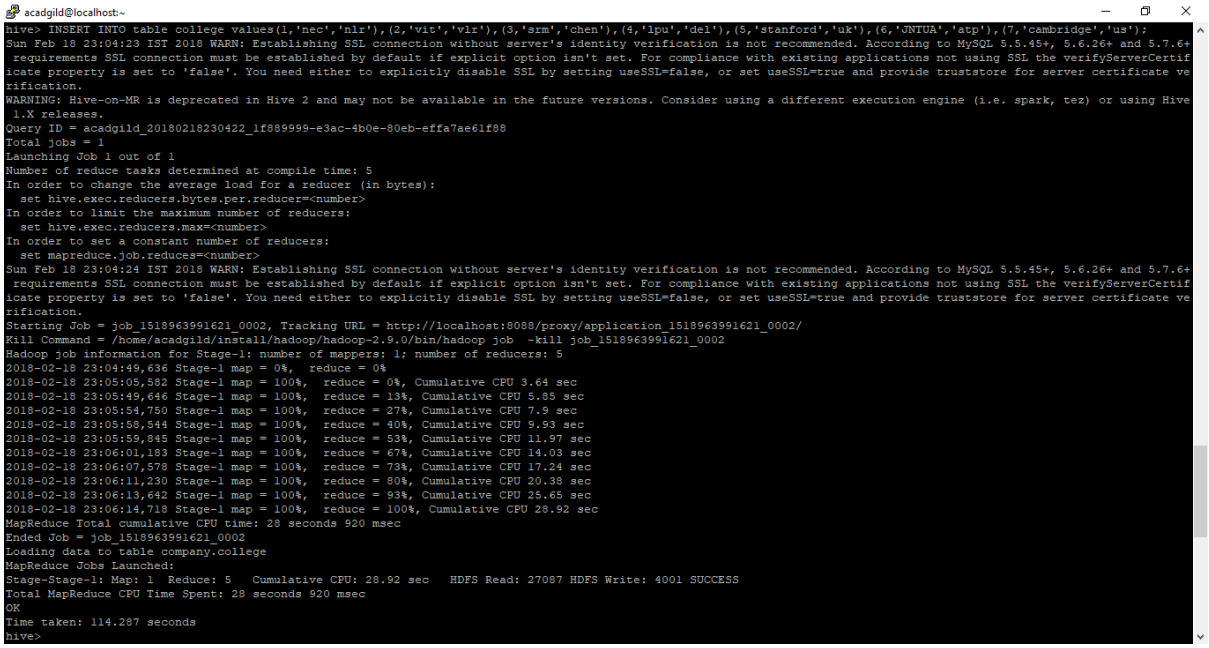
Creating a Table That Supports Hive Transactions

CREATE TABLE college(clg_id int,clg_name string,clg_loc string) clustered by (clg_id) into 5 buckets stored as orc TBLPROPERTIES('transactional'='true');



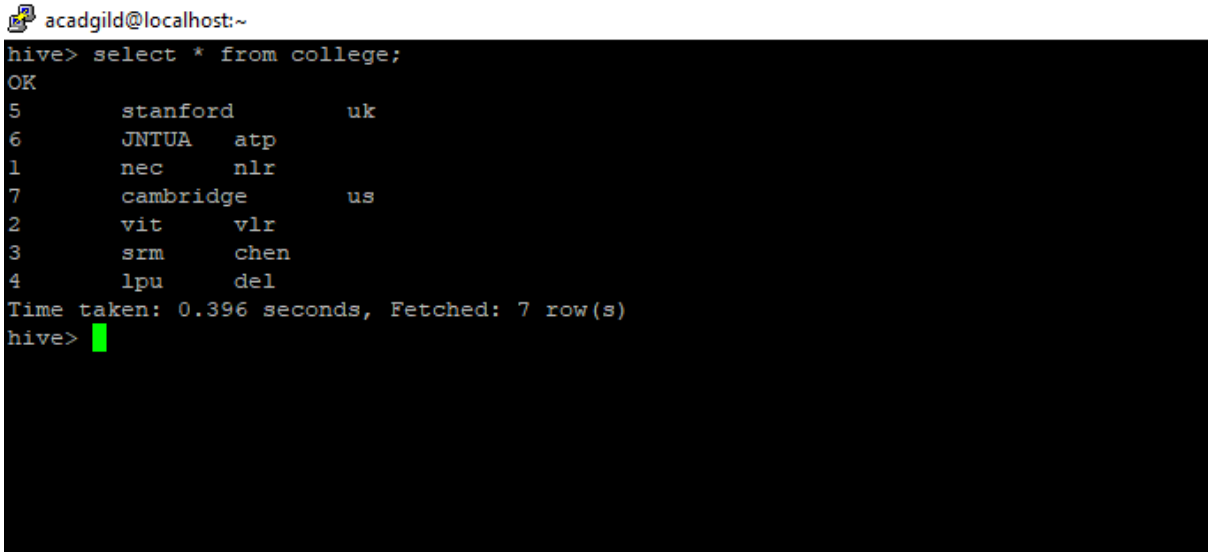
Inserting Data into a Hive Table

INSERT INTO table college values(1,'nec','nlr'),(2,'vit','vlr'),(3,'srm','chen'),(4,'lpu','del'),(5,'stanford','uk'),(6,'JNTUA','atp'),(7,'cambridge','us');



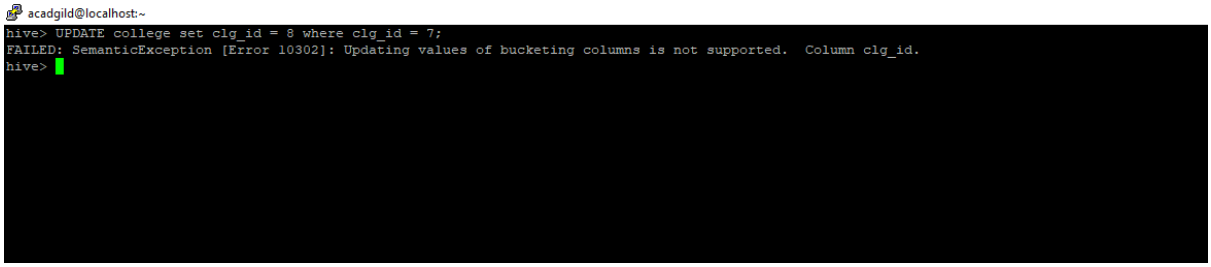
View the data:

select * from college

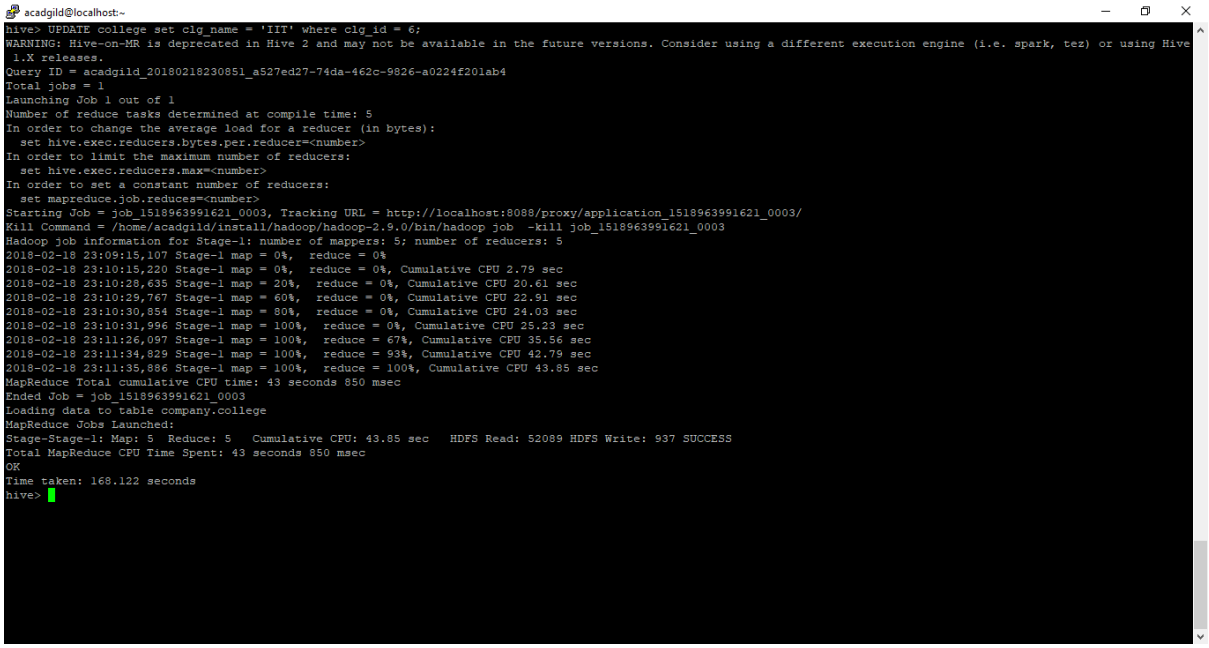


Updating the Data in Hive Table

UPDATE college set clg_id = 8 where clg_id = 7;

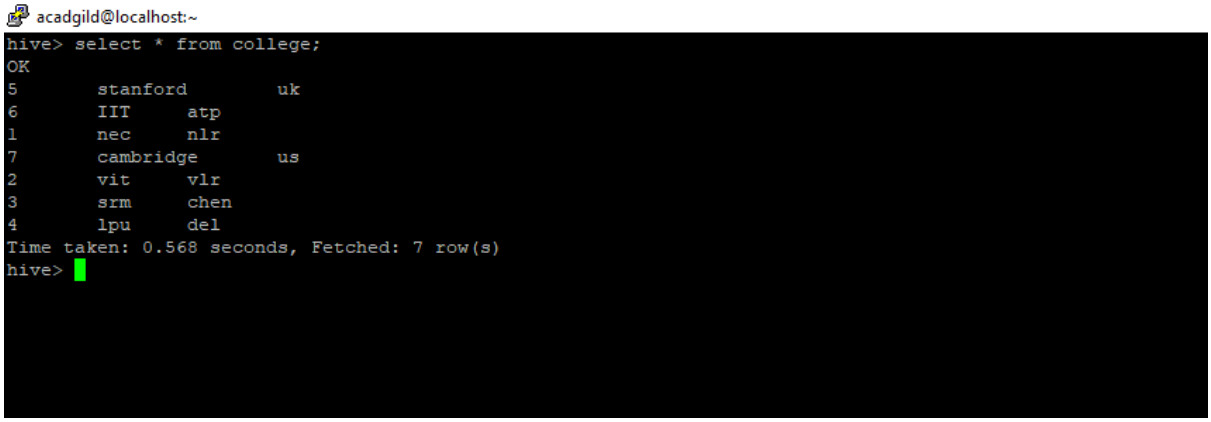


UPDATE college set clg_name = 'IIT' where clg_id = 6;



View the data:

Select *from college;



Deleting a Row from Hive Table

Delete from college where clg_id=5;

```
acadgild@localhost:~
hive> delete from college where clg_id=5;
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_20180218231739_66c6aa1f-540f-48f5-8a84-0d80768dd68c
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_1518963991621_0004, Tracking URL = http://localhost:8088/proxy/application_1518963991621_0004/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.9.0/bin/hadoop job  -kill job_1518963991621_0004
Hadoop job information for Stage-1: number of mappers: 5; number of reducers: 5
2018-02-18 23:17:59,758 Stage-1 map = 0%,  reduce = 0%
2018-02-18 23:18:59,990 Stage-1 map = 20%,  reduce = 0%, Cumulative CPU 7.7 sec
2018-02-18 23:19:05,117 Stage-1 map = 40%,  reduce = 0%, Cumulative CPU 12.69 sec
2018-02-18 23:19:06,414 Stage-1 map = 60%,  reduce = 0%, Cumulative CPU 16.47 sec
2018-02-18 23:19:09,032 Stage-1 map = 80%,  reduce = 0%, Cumulative CPU 20.63 sec
2018-02-18 23:19:10,352 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 21.11 sec
2018-02-18 23:20:01,710 Stage-1 map = 100%,  reduce = 13%, Cumulative CPU 23.38 sec
2018-02-18 23:20:04,262 Stage-1 map = 100%,  reduce = 53%, Cumulative CPU 29.55 sec
2018-02-18 23:20:09,209 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 31.73 sec
2018-02-18 23:20:14,246 Stage-1 map = 100%,  reduce = 93%, Cumulative CPU 38.86 sec
2018-02-18 23:20:15,440 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 40.64 sec
MapReduce Total cumulative CPU time: 40 seconds 640 msec
Ended Job = job_1518963991621_0004
Loading data to table company.college
MapReduce Jobs Launched:
Stage-Stage-1: Map: 5  Reduce: 5   Cumulative CPU: 40.64 sec   HDFS Read: 50313 HDFS Write: 752 SUCCESS
Total MapReduce CPU Time Spent: 40 seconds 640 msec
OK
Time taken: 159.596 seconds
hive>
```

View the data:

Select * from college;

```
acadgild@localhost:~
hive> select * from college;
OK
6       IIT     atp
1       nec     nlr
7       cambridge       us
2       vit     vlr
3       srm     chen
4       lpu     del
Time taken: 0.386 seconds, Fetched: 6 row(s)
hive>
```