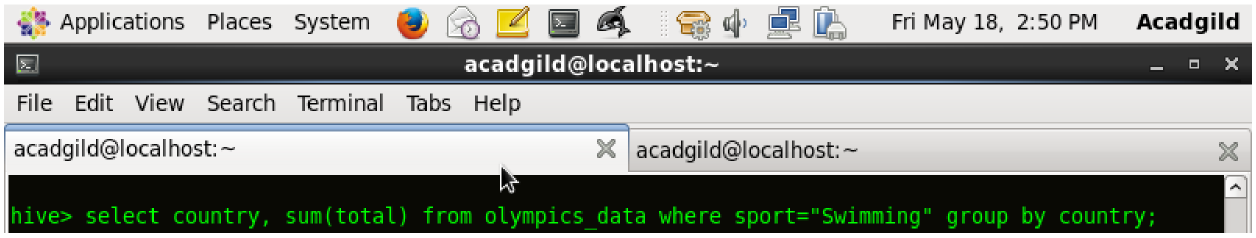
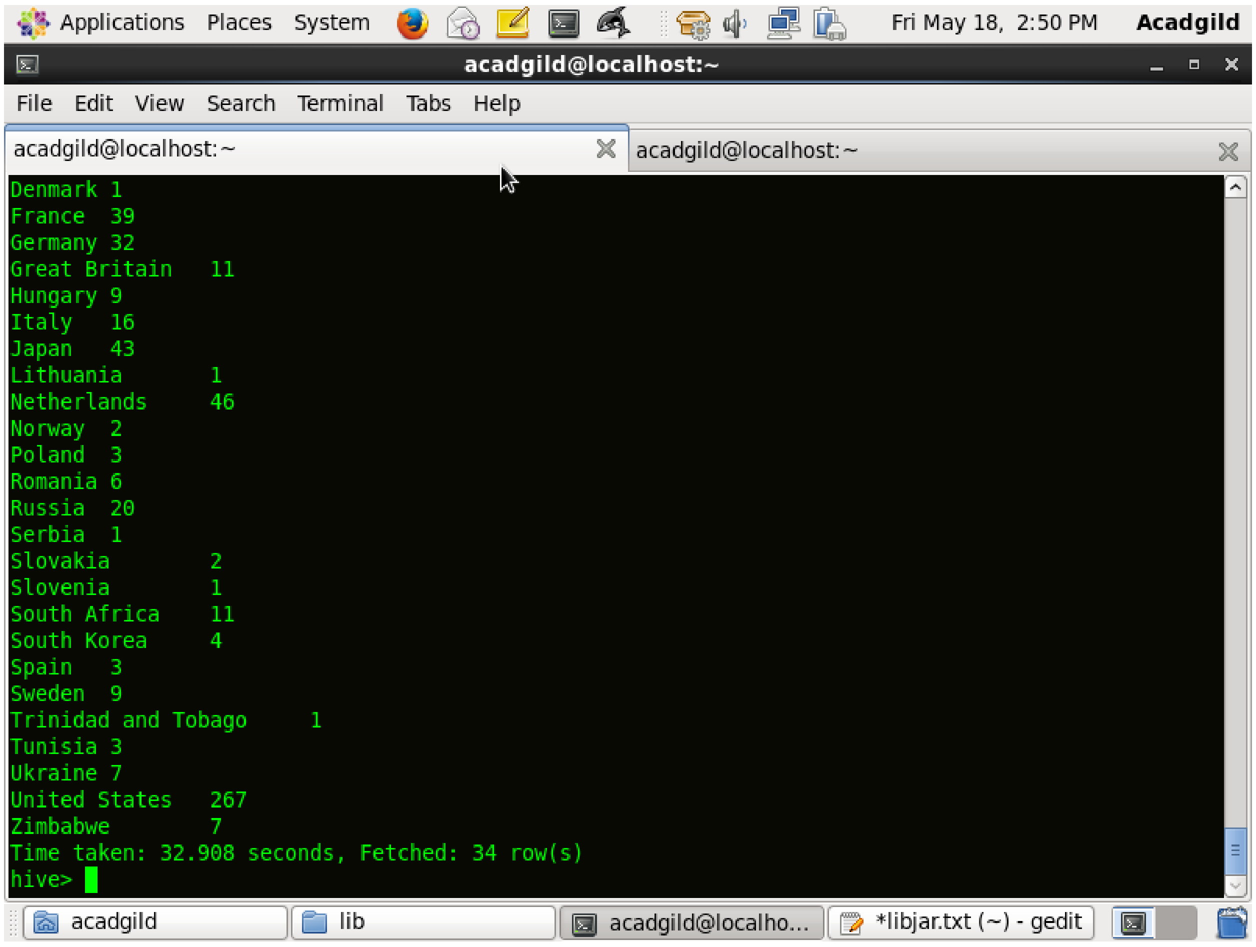
To find the number of medals won by each country in swimming, we will use group by country since the results are grouped by country.
Since we want only those records whose sport is swimming the where key word will help in filtering those records. Lastly by using the sum() function for total field, we will get medals won by each country in swimming



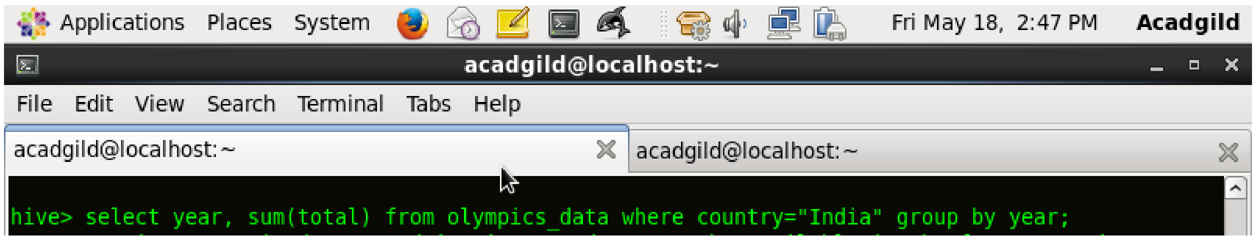
```
hive> select country, sum(total) from olympics_data where sport="Swimming" group by country;
```

Output:



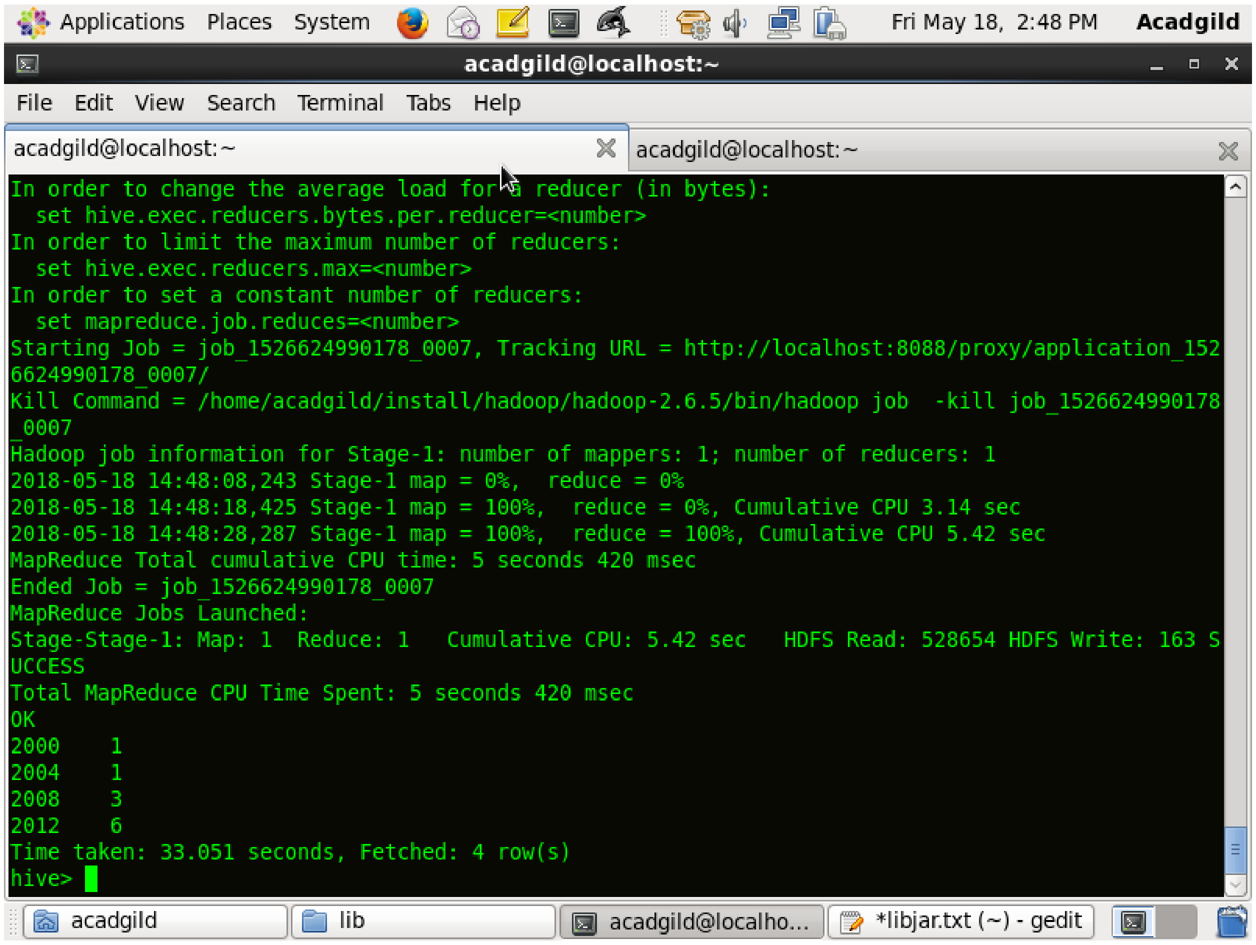
```
Denmark 1
France  39
Germany 32
Great Britain   11
Hungary 9
Italy   16
Japan   43
Lithuania       1
Netherlands     46
Norway  2
Poland  3
Romania 6
Russia  20
Serbia  1
Slovakia        2
Slovenia        1
South Africa    11
South Korea     4
Spain   3
Sweden  9
Trinidad and Tobago     1
Tunisia 3
Ukraine 7
United States   267
Zimbabwe        7
Time taken: 32.908 seconds, Fetched: 34 row(s)
hive> 
```

To find the number of medals India won year wise we will provide the where condition for country to filter records only where country is equal to India and we will group the filtered output according to the year since we want year wise output.


Applications Places System
Fri May 18, 2:47 PM Acadgild
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
acadgild@localhost:~

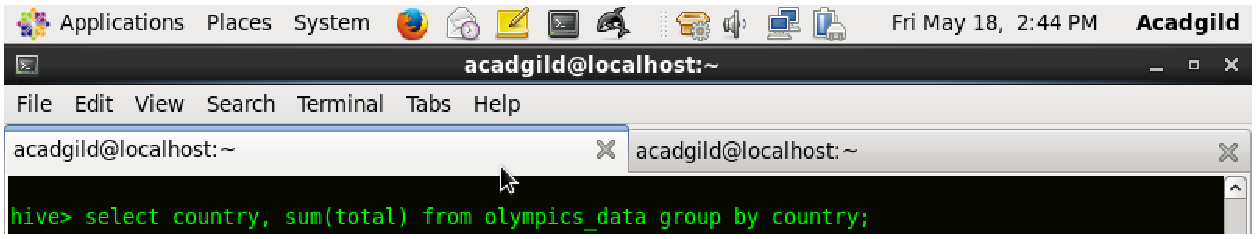
```
hive> select year, sum(total) from olympics_data where country="India" group by year;
```

Output:



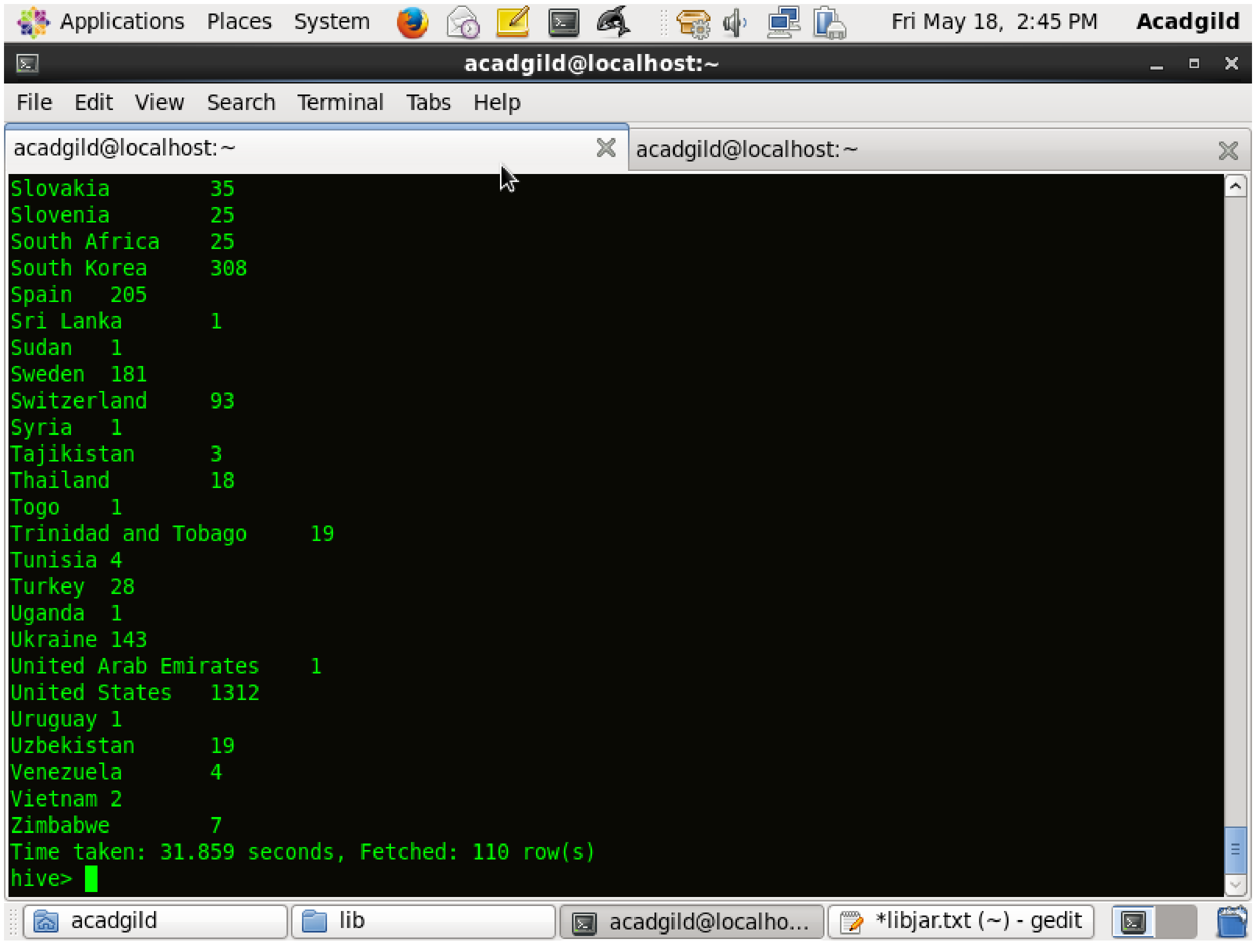
```
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1526624990178_0007, Tracking URL = http://localhost:8088/proxy/application_152
6624990178_0007/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1526624990178
_0007
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-05-18 14:48:08,243 Stage-1 map = 0%,  reduce = 0%
2018-05-18 14:48:18,425 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.14 sec
2018-05-18 14:48:28,287 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.42 sec
MapReduce Total cumulative CPU time: 5 seconds 420 msec
Ended Job = job_1526624990178_0007
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.42 sec   HDFS Read: 528654 HDFS Write: 163 S
UCCESS
Total MapReduce CPU Time Spent: 5 seconds 420 msec
OK
2000    1
2004    1
2008    3
2012    6
Time taken: 33.051 seconds, Fetched: 4 row(s)
hive> 
```

To find the total number of medals won by each country, we will just group the results by country and use the sum() function on total field.


Applications Places System
Fri May 18, 2:44 PM Acadgild
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
acadgild@localhost:~

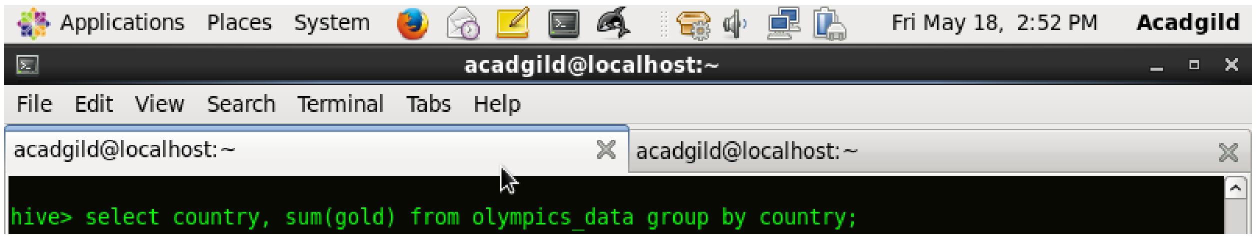
```
hive> select country, sum(total) from olympics_data group by country;
```

Output:



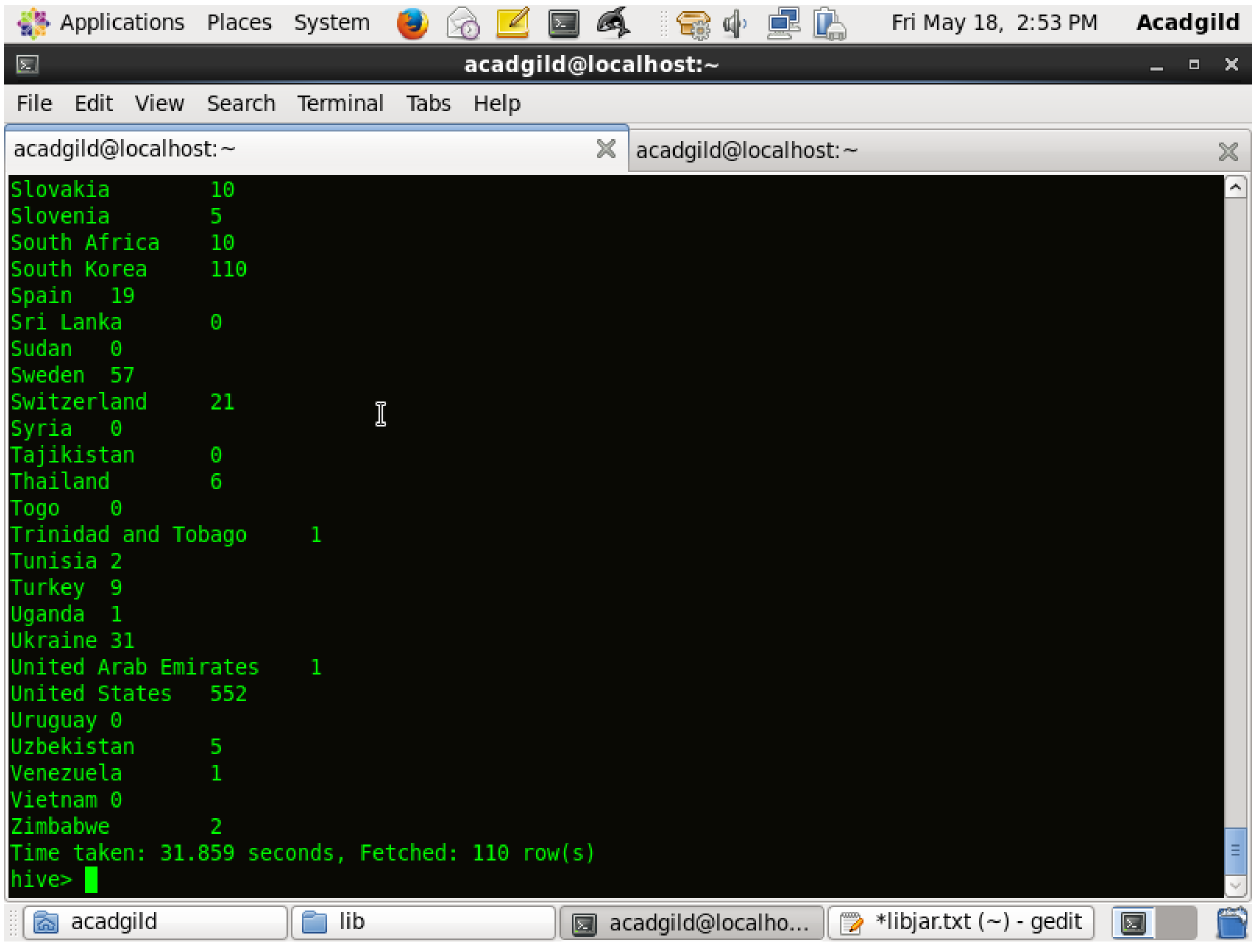
```
Slovakia        35
Slovenia        25
South Africa    25
South Korea     308
Spain   205
Sri Lanka       1
Sudan   1
Sweden  181
Switzerland     93
Syria   1
Tajikistan      3
Thailand        18
Togo    1
Trinidad and Tobago     19
Tunisia 4
Turkey  28
Uganda  1
Ukraine 143
United Arab Emirates    1
United States   1312
Uruguay 1
Uzbekistan      19
Venezuela       4
Vietnam 2
Zimbabwe        7
Time taken: 31.859 seconds, Fetched: 110 row(s)
hive>
```

To find the total number of gold medals each country won, we will apply the sum function on gold field and group it by country field


Applications Places System
Fri May 18, 2:52 PM Acadgild
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
acadgild@localhost:~

```
hive> select country, sum(gold) from olympics_data group by country;
```

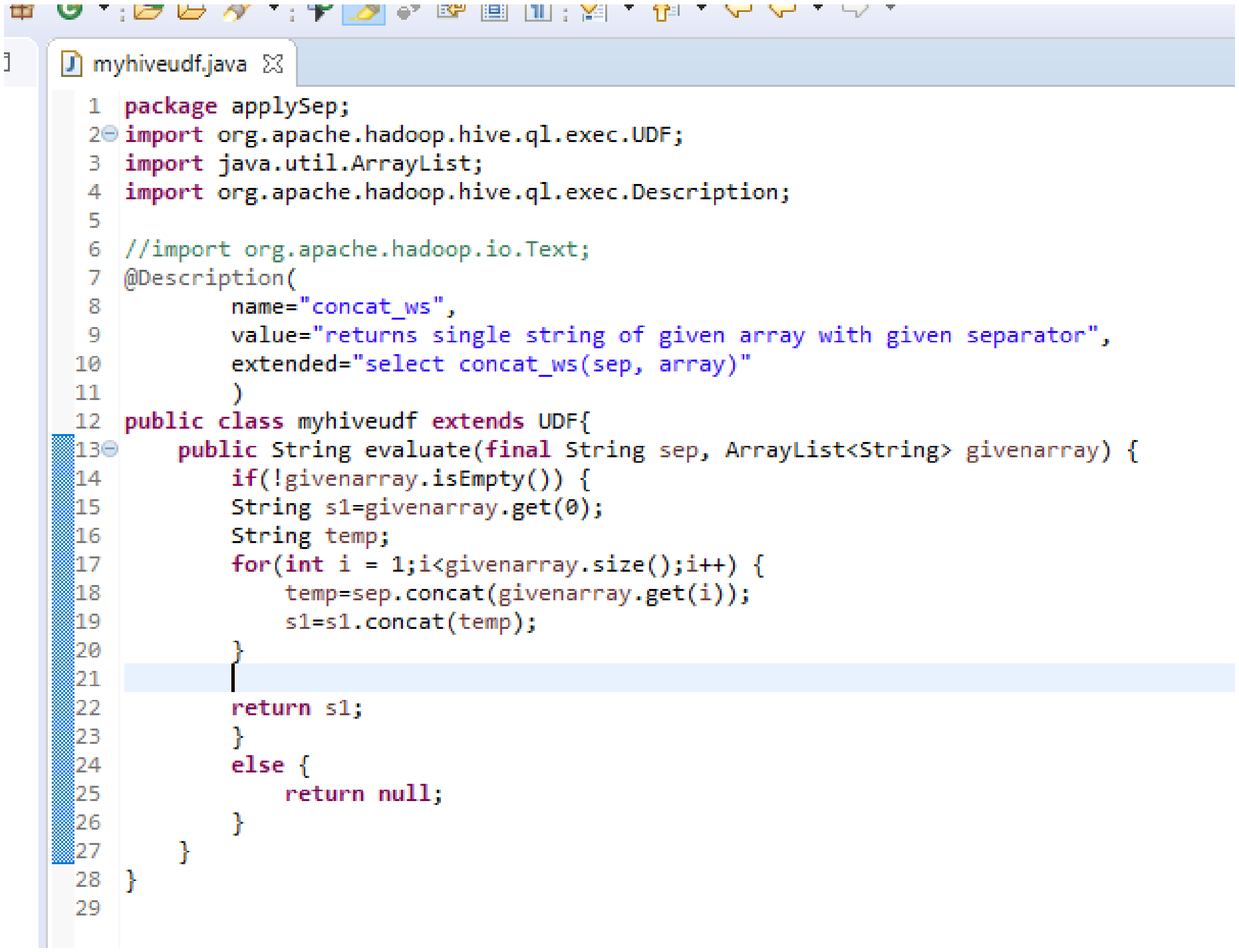
Output:



```
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
Zimbabwe        2
Time taken: 31.859 seconds, Fetched: 110 row(s)
hive>
```

Code for HIVE Udf:

The evaluate function will accept two arguments, the first argument is the separator of string data type and the second argument accepts array of type Arraylist.

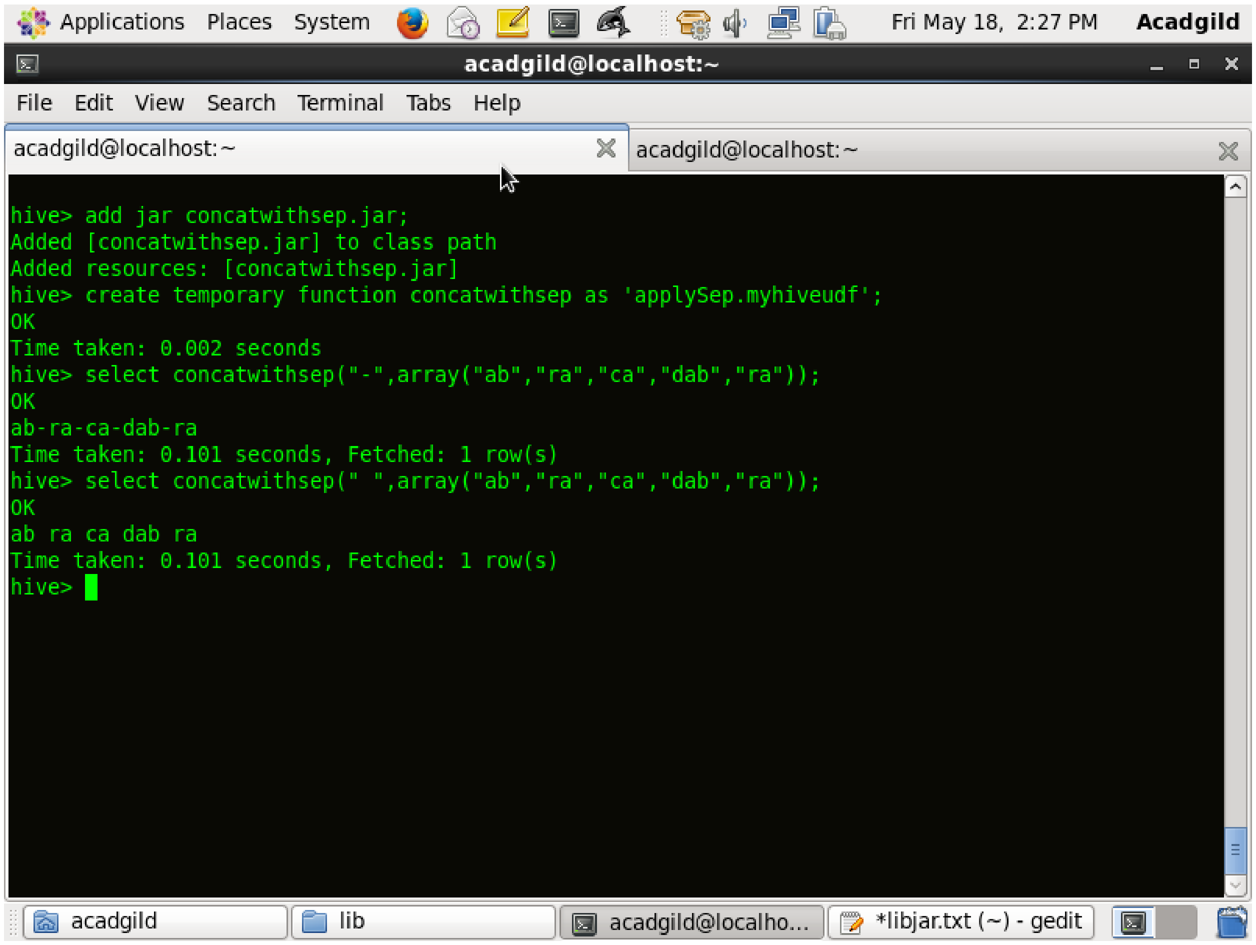


```java
package applySep;
import org.apache.hadoop.hive.ql.exec.UDF;
import java.util.ArrayList;
import org.apache.hadoop.hive.ql.exec.Description;

//import org.apache.hadoop.io.Text;
@Description(
        name="concat_ws",
        value="returns single string of given array with given separator",
        extended="select concat_ws(sep, array)"
        )
public class myhiveudf extends UDF{
    public String evaluate(final String sep, ArrayList<String> givenarray) {
        if(!givenarray.isEmpty()) {
        String s1=givenarray.get(0);
        String temp;
        for(int i = 1;i<givenarray.size();i++) {
            temp=sep.concat(givenarray.get(i));
            s1=s1.concat(temp);
        }

        return s1;
        }
        else {
            return null;
        }
    }
}

```

To use your custom hive UDF, add the jar to the shell by command:
add jar <jar-name>;

Then create your temporary function by command:
create temporary function <your-udf-function-name> as ‘<package-name>.<class-name>’;

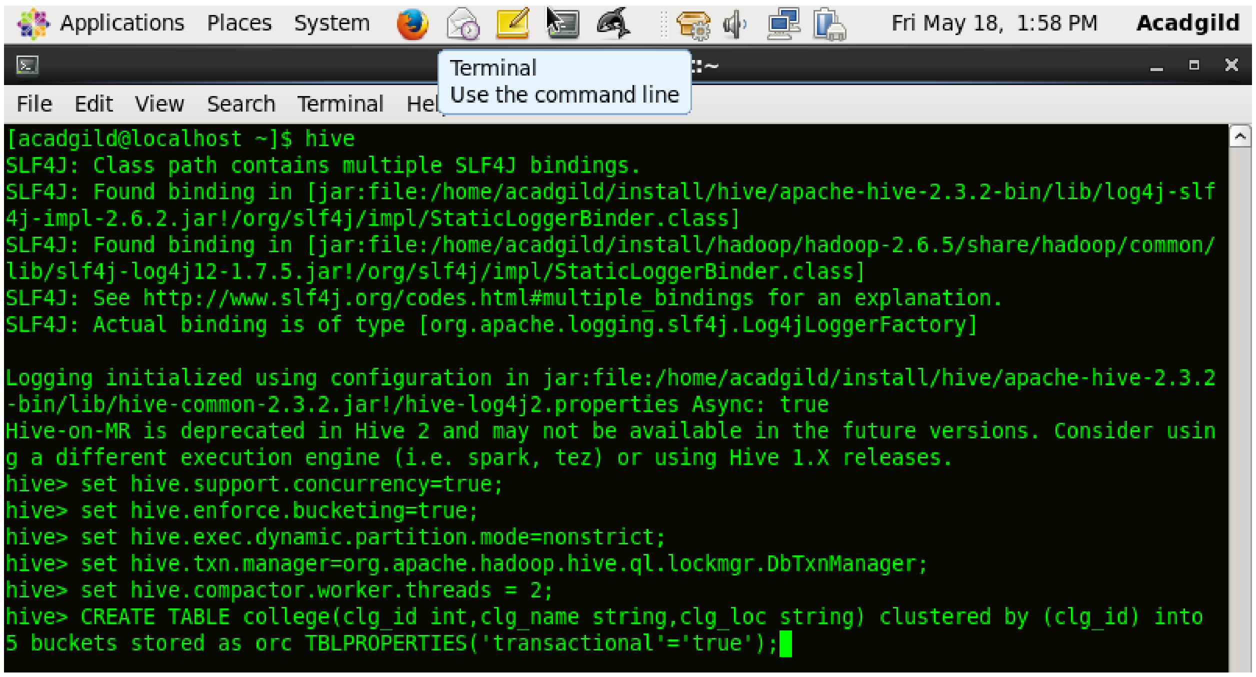
When I enter the command:
select concatwithsep(“-“,array(“ab”,”ra”,”ca”,”dab”,”ra”));
One will get output as ab-ra-ca-dab-ra



```
hive> add jar concatwithsep.jar;
Added [concatwithsep.jar] to class path
Added resources: [concatwithsep.jar]
hive> create temporary function concatwithsep as 'applySep.myhiveudf';
OK
Time taken: 0.002 seconds
hive> select concatwithsep("-",array("ab","ra","ca","dab","ra"));
OK
ab-ra-ca-dab-ra
Time taken: 0.101 seconds, Fetched: 1 row(s)
hive> select concatwithsep(" ",array("ab","ra","ca","dab","ra"));
OK
ab ra ca dab ra
Time taken: 0.101 seconds, Fetched: 1 row(s)
hive> 
```

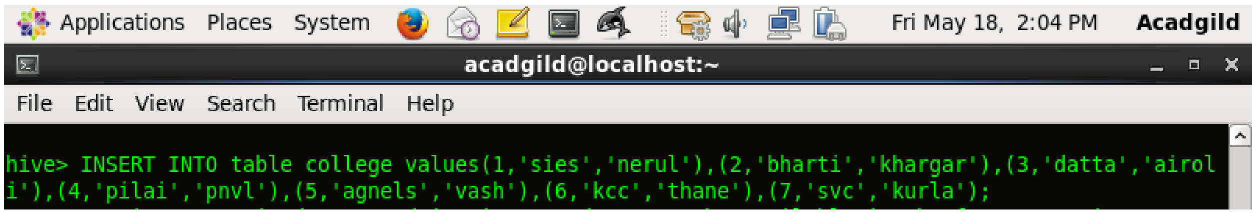
Output of Transactions in hive:

Setting below properties in hive shell and creating table


Applications Places System
Fri May 18, 1:58 PM Acadgild
Terminal
Use the command line
File Edit View Search Terminal Hel

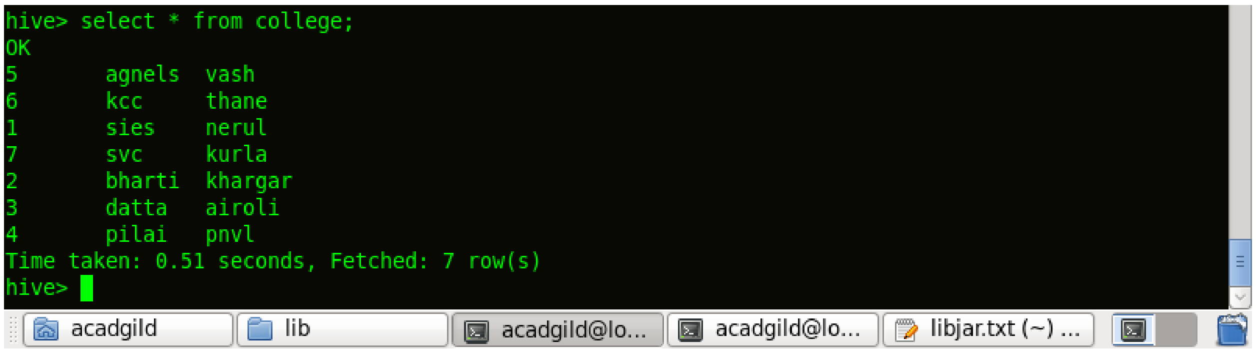
```
[acadgild@localhost ~]$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/log4j-slf
4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/share/hadoop/common/
lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/home/acadgild/install/hive/apache-hive-2.3.2
-bin/lib/hive-common-2.3.2.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider usin
g a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
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.worker.threads = 2;
hive> 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 values into table college


Applications Places System
Fri May 18, 2:04 PM Acadgild
acadgild@localhost:~
File Edit View Search Terminal Help

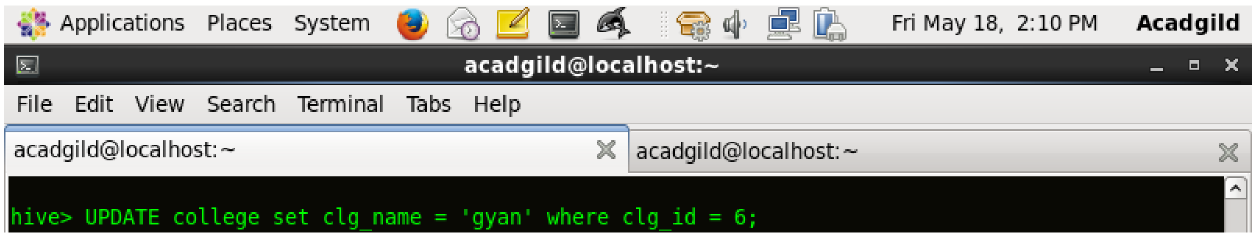
```
hive> INSERT INTO table college values(1,'sies','nerul'),(2,'bharti','khargar'),(3,'datta','airol
i'),(4,'pilai','pnvl'),(5,'agnels','vash'),(6,'kcc','thane'),(7,'svc','kurla');
```

Viewing the contents of table



```
hive> select * from college;
OK
5       agnels  vash
6       kcc     thane
1       sies    nerul
7       svc     kurla
2       bharti  khargar
3       datta   airoli
4       pilai   pnvl
Time taken: 0.51 seconds, Fetched: 7 row(s)
hive>
```

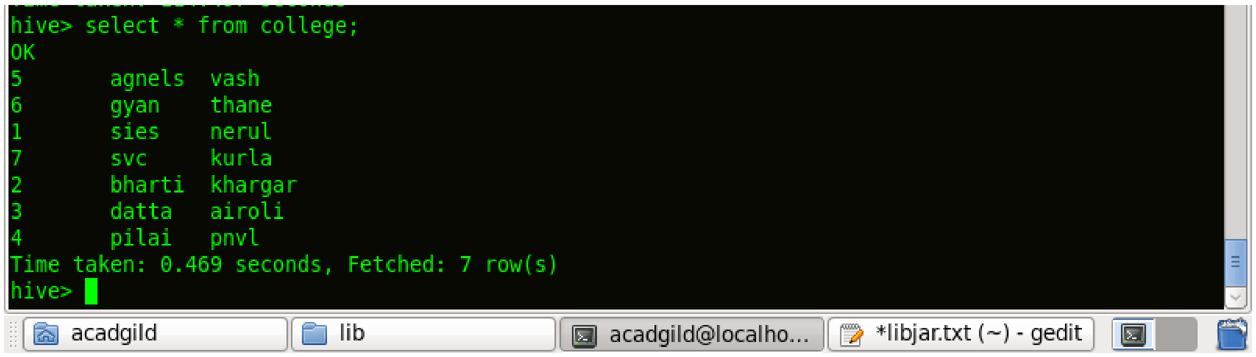
Updating a value in table



Applications Places System
Fri May 18, 2:10 PM Acadgild
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
acadgild@localhost:~

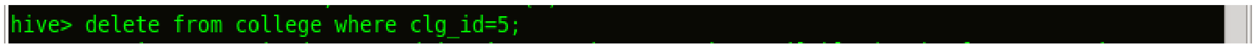
```
hive> UPDATE college set clg_name = 'gyan' where clg_id = 6;
```

‘kcc’ has changed to ‘gyan’



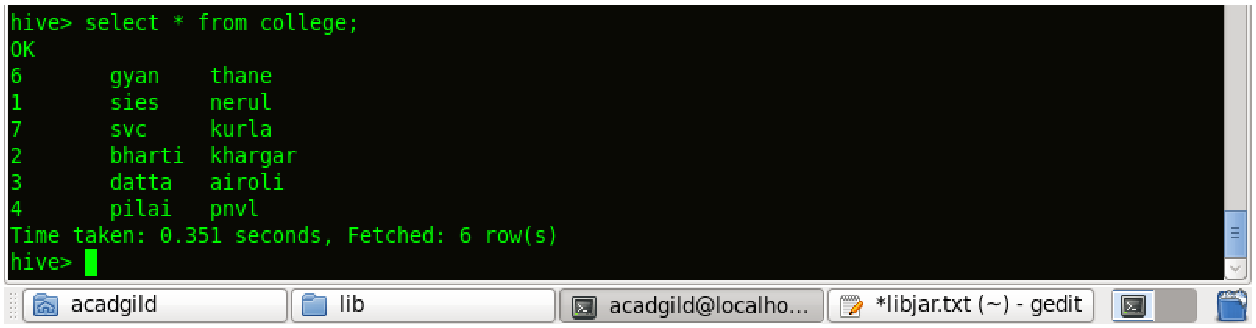
```
hive> select * from college;
OK
5       agnels  vash
6       gyan    thane
1       sies    nerul
7       svc     kurla
2       bharti  khargar
3       datta   airoli
4       pilai   pnvl
Time taken: 0.469 seconds, Fetched: 7 row(s)
hive> 
```

Deleting a record



```
hive> delete from college where clg_id=5;
```

The record whose id=5 has been deleted



```
hive> select * from college;
OK
6       gyan    thane
1       sies    nerul
7       svc     kurla
2       bharti  khargar
3       datta   airoli
4       pilai   pnvl
Time taken: 0.351 seconds, Fetched: 6 row(s)
hive> 
```