Session-5

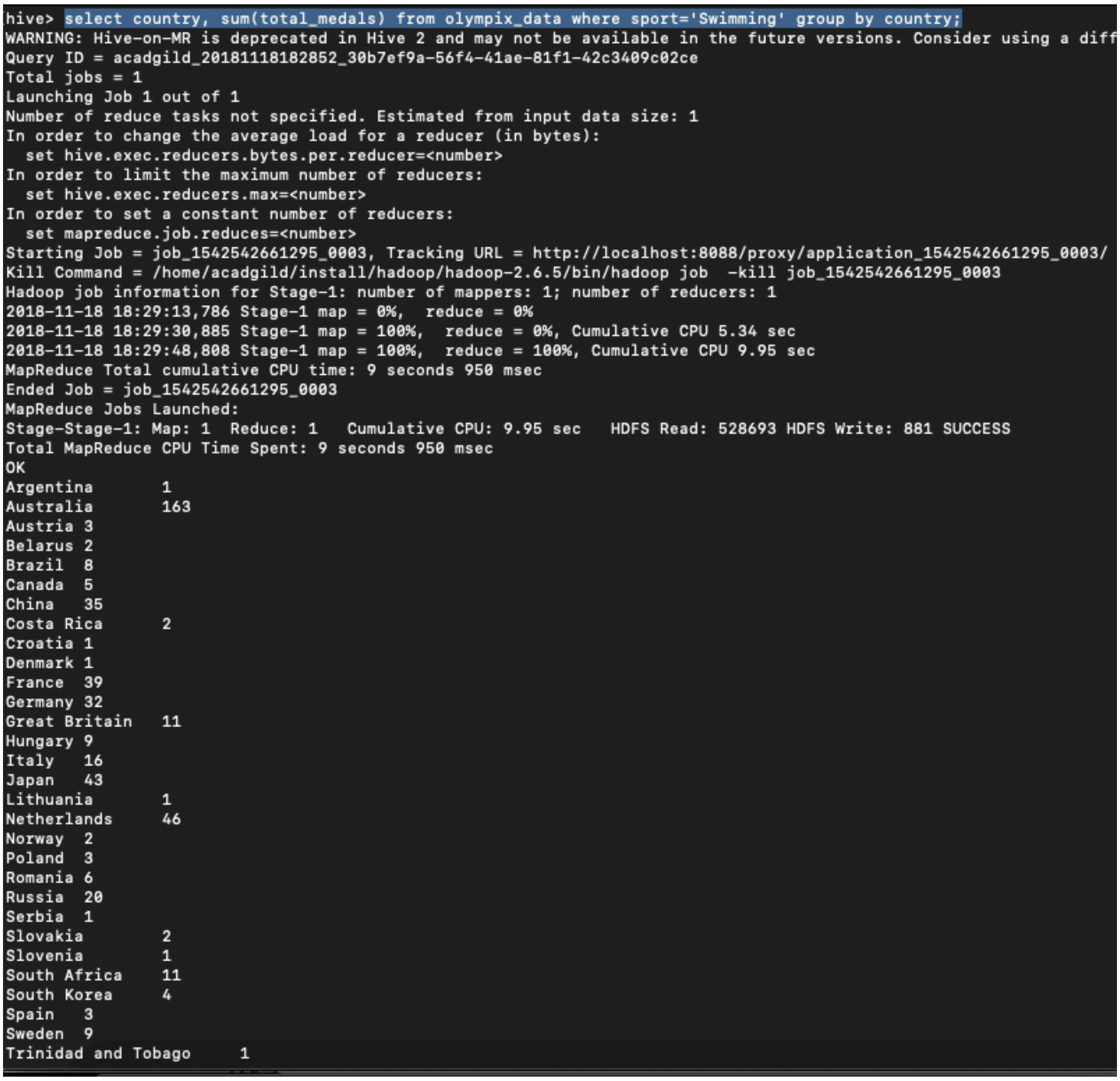
Advance Hive & HBASE BASICS

Assignment

Task 1.1

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

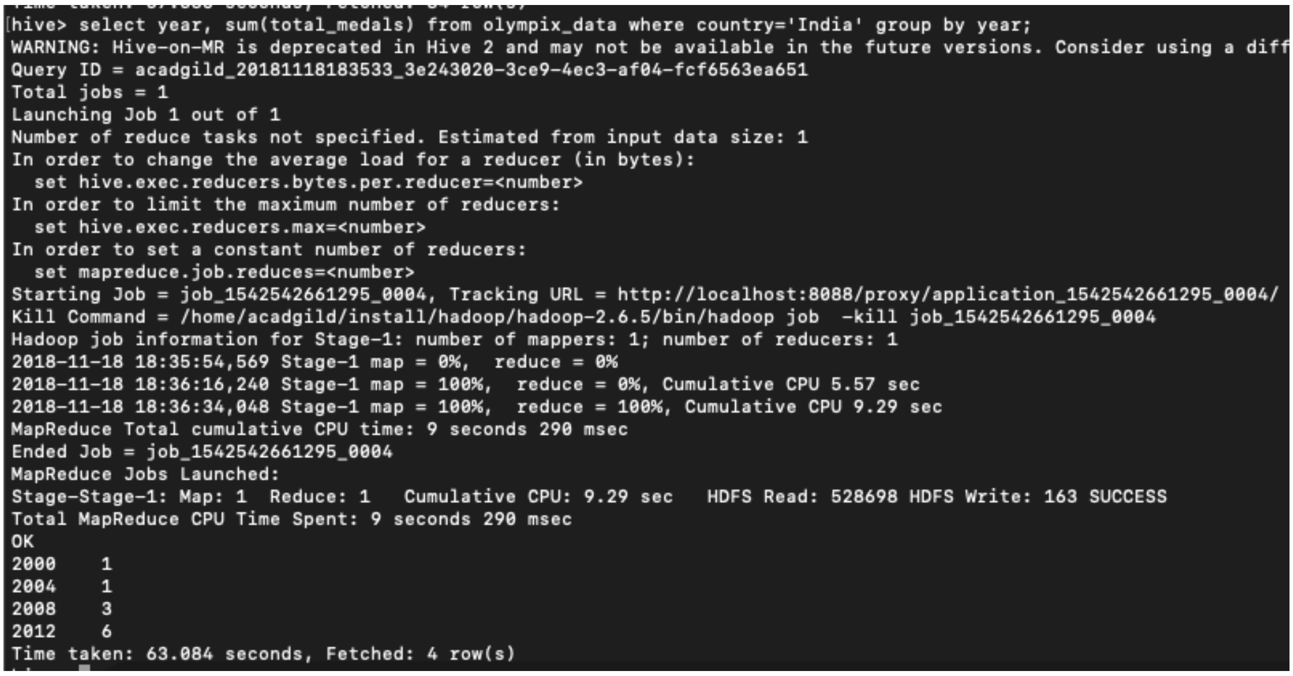
```
select country, sum(total_medals) as medals from olympix_data where
sport='Swimming' group by country;
```



```
hive> select country, sum(total_medals) from olympix_data where sport='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 diff
Query ID = acadgild_20181118182852_30b7ef9a-56f4-41ae-81f1-42c3409c02ce
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:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1542542661295_0003, Tracking URL = http://localhost:8088/proxy/application_1542542661295_0003/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1542542661295_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-11-18 18:29:13,786 Stage-1 map = 0%,  reduce = 0%
2018-11-18 18:29:30,885 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 5.34 sec
2018-11-18 18:29:48,808 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 9.95 sec
MapReduce Total cumulative CPU time: 9 seconds 950 msec
Ended Job = job_1542542661295_0003
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 9.95 sec   HDFS Read: 528693 HDFS Write: 881 SUCCESS
Total MapReduce CPU Time Spent: 9 seconds 950 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
```

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

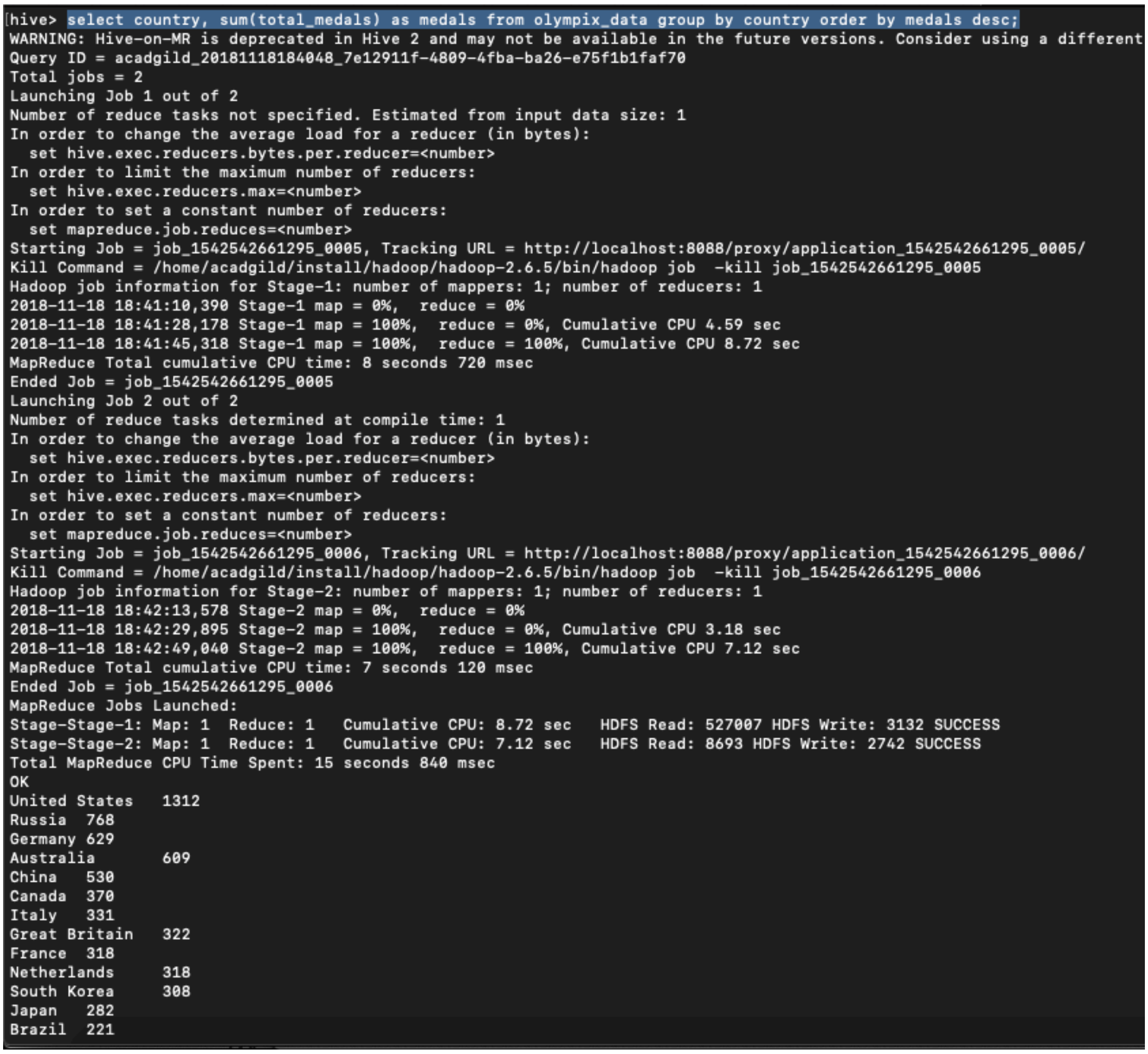
```
select year, sum(total_medals) as medals from olympix_data where
country='India' group by year;
```



```
hive> select year, sum(total_medals) from olympix_data where country='India' group by year;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a diff
Query ID = acadgild_20181118183533_3e243020-3ce9-4ec3-af04-fcf6563ea651
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:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1542542661295_0004, Tracking URL = http://localhost:8088/proxy/application_1542542661295_0004/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1542542661295_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-11-18 18:35:54,569 Stage-1 map = 0%,  reduce = 0%
2018-11-18 18:36:16,240 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 5.57 sec
2018-11-18 18:36:34,048 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 9.29 sec
MapReduce Total cumulative CPU time: 9 seconds 290 msec
Ended Job = job_1542542661295_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 9.29 sec   HDFS Read: 528698 HDFS Write: 163 SUCCESS
Total MapReduce CPU Time Spent: 9 seconds 290 msec
OK
2000    1
2004    1
2008    3
2012    6
Time taken: 63.084 seconds, Fetched: 4 row(s)
```

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

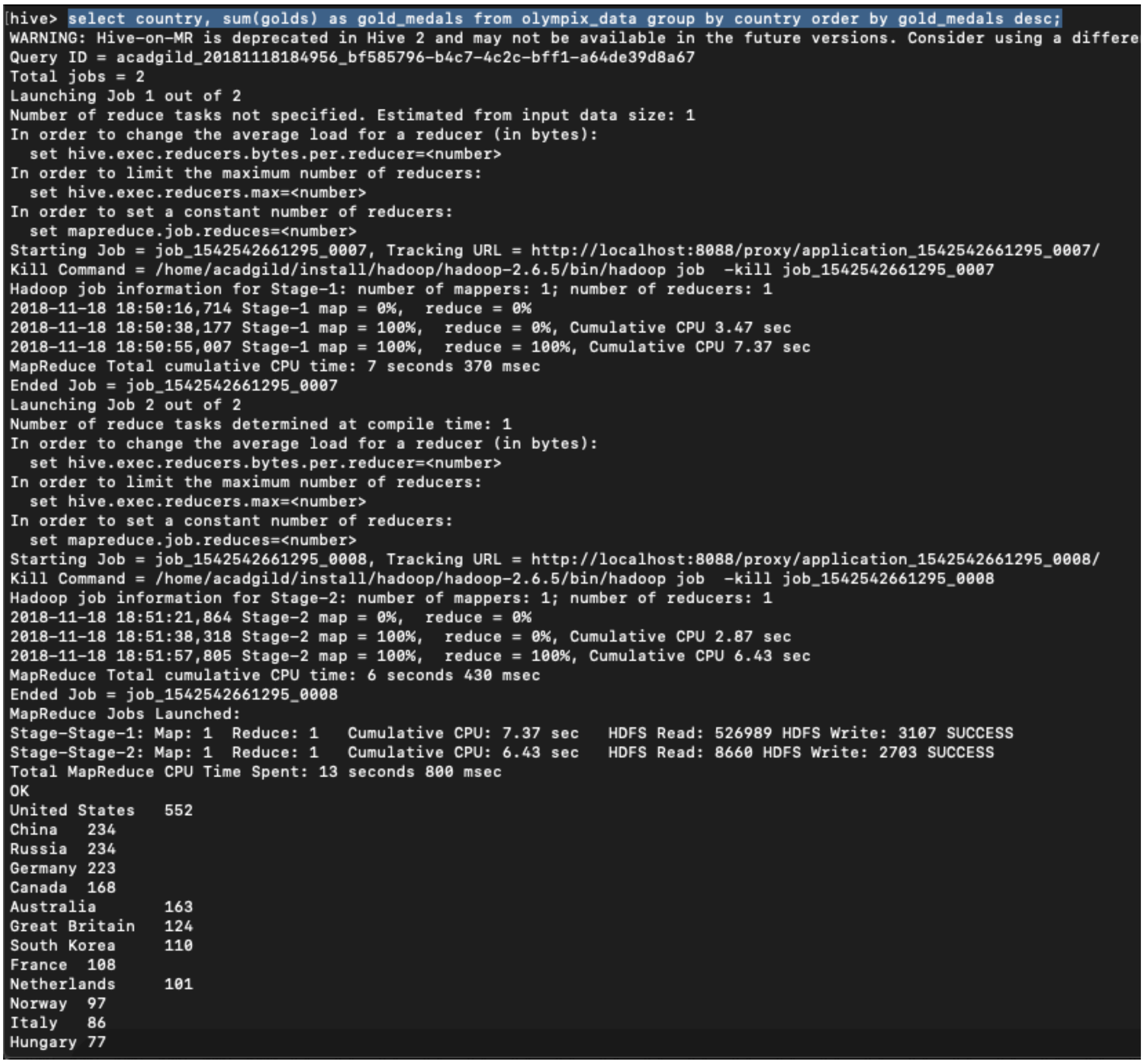
```
select country, sum(total_medals) as medals from olympix_data group by
country order by medals desc;
```



```
hive> select country, sum(total_medals) as medals from olympix_data group by country order by medals desc;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different
Query ID = acadgild_20181118184048_7e12911f-4809-4fba-ba26-e75f1b1faf70
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>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1542542661295_0005, Tracking URL = http://localhost:8088/proxy/application_1542542661295_0005/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1542542661295_0005
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-11-18 18:41:10,390 Stage-1 map = 0%,  reduce = 0%
2018-11-18 18:41:28,178 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.59 sec
2018-11-18 18:41:45,318 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 8.72 sec
MapReduce Total cumulative CPU time: 8 seconds 720 msec
Ended Job = job_1542542661295_0005
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 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>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1542542661295_0006, Tracking URL = http://localhost:8088/proxy/application_1542542661295_0006/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1542542661295_0006
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-11-18 18:42:13,578 Stage-2 map = 0%,  reduce = 0%
2018-11-18 18:42:29,895 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 3.18 sec
2018-11-18 18:42:49,040 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 7.12 sec
MapReduce Total cumulative CPU time: 7 seconds 120 msec
Ended Job = job_1542542661295_0006
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 8.72 sec   HDFS Read: 527007 HDFS Write: 3132 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 7.12 sec   HDFS Read: 8693 HDFS Write: 2742 SUCCESS
Total MapReduce CPU Time Spent: 15 seconds 840 msec
OK
United States	1312
Russia	768
Germany	629
Australia	609
China	530
Canada	370
Italy	331
Great Britain	322
France	318
Netherlands	318
South Korea	308
Japan	282
Brazil	221
```

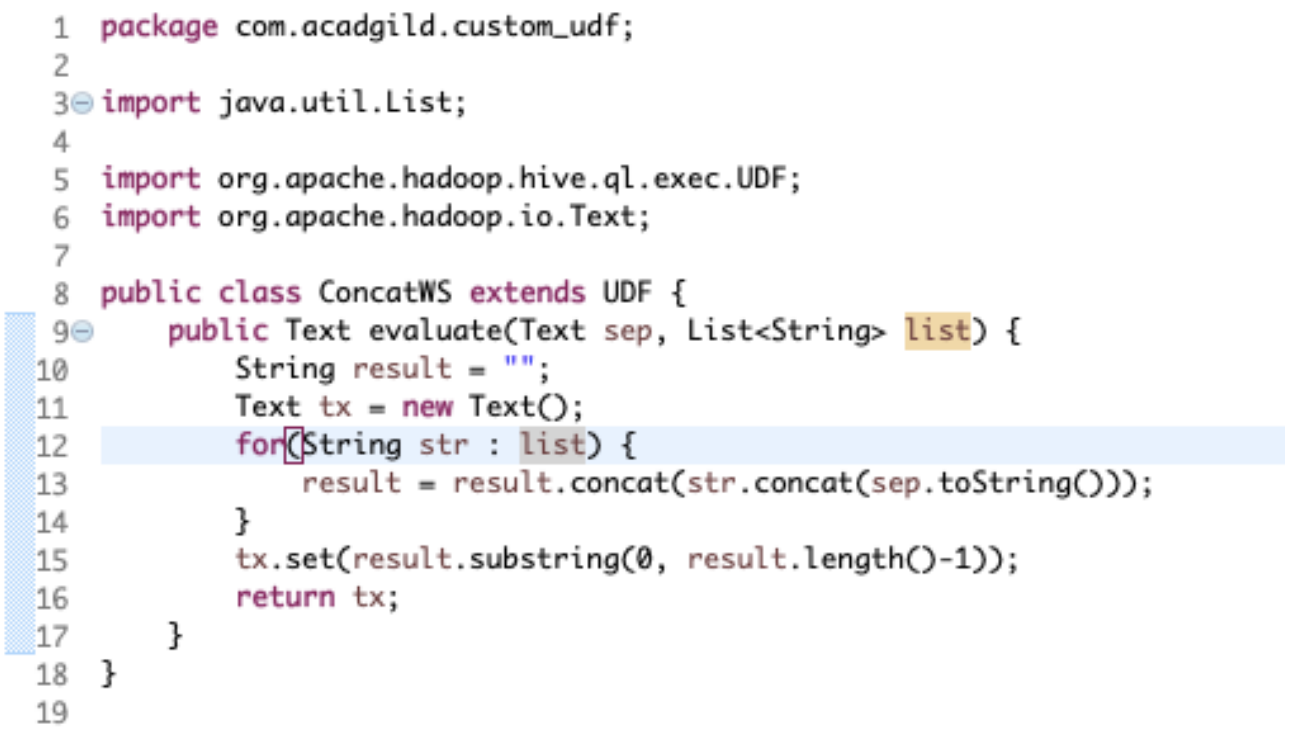
4. Write a Hive program to find the number of gold medals each country won.

```
select country, sum(golds) as gold_medals from olympix_data group by
country order by gold_medals desc;
```



```
hive> select country, sum(golds) as gold_medals from olympix_data group by country order by gold_medals desc;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a differe
Query ID = acadgild_20181118184956_bf585796-b4c7-4c2c-bff1-a64de39d8a67
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>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1542542661295_0007, Tracking URL = http://localhost:8088/proxy/application_1542542661295_0007/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1542542661295_0007
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-11-18 18:50:16,714 Stage-1 map = 0%,  reduce = 0%
2018-11-18 18:50:38,177 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.47 sec
2018-11-18 18:50:55,007 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 7.37 sec
MapReduce Total cumulative CPU time: 7 seconds 370 msec
Ended Job = job_1542542661295_0007
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 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>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1542542661295_0008, Tracking URL = http://localhost:8088/proxy/application_1542542661295_0008/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1542542661295_0008
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-11-18 18:51:21,864 Stage-2 map = 0%,  reduce = 0%
2018-11-18 18:51:38,318 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 2.87 sec
2018-11-18 18:51:57,805 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 6.43 sec
MapReduce Total cumulative CPU time: 6 seconds 430 msec
Ended Job = job_1542542661295_0008
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 7.37 sec   HDFS Read: 526989 HDFS Write: 3107 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 6.43 sec   HDFS Read: 8660 HDFS Write: 2703 SUCCESS
Total MapReduce CPU Time Spent: 13 seconds 800 msec
OK
United States	552
China	234
Russia	234
Germany	223
Canada	168
Australia	163
Great Britain	124
South Korea	110
France	108
Netherlands	101
Norway	97
Italy	86
Hungary	77
```

Task 1.2
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.



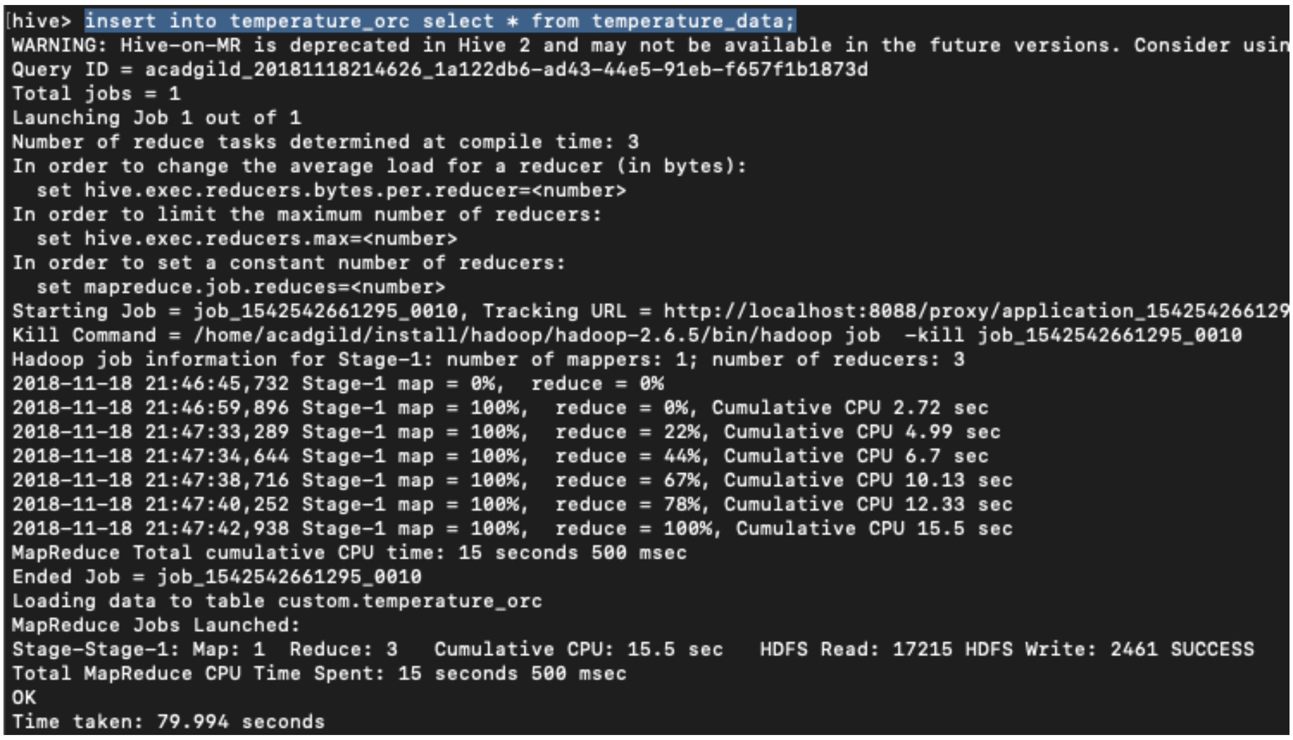
```java
package com.acadgild.custom_udf;

import java.util.List;

import org.apache.hadoop.hive.ql.exec.UDF;
import org.apache.hadoop.io.Text;

public class ConcatWS extends UDF {
    public Text evaluate(Text sep, List<String> list) {
        String result = "";
        Text tx = new Text();
        for(String str : list) {
            result = result.concat(str.concat(sep.toString()));
        }
        tx.set(result.substring(0, result.length()-1));
        return tx;
    }
}
```

Task 1.3

ACID Transactions – Transactions in Hive

```
hive> update temperature_data set temperature=25 where zip_code = 
560037;
FAILED: SemanticException [Error 10294]: Attempt to do update or delete 
using transaction manager that does not support these operations.
```

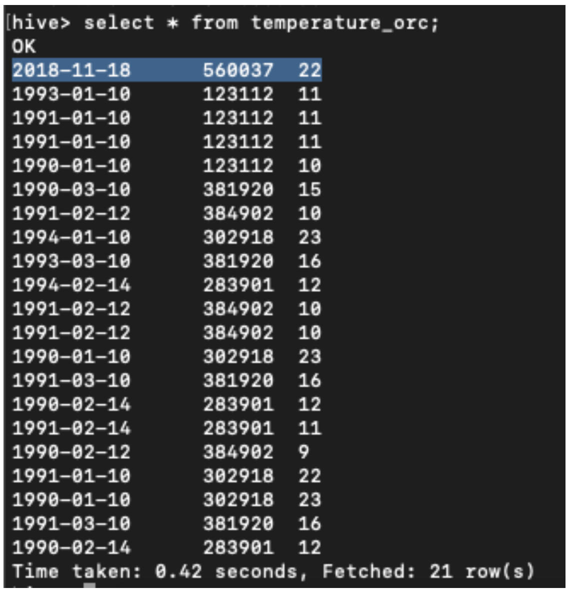
```
create table temperature_orc(temp_data date, zip_code int, temperature 
int) clustered by (zip_code) into 3 buckets stored as orc 
TBLPROPERTIES('transactional'='true');
```



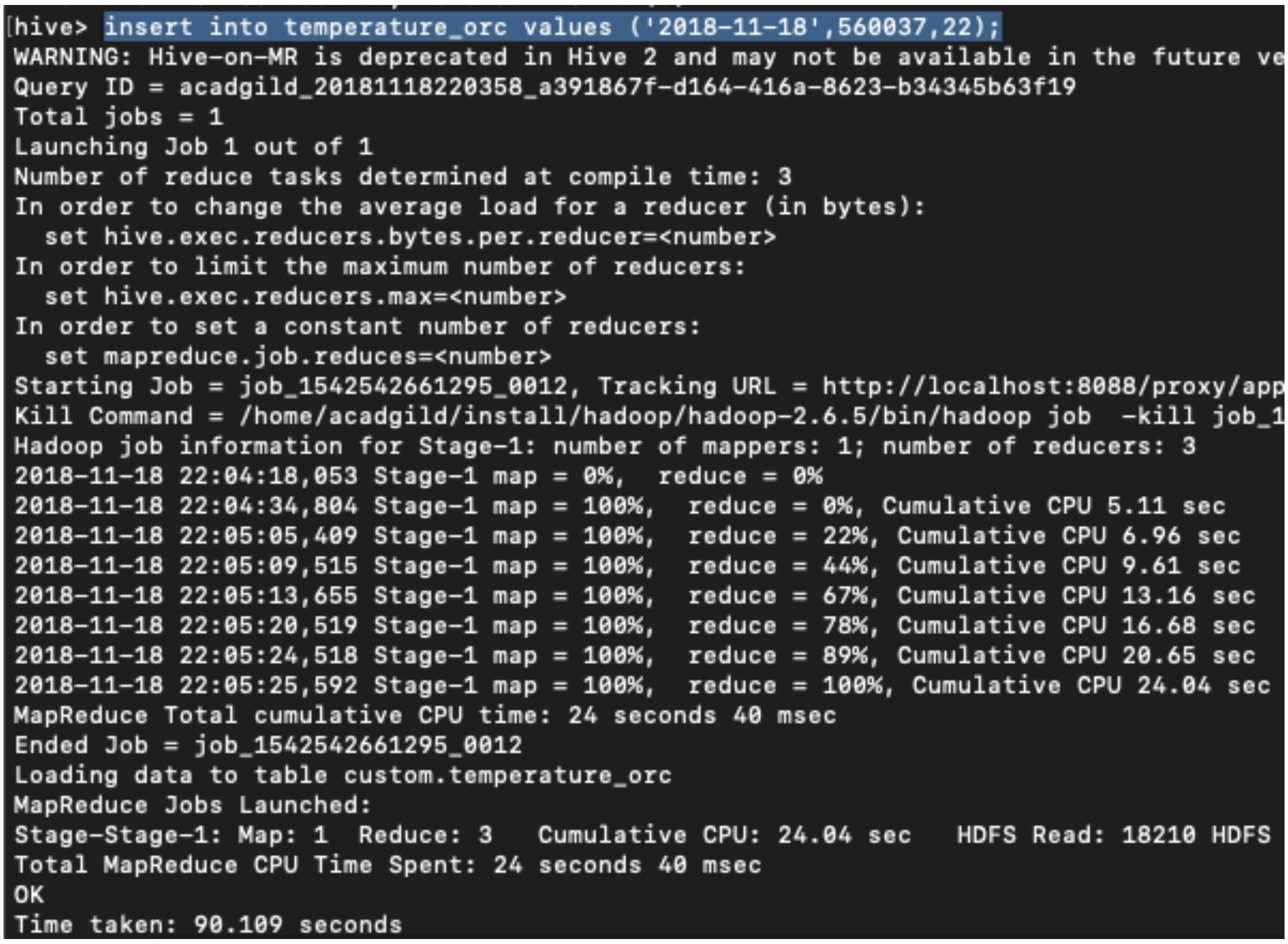
```
hive> insert into temperature_orc select * from temperature_data;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider usin
Query ID = acadgild_20181118214626_1a122db6-ad43-44e5-91eb-f657f1b1873d
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 3
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_1542542661295_0010, Tracking URL = http://localhost:8088/proxy/application_154254266129
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1542542661295_0010
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 3
2018-11-18 21:46:45,732 Stage-1 map = 0%,  reduce = 0%
2018-11-18 21:46:59,896 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.72 sec
2018-11-18 21:47:33,289 Stage-1 map = 100%,  reduce = 22%, Cumulative CPU 4.99 sec
2018-11-18 21:47:34,644 Stage-1 map = 100%,  reduce = 44%, Cumulative CPU 6.7 sec
2018-11-18 21:47:38,716 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 10.13 sec
2018-11-18 21:47:40,252 Stage-1 map = 100%,  reduce = 78%, Cumulative CPU 12.33 sec
2018-11-18 21:47:42,938 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 15.5 sec
MapReduce Total cumulative CPU time: 15 seconds 500 msec
Ended Job = job_1542542661295_0010
Loading data to table custom.temperature_orc
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 3   Cumulative CPU: 15.5 sec   HDFS Read: 17215 HDFS Write: 2461 SUCCESS
Total MapReduce CPU Time Spent: 15 seconds 500 msec
OK
Time taken: 79.994 seconds
```

<u>INSERT</u>

```
insert into temperature_orc values ('2018-11-18',560037,22);
```

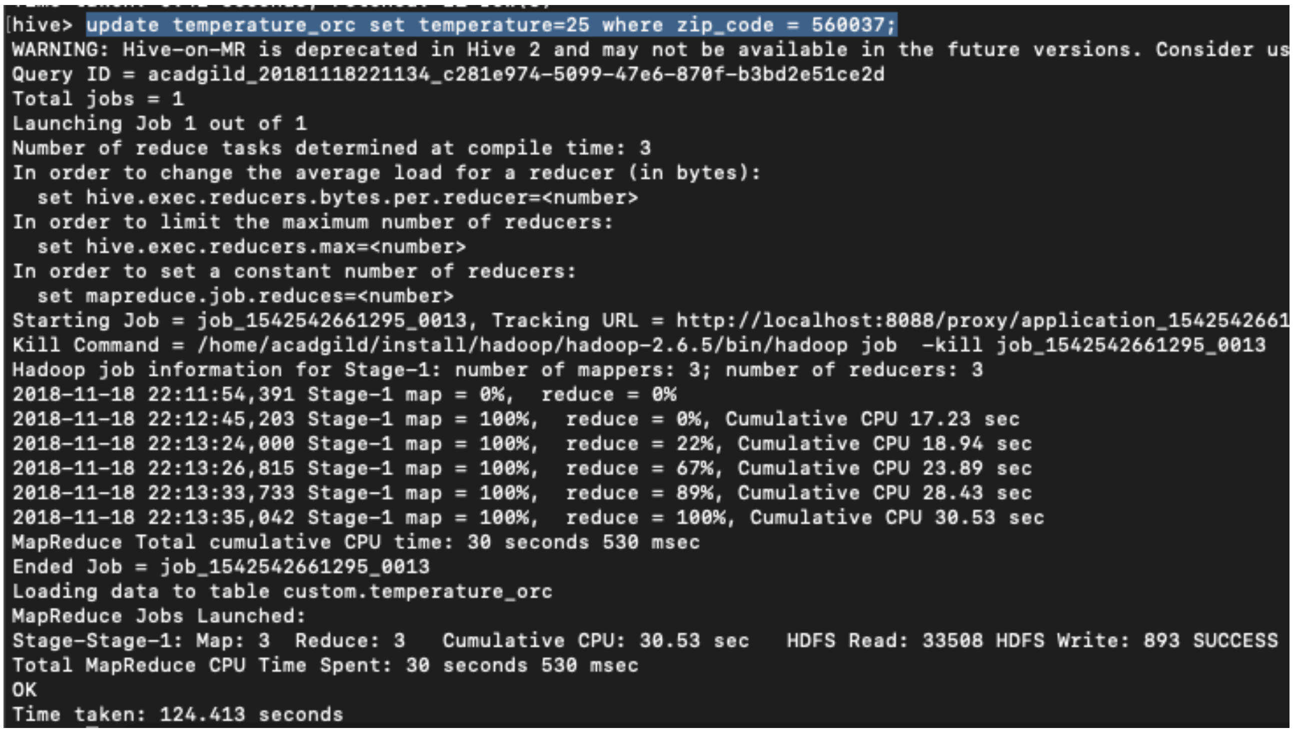


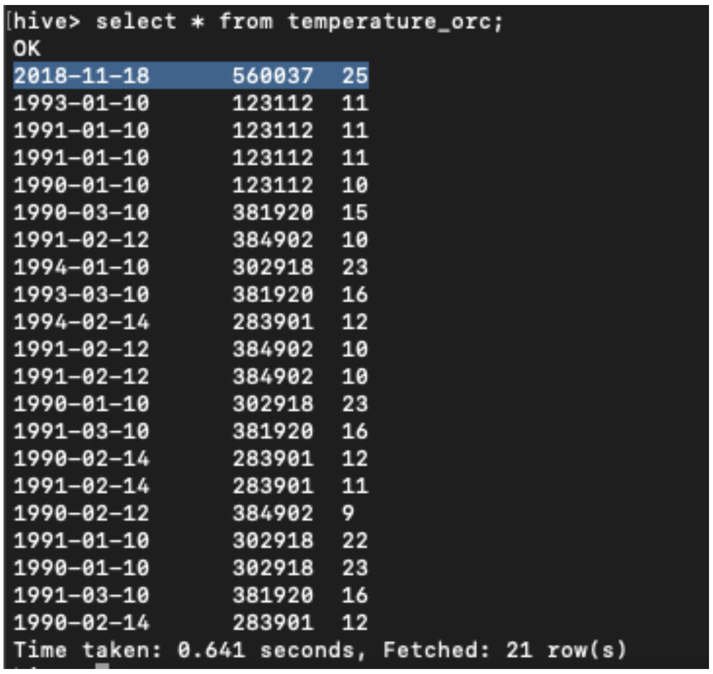
```
hive> select * from temperature_orc;
OK
2018-11-18      560037  22
1993-01-10      123112  11
1991-01-10      123112  11
1991-01-10      123112  11
1990-01-10      123112  10
1990-03-10      381920  15
1991-02-12      384902  10
1994-01-10      302918  23
1993-03-10      381920  16
1994-02-14      283901  12
1991-02-12      384902  10
1991-02-12      384902  10
1990-01-10      302918  23
1991-03-10      381920  16
1990-02-14      283901  12
1991-02-14      283901  11
1990-02-12      384902  9
1991-01-10      302918  22
1990-01-10      302918  23
1991-03-10      381920  16
1990-02-14      283901  12
Time taken: 0.42 seconds, Fetched: 21 row(s)
```


```
[hive> insert into temperature_orc values ('2018-11-18',560037,22);
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future ve
Query ID = acadgild_20181118220358_a391867f-d164-416a-8623-b34345b63f19
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 3
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_1542542661295_0012, Tracking URL = http://localhost:8088/proxy/app
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 3
2018-11-18 22:04:18,053 Stage-1 map = 0%,  reduce = 0%
2018-11-18 22:04:34,804 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 5.11 sec
2018-11-18 22:05:05,409 Stage-1 map = 100%,  reduce = 22%, Cumulative CPU 6.96 sec
2018-11-18 22:05:09,515 Stage-1 map = 100%,  reduce = 44%, Cumulative CPU 9.61 sec
2018-11-18 22:05:13,655 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 13.16 sec
2018-11-18 22:05:20,519 Stage-1 map = 100%,  reduce = 78%, Cumulative CPU 16.68 sec
2018-11-18 22:05:24,518 Stage-1 map = 100%,  reduce = 89%, Cumulative CPU 20.65 sec
2018-11-18 22:05:25,592 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 24.04 sec
MapReduce Total cumulative CPU time: 24 seconds 40 msec
Ended Job = job_1542542661295_0012
Loading data to table custom.temperature_orc
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 3   Cumulative CPU: 24.04 sec   HDFS Read: 18210 HDFS
Total MapReduce CPU Time Spent: 24 seconds 40 msec
OK
Time taken: 90.109 seconds
```

UPDATE

update temperature_orc set temperature=25 where zip_code = 560037;


```
[hive> update temperature_orc set temperature=25 where zip_code = 560037;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider us
Query ID = acadgild_20181118221134_c281e974-5099-47e6-870f-b3bd2e51ce2d
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 3
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_1542542661295_0013, Tracking URL = http://localhost:8088/proxy/application_1542542661
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1542542661295_0013
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 3
2018-11-18 22:11:54,391 Stage-1 map = 0%,  reduce = 0%
2018-11-18 22:12:45,203 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 17.23 sec
2018-11-18 22:13:24,000 Stage-1 map = 100%,  reduce = 22%, Cumulative CPU 18.94 sec
2018-11-18 22:13:26,815 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 23.89 sec
2018-11-18 22:13:33,733 Stage-1 map = 100%,  reduce = 89%, Cumulative CPU 28.43 sec
2018-11-18 22:13:35,042 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 30.53 sec
MapReduce Total cumulative CPU time: 30 seconds 530 msec
Ended Job = job_1542542661295_0013
Loading data to table custom.temperature_orc
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 3   Cumulative CPU: 30.53 sec   HDFS Read: 33508 HDFS Write: 893 SUCCESS
Total MapReduce CPU Time Spent: 30 seconds 530 msec
OK
Time taken: 124.413 seconds
```

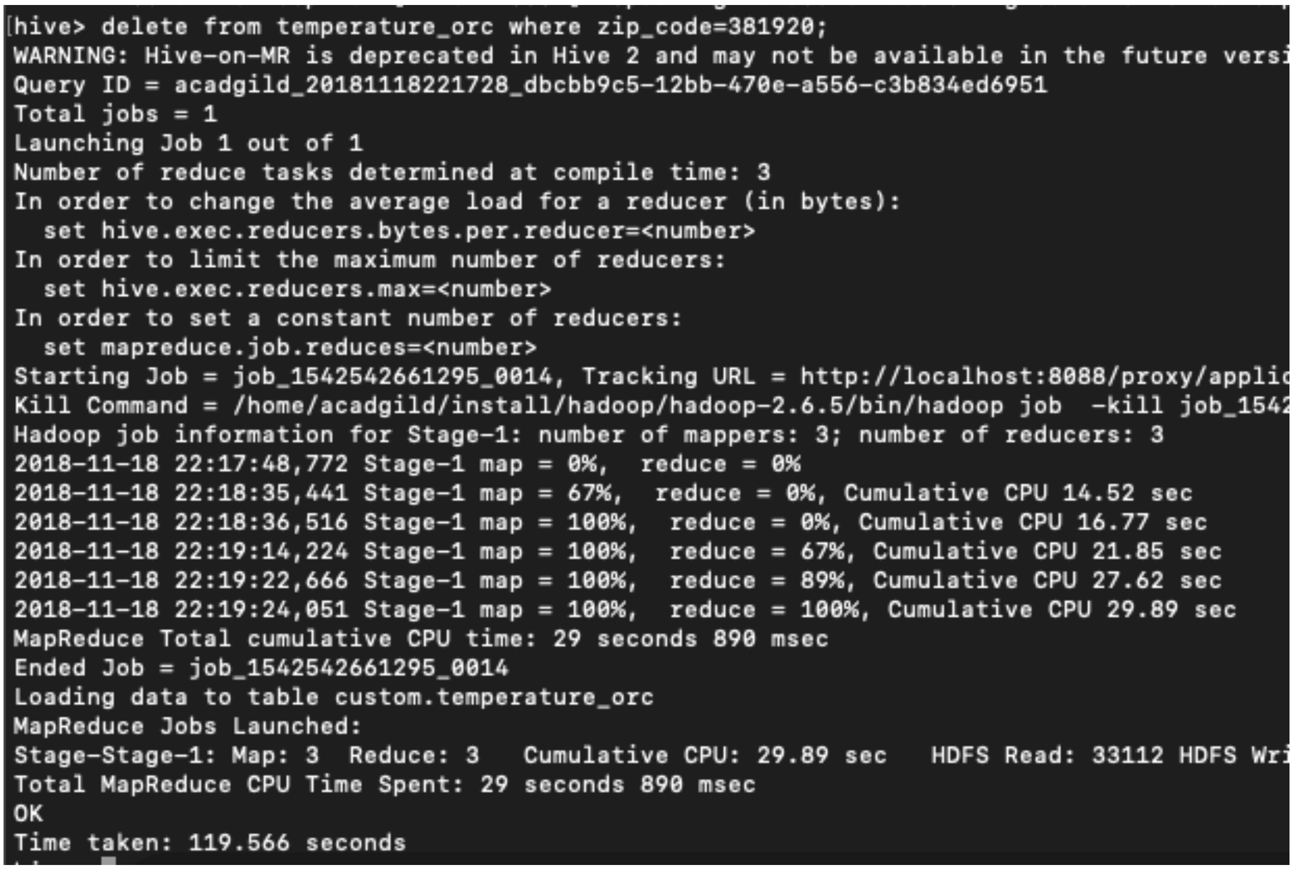


```
hive> select * from temperature_orc;
OK
2018-11-18      560037  25
1993-01-10      123112  11
1991-01-10      123112  11
1991-01-10      123112  11
1990-01-10      123112  10
1990-03-10      381920  15
1991-02-12      384902  10
1994-01-10      302918  23
1993-03-10      381920  16
1994-02-14      283901  12
1991-02-12      384902  10
1991-02-12      384902  10
1990-01-10      302918  23
1991-03-10      381920  16
1990-02-14      283901  12
1991-02-14      283901  11
1990-02-12      384902  9
1991-01-10      302918  22
1990-01-10      302918  23
1991-03-10      381920  16
1990-02-14      283901  12
Time taken: 0.641 seconds, Fetched: 21 row(s)
```

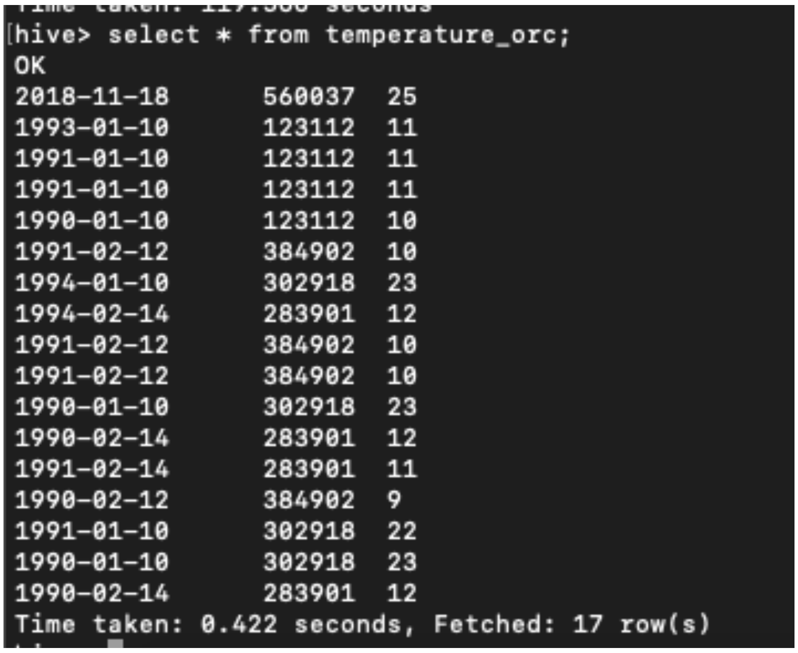
```
hive> update temperature_orc set zip_code=560035 where zip_code =
560037;
FAILED: SemanticException [Error 10302]: Updating values of bucketing
columns is not supported.  Column zip_code.
hive>
```

DELETE

```
delete from temperature_orc where zip_code=381920;
```



```
hive> delete from temperature_orc where zip_code=381920;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versi
Query ID = acadgild_20181118221728_dbcbb9c5-12bb-470e-a556-c3b834ed6951
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 3
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_1542542661295_0014, Tracking URL = http://localhost:8088/proxy/applic
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1542
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 3
2018-11-18 22:17:48,772 Stage-1 map = 0%,  reduce = 0%
2018-11-18 22:18:35,441 Stage-1 map = 67%,  reduce = 0%, Cumulative CPU 14.52 sec
2018-11-18 22:18:36,516 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 16.77 sec
2018-11-18 22:19:14,224 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 21.85 sec
2018-11-18 22:19:22,666 Stage-1 map = 100%,  reduce = 89%, Cumulative CPU 27.62 sec
2018-11-18 22:19:24,051 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 29.89 sec
MapReduce Total cumulative CPU time: 29 seconds 890 msec
Ended Job = job_1542542661295_0014
Loading data to table custom.temperature_orc
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 3   Cumulative CPU: 29.89 sec   HDFS Read: 33112 HDFS Wri
Total MapReduce CPU Time Spent: 29 seconds 890 msec
OK
Time taken: 119.566 seconds
```



```
hive> select * from temperature_orc;
OK
2018-11-18      560037  25
1993-01-10      123112  11
1991-01-10      123112  11
1991-01-10      123112  11
1990-01-10      123112  10
1991-02-12      384902  10
1994-01-10      302918  23
1994-02-14      283901  12
1991-02-12      384902  10
1991-02-12      384902  10
1990-01-10      302918  23
1990-02-14      283901  12
1991-02-14      283901  11
1990-02-12      384902  9
1991-01-10      302918  22
1990-01-10      302918  23
1990-02-14      283901  12
Time taken: 0.422 seconds, Fetched: 17 row(s)
```

Task 2.1

1.What is NoSQL data base?

NoSQL data base is a database management technology which does not strictly follow all the rules (ACID properties) of relational DBMS and data can not be queried using traditional sql language. That is why it is termed as NO-SQL or Not Only SQL.

2.How does data get stored in NoSQl database?

Data can be stored in four different manners :
Columnar Databases – Reads and writes columns of data rather than the rows. Each column is comparable to a container in RDBMS where a Key defines a row and single row has multiple columns.
Document Databases – Store and retrieve data in semi-structured format such as XML, JSON, etc.
Graph Databases – Stores data as entities and relations between them allowing faster traversal and joining operations to be performed.
In-Memory Key-Value Stores- Stores critical data in memory which in turn improves the performance of the systems.

3.What is a column family in HBase?
In the HBase data model columns are grouped into column families, which must be defined during table creation. A column family defines shared features to all columns that are created within them.

4.How many maximum number of columns can be added to HBase table?
There is no limit of number of column qualifiers.

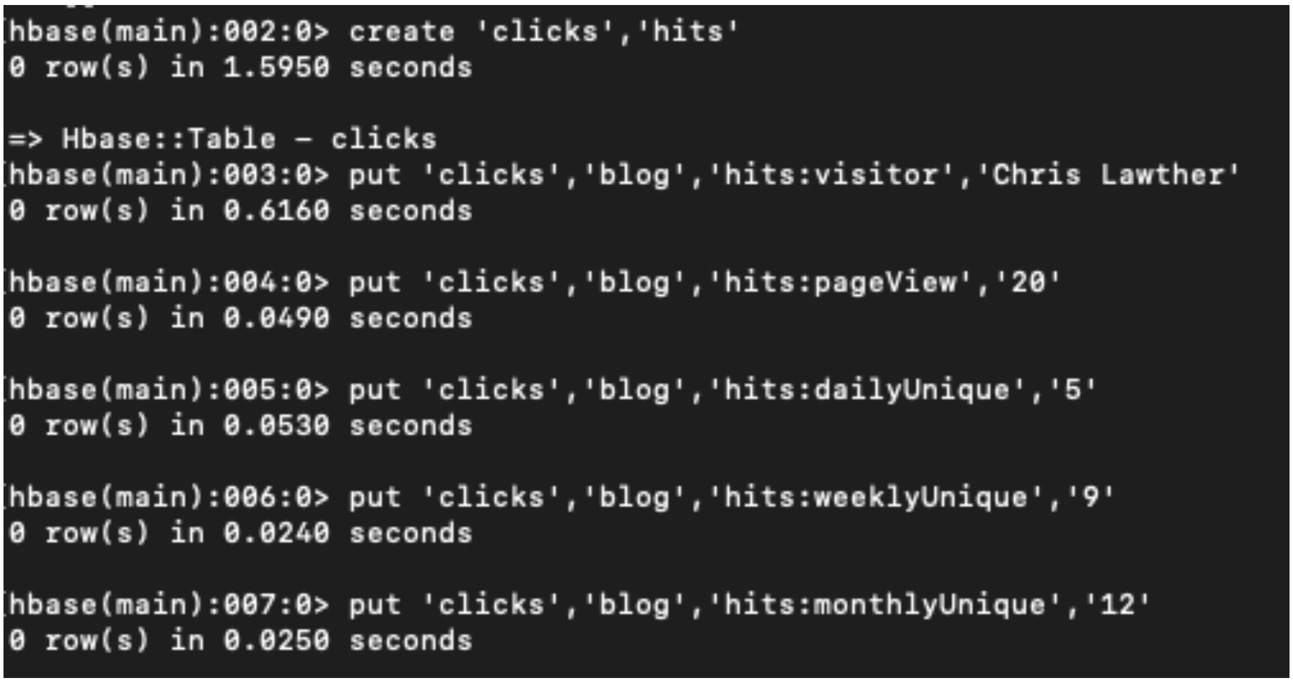
5.Why columns are not defined at the time of table creation in HBase?
column-qualifier is not defined during table creation. This is what makes HBase schema-less. Once table is created a row can be added which should belong to at-least one column-family in a table and then can belong to any column-qualifier. If the column-qualifier is not already present, it will be created.

6.How does data get managed in HBase?
Just like in a Relational Database, data in HBase is stored in Tables and these Tables are stored in Regions. When a Table becomes too big, the Table is partitioned into multiple Regions. These Regions are assigned to Region Servers across the cluster.

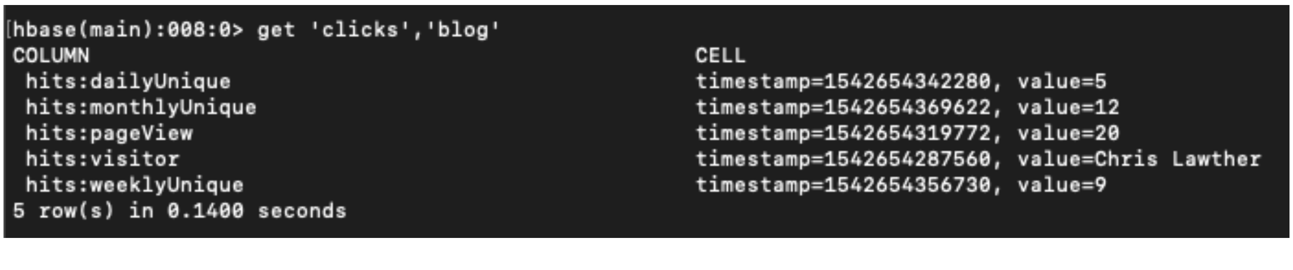
7.What happens internally when new data gets inserted into HBase table?
When the client issues a Put request:
1. write the data to the write-ahead log.
2. Once the data is written to the WAL, it is placed in the MemStore. Then, the put request acknowledgement returns to the client.
3. The MemStore stores updates in memory as sorted KeyValues, the same as it would be stored in an HFile. There is one MemStore per column family.
4. When the MemStore accumulates enough data, the entire sorted set is written to a new HFile in HDFS.

Task 2.2

1. Create an HBase table named 'clicks' with a column family 'hits' such that it should be able to store last 5 values of qualifiers inside 'hits' column family.

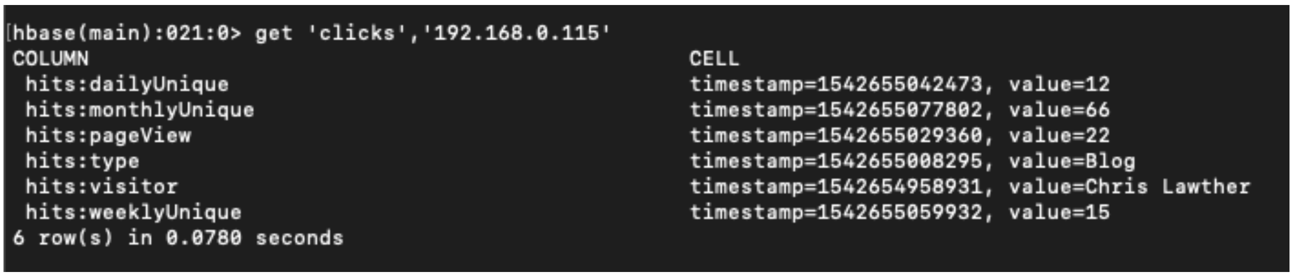


```
hbase(main):002:0> create 'clicks','hits'
0 row(s) in 1.5950 seconds

=> Hbase::Table - clicks
hbase(main):003:0> put 'clicks','blog','hits:visitor','Chris Lawther'
0 row(s) in 0.6160 seconds

hbase(main):004:0> put 'clicks','blog','hits:pageView','20'
0 row(s) in 0.0490 seconds

hbase(main):005:0> put 'clicks','blog','hits:dailyUnique','5'
0 row(s) in 0.0530 seconds

hbase(main):006:0> put 'clicks','blog','hits:weeklyUnique','9'
0 row(s) in 0.0240 seconds

hbase(main):007:0> put 'clicks','blog','hits:monthlyUnique','12'
0 row(s) in 0.0250 seconds
```

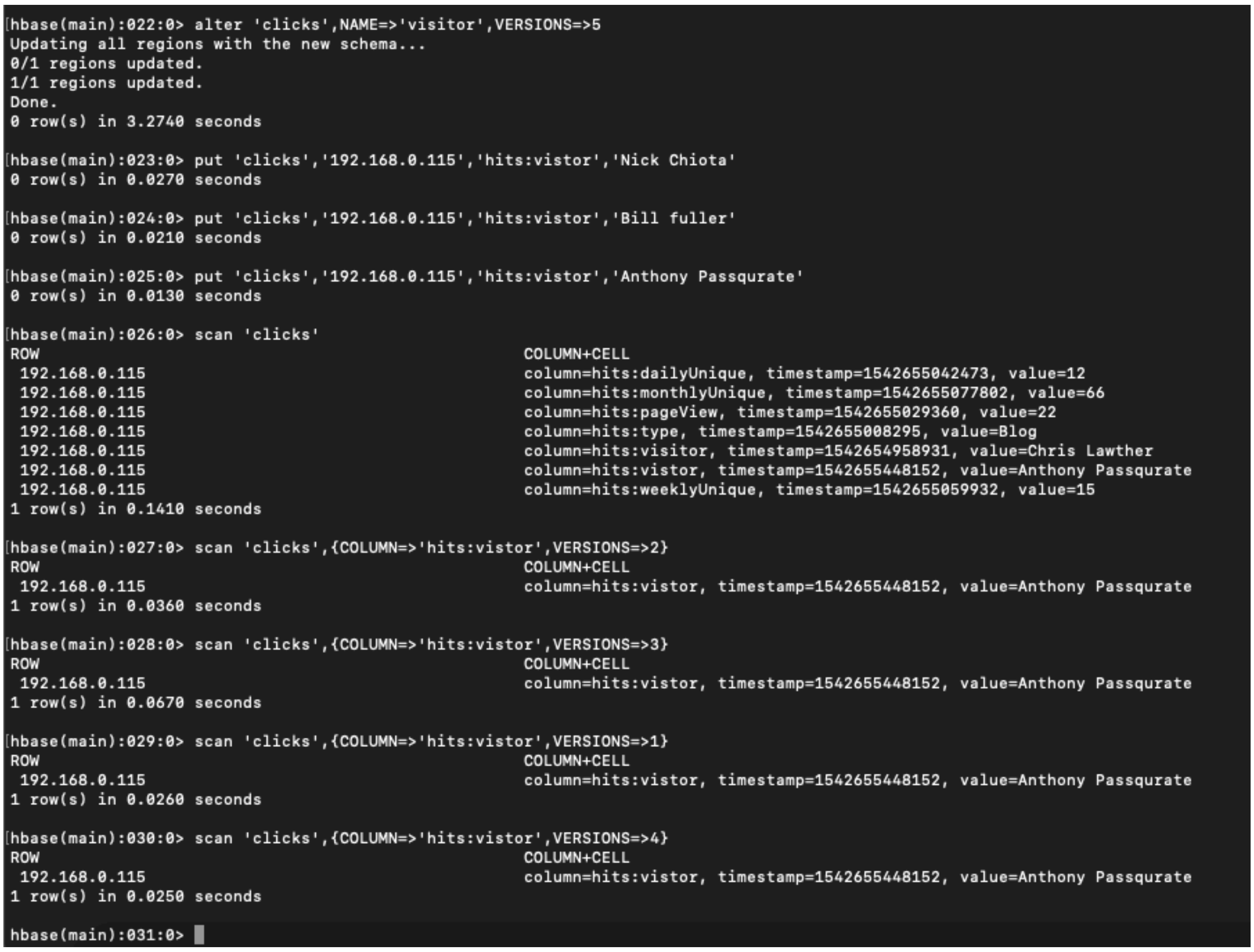


```
hbase(main):008:0> get 'clicks','blog'
COLUMN                                         CELL
 hits:dailyUnique                              timestamp=1542654342280, value=5
 hits:monthlyUnique                            timestamp=1542654369622, value=12
 hits:pageView                                 timestamp=1542654319772, value=20
 hits:visitor                                  timestamp=1542654287560, value=Chris Lawther
 hits:weeklyUnique                             timestamp=1542654356730, value=9
5 row(s) in 0.1400 seconds
```

5. Add few records in the table and update some of them. Use IP Address as row-key. Scan the table to view if all the previous versions are getting displayed.



```
hbase(main):021:0> get 'clicks','192.168.0.115'
COLUMN                                                CELL
 hits:dailyUnique                                     timestamp=1542655042473, value=12
 hits:monthlyUnique                                   timestamp=1542655077802, value=66
 hits:pageView                                        timestamp=1542655029360, value=22
 hits:type                                            timestamp=1542655008295, value=Blog
 hits:visitor                                         timestamp=1542654958931, value=Chris Lawther
 hits:weeklyUnique                                    timestamp=1542655059932, value=15
6 row(s) in 0.0780 seconds
```



```
hbase(main):022:0> alter 'clicks',NAME=>'visitor',VERSIONS=>5
Updating all regions with the new schema...
0/1 regions updated.
1/1 regions updated.
Done.
0 row(s) in 3.2740 seconds

hbase(main):023:0> put 'clicks','192.168.0.115','hits:vistor','Nick Chiota'
0 row(s) in 0.0270 seconds

hbase(main):024:0> put 'clicks','192.168.0.115','hits:vistor','Bill fuller'
0 row(s) in 0.0210 seconds

hbase(main):025:0> put 'clicks','192.168.0.115','hits:vistor','Anthony Passqurate'
0 row(s) in 0.0130 seconds

hbase(main):026:0> scan 'clicks'
ROW                                         COLUMN+CELL
 192.168.0.115                              column=hits:dailyUnique, timestamp=1542655042473, value=12
 192.168.0.115                              column=hits:monthlyUnique, timestamp=1542655077802, value=66
 192.168.0.115                              column=hits:pageView, timestamp=1542655029360, value=22
 192.168.0.115                              column=hits:type, timestamp=1542655008295, value=Blog
 192.168.0.115                              column=hits:visitor, timestamp=1542654958931, value=Chris Lawther
 192.168.0.115                              column=hits:vistor, timestamp=1542655448152, value=Anthony Passqurate
 192.168.0.115                              column=hits:weeklyUnique, timestamp=1542655059932, value=15
1 row(s) in 0.1410 seconds

hbase(main):027:0> scan 'clicks',{COLUMN=>'hits:vistor',VERSIONS=>2}
ROW                                         COLUMN+CELL
 192.168.0.115                              column=hits:vistor, timestamp=1542655448152, value=Anthony Passqurate
1 row(s) in 0.0360 seconds

hbase(main):028:0> scan 'clicks',{COLUMN=>'hits:vistor',VERSIONS=>3}
ROW                                         COLUMN+CELL
 192.168.0.115                              column=hits:vistor, timestamp=1542655448152, value=Anthony Passqurate
1 row(s) in 0.0670 seconds

hbase(main):029:0> scan 'clicks',{COLUMN=>'hits:vistor',VERSIONS=>1}
ROW                                         COLUMN+CELL
 192.168.0.115                              column=hits:vistor, timestamp=1542655448152, value=Anthony Passqurate
1 row(s) in 0.0260 seconds

hbase(main):030:0> scan 'clicks',{COLUMN=>'hits:vistor',VERSIONS=>4}
ROW                                         COLUMN+CELL
 192.168.0.115                              column=hits:vistor, timestamp=1542655448152, value=Anthony Passqurate
1 row(s) in 0.0250 seconds

hbase(main):031:0>
```