PICT, PUNE

SL-VI Assignment 4

Version-Hadoop-2.7.2

Version-Hive-2.2.0

Version-Hbase-1.2.6

4. Write an application using HBase and HiveQL for flight information system which will include

a) Creating, Dropping, and altering Database tables

b) Creating an external Hive table to connect to the HBase for Customer Information Table

c) Load table with data, insert new values and field in the table, Join tables with Hive

d) Create index on Flight information Table

e) Find the average departure delay per day in 2008.

a) Creating, Dropping, and altering Database tables

Using Hbase

#Create Table:

```
hbase(main):002:0> create 'flight','finfo','fsch'
0 row(s) in 4.6960 seconds
=> Hbase::Table - flight
```

#Table Created-list

```
hbase(main):003:0> list
TABLE
flight
table1
table2
3 row(s) in 0.0120 seconds
```

#Insert records in created table

```
hbase(main):004:0> put 'flight',1,'finfo:source','pune'
0 row(s) in 0.2480 seconds
hbase(main):008:0> put 'flight',1,'finfo:dest','mumbai'
0 row(s) in 0.0110 seconds
hbase(main):010:0> put 'flight',1,'fsch:at','10.25a.m.'
0 row(s) in 0.0060 seconds
```

```
hbase(main):011:0> put 'flight',1,'fsch:dt','11.25 a.m.'
0 row(s) in 0.0070 seconds
hbase(main):012:0> put 'flight',1,'fsch:delay','5min'
hbase(main):015:0> put 'flight',2,'finfo:source','pune'
0 row(s) in 0.0160 seconds
hbase(main):016:0> put 'flight',2,'finfo:dest','kolkata'
0 row(s) in 0.0070 seconds
hbase(main):017:0> put 'flight',2,'fsch:at','7.00a.m.'
0 row(s) in 0.0080 seconds

hbase(main):018:0> put 'flight',2,'fsch:dt','7.30a.m.'
0 row(s) in 0.0050 seconds
hbase(main):019:0> put 'flight',2,'fsch:delay','2 min'
0 row(s) in 0.0090 seconds
hbase(main):021:0> put 'flight',3,'finfo:source','mumbai'
0 row(s) in 0.0040 seconds
hbase(main):022:0> put 'flight',3,'finfo:dest','pune'
0 row(s) in 0.0070 seconds
hbase(main):023:0> put 'flight',3,'fsch:at','12.30p.m.'
0 row(s) in 0.0100 seconds
hbase(main):024:0> put 'flight',3,'fsch:dt','12.45p.m.'
0 row(s) in 0.0040 seconds
hbase(main):025:0> put 'flight',3,'fsch:delay','1 min'
0 row(s) in 0.0190 seconds
hbase(main):026:0> put 'flight',4,'finfo:source','mumbai'
0 row(s) in 0.0060 seconds
hbase(main):027:0> put 'flight',4,'finfo:dest','delhi'
0 row(s) in 0.0050 seconds
hbase(main):028:0> put 'flight',4,'fsch:at','2.00p.m.'
0 row(s) in 0.0080 seconds
hbase(main):029:0> put 'flight',4,'fsch:dt','2.45p.m.'
0 row(s) in 0.0040 seconds
hbase(main):030:0> put 'flight',4,'fsch:delay','10 min'
0 row(s) in 0.0140 seconds
```

#Display Records from Table ‘flight’

```
hbase(main):031:0> scan 'flight'
ROW                COLUMN+CELL
 1                 column=finfo:dest, timestamp=1521312730758, value=mumbai
 1                 column=finfo:source, timestamp=1521312493881, value=pune
 1                 column=fsch:at, timestamp=1521312789417, value=10.25a.m.
 1                 column=fsch:delay, timestamp=1521312850594, value=5min
 1                    column=fsch:dt, timestamp=1521312823256, value=11.25 a.m.
 2                    column=finfo:dest, timestamp=1521313135697, value=kolkata
 2                    column=finfo:source, timestamp=1521313092772, value=pune
 2                    column=fsch:at, timestamp=1521313166540, value=7.00a.m.
 2                    column=fsch:delay, timestamp=1521313229963, value=2 min
 2                    column=fsch:dt, timestamp=1521313202767, value=7.30a.m.
 3                    column=finfo:dest, timestamp=1521313310302, value=pune
 3                    column=finfo:source, timestamp=1521313290906, value=mumbai
 3                    column=fsch:at, timestamp=1521313333432, value=12.30p.m.
 3                    column=fsch:delay, timestamp=1521313379725, value=1 min
 3                    column=fsch:dt, timestamp=1521313353804, value=12.45p.m.
 4                    column=finfo:dest, timestamp=1521313419679, value=delhi
 4                    column=finfo:source, timestamp=1521313404831, value=mumbai
 4                    column=fsch:at, timestamp=1521313440328, value=2.00p.m.
 4                    column=fsch:delay, timestamp=1521313472389, value=10 min
 4                    column=fsch:dt, timestamp=1521313455226, value=2.45p.m.
4 row(s) in 0.0300 seconds
```

#Alter Table (add one more column family)

```
hbase(main):036:0> alter 'flight',NAME=>'revenue'
Updating all regions with the new schema...
0/1 regions updated.
1/1 regions updated.
Done.
0 row(s) in 3.7640 seconds

hbase(main):037:0> scan 'flight'
ROW                   COLUMN+CELL
 1                    column=finfo:dest, timestamp=1521312730758, value=mumbai
 1                    column=finfo:source, timestamp=1521312493881, value=pune
 1                    column=fsch:at, timestamp=1521312789417, value=10.25a.m.
 1                    column=fsch:delay, timestamp=1521312850594, value=5min
 1                    column=fsch:dt, timestamp=1521312823256, value=11.25 a.m.
 2                    column=finfo:dest, timestamp=1521313135697, value=kolkata
 2                    column=finfo:source, timestamp=1521313092772, value=pune
 2                    column=fsch:at, timestamp=1521313166540, value=7.00a.m.
 2                    column=fsch:delay, timestamp=1521313229963, value=2 min
 2                    column=fsch:dt, timestamp=1521313202767, value=7.30a.m.
 3                    column=finfo:dest, timestamp=1521313310302, value=pune
 3                    column=finfo:source, timestamp=1521313290906, value=mumbai
 3                    column=fsch:at, timestamp=1521313333432, value=12.30p.m.
 3                    column=fsch:delay, timestamp=1521313379725, value=1 min
 3                    column=fsch:dt, timestamp=1521313353804, value=12.45p.m.
 4                    column=finfo:dest, timestamp=1521313419679, value=delhi
 4                    column=finfo:source, timestamp=1521313404831, value=mumbai
 4                    column=fsch:at, timestamp=1521313440328, value=2.00p.m.
 4                    column=fsch:delay, timestamp=1521313472389, value=10 min
 4                column=fsch:dt, timestamp=1521313455226, value=2.45p.m.
4 row(s) in 0.0290 seconds
```

#Insert records into added column family

```
hbase(main):038:0> put 'flight',4,'revenue:rs','45000'
0 row(s) in 0.0100 seconds
```

#Check the updates

```
hbase(main):039:0> scan 'flight'
ROW                COLUMN+CELL
 1                column=finfo:dest, timestamp=1521312730758, value=mumbai
 1                column=finfo:source, timestamp=1521312493881, value=pune
 1                column=fsch:at, timestamp=1521312789417, value=10.25a.m.
 1                column=fsch:delay, timestamp=1521312850594, value=5min
 1                column=fsch:dt, timestamp=1521312823256, value=11.25 a.m.
 2                column=finfo:dest, timestamp=1521313135697, value=kolkata
 2                column=finfo:source, timestamp=1521313092772, value=pune
 2                column=fsch:at, timestamp=1521313166540, value=7.00a.m.
 2                column=fsch:delay, timestamp=1521313229963, value=2 min
 2                column=fsch:dt, timestamp=1521313202767, value=7.30a.m.
 3                column=finfo:dest, timestamp=1521313310302, value=pune
 3                column=finfo:source, timestamp=1521313290906, value=mumbai
 3                column=fsch:at, timestamp=1521313333432, value=12.30p.m.
 3                column=fsch:delay, timestamp=1521313379725, value=1 min
 3                column=fsch:dt, timestamp=1521313353804, value=12.45p.m.
 4                column=finfo:dest, timestamp=1521313419679, value=delhi
 4                column=finfo:source, timestamp=1521313404831, value=mumbai
 4                column=fsch:at, timestamp=1521313440328, value=2.00p.m.
 4                column=fsch:delay, timestamp=1521313472389, value=10 min
 4                column=fsch:dt, timestamp=1521313455226, value=2.45p.m.
 4                column=revenue:rs, timestamp=1521314406914, value=45000
4 row(s) in 0.0340 seconds
```

#Delete Column family

```
hbase(main):040:0> alter 'flight',NAME=>'revenue',METHOD=>'delete'
Updating all regions with the new schema...
0/1 regions updated.
1/1 regions updated.
Done.
0 row(s) in 3.7880 seconds
```

#changes Reflected in Table

```
hbase(main):041:0> scan 'flight'
ROW                COLUMN+CELL
 1                column=finfo:dest, timestamp=1521312730758, value=mumbai
 1                column=finfo:source, timestamp=1521312493881, value=pune
 1                column=fsch:at, timestamp=1521312789417, value=10.25a.m.
 1                column=fsch:delay, timestamp=1521312850594, value=5min
 1                column=fsch:dt, timestamp=1521312823256, value=11.25 a.m.
 2                column=finfo:dest, timestamp=1521313135697, value=kolkata
 2                  column=finfo:source, timestamp=1521313092772, value=pune
 2                  column=fsch:at, timestamp=1521313166540, value=7.00a.m.
 2                  column=fsch:delay, timestamp=1521313229963, value=2 min
 2                  column=fsch:dt, timestamp=1521313202767, value=7.30a.m.
 3                  column=finfo:dest, timestamp=1521313310302, value=pune
 3                  column=finfo:source, timestamp=1521313290906, value=mumbai
 3                  column=fsch:at, timestamp=1521313333432, value=12.30p.m.
 3                  column=fsch:delay, timestamp=1521313379725, value=1 min
 3                  column=fsch:dt, timestamp=1521313353804, value=12.45p.m.
 4                  column=finfo:dest, timestamp=1521313419679, value=delhi
 4                  column=finfo:source, timestamp=1521313404831, value=mumbai
 4                  column=fsch:at, timestamp=1521313440328, value=2.00p.m.
 4                  column=fsch:delay, timestamp=1521313472389, value=10 min
 4                  column=fsch:dt, timestamp=1521313455226, value=2.45p.m.
4 row(s) in 0.0280 seconds
```

#Drop Table

#Create Table for dropping

```
hbase(main):046:0* create 'tb1','cf'
0 row(s) in 2.3120 seconds
=> Hbase::Table - tb1

hbase(main):047:0> list
TABLE
flight
table1
table2
tb1
4 row(s) in 0.0070 seconds
=> ["flight", "table1", "table2", "tb1"]
```

#Drop Table

```
hbase(main):048:0> drop 'tb1'

ERROR: Table tb1 is enabled. Disable it first.

Here is some help for this command:
Drop the named table. Table must first be disabled:
  hbase> drop 't1'
  hbase> drop 'ns1:t1'
```

#Disable table

```
hbase(main):049:0> disable 'tb1'
0 row(s) in 4.3480 seconds

hbase(main):050:0> drop 'tb1'
0 row(s) in 2.3540 seconds

hbase(main):051:0> list
TABLE
```

```
flight
table1
table2
3 row(s) in 0.0170 seconds

=> ["flight", "table1", "table2"]
```

#Read data from table for row key 1:

```
hbase(main):052:0> get 'flight',1
COLUMN                CELL
 finfo:dest          timestamp=1521312730758, value=mumbai
 finfo:source         timestamp=1521312493881, value=pune
 fsch:at             timestamp=1521312789417, value=10.25a.m.
 fsch:delay           timestamp=1521312850594, value=5min
 fsch:dt             timestamp=1521312823256, value=11.25 a.m.
5 row(s) in 0.0450 seconds
```

Read data for particular column from HBase table:

```
hbase(main):053:0> get 'flight','1',COLUMN=>'finfo:source'
COLUMN                CELL
 finfo:source         timestamp=1521312493881, value=pune
1 row(s) in 0.0110 seconds
```

Read data for multiple columns in HBase Table:

```
hbase(main):054:0> get 'flight','1',COLUMN=>['finfo:source','finfo:dest']
COLUMN                CELL
 finfo:dest          timestamp=1521312730758, value=mumbai
 finfo:source         timestamp=1521312493881, value=pune
2 row(s) in 0.0190 seconds

hbase(main):055:0> scan 'flight',COLUMNS=>'finfo:source'
ROW                   COLUMN+CELL
 1                   column=finfo:source, timestamp=1521312493881, value=pune
 2                   column=finfo:source, timestamp=1521313092772, value=pune
 3                   column=finfo:source, timestamp=1521313290906, value=mumbai
 4                   column=finfo:source, timestamp=1521313404831, value=mumbai
4 row(s) in 0.0320 seconds
```

b) Creating an external Hive table to connect to the HBase for Customer Information Table

Covers===>

c) Load table with data, insert new values and field in the table, Join tables with Hive

Add these Jar files in Hive(on hive prompt)

```
add jar file:///usr/local/HBase/lib/zookeeper-3.4.6.jar;
add jar file:///usr/local/HBase/lib/guava-12.0.1.jar;
add jar file:///usr/local/HBase/lib/hbase-client-1.2.6.jar;
add jar file:///usr/local/HBase/lib/hbase-common-1.2.6.jar;
add jar file:///usr/local/HBase/lib/hbase-protocol-1.2.6.jar;
add jar file:///usr/local/HBase/lib/hbase-server-1.2.6.jar;
add jar file:///usr/local/HBase/lib/hbase-shell-1.2.6.jar;
add jar file:///usr/local/HBase/lib/hbase-thrift-1.2.6.jar;
add jar file:///usr/local/hive/lib/hive-hbase-handler-2.2.0.jar;
```

Set the values of variables in Hive

```
set hbase.zookeeprt.quorum=localhost;
set hive.metastore.client.setugi=true;
set hive.exec.stagingdir=/tmp/.hivestage;
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.auto.convert.join=false;
set hive.hbase.wal.enabled=false;
SET hive.exec.dynamic.partition = true;
SET hive.exec.dynamic.partition.mode = nonstrict;
SET hive.exec.max.dynamic.partitions = 10000;
SET hive.exec.max.dynamic.partitions.pernode = 1000;
```

# Create the external table emp using hive

```
hive>create external table empdata2 ( ename string, esal int)
row format delimited fields terminated by "," stored as textfile location
 "/home/hduser/Desktop/empdata2";

hive>load data local inpath '/home/hduser/Desktop/empdb.txt' into table empdata2;
```

#Create External Table in hive referring to hbase table

# create hbase table emphive first

```
hbase(main):003:0>  create 'emphive', 'cf'
0 row(s) in 4.6260 seconds
```

#create hive external table

```
CREATE external TABLE hive_table_emp(id int, name string, esal string)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,cf:name,cf:esal")
TBLPROPERTIES ("hbase.table.name" = "emphive");
```

# load data into hive_table_emp

(Hive doesn't allow directly inserting data into external hive table)

#for that create one hive table(managed table in hive)

Managed table and External **table in Hive**. There are two types of **tables in Hive** ,one is **Managed table** and second is external **table**. the difference is , when you drop a **table**, if it is **managed table hive** deletes both data and meta data,if it is external **table Hive** only deletes metadata.

```
hive>create table empdbnew(eno int, ename string, esal int) row format delimited fields terminated by ',' stored as textfile;
```

#load data in managed table

```
hive>load data local inpath '/home/hduser/Desktop/empdbnew.txt' into table empdbnew;
```

#Load data in external table from managed table.

```
hive>INSERT INTO hive_table_emp select * from empdbnew;
```

```
hive> select * from hive_table_emp;
OK
1	deepali	120000
2	mahesh		30000
3	mangesh		25000
4	ram	39000
5	brijesh	40000
6	john	300000
Time taken: 0.52 seconds, Fetched: 6 row(s)
```

#display records where salary is greater than 40000

```
hive> select * from hive_table_emp where esal>40000;
OK
1	deepali	120000
6	john	300000
Time taken: 0.546 seconds, Fetched: 2 row(s)
```

#Check hbase for updates(The records are available in associated Hbase table)

```
hbase(main):008:0> scan 'emphive'
ROW                  COLUMN+CELL
 1                column=cf:esal, timestamp=1522212425665, value=120000
 1                column=cf:name, timestamp=1522212425665, value=deepali
 2                column=cf:esal, timestamp=1522212425665, value=30000
 2                column=cf:name, timestamp=1522212425665, value=mahesh
 3                column=cf:esal, timestamp=1522212425665, value=25000
 3                column=cf:name, timestamp=1522212425665, value=mangesh
 4                column=cf:esal, timestamp=1522212425665, value=39000
 4                column=cf:name, timestamp=1522212425665, value=ram
 5                column=cf:esal, timestamp=1522212425665, value=40000
 5                column=cf:name, timestamp=1522212425665, value=brijesh
 6                column=cf:esal, timestamp=1522212425665, value=300000
 6                column=cf:name, timestamp=1522212425665, value=john
```

```
6 row(s) in 0.0700 seconds
```

# Creating external table in Hive referring to Hbase
#referring to flight table created in Hbase

```
CREATE external TABLE hbase_flight_new(fno int, fsource string,fdest string,fsh_at string,fsh_dt string,fsch_delay
string,delay int)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping" =
":key,finfo:source,finfo:dest,fsch:at,fsch:dt,fsch:delay,delay:dl")
TBLPROPERTIES ("hbase.table.name" = "flight");

hive> CREATE external TABLE hbase_flight_new(fno int, fsource string,fdest string,fsh_at string,fsh_dt
string,fsch_delay string,delay int)
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES ("hbase.columns.mapping"
=":key,finfo:source,finfo:dest,fsch:at,fsch:dt,fsch:delay,delay:dl")
    > TBLPROPERTIES ("hbase.table.name" = "flight");
OK
Time taken: 0.361 seconds
```

#table created in hive

```
hive> show tables;
OK
abc
ddl_hive
emp
empdata
empdata1
empdata2
empdbnew
hbase_flight
hbase_flight1
hbase_flight_new
hbase_table_1
hive_table_emp
Time taken: 0.036 seconds, Fetched: 12 row(s)
```

# Display records from that table

```
hive> select * from hbase_flight_new;
OK
1       pune    mumbai          10.25a.m.       11.25 a.m.      5min    10
2       pune    kolkata7.00a.m.         7.30a.m.        2 min   4
3       mumbai          pune    12.30p.m.       12.45p.m.       1 min   5
4       mumbai          delhi   2.00p.m.        2.45p.m.        10 min 16
Time taken: 0.581 seconds, Fetched: 4 row(s)
```

e) Find the average departure delay per day in 2008.

#calculate average delay

```
hive> select sum(delay) from hbase_flight_new;
```

```
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 Hive 1.X releases.
Query ID = hduser_20180328130004_47384e9a-7490-4dfb-809d-ae240507bfab
Total jobs = 1
Launching Job 1 out of 1
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_1522208646737_0003, Tracking URL =
http://localhost:8088/proxy/application_1522208646737_0003/
Kill Command = /usr/local/hadoop/bin/hadoop job  -kill job_1522208646737_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2018-03-28 13:00:20,256 Stage-1 map = 0%,  reduce = 0%
2018-03-28 13:00:28,747 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.68 sec
2018-03-28 13:00:35,101 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 6.26 sec
MapReduce Total cumulative CPU time: 6 seconds 260 msec
Ended Job = job_1522208646737_0003
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 6.26 sec   HDFS Read: 9095 HDFS Write:
102 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 260 msec
OK
35
Time taken: 31.866 seconds, Fetched: 1 row(s)
hive>
```

d) Create index on Flight information Table

#create index on hbase_flight_new

```
CREATE INDEX hbasefltnew_index
ON TABLE hbase_flight_new (delay)
AS 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler'
WITH DEFERRED REBUILD;

SHOW INDEX ON hbase_flight_new;
```

#create index on table hbase_flight_new

```
hive> CREATE INDEX hbasefltnew_index
    > ON TABLE hbase_flight_new (delay)
    > AS 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler'
    > WITH DEFERRED REBUILD;
OK
Time taken: 0.74 seconds
```

#show index on table hbase_flight_new

```
hive> SHOW INDEX ON hbase_flight_new;
OK
hbasefltnew_index	hbase_flight_new	delay
		default__hbase_flight_new_hbasefltnew_index__	compact
Time taken: 0.104 seconds, Fetched: 1 row(s)
```

#join two tables in Hive

#create table B for join

```
hive> create table empinfo(empno int, empgrade string) row format delimited fields terminated by
',' stored as textfile;
```

#Load Data into table

```
hive> load data local inpath '/home/hduser/Desktop/empinfo.txt' into table empinfo;
Loading data to table default.empinfo
OK
Time taken: 0.552 seconds
```

#insert data into the table

```
hive> load data local inpath '/home/hduser/Desktop/empinfo.txt' into table empinfo;
```

# Table A empdbnew

```
hive> select * from empdbnew;
OK
1	deepali	120000
2	mahesh		30000
3	mangesh		25000
4	ram	39000
5	brijesh	40000
6	john	300000
Time taken: 0.258 seconds, Fetched: 6 row(s)
```

# Table B empinfo

```
hive> select * from empinfo;
OK
1	A
2	B
3	B
4	B
5	B
6	A
Time taken: 0.207 seconds, Fetched: 6 row(s)
```

#Join two tables(empdbnew with empinfo on empno)

```
hive> SELECT eno, ename, empno, empgrade FROM empdbnew JOIN empinfo ON eno = empno;
```

#Join==> Result

```
hive> SELECT eno, ename, empno, empgrade
    > FROM empdbnew JOIN empinfo ON eno = empno;
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 Hive 1.X releases.
Query ID = hduser_20180328153258_bc345f46-a1f1-4589-ac5e-4c463834731a
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_1522208646737_0005, Tracking URL =
http://localhost:8088/proxy/application_1522208646737_0005/
Kill Command = /usr/local/hadoop/bin/hadoop job  -kill job_1522208646737_0005
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2018-03-28 15:33:09,615 Stage-1 map = 0%,  reduce = 0%
2018-03-28 15:33:18,231 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 8.17 sec
2018-03-28 15:33:24,476 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 10.61 sec
MapReduce Total cumulative CPU time: 10 seconds 610 msec
Ended Job = job_1522208646737_0005
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2  Reduce: 1   Cumulative CPU: 10.61 sec   HDFS Read: 15336 HDFS Write:
235 SUCCESS
Total MapReduce CPU Time Spent: 10 seconds 610 msec
OK
1	deepali	1	A
2	mahesh	2	B
3	mangesh	3	B
4	ram	4	B
5	brijesh	5	B
6	john	6	A
Time taken: 26.915 seconds, Fetched: 6 row(s)
```