

Assignment 10

Task1

Explain the below concepts with an example in brief

a. NoSQL databases

NoSQL means Not Only SQL.. Nowadays, with the rise of unstructured or semi-structured data, the storage or management of data is quite cumbersome for the relational databases. Such data are hard to be handled on the cluster. A NoSQL database solves this problem. A NoSQL database:

- Runs well on clusters
- Is an open-source
- Is schema-less
- Can support large volumes of data by running on clusters.
- File-based database and is not based on relational model of storing data.
- Is highly distributable.

Example: MongoDB, Cassandra, HBase databases.

b. Types of Nosql Databases

NoSQL databases are classified into 4 types:

- Key Value Store NoSQL Database

The key value type basically, uses a hash table in which there exists a unique key and a pointer to a particular item of data. There can be identical keys in different buckets. Performance is enhanced to a great degree because of the cache mechanisms that accompany the mappings. Not an ideal method if we are only looking to just update part of a value or query the database.

Demerits

- This model will not provide any kind of traditional database capabilities. Such capabilities must be provided by the application itself.
- As the volume of data increases, maintaining unique values as keys may become more difficult; addressing this issue requires the introduction of some complexity in generating character strings that will remain unique among an extremely large set of keys.

Example: Riak, Amazon's Dynamo

- Document Store NoSQL Database

This is 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. XML, JSON (Java Script Object Notation), BSON (which is a binary encoding of JSON objects) are some common standard encodings.

Since these databases are schema-less, adding fields to JSON documents becomes a simple task without having to define changes first.

Example: CouchBase, MongoDB

- Column Store NoSQL Database

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. Read and write is done using columns rather than rows.

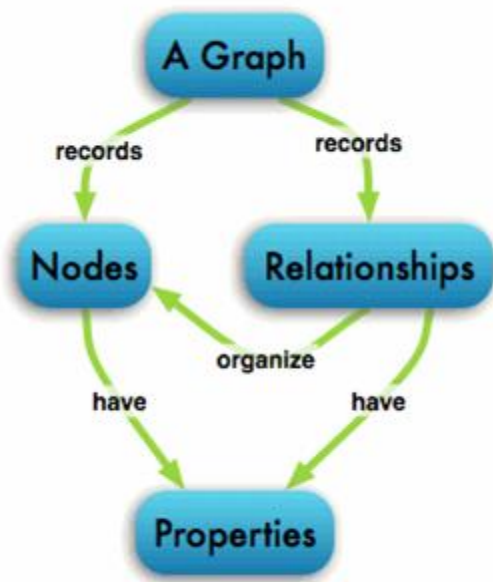
Relational databases store a single row as a continuous disk entry. Different rows are stored in different places on disk while Columnar databases store all the cells corresponding to a column as a continuous disk entry thus makes the search/access faster.

Example: HBase, Google's Big Table, Cassandra

- Graph Base NoSQL Database

Graph structures are used with edges, nodes and properties which provides index-free adjacency. Data can be easily transformed from one model to the

other using a Graph Base NoSQL database. In graph databases, traversing the joins or relationships is very fast. The relationship between nodes is not calculated at query time but is actually persisted as a relationship. Traversing persisted relationships is faster than calculating them for every query.



- These databases use edges and nodes to represent and store data.
- These nodes are organized by some relationships with one another, which is represented by edges between the nodes.
- Both the nodes and the relationships have some defined properties.

Example: InfoGrid, Infinite Graph

c. CAP Theorem

The CAP theorem states that in any distributed system we can choose only two of consistency, availability or partition tolerance, out of ACID properties of the RDBMS databases. The CAP theorem states that if you get a network partition, you have to trade off availability of data versus consistency of data. Durability can also be traded off against latency, particularly if you want to survive failures with replicated data.

Duplicate copy of same data is maintained on multiple machines. This decreases the consistency but the availability increases.

d. HBase Architecture

HBase is composed of three types of servers in a master slave type of architecture.

- Region servers
 - serve data for reads and writes.
 - They can be added or removed.
 - When accessing data, clients connect to Region Servers directly.
 - Region Servers serve the purpose of Data Node in Hbase.
 - It is the slave daemon of HBase.
 - HBase tables are horizontally divided by row key range into 'Regions', which contains all rows in the table.
 - Regions are similar to buckets in Hive. It can handle 1GB of data.
- HBase Master
 - Handles the Region assignment, DDL (create, delete tables) operations.
 - It is similar to Name Node in HDFS.
 - It re-assigns regions for every recovery on load balancing.
 - It is also responsible for co-ordination with the region servers.
 - Performs admin functions like deleting, updating regions.
- Zookeeper
 - It maintains a live cluster state.
 - It is an open source distributed technology.
 - It automates distributed co-ordination.
 - Zookeeper maintains reliable server state in the cluster.
 - It provides server notification failure.

The Hadoop DataNode stores the data that the Region Server is managing. All HBase data is stored in HDFS files. The NameNode maintains metadata information for all the physical data blocks that comprise the files. The client gets the Region server that hosts the META table from ZooKeeper.

e. HBase vs RDBMS

HBASE	RDBMS
-------	-------

Distributed, column-oriented database	Row-oriented database.
Schema-less	Fixed schema
Guarantee consistency and partition tolerance	Guarantee ACID properties
Uses JAVA client API and JRuby	Uses SQL to query the data.

Task2

Execute the commands to import Tsv data from HDFS to HBASE table

Step1

Creation of table,'bulktable' on hbase

The screenshot shows a MobaXterm window with a terminal session on a 127.0.0.1 (acadgild) host. The terminal output shows the execution of the HBase command to create a table:

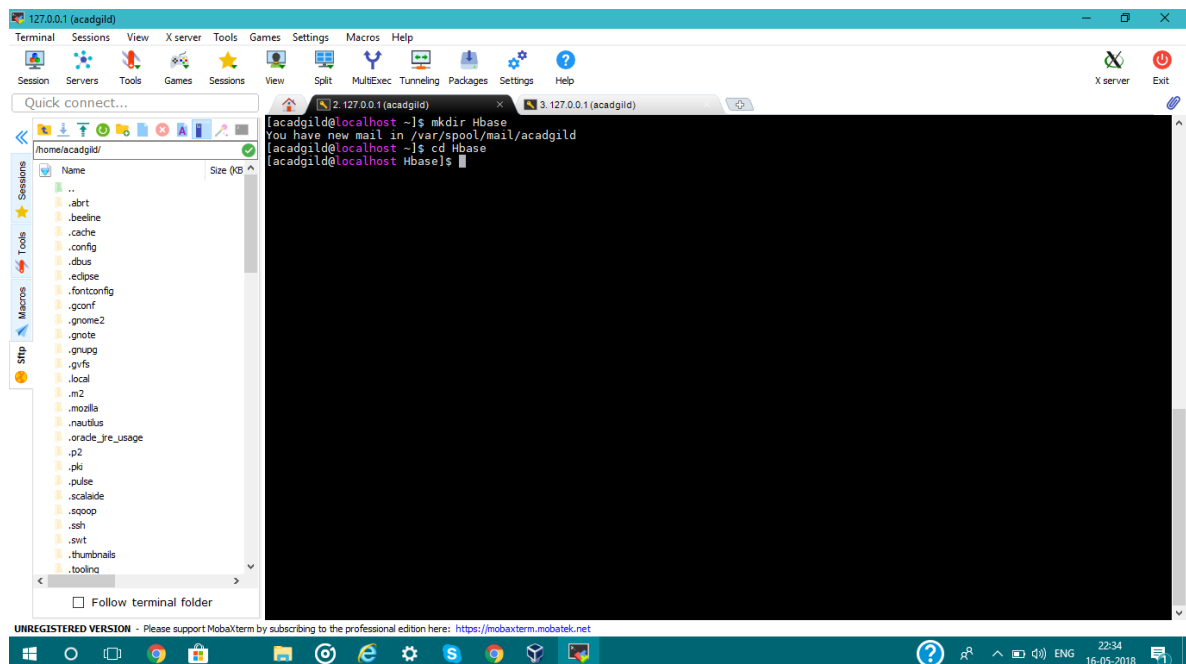
```
hbase(main):006:0> create 'bulktable', 'cf1', 'cf2'
0 row(s) in 2.9270 seconds
=> Hbase::Table - bulktable
hbase(main):007:0>
```

On the left side of the MobaXterm window, there is a file explorer pane showing the local file system structure under the path /home/acadgild/. The explorer lists various directories and files, including .abrt, .beeline, .cache, .config, .dbus, .eclipse, .fontconfig, .gconf, .gnome2, .gnote, .gnupg, .gvfs, .local, .m2, .mozilla, .nautilus, .oracle_jre_usage, .p2, .gk, .pulse, .scalaide, .sqoop, .ssh, .swt, .thumbnails, and .tooling. The 'Follow terminal folder' checkbox is checked.

At the bottom of the window, a status bar indicates the software is an 'UNREGISTERED VERSION' and provides a link to the professional edition: <https://mobaxterm.mobatek.net>. The system clock shows the time as 01:45 on 10-05-2018.

Step2

Creation of hbase directory



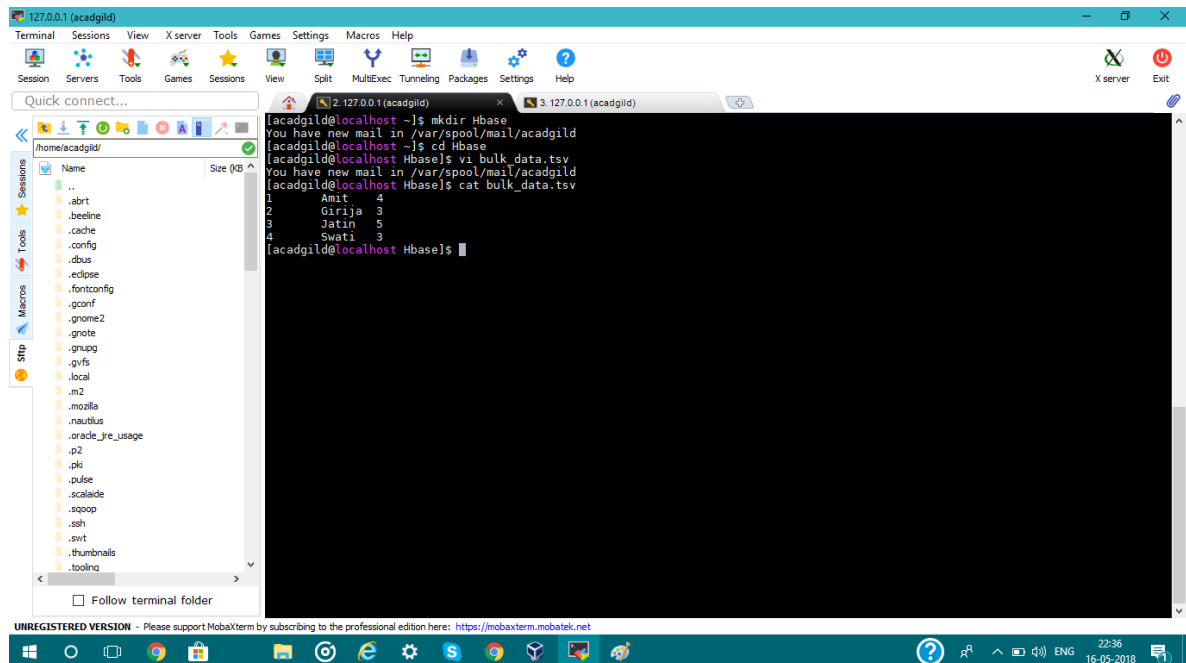
The screenshot shows the MobaXterm interface with a terminal window. The terminal displays the following commands and output:

```
[acagild@localhost ~]$ mkdir Hbase
You have new mail in /var/spool/mail/acagild
[acagild@localhost ~]$ cd Hbase
[acagild@localhost Hbase]$
```

The left sidebar shows a file explorer view of the local system, and the bottom status bar indicates the system is running on 16-05-2018 at 22:34.

Step3

Creation of 'bulk_data.tsv' in above directory



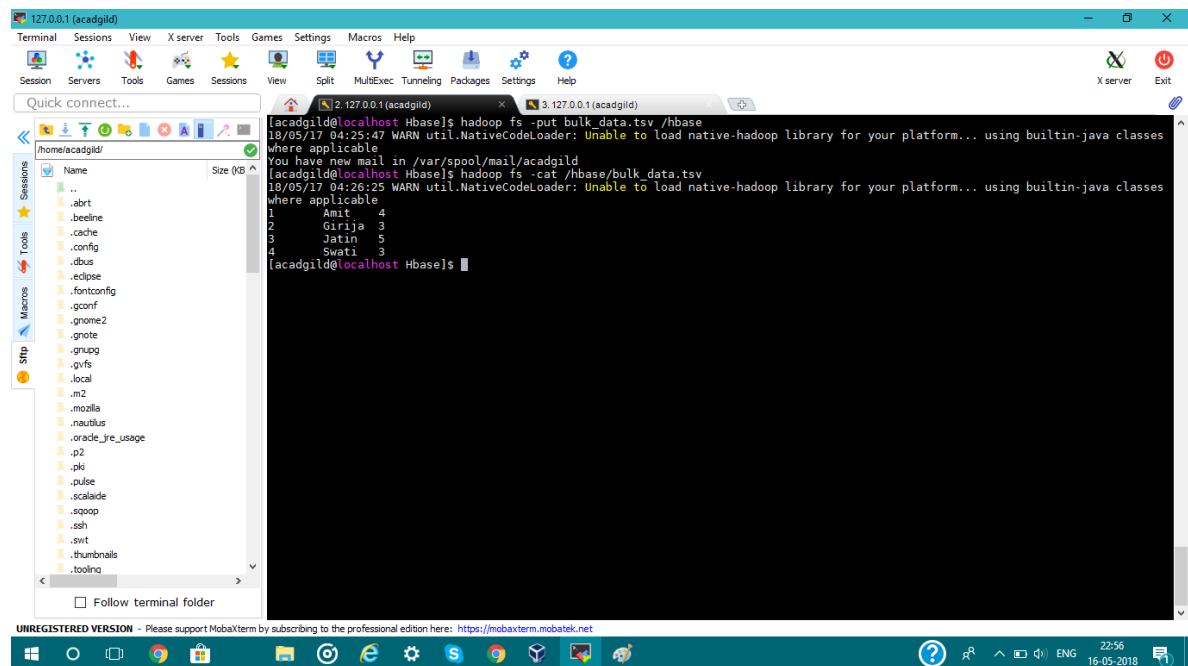
The screenshot shows the MobaXterm interface with a terminal window. The terminal displays the following commands and output:

```
[acagild@localhost ~]$ mkdir Hbase
You have new mail in /var/spool/mail/acagild
[acagild@localhost ~]$ cd Hbase
[acagild@localhost Hbase]$ vi bulk_data.tsv
[acagild@localhost Hbase]$ cat bulk_data.tsv
1      Amit      4
2      Girija    3
3      Jatin     5
4      Sveti     3
[acagild@localhost Hbase]$
```

The left sidebar shows a file explorer view of the local system, and the bottom status bar indicates the system is running on 16-05-2018 at 22:36.

Step4

Putting the above created file on HDFS



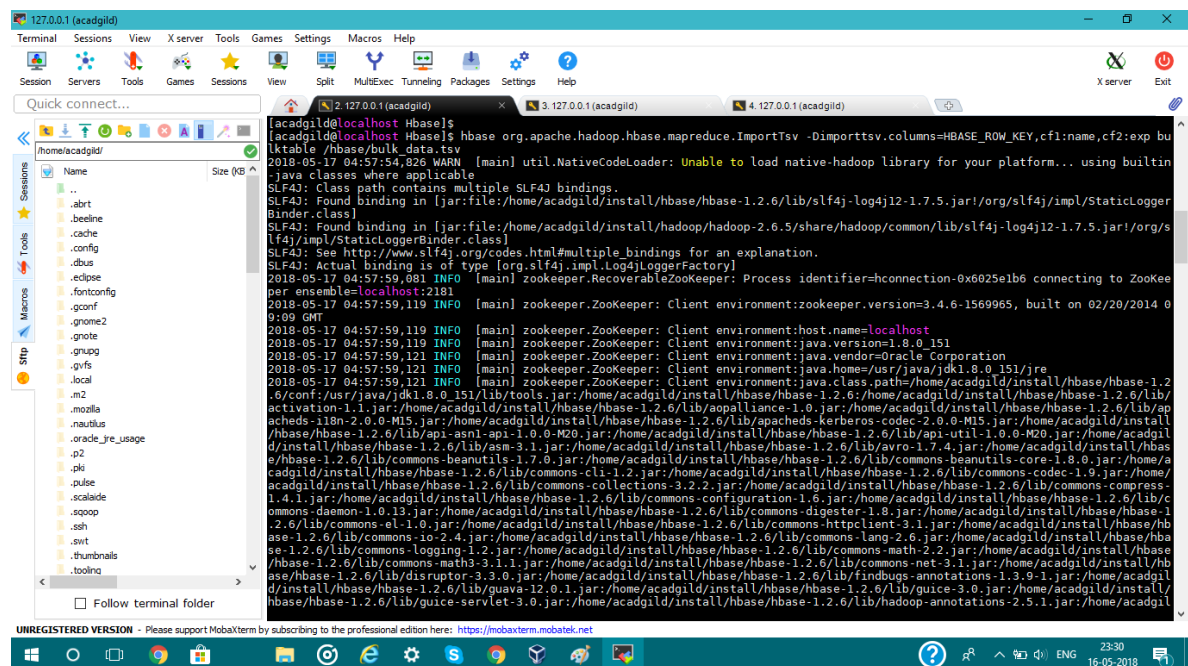
The screenshot shows a MobaXterm terminal window with the following commands and output:

```
[acadgild@localhost Hbase]$ hadoop fs -put bulk_data.tsv /hbase
18/05/17 04:25:47 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost Hbase]$ hadoop fs -cat /hbase/bulk_data.tsv
18/05/17 04:26:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
1      Amit      4
2      Girija    3
3      Jatin     5
4      Swati     3
[acadgild@localhost Hbase]$
```

The left sidebar shows the file explorer for the terminal session, listing files like .abrt, .baseline, .cache, .config, .dbus, .edipse, .fontconfig, .gnome2, .gnote, .gnupg, .gvfs, .local, .m2, .mozilla, .nautilus, .oracle_jre_usage, .p2, .pki, .pulse, .scalade, .sftp, .ssh, .swt, .thumbnails, and .tooling.

Step5

Importing the data from HDFS to HBASE table



The screenshot shows a MobaXterm terminal window with the following commands and output:

```
[acadgild@localhost Hbase]$ hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.columns=HBASE_ROW_KEY,cf1:name,cf2:exp bu
lktable /hbase/bulk_data.tsv
2018-05-17 04:57:54,826 WARN [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin
-java classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hbase/hbase-1.2.6/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLogger
Binder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/s
lf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
2018-05-17 04:57:59,081 INFO [main] zookeeper.RecoverableZooKeeper: Process identifier=hconnection-0x6025e1b6 connecting to ZooKee
per ensemble=localhost:2181
2018-05-17 04:57:59,119 INFO [main] zookeeper.ZooKeeper: Client environment:zookeeper.version=3.4.6-1569965, built on 02/20/2014 0
9:09 GMT
2018-05-17 04:57:59,119 INFO [main] zookeeper.ZooKeeper: Client environment:host.name=localhost
2018-05-17 04:57:59,119 INFO [main] zookeeper.ZooKeeper: Client environment:java.version=1.8.0_151
2018-05-17 04:57:59,121 INFO [main] zookeeper.ZooKeeper: Client environment:java.vendor=Oracle Corporation
2018-05-17 04:57:59,121 INFO [main] zookeeper.ZooKeeper: Client environment:java.home=/usr/java/jdk1.8.0_151/jre
2018-05-17 04:57:59,121 INFO [main] zookeeper.ZooKeeper: Client environment:java.class.path=/home/acadgild/install/hbase/hbase-1.2
.6/conf:/usr/java/jdk1.8.0_151/lib/tools.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/apalliance-1.0.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/ap
acheds-118n-2.0.0-M15.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/apacheds-kerberos-codec-2.0.0-M15.jar:/home/acadgild/install
/hbase/hbase-1.2.6/lib/api-asn1-api-1.0.0-M20.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/api-util-1.0.0-M20.jar:/home/acadgil
d/install/hbase/hbase-1.2.6/lib/asm-3.1.1.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/avro-1.7.4.jar:/home/acadgild/install/hbas
e/hbase-1.2.6/lib/commons-beanutils-1.7.0.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/commons-httpclient-3.1.jar:/home/acadgild/install/hbase/hb
ase-1.2.6/lib/commons-io-2.4.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/commons-lang-2.6.jar:/home/acadgild/install/hbase/hba
se-1.2.6/lib/commons-logging-1.2.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/commons-math-2.2.jar:/home/acadgild/install/hb
ase-1.2.6/lib/commons-math3-3.1.1.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/commons-net-3.1.jar:/home/acadgild/install/hb
ase/hbase-1.2.6/lib/disruptor-3.3.0.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/findbugs-annotations-1.3.9-1.jar:/home/acadgil
d/install/hbase/hbase-1.2.6/lib/guava-12.0.1.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/guice-3.0.jar:/home/acadgild/install
/hbase/hbase-1.2.6/lib/guice-servlet-3.0.jar:/home/acadgild/install/hbase/hbase-1.2.6/lib/hadoop-annotations-2.5.1.jar:/home/acadgil
```

The left sidebar shows the file explorer for the terminal session, listing files like .abrt, .baseline, .cache, .config, .dbus, .edipse, .fontconfig, .gnome2, .gnote, .gnupg, .gvfs, .local, .m2, .mozilla, .nautilus, .oracle_jre_usage, .p2, .pki, .pulse, .scalade, .sftp, .ssh, .swt, .thumbnails, and .tooling.

The screenshot shows a MobaXterm window with a terminal session. The terminal displays the output of a MapReduce job, indicating it has completed successfully. The output includes various counters and statistics.

```
2018-05-17 04:59:25,385 INFO [main] mapreduce.Job: map 0% reduce 0%
2018-05-17 05:00:00,086 INFO [main] mapreduce.Job: map 100% reduce 0%
2018-05-17 05:00:02,520 INFO [main] mapreduce.Job: Job job_1526512960092_0001 completed successfully
2018-05-17 05:00:03,135 INFO [main] mapreduce.Job: Counters: 31

File System Counters
  FILE: Number of bytes read=0
  FILE: Number of bytes written=139462
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=140
  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)=28127
  Total time spent by all reduces in occupied slots (ms)=0
  Total time spent by all map tasks (ms)=28127
  Total vcore-seconds taken by all map tasks=28127
  Total megabyte-seconds taken by all map tasks=28802048

Map-Reduce Framework
  Map input records=4
  Map output records=4
  Input split bytes=106
  Spilled Records=0
  Failed Shuffles=0
  Merged Map outputs=0
  GC time elapsed (ms)=339
  CPU time spent (ms)=4520
  Physical memory (bytes) snapshot=110243840
  Virtual memory (bytes) snapshot=2067746816
  Total committed heap usage (bytes)=48758784

ImportTsv
  Bad Lines=0
  File Input Format Counters
```

Output

The screenshot shows a MobaXterm window with a terminal session. The terminal displays the output of a scan operation on a bulktable, showing the results of the scan in a table format.

```
hbase(main):002:0> scan 'bulktable'
COLUMN+CELL
ROW
1 column=cf1:name, timestamp=1526513274135, value=Amit
1 column=cf2:exp, timestamp=1526513274135, value=4
2 column=cf1:name, timestamp=1526513274135, value=Girija
2 column=cf2:exp, timestamp=1526513274135, value=3
3 column=cf1:name, timestamp=1526513274135, value=Jatin
3 column=cf2:exp, timestamp=1526513274135, value=5
4 column=cf1:name, timestamp=1526513274135, value=Swati
4 column=cf2:exp, timestamp=1526513274135, value=3
4 row(s) in 1.1560 seconds

hbase(main):003:0>
```