BIG DATA HADOOP AND SPARK DEVLOPMENT

ASSIGNMENT 9

Table of Contents:

BIG DATA HADOOPAND SPARK DEVELOPMENT

1. Introduction

In this assignment, the given tasks are performed and Output of the tasks are recorded in the form of Screenshots.

2. Objective

This Assignment consolidates the deeper understanding of the Session – 9 Advanced concepts of HIVE

3. Associated Data Files

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

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

DATE SET DESCRIPTION

The data set consists of the following fields.

Athlete: This field consists of the athlete name

Age: This field consists of athlete ages

Country: This fields consists of the country names which participated in Olympics

Year: This field consists of the year

Closing Date: This field consists of the closing date of ceremony

Sport: Consists of the sports name

Gold Medals: No. of Gold medals

Silver Medals: No. of Silver medals

Bronze Medals: No. of Bronze medals

Total Medals: Consists of total no. of medals

4. Problem Statement

- ### Task 1

    1. Write a Hive program to find the number of medals won by each country in swimming.
    2. Write a Hive program to find the number of medals that India won year wise.
    3. Write a Hive Program to find the total number of medals each country won.
    4. Write a Hive program to find the number of gold medals each country won.

- ### 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.

- ### Task 3

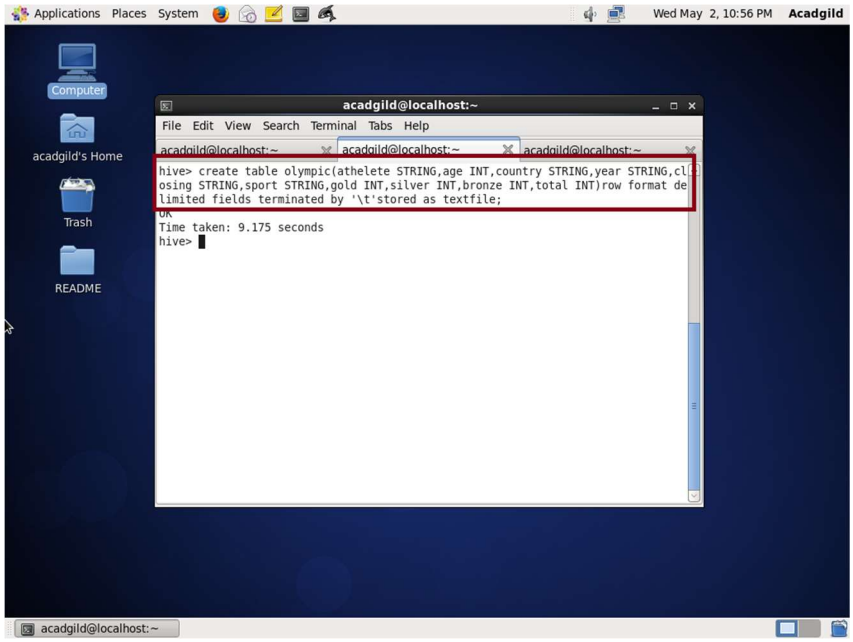
    Link: https://acadgild.com/blog/transactions-in-hive/

    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.

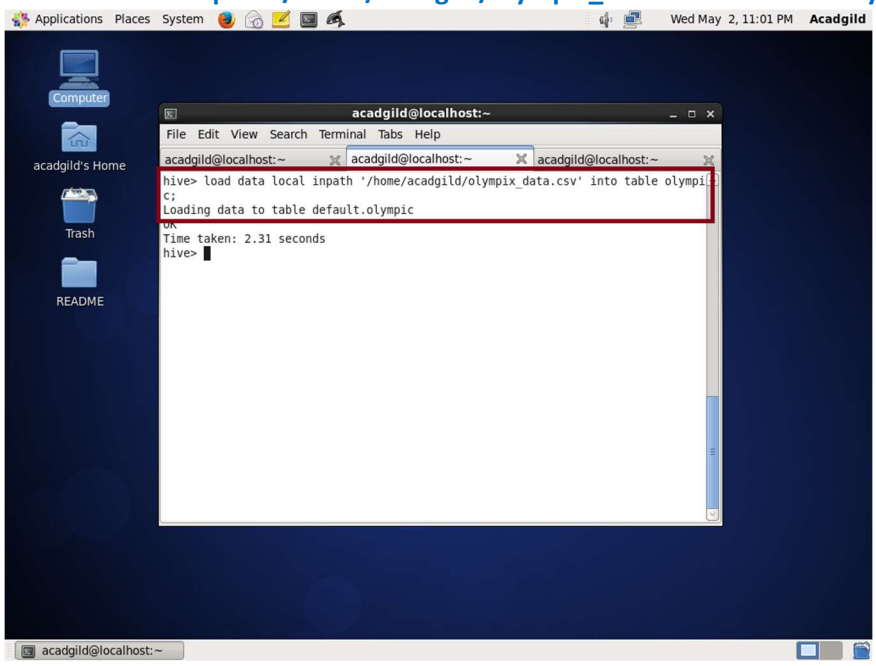
5. Expected Output

For performing the task 1, the Olympic table structure is created and the data is also loaded to the Olympic table.
Table Olympic is created by the following command

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' stored as textfile;



load data local inpath '/home/acadgild/olympix_data.csv' into table olympic;



- Task 1

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

GROUP BY clause in a SELECT statement:

The GROUP BY clause is used to group all the records in a result set using a particular collection column. It is used to query a group of records.

Syntax

The syntax of GROUP BY clause is as follows:

```
SELECT [ALL | DISTINCT] select_expr, select_expr, ...
FROM table_reference
[WHERE where_condition]
[GROUP BY col_list]
[HAVING having_condition]
[ORDER BY col_list]]
[LIMIT number];
```

Hive supports the following built-in **aggregate functions**.

count(*), count(expr)

count(*) - Returns the total number of retrieved rows.

sum(col), sum(DISTINCT col)

It returns the sum of the elements in the group or the sum of the distinct values of the column in the group.

avg(col), avg(DISTINCT col)

It returns the average of the elements in the group or the average of the distinct values of the column in the group.

min(col)

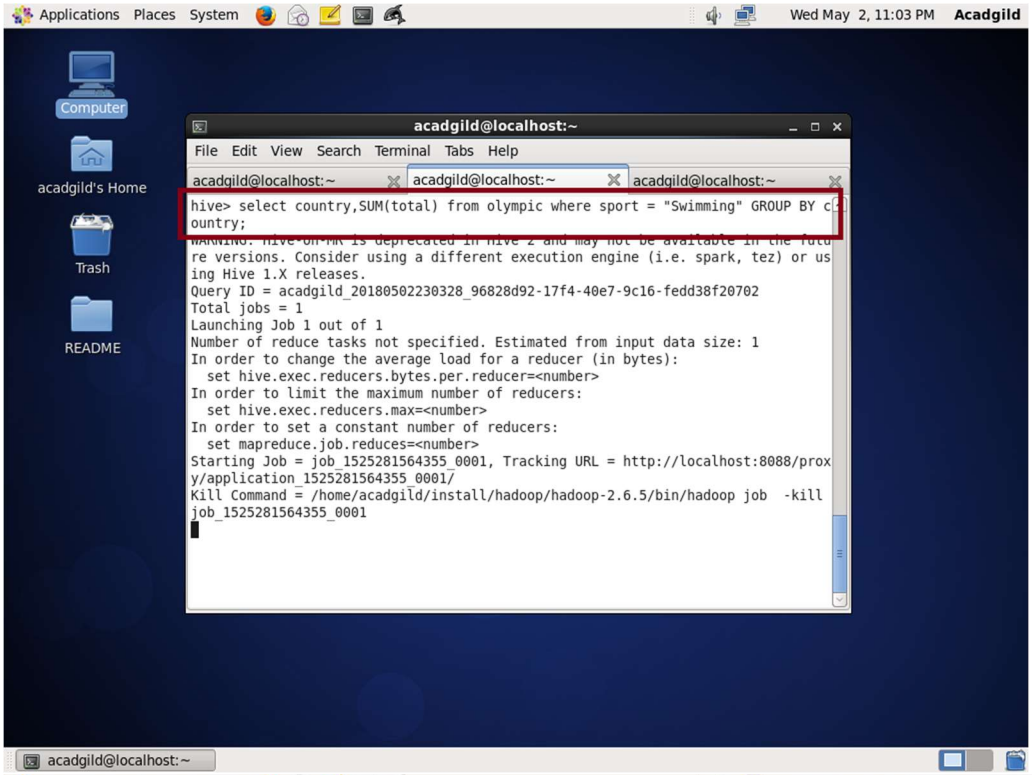
It returns the minimum value of the column in the group.

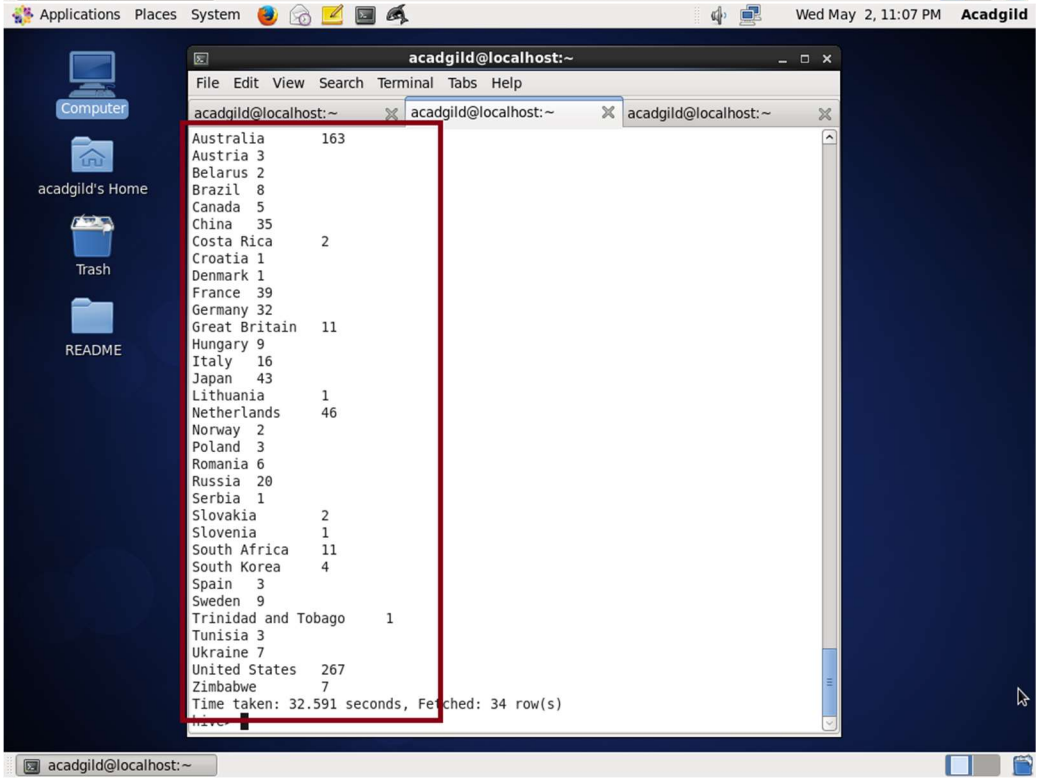
max(col)

It returns the maximum value of the column in the group.

By using the following command, the number of medals won by each country in swimming.

select country, SUM(total) from Olympic where sport = “Swimming” GROUP BY country;



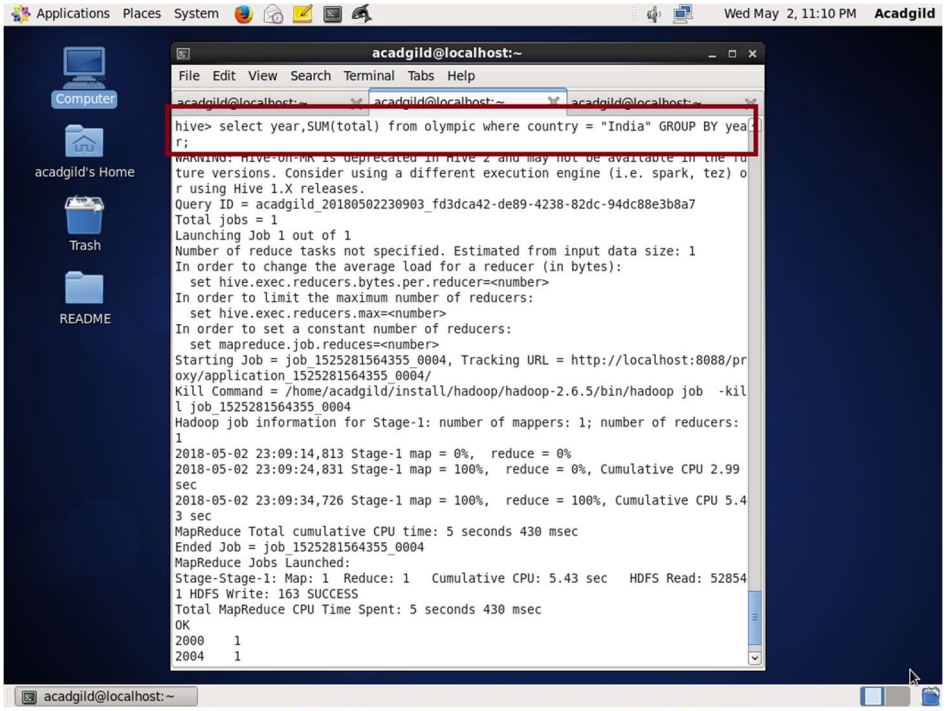


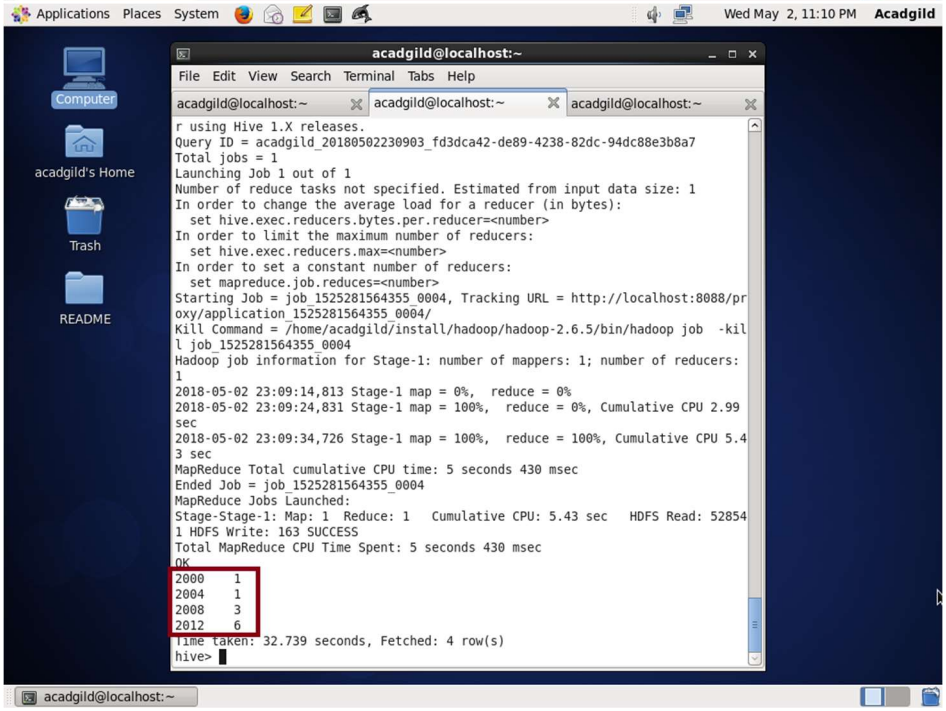
- **Write a Hive program to find the number of medals that India won year wise.**

By select query the year and sum of total from Olympic table where country is India.

By using the following command, the number of medals won by India are displayed with year.

select year, SUM(total) from olympic where country = “India” GROUP BY year;



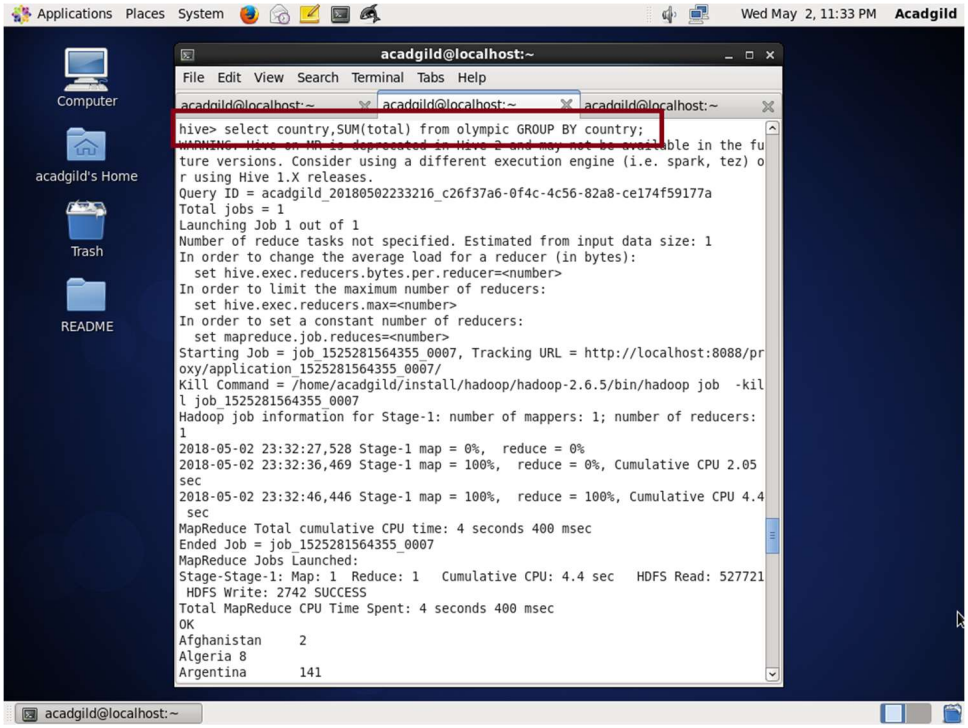


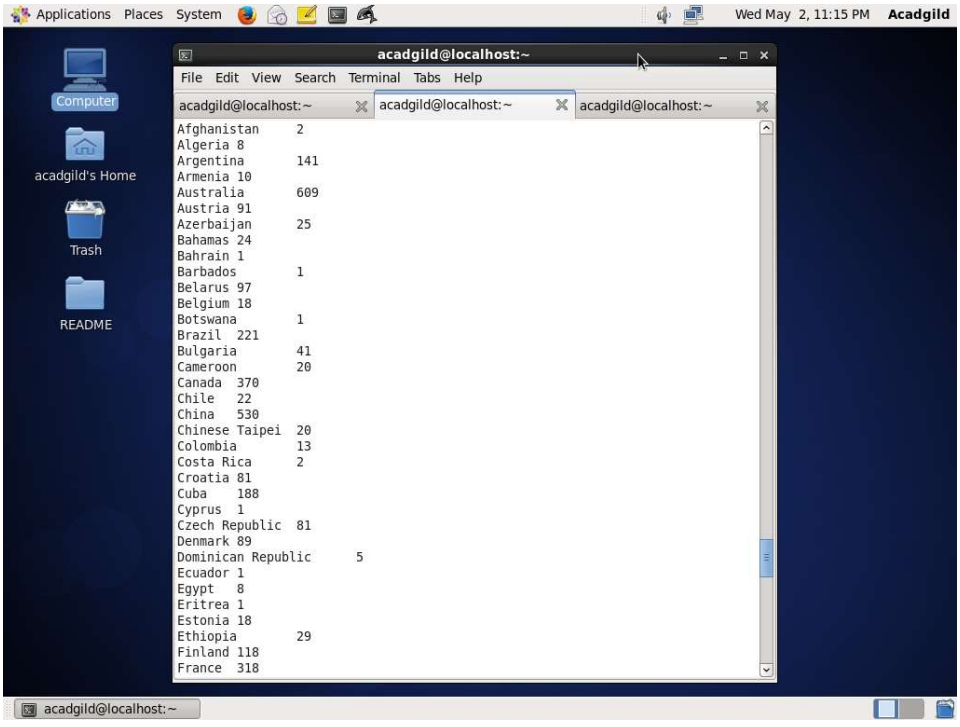
- **Write a Hive Program to find the total number of medals each country won.**

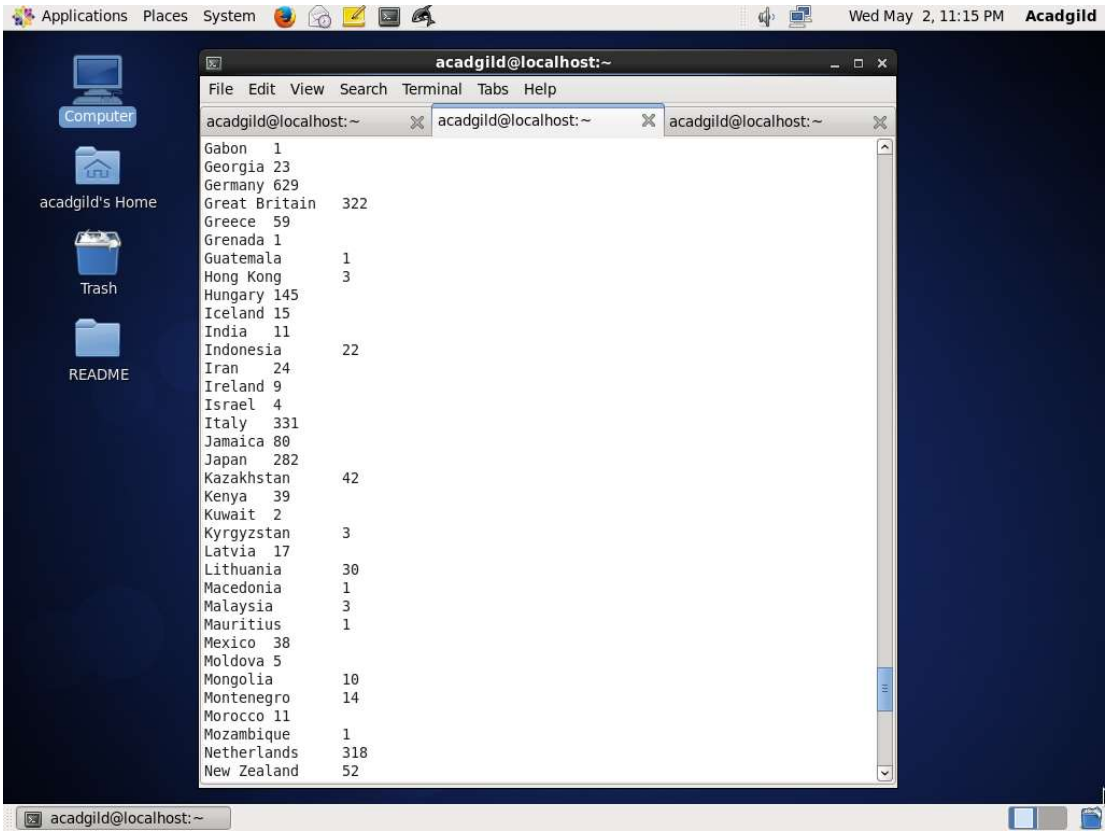
By select query the country and sum of total from Olympic table.

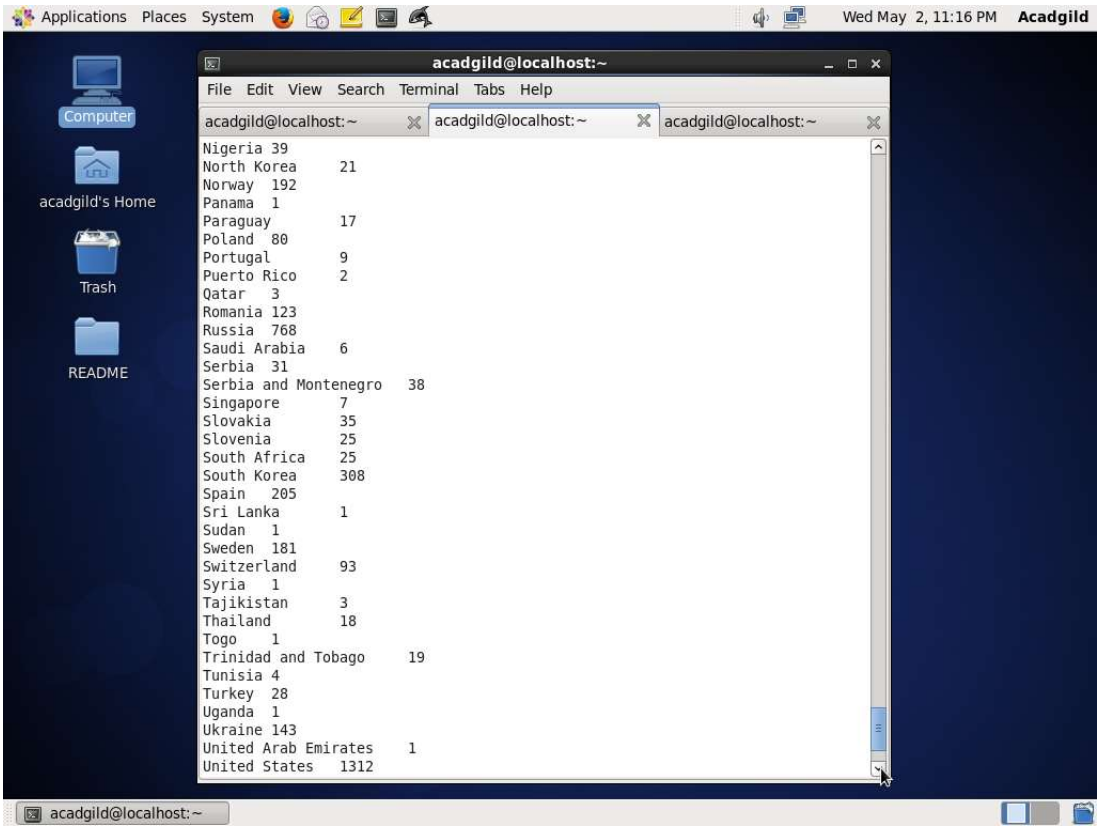
By using the following command, the total number of medals won by each country are displayed.

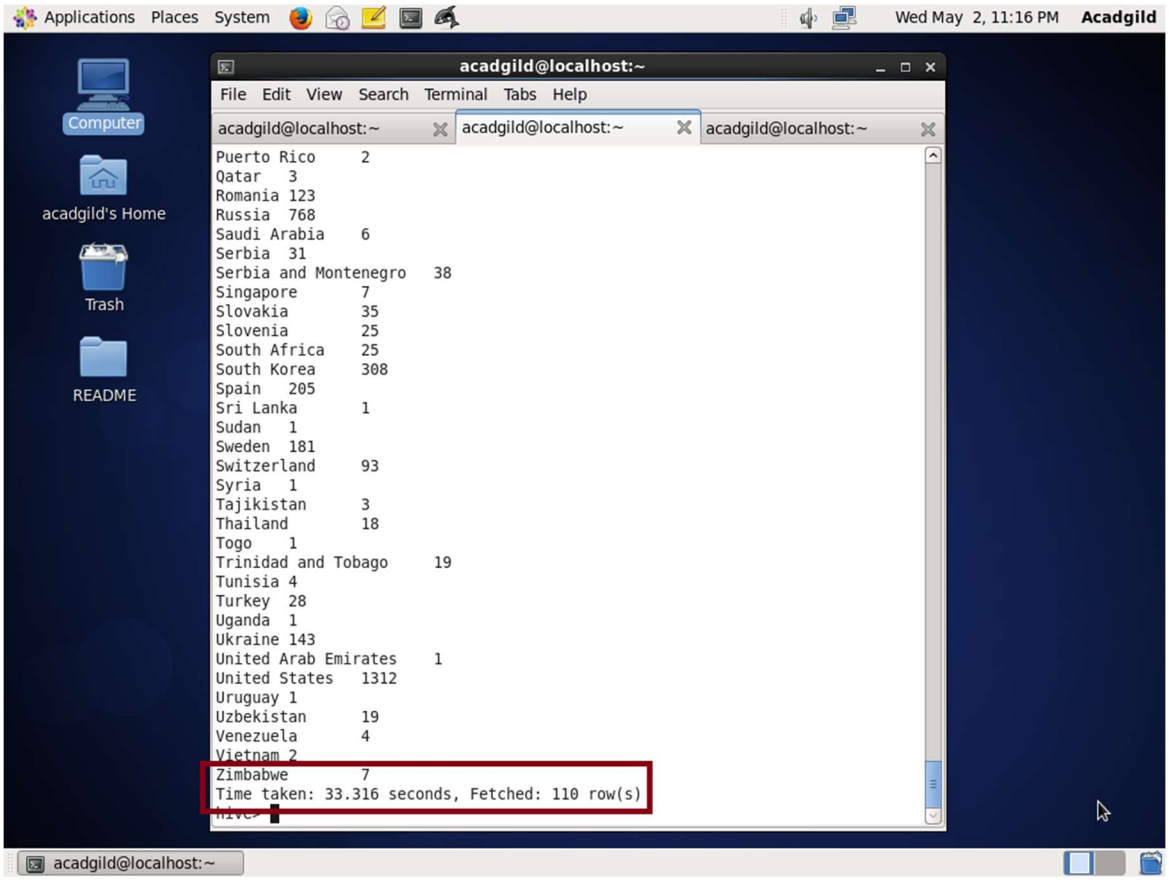
select country, SUM(total) from olympic GROUP BY country;






Applications Places System
Wed May 2, 11:15 PM Acadgild
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
acadgild@localhost:~
acadgild@localhost:~
Computer
acadgild's Home
Trash
README
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
Macedonia 1
Malaysia 3
Mauritius 1
Mexico 38
Moldova 5
Mongolia 10
Montenegro 14
Morocco 11
Mozambique 1
Netherlands 318
New Zealand 52
acadgild@localhost:~


Applications Places System
Wed May 2, 11:16 PM Acadgild
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
acadgild@localhost:~
acadgild@localhost:~
Computer
acadgild's Home
Trash
README
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
Turkey 28
Uganda 1
Ukraine 143
United Arab Emirates 1
United States 1312
acadgild@localhost:~

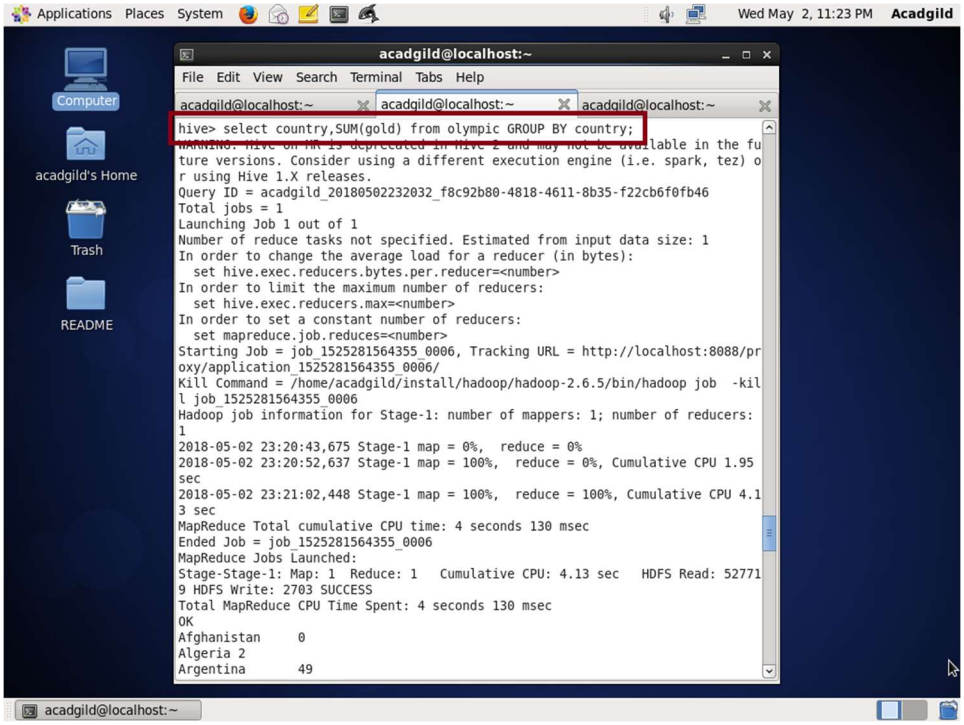

Applications Places System
Wed May 2, 11:16 PM Acadgild
Computer
acadgild's Home
Trash
README
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
acadgild@localhost:~
acadgild@localhost:~
Puerto Rico 2
Qatar 3
Romania 123
Russia 768
Saudi Arabia 6
Serbia 31
Serbia and Montenegro 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
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: 33.316 seconds, Fetched: 110 row(s)
hive>
acadgild@localhost:~

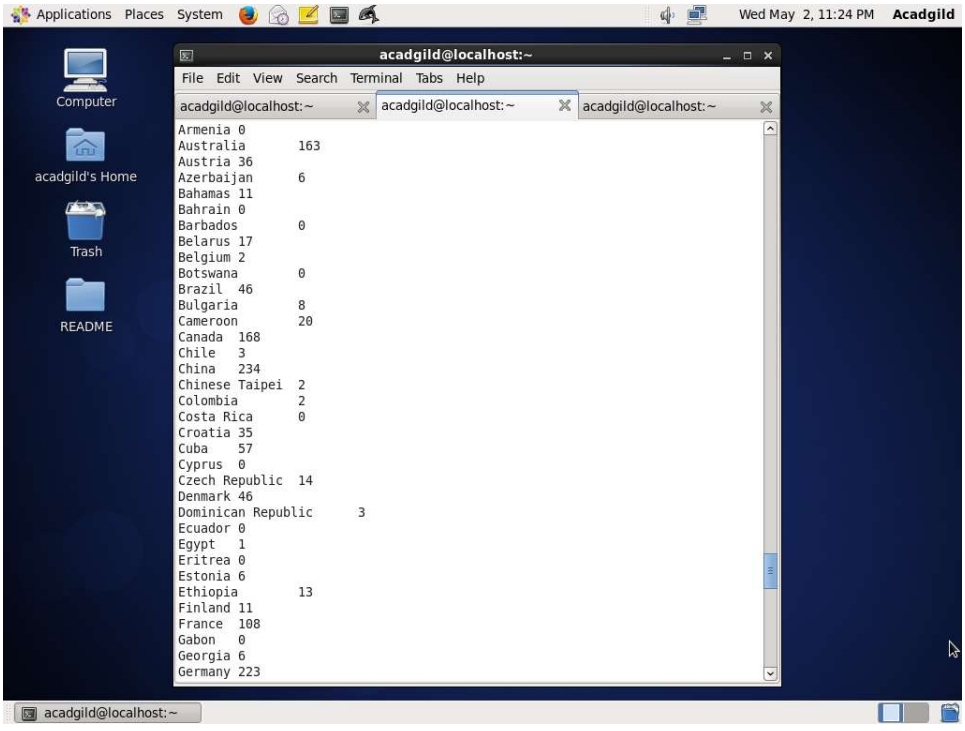
- **Write a Hive program to find the number of gold medals each country won.**

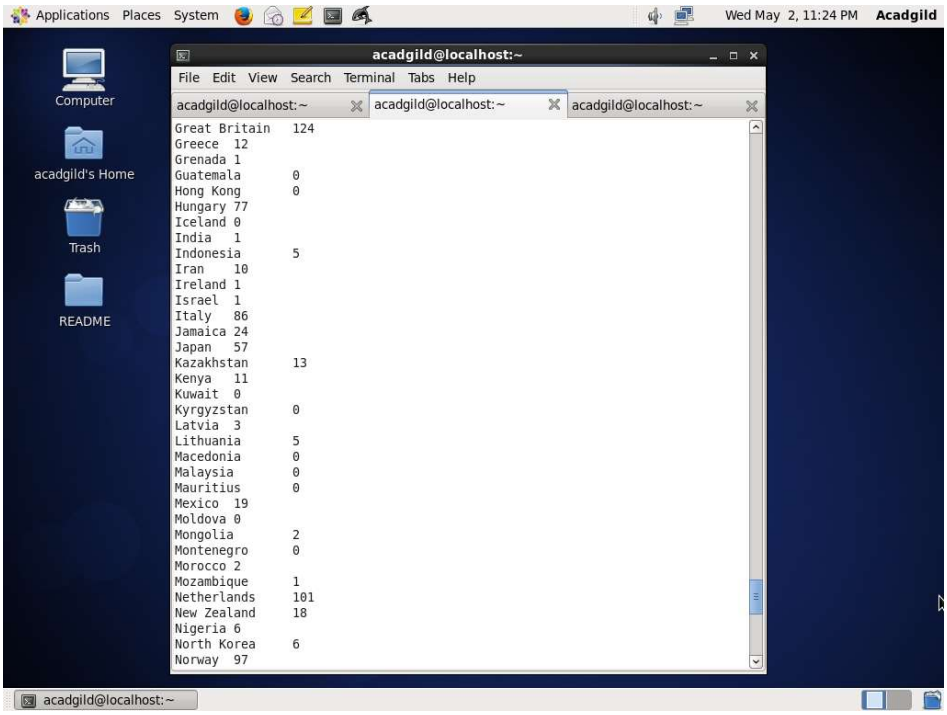
By select query the country and sum of gold medals from Olympic table.

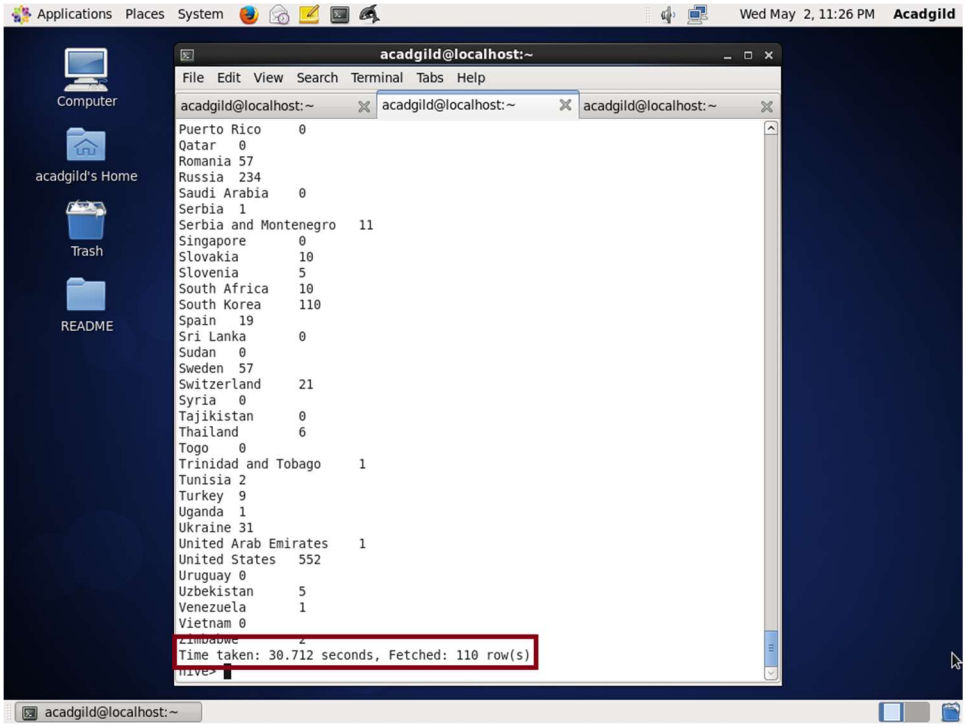
By using the following command, the total number of gold medals won by each country are displayed.

Select country, SUM(gold) from Olympic GROUP BY country;






Applications Places System
Wed May 2, 11:24 PM Acadgild
Computer
acadgild's Home
Trash
README
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
acadgild@localhost:~
acadgild@localhost:~
Great Britain 124
Greece 12
Grenada 1
Guatemala 0
Hong Kong 0
Hungary 77
Iceland 0
India 1
Indonesia 5
Iran 10
Ireland 1
Israel 1
Italy 86
Jamaica 24
Japan 57
Kazakhstan 13
Kenya 11
Kuwait 0
Kyrgyzstan 0
Latvia 3
Lithuania 5
Macedonia 0
Malaysia 0
Mauritius 0
Mexico 19
Moldova 0
Mongolia 2
Montenegro 0
Morocco 2
Mozambique 1
Netherlands 101
New Zealand 18
Nigeria 6
North Korea 6
Norway 97
acadgild@localhost:~


Applications Places System
Wed May 2, 11:26 PM Acadgild
Computer
acadgild's Home
Trash
README
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
acadgild@localhost:~
acadgild@localhost:~
Puerto Rico 0
Qatar 0
Romania 57
Russia 234
Saudi Arabia 0
Serbia 1
Serbia and Montenegro 11
Singapore 0
Slovakia 10
Slovenia 5
South Africa 10
South Korea 110
Spain 19
Sri Lanka 0
Sudan 0
Sweden 57
Switzerland 21
Syria 0
Tajikistan 0
Thailand 6
Togo 0
Trinidad and Tobago 1
Tunisia 2
Turkey 9
Uganda 1
Ukraine 31
United Arab Emirates 1
United States 552
Uruguay 0
Uzbekistan 5
Venezuela 1
Vietnam 0
Time taken: 30.712 seconds, Fetched: 110 row(s)
hive>
acadgild@localhost:~

- 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.

Java-based user-defined function (UDF)

Create a Java class for the User Defined Function which extends ora.apache.hadoop.hive.sq.exec.UDF and implements more than one evaluate() methods.

concatws.java

```
package concatws;

import org.apache.hadoop.hive.ql.exec.Description;
import org.apache.hadoop.hive.ql.exec.UDF;

// Description of the UDF
@Description (name="concatws",
    value="_FUNC_ (string SEP,array<string>)-RETURN_TYPE(STRING)\n")
public class concatws extends UDF {
    // Accepts two arguments, arg1 is string and arg2 is array of string
    public String evaluate (String arg1, String [] arg2) {
        String Output = "";
        // If the arg1 and arg2 are null then return null
        if (arg1== null && arg2== null)
        {
                return null;
        }
        // Length of arg2 is saved in length
        int length = arg2.length;
        for (int i=0; i<length; i++)
        {
        Output += arg2[i];
        }
        // Return the value of Concatenated arg1 string and arg2 array of string
        return(arg1.concat(Output));
    }
}
```

Jar file is created with the above code

Saved as concatws.jar

Using show databases, the databases are listed.

Show databases;

In that databases **custom** database is listed.

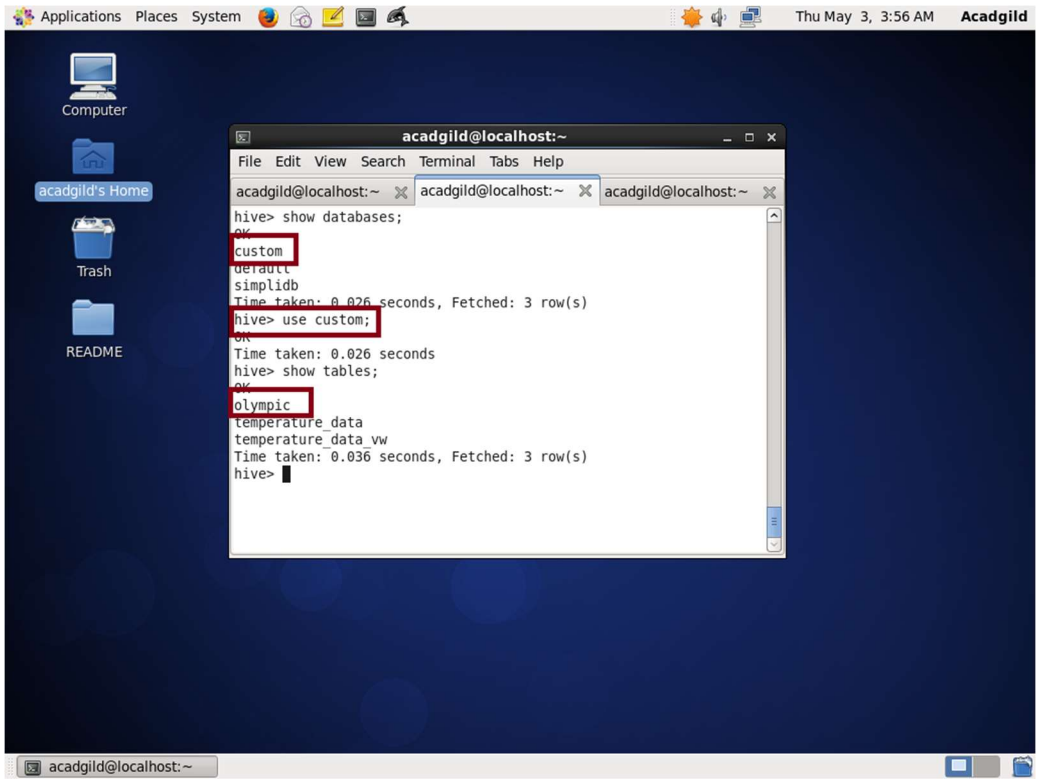
Using the command, use custom which is shown in the databases list.

Use custom;

Using show tables, the tables are listed.

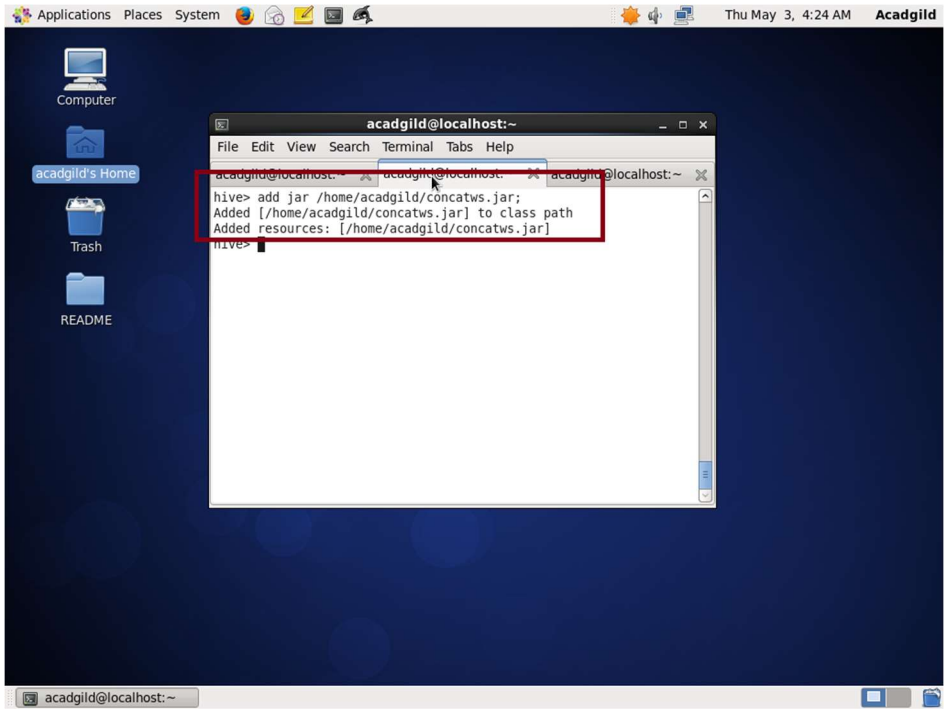
Show tables;

In that databases, **Olympic** table is listed.



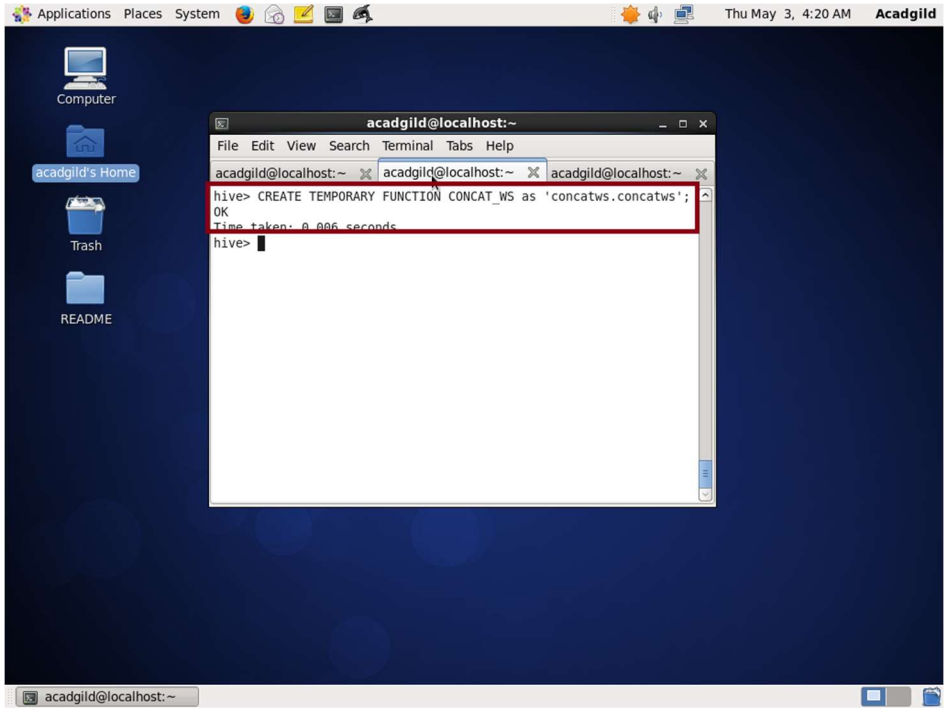
add jar /home/acadgild/hadoop/concatws.jar;

The concatws.jar which is created as UDF is added to the path file by using the above command and the screenshot is shown below.



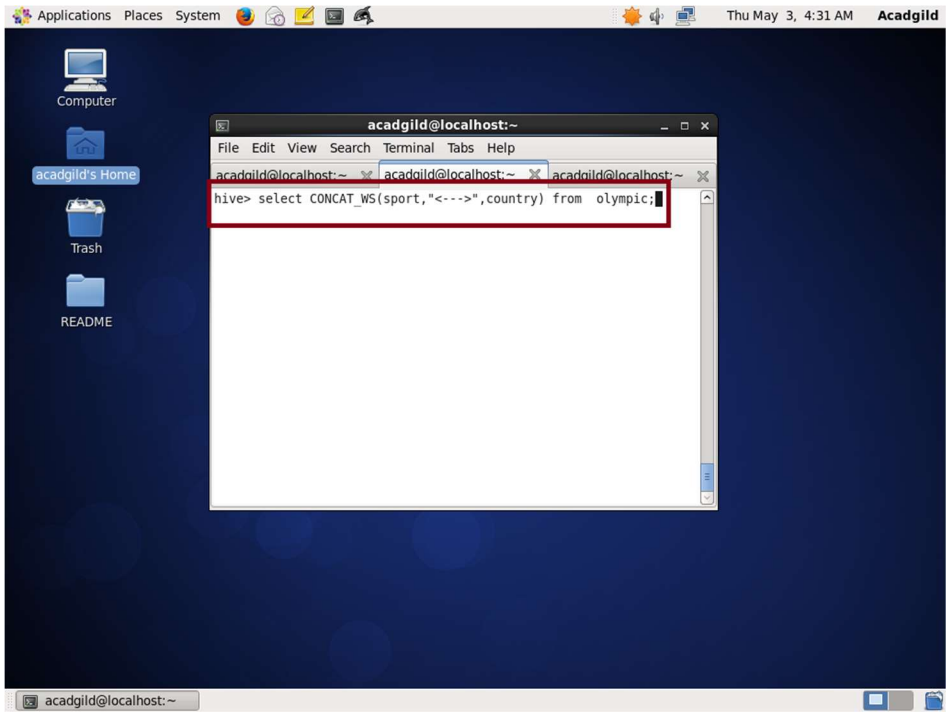
Temporary function is created as following,

CREATE TEMPORARY FUNCTION CONCAT_WS AS 'concatws.concatws';

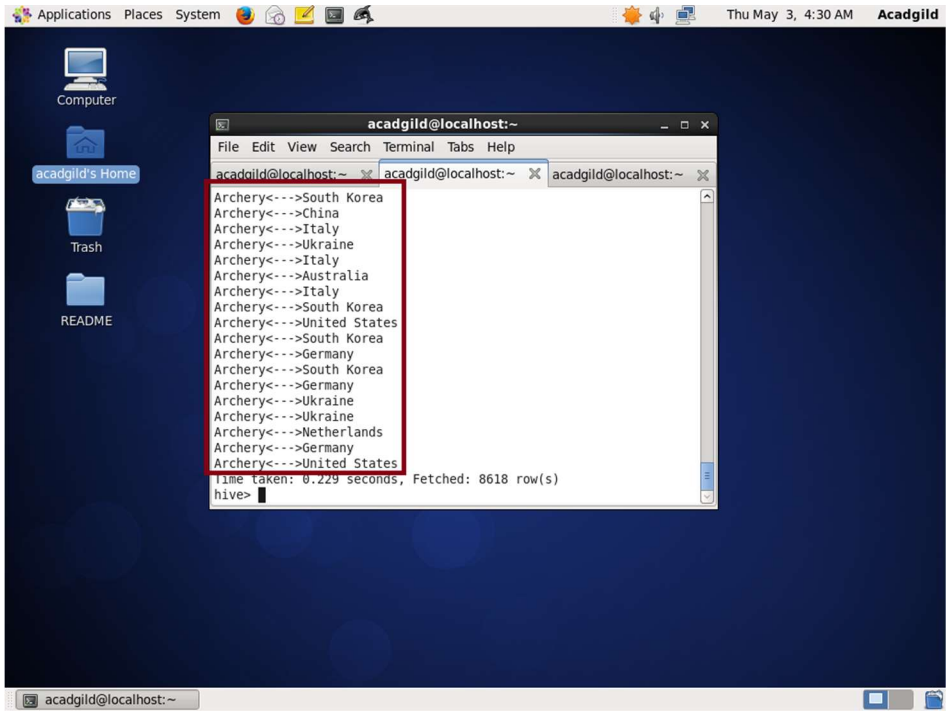


The temporary function is created is used as following in the Olympic table.

select CONCAT_WS(sport,'←→,country) from Olympic;



Select one column (sport) input as String and another array(country) as Array of Strings is concatenated by CONCAT_WS(string SEP, array<string>)



- 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.

Transactions in Hive

Transactions in Hive are introduced in **Hive 0.13**, but they only partially fulfill the ACID properties like atomicity, consistency, durability, at the partition level. Here, Isolation can be provided by turning on one of the locking mechanisms available with zookeeper or in memory.

But in **Hive 0.14**, new API's have been added to completely fulfill the ACID properties while performing any transaction.

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

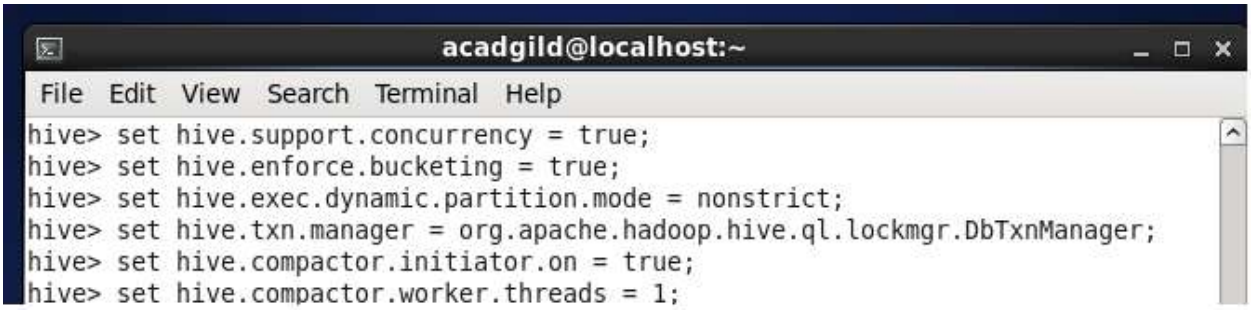
- Insert
- Delete
- Update

There are numerous limitations with the present transactions available in Hive 0.14. ORC is the file format supported by Hive transaction. It is now essential to have ORC file format for performing transactions in Hive. The table needs to be bucketed in order to support transactions.

Row-level Transactions Available in Hive 0.14

Let's perform some row-level transactions available in Hive 0.14. Before creating a Hive table that supports transactions, the transaction features present in Hive needs to be turned on, as by default they are turned off.

The below properties needs to be set appropriately in hive shell, order-wise to work with transactions in Hive:

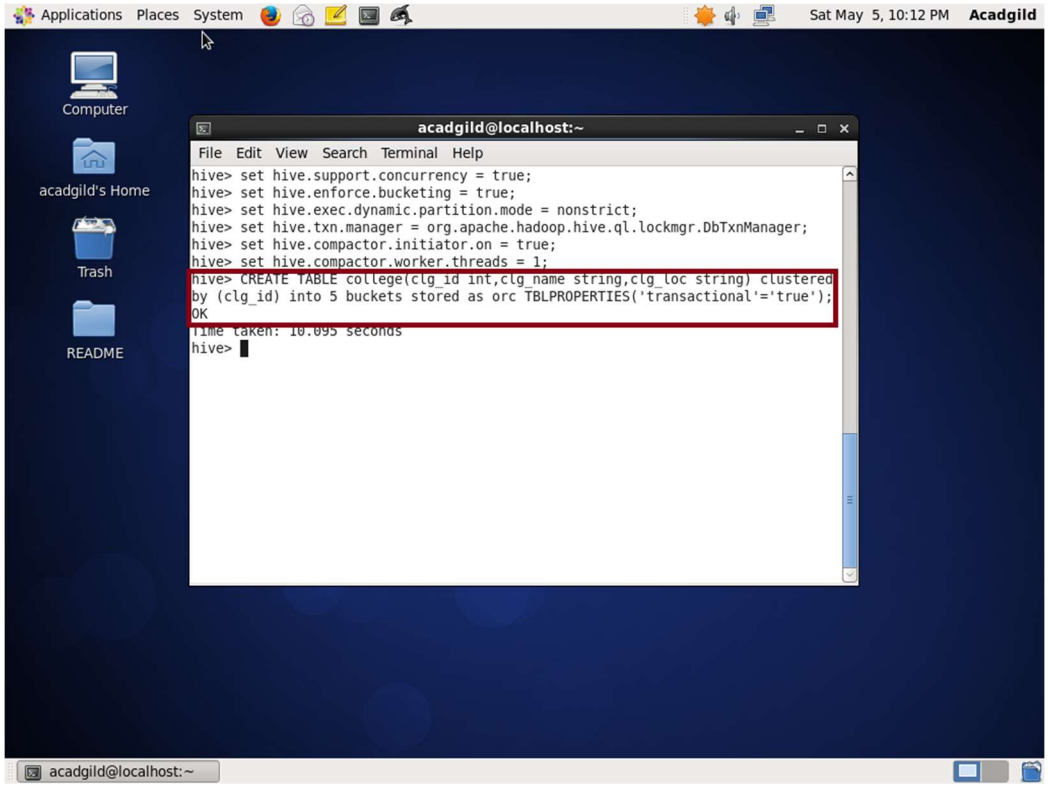


If the above properties are not set properly, the 'Insert' operation will work.

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');

The above syntax will create a table with name '*college*' and the columns present in the table are '*clg_id, clg_name, clg_loc*'. We are *bucketing* the table by '*clg_id*' and the table format is '*orc*', also we are enabling the transactions in the table by specifying it inside the *TBLPROPERTIES* as *'transactional'='true'*

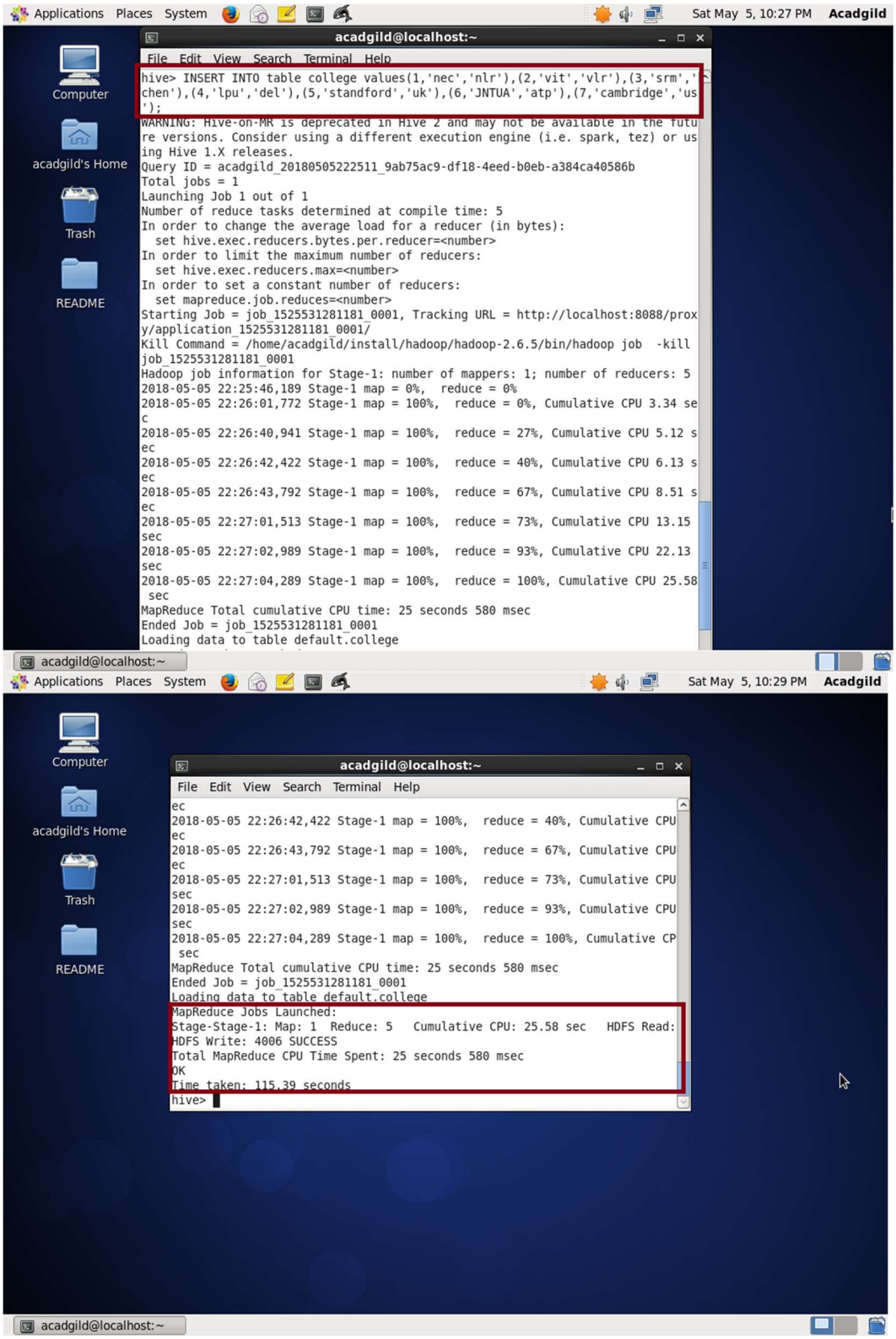


We have successfully created a table with name '*college*' which supports row-level transactions of Hive.

Inserting Data into a Hive Table

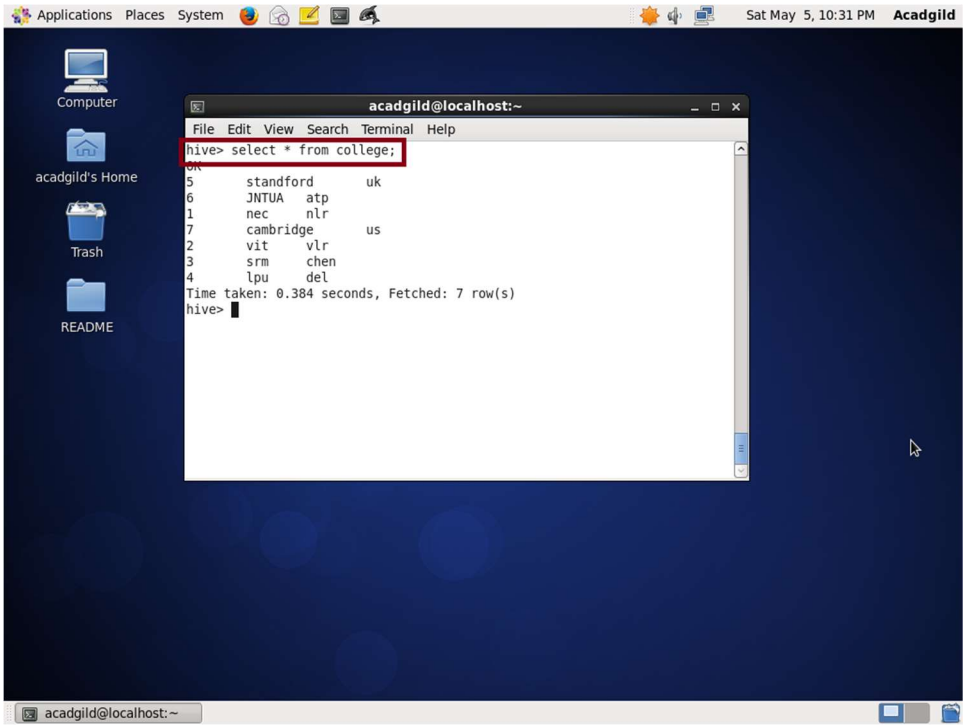
INSERT INTO table college values(1,'nec','nlr'),(2,'vit','vlr'),(3,'srm','chen'),(4,'lpu','del'),(5,'stanford','uk'),(6,'JNTUA','atp'),(7,'cambridge','us');

The above command is used to insert row wise data into the Hive table. Here, each row is separated by '*( )' brackets.*



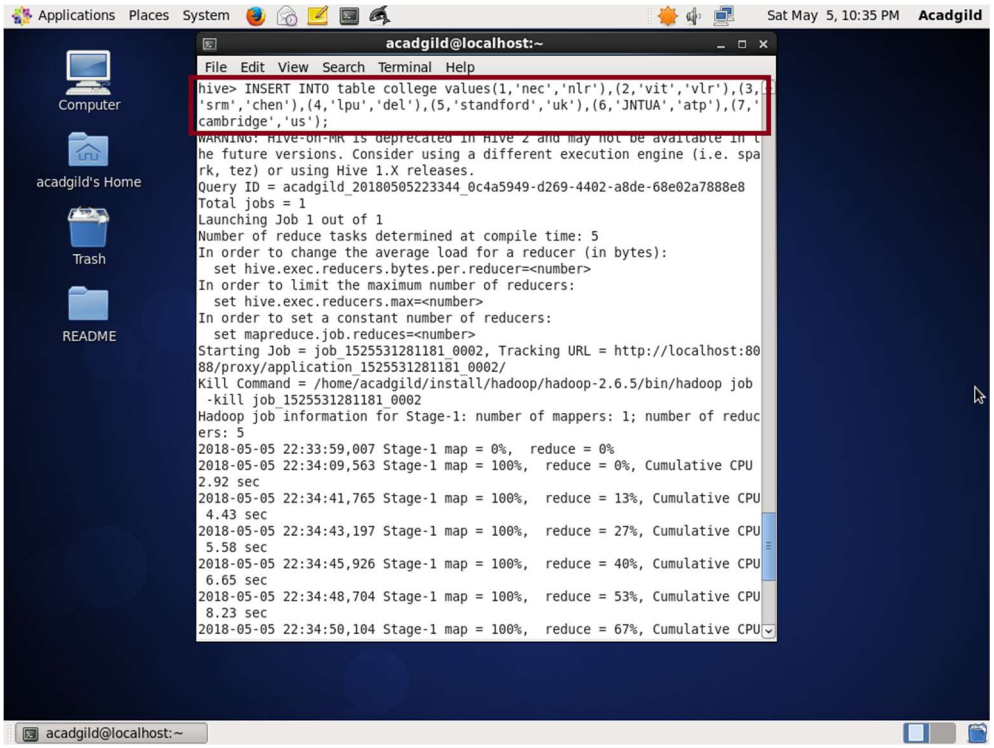
Now, we have successfully inserted the data into the Hive table.
The contents of the table can be viewed using the command

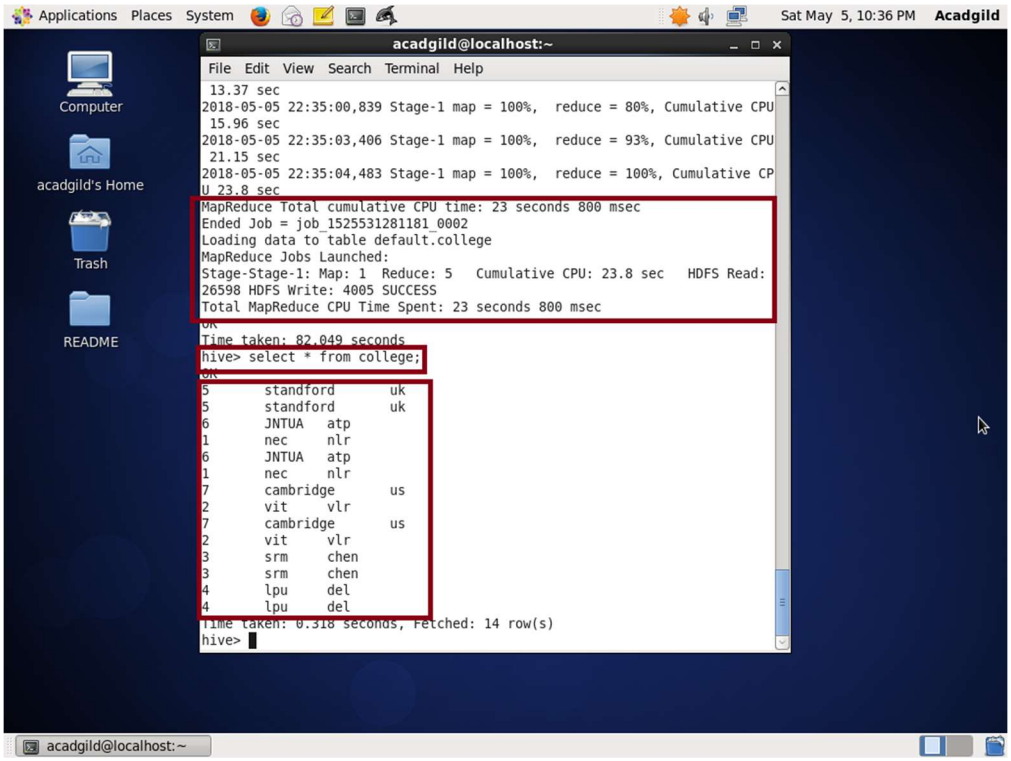
select * from college



From the above image, we can see that the data has been inserted successfully into the table.

Now if we try to re-insert the same data again, it will be appended to the previous data as shown below:

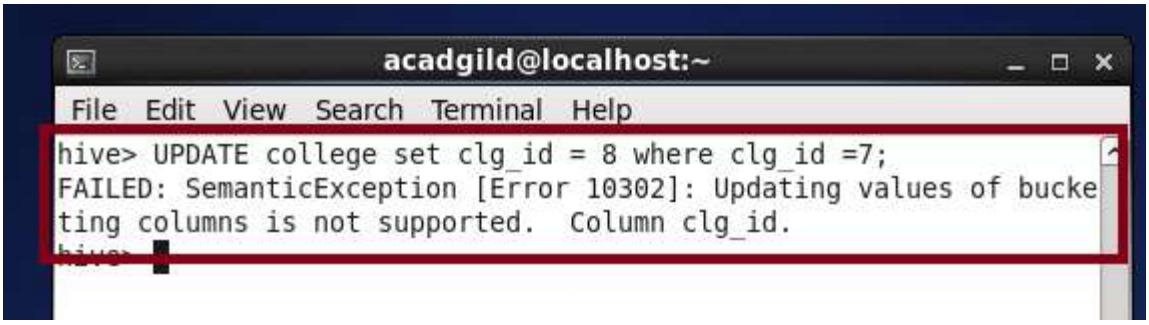




Updating the Data in Hive Table

UPDATE college set clg_id = 8 where clg_id = 7;

The above command is used to update a row in Hive table.

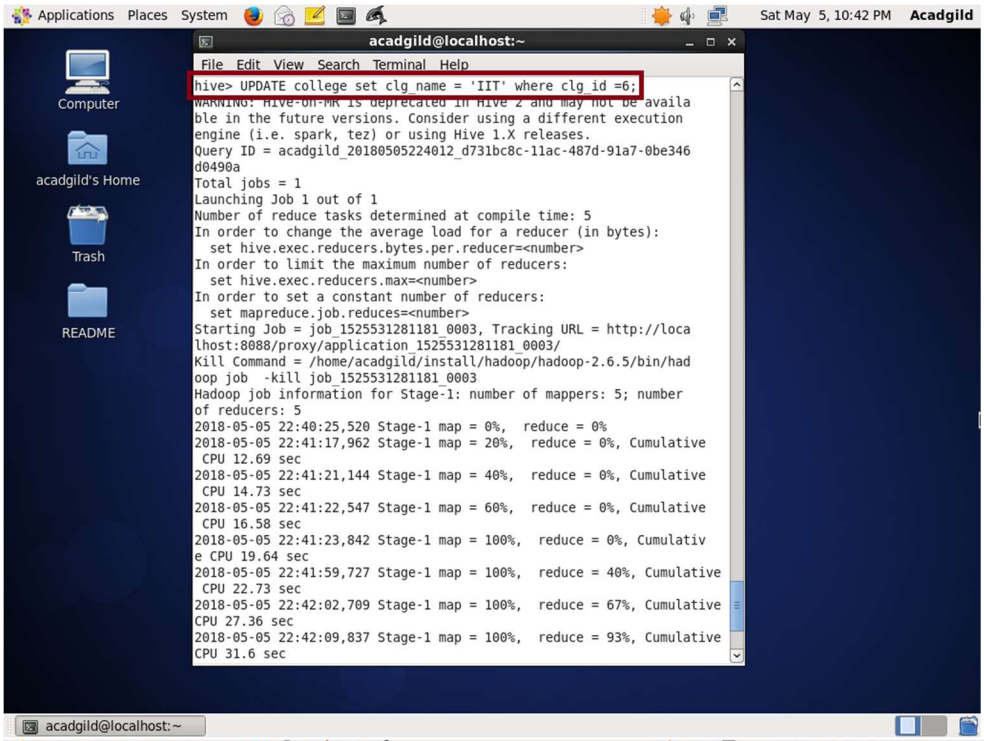


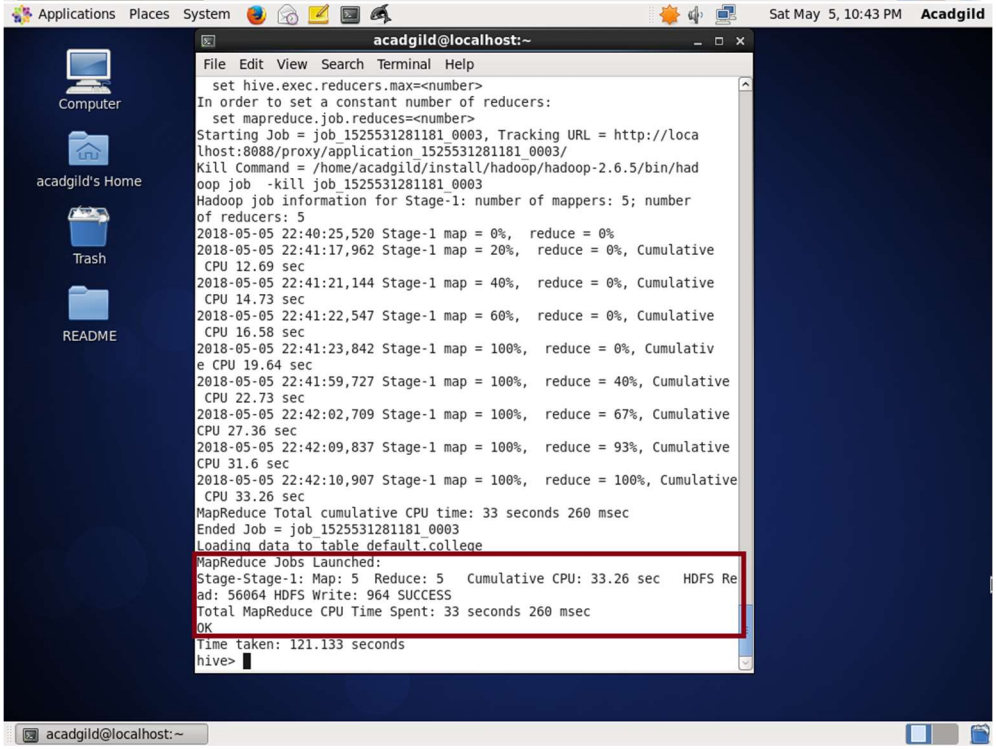
From the above image, we can see that we have received an error message. This means that the Update command is not supported on the columns that are bucketed.

In this table, we have bucketed the 'clg_id' column and performing the Update operation on the same column, so we have go the error

FAILED: SemanticException[Error 10302]: Updating values of bucketing columns is not supported. Column clg_id

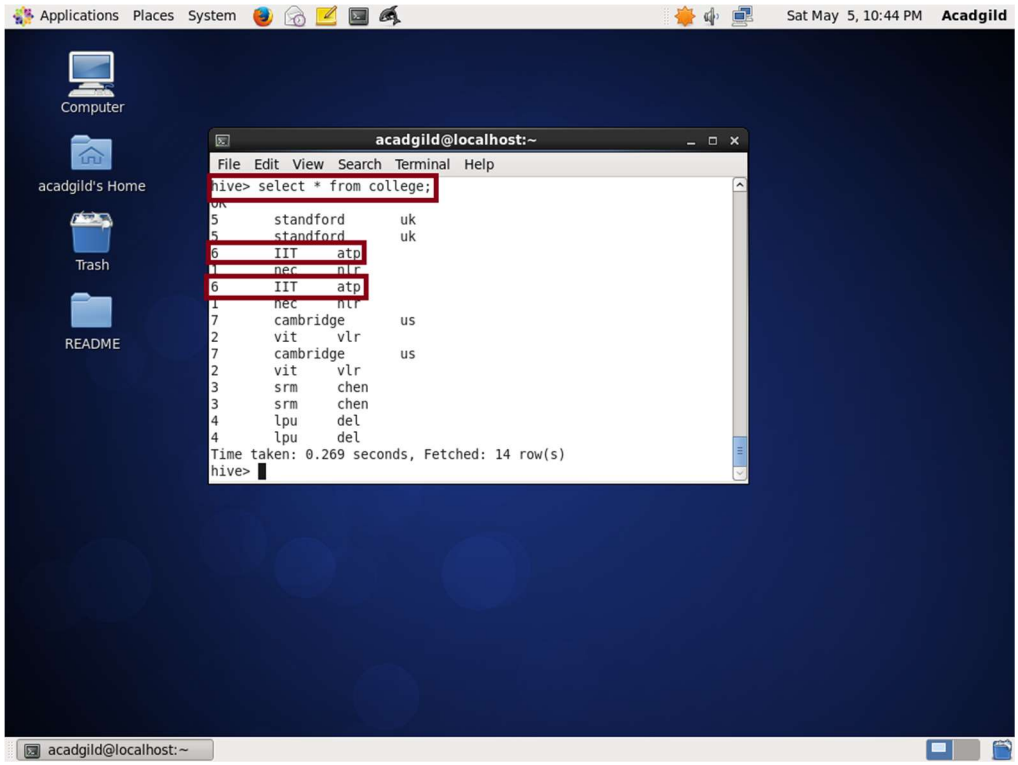
Now let's perform the update operation on Non bucketed column





We have successfully updated the data.

The updated data can be checked using the command select * from college.

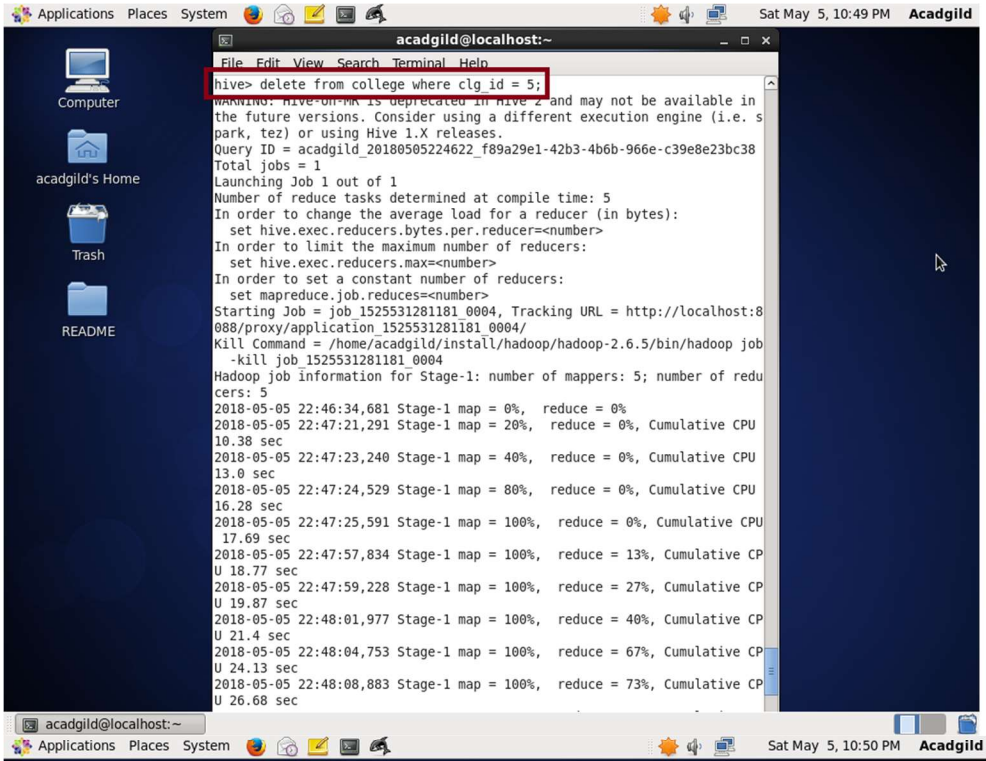


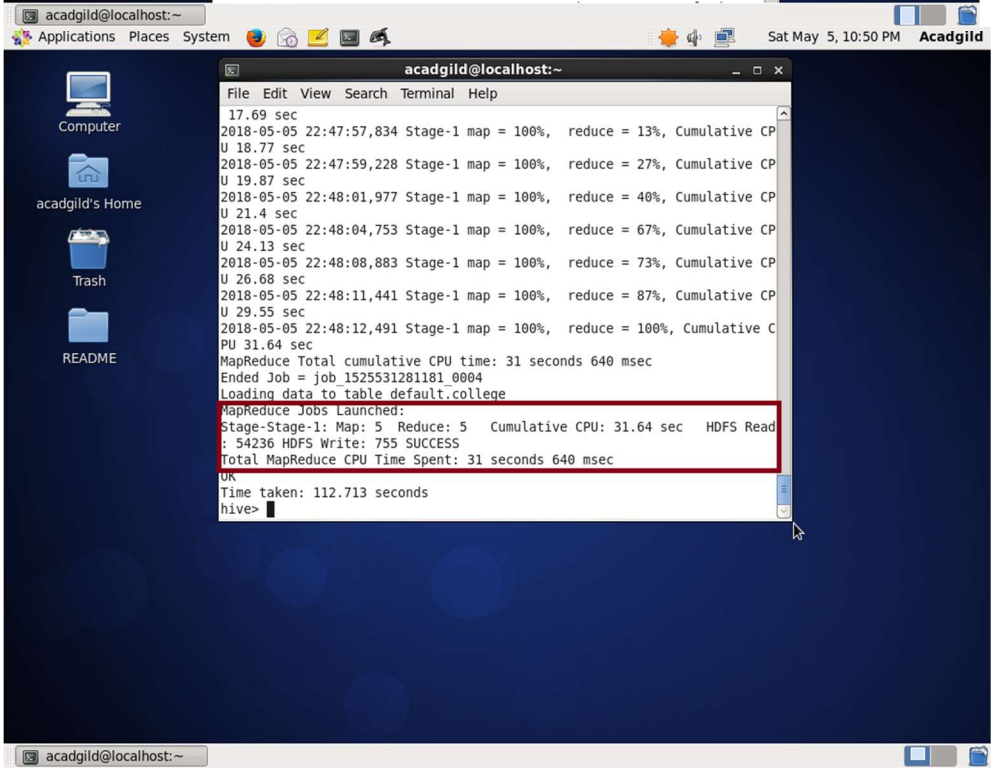
We can see that the data has been updated successfully.
Now let's perform the Delete operation on the same table.

Deleting a Row from Hive Table

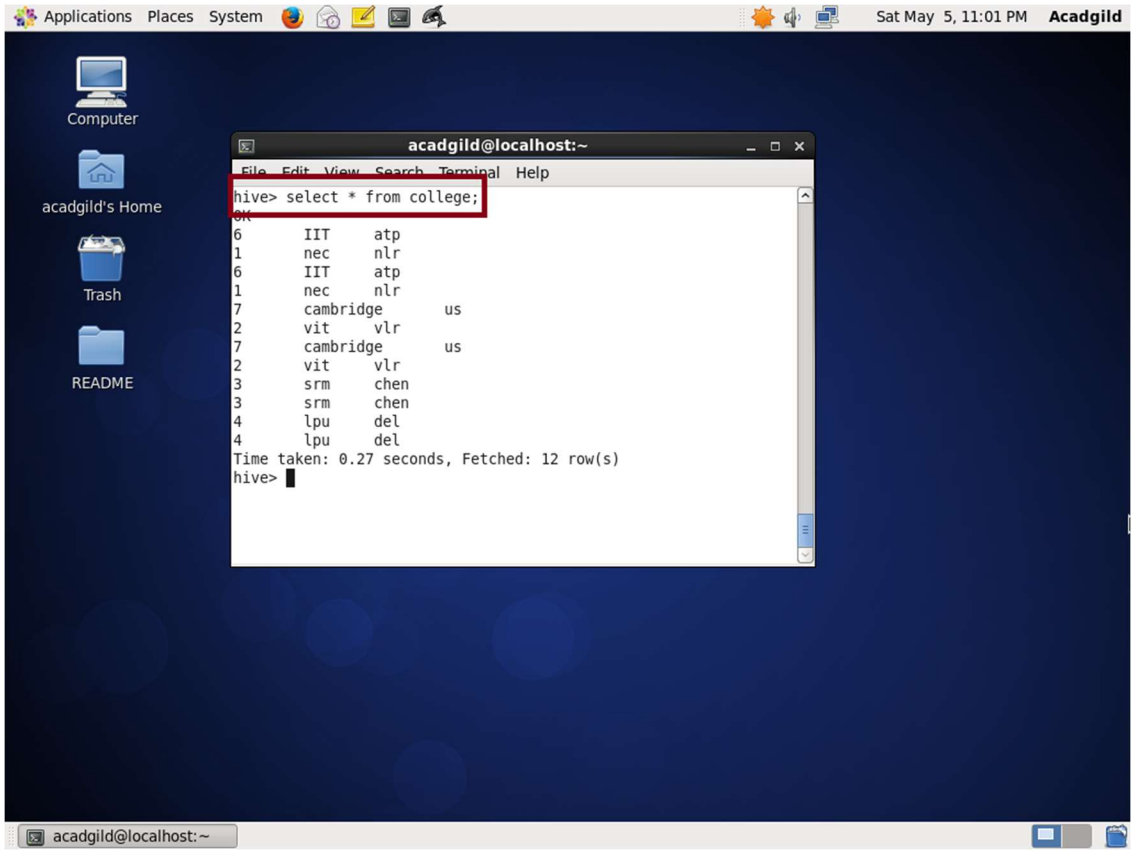
The above command will delete a single row in the Hive table.

delete from college where clg_id = 5;





We have now successfully deleted a row from the Hive table. This can be checked using the command **select * from college**



We can see that there is no row with ***clg_id =5***. This means that we have successfully deleted the row from the Hive table.

This is how the transactions or row-wise operations are performed in Hive.