Query to create a Database and a table providing the Input Dataset.

CREATE DATABASE olympic;

USE olympic;

CREATE TABLE olympic_data

(Athlete STRING, Age INT, Country STRING, Year INT, Closing_Date STRING, Sport STRING, Gold_Medals INT, Silver_Medals INT, Bronze_Medals INT,Total_Medals INT)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

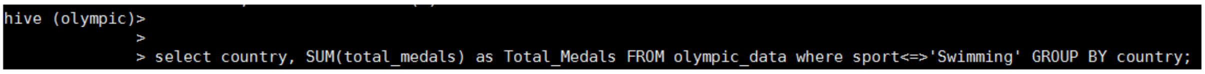
LOAD DATA LOCAL INPATH '/home/acadgild/hadoop/olympic_data.json'

INTO TABLE olympic_data;

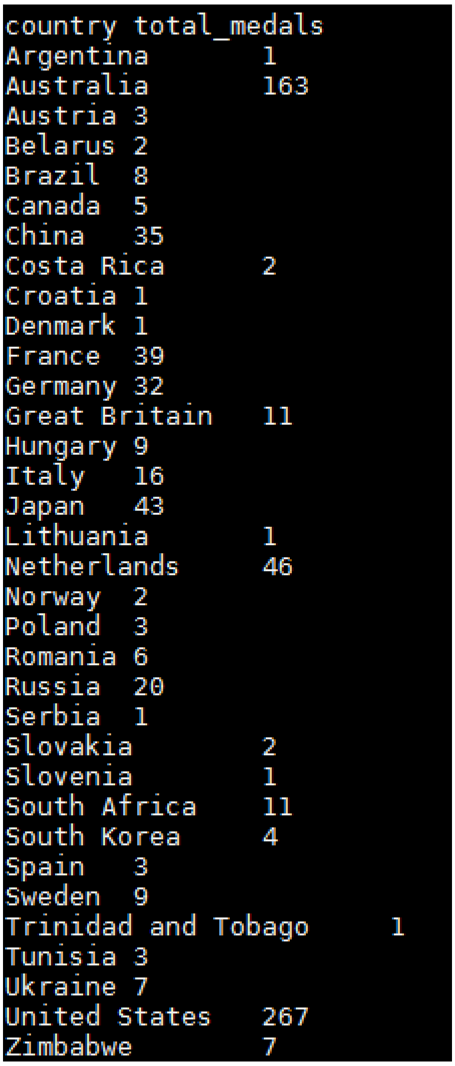
Task 1

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

SELECT country, SUM(total_medals) as Total_Medals FROM olympic_data WHERE sport<=>'Swimming' GROUP BY country;



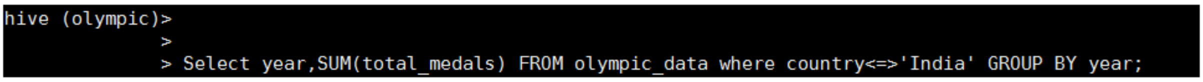
```
hive (olympic)>
              >
              > select country, SUM(total_medals) as Total_Medals FROM olympic_data where sport<=>'Swimming' GROUP BY country;
```



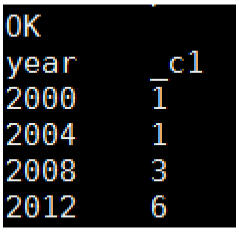
```
country total_medals
Argentina       1
Australia       163
Austria 3
Belarus 2
Brazil  8
Canada  5
China   35
Costa Rica      2
Croatia 1
Denmark 1
France  39
Germany 32
Great Britain   11
Hungary 9
Italy   16
Japan   43
Lithuania       1
Netherlands     46
Norway  2
Poland  3
Romania 6
Russia  20
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
```

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

SELECT year, SUM(total_medals) FROM olympic_data Where country<=>'India' GROUP BY year;



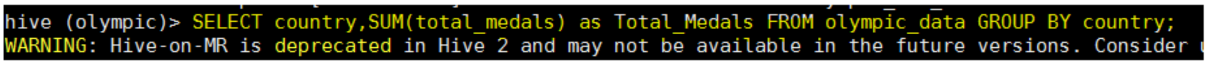
```
hive (olympic)>
              >
              > Select year,SUM(total_medals) FROM olympic_data where country<=>'India' GROUP BY year;
```



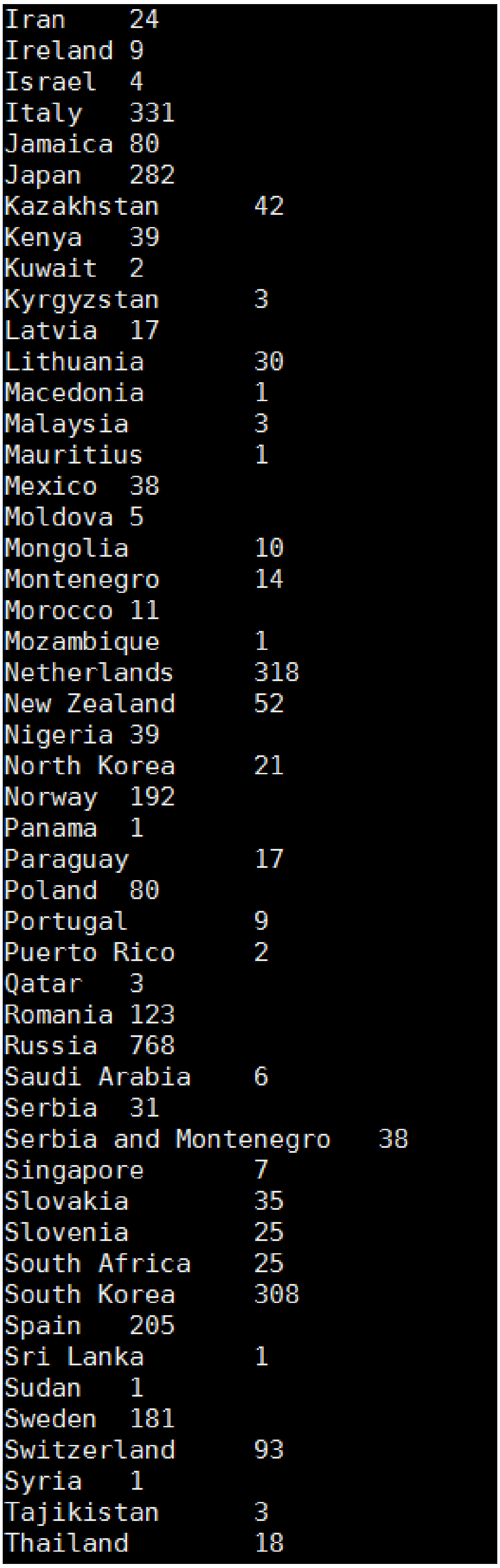
```
OK
year    _c1
2000    1
2004    1
2008    3
2012    6
```

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

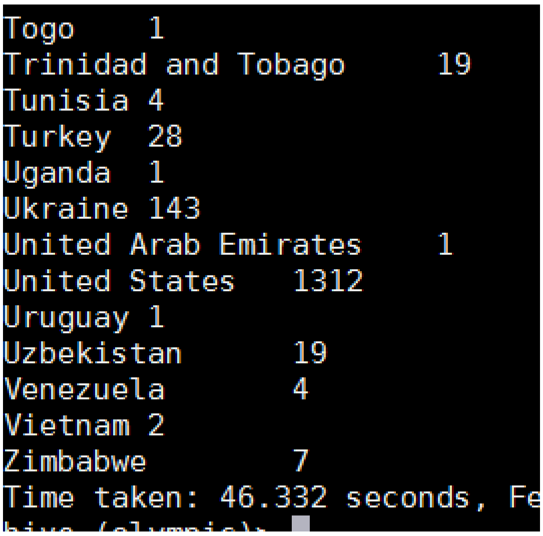
SELECT country, SUM(total_medals) as Total_Medals FROM olympic_data GROUP BY country;



```
hive (olympic)> SELECT country,SUM(total_medals) as Total_Medals FROM olympic_data GROUP BY country;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider
```



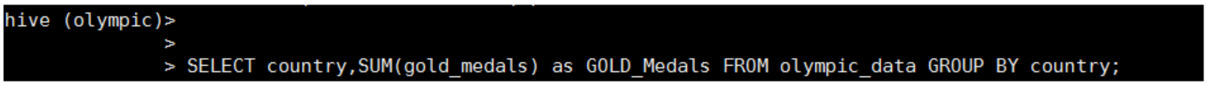
```
Iran    24
Ireland 9
Israel  4
Italy   331
Jamaica 80
Japan   282
Kazakhstan      42
Kenya   39
Kuwait  2
Kyrgyzstan      3
Latvia  17
Lithuania       30
Macedonia       1
Malaysia        3
Mauritius       1
Mexico  38
Moldova 5
Mongolia        10
Montenegro      14
Morocco 11
Mozambique      1
Netherlands     318
New Zealand     52
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
```



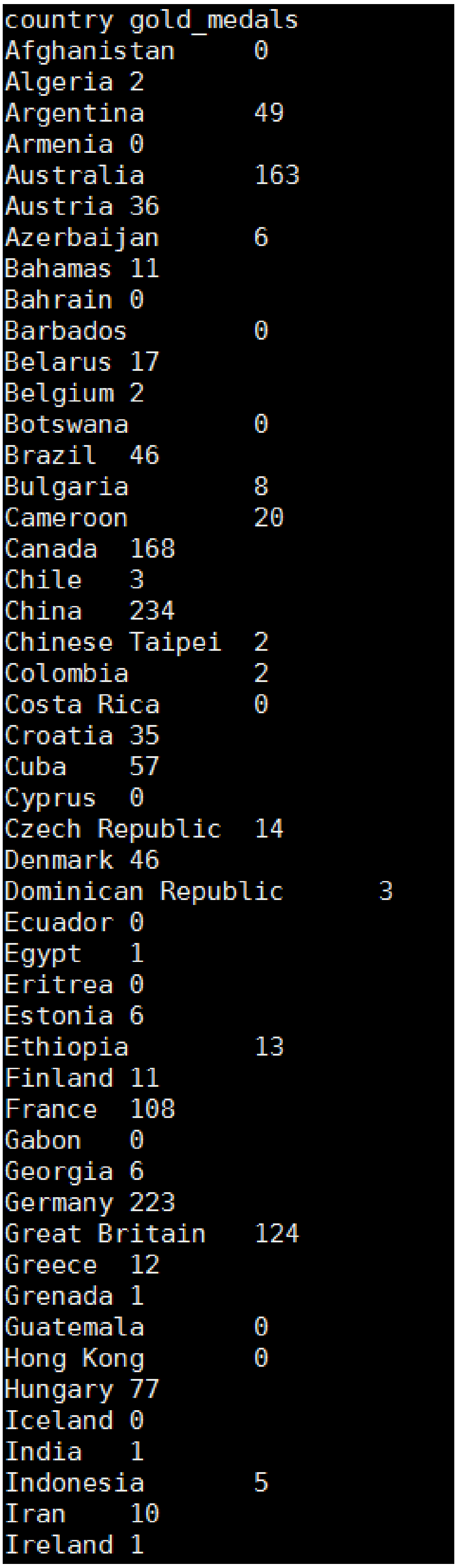
```
Togo    1
Trinidad and Tobago     19
Tunisia 4
Turkey  28
Uganda  1
Ukraine 143
United Arab Emirates    1
United States   1312
Uruguay 1
Uzbekistan      19
Venezuela       4
Vietnam 2
Zimbabwe        7
Time taken: 46.332 seconds, Fe
hive (olympic)>
```

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

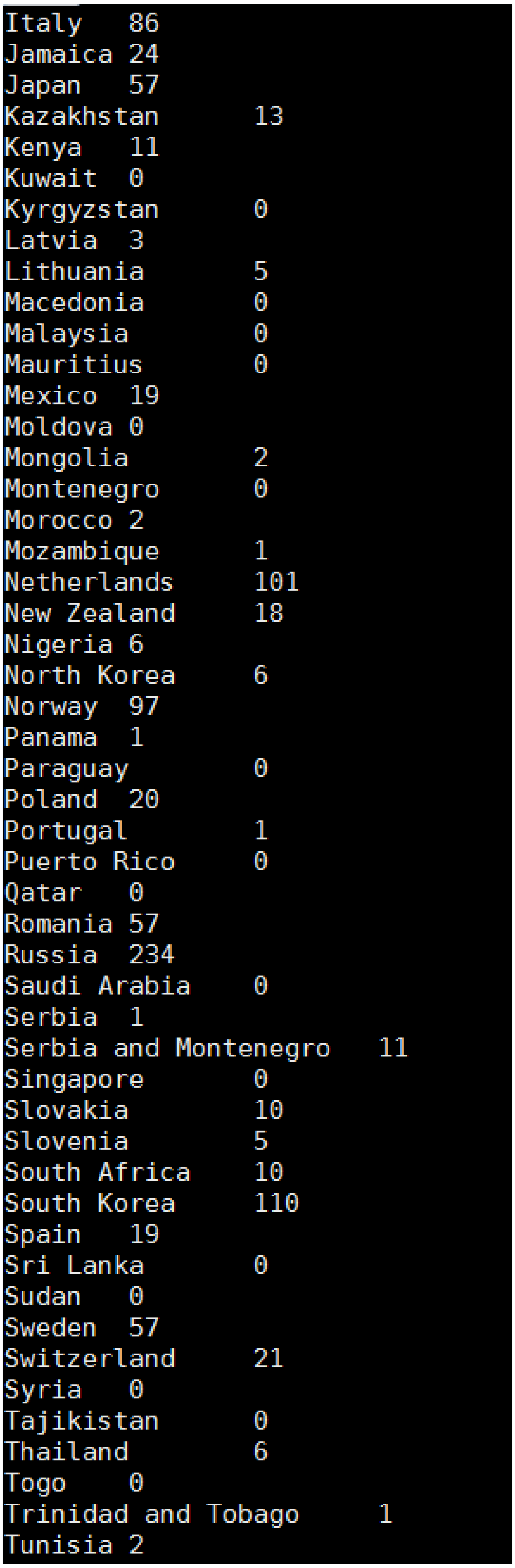
SELECT country, SUM(gold_medals) as GOLD_Medals FROM olympic_data GROUP BY country;



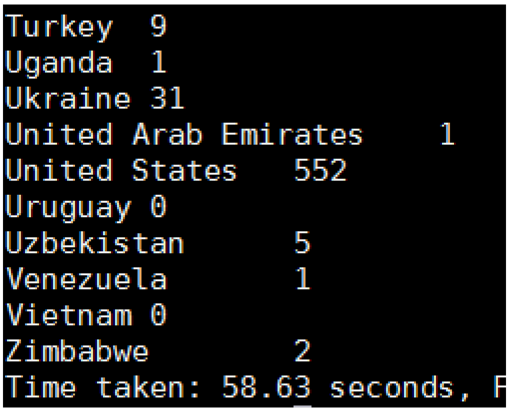
```
hive (olympic)>
              >
              > SELECT country,SUM(gold_medals) as GOLD_Medals FROM olympic_data GROUP BY country;
```



```
country gold_medals
Afghanistan     0
Algeria 2
Argentina       49
Armenia 0
Australia       163
Austria 36
Azerbaijan      6
Bahamas 11
Bahrain 0
Barbados        0
Belarus 17
Belgium 2
Botswana        0
Brazil  46
Bulgaria        8
Cameroon        20
Canada  168
Chile   3
China   234
Chinese Taipei  2
Colombia        2
Costa Rica      0
Croatia 35
Cuba    57
Cyprus  0
Czech Republic  14
Denmark 46
Dominican Republic      3
Ecuador 0
Egypt   1
Eritrea 0
Estonia 6
Ethiopia        13
Finland 11
France  108
Gabon   0
Georgia 6
Germany 223
Great Britain   124
Greece  12
Grenada 1
Guatemala       0
Hong Kong       0
Hungary 77
Iceland 0
India   1
Indonesia       5
Iran    10
Ireland 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
Panama  1
Paraguay        0
Poland  20
Portugal        1
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
Zimbabwe        2
Time taken: 58.63 seconds, F
```

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.

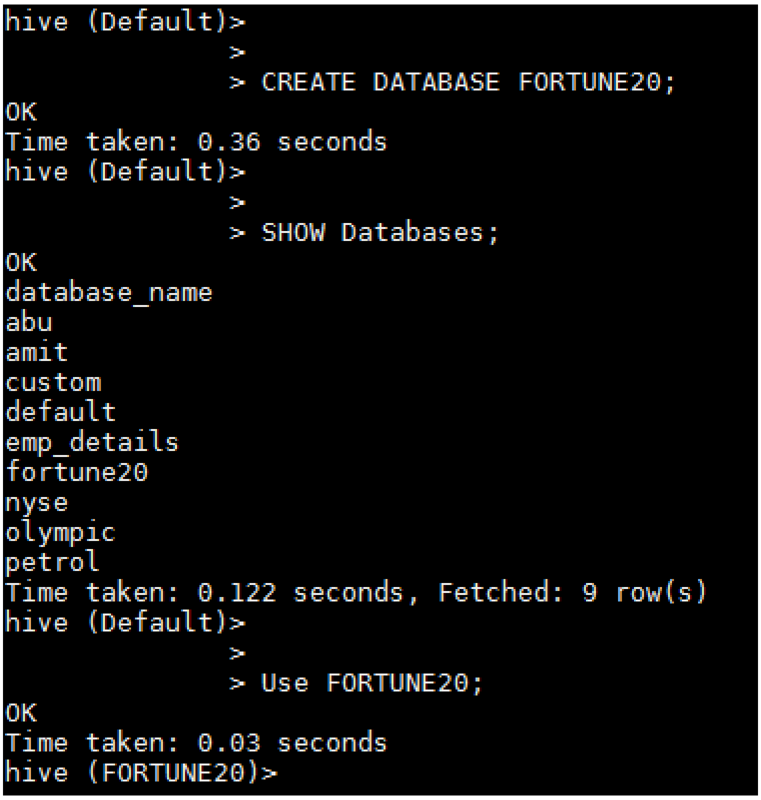
Create Database and Table

Create Database FORTUNE20

Query

CREATE DATABASE FORTUNE20

Use FORTUNE20;



```
hive (Default)>
              >
              > CREATE DATABASE FORTUNE20;
OK
Time taken: 0.36 seconds
hive (Default)>
              >
              > SHOW Databases;
OK
database_name
abu
amit
custom
default
emp_details
fortune20
nyse
olympic
petrol
Time taken: 0.122 seconds, Fetched: 9 row(s)
hive (Default)>
              >
              > Use FORTUNE20;
OK
Time taken: 0.03 seconds
hive (FORTUNE20)>
```

Create Table Fortune_company

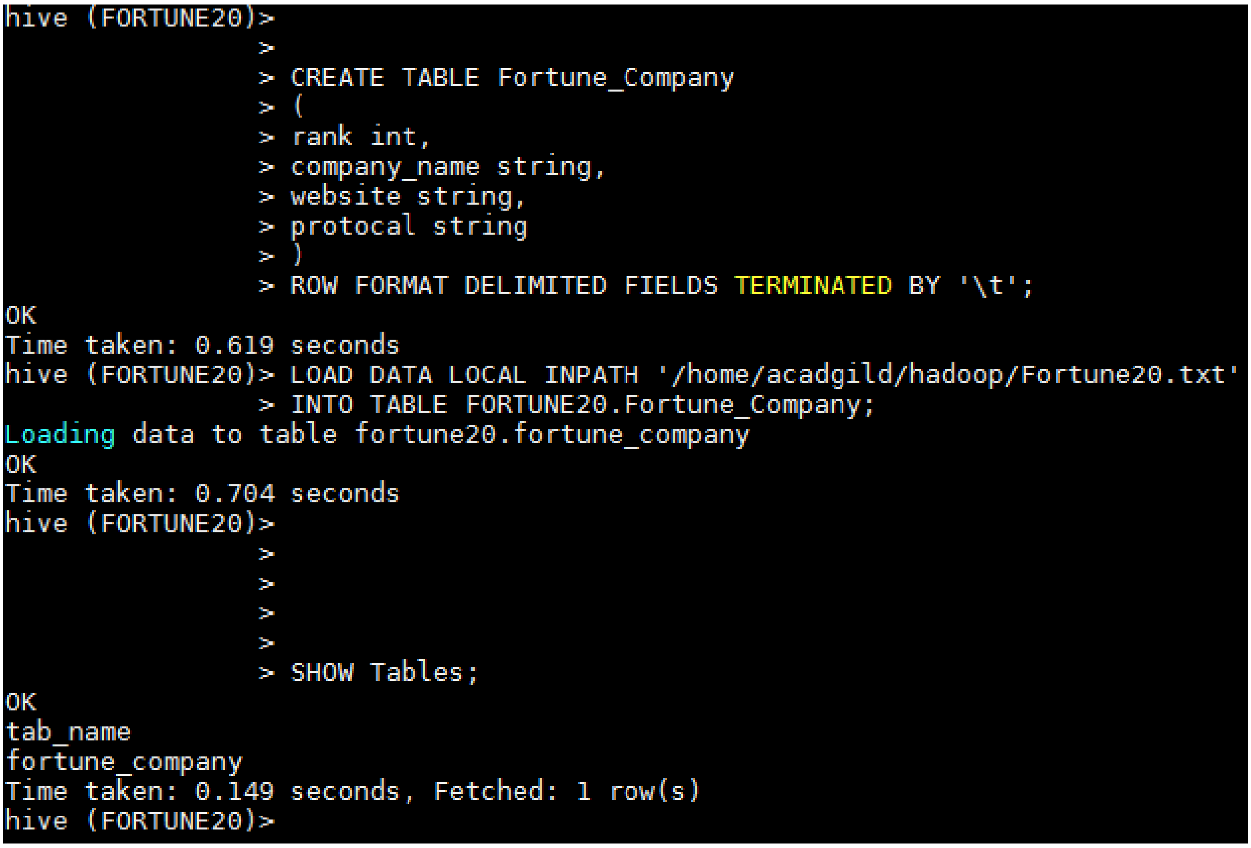
Query

CREATE TABLE fortune_company(rank int, company_name string,website string, protocal string)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

LOAD DATA LOCAL INPATH '/home/acadgild/hadoop/fortune20.txt'

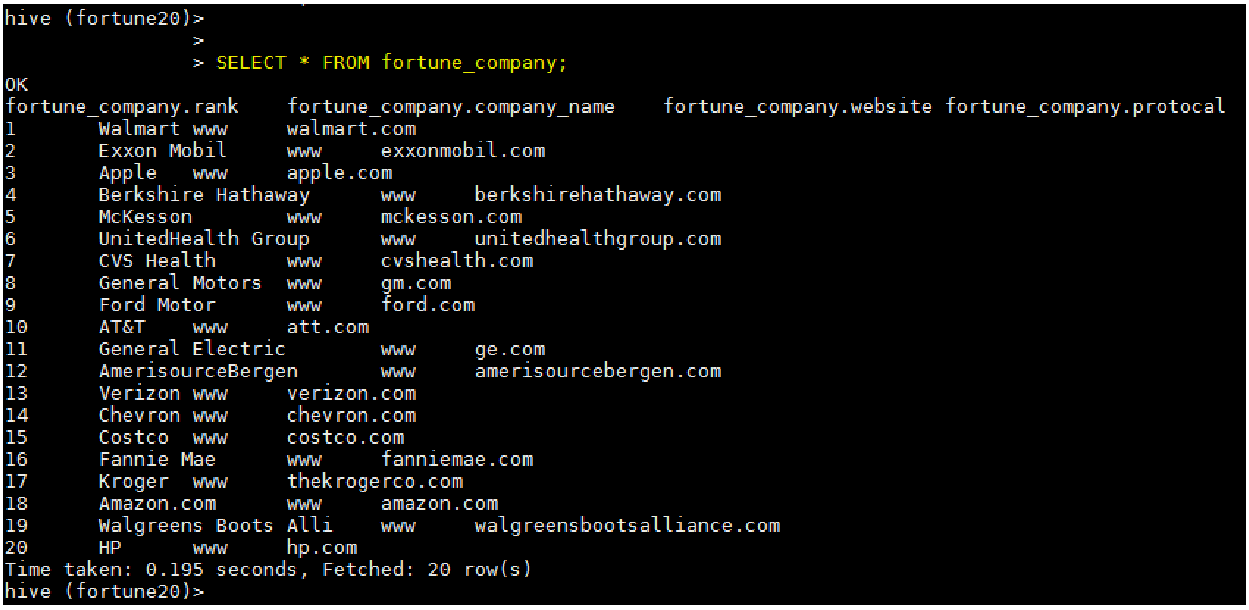
INTO TABLE fortune20.fortune_company;



```
hive (FORTUNE20)>
                > 
                > CREATE TABLE Fortune_Company
                > (
                > rank int,
                > company_name string,
                > website string,
                > protocal string
                > )
                > ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';
OK
Time taken: 0.619 seconds
hive (FORTUNE20)> LOAD DATA LOCAL INPATH '/home/acadgild/hadoop/Fortune20.txt'
                > INTO TABLE FORTUNE20.Fortune_Company;
Loading data to table fortune20.fortune_company
OK
Time taken: 0.704 seconds
hive (FORTUNE20)>
                > 
                > 
                > 
                > 
                > SHOW Tables;
OK
tab_name
fortune_company
Time taken: 0.149 seconds, Fetched: 1 row(s)
hive (FORTUNE20)>
```

Viewing the data in the table fortune_company,

SELECT * FROM fortune_company;



```
hive (fortune20)>
                >
                > SELECT * FROM fortune_company;
OK
fortune_company.rank    fortune_company.company_name    fortune_company.website fortune_company.protocal
1       Walmart www     walmart.com
2       Exxon Mobil     www     exxonmobil.com
3       Apple   www     apple.com
4       Berkshire Hathaway      www     berkshirehathaway.com
5       McKesson        www     mckesson.com
6       UnitedHealth Group      www     unitedhealthgroup.com
7       CVS Health      www     cvshealth.com
8       General Motors  www     gm.com
9       Ford Motor      www     ford.com
10      AT&T    www     att.com
11      General Electric        www     ge.com
12      AmerisourceBergen       www     amerisourcebergen.com
13      Verizon www     verizon.com
14      Chevron www     chevron.com
15      Costco  www     costco.com
16      Fannie Mae      www     fanniemae.com
17      Kroger  www     thekrogerco.com
18      Amazon.com      www     amazon.com
19      Walgreens Boots Alli    www     walgreensbootsalliance.com
20      HP      www     hp.com
Time taken: 0.195 seconds, Fetched: 20 row(s)
hive (fortune20)>
```

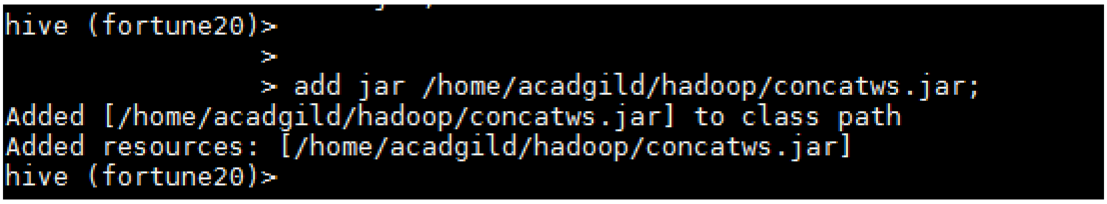
UDF Javacode

```
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 concatws 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-

HIVE UDF CONCAT_WS function

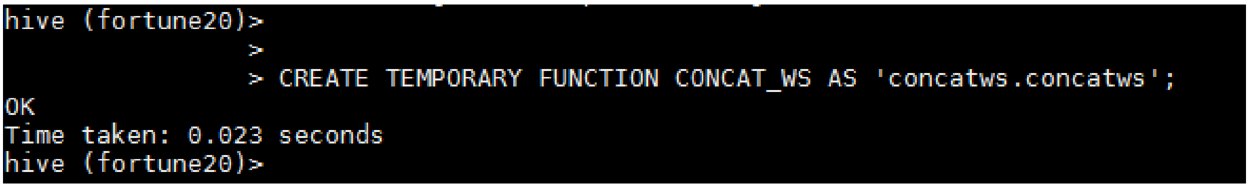
add jar /home/acadgild/hadoop/concatws.jar;



```
hive (fortune20)>
                 >
                 > add jar /home/acadgild/hadoop/concatws.jar;
Added [/home/acadgild/hadoop/concatws.jar] to class path
Added resources: [/home/acadgild/hadoop/concatws.jar]
hive (fortune20)>
```

After that we are creating a temporary function “CONCAT_WS” using below syntax-

CREATE TEMPORARY FUNCTION CONCAT_WS AS 'concatws.concatws';

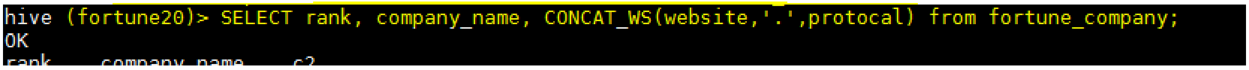


```
hive (fortune20)>
                 >
                 > CREATE TEMPORARY FUNCTION CONCAT_WS AS 'concatws.concatws';
OK
Time taken: 0.023 seconds
hive (fortune20)>
```

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,

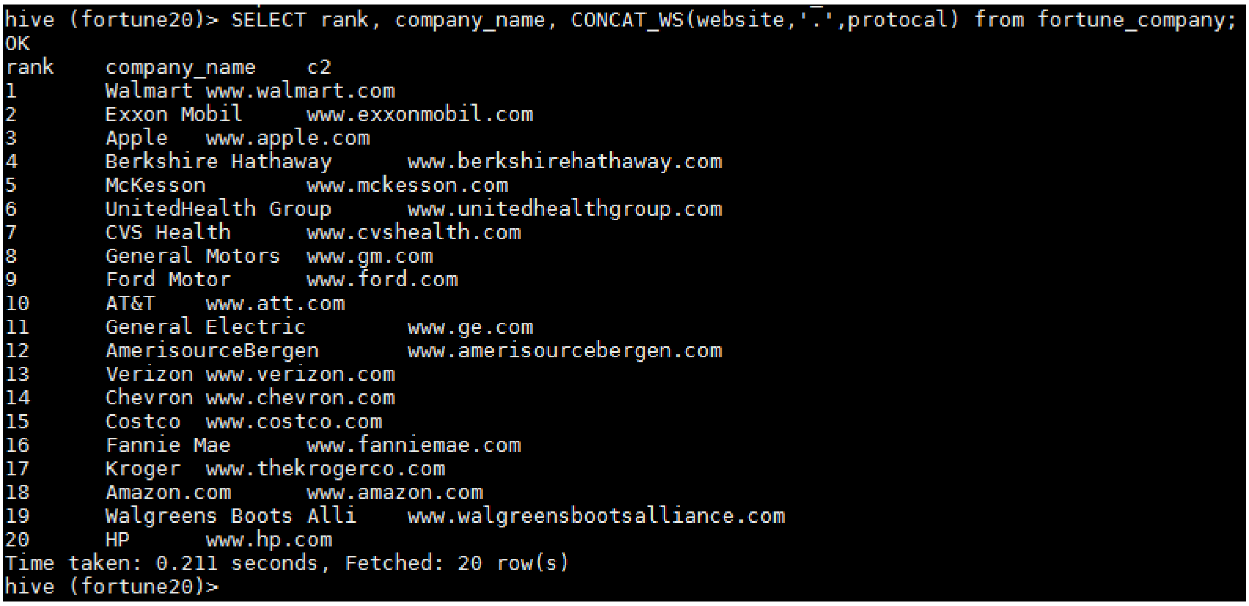
Query

SELECT rank, company_name, CONCAT_WS(website,'.',protocal) from fortune_company; SELECT rank, company_name, CONCAT_WS(website,'.',protocal) from fortune_company;



```
hive (fortune20)> SELECT rank, company_name, CONCAT_WS(website,'.',protocal) from fortune_company;
OK
rank    company_name    c2
```

Output



```
hive (fortune20)> SELECT rank, company_name, CONCAT_WS(website,'.',protocal) from fortune_company;
OK
rank    company_name    c2
1       Walmart www.walmart.com
2       Exxon Mobil     www.exxonmobil.com
3       Apple   www.apple.com
4       Berkshire Hathaway      www.berkshirehathaway.com
5       McKesson        www.mckesson.com
6       UnitedHealth Group      www.unitedhealthgroup.com
7       CVS Health      www.cvshealth.com
8       General Motors  www.gm.com
9       Ford Motor      www.ford.com
10      AT&T    www.att.com
11      General Electric        www.ge.com
12      AmerisourceBergen       www.amerisourcebergen.com
13      Verizon www.verizon.com
14      Chevron www.chevron.com
15      Costco  www.costco.com
16      Fannie Mae      www.fanniemae.com
17      Kroger  www.thekrogerco.com
18      Amazon.com      www.amazon.com
19      Walgreens Boots Alli    www.walgreensbootsalliance.com
20      HP      www.hp.com
Time taken: 0.211 seconds, Fetched: 20 row(s)
hive (fortune20)>
```

Task 3

Create a table that supports ACID Hive transactions

```
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');
```

Insert data into Hive Table

```
hive>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');
```

```
hive>select * from college
```

Update data in Hive Table

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

```
hive>select * from college
```

Delete a row from Hive Table

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

```
hive>select * from college
```