Assignment11

Task1:

Explain the below concepts with an example in brief.

- Nosql Databases

- NoSQL is an approach to database design that can accomodate a wide variety of data models, including key-value, document, columnar and graph formats.
- NoSQL, which stand for "not only SQL," is an alternative to traditional relational databases in which data is placed in tables and data schema is carefully designed before the database is built.
- NoSQL databases are especially useful for working with large sets of distributed data.
- NoSQL databases are very flexible in structure and can store all types of related data in one place
- User can retrieve the whole post with a single query avoiding joins thus increasing the performance
- Data on NoSQL databases scale out naturally and hence able to deal with the continuous streaming of posts
- Features:-
    - Generic Data Model :–

      Heterogeneous containers, including sets, maps, and arrays
    - Dynamic type discovery and conversion :–

      NoSQL analytics systems support runtime type identification and conversion so that custom business logic can be used to dictate analytic treatment of variation.
    - Non-relational and De-normalised :-

      Data is stored in single tables as compared to joining multiple tables.
    - Commodity hardware : –

      Adding more of the economical servers allows NoSQL databases to scale to handle more data.
    - Highly distributable :–

      Distributed databases can store and process a set of information on more than one device.
- Example: MongoDB, Cassandra

- Types of Nosql Databases

There are 4 basic types of NoSQL databases:
i)Key-Value Store :-

- Key value type basically, uses a hash table in which there exists a unique key and a
- pointer to a particular item of data.
- A bucket is a logical group of keys – but they don’t physically group the data.
- There can be identical keys in different buckets. The data which is a collection of key value pairs is compressed as a document store quite similar to a key-value store, but the only difference is that the values stored (referred to as “documents”) provide some structure and encoding of the managed data.
- There is no complexity around the Key Value Store database model as it can be
  implemented in a breeze
- Example- Riak, Amazon S3 (Dynamo)

ii) Document-based Store :-

- The data which is a collection of key value pairs is compressed as a document store quite similar to a key-value store but the only difference is that the values stored (referred to as “documents”)provide some structure and encoding of the managed data.
- It stores documents made up of tagged elements.
- XML, JSON (Java Script Object Notation), BSON (which is a binary encoding of JSON objects) are some common standard encodings
- Example- CouchDB

iii)Column-based Store :-

- In column-oriented NoSQL database, data is stored in cells grouped in columns of data rather than as rows of data.
- Columns are logically grouped into column families.
- Column families can contain a virtually unlimited number of columns that can be created at runtime or the definition of the schema.
- Read and write is done using columns rather than rows.
- Each storage block contains data from only one column.
- Column oriented databases are faster in access compared to row oriented like RDBMS as data retrieval become easy when stored as seperate column.

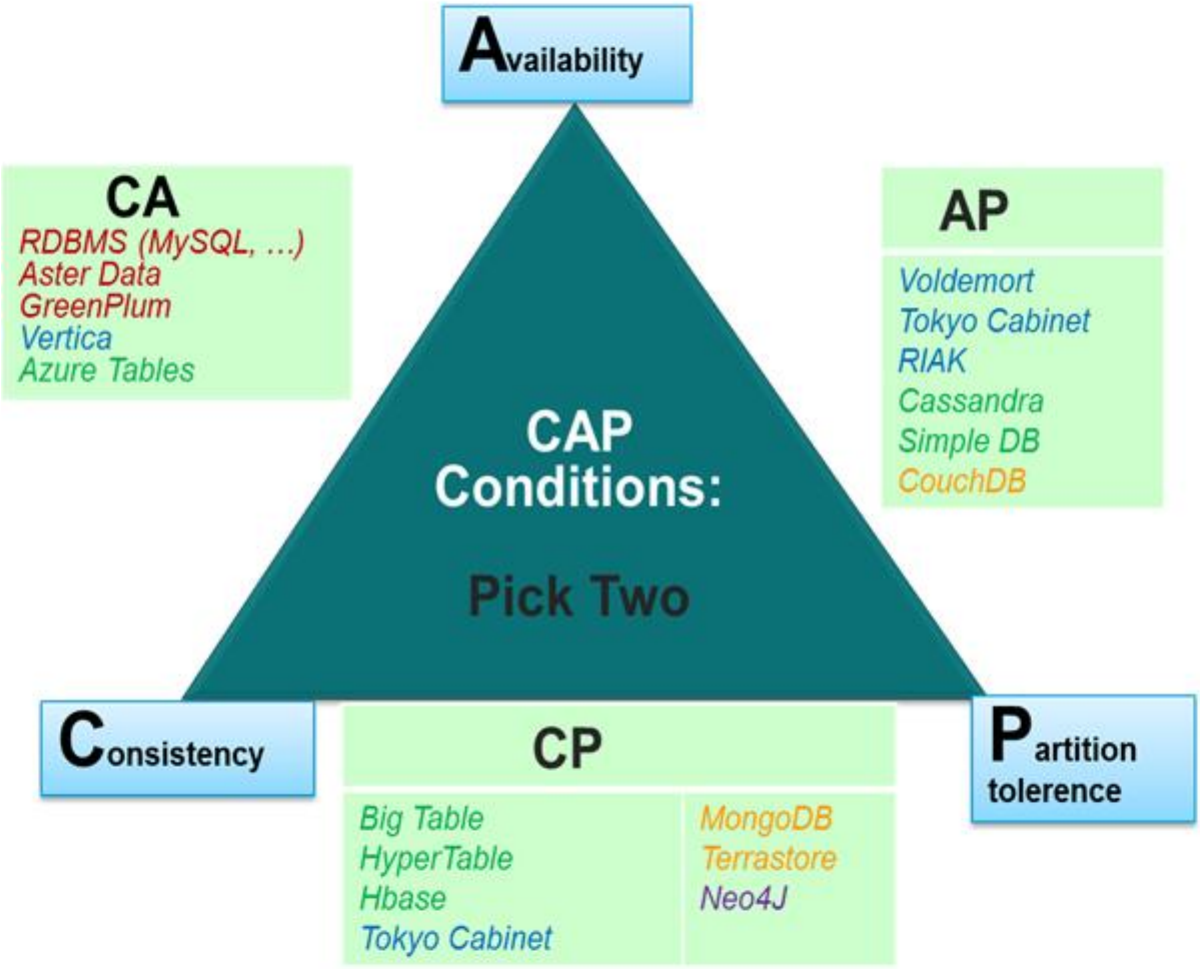
- HBASE uses column databases.
- Data is inserted in seperate column format
- Example- HBase, Cassandra

iv) Graph-based :-

- In a Graph Base NoSQL Database, you will not find the rigid format of SQL or the tables and columns representation, a flexible graphical representation is instead used which is perfect to address scalability concerns.
- Graph structures are used with edges,nodes and properties which provides index-free adjacency.
- A network database that uses edges and nodes to represent and store data.
- Data can be easily transformed from one model to the other using a Graph Base NoSQL database.
- Example- Neo4J

● CAP Theorem

- It stands for Consistency Availability Partition Tolerance
- Consistency-This means that the data in the database remains consistent after the
- execution of an operation. For example after an update operation, all clients see the
- same data.
- Availability-This means that the system is always on (service guarantee availability),
- no downtime.
- Partition Tolerance-This means that the system continues to function even if the
- communication among the servers is unreliable, i.e. the servers may be partitioned.



- HBase Architecture

HBASE is a distributed column oriented database built on top of hadoop to provide
real time access to Big Data.
HBase is composed of three types of servers in a master slave type of architecture.
Region servers serve data for reads and writes.
HBase Master process handles the Region assignment, DDL (create, delete tables) operations
Zookeeper maintains a live cluster state.
The Hadoop Data Node stores the data that the Region Server is managing.
All HBase data is stored in HDFS files.
The Name Node maintains metadata information for all the physical data blocks that comprise the files.

- HBase vs RDBMS

HBase	RDBMS
HBase is a distributed, column-oriented data Storage system	RDBMS is row-oriented databases
HBase table have Flexible schema	RDBMS tables have fixed-schema
Transaction are done in Single row only	Transactions are done in multiple row
Max data size 1PB	Max data size TBs
HBase uses Java client API and Jruby	RDBMS uses SQL (Structured query Language) to query the data
Read/write throughput limits -> Millions of queries/second	Read/write throughput limits -> 1000s queries/second

Task2

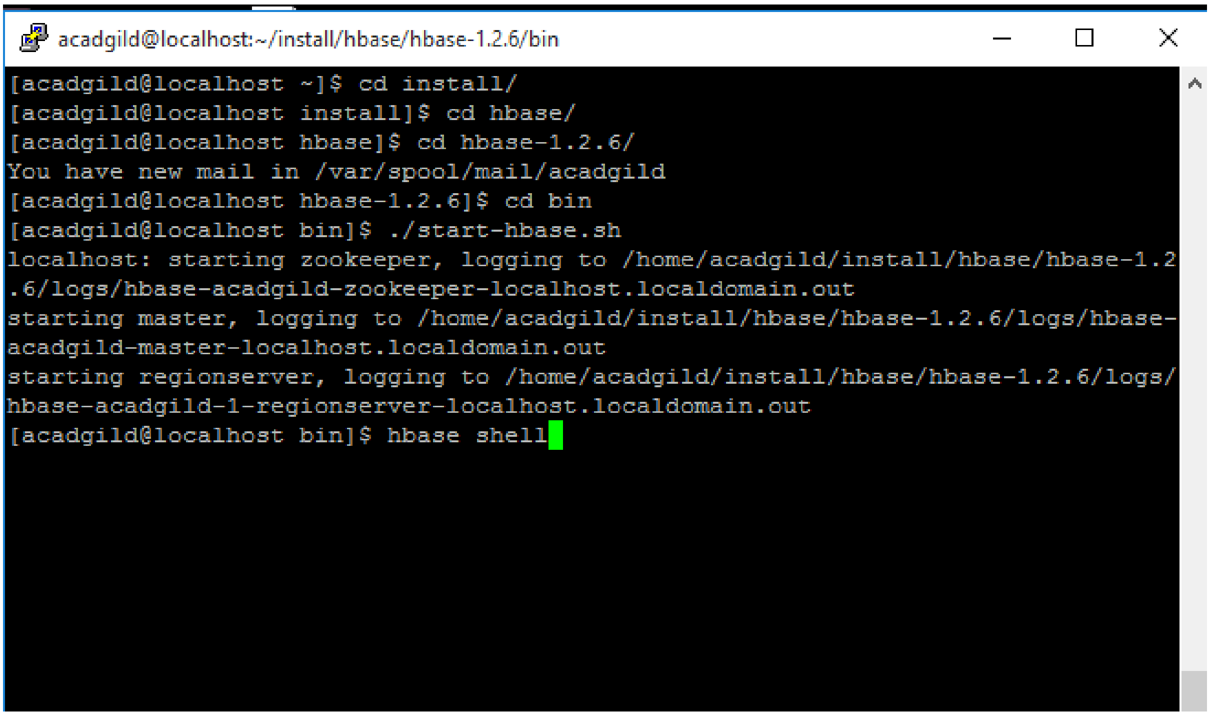
Execute blog present in below link

https://acadgild.com/blog/importtsv-data-from-hdfs-into-hbase/

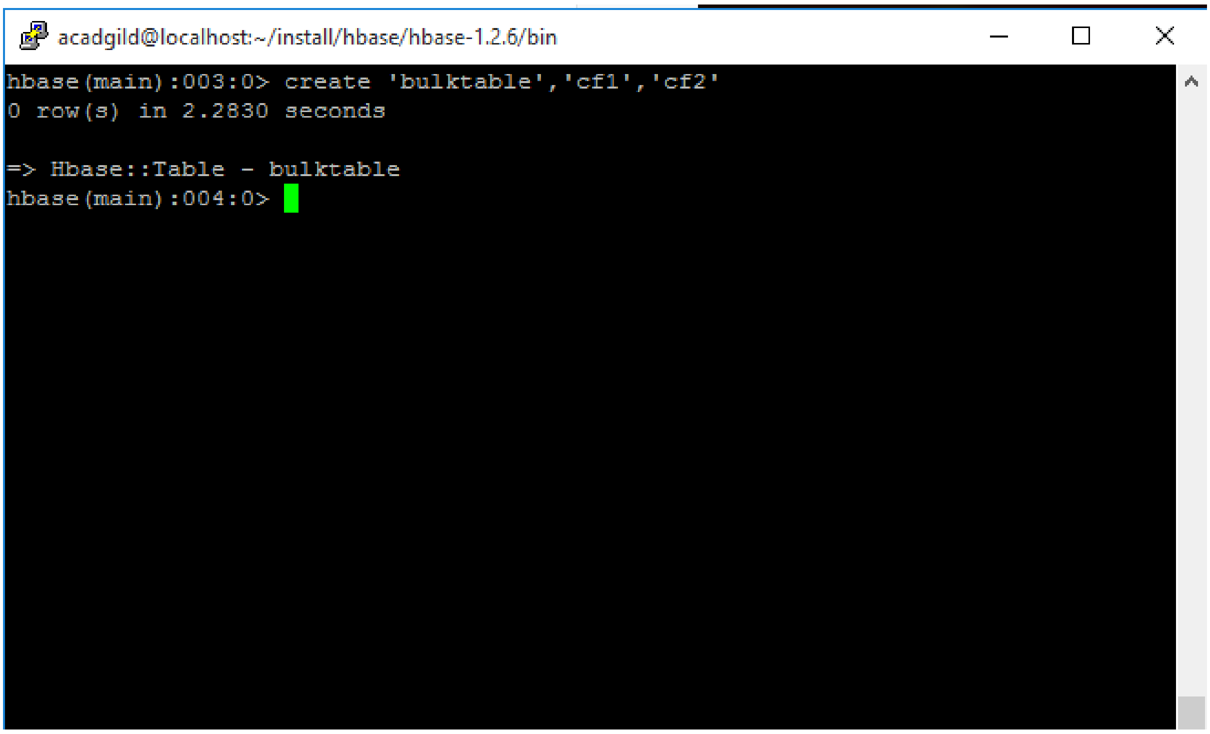
Step1:

Inside Hbase shell give the following command to create table along with 2 column family.

Create 'bulktable', 'cf1', 'cf2'



```
[acadgild@localhost ~]$ cd install/
[acadgild@localhost install]$ cd hbase/
[acadgild@localhost hbase]$ cd hbase-1.2.6/
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost hbase-1.2.6]$ cd bin
[acadgild@localhost bin]$ ./start-hbase.sh
localhost: starting zookeeper, logging to /home/acadgild/install/hbase/hbase-1.2
.6/logs/hbase-acadgild-zookeeper-localhost.localdomain.out
starting master, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/hbase-
acadgild-master-localhost.localdomain.out
starting regionserver, logging to /home/acadgild/install/hbase/hbase-1.2.6/logs/
hbase-acadgild-1-regionserver-localhost.localdomain.out
[acadgild@localhost bin]$ hbase shell
```



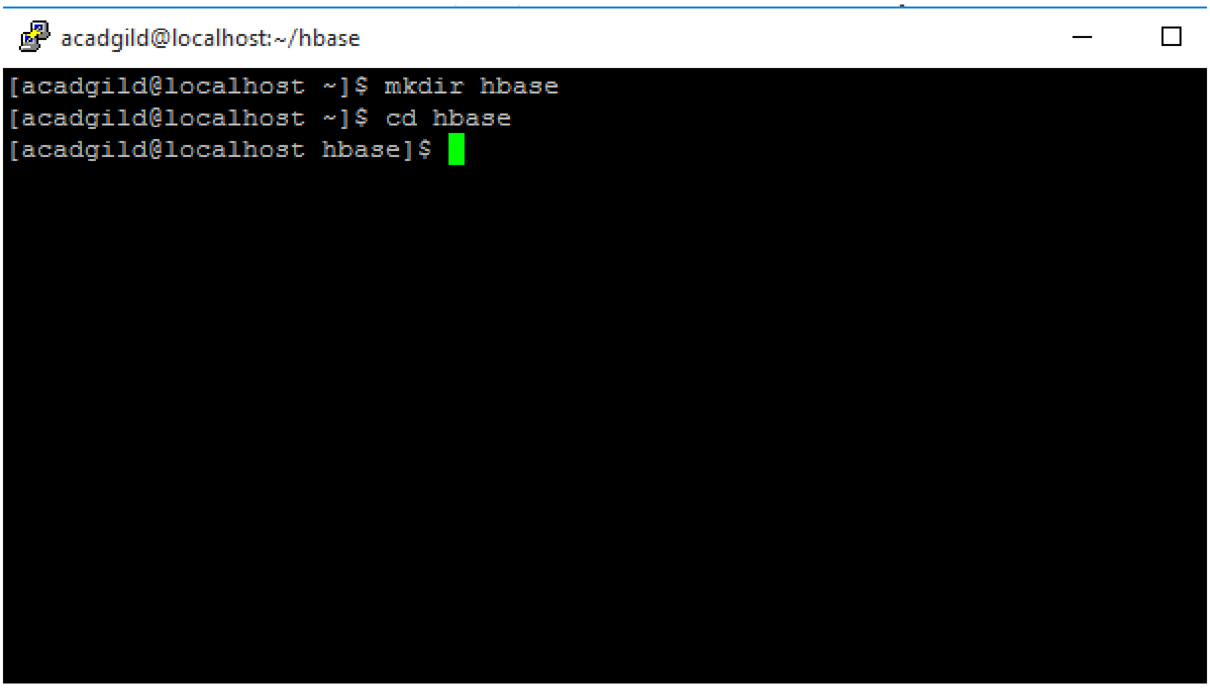
```
hbase(main):003:0> create 'bulktable','cf1','cf2'
0 row(s) in 2.2830 seconds

=> Hbase::Table - bulktable
hbase(main):004:0>
```

Step2:

Come out of HBase shell to the terminal and also make a directory for Hbase in the local drive; So since you have your own path you can use it.

mkdir hbase

cd hbase



```
[acadgild@localhost ~]$ mkdir hbase
[acadgild@localhost ~]$ cd hbase
[acadgild@localhost hbase]$
```

Step3:

Create a file inside the HBase directory named bulk_data.tsv with tab separated data inside using below command in terminal.

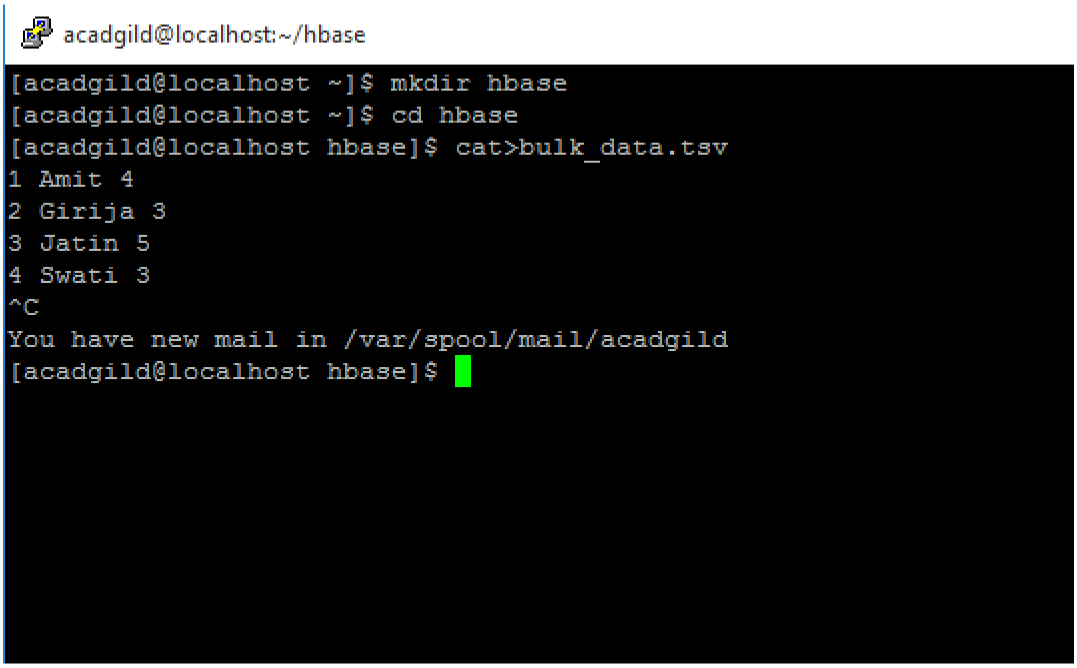
Cat&gt;bulk_data.tsv

1 Amit 4

2 Girija 3

3 Jatin 5

4 Swati 3



```
[acadgild@localhost ~]$ mkdir hbase
[acadgild@localhost ~]$ cd hbase
[acadgild@localhost hbase]$ cat>bulk_data.tsv
1 Amit 4
2 Girija 3
3 Jatin 5
4 Swati 3
^C
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost hbase]$
```

Step4:

Our data should be present in HDFS while performing the import task to Hbase.

In real time projects, the data will already be present inside HDFS.

Here for our learning purpose, we copy the data inside HDFS using below

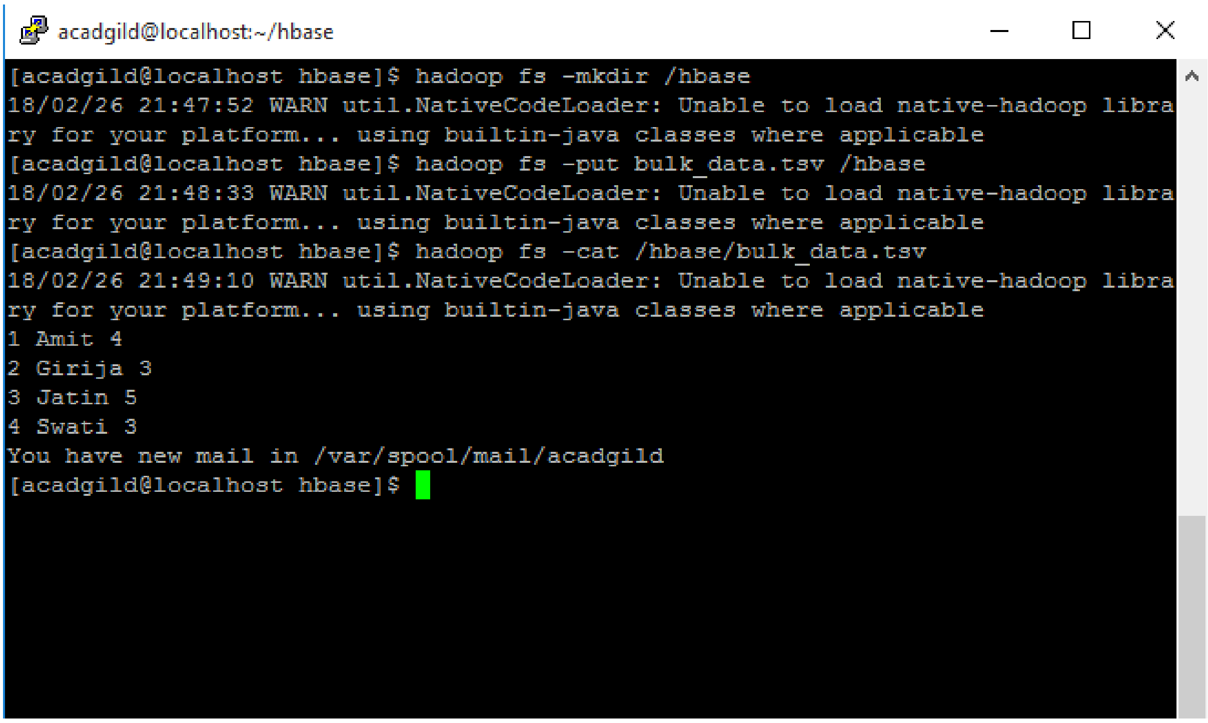
commands in terminal.

Commands:

hadoop fs -mkdir /hbase

hadoop fs -put bulk_data.tsv /hbase/

hadoop fs -cat /hbase/bulk_data.tsv



acadgild@localhost:~/hbase

```
[acadgild@localhost hbase]$ hadoop fs -mkdir /hbase
18/02/26 21:47:52 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
[acadgild@localhost hbase]$ hadoop fs -put bulk_data.tsv /hbase
18/02/26 21:48:33 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
[acadgild@localhost hbase]$ hadoop fs -cat /hbase/bulk_data.tsv
18/02/26 21:49:10 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
1 Amit 4
2 Girija 3
3 Jatin 5
4 Swati 3
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost hbase]$
```

Step5:

After the data is present now in HDFS.In terminal, we give the following

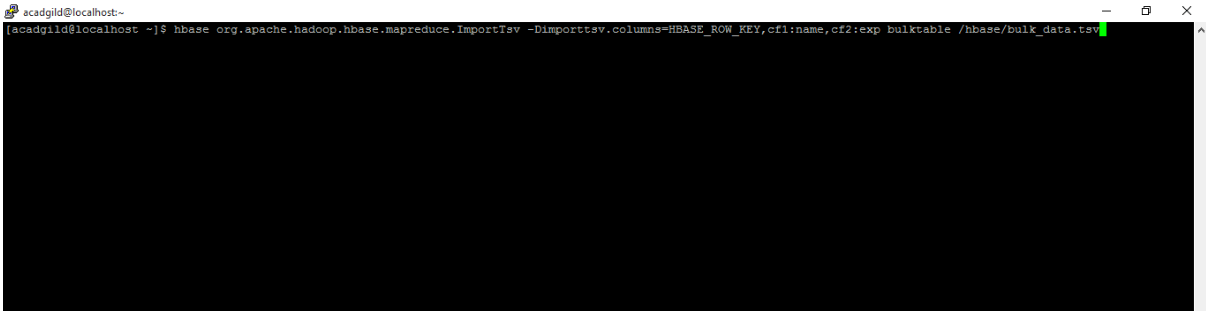
command along with arguments&lt;tablename&gt; and &lt;path of data in HDFS&gt;

Command:

hbase org.apache.hadoop.hbase.mapreduce.ImportTsv

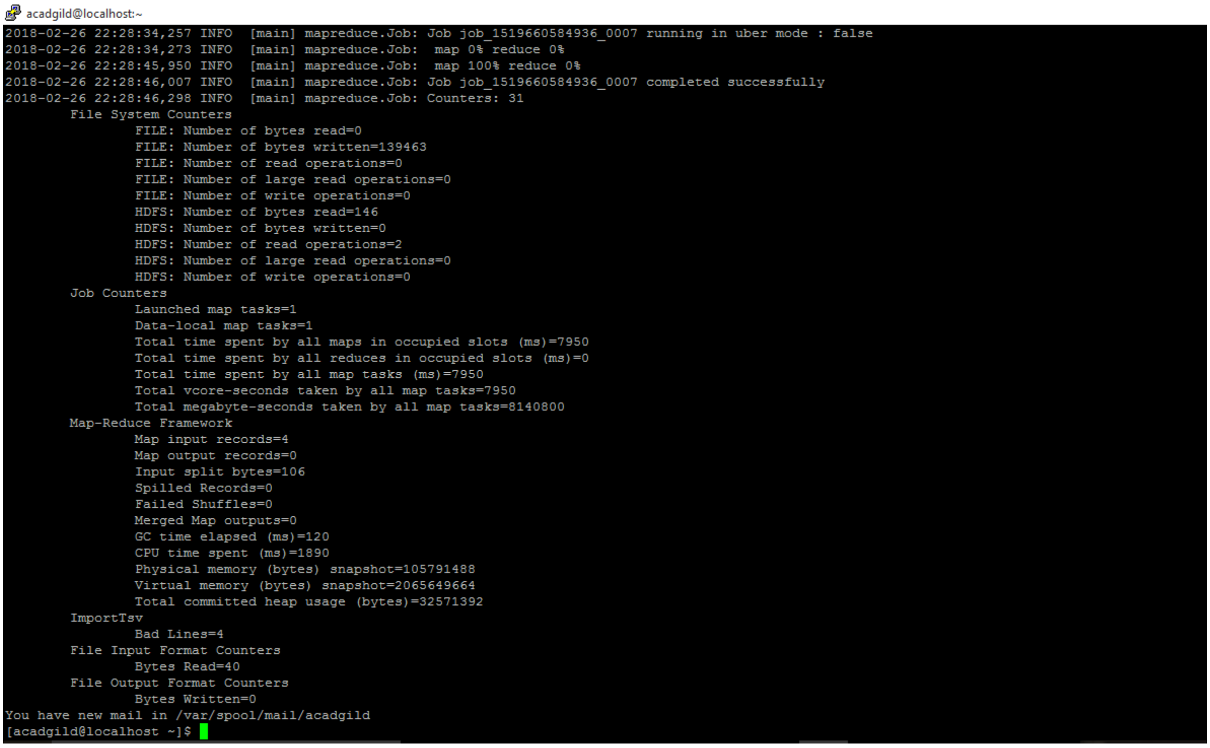
–Dimporttsv.columns=HBASE_ROW_KEY,cf1:name,cf2:exp

bulktable /hbase/bulk_data.tsv



acadgild@localhost:~

```
[acadgild@localhost ~]$ hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.columns=HBASE_ROW_KEY,cf1:name,cf2:exp bulktable /hbase/bulk_data.tsv
```



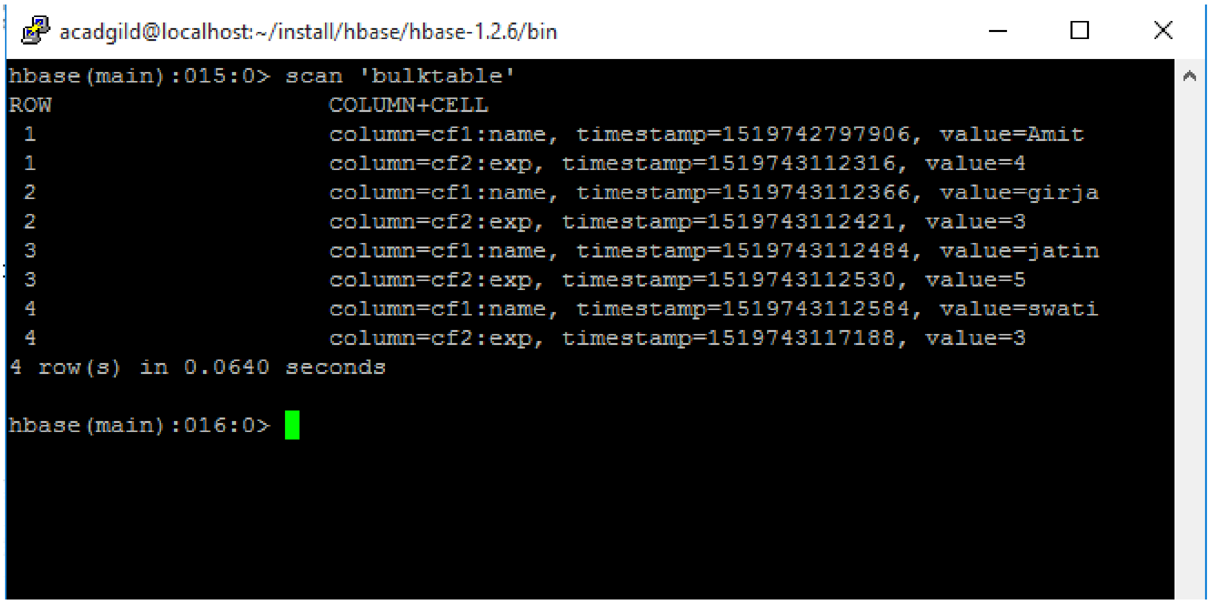
```
2018-02-26 22:28:34,257 INFO  [main] mapreduce.Job: Job job_1519660584936_0007 running in uber mode : false
2018-02-26 22:28:34,273 INFO  [main] mapreduce.Job:  map 0% reduce 0%
2018-02-26 22:28:45,950 INFO  [main] mapreduce.Job:  map 100% reduce 0%
2018-02-26 22:28:46,007 INFO  [main] mapreduce.Job: Job job_1519660584936_0007 completed successfully
2018-02-26 22:28:46,298 INFO  [main] mapreduce.Job: Counters: 31
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=139463
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=146
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=2
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=0
        Job Counters
                Launched map tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=7950
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=7950
                Total vcore-seconds taken by all map tasks=7950
                Total megabyte-seconds taken by all map tasks=8140800
        Map-Reduce Framework
                Map input records=4
                Map output records=0
                Input split bytes=106
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=120
                CPU time spent (ms)=1890
                Physical memory (bytes) snapshot=105791488
                Virtual memory (bytes) snapshot=2065649664
                Total committed heap usage (bytes)=32571392
        ImportTsv
                Bad Lines=4
        File Input Format Counters
                Bytes Read=40
        File Output Format Counters
                Bytes Written=0
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ 
```

Observe that the map is done 100% although we get an error afterward.

For now, ignore the error message due to our task is to map data in HBase table.

Now,also let us check whether we actually got the data inside HBase by using the below command.

Scan ‘bulktable’



```
hbase(main):015:0> scan 'bulktable'
ROW                      COLUMN+CELL
 1                       column=cf1:name, timestamp=1519742797906, value=Amit
 1                       column=cf2:exp, timestamp=1519743112316, value=4
 2                       column=cf1:name, timestamp=1519743112366, value=girja
 2                       column=cf2:exp, timestamp=1519743112421, value=3
 3                       column=cf1:name, timestamp=1519743112484, value=jatin
 3                       column=cf2:exp, timestamp=1519743112530, value=5
 4                       column=cf1:name, timestamp=1519743112584, value=swati
 4                       column=cf2:exp, timestamp=1519743117188, value=3
4 row(s) in 0.0640 seconds

hbase(main):016:0>
```