

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.



LAB REPORT
on

BIG DATA ANALYTICS **(20CS6PEBDA)**

Submitted by

ANAGHA ACHARYA(1BM19CS224)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019

May-2022 to July-2022

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering



CERTIFICATE

This is to certify that the Lab work entitled “**BIG DATA ANALYTICS**” carried out by **ANAGHA ACHARYA(1BM19CS224)**, who is bonafde student of **B. M. S. College of Engineering**. It is in partialfulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **BIG DATA ANALYTICS - (20CS6PEBDA)** work prescribed for the said degree.

Dr. Shyamala G
Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

Index Sheet

Sl. No.	Experiment Title
1	Employee Database
2	Library
3	Mongo (CRUD)
4	Hadoop installation
5	HDFS Commands
6	Create a Map Reduce program to a) find average temperature for each yearfrom NCDC data set. b) find the mean max temperature for every month
7	For a given Text file, create a Map Reduce program to sort the content in an alphabeticorder listing only top 10 maximum occurrences of words.
8	Create a Map Reduce program to demonstrating join operation
9	Program to print word count on scala shell and print “Hello world” on scala IDE
10	Using RDD and FlatMap count how many times each word appears in a file and writeout a list of words whose count is strictly greater than 4 using Spark

Course Outcome

CO1	Apply the concept of NoSQL, Hadoop or Spark for a given task
CO2	Analyze the Big Data and obtain insight using data analytics mechanisms.
CO3	Design and implement Big data applications by applying NoSQL, Hadoop or Spark

Lab1

Perform the following DB operations using Cassandra.

1. Create a keyspace by name Employee
2. Create a column family by name Employee-Info with attributes Emp_Id Primary Key, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name
3. Insert the values into the table in batch
4. Update Employee name and Department of Emp-Id 121
5. Sort the details of Employee records based on salary
6. Alter the schema of the table Employee_Info to add a column Projects which stores a set of Projects done by the corresponding Employee.
7. Update the altered table to add project names.
8. Create a TTL of 15 seconds to display the values of Employees.

```
bmsce@bmsce-not-so-precised-animo-succker:~$ cqlsh
```

```
Connected to Test Cluster at 127.0.0.1:9042.
```

```
[cqlsh 5.0.1 | Cassandra 3.11.4 | CQL spec 3.4.4 | Native protocol v4]
```

```
Use HELP for help.
```

```
cqlsh> create keyspace employee_224 with replication={'class':'SimpleStrategy','replication_factor':1};
```

```
cqlsh> use employee_224;
```

```
cqlsh:employee_224> create table empinfo(emp_id int, emp_name text, designation text, doj date, salary double, dept_name text, primary key(emp_id, salary));
```

```
cqlsh:employee_224> describe empinfo;
```

```
CREATE TABLE employee_224.empinfo (
```

```
    emp_id int,
```

```
    salary double,
```

```
    dept_name text,
```

```
    designation text,
```

```
    doj date,
```

```
    emp_name text,
```

```
    PRIMARY KEY (emp_id, salary)
```

```
) WITH CLUSTERING ORDER BY (salary 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.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
```

```
    AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassandra.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';
```

```
cqlsh:employee_224> begin batch
```

```
... insert into empinfo(emp_id,emp_name,designation,doj,salary,dept_name) values(1,'Abhinav','Director','2010-08-28',5000000,'Business')
```

```
... apply batch;
```

```
cqlsh:employee_224> begin batch
```

```
... insert into empinfo(emp_id,emp_name,designation,doj,salary,dept_name) values(2,'Saurabh','Manager','2020-05-20',800000,'Sales')
```

```
... insert into empinfo(emp_id,emp_name,designation,doj,salary,dept_name) values(3,'Rythm','HR','2021-10-10',1000000,'HR')
```

```
... insert into empinfo(emp_id,emp_name,designation,doj,salary,dept_name) values(4,'Anagha','Executive Director','2016-03-27',3000000,'R&D')
```

```
... apply batch;
```

```
cqlsh:employee_224> select * from empinfo;
```

emp_id	salary	dept_name	designation	doj	emp_name
1	5e+06	Business	Director	2010-08-28	Abhinav
2	8e+05	Sales	Manager	2020-05-20	Saurabh
4	3e+06	R&D	Executive Director	2016-03-27	Anagha
3	1e+06	HR	HR	2021-10-10	Rythm

(4 rows)

```
cqlsh:employee_224> update empinfo set emp_name='Manu' where emp_id=3 and salary=1000000;
```

```
cqlsh:employee_224> select * from empinfo;
```

emp_id	salary	dept_name	designation	doj	emp_name
1	5e+06	Business	Director	2010-08-28	Abhinav
2	8e+05	Sales	Manager	2020-05-20	Saurabh
4	3e+06	R&D	Executive Director	2016-03-27	Anagha
3	1e+06	HR	HR	2021-10-10	Manu

```
cqlsh:employee_224> alter table empinfo add projects set<text>;
```

```
cqlsh:employee_224> update empinfo set projects=projects+{'Psychology of body','Strength and conditioning'} where emp_id=4 and salary=3000000;
```

```
cqlsh:employee_224> update empinfo set projects=projects+{'Analytics','Risk assesment'} where emp_id=2 and salary=800000;
```

```
qlsh:employee_224> update empinfo set projects=projects+{'Diversity management'} where emp_id=3 and salary=1000000;
```

```
cqlsh:employee_224> update empinfo set projects=projects+{'Role of motivation in improving organisational performance'} where emp_id=1 and salary=5000000;
```

```
cqlsh:employee_224> select * from empinfo;
```

emp_id	salary	dept_name	designation	doj	emp_name	projects
1	5e+06	Business	Director	2010-08-28	Abhinav	{ 'Role of motivation in improving organisational performance' }
2	8e+05	Sales	Manager	2020-05-20	Saurabh	{ 'Analytics', 'Risk assesment' }
4	3e+06	R&D	Executive Director	2016-03-27	Anagha	{ 'Psychology of body', 'Strength and

conditioning'}

```
3 | 1e+06 | HR | HR | 2021-10-10 | Manu | {'Diversity management'}
cqlsh:employee_224> insert into empinfo(emp_id,emp_name,designation,doj,salary,dept_name,projects)
values(5,'Rakesh','Lawyer','2012-05-07',1200000,'Legal',{'Doping crime in international law'}) using ttl 15;
cqlsh:employee_224> select * from empinfo;
```

emp_id	salary	dept_name	designation	doj	emp_name	projects
5	1.2e+06	Legal	Lawyer	2012-05-07	Rakesh	{ 'Doping crime in international law' }
1	5e+06	Business	Director	2010-08-28	Abhinav	{ 'Role of motivation in improving organisational performance' }
2	8e+05	Sales	Manager	2020-05-20	Saurabh	{ 'Analytics', 'Risk assesment' }
4	3e+06	R&D	Executive Director	2016-03-27	Anagha	{ 'Psychology of body', 'Strength and conditioning' }
3	1e+06	HR	HR	2021-10-10	Manu	{ 'Diversity management' }

(5 rows)

```
cqlsh:employee_224> select ttl(emp_name) from empinfo where emp_id=5;
```

ttl(emp_name)

8
cqlsh:employee_224> select * from empinfo;

emp_id	salary	dept_name	designation	doj	emp_name	projects
1	5e+06	Business	Director	2010-08-28	Abhinav	{ 'Role of motivation in improving organisational performance' }
2	8e+05	Sales	Manager	2020-05-20	Saurabh	{ 'Analytics', 'Risk assesment' }
4	3e+06	R&D	Executive Director	2016-03-27	Anagha	{ 'Psychology of body', 'Strength and conditioning' }
3	1e+06	HR	HR	2021-10-10	Manu	{ 'Diversity management' }

(4 rows)

Lab2

Perform the following DB operations using Cassandra.

1. Create a keyspace by name Library
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
3. Insert the values into the table in batch
4. Display the details of the table created and increase the value of the counter
5. Write a query to show that a student with id 112 has taken a book "BDA" 2 times.
6. Export the created column to a csv file
7. Import a given csv dataset from local file system into Cassandra column family

Connected to Test Cluster at 127.0.0.1:9042.

[cqlsh 5.0.1 | Cassandra 3.11.8 | CQL spec 3.4.4 | Native protocol v4]

```
cqlsh:library> create table library_info(stud_id int,
... counter_value counter,
... stud_name text,
... book_name text,
... book_id int,
... date_of_issue date,
... primary key(stud_id,stud_name,book_name,book_id,date_of_issue));
cqlsh:library> describe table library_info;
```

```
CREATE TABLE library.library_info (
  stud_id int,
  stud_name text,
  book_name text,
  book_id int,
  date_of_issue date,
  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.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
  AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassandra.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';
cqlsh:library> update library_info set counter_value=counter_value+1 where stud_id=1 and stud_name='Anagha' and
```

```

book_name='BDA' and book_id=10 and date_of_issue='2022-06-10';
cqlsh:library> update library_info set counter_value=counter_value+1 where stud_id=2 and stud_name='Trisha' and
book_name='OOMB' and book_id=11 and date_of_issue='2022-05-10';
cqlsh:library> update library_info set counter_value=counter_value+1 where stud_id=3 and stud_name='Ramya' and
book_name='ML' and book_id=15 and date_of_issue='2022-05-15';
cqlsh:library> update library_info set counter_value=counter_value+1 where stud_id=4 and stud_name='Tasmiya' and
book_name='Management' and book_id=18 and date_of_issue='2022-05-12';
cqlsh:library> select * from library_info;

```

stud_id	stud_name	book_name	book_id	date_of_issue	counter_value
1	Anagha	BDA	10	2022-06-10	1
2	Trisha	OOMB	11	2022-05-10	1
4	Tasmiya	Management	18	2022-05-12	1
3	Ramya	ML	15	2022-05-15	1

(4 rows)

```

cqlsh:library> update library_info set counter_value=counter_value+1 where stud_id=1 and stud_name='Anagha' and
book_name='BDA' and book_id=10 and date_of_issue='2022-06-10';
cqlsh:library> select * from library_info;

```

stud_id	stud_name	book_name	book_id	date_of_issue	counter_value
1	Anagha	BDA	10	2022-06-10	2
2	Trisha	OOMB	11	2022-05-10	1
4	Tasmiya	Management	18	2022-05-12	1
3	Ramya	ML	15	2022-05-15	1

(4 rows)

```

cqlsh:library> select * from library_info where counter_value=2 allow filtering;

```

stud_id	stud_name	book_name	book_id	date_of_issue	counter_value
1	Anagha	BDA	10	2022-06-10	2

(1 rows)

```

cqlsh:library> copy library_info(stud_id,stud_name,book_id,book_name,date_of_issue,counter_value) to
'Desktop/library_info.csv';
Using 7 child processes

```

Starting copy of library.library_info with columns [stud_id, stud_name, book_id, book_name, date_of_issue, counter_value].

Processed: 4 rows; Rate: 7 rows/s; Avg. rate: 2 rows/s

4 rows exported to 1 files in 2.441 seconds.

```

cqlsh:library> truncate library_info;
cqlsh:library> select * from library_info;

```

stud_id	stud_name	book_name	book_id	date_of_issue	counter_value
---------	-----------	-----------	---------	---------------	---------------

(0 rows)

```
cqlsh:library> copy library_info(stud_id,stud_name,book_id,book_name,date_of_issue,counter_value) from
'Desktop/library_info.csv';
```

Using 7 child processes

Starting copy of library.library_info with columns [stud_id, stud_name, book_id, book_name, date_of_issue, counter_value].

Process ImportProcess-8: 2 rows/s; Avg. rate: 2 rows/s

Processed: 4 rows; Rate: 1 rows/s; Avg. rate: 2 rows/s

4 rows imported from 1 files in 2.507 seconds (0 skipped).

```
cqlsh:library> select * from library_info;
```

stud_id	stud_name	book_name	book_id	date_of_issue	counter_value
1	Anagha	BDA	10	2022-06-10	2
2	Trisha	OOMD	11	2022-05-10	1
4	Tasmiya	Management	18	2022-05-12	1
3	Ramya	ML	15	2022-05-15	1

(4 rows)

Lab 3

MongoDB- CRUD Demonstration

```
use anagha;
db;
show dbs;
db.createCollection("student")

db.student.insert({_id:0,StudName:"Tasmiya",Sem:"VI",Hobbies:"Singing"});
db.student.insert({_id:1,StudName:"Trisha",Sem:"IV",Hobbies:"Carrom"});
db.student.insert({_id:2,StudName:"Anagha",Sem:"V",Hobbies:"Drawing"});
db.student.insert({_id:3,StudName:"Ramya",Sem:"VI",Hobbies:"Violin"});

db.student.find({}, {_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();

db.student.update({_id:1},{ $set: {Hobbies:"Cricket"}},{upsert:true});
db.student.find({}, {_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();

db.student.update({StudName:'Rahul'}, { $set: {Location:'BMS'}},{upsert:true});
db.student.find({}, {_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();

db.student.update({StudName:'Meenakshi'}, { $set: {Location:null}});
db.student.find({}, {_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();

db.student.update({StudName:'Rahul'}, { $unset: {Location:'BMS'}});
db.student.find({}, {_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();

db.student.find({StudName:'Tasmiya'}).pretty();
db.student.find({}, {_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();

db.student.find({Sem:'VI'}).pretty();
db.student.find({}, {_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();

db.student.find({StudName:{$ne:'Ramya'},Sem:{$ne:'V'}}).pretty();
```

```
db.student.find({},{_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();
```

```
db.student.find({Sem:{$lte:'V'}}).pretty();
```

```
db.student.find({},{_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();
```

```
db.student.find({StudName:/^T/}).pretty();
```

```
db.student.find({},{_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();
```

```
db.student.find({StudName:/a$/}).pretty();
```

```
db.student.find({},{_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();
```

```
db.student.find({StudName:/r|R/}).pretty();
```

```
db.student.find({},{_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();
```

```
db.student.find({Hobbies:{$in:['Drawing','Violin']}}).pretty();
```

```
db.student.find({},{_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();
```

```
db.student.find().sort({Sem:1}).pretty();
```

```
db.student.find({},{_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();
```

```
db.student.find().sort({StudName:-1}).pretty();
```

```
db.student.find({},{_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();
```

```
db.student.save({StudName:'Meenakshi',Sem:'TV'});
```

```
db.student.find({},{_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();
```

```
db.student.count();
```

```
db.student.count({Sem:'VI'})
```

```
db.student.find({Sem:'TV'}).limit(1).pretty();
```

```
db.student.find().skip(2).pretty();
```

```
db.student.remove({StudName:'Rahul'}).pretty();
db.student.find({}, {_id:0,StudName:1,Sem:1,Hobbies:1,Location:1}).pretty();

db.student.drop();
```

OUTPUT:

```
switched to db anagha
anagha
admin    0.000GB
config   0.000GB
harryKart 0.000GB
local    0.000GB
school    0.000GB
{ "ok" : 1 }
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
WriteResult({ "nInserted" : 1 })
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Carrom" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
WriteResult({
  "nMatched" : 0,
```

```
"nUpserted" : 1,
"nModified" : 0,
"_id" : ObjectId("62d4c66a9e01af7adee22d59"))
})
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul", "Location" : "BMS" }
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul", "Location" : "BMS" }
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "_id" : 0, "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "_id" : 0, "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "_id" : 3, "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
```

```
{ "StudName" : "Rahul" }
{ "_id" : 0, "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "_id" : 1, "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "_id" : ObjectId("62d4c66a9e01af7adee22d59"), "StudName" : "Rahul" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "_id" : 1, "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "_id" : 2, "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "_id" : 0, "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "_id" : 1, "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "_id" : 0, "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "_id" : 1, "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "_id" : 2, "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "_id" : 3, "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "_id" : 1, "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
```

```
{ "_id" : 3, "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "_id" : ObjectId("62d4c66a9e01af7adee22d59"), "StudName" : "Rahul" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "_id" : 2, "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "_id" : 3, "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "_id" : ObjectId("62d4c66a9e01af7adee22d59"), "StudName" : "Rahul" }
{ "_id" : 1, "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "_id" : 2, "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "_id" : 0, "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "_id" : 3, "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "_id" : 1, "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "_id" : 0, "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "_id" : 3, "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "_id" : ObjectId("62d4c66a9e01af7adee22d59"), "StudName" : "Rahul" }
{ "_id" : 2, "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
```

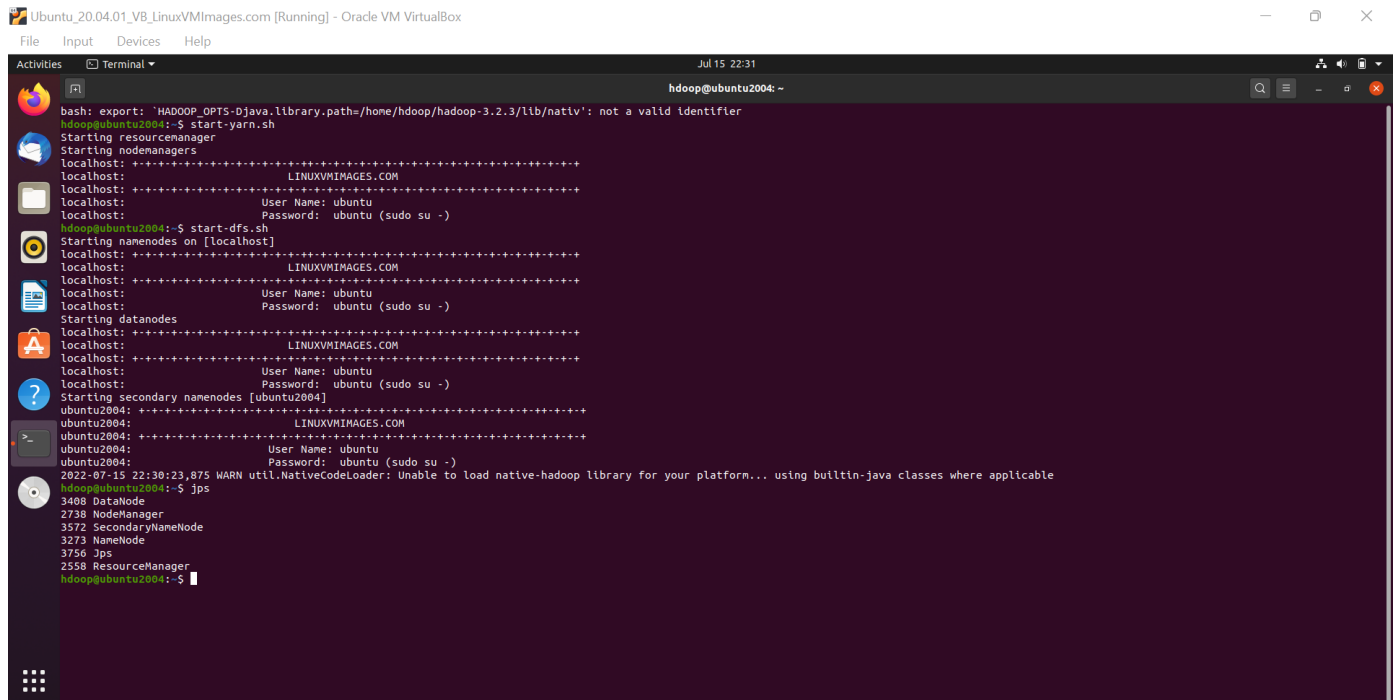
```

{ "StudName" : "Rahul" }
WriteResult({ "nInserted" : 1 })
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Rahul" }
{ "StudName" : "Meenakshi", "Sem" : "IV" }
6
2
{ "_id" : 1, "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "_id" : 2, "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "_id" : 3, "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "_id" : ObjectId("62d4c66a9e01af7adee22d59"), "StudName" : "Rahul" }
{
  "_id" : ObjectId("62d4c66a7a576a20603d9453"),
  "StudName" : "Meenakshi",
  "Sem" : "IV"
}
WriteResult({ "nRemoved" : 1 })
{ "StudName" : "Tasmiya", "Sem" : "VI", "Hobbies" : "Singing" }
{ "StudName" : "Trisha", "Sem" : "IV", "Hobbies" : "Cricket" }
{ "StudName" : "Anagha", "Sem" : "V", "Hobbies" : "Drawing" }
{ "StudName" : "Ramya", "Sem" : "VI", "Hobbies" : "Violin" }
{ "StudName" : "Meenakshi", "Sem" : "IV" }
true

```

Lab 4

Screenshot of Hadoop installation



```
Ubuntu_20.04.01_VB_LinuxVMImages.com [Running] - Oracle VM VirtualBox
File Input Devices Help
Activities Terminal Jul 15 22:31
hadoop@ubuntu2004: ~
bash: export: 'HADOOP_OPTS-Djava.library.path=/home/hadoop/hadoop-3.2.3/lib/native': not a valid identifier
hadoop@ubuntu2004:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
localhost: +-----+
localhost: LINUXVMIMAGES.COM
localhost: +-----+
localhost: User Name: ubuntu
localhost: Password: ubuntu (sudo su -)
hadoop@ubuntu2004:~$ start-dfs.sh
Starting namenodes on [localhost]
localhost: +-----+
localhost: LINUXVMIMAGES.COM
localhost: +-----+
localhost: User Name: ubuntu
localhost: Password: ubuntu (sudo su -)
Starting datanodes
localhost: +-----+
localhost: LINUXVMIMAGES.COM
localhost: +-----+
localhost: User Name: ubuntu
localhost: Password: ubuntu (sudo su -)
Starting secondary namenodes [ubuntu2004]
ubuntu2004: +-----+
ubuntu2004: LINUXVMIMAGES.COM
ubuntu2004: +-----+
ubuntu2004: User Name: ubuntu
ubuntu2004: Password: ubuntu (sudo su -)
2022-07-15 22:30:23,875 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hadoop@ubuntu2004:~$ jps
3408 DataNode
2738 NodeManager
3572 SecondaryNameNode
3273 NameNode
3756 Jps
2558 ResourceManager
hadoop@ubuntu2004:~$
```

Lab 5

Execution of HDFS Commands for interaction with Hadoop Environment. (Minimum 10 commands to be executed)

```
hduser@bmsce-Precision-T1700:~$ start-all.sh
```

This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh

WARNING: An illegal reflective access operation has occurred

WARNING: Illegal reflective access by

org.apache.hadoop.security.authentication.util.KerberosUtil

(file:/usr/local/hadoop/share/hadoop/common/lib/hadoop-auth-2.6.0.jar) to
method sun.security.krb5.Config.getInstance()

WARNING: Please consider reporting this to the maintainers of

org.apache.hadoop.security.authentication.util.KerberosUtil

WARNING: Use --illegal-access=warn to enable warnings of further illegal
reflective access operations

WARNING: All illegal access operations will be denied in a future release

Starting namenodes on [localhost]

```
hduser@localhost's password:
```

```
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hduser-  
namenode-bmsce-Precision-T1700.out
```

```
localhost: WARNING: An illegal reflective access operation has occurred
```

```
localhost: WARNING: Illegal reflective access by
```

```
org.apache.hadoop.security.authentication.util.KerberosUtil
```

```
(file:/usr/local/hadoop/share/hadoop/common/lib/hadoop-auth-2.6.0.jar) to  
method sun.security.krb5.Config.getInstance()
```

```
localhost: WARNING: Please consider reporting this to the maintainers of
```

```
org.apache.hadoop.security.authentication.util.KerberosUtil
```

```
localhost: WARNING: Use --illegal-access=warn to enable warnings of further  
illegal reflective access operations
```

```
localhost: WARNING: All illegal access operations will be denied in a future  
release
```

```
hduser@localhost's password:
```

localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-hduser-datanode-bmsce-Precision-T1700.out

localhost: WARNING: An illegal reflective access operation has occurred

localhost: WARNING: Illegal reflective access by

org.apache.hadoop.security.authentication.util.KerberosUtil

(file:/usr/local/hadoop/share/hadoop/common/lib/hadoop-auth-2.6.0.jar) to
method sun.security.krb5.Config.getInstance()

localhost: WARNING: Please consider reporting this to the maintainers of
org.apache.hadoop.security.authentication.util.KerberosUtil

localhost: WARNING: Use --illegal-access=warn to enable warnings of further
illegal reflective access operations

localhost: WARNING: All illegal access operations will be denied in a future
release

Starting secondary namenodes [0.0.0.0]

hduser@0.0.0.0's password:

0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-hduser-secondarynamenode-bmsce-Precision-T1700.out

0.0.0.0: WARNING: An illegal reflective access operation has occurred

0.0.0.0: WARNING: Illegal reflective access by

org.apache.hadoop.security.authentication.util.KerberosUtil

(file:/usr/local/hadoop/share/hadoop/common/lib/hadoop-auth-2.6.0.jar) to
method sun.security.krb5.Config.getInstance()

0.0.0.0: WARNING: Please consider reporting this to the maintainers of
org.apache.hadoop.security.authentication.util.KerberosUtil

0.0.0.0: WARNING: Use --illegal-access=warn to enable warnings of further
illegal reflective access operations

0.0.0.0: WARNING: All illegal access operations will be denied in a future
release

WARNING: An illegal reflective access operation has occurred

WARNING: Illegal reflective access by

org.apache.hadoop.security.authentication.util.KerberosUtil

(file:/usr/local/hadoop/share/hadoop/common/lib/hadoop-auth-2.6.0.jar) to

method sun.security.krb5.Config.getInstance()

WARNING: Please consider reporting this to the maintainers of
org.apache.hadoop.security.authentication.util.KerberosUtil

WARNING: Use --illegal-access=warn to enable warnings of further illegal
reflective access operations

WARNING: All illegal access operations will be denied in a future release
starting yarn daemons

starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-hduser-
resourcemanager-bmsce-Precision-T1700.out

WARNING: An illegal reflective access operation has occurred

WARNING: Illegal reflective access by
org.apache.hadoop.security.authentication.util.KerberosUtil
(file:/usr/local/hadoop/share/hadoop/common/lib/hadoop-auth-2.6.0.jar) to
method sun.security.krb5.Config.getInstance()

WARNING: Please consider reporting this to the maintainers of
org.apache.hadoop.security.authentication.util.KerberosUtil

WARNING: Use --illegal-access=warn to enable warnings of further illegal
reflective access operations

WARNING: All illegal access operations will be denied in a future release
hduser@localhost's password:

localhost: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-hduser-
nodemanager-bmsce-Precision-T1700.out

localhost: WARNING: An illegal reflective access operation has occurred

localhost: WARNING: Illegal reflective access by
org.apache.hadoop.security.authentication.util.KerberosUtil
(file:/usr/local/hadoop/share/hadoop/common/lib/hadoop-auth-2.6.0.jar) to
method sun.security.krb5.Config.getInstance()

localhost: WARNING: Please consider reporting this to the maintainers of
org.apache.hadoop.security.authentication.util.KerberosUtil

localhost: WARNING: Use --illegal-access=warn to enable warnings of further
illegal reflective access operations

localhost: WARNING: All illegal access operations will be denied in a future

release

hduser@bmsce-Precision-T1700:~\$ jps

8386 NodeManager

7654 DataNode

7879 SecondaryNameNode

7463 NameNode

9143 Jps

8044 ResourceManager

hduser@bmsce-Precision-T1700:~\$ hdfs dfs -mkdir /224new

hduser@bmsce-Precision-T1700:~\$ hadoop fs -ls /

Found 11 items

drwxr-xr-x	- hduser supergroup	0 2022-06-01 10:12 /1bm19cs186
drwxr-xr-x	- hduser supergroup	0 2022-06-04 09:27 /224new
drwxr-xr-x	- hduser supergroup	0 2022-06-03 12:20 /Copy-Secure
drwxr-xr-x	- hduser supergroup	0 2022-06-03 12:06 /Sharan
drwxr-xr-x	- hduser supergroup	0 2022-06-03 14:57 /bda
drwxr-xr-x	- hduser supergroup	0 2022-06-01 09:32 /firstlab
drwxr-xr-x	- hduser supergroup	0 2022-06-01 09:32 /lab
drwxr-xr-x	- hduser supergroup	0 2022-06-01 14:59 /nothing
drwxr-xr-x	- hduser supergroup	0 2022-06-01 15:27 /something
drwxrwxr-x	- hduser supergroup	0 2019-08-01 16:19 /tmp
drwxr-xr-x	- hduser supergroup	0 2019-08-01 16:03 /user

hduser@bmsce-Precision-T1700:~\$ hdfs dfs -put

/home/hduser/Desktop/Welcome.txt /224new/WC.txt

hduser@bmsce-Precision-T1700:~\$ hadoop fs -ls /224new

Found 1 items

```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:33 /224new/WC.txt
```

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /224new/WC.txt
```

Hello! Welcome

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -copyFromLocal
```

```
/home/hduser/Desktop/Welcome.txt /224new/WC1.txt
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /224new
```

Found 2 items

```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:33 /224new/WC.txt
```

```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:37 /224new/WC1.txt
```

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /224new/WC1.txt
```

Hello! Welcome

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -get /224new/WC.txt
```

```
/home/hduser/Downloads/WWC.txt
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /
```

Found 11 items

```
drwxr-xr-x - hduser supergroup      0 2022-06-01 10:12 /bm19cs186
```

```
drwxr-xr-x - hduser supergroup      0 2022-06-04 09:37 /224new
```

```
drwxr-xr-x - hduser supergroup      0 2022-06-03 12:20 /Copy-Secure
```

```
drwxr-xr-x - hduser supergroup      0 2022-06-03 12:06 /Sharan
```

```
drwxr-xr-x - hduser supergroup      0 2022-06-03 14:57 /bda
```

```
drwxr-xr-x - hduser supergroup      0 2022-06-01 09:32 /firstlab
```

```
drwxr-xr-x - hduser supergroup      0 2022-06-01 09:32 /lab
```

```
drwxr-xr-x - hduser supergroup      0 2022-06-01 14:59 /nothing
```

```
drwxr-xr-x - hduser supergroup    0 2022-06-01 15:27 /something
drwxrwxr-x - hduser supergroup    0 2019-08-01 16:19 /tmp
drwxr-xr-x - hduser supergroup    0 2019-08-01 16:03 /user
```

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -getmerge /224new/WC.txt
/224new/WC1.txt /home/hduser/Desktop/Merge.txt
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -getfacl /224new/
# file: /224new
# owner: hduser
# group: supergroup
user::rwx
group::r-x
other::r-x
```

```
hduser@bmsce-Precision-T1700:~$ sudo nano abc.txt
[sudo] password for hduser:
```

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -put /home/hduser/abc.txt
/224new/name.txt
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /224new
Found 3 items
```

```
-rw-r--r--  1 hduser supergroup    15 2022-06-04 09:33 /224new/WC.txt
-rw-r--r--  1 hduser supergroup    15 2022-06-04 09:37 /224new/WC1.txt
-rw-r--r--  1 hduser supergroup    20 2022-06-04 09:51 /224new/name.txt
```

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /224new/name.txt
This is Anagha here!
```

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -copyToLocal /224new/name.txt
/home/hduser/Desktop
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /
```

```
Found 11 items
```

```
drwxr-xr-x - hduser supergroup      0 2022-06-01 10:12 /1bm19cs186
drwxr-xr-x - hduser supergroup      0 2022-06-04 09:51 /224new
drwxr-xr-x - hduser supergroup      0 2022-06-03 12:20 /Copy-Secure
drwxr-xr-x - hduser supergroup      0 2022-06-03 12:06 /Sharan
drwxr-xr-x - hduser supergroup      0 2022-06-03 14:57 /bda
drwxr-xr-x - hduser supergroup      0 2022-06-01 09:32 /firstlab
drwxr-xr-x - hduser supergroup      0 2022-06-01 09:32 /lab
drwxr-xr-x - hduser supergroup      0 2022-06-01 14:59 /nothing
drwxr-xr-x - hduser supergroup      0 2022-06-01 15:27 /something
drwxrwxr-x - hduser supergroup      0 2019-08-01 16:19 /tmp
drwxr-xr-x - hduser supergroup      0 2019-08-01 16:03 /user
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -mv /224new /224newer
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /224newer
```

```
Found 3 items
```

```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:33 /224newer/WC.txt
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:37 /224newer/WC1.txt
-rw-r--r-- 1 hduser supergroup      20 2022-06-04 09:51 /224newer/name.txt
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -cp /224newer/ /224new
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /224new
```

```
Found 3 items
```

```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:58 /224new/WC.txt
```



```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:58 /224new/WC1.txt
-rw-r--r-- 1 hduser supergroup      20 2022-06-04 09:58 /224new/name.txt
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /224newer
```

```
Found 3 items
```

```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:33 /224newer/WC.txt
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:37 /224newer/WC1.txt
-rw-r--r-- 1 hduser supergroup      20 2022-06-04 09:51 /224newer/name.txt
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -cp /224newer/name.txt
```

```
/224new
```

```
cp: `/224new/name.txt': File exists
```

```
hduser@bmsce-Precision-T1700:~$ sudo nano hello.txt
```

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -put /home/hduser/hello.txt
```

```
/224newer/hello.txt
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /224newer
```

```
Found 4 items
```

```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:33 /224newer/WC.txt
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:37 /224newer/WC1.txt
-rw-r--r-- 1 hduser supergroup      13 2022-06-04 10:02 /224newer/hello.txt
-rw-r--r-- 1 hduser supergroup      20 2022-06-04 09:51 /224newer/name.txt
```

```
hduser@bmsce-Precision-T1700:~$ hdfs dfs -cat /224newer/hello.txt
```

```
hi hello bye
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -cp /224newer/hello.txt /224new
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /224newer
```

```
Found 4 items
```

```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:33 /224newer/WC.txt
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:37 /224newer/WC1.txt
-rw-r--r-- 1 hduser supergroup      13 2022-06-04 10:02 /224newer/hello.txt
-rw-r--r-- 1 hduser supergroup      20 2022-06-04 09:51 /224newer/name.txt
```

```
hduser@bmsce-Precision-T1700:~$ hadoop fs -ls /224new
```

```
Found 4 items
```

```
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:58 /224new/WC.txt
-rw-r--r-- 1 hduser supergroup      15 2022-06-04 09:58 /224new/WC1.txt
-rw-r--r-- 1 hduser supergroup      13 2022-06-04 10:03 /224new/hello.txt
-rw-r--r-- 1 hduser supergroup      20 2022-06-04 09:58 /224new/name.txt
```

```
hduser@bmsce-Precision-T1700:~$
```

Lab 6

Create a Map Reduce program to a) find average temperature for each year from NCDC data set.
b) find the mean max temperature for every month

```
// AverageDriver.java package temperature;
```

```
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);
    }
}
```

```
//AverageMapper.java package temperature;
```

```
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 package temperature;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
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));
    }
}

```



```

c:\hadoop_new\sbin>hdfs dfs -cat /tempAverageOutput/part-r-00000
1901    46
1949    94
1950     3

```

```

//TempDriver.java package temperatureMax;

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 TempDriver
{
    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(TempDriver.class);
    job.setJobName("Max temperature");
    FileInputFormat.addInputPath(job,new Path(args[0]));
    FileOutputFormat.setOutputPath(job,new Path (args[1]));

    job.setMapperClass(TempMapper.class);
    job.setReducerClass(TempReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    System.exit(job.waitForCompletion(true)?0:1);
}
}

```

//TempMapper.java package temperatureMax;

```

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

```

```

public class TempMapper 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 month = line.substring(19,21);
        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(month),new IntWritable(temperature));
}
}

//TempReducer.java package temperatureMax;

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

public class TempMapper 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 month = line.substring(19,21);
    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(month),new IntWritable(temperature));
    }
}

```

```
c:\hadoop_new\sbin>hdfs dfs -cat /tempMaxOutput/part-r-00000
01      44
02      17
03     111
04     194
05     256
06     278
07     317
08     283
09     211
10     156
11      89
12     117
```

Lab 7

For a given Text file, create a Map Reduce program to sort the content in an alphabetic order listing only top 'n' maximum occurrence of words.

```
// TopN.java package sortWords;
```

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import org.apache.hadoop.util.MiscUtils;
```

```
import java.io.IOException; import java.util.*;
```

```
public class TopN {
```

```
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
        if (otherArgs.length != 2) {
            System.err.println("Usage: TopN<in><out>");
            System.exit(2);
        }
```

```
        Job job = Job.getInstance(conf);
        job.setJobName("Top N");
        job.setJarByClass(TopN.class);
        job.setMapperClass(TopNMapper.class);    //job.setCombinerClass(TopNReducer.class);
        job.setReducerClass(TopNReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
```

```
    /**
     * The mapper reads one line at the time, splits it into an array of single words and emits every * word to the reducers
     with the value of 1.
     */
```

```
    public static class TopNMapper extends Mapper<Object, Text, Text, IntWritable> {
```

```
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();
```



```
private String tokens = "[_!$#<>\\^=\\[\\]\\*\\/\\|\\|,;.:()?!\"']";
```

```
@Override
```

```
public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
```

```
String cleanLine = value.toString().toLowerCase().replaceAll(tokens, " ");
```

```
StringTokenizer itr = new StringTokenizer(cleanLine);
```

```
while (itr.hasMoreTokens()) {
```

```
word.set(itr.nextToken().trim());
```

```
context.write(word, one);
```

```
}
```

```
}
```

```
}
```

```
/**
```

```
* The reducer retrieves every word and puts it into a Map: if the word already exists in the * map, increments its value, otherwise sets it to 1.
```

```
*/
```

```
public static class TopNReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
```

```
private Map<Text, IntWritable> countMap = new HashMap<>();
```

```
@Override
```

```
public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {
```

```
// computes the number of occurrences of a single word
```

```
int sum = 0;
```

```
for (IntWritable val : values) {
```

```
sum += val.get();
```

```
}
```

```
// puts the number of occurrences of this word into the map.
```

```
// We need to create another Text object because the Text instance
```

```
// we receive is the same for all the words
```

```
countMap.put(new Text(key), new IntWritable(sum));
```

```
}
```

```
@Override protected void cleanup(Context context) throws IOException, InterruptedException {
```

```
Map<Text, IntWritable> sortedMap = MiscUtils.sortByValues(countMap);
```

```
int counter = 0;
```

```
for (Text key : sortedMap.keySet()) {
```

```
if (counter++ == 3) {
```

```
break;
```

```
}
```

```
context.write(key, sortedMap.get(key));
```

```
}
```

```
}
```

```
}
```

```
/**
```

```
* The combiner retrieves every word and puts it into a Map: if the word already exists in the * map, increments its value, otherwise sets it to 1. */
```

```

public static class TopNCombiner extends Reducer<Text, IntWritable, Text, IntWritable> {

    @Override public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,
    InterruptedException {

        // computes the number of occurrences of a single word
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        context.write(key, new IntWritable(sum));
    }
}

// MiscUtils.java package utils;

import java.util.*;

public class MiscUtils {

    /**
    sorts the map by values. Taken from:
    http://javarevisited.blogspot.it/2012/12/how-to-sort-hashmap-java-by-key-and-value.html */
    public static <K extends Comparable, V extends Comparable> Map<K, V> sortByValues(Map<K, V> map) {
        List<Map.Entry<K, V>> entries = new LinkedList<Map.Entry<K, V>>(map.entrySet());
        Collections.sort(entries, new Comparator<Map.Entry<K, V>>() {
            @Override public int compare(Map.Entry<K, V> o1, Map.Entry<K, V> o2) {
                return o2.getValue().compareTo(o1.getValue());
            }
        });

        //LinkedHashMap will keep the keys in the order they are inserted
        //which is currently sorted on natural ordering
        Map<K, V> sortedMap = new LinkedHashMap<K, V>();
        for (Map.Entry<K, V> entry : entries) {
            sortedMap.put(entry.getKey(), entry.getValue());
        }
        return sortedMap;
    }
}

```

```

C:\hadoop_new\share\hadoop\mapreduce>hdfs dfs -cat \sortwordsOutput\part-r-00000
car      7
deer     6
bear     3

```

Lab 8

Create a Hadoop Map Reduce program to combine information from the users file along with Information from the posts file by using the concept of join and display user_id, Reputation and Score.

```
// JoinDriver.java
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.mapred.libMultipleInputs;
import org.apache.hadoop.util.*;
public class JoinDriver extends Configured implements Tool {
    public static class KeyPartitioner implements Partitioner<TextPair, Text> {
        @Override

        public void configure(JobConf job) {}
        @Override
        public int getPartition(TextPair key, Text value, int numPartitions) {
            return (key.getFirst().hashCode() & Integer.MAX_VALUE) %
                numPartitions;
        }
    }
    @Override
    public int run(String[] args) throws Exception {
        if (args.length != 3) {
            System.out.println("<Usage: <Department Emp Strength input>
            <Department Name input> <output>");
            return -1;
        }
        JobConf conf = new JobConf(getConf(), getClass());
        conf.setJobName("<Join &#39;Department Emp Strength input&#39; with &#39;Department Name
        input&#39;>");
        Path AInputPath = new Path(args[0]);
        Path BInputPath = new Path(args[1]);
        Path outputPath = new Path(args[2]);
        MultipleInputs.addInputPath(conf, AInputPath, TextInputFormat.class,
            Posts.class);
        MultipleInputs.addInputPath(conf, BInputPath, TextInputFormat.class,
            User.class);
        FileOutputFormat.setOutputPath(conf, outputPath);
        conf.setPartitionerClass(KeyPartitioner.class);
        conf.setOutputValueGroupingComparator(TextPair.FirstComparator.class);
        conf.setMapOutputKeyClass(TextPair.class);
        conf.setReducerClass(JoinReducer.class);
        conf.setOutputKeyClass(Text.class);

        JobClient.runJob(conf);
        return 0;
    }
}
```

```

}
public static void main(String[] args) throws Exception {
    int exitCode = ToolRunner.run(new JoinDriver(), args);
    System.exit(exitCode);
}
}

// JoinReducer.java
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.*;
public class JoinReducer extends MapReduceBase implements Reducer<TextPair, Text,
Text,
Text>; {
    @Override
    public void reduce (TextPair key, Iterator<Text> values, OutputCollector<Text, Text>
output, Reporter reporter)
        throws IOException
    {
        Text nodeId = new Text(values.next());
        while (values.hasNext()) {
            Text node = values.next();
            Text outValue = new Text(nodeId.toString() + "<td>" + node.toString());
            output.collect(key.getFirst(), outValue);
        }
    }
}

// User.java
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FSDataInputStream;
import org.apache.hadoop.fs.FSDataOutputStream;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.io.IntWritable;
public class User extends MapReduceBase implements Mapper<LongWritable, Text,
TextPair,
Text>; {
    @Override
    public void map(LongWritable key, Text value, OutputCollector<TextPair, Text> output,
Reporter reporter)
        throws IOException
    {
        String valueString = value.toString();
        String[] SingleNodeData = valueString.split("<td>");
    }
}

```

```

output.collect(new TextPair(SingleNodeData[0], &quot;1&quot;), new
Text(SingleNodeData[1]));
}
}
//Posts.java
import java.io.IOException;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
public class Posts extends MapReduceBase implements Mapper<LongWritable, Text,
TextPair,
Text>; {
@Override
public void map(LongWritable key, Text value, OutputCollector<TextPair, Text> output,
Reporter reporter)
throws IOException
{
String valueString = value.toString();
String[] SingleNodeData = valueString.split(&quot;\t&quot;);
output.collect(new TextPair(SingleNodeData[3], &quot;0&quot;), new
Text(SingleNodeData[9]));
}
}

```

```

// TextPair.java
import java.io.*;
import org.apache.hadoop.io.*;
public class TextPair implements WritableComparable<TextPair>; {
private Text first;
private Text second;
public TextPair() {
set(new Text(), new Text());
}
public TextPair(String first, String second) {
set(new Text(first), new Text(second));
}
public TextPair(Text first, Text second) {
set(first, second);
}
public void set(Text first, Text second) {
this.first = first;
this.second = second;
}
public Text getFirst() {
return first;
}
public Text getSecond() {
return second;
}
@Override
public void write(DataOutput out) throws IOException {

```

```

first.write(out);
second.write(out);
}
@Override
public void readFields(DataInput in) throws IOException {
first.readFields(in);
second.readFields(in);
}
@Override
public int hashCode() {

return first.hashCode() * 163 + second.hashCode();
}
@Override
public boolean equals(Object o) {
if (o instanceof TextPair) {
TextPair tp = (TextPair) o;
return first.equals(tp.first) && second.equals(tp.second);
}
return false;
}
@Override
public String toString() {
return first + " " + second;
}
@Override
public int compareTo(TextPair tp) {
int cmp = first.compareTo(tp.first);
if (cmp != 0) {
return cmp;
}
return second.compareTo(tp.second);
}
// ^^ TextPair
// vv TextPairComparator
public static class Comparator extends WritableComparator {
private static final Text.Comparator TEXT_COMPARATOR = new Text.Comparator();
public Comparator() {
super(TextPair.class);
}
@Override
public int compare(byte[] b1, int s1, int l1,
byte[] b2, int s2, int l2) {
try {
int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) + readVInt(b1, s1);
int firstL2 = WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2);
int cmp = TEXT_COMPARATOR.compare(b1, s1, firstL1, b2, s2, firstL2);
if (cmp != 0) {
return cmp;
}
}
}

```

```

return TEXT_COMPARATOR.compare(b1, s1 + firstL1, l1 - firstL1,
b2, s2 + firstL2, l2 - firstL2);
} catch (IOException e) {
throw new IllegalArgumentException(e);
}
}
}
static {
WritableComparator.define(TextPair.class, new Comparator());
}
public static class FirstComparator extends WritableComparator {
private static final Text.Comparator TEXT_COMPARATOR = new Text.Comparator();
public FirstComparator() {
super(TextPair.class);
}
@Override
public int compare(byte[] b1, int s1, int l1,
byte[] b2, int s2, int l2) {
try {
int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) + readVInt(b1, s1);
int firstL2 = WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2);
return TEXT_COMPARATOR.compare(b1, s1, firstL1, b2, s2, firstL2);
} catch (IOException e) {
throw new IllegalArgumentException(e);
}
}
@Override
public int compare(WritableComparable a, WritableComparable b) {
if (a instanceof TextPair && b instanceof TextPair) {
return ((TextPair) a).first.compareTo(((TextPair) b).first);
}
return super.compare(a, b);
}
} }

```

```

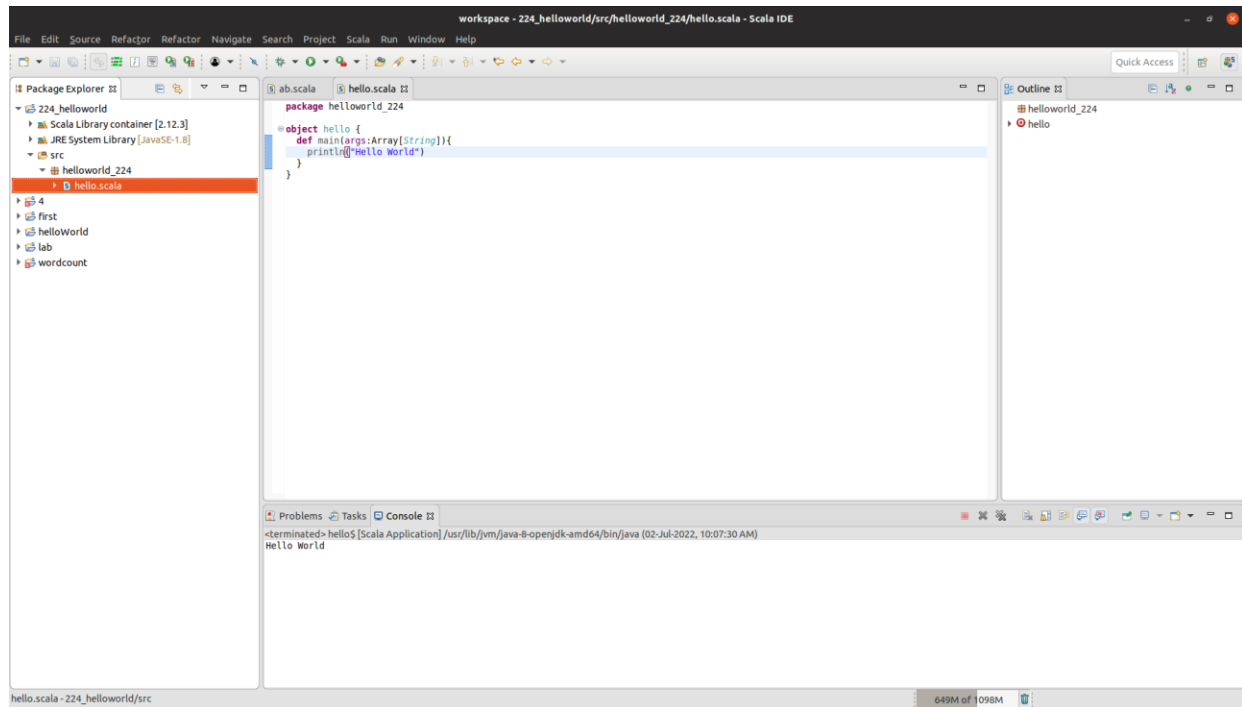
c:\hadoop_new\share\hadoop\mapreduce>hdfs dfs -cat \joinOutput\part-00000
"100005361"      "2"      "36134"
"100018705"      "2"      "76"
"100022094"      "0"      "6354"

```

Lab 9

Program to print word count on scala shell and print “Hello world” on scala IDE

```
(base) bmsce@bmsce-Precision-T1700:~$ spark-shell
scala>println("Hello World!");
Hello World!
```



Lab 10

Using RDD and Flat Map count how many times each word appears in a file and write out a list of words whose count is strictly greater than 4 using Spark

```
scala> val textFile=sc.textFile("/home/bmsce/Desktop/sparkdata.txt")
```

```
textFile: org.apache.spark.rdd.RDD[String] = /home/bmsce/Desktop/sparkdata.txt MapPartitionsRDD[6] at textFile at <console>:24
```

```
scala> val counts=textFile.flatMap(line=>line.split(" ")).map(word=>(word,1)).reduceByKey(_+_);  
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[9] at reduceByKey at <console>:25
```

```
scala> import scala.collection.immutable.ListMap;  
import scala.collection.immutable.ListMap
```

```
scala> val sorted=ListMap(counts.collect.sortWith(_._2>_._2):_*)  
sorted: scala.collection.immutable.ListMap[String,Int] = Map(bms -> 5, college -> 4, of -> 2, university -> 1, evening -> 1, women's -> 1, technological -> 1, engineering -> 1, architecture -> 1, id -> 1, visweswariah -> 1)
```

```
scala> println(sorted)  
Map(bms -> 5, college -> 4, of -> 2, university -> 1, evening -> 1, women's -> 1, technological -> 1, engineering -> 1, architecture -> 1, id -> 1, visweswariah -> 1)
```

```
scala> for((k,v)<-sorted)
```

```
| {  
| if(v>4)  
| {  
| print(k+",")  
| print(v)  
| println()  
| }  
| }
```

```
bms,5
```

