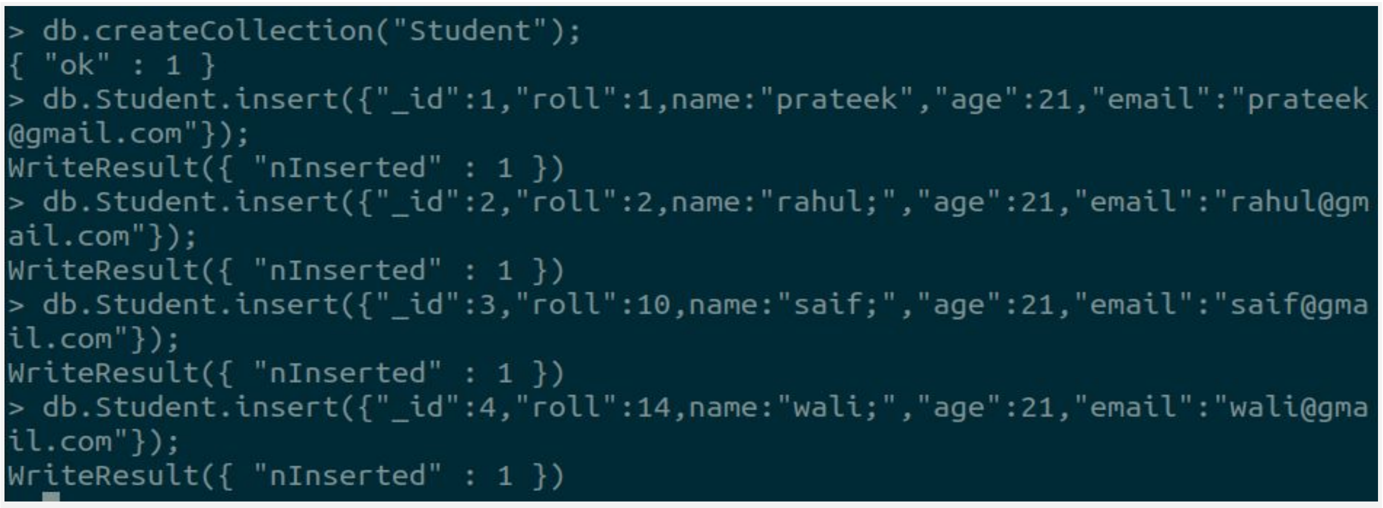
Question 1: Do the following MongoDB operations.

1. **Create a database “Student” with the following attributes Rollno, Age, ContactNo, Email-Id.**

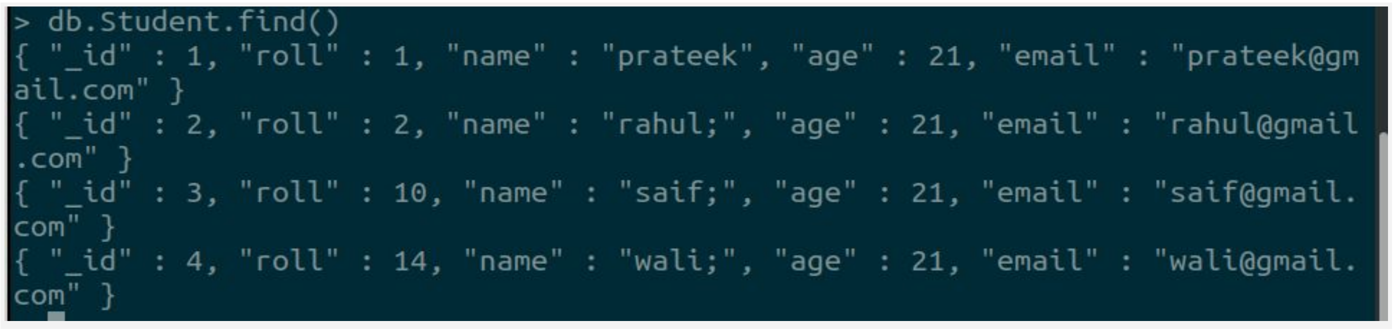
   use collegedb;
   db.createCollection("Student");

2. **Insert appropriate values**

   db.insert({_id:1,Rollno:1,Age:22,Name:"Prateek",ContactNo:991231233,EmailId : "prateek@gmail.com"});
   db.insert({_id:2,Rollno:10,Age:22,Name:"Mayank",ContactNo: 991231234, EmailId:"mayank@gmail.com"});
   db.insert({_id:3,Rollno:11,Age:22,Name:"ABC",ContactNo: 991231235, EmailId:"abc@gmail.com"});
   db.Sprateek@gmail.comtudent.find().pretty();



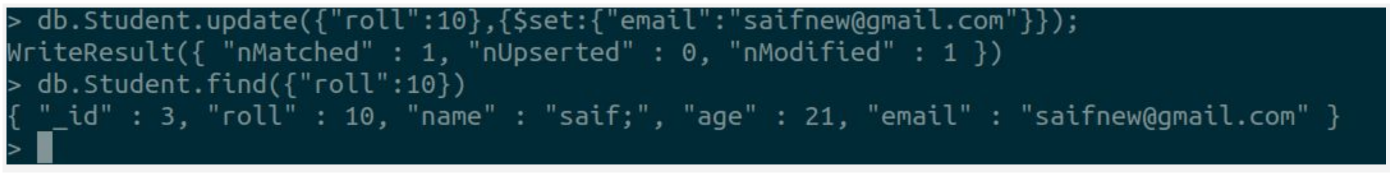
```
> db.createCollection("Student");
{ "ok" : 1 }
> db.Student.insert({"_id":1,"roll":1,name:"prateek","age":21,"email":"prateek
@gmail.com"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({"_id":2,"roll":2,name:"rahul;","age":21,"email":"rahul@gm
ail.com"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({"_id":3,"roll":10,name:"saif;","age":21,"email":"saif@gma
il.com"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({"_id":4,"roll":14,name:"wali;","age":21,"email":"wali@gma
il.com"});
WriteResult({ "nInserted" : 1 })
```



```
> db.Student.find()
{ "_id" : 1, "roll" : 1, "name" : "prateek", "age" : 21, "email" : "prateek@gm
ail.com" }
{ "_id" : 2, "roll" : 2, "name" : "rahul;", "age" : 21, "email" : "rahul@gmail
.com" }
{ "_id" : 3, "roll" : 10, "name" : "saif;", "age" : 21, "email" : "saif@gmail.
com" }
{ "_id" : 4, "roll" : 14, "name" : "wali;", "age" : 21, "email" : "wali@gmail.
com" }
```

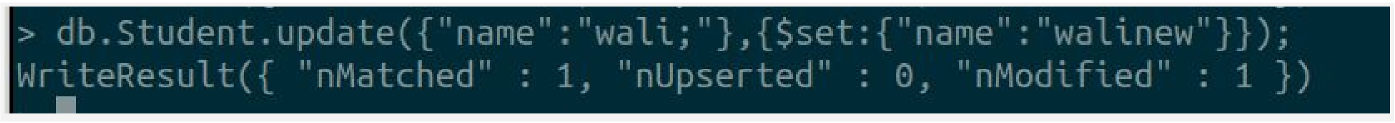
3. **Write a query to update the Email-Id of a student with roll 10.**

db.Student.update({Rollno:10},{$set:{EmailId:"mayankwali@gmail.com"}});



```
> db.Student.update({"roll":10},{$set:{"email":"saifnew@gmail.com"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Student.find({"roll":10})
{ "_id" : 3, "roll" : 10, "name" : "saif;", "age" : 21, "email" : "saifnew@gmail.com" }
>
```

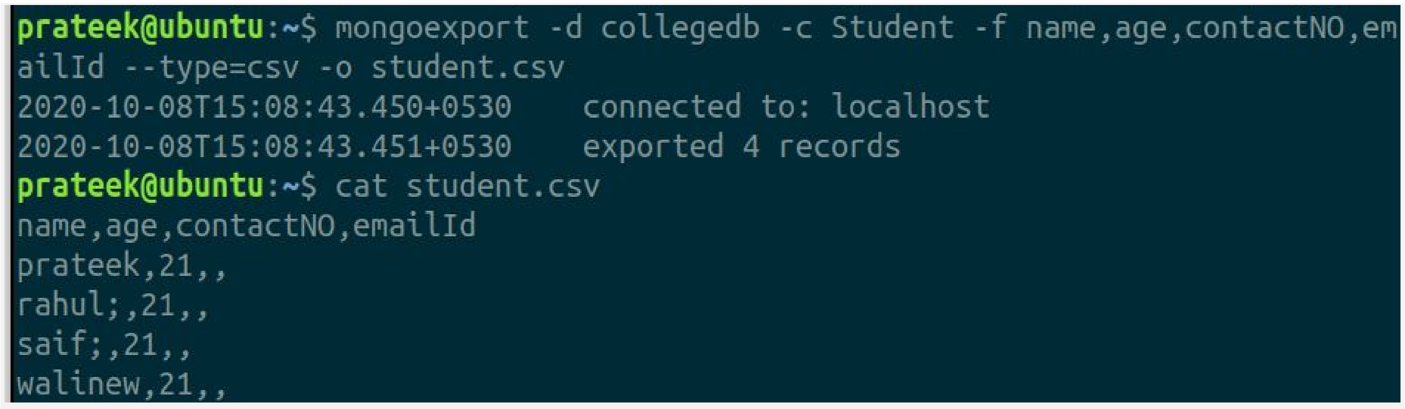
4. **Replace the student name from "ABC" to "FEM" of rollno 1.**
   db.Student.update({Rollno:11},{$set:{Name:"FEM"}})



```
> db.Student.update({"name":"wali;"},{$set:{"name":"walinew"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

5. **Export the created table into the local file system.**

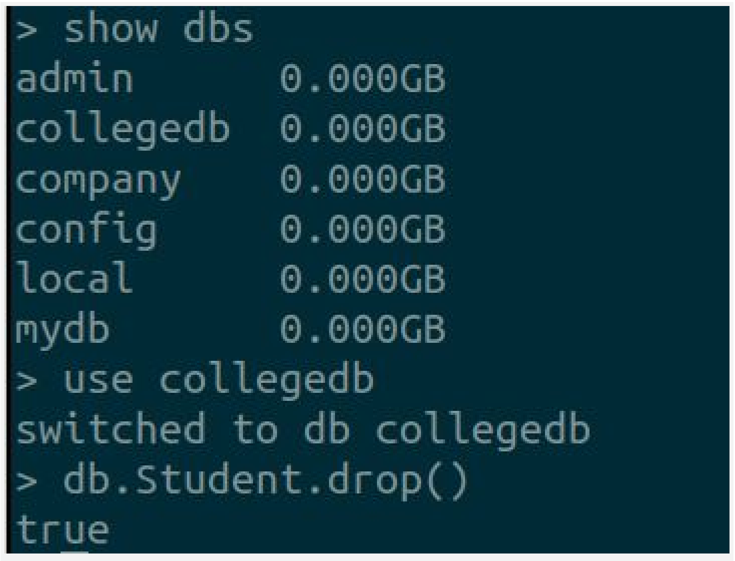
mongoexport -d collegedb -c Student -f Rollno, Age, Name, EmailId --type=csv -o student.csv



```
prateek@ubuntu:~$ mongoexport -d collegedb -c Student -f name,age,contactNO,emailId --type=csv -o student.csv
2020-10-08T15:08:43.450+0530    connected to: localhost
2020-10-08T15:08:43.451+0530    exported 4 records
prateek@ubuntu:~$ cat student.csv
name,age,contactNO,emailId
prateek,21,,
rahul;,21,,
saif;,21,,
walinew,21,,
```

6. **Drop the table**

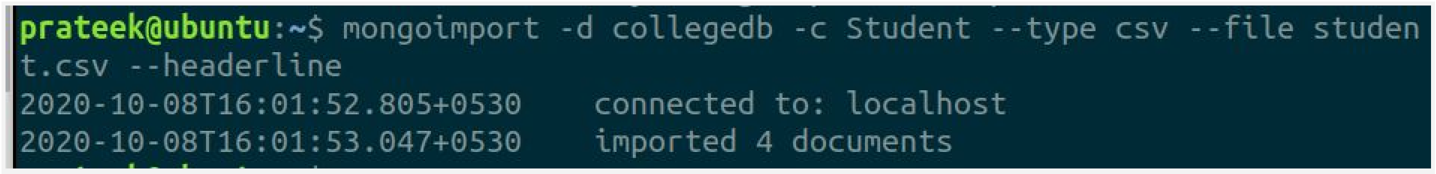
mongoexport -d collegedb -c Student -f Rollno, Age, Name, EmailId --type=csv -o student.csv



```
> show dbs
admin      0.000GB
collegedb  0.000GB
company    0.000GB
config     0.000GB
local      0.000GB
mydb       0.000GB
> use collegedb
switched to db collegedb
> db.Student.drop()
true
```

7. **Import a given CSV dataset from the local file system into MongoDB collection.**

mongoimport -d collegedb -c Student --type csv --file student.csv --headerline



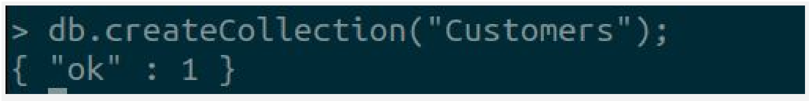
```
prateek@ubuntu:~$ mongoimport -d collegedb -c Student --type csv --file studen
t.csv --headerline
2020-10-08T16:01:52.805+0530    connected to: localhost
2020-10-08T16:01:53.047+0530    imported 4 documents
```

2. Perform the following DB operations using MongoDB.

1. **Create a collection by name Customers with the following attributes.**

Cust_id, Acc_Bal, Acc_Type

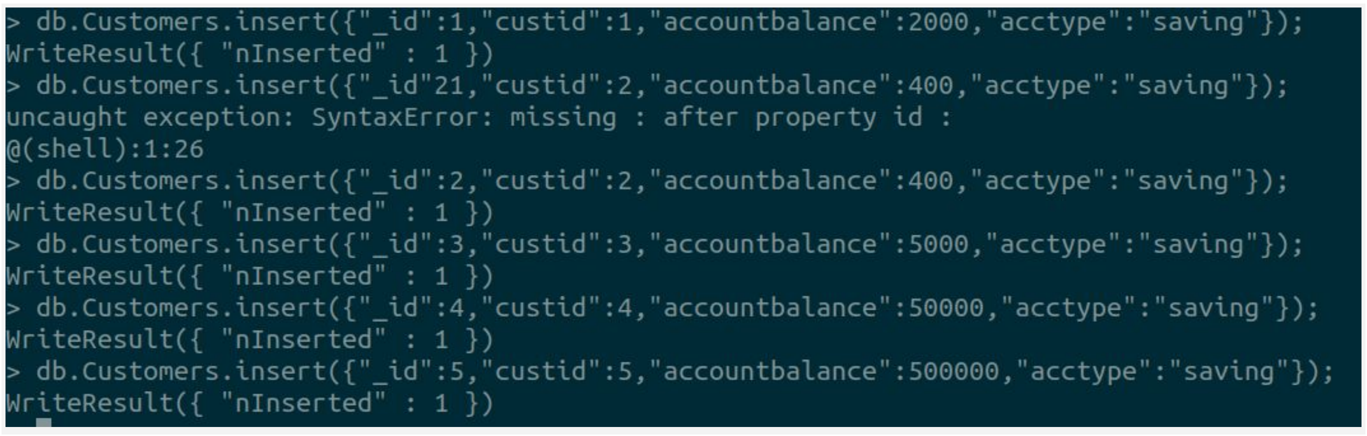
db.createCollection("Customers");



```
> db.createCollection("Customers");
{ "ok" : 1 }
```

2. **Insert at least 5 values into the table**

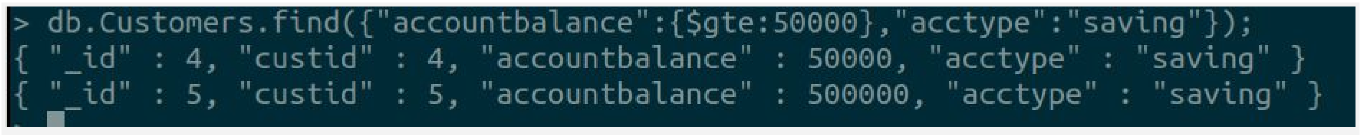
```
db.Customers.insert({_id:1,
 custid:1,
 accountbalance:2000,
 accounttype:"closing"
});
db.Customers.insert({_id:2,
 custid:2,
 accountbalance:3000,
 accounttype:"saving"
});
db.Customers.insert({_id:3,
 custid:3,
 accountbalance:4000,
 accounttype:"closing"
});
db.Customers.insert({_id:4,
 custid:4,
 accountbalance:5000,
 accounttype:"saving"
});
db.Customers.insert({_id:5,
 custid:5,
 accountbalance:6000,
 accounttype:"closing"
});
```



```
> db.Customers.insert({"_id":1,"custid":1,"accountbalance":2000,"acctype":"saving"});
WriteResult({ "nInserted" : 1 })
> db.Customers.insert({"_id"21,"custid":2,"accountbalance":400,"acctype":"saving"});
uncaught exception: SyntaxError: missing : after property id :
@(shell):1:26
> db.Customers.insert({"_id":2,"custid":2,"accountbalance":400,"acctype":"saving"});
WriteResult({ "nInserted" : 1 })
> db.Customers.insert({"_id":3,"custid":3,"accountbalance":5000,"acctype":"saving"});
WriteResult({ "nInserted" : 1 })
> db.Customers.insert({"_id":4,"custid":4,"accountbalance":50000,"acctype":"saving"});
WriteResult({ "nInserted" : 1 })
> db.Customers.insert({"_id":5,"custid":5,"accountbalance":500000,"acctype":"saving"});
WriteResult({ "nInserted" : 1 })
```

3. **Write a query to display those records whose total account balance is greater than 1200 of account type 'Z' for each customer_id.**

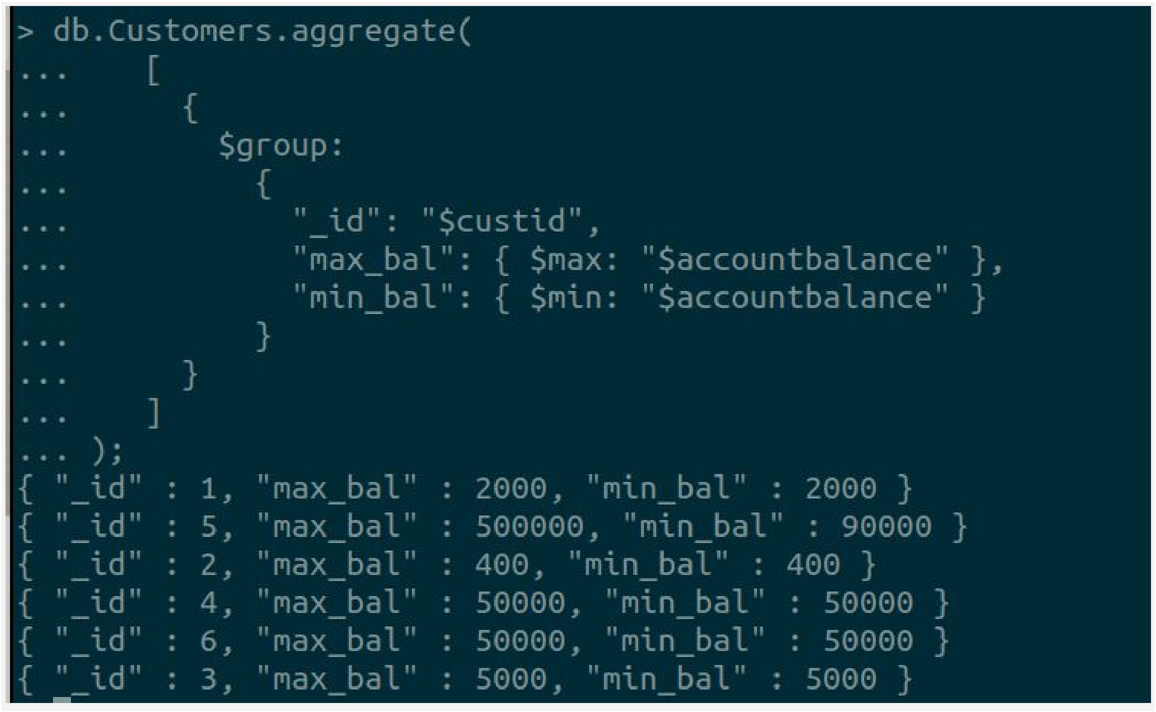
```
db.Customers.find({
  accountbalance:{$gte:1200},
  accounttype:"saving"
});
```



```
> db.Customers.find({"accountbalance":{$gte:50000},"acctype":"saving"});
{ "_id" : 4, "custid" : 4, "accountbalance" : 50000, "acctype" : "saving" }
{ "_id" : 5, "custid" : 5, "accountbalance" : 500000, "acctype" : "saving" }
```

4. **Determine Minimum and Maximum account balance for each customer_id.**

```
db.Customers.aggregate(
 [
   {
     $group:
     {
       _id:$custid,
        max_bal:{$max:$accountbalance},
        min_bal:{$min:$accountbalance}
     }
   }
 ]
```



```
> db.Customers.aggregate(
...     [
...       {
...         $group:
...           {
...             "_id": "$custid",
...             "max_bal": { $max: "$accountbalance" },
...             "min_bal": { $min: "$accountbalance" }
...           }
...       }
...     ]
... );
{ "_id" : 1, "max_bal" : 2000, "min_bal" : 2000 }
{ "_id" : 5, "max_bal" : 500000, "min_bal" : 90000 }
{ "_id" : 2, "max_bal" : 400, "min_bal" : 400 }
{ "_id" : 4, "max_bal" : 50000, "min_bal" : 50000 }
{ "_id" : 6, "max_bal" : 50000, "min_bal" : 50000 }
{ "_id" : 3, "max_bal" : 5000, "min_bal" : 5000 }
```

5. Export created collection into the local file system

mongoexport -d labtestdb -c Customers -f custid, accountbalance, accounttype --type=csv -o Customers.csv.

6. Drop the table

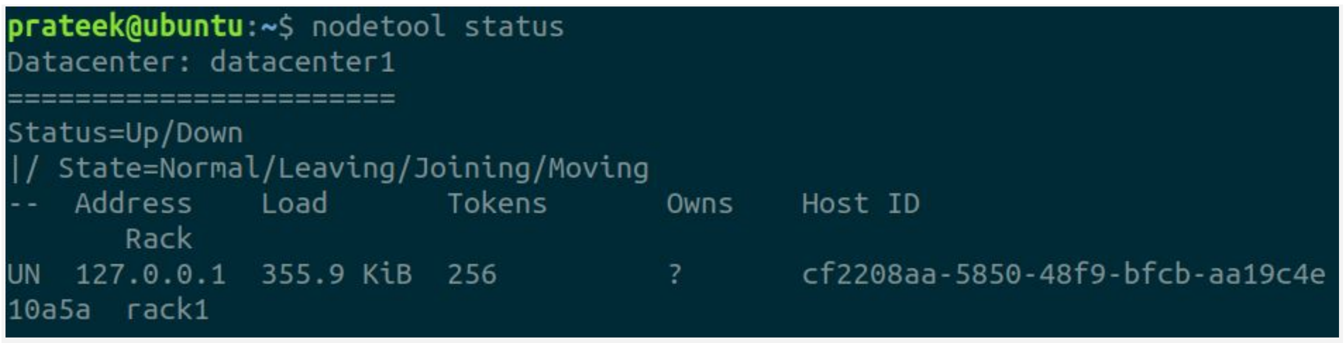
db.Customers.drop();

7. Import a given CSV dataset from the local file system into MongoDB collection.

mongoimport -d labtestdb -c Customers --type csv --file Customers.csv --headerline

Cassandra Programs-To Do

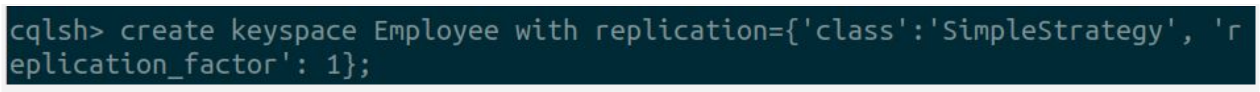
Question 3. Perform the following DB operations using Cassandra.



```
prateek@ubuntu:~$ nodetool status
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address    Load       Tokens       Owns    Host ID                               Rack
UN  127.0.0.1  355.9 KiB  256          ?       cf2208aa-5850-48f9-bfcb-aa19c4e10a5a  rack1
```

1. Create a keyspace by name Employee

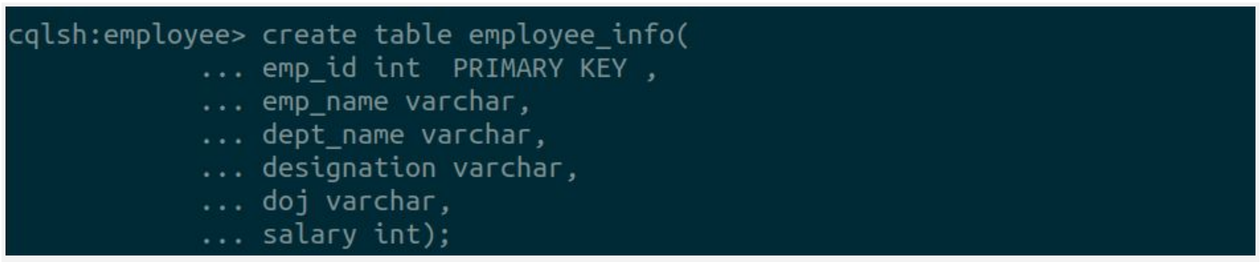
CREATE KEYSPACE Employee WITH replication = {'class':'SimpleStrategy','replication_factor':3};



```
cqlsh> create keyspace Employee with replication={'class':'SimpleStrategy', 'replication_factor': 1};
```

2. Create a column family by name Employee-Info with attributes Emp_Id Primary Key, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name

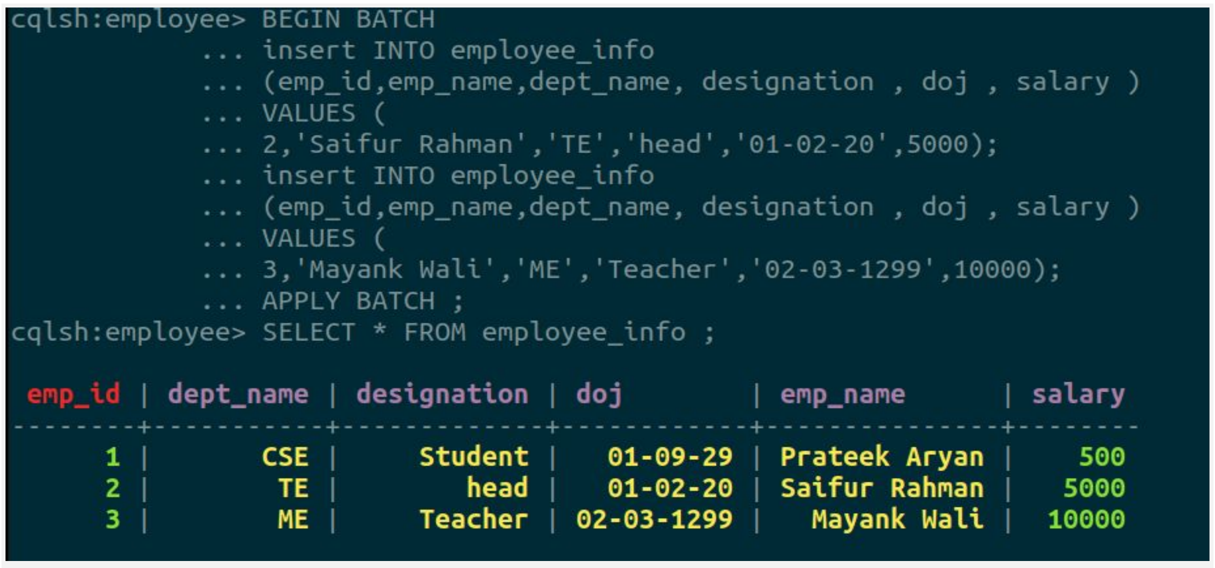
CREATE COLUMNFAMILY employee_info(emp_id INT PRIMARY KEY,emp_name VARCHAR,desgination VARCHAR,doj VARCHAR,dept_name VARCHAR,salary INT);



```
cqlsh:employee> create table employee_info(
            ... emp_id int  PRIMARY KEY ,
            ... emp_name varchar,
            ... dept_name varchar,
            ... designation varchar,
            ... doj varchar,
            ... salary int);
```

3. Insert the values into the table in batch

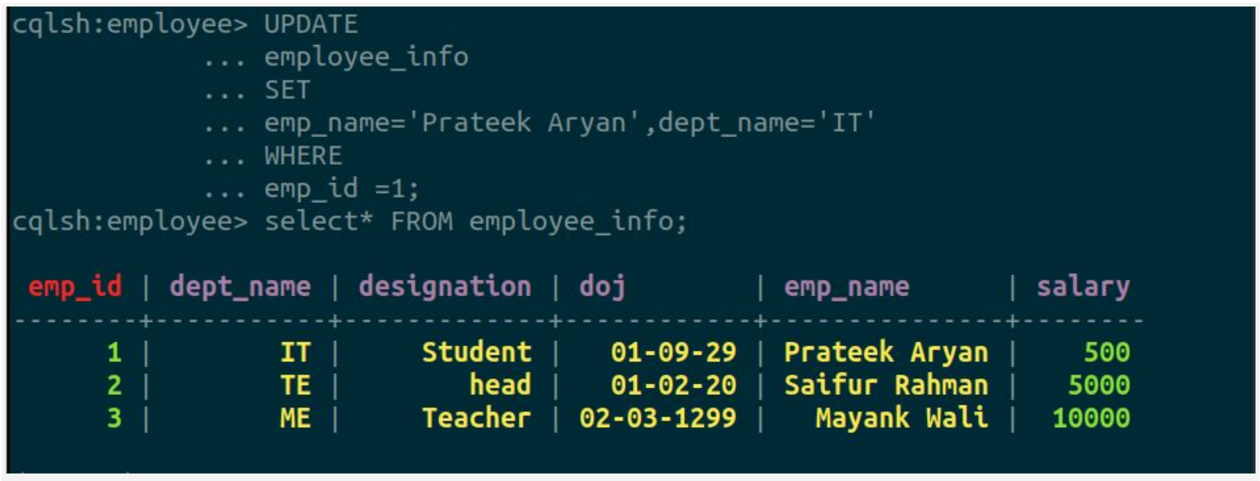
```
BEGIN BATCH
	INSERT INTO employee_info(emp_id,dept_name,desgination,doj,emp_name,salary)values(120,'Development','CTO','10/11/2015','Prateek Aryan',2000000);
	INSERT INTO employee_info(emp_id,dept_name,desgination,doj,emp_name,salary)values(121,'HR','Employee','20/01/2011','Mayank Wali',1500000);
	INSERT INTO employee_info(emp_id,dept_name,desgination,doj,emp_name,salary)values(122,'Maintainance','staff','10/07/2020','Saifur Rahman',50000);
	INSERT INTO employee_info(emp_id,dept_name,desgination,doj,emp_name,salary)values(123,'IT','Assistant','25/07/2014','Parth Chatta',100000);
APPLY BATCH;
select * from employee_info;
```



```
cqlsh:employee> BEGIN BATCH
            ... insert INTO employee_info
            ... (emp_id,emp_name,dept_name, designation , doj , salary )
            ... VALUES (
            ... 2,'Saifur Rahman','TE','head','01-02-20',5000);
            ... insert INTO employee_info
            ... (emp_id,emp_name,dept_name, designation , doj , salary )
            ... VALUES (
            ... 3,'Mayank Wali','ME','Teacher','02-03-1299',10000);
            ... APPLY BATCH ;
cqlsh:employee> SELECT * FROM employee_info ;

 emp_id | dept_name | designation | doj        | emp_name      | salary
--------+-----------+-------------+------------+---------------+--------
      1 |       CSE |     Student |   01-09-29 | Prateek Aryan |    500
      2 |        TE |        head |   01-02-20 | Saifur Rahman |   5000
      3 |        ME |     Teacher | 02-03-1299 |   Mayank Wali |  10000
```

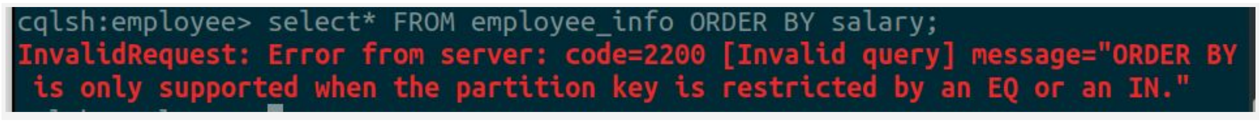
3. Update Employee name and Department of Emp-Id 121

UPDATE employee_info SET emp_name = 'Aakash',dept_name = 'IT' WHERE emp_id = 121;
select * from employee_info;



```
cqlsh:employee> UPDATE
            ... employee_info
            ... SET
            ... emp_name='Prateek Aryan',dept_name='IT'
            ... WHERE
            ... emp_id =1;
cqlsh:employee> select* FROM employee_info;

 emp_id | dept_name | designation | doj        | emp_name      | salary
--------+-----------+-------------+------------+---------------+--------
      1 |        IT |     Student |   01-09-29 | Prateek Aryan |    500
      2 |        TE |        head |   01-02-20 | Saifur Rahman |   5000
      3 |        ME |     Teacher | 02-03-1299 |   Mayank Wali |  10000
```

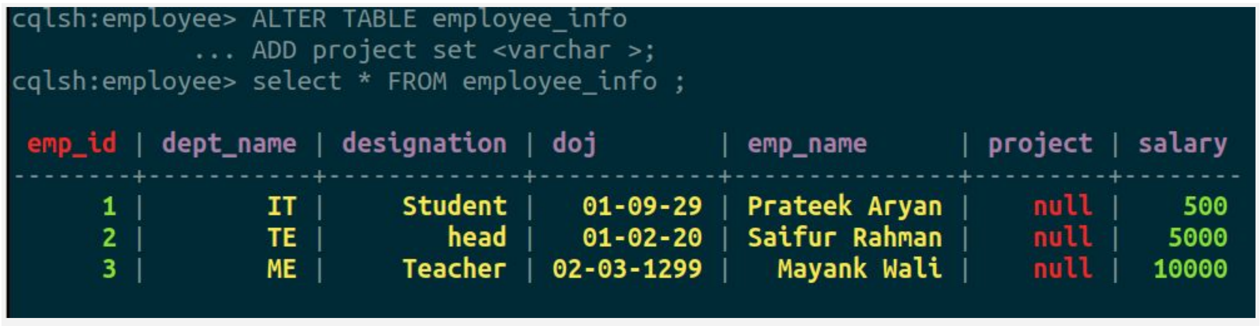
4. Sort the details of Employee records based on salary



```
cqlsh:employee> select* FROM employee_info ORDER BY salary;
InvalidRequest: Error from server: code=2200 [Invalid query] message="ORDER BY
 is only supported when the partition key is restricted by an EQ or an IN."
```

5. Alter the schema of the table Employee_Info to add a column Projects which stores a set of Projects done by the corresponding Employee.

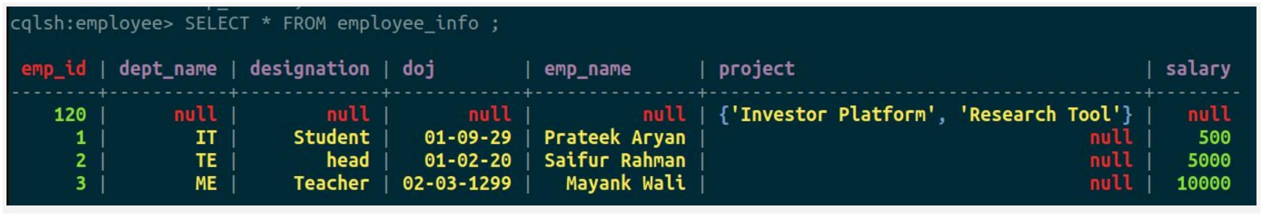
ALTER TABLE employee_info ADD Project VARCHAR;



```
cqlsh:employee> ALTER TABLE employee_info
            ... ADD project set <varchar >;
cqlsh:employee> select * FROM employee_info ;

 emp_id | dept_name | designation | doj        | emp_name      | project | salary
--------+-----------+-------------+------------+---------------+---------+--------
      1 |        IT |     Student |   01-09-29 | Prateek Aryan |    null |    500
      2 |        TE |        head |   01-02-20 | Saifur Rahman |    null |   5000
      3 |        ME |     Teacher | 02-03-1299 |   Mayank Wali |    null |  10000
```

6. Update the altered table to add project names.

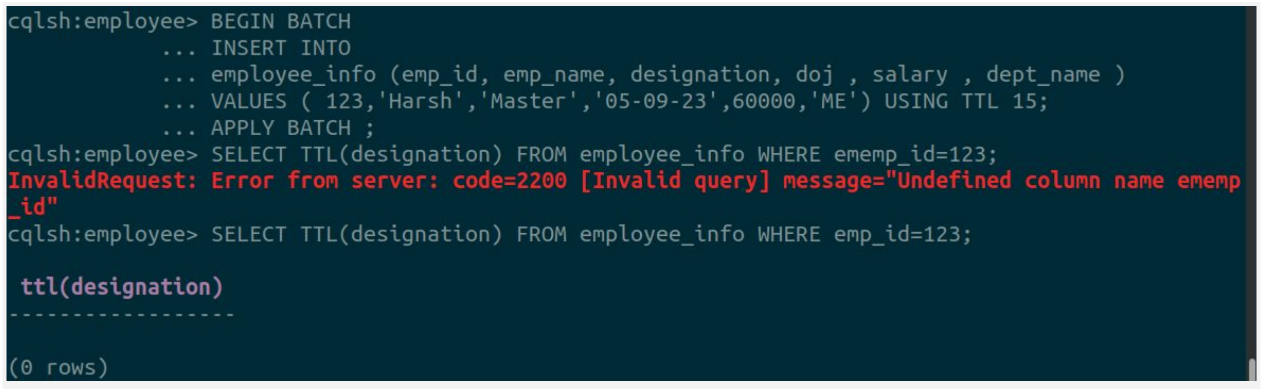
UPDATE employee_info SET project='TIP' WHERE emp_id=120;
UPDATE employee_info SET project='Sentiment Analysis' WHERE emp_id=121;
UPDATE employee_info SET project='Facial recognition' WHERE emp_id=123;
select * from employee_info;



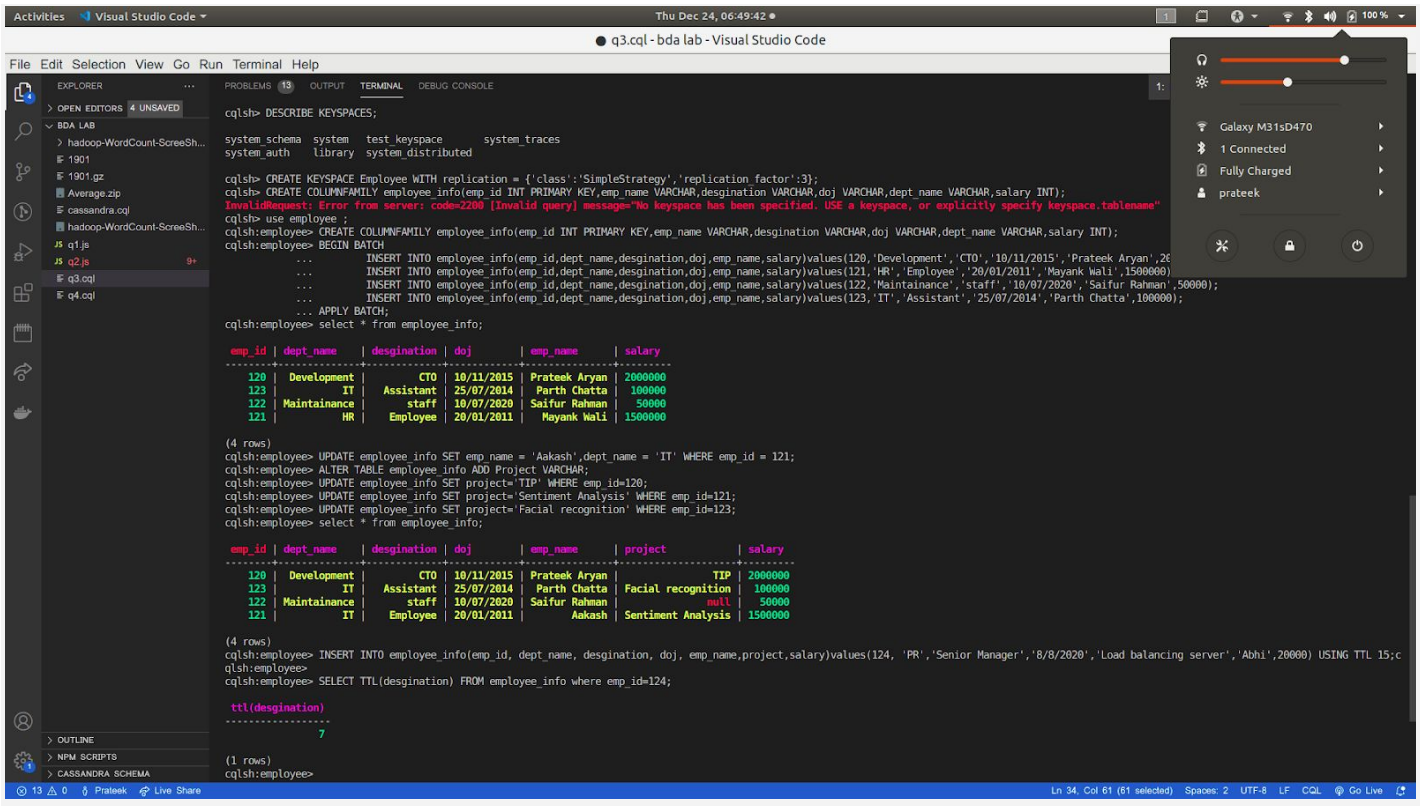
```
cqlsh:employee> SELECT * FROM employee_info ;

 emp_id | dept_name | designation | doj        | emp_name      | project                                | salary
--------+-----------+-------------+------------+---------------+----------------------------------------+--------
    120 |      null |        null |       null |          null | {'Investor Platform', 'Research Tool'} |   null
      1 |        IT |     Student |   01-09-29 | Prateek Aryan |                                   null |    500
      2 |        TE |        head |   01-02-20 | Saifur Rahman |                                   null |   5000
      3 |        ME |     Teacher | 02-03-1299 |   Mayank Wali |                                   null |  10000
```

7. Create a TTL of 15 seconds to display the values of Employees.

INSERT INTO employee_info(emp_id, dept_name, desgination, doj, emp_name,project,salary)values(124, 'PR','Senior Manager','8/8/2020','Load balancing server','Abhi',20000) USING TTL 15;
SELECT TTL(desgination) FROM employee_info where emp_id=124;



```
cqlsh:employee> BEGIN BATCH
            ... INSERT INTO
            ... employee_info (emp_id, emp_name, designation, doj , salary , dept_name )
            ... VALUES ( 123,'Harsh','Master','05-09-23',60000,'ME') USING TTL 15;
            ... APPLY BATCH ;
cqlsh:employee> SELECT TTL(designation) FROM employee_info WHERE ememp_id=123;
InvalidRequest: Error from server: code=2200 [Invalid query] message="Undefined column name ememp
_id"
cqlsh:employee> SELECT TTL(designation) FROM employee_info WHERE emp_id=123;

 ttl(designation)
------------------

(0 rows)
```



```
cqlsh> DESCRIBE KEYSPACES;

system_schema  system   test_keyspace       system_traces
system_auth    library  system_distributed

cqlsh> CREATE KEYSPACE Employee WITH replication = {'class':'SimpleStrategy','replication_factor':3};
cqlsh> CREATE COLUMNFAMILY employee_info(emp_id INT PRIMARY KEY,emp_name VARCHAR,desgination VARCHAR,doj VARCHAR,dept_name VARCHAR,salary INT);
InvalidRequest: Error from server: code=2200 [Invalid query] message="No keyspace has been specified. USE a keyspace, or explicitly specify keyspace.tablename"
cqlsh> use employee ;
cqlsh:employee> CREATE COLUMNFAMILY employee_info(emp_id INT PRIMARY KEY,emp_name VARCHAR,desgination VARCHAR,doj VARCHAR,dept_name VARCHAR,salary INT);
cqlsh:employee> BEGIN BATCH
            ...         INSERT INTO employee_info(emp_id,dept_name,desgination,doj,emp_name,salary)values(120,'Development','CTO','10/11/2015','Prateek Aryan',20
            ...         INSERT INTO employee_info(emp_id,dept_name,desgination,doj,emp_name,salary)values(121,'HR','Employee','20/01/2011','Mayank Wali',1500000)
            ...         INSERT INTO employee_info(emp_id,dept_name,desgination,doj,emp_name,salary)values(122,'Maintainance','staff','10/07/2020','Saifur Rahman',50000);
            ...         INSERT INTO employee_info(emp_id,dept_name,desgination,doj,emp_name,salary)values(123,'IT','Assistant','25/07/2014','Parth Chatta',100000);
            ... APPLY BATCH;
cqlsh:employee> select * from employee_info;

 emp_id | dept_name    | desgination | doj        | emp_name      | salary
--------+--------------+-------------+------------+---------------+---------
    120 |  Development |         CTO | 10/11/2015 | Prateek Aryan | 2000000
    123 |           IT |   Assistant | 25/07/2014 |  Parth Chatta |  100000
    122 | Maintainance |       staff | 10/07/2020 | Saifur Rahman |   50000
    121 |           HR |    Employee | 20/01/2011 |   Mayank Wali | 1500000

(4 rows)
cqlsh:employee> UPDATE employee_info SET emp_name = 'Aakash',dept_name = 'IT' WHERE emp_id = 121;
cqlsh:employee> ALTER TABLE employee_info ADD Project VARCHAR;
cqlsh:employee> UPDATE employee_info SET project='TIP' WHERE emp_id=120;
cqlsh:employee> UPDATE employee_info SET project='Sentiment Analysis' WHERE emp_id=121;
cqlsh:employee> UPDATE employee_info SET project='Facial recognition' WHERE emp_id=123;
cqlsh:employee> select * from employee_info;

 emp_id | dept_name    | desgination | doj        | emp_name      | project            | salary
--------+--------------+-------------+------------+---------------+--------------------+---------
    120 |  Development |         CTO | 10/11/2015 | Prateek Aryan |                TIP | 2000000
    123 |           IT |   Assistant | 25/07/2014 |  Parth Chatta | Facial recognition |  100000
    122 | Maintainance |       staff | 10/07/2020 | Saifur Rahman |               null |   50000
    121 |           IT |    Employee | 20/01/2011 |        Aakash | Sentiment Analysis | 1500000

(4 rows)
cqlsh:employee> INSERT INTO employee_info(emp_id, dept_name, desgination, doj, emp_name,project,salary)values(124, 'PR','Senior Manager','8/8/2020','Load balancing server','Abhi',20000) USING TTL 15;c
qlsh:employee>
cqlsh:employee> SELECT TTL(desgination) FROM employee_info where emp_id=124;

 ttl(desgination)
------------------
                7

(1 rows)
cqlsh:employee>
```

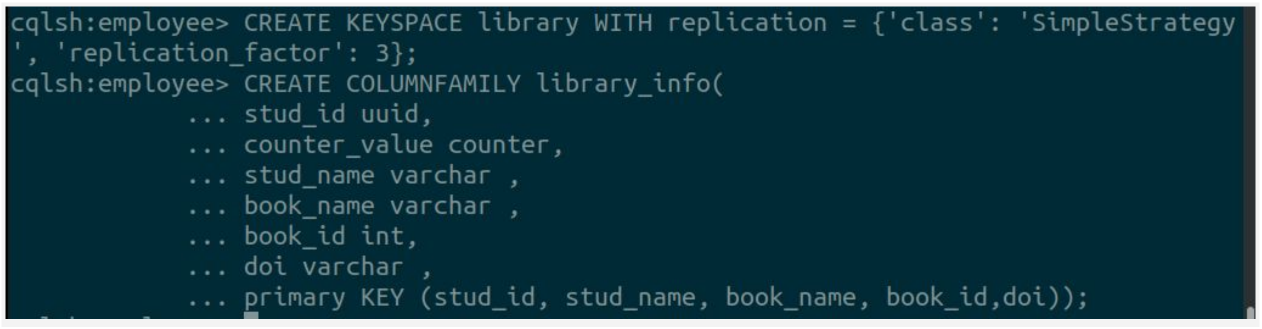
Question 4. Perform the following DB operations using Cassandra.

1. **Create a keyspace by name Library**

CREATE KEYSPACE library WITH replication = { 'class':'SimpleStrategy','replication_factor':3};

2. **Create a column family by name Library-Info with attributes**
Stud_Id Primary Key,
Counter_value of type Counter,
Stud_Name, Book-Name, Book-Id, Date_of_issue

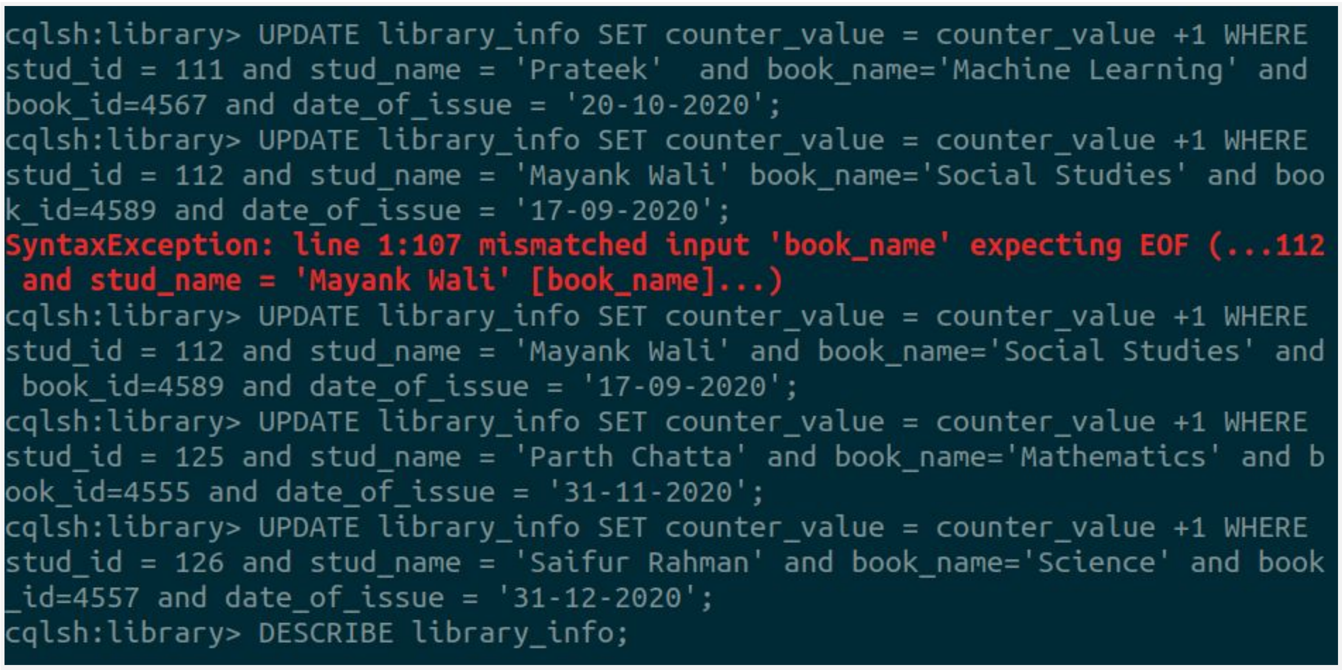
CREATE COLUMNFAMILY library_info(stud_id int, counter_value counter, stud_name text, book_name text, book_id int, date_of_issue text, PRIMARY KEY(stud_id, stud_name, book_name, book_id, date_of_issue));



```
cqlsh:employee> CREATE KEYSPACE library WITH replication = {'class': 'SimpleStrategy
', 'replication_factor': 3};
cqlsh:employee> CREATE COLUMNFAMILY library_info(
            ... stud_id uuid,
            ... counter_value counter,
            ... stud_name varchar ,
            ... book_name varchar ,
            ... book_id int,
            ... doi varchar ,
            ... primary KEY (stud_id, stud_name, book_name, book_id,doi));
```

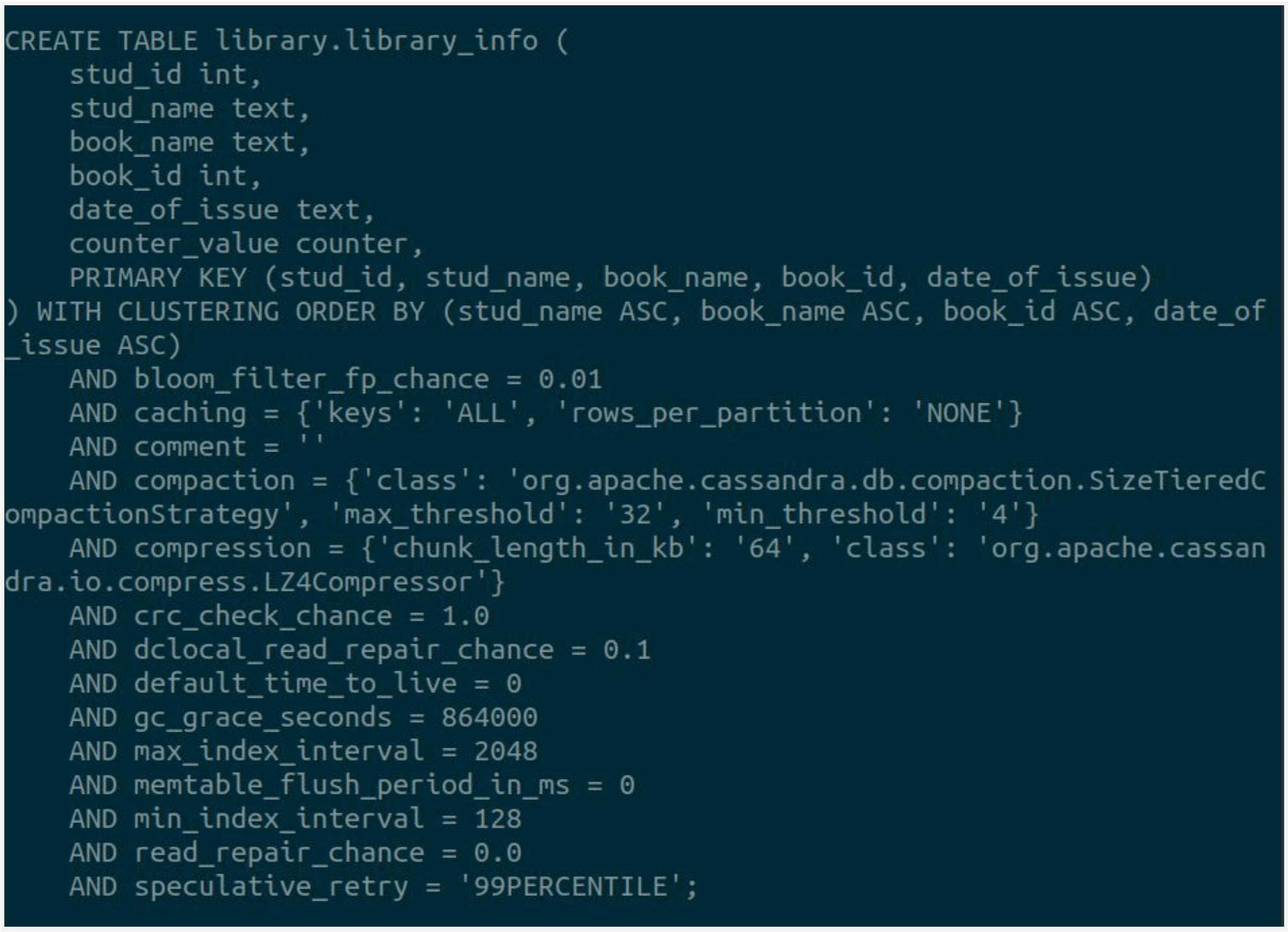
3. Insert the values into the table in batch

UPDATE library_info SET counter_value = counter_value +1 WHERE stud_id = 111 and stud_name = 'Prateek' and book_name='Machine Learning' and book_id=4567 and date_of_issue = '20-10-2020';
UPDATE library_info SET counter_value = counter_value +1 WHERE stud_id = 112 and stud_name = 'Mayank Wali' and book_name='Social Studies' and book_id=4589 and date_of_issue = '17-09-2020';
UPDATE library_info SET counter_value = counter_value +1 WHERE stud_id = 125 and stud_name = 'Parth Chatta' and book_name='Mathematics' and book_id=4555 and date_of_issue = '31-11-2020';
UPDATE library_info SET counter_value = counter_value +1 WHERE stud_id = 126 and stud_name = 'Saifur Rahman' and book_name='Science' and book_id=4557 and date_of_issue = '31-12-2020';


```
cqlsh:library> UPDATE library_info SET counter_value = counter_value +1 WHERE
stud_id = 111 and stud_name = 'Prateek'  and book_name='Machine Learning' and
book_id=4567 and date_of_issue = '20-10-2020';
cqlsh:library> UPDATE library_info SET counter_value = counter_value +1 WHERE
stud_id = 112 and stud_name = 'Mayank Wali' book_name='Social Studies' and boo
k_id=4589 and date_of_issue = '17-09-2020';
SyntaxException: line 1:107 mismatched input 'book_name' expecting EOF (...112
 and stud_name = 'Mayank Wali' [book_name]...)
cqlsh:library> UPDATE library_info SET counter_value = counter_value +1 WHERE
stud_id = 112 and stud_name = 'Mayank Wali' and book_name='Social Studies' and
 book_id=4589 and date_of_issue = '17-09-2020';
cqlsh:library> UPDATE library_info SET counter_value = counter_value +1 WHERE
stud_id = 125 and stud_name = 'Parth Chatta' and book_name='Mathematics' and b
ook_id=4555 and date_of_issue = '31-11-2020';
cqlsh:library> UPDATE library_info SET counter_value = counter_value +1 WHERE
stud_id = 126 and stud_name = 'Saifur Rahman' and book_name='Science' and book
_id=4557 and date_of_issue = '31-12-2020';
cqlsh:library> DESCRIBE library_info;
```

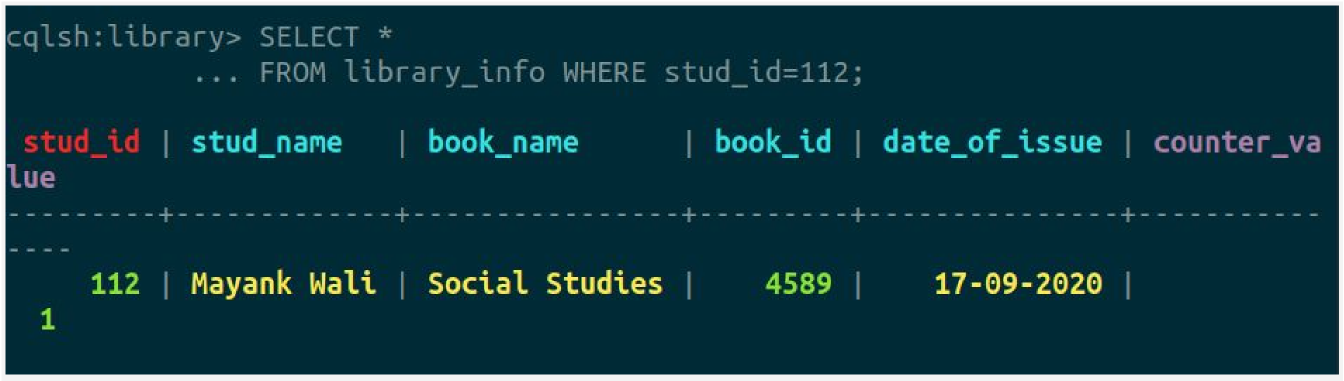
3. **Display the details of the table created and increase the value of the counter**

DESCRIBE library_info;


```
CREATE TABLE library.library_info (
    stud_id int,
    stud_name text,
    book_name text,
    book_id int,
    date_of_issue text,
    counter_value counter,
    PRIMARY KEY (stud_id, stud_name, book_name, book_id, date_of_issue)
) WITH CLUSTERING ORDER BY (stud_name ASC, book_name ASC, book_id ASC, date_of
_issue ASC)
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredC
ompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassan
dra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.1
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair_chance = 0.0
    AND speculative_retry = '99PERCENTILE';
```

4. Write a query to show that a student with id 112 has taken a book "BDA" 2 times.

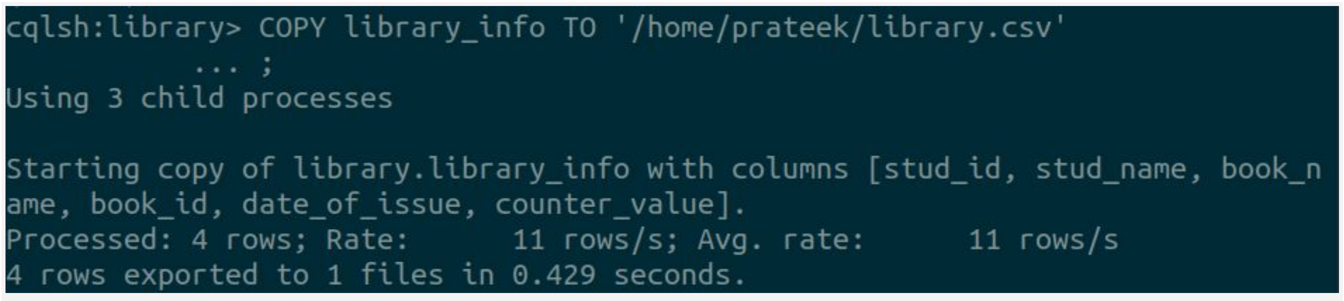
SELECT * FROM library_info WHERE stud_id=112;



```
cqlsh:library> SELECT *
           ... FROM library_info WHERE stud_id=112;

 stud_id | stud_name    | book_name      | book_id | date_of_issue | counter_va
lue
---------+--------------+----------------+---------+---------------+-----------
----
     112 | Mayank Wali | Social Studies |    4589 |    17-09-2020 |
  1
```

5. Export the created column to a CSV file.

COPY library_info TO '/home/prateek/library.csv';



```
cqlsh:library> COPY library_info TO '/home/prateek/library.csv'
           ... ;
Using 3 child processes

Starting copy of library.library_info with columns [stud_id, stud_name, book_n
ame, book_id, date_of_issue, counter_value].
Processed: 4 rows; Rate:      11 rows/s; Avg. rate:      11 rows/s
4 rows exported to 1 files in 0.429 seconds.
```

6. Import a given csv dataset from local file system into Cassandra column family

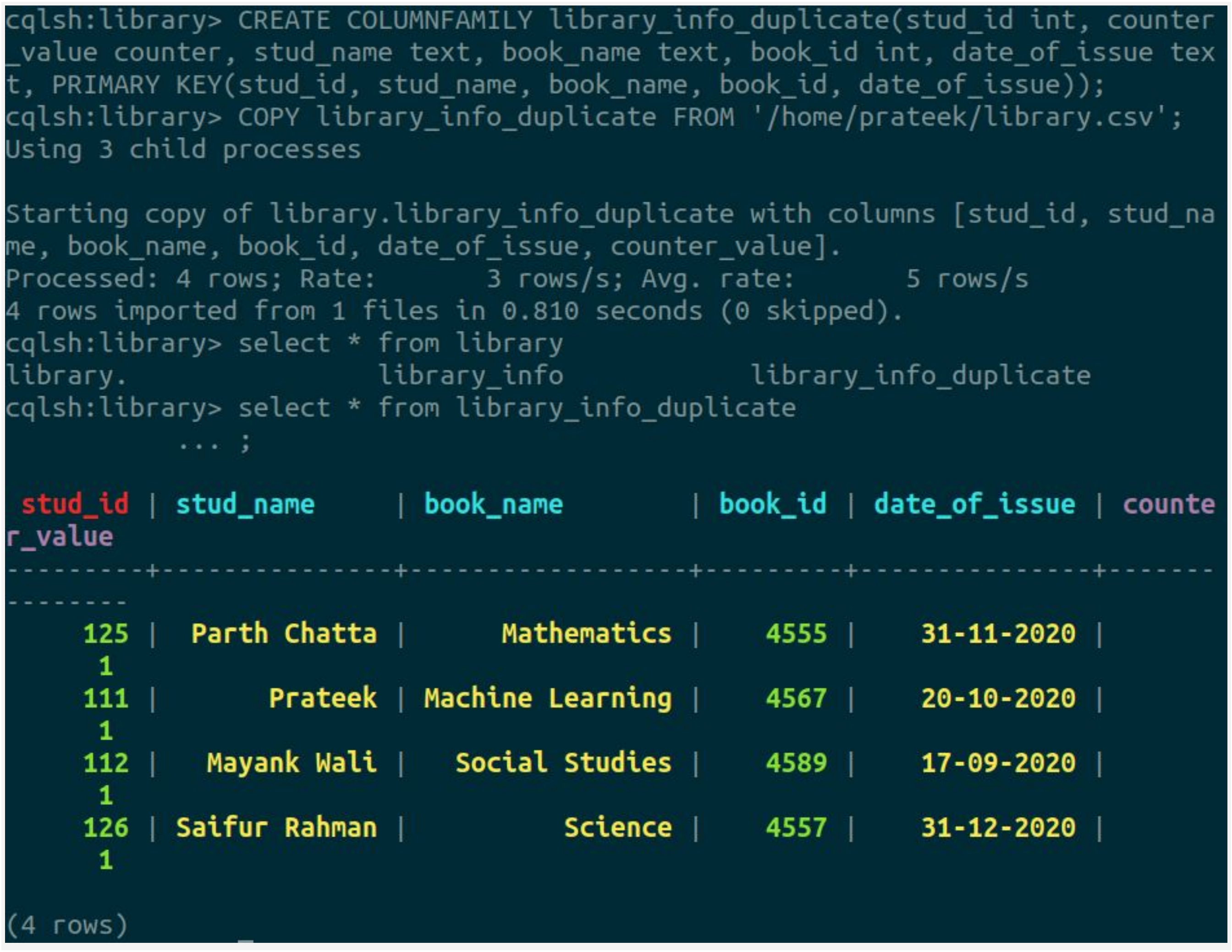
CREATE COLUMNFAMILY library_info_duplicate(stud_id int, counter_value counter, stud_name

text, book_name text, book_id int, date_of_issue text, PRIMARY KEY(stud_id, stud_name,

book_name, book_id, date_of_issue));

COPY library_info_duplicate FROM '/home/prateek/library.csv';

SELECT * FROM library_info_duplicate;



```
cqlsh:library> CREATE COLUMNFAMILY library_info_duplicate(stud_id int, counter
_value counter, stud_name text, book_name text, book_id int, date_of_issue tex
t, PRIMARY KEY(stud_id, stud_name, book_name, book_id, date_of_issue));
cqlsh:library> COPY library_info_duplicate FROM '/home/prateek/library.csv';
Using 3 child processes

Starting copy of library.library_info_duplicate with columns [stud_id, stud_na
me, book_name, book_id, date_of_issue, counter_value].
Processed: 4 rows; Rate:       3 rows/s; Avg. rate:       5 rows/s
4 rows imported from 1 files in 0.810 seconds (0 skipped).
cqlsh:library> select * from library
library.                library_info            library_info_duplicate
cqlsh:library> select * from library_info_duplicate
           ... ;

 stud_id | stud_name     | book_name        | book_id | date_of_issue | counte
r_value
---------+---------------+------------------+---------+---------------+-------
--------
     125 |  Parth Chatta |      Mathematics |    4555 |    31-11-2020 |
       1
     111 |       Prateek | Machine Learning |    4567 |    20-10-2020 |
       1
     112 |    Mayank Wali |   Social Studies |    4589 |    17-09-2020 |
       1
     126 | Saifur Rahman |          Science |    4557 |    31-12-2020 |
       1

(4 rows)
```

Question 5. Develop a MapReduce program to count the number of occurrences of words in a given file.

WCMapper.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;

public class WCMapper extends MapReduceBase implements Mapper<LongWritable,
                        Text, Text, IntWritable> {

  // Map function
  public void map(LongWritable key, Text value, OutputCollector<Text,
        IntWritable> output, Reporter rep) throws IOException
  {

    String line = value.toString();

    // Splitting the line on spaces
    for (String word : line.split(" "))
    {
      if (word.length() > 0)
      {
        output.collect(new Text(word), new IntWritable(1));
      }
    }
  }
}
```

WCReducer.java

```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;

public class WCReducer extends MapReduceBase implements Reducer<Text,
                                              IntWritable, Text, IntWritable> {
```

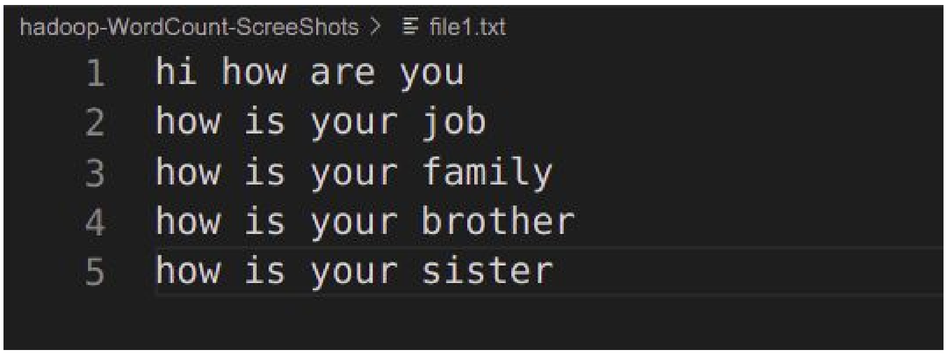
```
        // Reduce function
        public void reduce(Text key, Iterator<IntWritable> value,
                                OutputCollector<Text, IntWritable> output,
                                                Reporter rep) throws IOException
        {

                int count = 0;

                // Counting the frequency of each words
                while (value.hasNext())
                {
                        IntWritable i = value.next();
                        count += i.get();
                }

                output.collect(key, new IntWritable(count));
        }
}
```

WCDriver.java

```
import java.io.IOException;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class WCDriver extends Configured implements Tool {

        public int run(String args[]) throws IOException
        {
                if (args.length < 2)
                {
                        System.out.println("Please give valid inputs");
                        return -1;
                }

                JobConf conf = new JobConf(WCDriver.class);
                FileInputFormat.setInputPaths(conf, new Path(args[0]));
                FileOutputFormat.setOutputPath(conf, new Path(args[1]));
                conf.setMapperClass(WCMapper.class);
                conf.setReducerClass(WCReducer.class);
                conf.setMapOutputKeyClass(Text.class);
                conf.setMapOutputValueClass(IntWritable.class);
                conf.setOutputKeyClass(Text.class);
                conf.setOutputValueClass(IntWritable.class);
                JobClient.runJob(conf);
                return 0;
        }

        // Main Method
        public static void main(String args[]) throws Exception
        {
                int exitCode = ToolRunner.run(new WCDriver(), args);
                System.out.println(exitCode);
        }
}
```

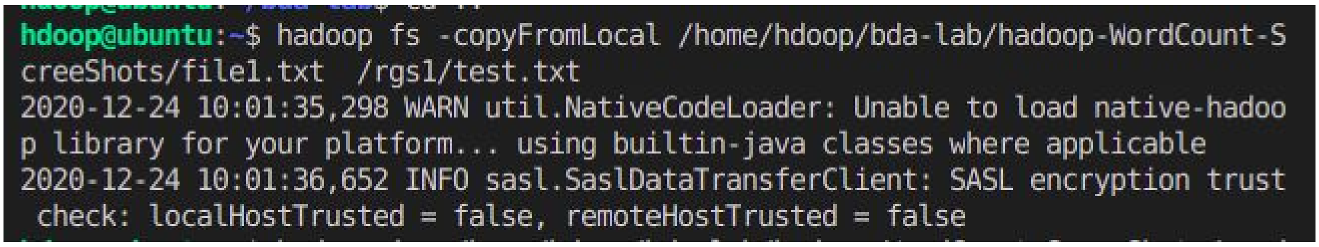
HADOOP COMMANDS :

file1.txt


hadoop-WordCount-ScreeShots > file1.txt

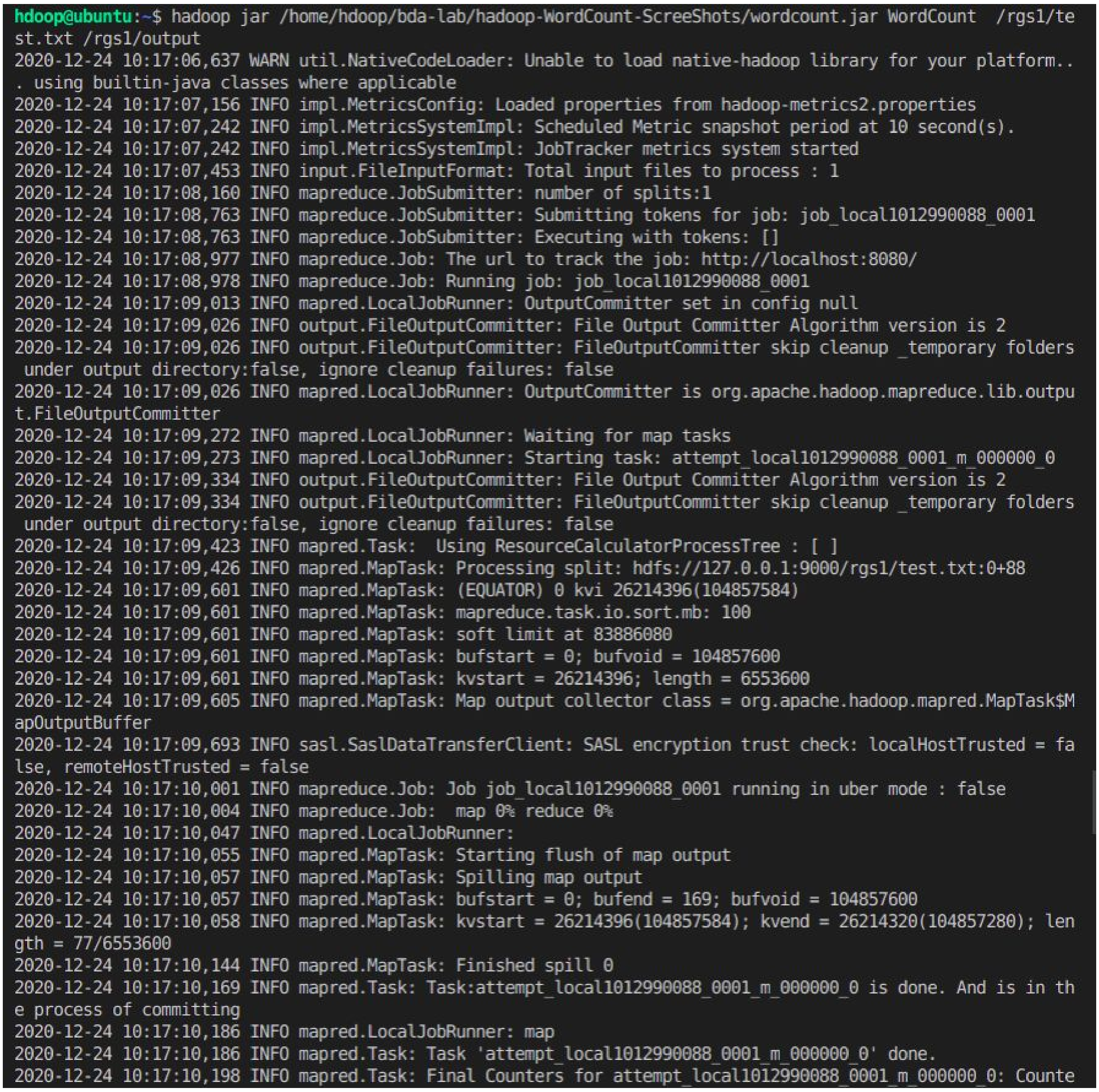
```
hi how are you
how is your job
how is your family
how is your brother
how is your sister
```

hadoop fs -copyFromLocal /home/hdoop/bda-lab/hadoop-WordCount-ScreeShots/file1.txt /rgs1/test.txt



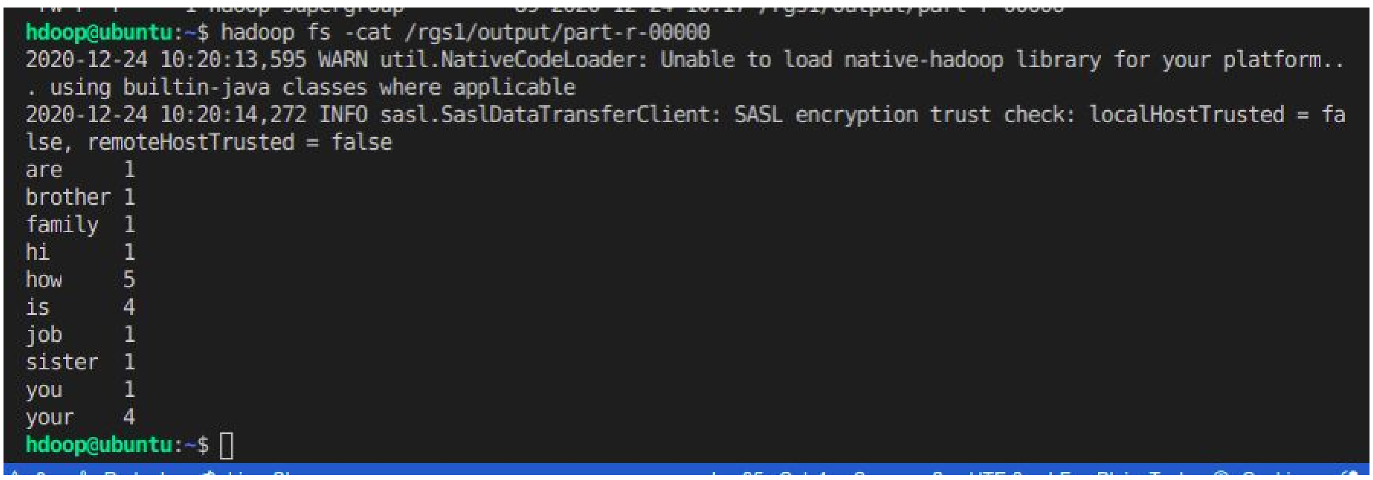
```
hdoop@ubuntu:~$ hadoop fs -copyFromLocal /home/hdoop/bda-lab/hadoop-WordCount-S
creeShots/file1.txt  /rgs1/test.txt
2020-12-24 10:01:35,298 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
2020-12-24 10:01:36,652 INFO sasl.SaslDataTransferClient: SASL encryption trust
 check: localHostTrusted = false, remoteHostTrusted = false
```

hadoop jar /home/hdoop/bda-lab/hadoop-WordCount-ScreeShots/wordcount.jar WordCount /rgs1/test.txt /rgs1/output



```
hdoop@ubuntu:~$ hadoop jar /home/hdoop/bda-lab/hadoop-WordCount-ScreeShots/wordcount.jar WordCount  /rgs1/te
st.txt /rgs1/output
2020-12-24 10:17:06,637 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform..
. using builtin-java classes where applicable
2020-12-24 10:17:07,156 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2020-12-24 10:17:07,242 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2020-12-24 10:17:07,242 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2020-12-24 10:17:07,453 INFO input.FileInputFormat: Total input files to process : 1
2020-12-24 10:17:08,160 INFO mapreduce.JobSubmitter: number of splits:1
2020-12-24 10:17:08,763 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1012990088_0001
2020-12-24 10:17:08,763 INFO mapreduce.JobSubmitter: Executing with tokens: []
2020-12-24 10:17:08,977 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2020-12-24 10:17:08,978 INFO mapreduce.Job: Running job: job_local1012990088_0001
2020-12-24 10:17:09,013 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2020-12-24 10:17:09,026 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2020-12-24 10:17:09,026 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders
 under output directory:false, ignore cleanup failures: false
2020-12-24 10:17:09,026 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.lib.outpu
t.FileOutputCommitter
2020-12-24 10:17:09,272 INFO mapred.LocalJobRunner: Waiting for map tasks
2020-12-24 10:17:09,273 INFO mapred.LocalJobRunner: Starting task: attempt_local1012990088_0001_m_000000_0
2020-12-24 10:17:09,334 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2020-12-24 10:17:09,334 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders
 under output directory:false, ignore cleanup failures: false
2020-12-24 10:17:09,423 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
2020-12-24 10:17:09,426 INFO mapred.MapTask: Processing split: hdfs://127.0.0.1:9000/rgs1/test.txt:0+88
2020-12-24 10:17:09,601 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
2020-12-24 10:17:09,601 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
2020-12-24 10:17:09,601 INFO mapred.MapTask: soft limit at 83886080
2020-12-24 10:17:09,601 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600
2020-12-24 10:17:09,601 INFO mapred.MapTask: kvstart = 26214396; length = 6553600
2020-12-24 10:17:09,605 INFO mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$M
apOutputBuffer
2020-12-24 10:17:09,693 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = fa
lse, remoteHostTrusted = false
2020-12-24 10:17:10,001 INFO mapreduce.Job: Job job_local1012990088_0001 running in uber mode : false
2020-12-24 10:17:10,004 INFO mapreduce.Job:  map 0% reduce 0%
2020-12-24 10:17:10,047 INFO mapred.LocalJobRunner:
2020-12-24 10:17:10,055 INFO mapred.MapTask: Starting flush of map output
2020-12-24 10:17:10,057 INFO mapred.MapTask: Spilling map output
2020-12-24 10:17:10,057 INFO mapred.MapTask: bufstart = 0; bufend = 169; bufvoid = 104857600
2020-12-24 10:17:10,058 INFO mapred.MapTask: kvstart = 26214396(104857584); kvend = 26214320(104857280); len
gth = 77/6553600
2020-12-24 10:17:10,144 INFO mapred.MapTask: Finished spill 0
2020-12-24 10:17:10,169 INFO mapred.Task: Task:attempt_local1012990088_0001_m_000000_0 is done. And is in th
e process of committing
2020-12-24 10:17:10,186 INFO mapred.LocalJobRunner: map
2020-12-24 10:17:10,186 INFO mapred.Task: Task 'attempt_local1012990088_0001_m_000000_0' done.
2020-12-24 10:17:10,198 INFO mapred.Task: Final Counters for attempt_local1012990088_0001_m_000000_0: Counte
```

hadoop fs -ls /rgs1/output/part-r-00000


```
hdoop@ubuntu:~$ hadoop fs -cat /rgs1/output/part-r-00000
2020-12-24 10:20:13,595 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform..
. using builtin-java classes where applicable
2020-12-24 10:20:14,272 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = fa
lse, remoteHostTrusted = false
are     1
brother 1
family  1
hi      1
how     5
is      4
job     1
sister  1
you     1
your    4
hdoop@ubuntu:~$ 
```

Question 6. For the given file, Create a Map Reduce program.

a) Find the average temperature for each year from the NCDC data set.

AverageMapper.java

```
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import java.io.IOException;

public class AverageMapper extends Mapper <LongWritable, Text, Text, IntWritable>
{

public static final int MISSING = 9999;

public void map(LongWritable key, Text value, Context context) throws
IOException,  InterruptedException
    {
        String line = value.toString();
        String year = line.substring(15,19);
        int temperature;
        if (line.charAt(87)=='+')
            temperature = Integer.parseInt(line.substring(88, 92));
        else
```

```
                        temperature = Integer.parseInt(line.substring(87, 92));

                    String quality = line.substring(92, 93);
                    if(temperature != MISSING && quality.matches("[01459]"))
                    context.write(new Text(year),new IntWritable(temperature));
            }
        }
```

AverageReducer.java

```
        import org.apache.hadoop.mapreduce.*;
        import java.io.IOException;

        public class AverageReducer extends Reducer <Text, IntWritable,Text, IntWritable >
         {
        public void reduce(Text key,  Iterable<IntWritable> values, Context context) throws IOException,
        InterruptedException
            {
            int max_temp = 0;
            int count = 0;
            for (IntWritable value : values)
                {
                    max_temp += value.get();
                    count+=1;
                }
            context.write(key, new IntWritable(max_temp/count));
            }
        }
```

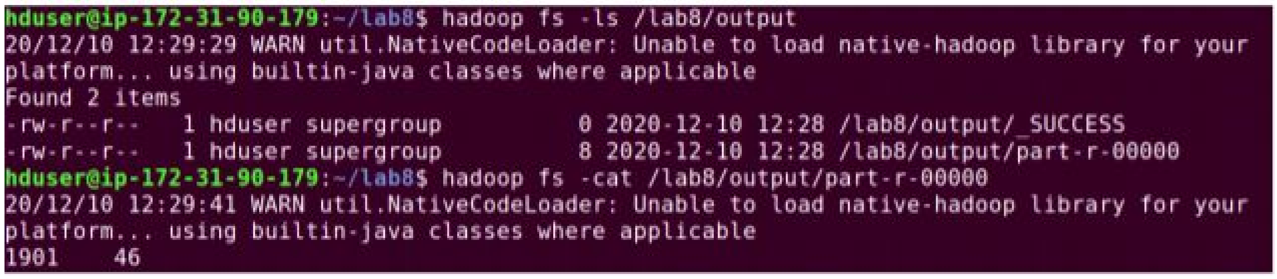
AverageDriver.java

```
        import org.apache.hadoop.io.*;
        import org.apache.hadoop.fs.*;
        import org.apache.hadoop.mapreduce.*;
        import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
        import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

        public class AverageDriver
        {

            public static void main (String[] args) throws Exception
            {
                if (args.length != 2)
                {
                    System.err.println("Please Enter the input and output parameters");
                    System.exit(-1);
                }
                Job job = new Job();
                job.setJarByClass(AverageDriver.class);
                job.setJobName("Max temperature");

                FileInputFormat.addInputPath(job,new Path(args[0]));
                FileOutputFormat.setOutputPath(job,new Path (args[1]));

                job.setMapperClass(AverageMapper.class);
                job.setReducerClass(AverageReducer.class);

                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(IntWritable.class);

                System.exit(job.waitForCompletion(true)?0:1);
        }
    }
```

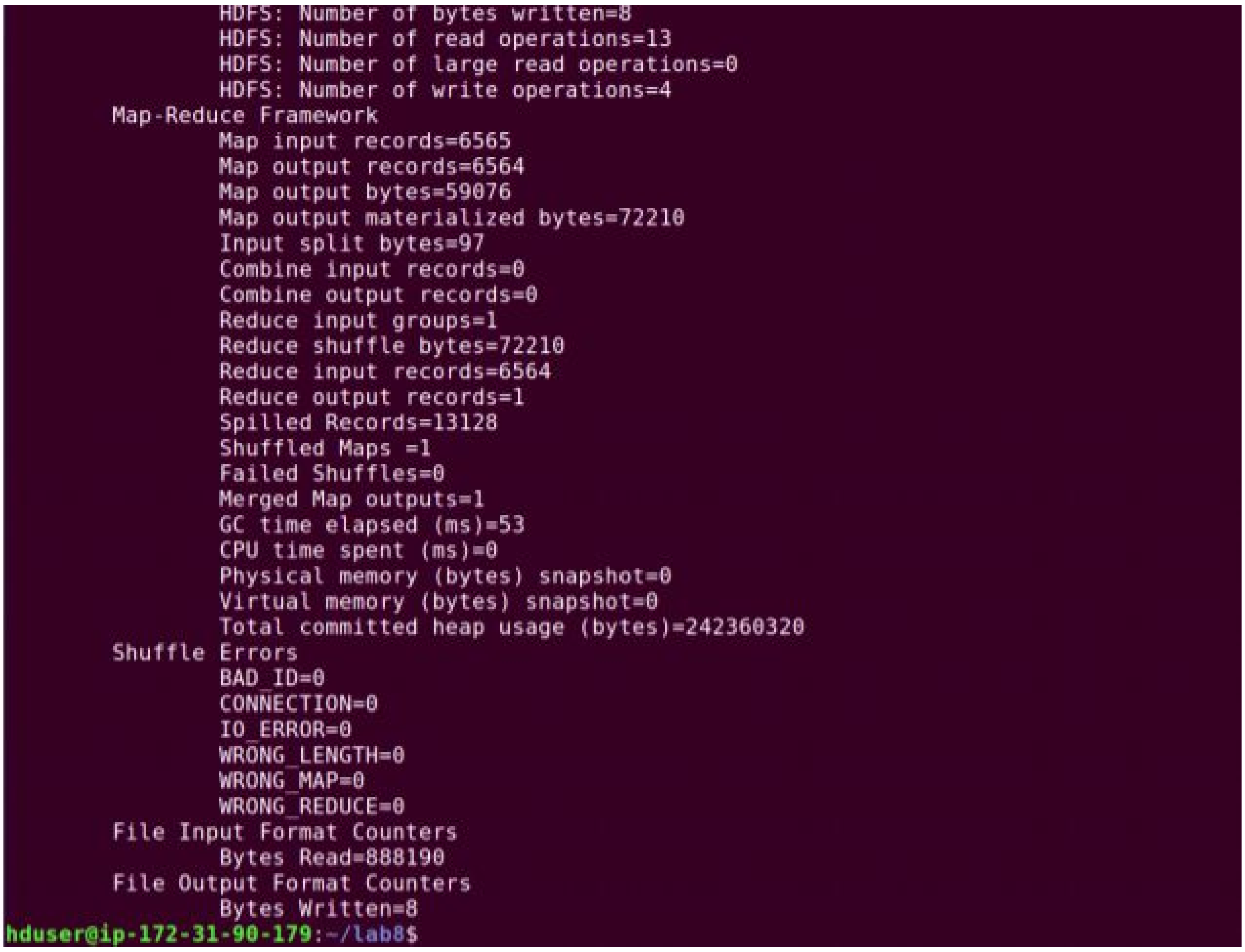
Hadoop Commands :

```
start-dfs.sh
start-yarn.sh
hdfs dfs -put /home/hdoop/Average/1909.txt
hadoop jar /home/hdoop/bda-lab/average/wordcount.jar AverageDriver  /average/test.txt
/average/output
```

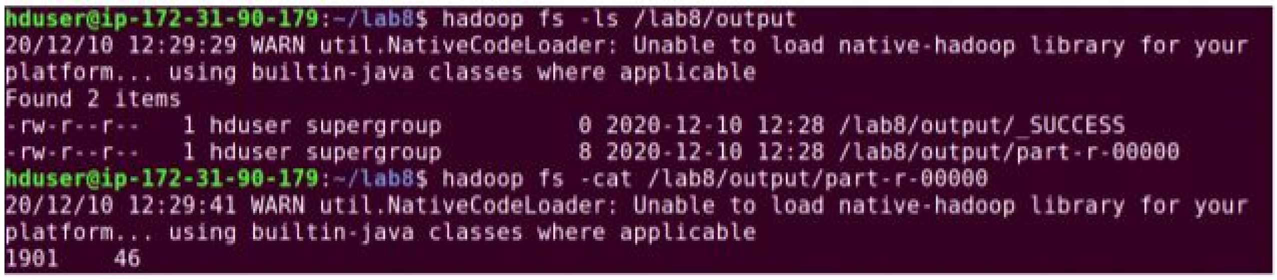
Creating a jar file.



```
hduser@ip-172-31-90-179:~/lab8$ hadoop fs -ls /lab8/output
20/12/10 12:29:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 hduser supergroup          0 2020-12-10 12:28 /lab8/output/_SUCCESS
-rw-r--r--   1 hduser supergroup          8 2020-12-10 12:28 /lab8/output/part-r-00000
hduser@ip-172-31-90-179:~/lab8$ hadoop fs -cat /lab8/output/part-r-00000
20/12/10 12:29:41 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable
1901    46
```



```
                HDFS: Number of bytes written=8
                HDFS: Number of read operations=13
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=4
        Map-Reduce Framework
                Map input records=6565
                Map output records=6564
                Map output bytes=59076
                Map output materialized bytes=72210
                Input split bytes=97
                Combine input records=0
                Combine output records=0
                Reduce input groups=1
                Reduce shuffle bytes=72210
                Reduce input records=6564
                Reduce output records=1
                Spilled Records=13128
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=53
                CPU time spent (ms)=0
                Physical memory (bytes) snapshot=0
                Virtual memory (bytes) snapshot=0
                Total committed heap usage (bytes)=242360320
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=888190
        File Output Format Counters
                Bytes Written=8
hduser@ip-172-31-90-179:~/lab8$
```



```
hduser@ip-172-31-90-179:~/lab8$ hadoop fs -ls /lab8/output
20/12/10 12:29:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 hduser supergroup          0 2020-12-10 12:28 /lab8/output/_SUCCESS
-rw-r--r--   1 hduser supergroup          8 2020-12-10 12:28 /lab8/output/part-r-00000
hduser@ip-172-31-90-179:~/lab8$ hadoop fs -cat /lab8/output/part-r-00000
20/12/10 12:29:41 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable
1901    46
```

OUTPUT :
1901 46

Question 7. Write Queries in Hive to do the following.

1. Create an external table named with the following attributes
-> Empl_ID ->Emp_Name -> Designation -> Salary

```
CREATE DATABASE IF NOT EXISTS lab9 COMMENT 'employee program' WITH DBPROPERTIES
('creator'=PRATEEK);
SHOW DATABASES;
DESCRIBE DATABASE lab9;
USE lab9;
CREATE EXTERNAL TABLE IF NOT EXISTS Employee(EmpID INT,EmpName
STRING, Designation STRING,Salary FLOAT) ROW FORMAT DELIMITED FIELDS TERMINATED
BY '\t';
```

2. Load data into table from a given file

```
LOAD DATA LOCAL INPATH '/home/prateek/Downloads/employeeInput.txt' OVERWRITE INTO
TABLE Employee;
SELECT * FROM Employee;
```

3. Create a view to Generate a query to retrieve the employee details who earn a salary of more than Rs 30000.

```
CREATE VIEW emp_30000 AS SELECT * FROM Employee WHERE Salary>30000;
SELECT * FROM emp_30000;
```

4. Alter the table to add a column Dept_Id and Generate a query to retrieve the employee details in order by using Dept_Id

```
ALTER TABLE Employee ADD COLUMNS(DeptID INT);
LOAD DATA LOCAL INPATH '/home/prateek/Downloads/employeeInputAltered.txt'
OVERWRITE INTO TABLE Employee;
SELECT * FROM Employee;
SELECT * FROM Employee ORDER BY DeptID;
```

5. Generate a query to retrieve the number of employees in each department whose salary is greater than 30000

```
SELECT DeptID,count(*) FROM Employee WHERE Salary>=30000 GROUP BY DeptID;
```

6. Create another table Department with attributes
-> Dept_Id ->Dept_name ->Emp_Id

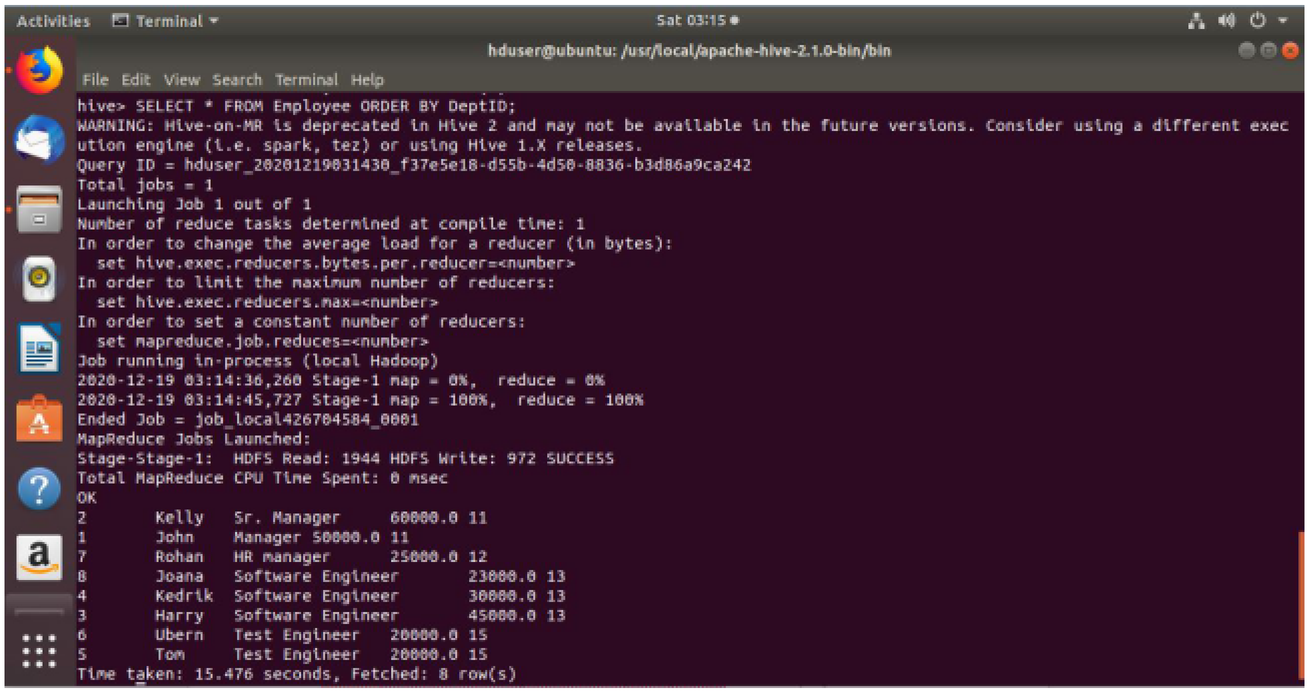
```
CREATE EXTERNAL TABLE IF NOT EXISTS Department(DeptId INT,DeptName STRING) ROW
FORMAT DELIMITED FIELDS TERMINATED BY '\t';
```

LOAD DATA LOCAL INPATH '/home/prateek/Downloads/DepartmentInput.txt' OVERWRITE INTO TABLE Department;
SELECT * FROM Department;

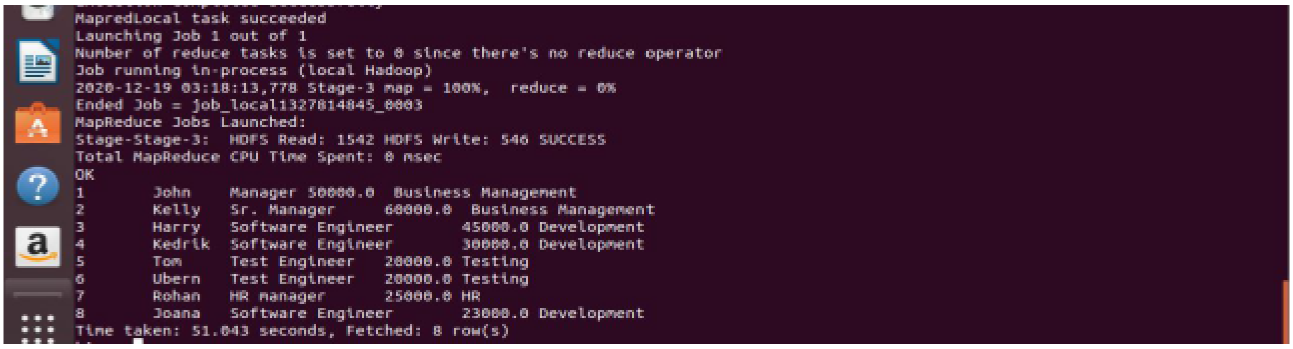
7.Display the cumulative details of each employee along with department details

SELECT a.EmpID,a.EmpName,a.Designation,a.Salary,b.DeptName FROM Employee a JOIN Department b ON a.DeptID=b.DeptId;

Screenshots:


Activities
Terminal
Sat 03:15
hduser@ubuntu: /usr/local/apache-hive-2.1.0-bin/bin
File Edit View Search Terminal Help

```
hive> SELECT * FROM Employee ORDER BY DeptID;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different exec
ution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = hduser_20201219031430_f37e5e18-d55b-4d50-8836-b3d86a9ca242
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>
Job running in-process (local Hadoop)
2020-12-19 03:14:36,260 Stage-1 map = 0%,  reduce = 0%
2020-12-19 03:14:45,727 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local426704584_0001
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 1944 HDFS Write: 972 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
2       Kelly   Sr. Manager     60000.0 11
1       John    Manager 50000.0 11
7       Rohan   HR manager      25000.0 12
8       Joana   Software Engineer       23000.0 13
4       Kedrik  Software Engineer       30000.0 13
3       Harry   Software Engineer       45000.0 13
6       Ubern   Test Engineer   20000.0 15
5       Tom     Test Engineer   20000.0 15
Time taken: 15.476 seconds, Fetched: 8 row(s)
```



```
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Job running in-process (local Hadoop)
2020-12-19 03:18:13,778 Stage-3 map = 100%,  reduce = 0%
Ended Job = job_local1327814845_0003
MapReduce Jobs Launched:
Stage-Stage-3:  HDFS Read: 1542 HDFS Write: 546 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
1       John    Manager 50000.0  Business Management
2       Kelly   Sr. Manager     60000.0  Business Management
3       Harry   Software Engineer       45000.0 Development
4       Kedrik  Software Engineer       30000.0 Development
5       Tom     Test Engineer   20000.0 Testing
6       Ubern   Test Engineer   20000.0 Testing
7       Rohan   HR manager      25000.0 HR
8       Joana   Software Engineer       23000.0 Development
Time taken: 51.043 seconds, Fetched: 8 row(s)
```