Session 9:
Advance Hive
Assignment 1

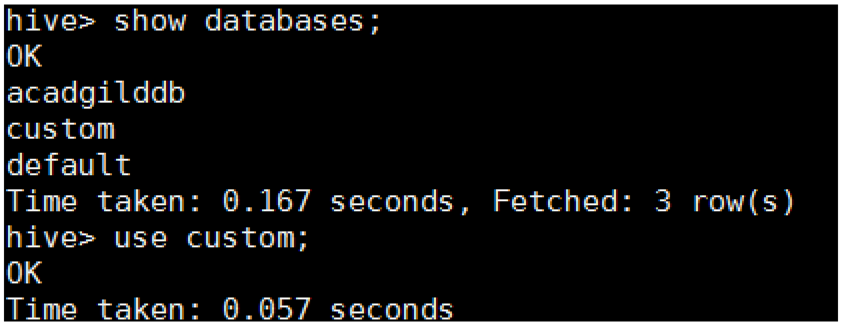
- By Prachi Mohite

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

Solution Approach

Here we will be using the database named as ‘custom’ created in HIVE assignment 8

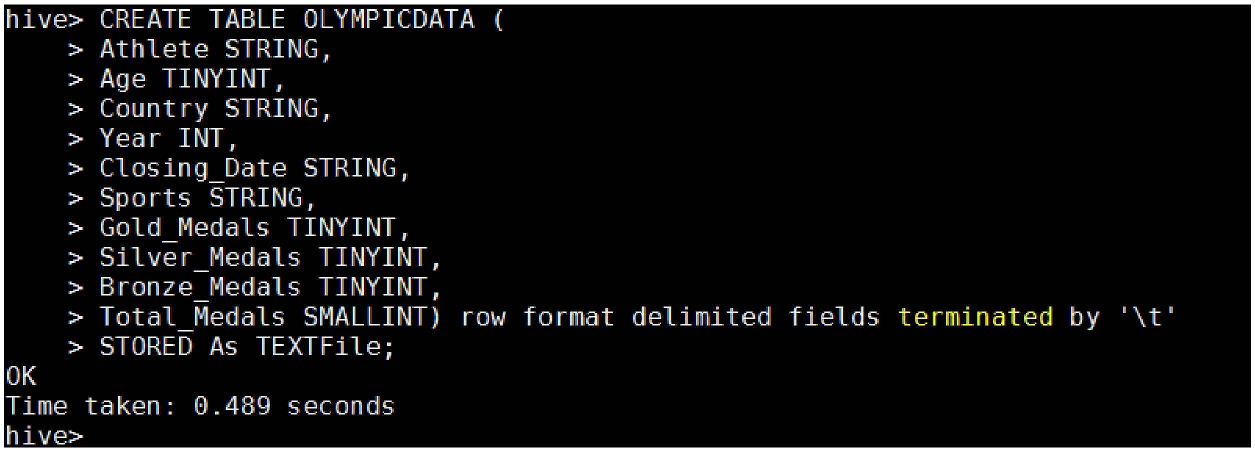


```
hive> show databases;
OK
acadgilddb
custom
default
Time taken: 0.167 seconds, Fetched: 3 row(s)
hive> use custom;
OK
Time taken: 0.057 seconds
```

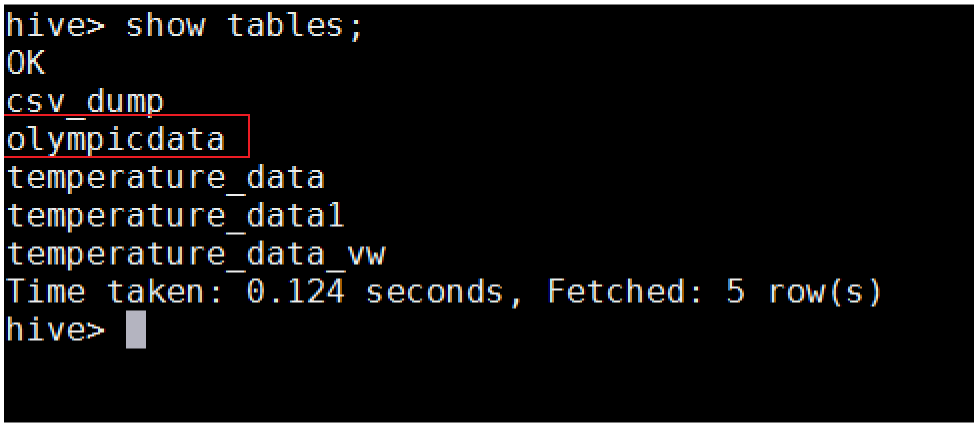
Creating a table

```
CREATE TABLE OLYMPICDATA (
Athlete STRING,
Age TINYINT,
Country STRING,
Year INT,
Closing_Date STRING,
Sports STRING,
Gold_Medals TINYINT,
Silver_Medals TINYINT,
Bronze_Medals TINYINT,
Total_Medals SMALLINT) row format delimited fields terminated by '\t'
```

STORED As TEXTFile;



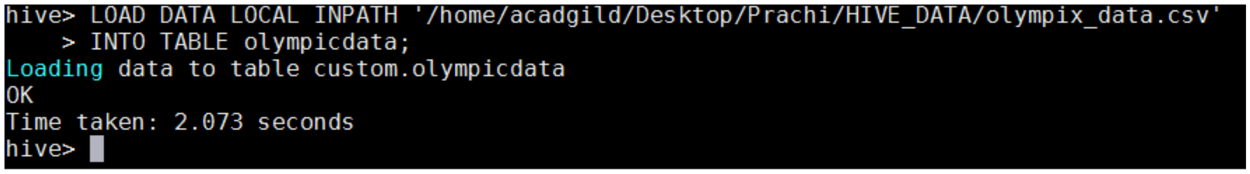
```
hive> CREATE TABLE OLYMPICDATA (
    > Athlete STRING,
    > Age TINYINT,
    > Country STRING,
    > Year INT,
    > Closing_Date STRING,
    > Sports STRING,
    > Gold_Medals TINYINT,
    > Silver_Medals TINYINT,
    > Bronze_Medals TINYINT,
    > Total_Medals SMALLINT) row format delimited fields terminated by '\t'
    > STORED As TEXTFile;
OK
Time taken: 0.489 seconds
hive>
```



```
hive> show tables;
OK
csv_dump
olympicdata
temperature_data
temperature_data1
temperature_data_vw
Time taken: 0.124 seconds, Fetched: 5 row(s)
hive>
```

Loading the data into newly created table named as olympicdata

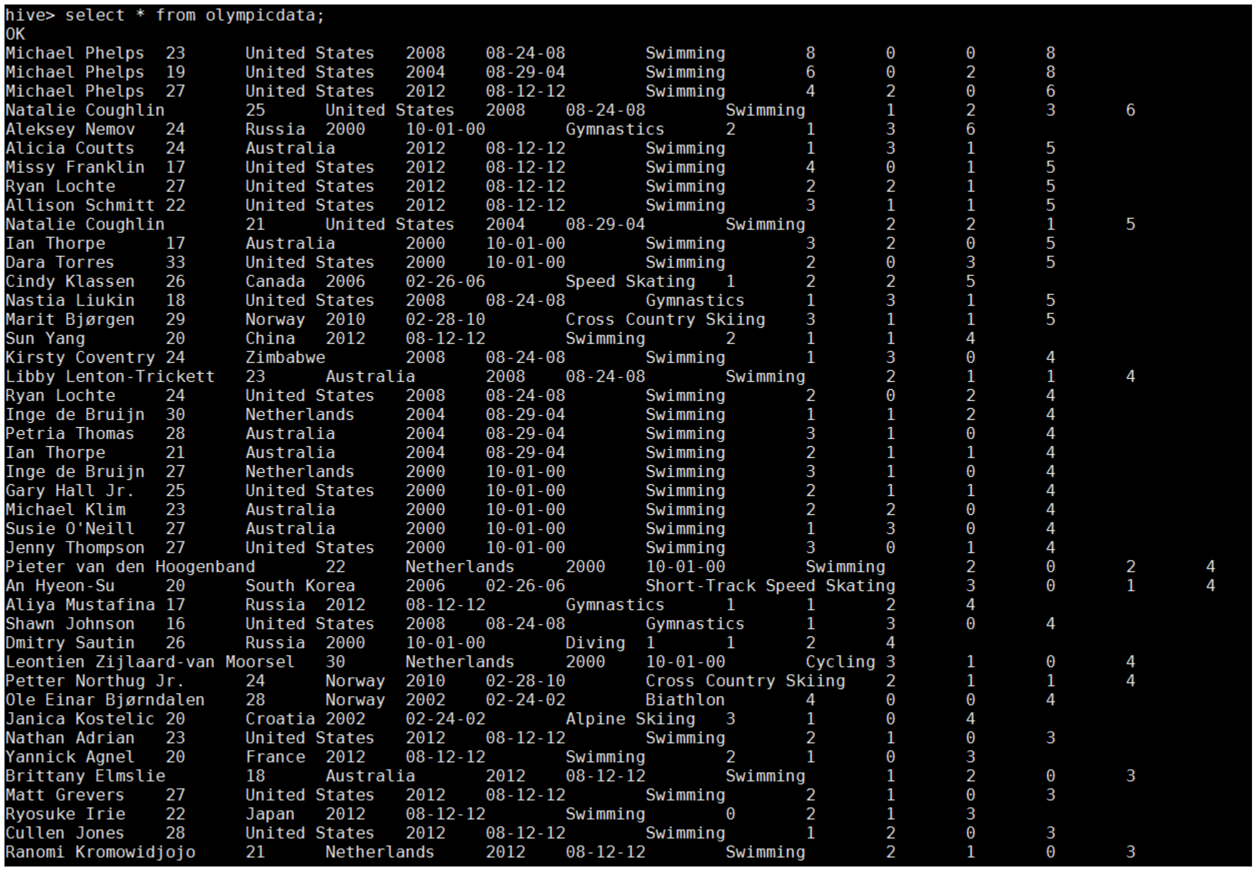
LOAD DATA LOCAL INPATH '/home/acadgild/Desktop/Prachi/HIVE_DATA/olympix_data.csv'INTO TABLE olympicdata;



```
hive> LOAD DATA LOCAL INPATH '/home/acadgild/Desktop/Prachi/HIVE_DATA/olympix_data.csv'
    > INTO TABLE olympicdata;
Loading data to table custom.olympicdata
OK
Time taken: 2.073 seconds
hive>
```

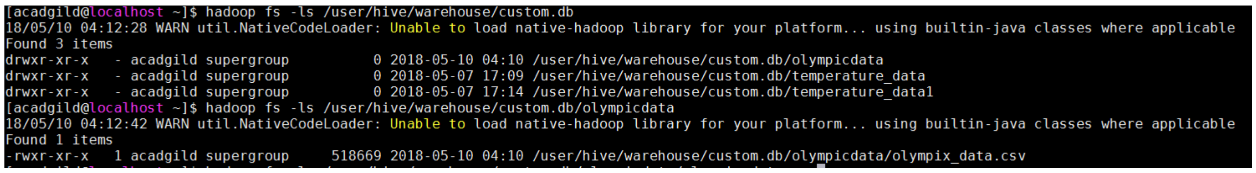
Verify if data is loaded properly or not

- With simple select command

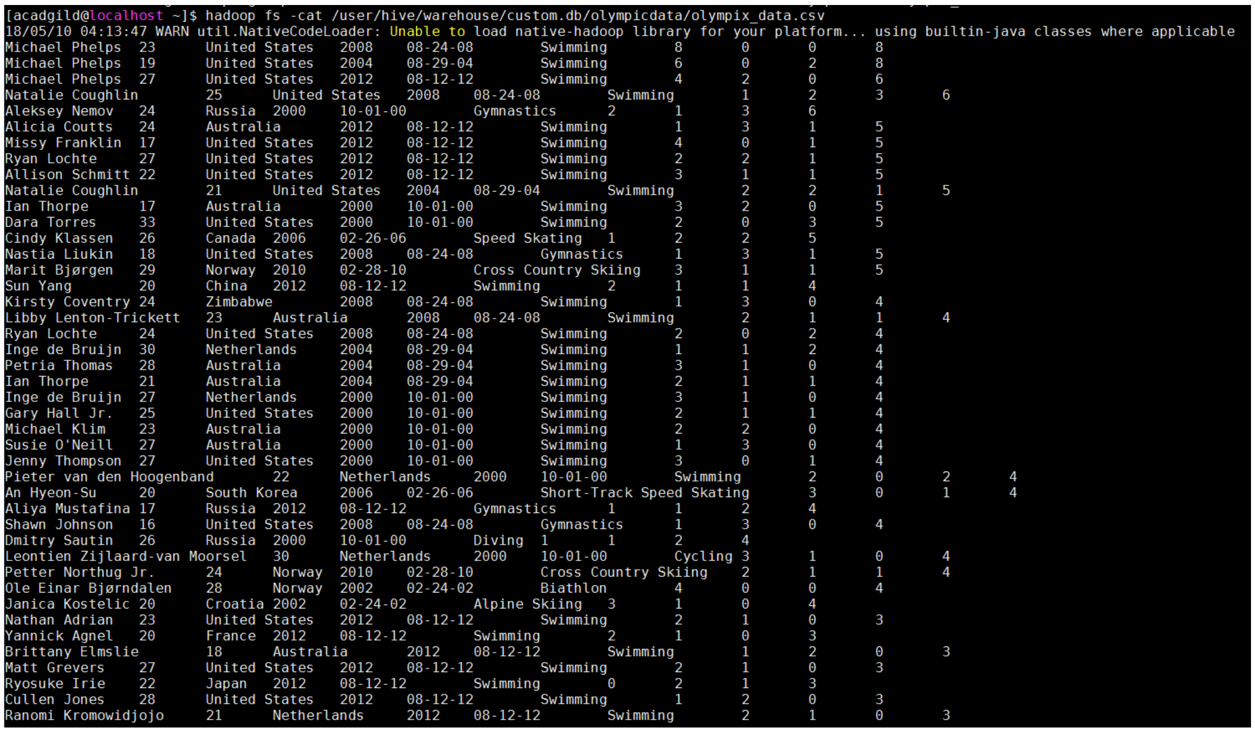


```
hive> select * from olympicdata;
OK
Michael Phelps  23      United States   2008    08-24-08        Swimming        8       0       0       8
Michael Phelps  19      United States   2004    08-29-04        Swimming        6       0       2       8
Michael Phelps  27      United States   2012    08-12-12        Swimming        4       2       0       6
Natalie Coughlin        25      United States   2008    08-24-08        Swimming        1       2       3       6
Aleksey Nemov   24      Russia  2000    10-01-00        Gymnastics      2       1       3       6
Alicia Coutts   24      Australia       2012    08-12-12        Swimming        1       3       1       5
Missy Franklin  17      United States   2012    08-12-12        Swimming        4       0       1       5
Ryan Lochte     27      United States   2012    08-12-12        Swimming        2       2       1       5
Allison Schmitt 22      United States   2012    08-12-12        Swimming        3       1       1       5
Natalie Coughlin        21      United States   2004    08-29-04        Swimming        2       2       1       5
Ian Thorpe      17      Australia       2000    10-01-00        Swimming        3       2       0       5
Dara Torres     33      United States   2000    10-01-00        Swimming        2       0       3       5
Cindy Klassen   26      Canada  2006    02-26-06        Speed Skating   1       2       2       5
Nastia Liukin   18      United States   2008    08-24-08        Gymnastics      1       3       1       5
Marit Bjørgen   29      Norway  2010    02-28-10        Cross Country Skiing    3       1       1       5
Sun Yang        20      China   2012    08-12-12        Swimming        2       1       1       4
Kirsty Coventry 24      Zimbabwe        2008    08-24-08        Swimming        1       3       0       4
Libby Lenton-Trickett   23      Australia       2008    08-24-08        Swimming        2       1       1       4
Ryan Lochte     24      United States   2008    08-24-08        Swimming        2       0       2       4
Inge de Bruijn  30      Netherlands     2004    08-29-04        Swimming        1       1       2       4
Petria Thomas   28      Australia       2004    08-29-04        Swimming        3       1       0       4
Ian Thorpe      21      Australia       2004    08-29-04        Swimming        2       1       1       4
Inge de Bruijn  27      Netherlands     2000    10-01-00        Swimming        3       1       0       4
Gary Hall Jr.   25      United States   2000    10-01-00        Swimming        2       1       1       4
Michael Klim    23      Australia       2000    10-01-00        Swimming        2       2       0       4
Susie O'Neill   27      Australia       2000    10-01-00        Swimming        1       3       0       4
Jenny Thompson  27      United States   2000    10-01-00        Swimming        3       0       1       4
Pieter van den Hoogenband       22      Netherlands     2000    10-01-00        Swimming        2       0       2       4
An Hyeon-Su     20      South Korea     2006    02-26-06        Short-Track Speed Skating       3       0       1       4
Aliya Mustafina 17      Russia  2012    08-12-12        Gymnastics      1       1       2       4
Shawn Johnson   16      United States   2008    08-24-08        Gymnastics      1       3       0       4
Dmitry Sautin   26      Russia  2000    10-01-00        Diving  1       1       2       4
Leontien Zijlaard-van Moorsel   30      Netherlands     2000    10-01-00        Cycling 3       1       0       4
Petter Northug Jr.      24      Norway  2010    02-28-10        Cross Country Skiing    2       1       1       4
Ole Einar Bjørndalen    28      Norway  2002    02-24-02        Biathlon        4       0       0       4
Janica Kostelic 20      Croatia 2002    02-24-02        Alpine Skiing   3       1       0       4
Nathan Adrian   23      United States   2012    08-12-12        Swimming        2       1       0       3
Yannick Agnel   20      France  2012    08-12-12        Swimming        2       1       0       3
Brittany Elmslie        18      Australia       2012    08-12-12        Swimming        1       2       0       3
Matt Grevers    27      United States   2012    08-12-12        Swimming        2       1       0       3
Ryosuke Irie    22      Japan   2012    08-12-12        Swimming        0       2       1       3
Cullen Jones    28      United States   2012    08-12-12        Swimming        1       2       0       3
Ranomi Kromowidjojo     21      Netherlands     2012    08-12-12        Swimming        2       1       0       3
```

From HDFS



```
[acadgild@localhost ~]$ hadoop fs -ls /user/hive/warehouse/custom.db
18/05/10 04:12:28 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 3 items
drwxr-xr-x   - acadgild supergroup          0 2018-05-10 04:10 /user/hive/warehouse/custom.db/olympicdata
drwxr-xr-x   - acadgild supergroup          0 2018-05-07 17:09 /user/hive/warehouse/custom.db/temperature_data
drwxr-xr-x   - acadgild supergroup          0 2018-05-07 17:14 /user/hive/warehouse/custom.db/temperature_data1
[acadgild@localhost ~]$ hadoop fs -ls /user/hive/warehouse/custom.db/olympicdata
18/05/10 04:12:42 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rwxr-xr-x   1 acadgild supergroup     518669 2018-05-10 04:10 /user/hive/warehouse/custom.db/olympicdata/olympix_data.csv
```



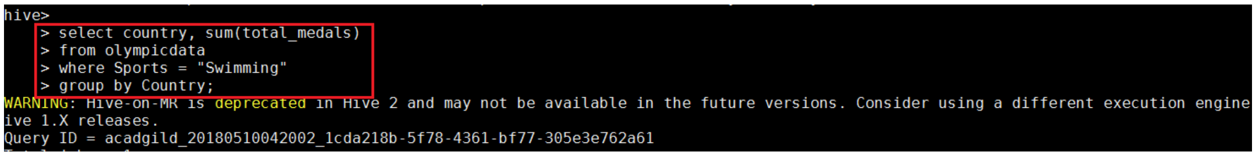
```
[acadgild@localhost ~]$ hadoop fs -cat /user/hive/warehouse/custom.db/olympicdata/olympix_data.csv
18/05/10 04:13:47 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Michael Phelps  23      United States   2008    08-24-08        Swimming        8       0       0       8
Michael Phelps  19      United States   2004    08-29-04        Swimming        6       0       2       8
Michael Phelps  27      United States   2012    08-12-12        Swimming        4       2       0       6
Natalie Coughlin        25      United States   2008    08-24-08        Swimming        1       2       3       6
Aleksey Nemov   24      Russia  2000    10-01-00        Gymnastics      2       1       3       6
Alicia Coutts   24      Australia       2012    08-12-12        Swimming        1       3       1       5
Missy Franklin  17      United States   2012    08-12-12        Swimming        4       0       1       5
Ryan Lochte     27      United States   2012    08-12-12        Swimming        2       2       1       5
Allison Schmitt 22      United States   2012    08-12-12        Swimming        3       1       1       5
Natalie Coughlin        21      United States   2004    08-29-04        Swimming        2       2       1       5
Ian Thorpe      17      Australia       2000    10-01-00        Swimming        3       2       0       5
Dara Torres     33      United States   2000    10-01-00        Swimming        2       0       3       5
Cindy Klassen   26      Canada  2006    02-26-06        Speed Skating   1       2       2       5
Nastia Liukin   18      United States   2008    08-24-08        Gymnastics      1       3       1       5
Marit Bjørgen   29      Norway  2010    02-28-10        Cross Country Skiing    3       1       1       5
Sun Yang        20      China   2012    08-12-12        Swimming        2       1       1       4
Kirsty Coventry 24      Zimbabwe        2008    08-24-08        Swimming        1       3       0       4
Libby Lenton-Trickett   23      Australia       2008    08-24-08        Swimming        2       1       1       4
Ryan Lochte     24      United States   2008    08-24-08        Swimming        2       0       2       4
Inge de Bruijn  30      Netherlands     2004    08-29-04        Swimming        1       1       2       4
Petria Thomas   28      Australia       2004    08-29-04        Swimming        3       1       0       4
Ian Thorpe      21      Australia       2004    08-29-04        Swimming        2       1       1       4
Inge de Bruijn  27      Netherlands     2000    10-01-00        Swimming        3       1       0       4
Gary Hall Jr.   25      United States   2000    10-01-00        Swimming        2       1       1       4
Michael Klim    23      Australia       2000    10-01-00        Swimming        2       2       0       4
Susie O'Neill   27      Australia       2000    10-01-00        Swimming        1       3       0       4
Jenny Thompson  27      United States   2000    10-01-00        Swimming        3       0       1       4
Pieter van den Hoogenband       22      Netherlands     2000    10-01-00        Swimming        2       0       2       4
An Hyeon-Su     20      South Korea     2006    02-26-06        Short-Track Speed Skating       3       0       1       4
Aliya Mustafina 17      Russia  2012    08-12-12        Gymnastics      1       1       2       4
Shawn Johnson   16      United States   2008    08-24-08        Gymnastics      1       3       0       4
Dmitry Sautin   26      Russia  2000    10-01-00        Diving  1       1       2       4
Leontien Zijlaard-van Moorsel   30      Netherlands     2000    10-01-00        Cycling 3       1       0       4
Petter Northug Jr.      24      Norway  2010    02-28-10        Cross Country Skiing    2       1       1       4
Ole Einar Bjørndalen    28      Norway  2002    02-24-02        Biathlon        4       0       0       4
Janica Kostelic 20      Croatia 2002    02-24-02        Alpine Skiing   3       1       0       4
Nathan Adrian   23      United States   2012    08-12-12        Swimming        2       1       0       3
Yannick Agnel   20      France  2012    08-12-12        Swimming        2       1       0       3
Brittany Elmslie        18      Australia       2012    08-12-12        Swimming        1       2       0       3
Matt Grevers    27      United States   2012    08-12-12        Swimming        2       1       0       3
Ryosuke Irie    22      Japan   2012    08-12-12        Swimming        0       2       1       3
Cullen Jones    28      United States   2012    08-12-12        Swimming        1       2       0       3
Ranomi Kromowidjojo     21      Netherlands     2012    08-12-12        Swimming        2       1       0       3
```

Task 1

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

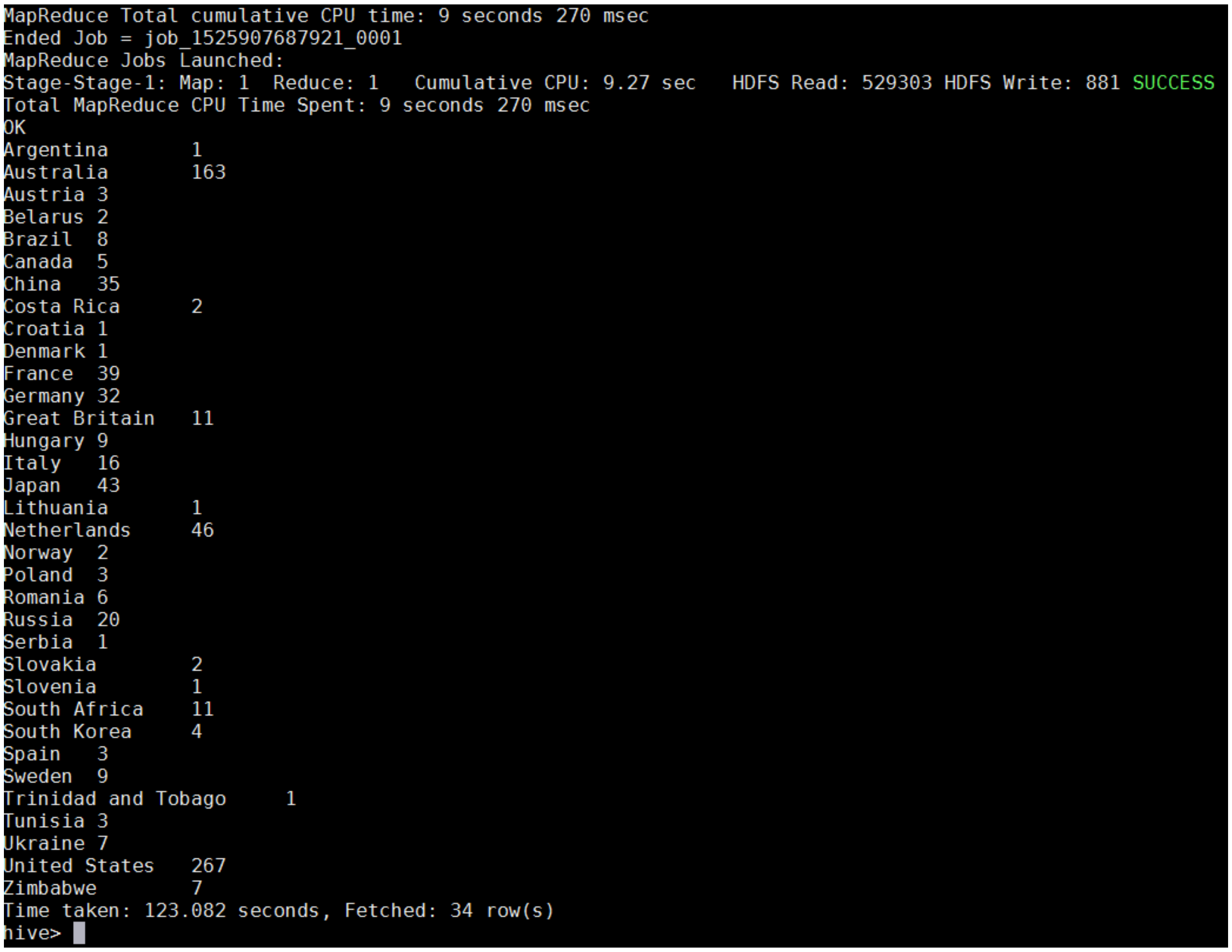
Solution Approach

- Group on Country and get sum of medals
- Where Clause on Sports column to get swimming medals



```
hive>
    > select country, sum(total_medals)
    > from olympicdata
    > where Sports = "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
ive 1.X releases.
Query ID = acadgild_20180510042002_1cda218b-5f78-4361-bf77-305e3e762a61
```

Output

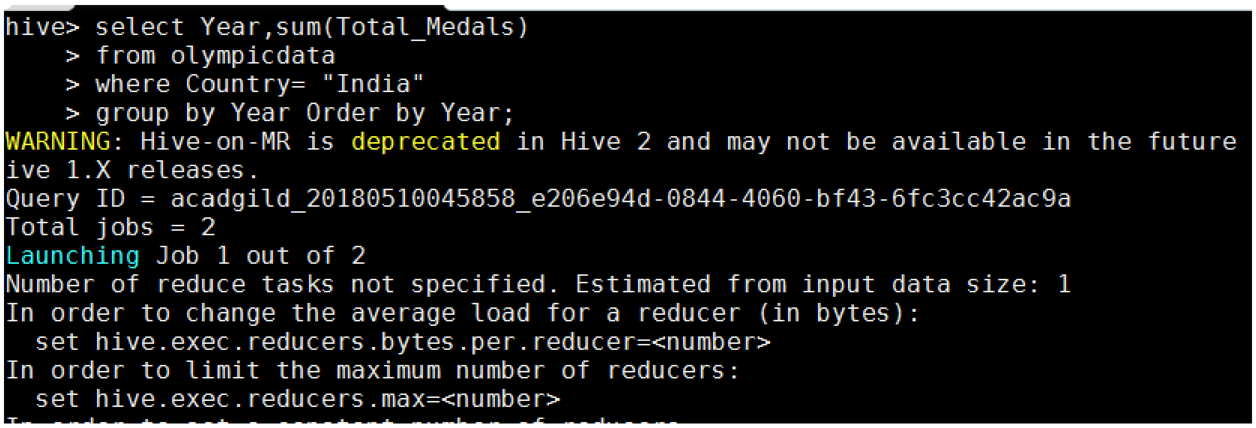


```
MapReduce Total cumulative CPU time: 9 seconds 270 msec
Ended Job = job_1525907687921_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 9.27 sec   HDFS Read: 529303 HDFS Write: 881 SUCCESS
Total MapReduce CPU Time Spent: 9 seconds 270 msec
OK
Argentina       1
Australia       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
Time taken: 123.082 seconds, Fetched: 34 row(s)
hive>
```

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

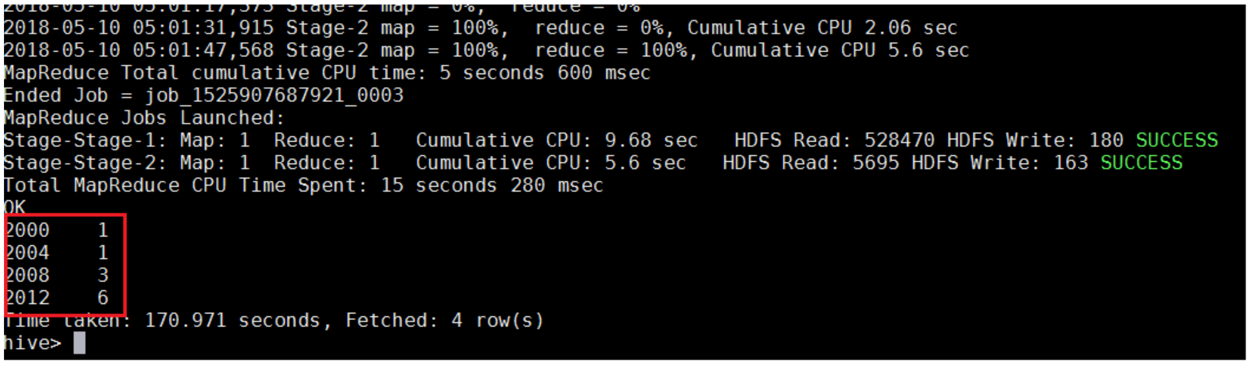
Solution Approach

- Group on Year
- Where Clause on Country column to get India's medals



```
hive> select Year,sum(Total_Medals)
    > from olympicdata
    > where Country= "India"
    > group by Year Order by Year;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future
ive 1.X releases.
Query ID = acadgild_20180510045858_e206e94d-0844-4060-bf43-6fc3cc42ac9a
Total jobs = 2
Launching Job 1 out of 2
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>
```

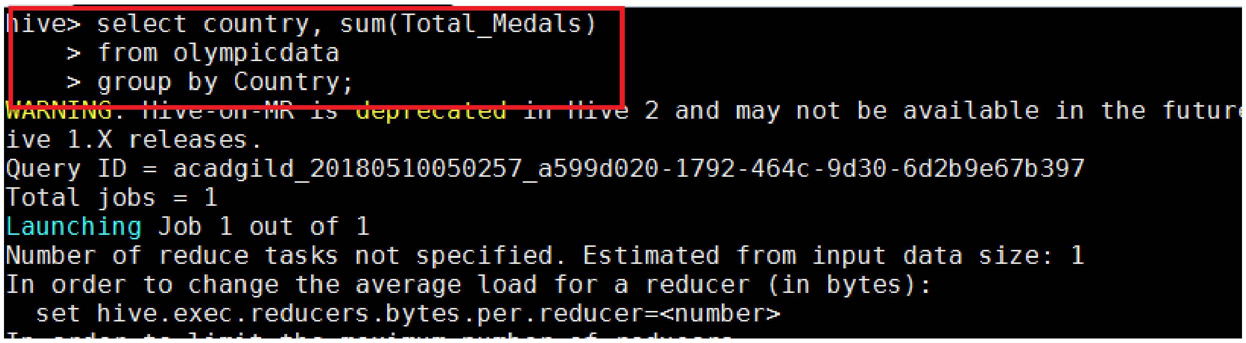
Output


```
2018-05-10 05:01:31,915 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 2.06 sec
2018-05-10 05:01:47,568 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 5.6 sec
MapReduce Total cumulative CPU time: 5 seconds 600 msec
Ended Job = job_1525907687921_0003
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 9.68 sec   HDFS Read: 528470 HDFS Write: 180 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 5.6 sec   HDFS Read: 5695 HDFS Write: 163 SUCCESS
Total MapReduce CPU Time Spent: 15 seconds 280 msec
OK
2000    1
2004    1
2008    3
2012    6
Time taken: 170.971 seconds, Fetched: 4 row(s)
hive>
```

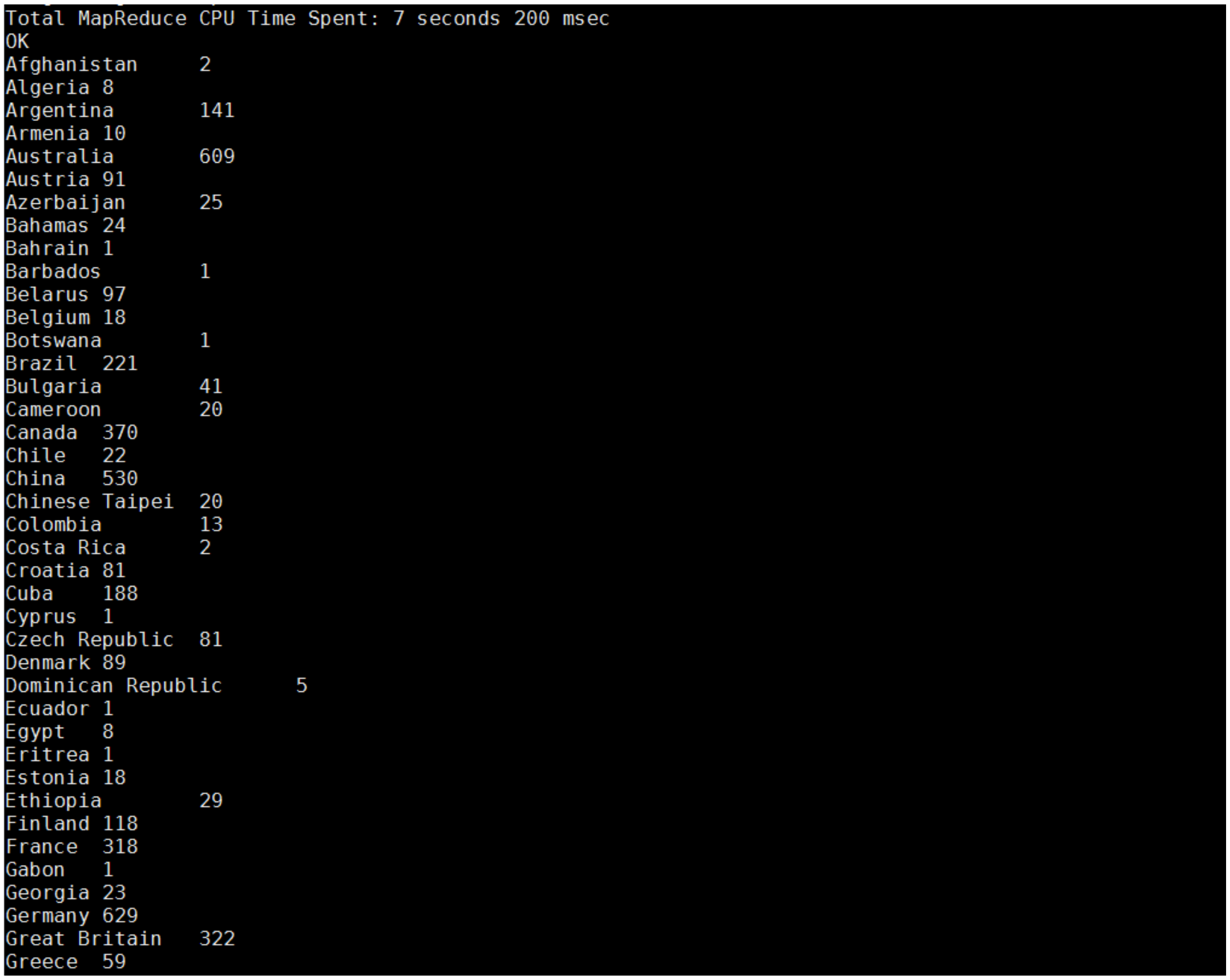
1.3. Write a Hive Program to find the total number of medals each country won.

Solution Approach

- Group on Country and sum of total medals


```
hive> select country, sum(Total_Medals)
    > from olympicdata
    > group by Country;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the futur
ive 1.X releases.
Query ID = acadgild_20180510050257_a599d020-1792-464c-9d30-6d2b9e67b397
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>
```

Output

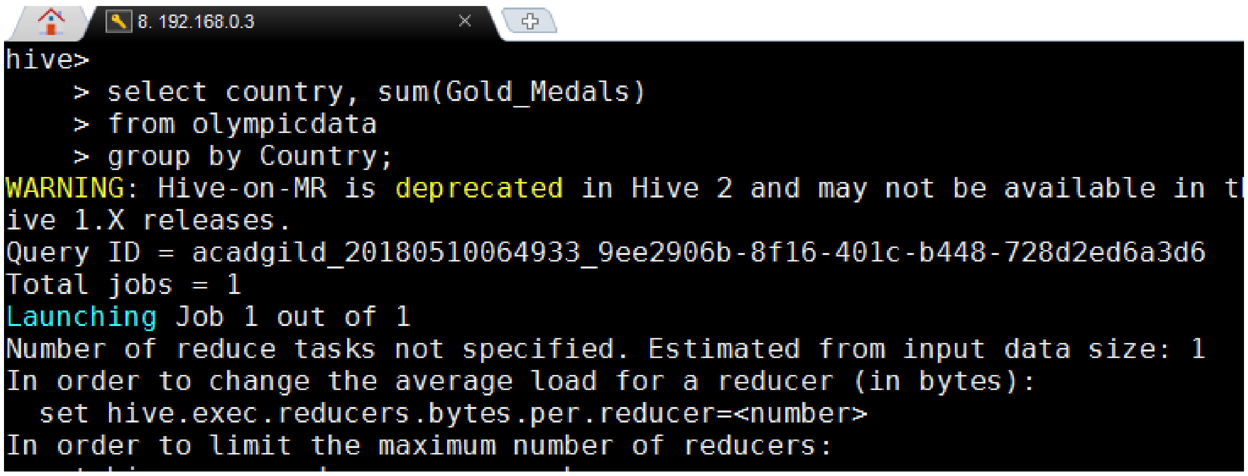


```
Total MapReduce CPU Time Spent: 7 seconds 200 msec
OK
Afghanistan     2
Algeria 8
Argentina       141
Armenia 10
Australia       609
Austria 91
Azerbaijan      25
Bahamas 24
Bahrain 1
Barbados        1
Belarus 97
Belgium 18
Botswana        1
Brazil  221
Bulgaria        41
Cameroon        20
Canada  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
```

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

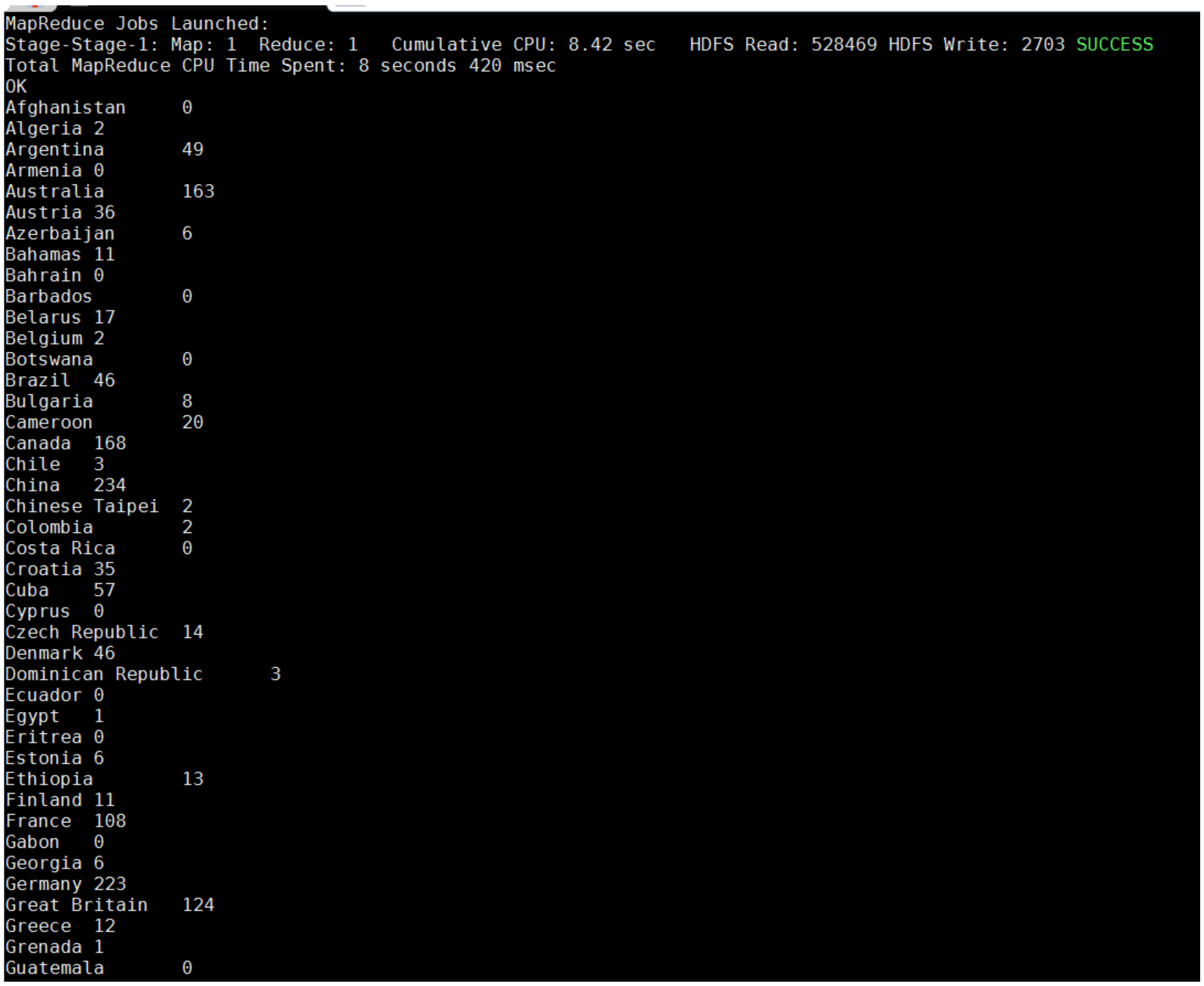
Solution Approach

- Group on Country and get sum of gold medals



```
hive>
    > select country, sum(Gold_Medals)
    > from olympicdata
    > group by Country;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in t
ive 1.X releases.
Query ID = acadgild_20180510064933_9ee2906b-8f16-401c-b448-728d2ed6a3d6
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:
```

Output



```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 8.42 sec   HDFS Read: 528469 HDFS Write: 2703 SUCCESS
Total MapReduce CPU Time Spent: 8 seconds 420 msec
OK
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
```

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.

UDFs

User Defined Functions, also known as UDF, allow you to create custom functions to process records or groups of records.

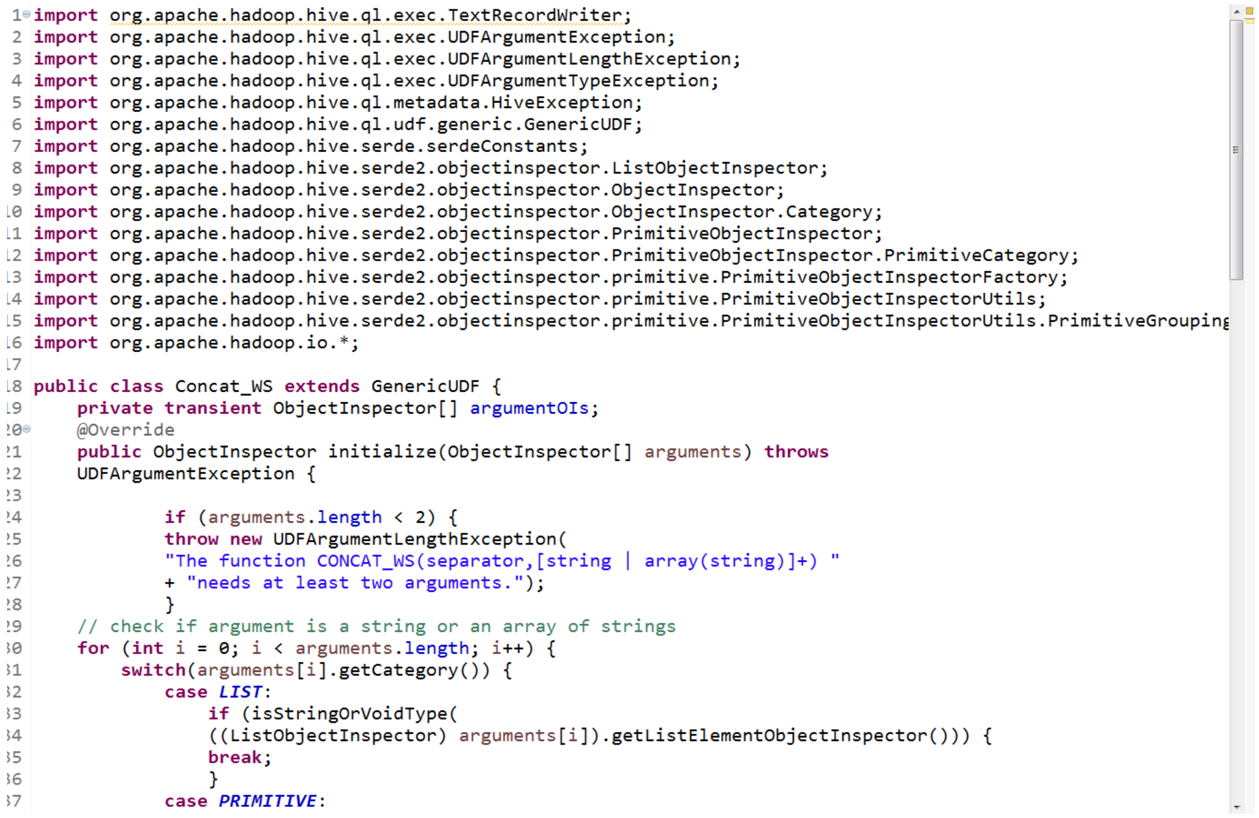
Hive has

- UDF
    - A UDF processes one or several columns of one row and outputs one value.
    - UDAF - User-Defined Aggregation Functions
        - Aggregate functions perform a calculation on a set of values and return a single value. An aggregate function is more difficult to write than a regular UDF. Values are aggregated in chunks (potentially across many tasks), so the implementation has to be capable of combiningpartial aggregations into a final result.
    - Simple UDF
        - Where all complex datatypes cannot be handled
        - Reduced performance due to use of reflection: each call of the evaluate method is reflective. Furthermore, all arguments are evaluated and parsed.
    - Generic UDF
        - A generic UDF is written by extending the GenericUDF class.
        - All complex parameters are supported (even nested ones like array<array>
        - Variable number of arguments are supported

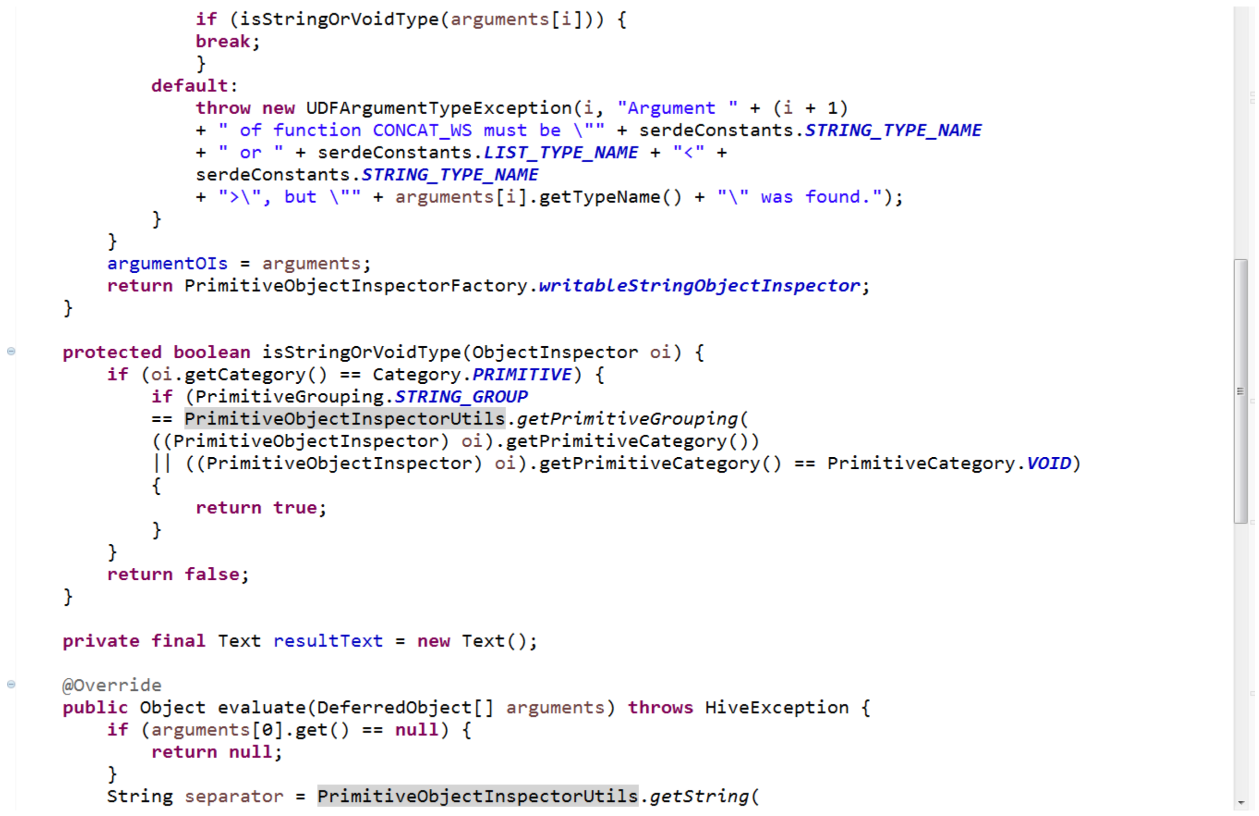
Solution Approach

Here we will create Generic UDF as Generic UDF allows to accept Complex Datatypes like Array of string

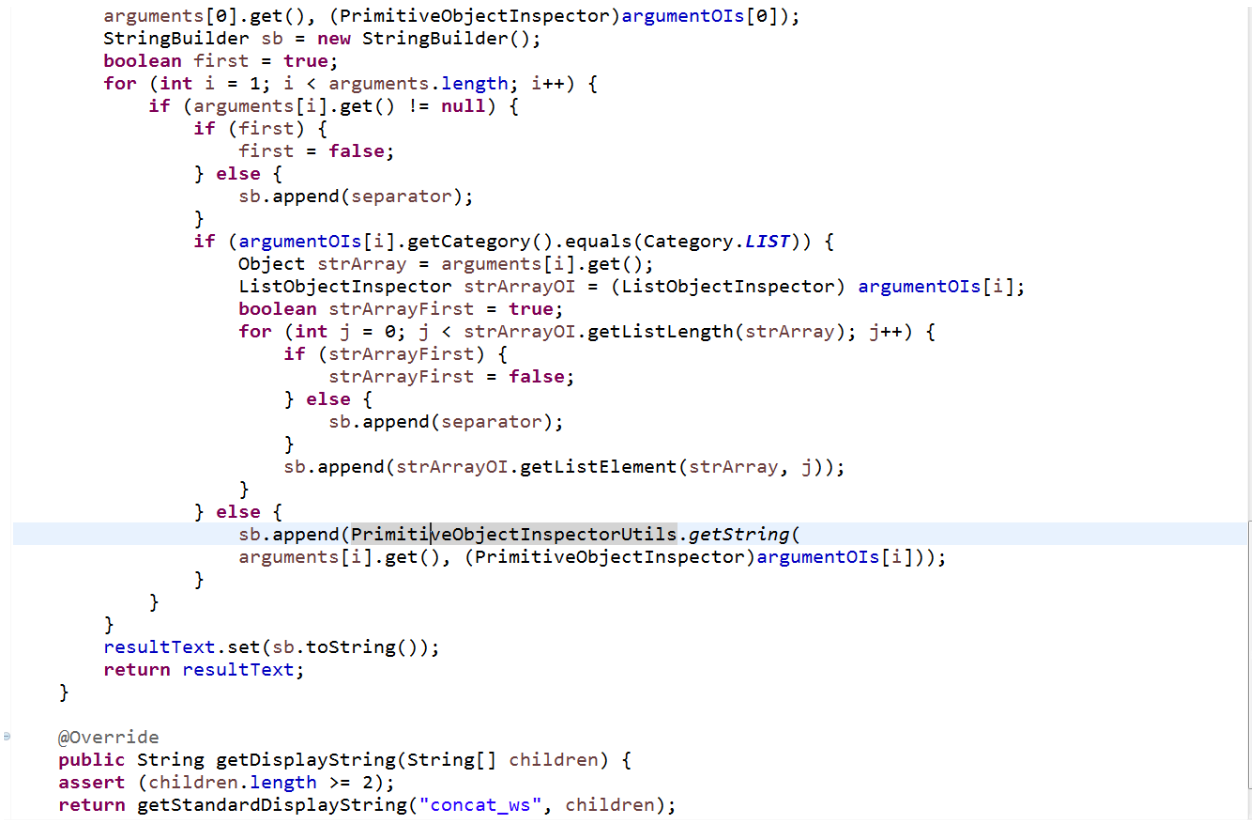
- To create a GenericUDF, need to inherit the abstract class '**GenericUDF'**
- Need to overwrite below methods
- **ObjectInspector initialize(ObjectInspector[] arguments)**
    - To verify what type of data types / arguments are accepted in UDF
    - Set up and return an ObjectInspector for the type of the output of the UDF
    - Store in global variables the ObjectInspectors for the elements of the input
    - Set up the storage variable for the output
- **Object evaluate(DeferredObject[] arguments)**
    - Actual functionality of generic UDF which should get executed once UDF is called through hive shell / script
- **String getDisplayString(String[] children)**
    - returns the string that will be returned when explain is used



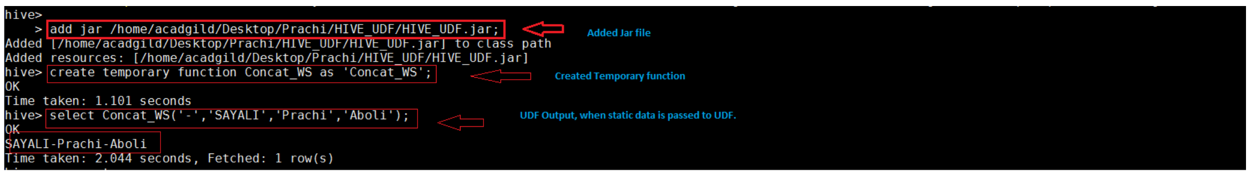
```java
import org.apache.hadoop.hive.ql.exec.TextRecordWriter;
import org.apache.hadoop.hive.ql.exec.UDFArgumentException;
import org.apache.hadoop.hive.ql.exec.UDFArgumentLengthException;
import org.apache.hadoop.hive.ql.exec.UDFArgumentTypeException;
import org.apache.hadoop.hive.ql.metadata.HiveException;
import org.apache.hadoop.hive.ql.udf.generic.GenericUDF;
import org.apache.hadoop.hive.serde.serdeConstants;
import org.apache.hadoop.hive.serde2.objectinspector.ListObjectInspector;
import org.apache.hadoop.hive.serde2.objectinspector.ObjectInspector;
import org.apache.hadoop.hive.serde2.objectinspector.ObjectInspector.Category;
import org.apache.hadoop.hive.serde2.objectinspector.PrimitiveObjectInspector;
import org.apache.hadoop.hive.serde2.objectinspector.PrimitiveObjectInspector.PrimitiveCategory;
import org.apache.hadoop.hive.serde2.objectinspector.primitive.PrimitiveObjectInspectorFactory;
import org.apache.hadoop.hive.serde2.objectinspector.primitive.PrimitiveObjectInspectorUtils;
import org.apache.hadoop.hive.serde2.objectinspector.primitive.PrimitiveObjectInspectorUtils.PrimitiveGrouping
import org.apache.hadoop.io.*;

public class Concat_WS extends GenericUDF {
    private transient ObjectInspector[] argumentOIs;
    @Override
    public ObjectInspector initialize(ObjectInspector[] arguments) throws
    UDFArgumentException {

            if (arguments.length < 2) {
            throw new UDFArgumentLengthException(
            "The function CONCAT_WS(separator,[string | array(string)]+) "
            + "needs at least two arguments.");
            }
    // check if argument is a string or an array of strings
    for (int i = 0; i < arguments.length; i++) {
        switch(arguments[i].getCategory()) {
            case LIST:
                if (isStringOrVoidType(
                ((ListObjectInspector) arguments[i]).getListElementObjectInspector())) {
                break;
                }
            case PRIMITIVE:
```



```java
                if (isStringOrVoidType(arguments[i])) {
                break;
                }
            default:
                throw new UDFArgumentTypeException(i, "Argument " + (i + 1)
                + " of function CONCAT_WS must be \"" + serdeConstants.STRING_TYPE_NAME
                + " or " + serdeConstants.LIST_TYPE_NAME + "<" +
                serdeConstants.STRING_TYPE_NAME
                + ">\", but \"" + arguments[i].getTypeName() + "\" was found.");
            }
        }
        argumentOIs = arguments;
        return PrimitiveObjectInspectorFactory.writableStringObjectInspector;
    }

    protected boolean isStringOrVoidType(ObjectInspector oi) {
        if (oi.getCategory() == Category.PRIMITIVE) {
            if (PrimitiveGrouping.STRING_GROUP
            == PrimitiveObjectInspectorUtils.getPrimitiveGrouping(
            ((PrimitiveObjectInspector) oi).getPrimitiveCategory())
            || ((PrimitiveObjectInspector) oi).getPrimitiveCategory() == PrimitiveCategory.VOID)
            {
                return true;
            }
        }
        return false;
    }

    private final Text resultText = new Text();

    @Override
    public Object evaluate(DeferredObject[] arguments) throws HiveException {
        if (arguments[0].get() == null) {
            return null;
        }
        String separator = PrimitiveObjectInspectorUtils.getString(
```



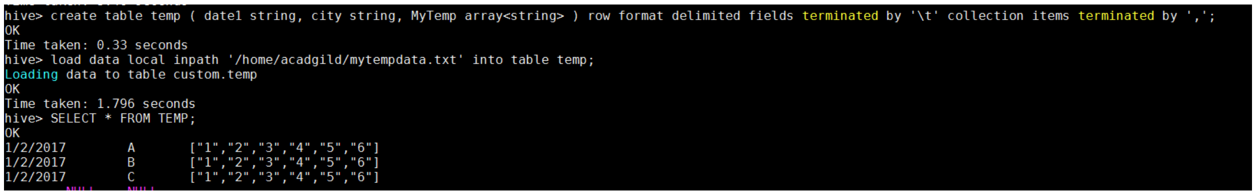
```java
        arguments[0].get(), (PrimitiveObjectInspector)argumentOIs[0]);
        StringBuilder sb = new StringBuilder();
        boolean first = true;
        for (int i = 1; i < arguments.length; i++) {
            if (arguments[i].get() != null) {
                if (first) {
                    first = false;
                } else {
                    sb.append(separator);
                }
                if (argumentOIs[i].getCategory().equals(Category.LIST)) {
                    Object strArray = arguments[i].get();
                    ListObjectInspector strArrayOI = (ListObjectInspector) argumentOIs[i];
                    boolean strArrayFirst = true;
                    for (int j = 0; j < strArrayOI.getListLength(strArray); j++) {
                        if (strArrayFirst) {
                            strArrayFirst = false;
                        } else {
                            sb.append(separator);
                        }
                        sb.append(strArrayOI.getListElement(strArray, j));
                    }
                } else {
                    sb.append(PrimitiveObjectInspectorUtils.getString(
                    arguments[i].get(), (PrimitiveObjectInspector)argumentOIs[i]));
                }
            }
        }
        resultText.set(sb.toString());
        return resultText;
    }

    @Override
    public String getDisplayString(String[] children) {
    assert (children.length >= 2);
    return getStandardDisplayString("concat_ws", children);
```



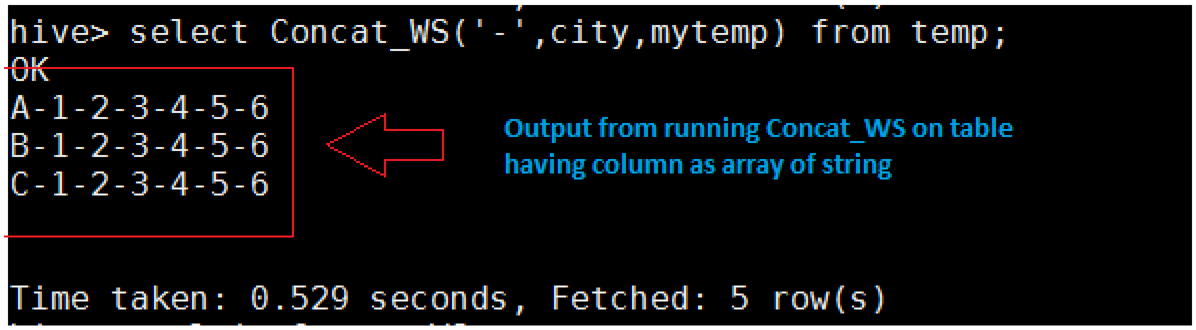
```
hive>
    > add jar /home/acadgild/Desktop/Prachi/HIVE_UDF/HIVE_UDF.jar;
Added [/home/acadgild/Desktop/Prachi/HIVE_UDF/HIVE_UDF.jar] to class path
Added resources: [/home/acadgild/Desktop/Prachi/HIVE_UDF/HIVE_UDF.jar]
hive> create temporary function Concat_WS as 'Concat_WS';
OK
Time taken: 1.101 seconds
hive> select Concat_WS('-','SAYALI','Prachi','Aboli');
OK
SAYALI-Prachi-Aboli
Time taken: 2.044 seconds, Fetched: 1 row(s)
```

Added Jar file
Created Temporary function
UDF Output, when static data is passed to UDF.

Executing UDF on Table which has one column as array of strings



```
hive> create table temp ( date1 string, city string, MyTemp array<string> ) row format delimited fields terminated by '\t' collection items terminated by ',';
OK
Time taken: 0.33 seconds
hive> load data local inpath '/home/acadgild/mytempdata.txt' into table temp;
Loading data to table custom.temp
OK
Time taken: 1.796 seconds
hive> SELECT * FROM TEMP;
OK
1/2/2017        A       ["1","2","3","4","5","6"]
1/2/2017        B       ["1","2","3","4","5","6"]
1/2/2017        C       ["1","2","3","4","5","6"]
```



```
hive> select Concat_WS('-',city,mytemp) from temp;
OK
A-1-2-3-4-5-6
B-1-2-3-4-5-6
C-1-2-3-4-5-6

Time taken: 0.529 seconds, Fetched: 5 row(s)
```

Output from running Concat_WS on table having column as array of 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.

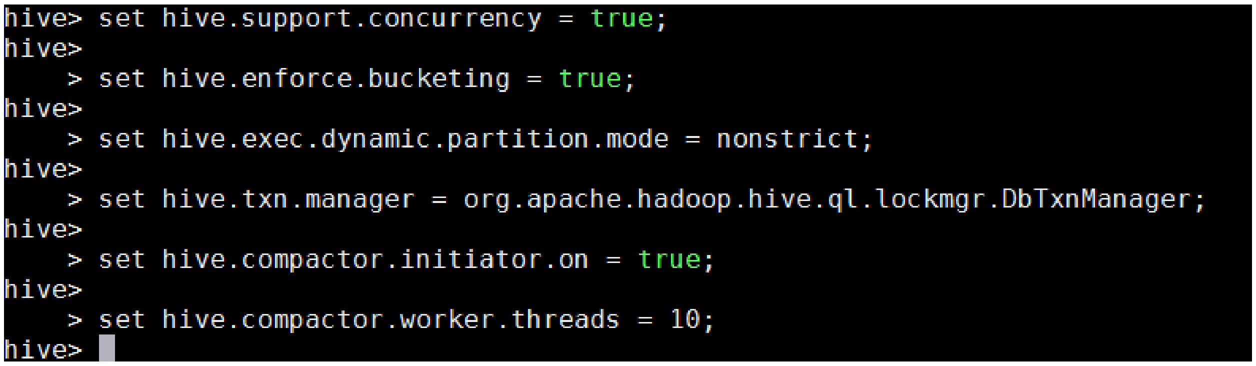
Transactions are provided at the row-level in Hive 0.14. The different row-level transactions available in Hive 0.14 are as follows:

1. Insert
2. Delete
3. Update

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 transaction features present in Hive needs to be turned on, as by default they are turned off.

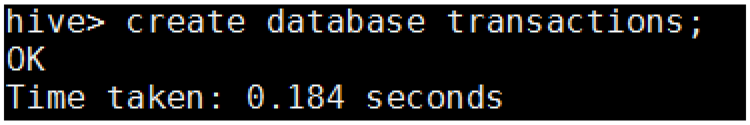
- set hive.support.concurrency = true;
  - Defualt value is false
- set hive.enforce.bucketing = true;
  - Ensures we can create buckets on a table
- set hive.exec.dynamic.partition.mode = nonstrict;
  - This is to set dynamic partition mode on
- set hive.txn.manager = org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;
- set hive.compactor.initiator.on = true;
- set hive.compactor.worker.threads = a positive number on at least one instance of the Thrift metastore service



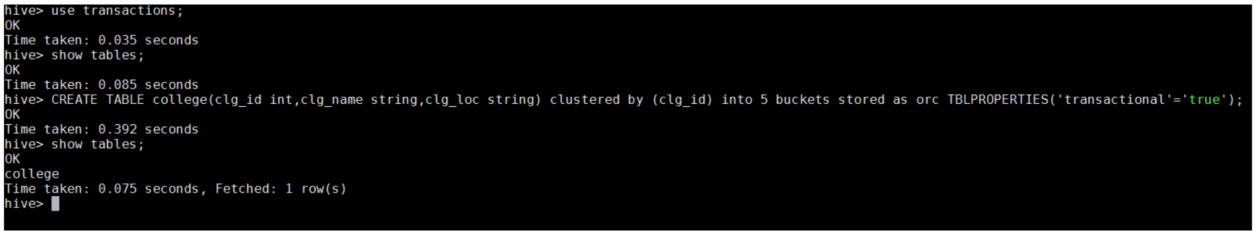
```
hive> set hive.support.concurrency = true;
hive>
    > set hive.enforce.bucketing = true;
hive>
    > set hive.exec.dynamic.partition.mode = nonstrict;
hive>
    > set hive.txn.manager = org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;
hive>
    > set hive.compactor.initiator.on = true;
hive>
    > set hive.compactor.worker.threads = 10;
hive>
```

Create a Buckted table which will support transaction

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');



```
hive> create database transactions;
OK
Time taken: 0.184 seconds
```

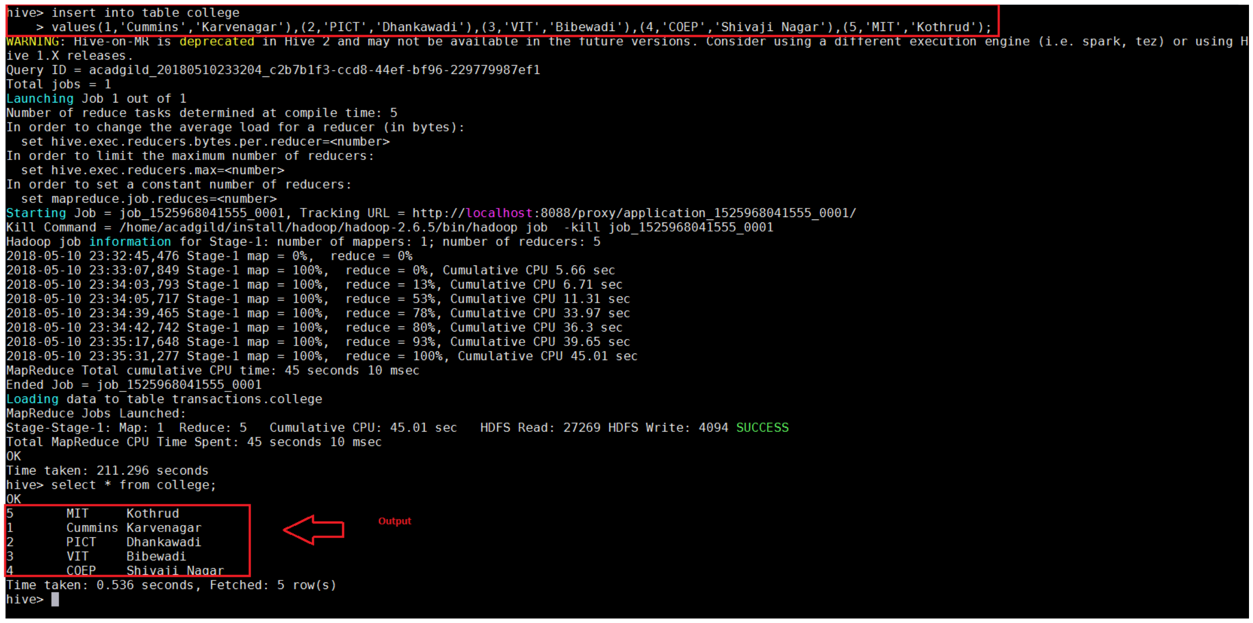


```
hive> use transactions;
OK
Time taken: 0.035 seconds
hive> show tables;
OK
Time taken: 0.085 seconds
hive> CREATE TABLE college(clg_id int,clg_name string,clg_loc string) clustered by (clg_id) into 5 buckets stored as orc TBLPROPERTIES('transactional'='true');
OK
Time taken: 0.392 seconds
hive> show tables;
OK
college
Time taken: 0.075 seconds, Fetched: 1 row(s)
hive> █
```

Inserting values in college table

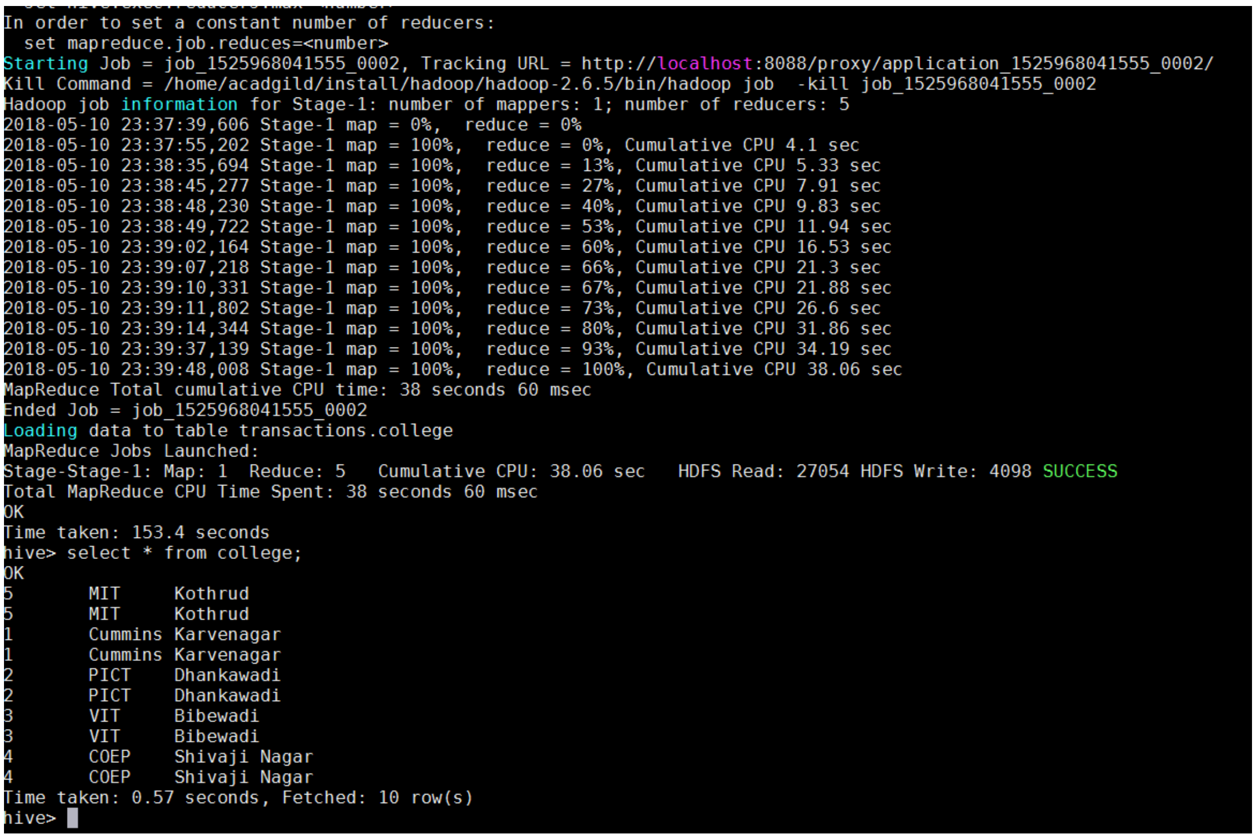
insert into table college values(1,'Cummins','Karvenagar'),(2,'PICT','Dhankawadi'),(3,'VIT','Bibewadi'),(4,'COEP','Shivaji Nagar'),(5,'MIT','Kothrud');

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



```
hive> insert into table college
    > values(1,'Cummins','Karvenagar'),(2,'PICT','Dhankawadi'),(3,'VIT','Bibewadi'),(4,'COEP','Shivaji Nagar'),(5,'MIT','Kothrud');
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 H
ive 1.X releases.
Query ID = acadgild_20180510233204_c2b7b1f3-ccd8-44ef-bf96-229779987ef1
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_1525968041555_0001, Tracking URL = http://localhost:8088/proxy/application_1525968041555_0001/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1525968041555_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 5
2018-05-10 23:32:45,476 Stage-1 map = 0%,  reduce = 0%
2018-05-10 23:33:07,849 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 5.66 sec
2018-05-10 23:34:03,793 Stage-1 map = 100%,  reduce = 13%, Cumulative CPU 6.71 sec
2018-05-10 23:34:05,717 Stage-1 map = 100%,  reduce = 53%, Cumulative CPU 11.31 sec
2018-05-10 23:34:39,465 Stage-1 map = 100%,  reduce = 78%, Cumulative CPU 33.97 sec
2018-05-10 23:34:42,742 Stage-1 map = 100%,  reduce = 80%, Cumulative CPU 36.3 sec
2018-05-10 23:35:17,648 Stage-1 map = 100%,  reduce = 93%, Cumulative CPU 39.65 sec
2018-05-10 23:35:31,277 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 45.01 sec
MapReduce Total cumulative CPU time: 45 seconds 10 msec
Ended Job = job_1525968041555_0001
Loading data to table transactions.college
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 5   Cumulative CPU: 45.01 sec   HDFS Read: 27269 HDFS Write: 4094 SUCCESS
Total MapReduce CPU Time Spent: 45 seconds 10 msec
OK
Time taken: 211.296 seconds
hive> select * from college;
OK
5       MIT     Kothrud
1       Cummins Karvenagar
2       PICT    Dhankawadi
3       VIT     Bibewadi
4       COEP    Shivaji Nagar
Time taken: 0.536 seconds, Fetched: 5 row(s)
hive> █
```

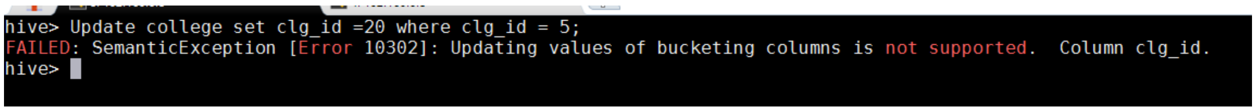
we will re-insert the same data again, it will be appended to the previous data as shown below:



```
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1525968041555_0002, Tracking URL = http://localhost:8088/proxy/application_1525968041555_0002/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1525968041555_0002
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 5
2018-05-10 23:37:39,606 Stage-1 map = 0%,  reduce = 0%
2018-05-10 23:37:55,202 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.1 sec
2018-05-10 23:38:35,694 Stage-1 map = 100%,  reduce = 13%, Cumulative CPU 5.33 sec
2018-05-10 23:38:45,277 Stage-1 map = 100%,  reduce = 27%, Cumulative CPU 7.91 sec
2018-05-10 23:38:48,230 Stage-1 map = 100%,  reduce = 40%, Cumulative CPU 9.83 sec
2018-05-10 23:38:49,722 Stage-1 map = 100%,  reduce = 53%, Cumulative CPU 11.94 sec
2018-05-10 23:39:02,164 Stage-1 map = 100%,  reduce = 60%, Cumulative CPU 16.53 sec
2018-05-10 23:39:07,218 Stage-1 map = 100%,  reduce = 66%, Cumulative CPU 21.3 sec
2018-05-10 23:39:10,331 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 21.88 sec
2018-05-10 23:39:11,802 Stage-1 map = 100%,  reduce = 73%, Cumulative CPU 26.6 sec
2018-05-10 23:39:14,344 Stage-1 map = 100%,  reduce = 80%, Cumulative CPU 31.86 sec
2018-05-10 23:39:37,139 Stage-1 map = 100%,  reduce = 93%, Cumulative CPU 34.19 sec
2018-05-10 23:39:48,008 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 38.06 sec
MapReduce Total cumulative CPU time: 38 seconds 60 msec
Ended Job = job_1525968041555_0002
Loading data to table transactions.college
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 5   Cumulative CPU: 38.06 sec   HDFS Read: 27054 HDFS Write: 4098 SUCCESS
Total MapReduce CPU Time Spent: 38 seconds 60 msec
OK
Time taken: 153.4 seconds
hive> select * from college;
OK
5       MIT     Kothrud
5       MIT     Kothrud
1       Cummins Karvenagar
1       Cummins Karvenagar
2       PICT    Dhankawadi
2       PICT    Dhankawadi
3       VIT     Bibewadi
3       VIT     Bibewadi
4       COEP    Shivaji Nagar
4       COEP    Shivaji Nagar
Time taken: 0.57 seconds, Fetched: 10 row(s)
hive>
```

Updating the Data in Hive Table on bucketed column

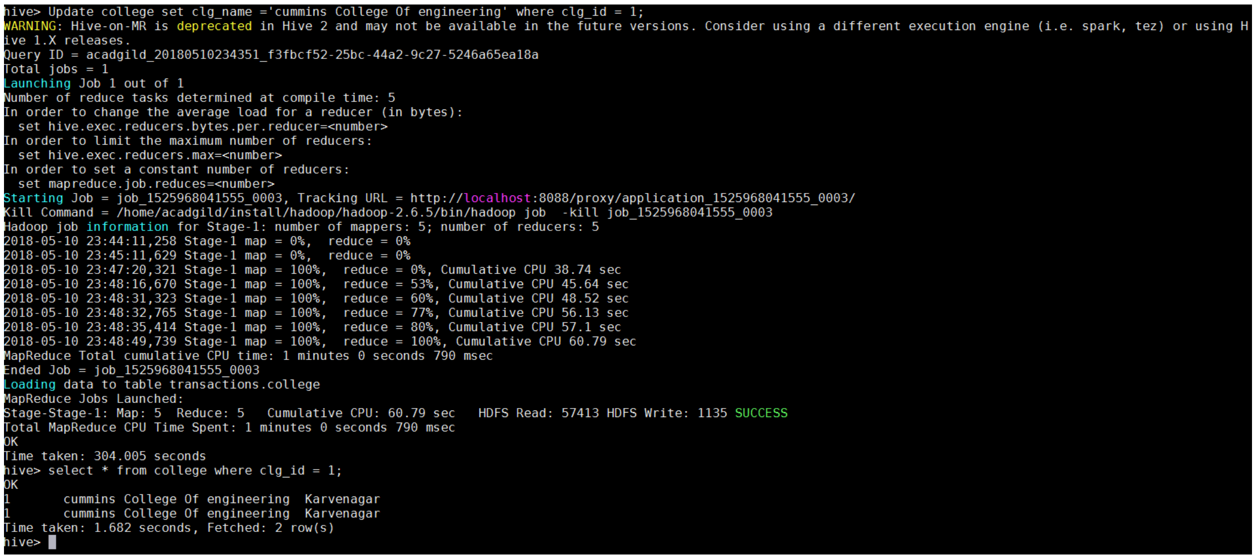
Update college set clg_id =20 where clg_id = 5;



```
hive> Update college set clg_id =20 where clg_id = 5;
FAILED: SemanticException [Error 10302]: Updating values of bucketing columns is not supported.  Column clg_id.
hive>
```

But we can perform update operation on Non bucketed column

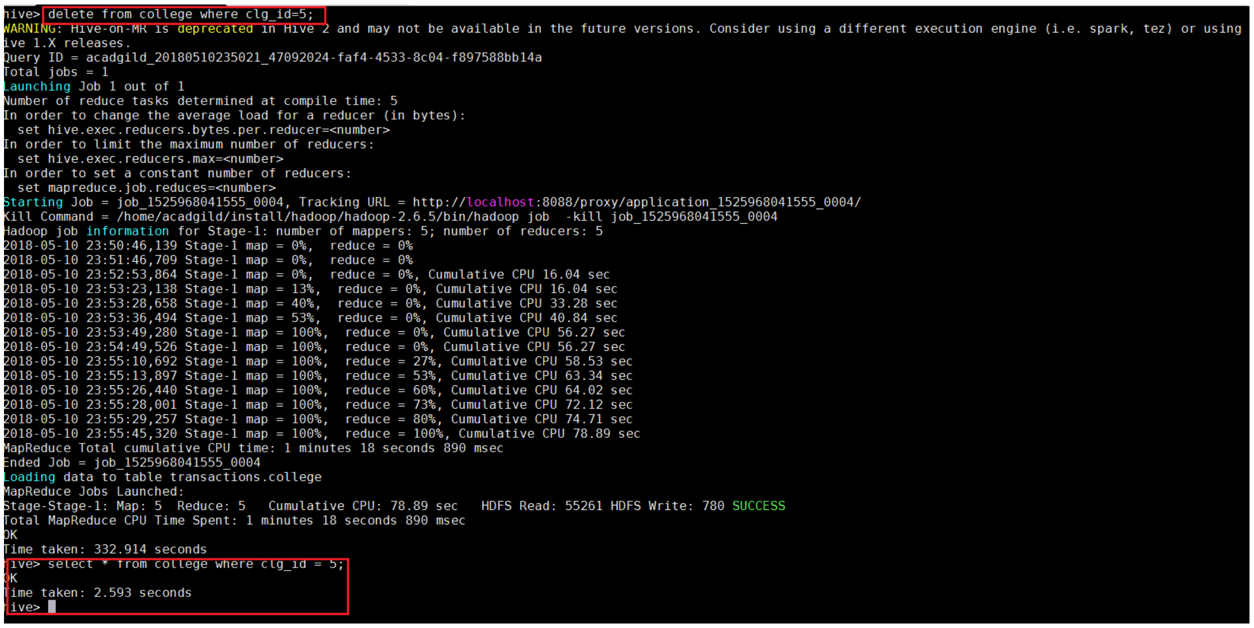
Update college set clg_name ='cummins College Of engineering' where clg_id = 1;



```
hive> Update college set clg_name ='cummins College Of engineering' 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 H
ive 1.X releases.
Query ID = acadgild_20180510234351_f3fbcf52-25bc-44a2-9c27-5246a65ea18a
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_1525968041555_0003, Tracking URL = http://localhost:8088/proxy/application_1525968041555_0003/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1525968041555_0003
Hadoop job information for Stage-1: number of mappers: 5; number of reducers: 5
2018-05-10 23:44:11,258 Stage-1 map = 0%,  reduce = 0%
2018-05-10 23:45:11,629 Stage-1 map = 0%,  reduce = 0%
2018-05-10 23:47:20,321 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 38.74 sec
2018-05-10 23:48:16,670 Stage-1 map = 100%,  reduce = 53%, Cumulative CPU 45.64 sec
2018-05-10 23:48:31,323 Stage-1 map = 100%,  reduce = 60%, Cumulative CPU 48.52 sec
2018-05-10 23:48:32,765 Stage-1 map = 100%,  reduce = 77%, Cumulative CPU 56.13 sec
2018-05-10 23:48:35,414 Stage-1 map = 100%,  reduce = 80%, Cumulative CPU 57.1 sec
2018-05-10 23:48:49,739 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 60.79 sec
MapReduce Total cumulative CPU time: 1 minutes 0 seconds 790 msec
Ended Job = job_1525968041555_0003
Loading data to table transactions.college
MapReduce Jobs Launched:
Stage-Stage-1: Map: 5  Reduce: 5   Cumulative CPU: 60.79 sec   HDFS Read: 57413 HDFS Write: 1135 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 0 seconds 790 msec
OK
Time taken: 304.005 seconds
hive> select * from college where clg_id = 1;
OK
1       cummins College Of engineering  Karvenagar
1       cummins College Of engineering  Karvenagar
Time taken: 1.682 seconds, Fetched: 2 row(s)
hive> 
```

Deleting a Row from Hive Table:

delete from college where clg_id=5;



```
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
ive 1.X releases.
Query ID = acadgild_20180510235021_47092024-faf4-4533-8c04-f897588bb14a
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_1525968041555_0004, Tracking URL = http://localhost:8088/proxy/application_1525968041555_0004/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1525968041555_0004
Hadoop job information for Stage-1: number of mappers: 5; number of reducers: 5
2018-05-10 23:50:46,139 Stage-1 map = 0%,  reduce = 0%
2018-05-10 23:51:46,709 Stage-1 map = 0%,  reduce = 0%
2018-05-10 23:52:53,864 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 16.04 sec
2018-05-10 23:53:23,138 Stage-1 map = 13%,  reduce = 0%, Cumulative CPU 16.04 sec
2018-05-10 23:53:28,658 Stage-1 map = 40%,  reduce = 0%, Cumulative CPU 33.28 sec
2018-05-10 23:53:36,494 Stage-1 map = 53%,  reduce = 0%, Cumulative CPU 40.84 sec
2018-05-10 23:53:49,280 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 56.27 sec
2018-05-10 23:54:49,526 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 56.27 sec
2018-05-10 23:55:10,692 Stage-1 map = 100%,  reduce = 27%, Cumulative CPU 58.53 sec
2018-05-10 23:55:13,897 Stage-1 map = 100%,  reduce = 53%, Cumulative CPU 63.34 sec
2018-05-10 23:55:26,440 Stage-1 map = 100%,  reduce = 60%, Cumulative CPU 64.02 sec
2018-05-10 23:55:28,001 Stage-1 map = 100%,  reduce = 73%, Cumulative CPU 72.12 sec
2018-05-10 23:55:29,257 Stage-1 map = 100%,  reduce = 80%, Cumulative CPU 74.71 sec
2018-05-10 23:55:45,320 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 78.89 sec
MapReduce Total cumulative CPU time: 1 minutes 18 seconds 890 msec
Ended Job = job_1525968041555_0004
Loading data to table transactions.college
MapReduce Jobs Launched:
Stage-Stage-1: Map: 5  Reduce: 5   Cumulative CPU: 78.89 sec   HDFS Read: 55261 HDFS Write: 780 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 18 seconds 890 msec
OK
Time taken: 332.914 seconds
hive> select * from college where clg_id = 5;
OK
Time taken: 2.593 seconds
hive> 
```