

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT
on

BIG DATA ANALYTICS **(20CS6PEBDA)**

Submitted by

PUNEETH K (1BM19CS125)

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 **PUNEETH K(1BM19CS125)**, who is bonafide student of **B. M. S. College of Engineering**. It is in partial fulfillment 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.

Name of the Lab-In charge
Designation
Department of CSE
BMSCE, Bengaluru

ANTARA ROY CHOUDHURY
Assistant Professor
Department of CSE
BMSCE, Bengaluru

INDEX

Sl. No.	Experiment Title	Page No.
1	Cassandra Lab Program 1: - Student Database	5-7
2	Cassandra Lab Program 2: - Library Database	8-9
3	MongoDB- CRUD Demonstration	10-21
4	Hadoop Installation	22
5	Hadoop Commands	23
7	Hadoop Programs: Top N	24-30
8	Hadoop Programs: Average Temperature	31-37
9	Hadoop Programs: Join	38-45
10	Scala Programs: Word Count	46
11	Scala Programs: Word Count greater than 4	47-48

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

LAB 1 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.

1. Create a key space by name Employee

```
cqlsh> create keyspace LAB1_Employee with replication = { 'class':'SimpleStrategy','replication_factor':1};
cqlsh> use LAB1_Employee;
cqlsh:lab1_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

```
cqlsh:lab1_employee> create table Employee_info(Emp_id int ,Emp_name text ,Designation text ,Date_of_joining timestamp,Salary double,Dept_name text,primary key(Emp_id));
cqlsh:lab1_employee> |
```

3. Insert the values into the table in batch

```
cqlsh:lab1_employee> begin batch insert into employee_info(Emp_id,Emp_name,Designation,Date_of_joining,Salary,Dept_name)values(11,'Pankaj','Senior_Developer','2022-05-12',4500000,'Developing') insert into employee_info(Emp_id,Emp_name,Designation,Date_of_joining,Salary,Dept_name)values(12,'Preetham','Manager','2022-05-13',6500000,'Developing') insert into employee_info(Emp_id,Emp_name,Designation,Date_of_joining,Salary,Dept_name)values(13,'Prithvi','CEO','2012-05-13',8500000,'Overall') apply batch;
cqlsh:lab1_employee> select * from employee_info;
```

emp_id	date_of_joining	dept_name	designation	emp_name	salary
13	2012-05-12 18:30:00.000000+0000	Overall	CEO	Prithvi	8.5e+06
11	2022-05-11 18:30:00.000000+0000	Developing	Senior_Developer	Pankaj	4.5e+06
12	2022-05-12 18:30:00.000000+0000	Developing	Manager	Preetham	6.5e+06

(3 rows)

```
cqlsh:lab1_employee> |
```

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

```
cqlsh:lab1_employee> update employee_info set Emp_name='Puneeth' ,Dept_name='Sales' where Emp_id=13;  
cqlsh:lab1_employee> select * from employee_info;
```

emp_id	date_of_joining	dept_name	designation	emp_name	salary
13	2012-05-12 18:30:00.000000+0000	Sales	CEO	Puneeth	8.5e+06
11	2022-05-11 18:30:00.000000+0000	Developing	Senior_Developer	Pankaj	4.5e+06
12	2022-05-12 18:30:00.000000+0000	Developing	Manager	Preetham	6.5e+06

(3 rows)

5. Sort the details of Employee records based on salary

```
cqlsh:lab1_employee> begin batch  
... insert into emp(id,salary,name)values(5,45000,'Pankaj')  
... insert into emp(id,salary,name)values(7,455000,'Preetham')  
... insert into emp(id,salary,name)values(9,55000,'ram')  
... apply batch;  
cqlsh:lab1_employee> select * from emp;
```

id	salary	name
5	45000	Pankaj
7	4.55e+05	Preetham
9	55000	ram

(3 rows)

```
cqlsh:lab1_employee> paging off;  
Disabled Query paging.  
cqlsh:lab1_employee> select * from emp where id in (5,7,9) order by salary;
```

id	salary	name
5	45000	Pankaj
9	55000	ram
7	4.55e+05	Preetham

(3 rows)

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.

```
cqlsh:lab1_employee> alter table employee_info add projects text;  
cqlsh:lab1_employee> select * from employee_info;
```

emp_id	date_of_joining	dept_name	designation	emp_name	projects	salary
13	2012-05-12 18:30:00.000000+0000	Sales	CEO	Puneeth	null	8.5e+06
11	2022-05-11 18:30:00.000000+0000	Developing	Senior_Developer	Pankaj	null	4.5e+06
12	2022-05-12 18:30:00.000000+0000	Developing	Manager	Preetham	null	6.5e+06

(3 rows)

7. Update the altered table to add project names.

```
cqlsh:lab1_employee> update Employee_info set projects='Kubernetes' where Emp_id=11;
cqlsh:lab1_employee> update Employee_info set projects='node_js' where Emp_id=12;
cqlsh:lab1_employee> update Employee_info set projects='Mobile_app' where Emp_id=13;
cqlsh:lab1_employee> select * from employee_info;
```

emp_id	date_of_joining	dept_name	designation	emp_name	projects	salary
13	2012-05-12 18:30:00.000000+0000	Sales	CEO	Puneeth	Mobile_app	8.5e+06
11	2022-05-11 18:30:00.000000+0000	Developing	Senior_Developer	Pankaj	Kubernetes	4.5e+06
12	2022-05-12 18:30:00.000000+0000	Developing	Manager	Preetham	node_js	6.5e+06

(3 rows)

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

```
cqlsh:lab1_employee> insert into Employee_info (Emp_id,Emp_name,Designation,Date_of_joining,Salary,Dept_name)values(19,'Prithvi','Senior_Developer','2022-08-12',400000,'Developing') using TTL 50;
cqlsh:lab1_employee> select TTL(emp_name) from Employee_info where Emp_id=19;
```

```
ttl(emp_name)
-----
45
```

LAB 2 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

1 Create a key space by name Library

```
cqlsh> create keyspace lab2_library with replication={'class':'SimpleStrategy','replication_factor':1};
cqlsh> use lab2_library;
cqlsh:lab2_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

```
cqlsh:lab2_library> create table library_info(stud_id int,counter_value counter,stud_name text,book_id int
,date_of_issue timestamp,primary key(stud_id,stud_name,book_id,date_of_issue));
cqlsh:lab2_library> A
```

3. Insert the values into the table in batch

```
cqlsh:lab2_library> update library_info set counter_value=counter_value + 2 where stud_id=2 and stud_name=
'Pankaj' and book_id=145 and date_of_issue='2022-08-04';
cqlsh:lab2_library> select * from library_info;
```

stud_id	stud_name	book_id	date_of_issue	counter_value
2	Pankaj	145	2022-08-03 18:30:00.000000+0000	4

4. Display the details of the table created and increase the value of the counter

```
cqlsh:lab2_library> update library_info set counter_value=counter_value + 2 where stud_id=2 and stud_name=
'Pankaj' and book_id=145 and date_of_issue='2022-08-04';
cqlsh:lab2_library> select * from library_info;
```

stud_id	stud_name	book_id	date_of_issue	counter_value
2	Pankaj	145	2022-08-03 18:30:00.000000+0000	4

5. Write a query to show that a student with id 112 has taken a book “BDA” 2 times.

```
cqlsh:lab2_library> update library_info set counter_value=counter_value + 2 where stud_id=112 and stud_name='Preetham' and book_id=145 and date_of_issue='2022-08-04';
cqlsh:lab2_library> select counter_value from library_info where stud_id=112;
```

counter_value
2

6. Export the created column to a csv file

```
cqlsh:lab2_library> copy library_info(stud_id,stud_name,book_id,date_of_issue,counter_value)to 'lib.csv';
Using 7 child processes

Starting copy of lab2_library.library_info with columns [stud_id, stud_name, book_id, date_of_issue, counter_value].
Processed: 2 rows; Rate:      9 rows/s; Avg. rate:      9 rows/s
2 rows exported to 1 files in 0.250 seconds.
```

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

```
cqlsh:lab2_library> create table library_info2(stud_id int,counter_value counter,stud_name text,book_id int,date_of_issue timestamp,primary key(stud_id,stud_name,book_id,date_of_issue));
cqlsh:lab2_library> copy library_info2(stud_id,stud_name,book_id,date_of_issue,counter_value)from 'lib.csv';
Using 7 child processes

Starting copy of lab2_library.library_info2 with columns [stud_id, stud_name, book_id, date_of_issue, counter_value].
Processed: 2 rows; Rate:      4 rows/s; Avg. rate:      6 rows/s
2 rows imported from 1 files in 0.356 seconds (0 skipped).
cqlsh:lab2_library> select * from library_info;
```

stud_id	stud_name	book_id	date_of_issue	counter_value
2	Pankaj	145	2022-08-03 18:30:00.000000+0000	4
112	Preetham	145	2022-08-03 18:30:00.000000+0000	2

(2 rows)

```
cqlsh:lab2_library> select * from library_info2;
```

stud_id	stud_name	book_id	date_of_issue	counter_value
2	Pankaj	145	2022-08-03 18:30:00.000000+0000	4
112	Preetham	145	2022-08-03 18:30:00.000000+0000	2

```
cqlsh:lab2_library>
```

LAB 3 MongoDB- CRUD Demonstration

I. CREATE DATABASE IN MONGODB.

use myDB; db; (Confirm the

existence of your database)

show dbs; (To list all databases)

```
Command Prompt - mongo
Microsoft Windows [Version 10.0.22000.675]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>mongo
MongoDB shell version v5.0.9
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("484a3dd6-af99-4170-a440-b1c0987ab04e") }
MongoDB server version: 5.0.9
=====
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility. The "mongo" shell has been deprecated and will be removed in
an upcoming release.
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
=====
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
https://community.mongodb.com
---
The server generated these startup warnings when booting:
  2022-06-03T06:17:24.092+05:30: Access control is not enabled for the database. Read and write access to data a
nd configuration is unrestricted
---
---
  Enable MongoDB's free cloud-based monitoring service, which will then receive and display
  metrics about your deployment (disk utilization, CPU, operation statistics, etc).

  The monitoring data will be available on a MongoDB website with a unique URL accessible to you
  and anyone you share the URL with. MongoDB may use this information to make product
  improvements and to suggest MongoDB products and deployment options to you.

  To enable free monitoring, run the following command: db.enableFreeMonitoring()
  To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> show dbs
admin    0.000GB
config  0.000GB
local    0.000GB
> use myDB;
switched to db myDB
> db;
myDB
> show dbs;
admin    0.000GB
config  0.000GB
local    0.000GB
> _
```

II. CRUD (CREATE, READ, UPDATE, DELETE) OPERATIONS

1. To create a collection by the name “Student”. Let us take a look at the collection list prior to the creation of the new collection “Student”.

```
db.createCollection("Student"); => sql equivalent CREATE TABLE STUDENT(...);
```

2. To drop a collection by the name “Student”.

```
db.Student.drop();
```

3. Create a collection by the name “Students” and store the following data in it.

```
db.Student.insert({_id:1,StudName:"MichelleJacintha",Grade:"VII",Hobbies:"InternetSurfing"});
```

4. Insert the document for “AryanDavid” in to the Students collection only if it does not already exist in the collection. However, if it is already present in the collection, then update the document with new values. (Update his Hobbies from “Skating” to “Chess”.) Use “Update else insert” (if there is an existing document, it will attempt to update it, if there is no existing document then it will insert it).

```
db.Student.update({_id:3,StudName:"AryanDavid",Grade:"VII"},{$set:{Hobbies:"Skating"}},{upsert:true});
```

```
local 0.000GB
> db.createCollection("Student");
{ "ok" : 1 }
> db.Student.drop();
true
> db.createCollection("Student");
{ "ok" : 1 }
> db.Student.insert({_id:1, StudName:"MichelleJacintha", Grade:"VII", Hobbies:"InternetSurfing"});
WriteResult({ "nInserted" : 1 })
> db.Student.insert({_id:1, StudName:"MichelleJacintha", Grade:"VII", Hobbies:"InternetSurfing"});
WriteResult({
  "nInserted" : 0,
  "writeError" : {
    "code" : 11000,
    "errmsg" : "E11000 duplicate key error collection: myDB.Student index: _id_ dup key: { _id: 1.0 }"
  }
})
> db.Student.updateelseinsert({_id:3, StudName:"AryanDavid", Grade:"VII"},{$set:{Hobbies:"Skating"}},{upsert:true});
uncaught exception: TypeError: db.Student.updateelseinsert is not a function :
@(shell):1:1
> db.Student.update({_id:3, StudName:"AryanDavid", Grade:"VII"},{$set:{Hobbies:"Skating"}},{upsert:true});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 3 })
>
```

```
Command Prompt - mongo
> show collections
Student
> db.Student.find();
{ "_id" : 1, "StudName" : "MichelleJacintha", "Grade" : "VII", "Hobbies" : "InternetSurfing" }
{ "_id" : 3, "Grade" : "VII", "StudName" : "AryanDavid", "Hobbies" : "Skating" }
>
```

5. FIND METHOD

A. To search for documents from the “Students” collection based on certain search criteria.

```
db.Student.find({StudName:"Aryan David"});
```

```
({cond..},{columns.. column:1, columnname:0} )
```

```
> db.Student.find({StudName:"AryanDavid"});
{ "_id" : 3, "Grade" : "VII", "StudName" : "AryanDavid", "Hobbies" : "Skating" }
>
```

B. To display only the StudName and Grade from all the documents of the Students collection. The identifier _id should be suppressed and NOT displayed.

```
db.Student.find({}, {StudName:1, Grade:1, _id:0});
```

```
Command Prompt - mongo
> db.Student.find({}, {StudName:1, Grade:1, _id:0});
{ "StudName" : "MichelleJacintha", "Grade" : "VII" }
{ "Grade" : "VII", "StudName" : "AryanDavid" }
>
```

C. To find those documents where the Grade is set to ‘VII’

```
db.Student.find({Grade:{$eq:"VII"}}).pretty();
```

```
Command Prompt - mongo
> db.Student.find({Grade:{$eq:'VII'}}).pretty();
{
  "_id" : 1,
  "StudName" : "MichelleJacintha",
  "Grade" : "VII",
  "Hobbies" : "InternetSurfing"
}
{
  "_id" : 3,
  "Grade" : "VII",
  "StudName" : "AryanDavid",
  "Hobbies" : "Skating"
}
```

D. To find those documents from the Students collection where the Hobbies is set to either 'Chess' or is set to 'Skating'. `db.Student.find({Hobbies : { $in: ['Chess', 'Skating'] } }).pretty ();`

```
Command Prompt - mongo
> db.Student.find({Hobbies:{$in: ['Chess','Skating']}}).pretty();
{
  "_id" : 3,
  "Grade" : "VII",
  "StudName" : "AryanDavid",
  "Hobbies" : "Skating"
}
```

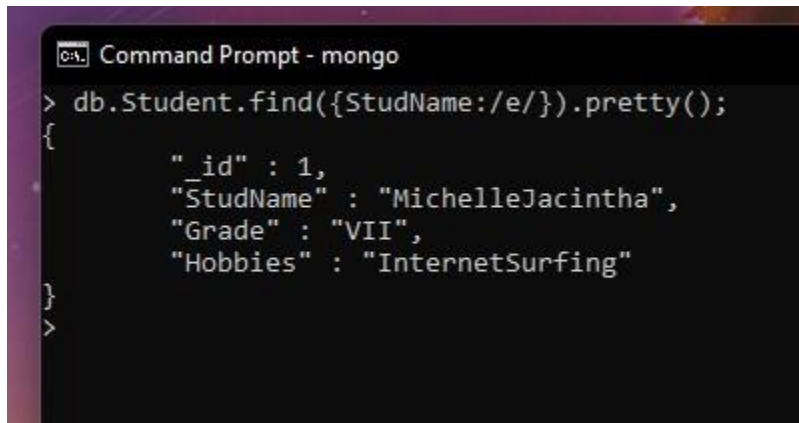
E. To find documents from the Students collection where the StudName begins with "M". `db.Student.find({StudName:/^M/}).pretty();`

```
Command Prompt - mongo
> db.Student.find({StudName:/^M/}).pretty();
{
  "_id" : 1,
  "StudName" : "MichelleJacintha",
  "Grade" : "VII",
  "Hobbies" : "InternetSurfing"
}
```

F. To find documents from the

Students collection where the StudName has an "e" in any position.

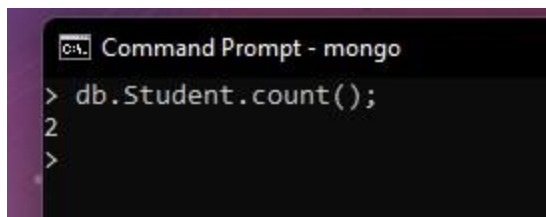
```
db.Student.find({StudName:/e/}).pretty();
```

A screenshot of a MongoDB Command Prompt window titled "Command Prompt - mongo". The prompt shows the command `> db.Student.find({StudName:/e/}).pretty();` and its output, which is a JSON document: `{ "_id" : 1, "StudName" : "MichelleJacintha", "Grade" : "VII", "Hobbies" : "InternetSurfing" }`.

```
Command Prompt - mongo
> db.Student.find({StudName:/e/}).pretty();
{
  "_id" : 1,
  "StudName" : "MichelleJacintha",
  "Grade" : "VII",
  "Hobbies" : "InternetSurfing"
}
```

G. To find the number of documents in the Students collection.

```
db.Student.count();
```

A screenshot of a MongoDB Command Prompt window titled "Command Prompt - mongo". The prompt shows the command `> db.Student.count();` and its output, which is the number `2`.

```
Command Prompt - mongo
> db.Student.count();
2
```

H. To sort the documents from the Students collection in the descending order of StudName. `db.Student.find().sort({StudName:-1}).pretty();`


```
Command Prompt - mongo
> db.Student.find().sort({StudNam:-1}).pretty();
{
  "_id" : 1,
  "StudName" : "MichelleJacintha",
  "Grade" : "VII",
  "Hobbies" : "InternetSurfing"
}
{
  "_id" : 3,
  "Grade" : "VII",
  "StudName" : "AryanDavid",
  "Hobbies" : "Skating"
}
>
```

III. Import data from a CSV file

Given a CSV file “sample.txt” in the D:drive, import the file into the MongoDB collection, “SampleJSON”. The collection is in the database “test”.

```
mongoimport --db Student --collection airlines --type csv --headerline --file
/home/hduser/Desktop/airline.csv
```

```
Command Prompt
C:\Program Files\MongoDB\Server\5.0\bin>mongoimport --db Student --collection airlines --type csv --file "C:\Program Files\MongoDB\airline.csv" --headerline
2022-06-03T08:24:18.366+0530    connected to: mongod://localhost/
2022-06-03T08:24:18.395+0530    6 document(s) imported successfully. 0 document(s) failed to import.
C:\Program Files\MongoDB\Server\5.0\bin>
```

IV. Export data to a CSV file

This command used at the command prompt exports MongoDB JSON documents from

“Customers” collection in the “test” database into a CSV file “Output.txt” in the D:drive.

```
mongoexport --host localhost --db Student --collection airlines --csv --out  
/home/hduser/Desktop/output.txt --fields "Year","Quarter"
```

```
C:\Program Files\MongoDB\Server\5.0\bin>mongoexport --host localhost --db Student --collection airlines  
--csv --out "C:\home\hduser\Desktop\output.txt" --fields "Year","Quarter"  
2022-06-03T08:28:58.325+0530 csv flag is deprecated; please use --type=csv instead  
2022-06-03T08:28:58.946+0530 connected to: mongod://localhost/  
2022-06-03T08:28:58.972+0530 exported 6 records  
C:\Program Files\MongoDB\Server\5.0\bin>_
```

V. Save Method :

Save() method will insert a new document, if the document with the _id does not exist. If it exists it will replace the existing document.

```
db.Students.save({StudName:"Vamsi", Grade:"VI"})
```

```
switched to db Student  
> db.Students.save({StudName:"Vamsi",Grade:"VII"})  
WriteResult({ "nInserted" : 1 })  
> _
```

VI. Add a new field to existing Document:

```
db.Students.update({_id:4},{ $set:{Location:"Network"}})
```

```
> db.Students.update({_id:4},{ $set:{Location:"Network"}})  
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })  
> _
```


VII. Remove the field in an existing Document

```
db.Students.update({_id:4},{ $unset:{Location:"Network"}})
```

```
C:\> Command Prompt - mongo
> db.Students.update({_id:4},{ $unset:{Location:"Network"}})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
>
```

VIII. Finding Document based on search criteria suppressing few fields

```
db.Student.find({_id:1},{StudName:1,Grade:1,_id:0});
```

To find those documents where the Grade is not set to 'VII'

```
db.Student.find({Grade:{$ne:"VII"}}).pretty();
```

To find documents from the Students collection where the StudName ends with s.

```
db.Student.find({StudName:/s$/}).pretty();
```

```
> db.Student.find({_id:1},{StudName:1,Grade:1,_id:0});
>
```

```
C:\> Command Prompt - mongo
> db.Student.find({Grade:{$ne:'VII'}}).pretty();
> db.Student.find({StudName:/s$/}).pretty();
>
```

IX. to set a particular field value to NULL

```
> db.Students.update({_id:3},{ $set:{Location:null}})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
>
```

X Count the number of documents in Student Collections

```
> db.Student.count()
0
>
```

XI. Count the number of documents in Student Collections with grade :VII

db.Students.count({Grade:"VII"}) retrieve first 3 documents

db.Students.find({Grade:"VII"}).limit(3).pretty(); Sort the document in Ascending order

db.Students.find().sort({StudName:1}).pretty(); Note: for descending order : db.Students.find().sort({StudName:-1}).pretty(); to Skip the 1 st two documents from the Students Collections db.Students.find().skip(2).pretty()

```
> db.Students.find().sort({StudName:1}).pretty();
{
  "_id" : ObjectId("629979944de3211e43081306"),
  "StudName" : "Vamsi",
  "Grade" : "VII"
}
```

XII. Create a collection by name “food” and add to each document add a “fruits” array db.food.insert({ _id:1,

fruits:['grapes','mango','apple'] })

db.food.insert({ _id:2,

fruits:['grapes','mango','cherry'] })

db.food.insert({ _id:3, fruits:['banana','mango'] })

```
C:\> Command Prompt - mongo
> db.food.insert({_id:1,fruits:['grapes','mango','apple']})
WriteResult({ "nInserted" : 1 })
> db.food.insert({_id:2,fruits:['grapes','mango','cherry']})
WriteResult({ "nInserted" : 1 })
> db.food.insert({_id:3,fruits:['banana','mango']})
WriteResult({ "nInserted" : 1 })
>
```

To find those documents from the “food” collection which has the “fruits array” constitute of “grapes”, “mango” and “apple”. db.food.find ({fruits: ['grapes','mango','apple'] }). pretty().

```
> db.food.find({fruits:['grapes','mango','apple']}).pretty()
{ "_id" : 1, "fruits" : [ "grapes", "mango", "apple" ] }
>
```

To find in “fruits” array having “mango” in the first index position.

db.food.find ({'fruits.1':'grapes'})

```
> db.food.find({'fruits.1':'grapes'})
>
```

To find those documents from the “food” collection where the size of the array is

two. db.food.find ({“fruits”: {\$size:2}})

```
> db.food.find ( {"fruits": {$size:2}} )
{ "_id" : 3, "fruits" : [ "banana", "mango" ] }
> _
```

To find the document with a particular id and display the first two elements from the array “fruits”

db.food.find({_id:1},{“fruits”:{\$slice:2}})

```
> db.food.find({_id:1},{“fruits”:{$slice:2}})
{ "_id" : 1, "fruits" : [ "grapes", "mango" ] }
> _
```

To find all the documets from the food collection which have elements mango and grapes in the array “fruits”

db.food.find({fruits:{\$all:[“mango”,”grapes”]}})

```
> db.food.find({fruits:{$all:["mango","grapes"]}})
{ "_id" : 1, "fruits" : [ "grapes", "mango", "apple" ] }
{ "_id" : 2, "fruits" : [ "grapes", "mango", "cherry" ] }
>
```

update on Array: using particular id replace the element present in the 1 st index position of the fruits array with apple

```
db.food.update({_id:3},{ $set:{'fruits.1':'apple'}})
```

insert new key value pairs in the fruits array

```
db.food.update({_id:2},{ $push:{price:{grapes:80,mango:200,cherry:100}}})
```



```
> db.food.update({_id:3},{ $set:{'fruits.1':'apple'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.food.update({_id:2},{ $push:{price:{grapes:80,mango:200,cherry:100}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> _
```

Note: perform query operations using - pop, addToSet, pullAll and pull

XII. Aggregate Function :

Create a collection Customers with fields custID, AcctBal, AcctType.

Now group on “custID” and compute the sum of “AccBal”. db.Customers.aggregate

```
( { $group : { _id : “$custID”,TotAccBal : { $sum:”$AccBal” } } } ); match on
```

AcctType:”S” then group on “CustID” and compute the sum of “AccBal”.

```
db.Customers.aggregate ( { $match:{AcctType:”S”}}, { $group : { _id :
```

```
“$custID”,TotAccBal :
```

```
{ $sum:”$AccBal” } } } );
```

match on AcctType:”S” then group on “CustID” and compute the sum of

“AccBal” and total balance greater than 1200.

```
db.Customers.aggregate ( { $match:{AcctType:”S”}}, { $group : { _id : “$custID”,TotAccBal :
```

```
{ $sum:”$AccBal” } } }, { $match:{TotAccBal:{ $gt:1200}}});
```

```
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Customers.aggregate ( {$group : { _id : "$custID",TotAccBal : {$sum:"$AccBal"} } } );
> db.Customers.aggregate ( {$match:{AcctType:"S"}},{$group : { _id : "$custID",TotAccBal :
... {$sum:"$AccBal"} } } );
uncaught exception: SyntaxError: illegal character :
@(shell):1:43
> db.Customers.aggregate ( {$match:{AcctType:"S"}},{$group : { _id : "$custID",TotAccBal :{$sum:"$AccBal
"} } } );
> db.Customers.aggregate ( {$match:{AcctType:"S"}},{$group : { _id : "$custID",TotAccBal :{$sum:"$AccBa
l"} } }, {$match:{TotAccBal:{$gt:1200}}});
>
```

LAB 4 Screenshot of Hadoop installed

```
puneethk@ubuntu:~$ sudo su hadoop
[sudo] password for puneethk:
hadoop@ubuntu:/home/puneethk$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [ubuntu]
Starting resourcemanager
Starting nodemanagers
hadoop@ubuntu:/home/puneethk$ jps
4160 DataNode
3699 NameNode
4788 NodeManager
5163 Jps
4414 SecondaryNameNode
4654 ResourceManager
hadoop@ubuntu:/home/puneethk$
```


LAB 5 Execution of HDFS Commands for interaction with Hadoop Environment. (Minimum 10 commands to be executed)

```
hadoop@ubuntu:/home/puneethk$ hdfs dfs -mkdir /puni
hadoop@ubuntu:/home/puneethk$ hdfs dfs -ls /
Found 4 items
drwxr-xr-x - hadoop supergroup          0 2022-06-28 14:15 /demo
drwxr-xr-x - hadoop supergroup          0 2022-07-12 00:50 /puni
drwx----- - hadoop supergroup          0 2022-07-01 23:02 /tmp
drwxr-xr-x - hadoop supergroup          0 2022-07-01 21:46 /wordcount
hadoop@ubuntu:/home/puneethk$ hdfs dfs -put /home/hadoop/sm.txt /puni/this.txt
hadoop@ubuntu:/home/puneethk$ hdfs dfs -ls /puni
Found 1 items
-rw-r--r-- 1 hadoop supergroup          6 2022-07-12 00:51 /puni/this.txt
hadoop@ubuntu:/home/puneethk$ hdfs dfs -copyFromLocal /home/hadoop/sm.txt /puni/new.txt
hadoop@ubuntu:/home/puneethk$ hdfs dfs -ls /puni
Found 2 items
-rw-r--r-- 1 hadoop supergroup          6 2022-07-12 00:52 /puni/new.txt
-rw-r--r-- 1 hadoop supergroup          6 2022-07-12 00:51 /puni/this.txt
hadoop@ubuntu:/home/puneethk$ hdfs dfs -copyToLocal /puni/this.txt /home/hadoop/local.txt
hadoop@ubuntu:/home/puneethk$ hdfs dfs -get /puni/new.txt /home/hadoop/old.txt
hadoop@ubuntu:/home/puneethk$ hdfs dfs -getmerge /puni/new.txt /puni/this.txt /home/hadoop/merge.txt
```

```
hadoop@ubuntu:/home/puneethk$ hdfs dfs -getmerge /puni/new.txt /puni/this.txt /home/hadoop/merge.txt
hadoop@ubuntu:/home/puneethk$ hdfs dfs -cat /puni/this.txt
dummy
hadoop@ubuntu:/home/puneethk$ hdfs dfs -mv /puni/this.txt /abc
hadoop@ubuntu:/home/puneethk$ hdfs dfs -ls /abc
-rw-r--r-- 1 hadoop supergroup          6 2022-07-12 00:51 /abc
hadoop@ubuntu:/home/puneethk$ hdfs dfs -ls /abc/abc
ls: '/abc/abc': No such file or directory
hadoop@ubuntu:/home/puneethk$ hdfs dfs -cp /puni/this.txt /abc/newone.txt
cp: '/puni/this.txt': No such file or directory
hadoop@ubuntu:/home/puneethk$ hdfs dfs -cp /puni/new.txt /abc/newone.txt
cp: /abc (is not a directory)
hadoop@ubuntu:/home/puneethk$ hdfs dfs -cp /puni/new.txt /newone.txt
hadoop@ubuntu:/home/puneethk$ hdfs dfs -ls /
Found 6 items
-rw-r--r-- 1 hadoop supergroup          6 2022-07-12 00:51 /abc
drwxr-xr-x - hadoop supergroup          0 2022-06-28 14:15 /demo
-rw-r--r-- 1 hadoop supergroup          6 2022-07-12 00:57 /newone.txt
drwxr-xr-x - hadoop supergroup          0 2022-07-12 00:56 /puni
drwx----- - hadoop supergroup          0 2022-07-01 23:02 /tmp
drwxr-xr-x - hadoop supergroup          0 2022-07-01 21:46 /wordcount
hadoop@ubuntu:/home/puneethk$
```

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

```
package temp;
```

```
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.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


```
package temp;
```

```
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.mapreduce.Mapper;
```

```
public class AverageMapper extends Mapper<LongWritable, Text, Text,  
IntWritable> {  
    public static final int MISSING = 9999;
```

```
    public void map(LongWritable key, Text value, Mapper<LongWritable, Text,  
Text, IntWritable>.Context context) throws IOException, InterruptedException {  
        int temperature;  
        String line = value.toString();  
        String year = line.substring(15, 19);  
        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 != 9999 && quality.matches("[01459]"))  
            context.write(new Text(year), new IntWritable(temperature));  
    }  
}
```

AverageReducer

```
package temp;
```

```
import java.io.IOException;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.Reducer;
```

```

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

```

```

.:\\hadoop-3.3.0\\sbin>hadoop jar C:\\avgtemp.jar temp.AverageDriver /input dir/temp.txt /avgtemp.outputdir
1021-05-15 14:52:50,635 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
1021-05-15 14:52:51,005 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
1021-05-15 14:52:51,111 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Anusree/.staging/job_1621060230696_0005
1021-05-15 14:52:51,735 INFO input.FileInputFormat: Total input files to process : 1
1021-05-15 14:52:52,751 INFO mapreduce.JobSubmitter: number of splits:1
1021-05-15 14:52:53,073 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1621060230696_0005
1021-05-15 14:52:53,073 INFO mapreduce.JobSubmitter: Executing with tokens: []
1021-05-15 14:52:53,237 INFO conf.Configuration: resource-types.xml not found
1021-05-15 14:52:53,238 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
1021-05-15 14:52:53,312 INFO impl.YarnClientImpl: Submitted application application_1621060230696_0005
1021-05-15 14:52:53,352 INFO mapreduce.Job: The url to track the job: http://LAPTOP-JG329ESD:8088/proxy/application_1621060230696_0005/
1021-05-15 14:52:53,353 INFO mapreduce.Job: Running job: job_1621060230696_0005
1021-05-15 14:53:06,640 INFO mapreduce.Job: Job job_1621060230696_0005 running in uber mode : false
1021-05-15 14:53:06,643 INFO mapreduce.Job:  map 0% reduce 0%
1021-05-15 14:53:12,758 INFO mapreduce.Job:  map 100% reduce 0%
1021-05-15 14:53:19,860 INFO mapreduce.Job:  map 100% reduce 100%
1021-05-15 14:53:25,967 INFO mapreduce.Job: Job job_1621060230696_0005 completed successfully
1021-05-15 14:53:26,096 INFO mapreduce.Job: Counters: 54
File System Counters
  FILE: Number of bytes read=72210
  FILE: Number of bytes written=674341
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=894860
  HDFS: Number of bytes written=8
  HDFS: Number of read operations=8
  HDFS: Number of large read operations=0
  HDFS: Number of write operations=2
  HDFS: Number of bytes read erasure-coded=0
Job Counters
  Launched map tasks=1
  Launched reduce tasks=1
  Data-local map tasks=1
  Total time spent by all maps in occupied slots (ms)=3782

```

```

C:\hadoop-3.3.0\sbin>hdfs dfs -ls /avgtemp_outputdir
Found 2 items
-rw-r--r--  1 Anusree supergroup          0 2021-05-15 14:53 /avgtemp_outputdir/_SUCCESS
-rw-r--r--  1 Anusree supergroup          8 2021-05-15 14:53 /avgtemp_outputdir/part-r-00000

C:\hadoop-3.3.0\sbin>hdfs dfs -cat /avgtemp_outputdir/part-r-00000
1901    46

C:\hadoop-3.3.0\sbin>

```

b) find the mean max temperature for every month

MeanMax

MeanMaxDriver.class

package meanmax;

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.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class MeanMaxDriver {

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(MeanMaxDriver.class);

 job.setJobName("Max temperature");

 FileInputFormat.addInputPath(job, **new** Path(args[0]));

 FileOutputFormat.setOutputPath(job, **new** Path(args[1]));

 job.setMapperClass(MeanMaxMapper.class);

 job.setReducerClass(MeanMaxReducer.class);

 job.setOutputKeyClass(Text.class);

 job.setOutputValueClass(IntWritable.class);

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

 }

}

MeanMaxMapper.class

package meanmax;

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.mapreduce.Mapper;

public class MeanMaxMapper **extends** Mapper<LongWritable, Text, Text, IntWritable> {

public static final int MISSING = 9999;

public void map(LongWritable key, Text value, Mapper<LongWritable, Text, Text, IntWritable>.Context context) **throws** IOException, InterruptedException {

int temperature;

 String line = value.toString();

 String month = line.substring(19, 21);

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 != 9999 && quality.matches("[01459]"))

 context.write(**new** Text(month), **new** IntWritable(temperature));

 }

}

MeanMaxReducer.class

package meanmax;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class MeanMaxReducer **extends** Reducer<Text, IntWritable, Text, IntWritable> {

```
public void reduce(Text key, Iterable<IntWritable> values, Reducer<Text,
IntWritable, Text, IntWritable>.Context context) throws IOException,
InterruptedException {
    int max_temp = 0;
    int total_temp = 0;
    int count = 0;
    int days = 0;
    for (IntWritable value : values) {
        int temp = value.get();
        if (temp > max_temp)
            max_temp = temp;
        count++;
        if (count == 3) {
            total_temp += max_temp;
            max_temp = 0;
            count = 0;
            days++;
        }
    }
    context.write(key, new IntWritable(total_temp / days));
}
```

```

C:\hadoop-3.3.0\sbin>hadoop jar C:\meanmax.jar meanmax.MeanMaxDriver /input_dir/temp.txt /meanmax_output
2021-05-21 20:28:05,250 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2021-05-21 20:28:06,662 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2021-05-21 20:28:06,916 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Anusree/.staging/job_1621608943095_0001
2021-05-21 20:28:08,426 INFO input.FileInputFormat: Total input files to process : 1
2021-05-21 20:28:09,107 INFO mapreduce.JobSubmitter: number of splits:1
2021-05-21 20:28:09,741 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1621608943095_0001
2021-05-21 20:28:09,741 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-05-21 20:28:10,029 INFO conf.Configuration: resource-types.xml not found
2021-05-21 20:28:10,030 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-05-21 20:28:10,676 INFO impl.YarnClientImpl: Submitted application application_1621608943095_0001
2021-05-21 20:28:11,005 INFO mapreduce.Job: The url to track the job: http://LAPTOP-JG329ESD:8088/proxy/application_1621608943095_0001/
2021-05-21 20:28:11,006 INFO mapreduce.Job: Running job: job_1621608943095_0001
2021-05-21 20:28:29,385 INFO mapreduce.Job: Job job_1621608943095_0001 running in uber mode : false
2021-05-21 20:28:29,389 INFO mapreduce.Job: map 0% reduce 0%
2021-05-21 20:28:40,664 INFO mapreduce.Job: map 100% reduce 0%
2021-05-21 20:28:50,832 INFO mapreduce.Job: map 100% reduce 100%
2021-05-21 20:28:58,965 INFO mapreduce.Job: Job job_1621608943095_0001 completed successfully
2021-05-21 20:28:59,178 INFO mapreduce.Job: Counters: 54

```

File System Counters

```

FILE: Number of bytes read=59082
FILE: Number of bytes written=648091
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=894860
HDFS: Number of bytes written=74
HDFS: Number of read operations=8
HDFS: Number of large read operations=0
HDFS: Number of write operations=2
HDFS: Number of bytes read erasure-coded=0

```

Job Counters

```

Launched map tasks=1
Launched reduce tasks=1
Data-local map tasks=1
Total time spent by all maps in occupied slots (ms)=8077
Total time spent by all reduces in occupied slots (ms)=7511
Total time spent by all map tasks (ms)=8077
Total time spent by all reduce tasks (ms)=7511
Total vcore-milliseconds taken by all map tasks=8077
Total vcore-milliseconds taken by all reduce tasks=7511
Total megabyte-milliseconds taken by all map tasks=8270848
Total megabyte-milliseconds taken by all reduce tasks=7691264

```

```

C:\hadoop-3.3.0\sbin>hdfs dfs -cat /meanmax_output/*

```

```

01      4
02      0
03      7
04     44
05    100
06    168
07    219
08    198
09    141
10    100
11     19
12     3

```

```

C:\hadoop-3.3.0\sbin>

```

LAB 7 For a given Text file, create a Map Reduce program to sort the content in an alphabetic order listing only top 10 maximum occurrences of words.

Driver-TopN.class

```
package samples.topn;

import java.io.IOException;
import java.util.StringTokenizer;
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.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;

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("&quot;Top N&quot;");
job.setJarByClass(TopN.class);
job.setMapperClass(TopNMapper.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);
}

public static class TopNMapper extends Mapper<Object, Text,
Text, IntWritable> {
private static final IntWritable one = new IntWritable(1);

private Text word = new Text();
private String tokens = "&quot;[_$#&lt;&gt;\\^=\\[\\]\\*^\\\\\\,;\\.\\|-
:()?!\\&quot;&#39;]&quot;;";
public void map(Object key, Text value, Mapper<Object,
Text, Text, IntWritable>.Context context) throws IOException,
InterruptedException {
String cleanLine =
value.toString().toLowerCase().replaceAll(this.tokens, "&quot; &quot;");
StringTokenizer itr = new StringTokenizer(cleanLine);

```



```
while (itr.hasMoreTokens()) {  
    this.word.set(itr.nextToken().trim());  
    context.write(this.word, one);  
}  
}  
}  
}
```

```
TopNCombiner.class  
package samples.topn;  
import java.io.IOException;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.Reducer;  
public class TopNCombiner extends Reducer<Text, IntWritable,  
Text, IntWritable> {  
    public void reduce(Text key, Iterable<IntWritable> values,  
Reducer<Text, IntWritable, Text, IntWritable>.Context context)  
throws IOException, InterruptedException {  
        int sum = 0;  
        for (IntWritable val : values)  
            sum += val.get();  
        context.write(key, new IntWritable(sum));  
    }  
}
```

TopNMapper.class

```
package samples.topn;

import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

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

    private static final IntWritable one = new IntWritable(1);
    private Text word = new Text();
    private String tokens = "&quot;[_$#&lt;&gt;\\^=\\[\\]\\|\\*\\/\\\\\\,;\\.\\|\\-
:()?!\\&quot;&#39;]&quot;;";

    public void map(Object key, Text value, Mapper<Object,
Text, Text, IntWritable>.Context context) throws IOException,
InterruptedException {

        String cleanLine =
value.toString().toLowerCase().replaceAll(this.tokens, "&quot; &quot;");
        StringTokenizer itr = new StringTokenizer(cleanLine);
        while (itr.hasMoreTokens()) {
            this.word.set(itr.nextToken().trim());
            context.write(this.word, one);
        }
    }
}
```

```
}
```

```
TopNReducer.class
```

```
package samples.topn;
```

```
import java.io.IOException;
```

```
import java.util.HashMap;
```

```
import java.util.Map;
```

```
import org.apache.hadoop.io.IntWritable;
```

```
import org.apache.hadoop.io.Text;
```

```
import org.apache.hadoop.mapreduce.Reducer;
```

```
import utils.MiscUtils;
```

```
public class TopNReducer extends Reducer<Text, IntWritable,
```

```
Text, IntWritable> {
```

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

```
    public void reduce(Text key, Iterable<IntWritable> values,
```

```
        Reducer<Text, IntWritable, Text, IntWritable>.Context context)
```

```
        throws IOException, InterruptedException {
```

```
        int sum = 0;
```

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

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

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

```
    }
```

```
    protected void cleanup(Reducer<Text, IntWritable, Text, IntWritable>.Context context)
```

```
        throws IOException, InterruptedException {
```

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

```
        int counter = 0;
```

```

for (Text key : sortedMap.keySet()) {
    if (counter++ == 20)
        break;
    context.write(key, sortedMap.get(key));
}
}
}
}

```

```

C:\hadoop-3.3.0\sbin>hadoop jar C:\sort.jar samples.topn.TopN /input_dir/input.txt /output_dir
2021-05-08 19:54:54,582 INFO client.DefaultHadoopFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2021-05-08 19:54:55,291 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Anusree/.staging/job_1620483374279_0001
2021-05-08 19:54:55,821 INFO input.FileInputFormat: Total input files to process : 1
2021-05-08 19:54:56,261 INFO mapreduce.JobSubmitter: number of splits:1
2021-05-08 19:54:56,552 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1620483374279_0001
2021-05-08 19:54:56,552 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-05-08 19:54:56,843 INFO conf.Configuration: resource-types.xml not found
2021-05-08 19:54:56,843 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-05-08 19:54:57,387 INFO impl.YarnClientImpl: Submitted application application_1620483374279_0001
2021-05-08 19:54:57,587 INFO mapreduce.Job: The url to track the job: http://LAPTOP-JG329ESD:8088/proxy/application_1620483374279_0001/
2021-05-08 19:54:57,588 INFO mapreduce.Job: Running job: job_1620483374279_0001
2021-05-08 19:55:13,792 INFO mapreduce.Job: Job job_1620483374279_0001 running in uber mode : false
2021-05-08 19:55:13,794 INFO mapreduce.Job:  map 0% reduce 0%
2021-05-08 19:55:20,020 INFO mapreduce.Job:  map 100% reduce 0%
2021-05-08 19:55:27,116 INFO mapreduce.Job:  map 100% reduce 100%
2021-05-08 19:55:33,199 INFO mapreduce.Job: Job job_1620483374279_0001 completed successfully
2021-05-08 19:55:33,334 INFO mapreduce.Job: Counters: 54
    File System Counters
        FILE: Number of bytes read=65
        FILE: Number of bytes written=530397
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=142
        HDFS: Number of bytes written=31
        HDFS: Number of read operations=8
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
        HDFS: Number of bytes read erasure-coded=0

```

```

C:\hadoop-3.3.0\sbin>hdfs dfs -cat /output_dir/*
hello      2
hadoop     1
world      1
bye        1

C:\hadoop-3.3.0\sbin>

```

```
C:\hadoop-3.3.0\sbin>jps
11072 DataNode
20528 Jps
5620 ResourceManager
15532 NodeManager
6140 NameNode

C:\hadoop-3.3.0\sbin>hdfs dfs -mkdir /input_dir

C:\hadoop-3.3.0\sbin>hdfs dfs -ls /
Found 1 items
drwxr-xr-x - Anusree supergroup      0 2021-05-08 19:46 /input_dir

C:\hadoop-3.3.0\sbin>hdfs dfs -copyFromLocal C:\input.txt /input_dir

C:\hadoop-3.3.0\sbin>hdfs dfs -ls /input_dir
Found 1 items
-rw-r--r--  1 Anusree supergroup      36 2021-05-08 19:48 /input_dir/input.txt

C:\hadoop-3.3.0\sbin>hdfs dfs -cat /input_dir/input.txt
hello
world
hello
hadoop
bye
```

LAB 8 Create a Map Reduce program to demonstrating join operation

```
// 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.lib.MultipleInputs;
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 'Department Emp Strength input' with 'Department Name
            input'");
```

```
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() + "\t\t" + 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("\t");
        output.collect(new TextPair(SingleNodeData[0], "1"), 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("\t");
    }
}

```

```
output.collect(new TextPair(SingleNodeData[3], "0"), 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 + "\t" + 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-3.3.0\sbin>hdfs dfs -ls /join8_output/
Found 2 items
-rw-r--r--  1 Anusree supergroup      0 2021-06-13 12:16 /join8_output/_SUCCESS
-rw-r--r--  1 Anusree supergroup    71 2021-06-13 12:16 /join8_output/part-00000

C:\hadoop-3.3.0\sbin>hdfs dfs -cat /join8_output/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

a) Program to print word count on scala shell

```
var map = sc.textFile("/path/to/text/file").flatMap(line => line.split(" ")).map(word => (word,1));
```

```
/** reduce */
```

```
var counts = map.reduceByKey(_ + _);
```

```
/** save the output to file */
```

```
counts.saveAsTextFile("/path/to/output/")
```

b) print “Hello world” on scala IDE

```
object Sample{
```

```
  def main(args:Array[String]){
```

```
    println ("Hello World")
```

```
  }
```

```
}
```

```
scala> val data = sc.textFile("input.txt")
data: org.apache.spark.rdd.RDD[String] = input.txt MapPartitionsRDD[3] at textFile at <console>:23

scala> data.collect()
res3: Array[String] = Array(hi there im khushil, im here to run spark and hadoop, lets see which is better)

scala> val splitdata = data.flatMap(line => line.split(" "));
splitdata: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[4] at flatMap at <console>:23

scala> splitdata.collect();
res4: Array[String] = Array(hi, there, im, khushil, im, here, to, run, spark, and, hadoop, lets, see, which, is, better)

scala> val mapdata = splitdata.map(word=>(word,1));
mapdata: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[5] at map at <console>:23

scala> val reducedata = mapdata.reduceByKey(_+_);
reducedata: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[6] at reduceByKey at <console>:23

scala> reducedata.collect();
res5: Array[(String, Int)] = Array((im,2), (is,1), (here,1), (there,1), (better,1), (khushil,1), (lets,1), (spark,1), (run,1), (hadoop,1), (hi,1), (to,1), (see,1), (which,1), (and,1))

scala> reducedata.saveAsTextFile("output.txt");

scala> _
```

LAB 10 Using RDD and FlatMap 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

```
val data=sc.textFile("sparkdata.txt")
val splitdata = data.flatMap(line => line.split(" "));
val mapdata = splitdata.map(word => (word,1));
val reducedata = mapdata.reduceByKey(_+_);
import scala.collection.immutable.ListMap
val sorted=ListMap(count.collect.startWith(_ . _ 2>_ . _2:_*))
println(sorted)
for ((K<V)<-sorted)
{
  if(v>4)
  {
    print(K+"",")
  }
  print(v)
  println()
}
}
```

```

scala> val filerdd = sc.textFile("input.txt");
filerdd: org.apache.spark.rdd.RDD[String] = input.txt MapPartitionsRDD[13] at textFile at <console>:24

scala> val counts = filerdd.flatMap(line=>line.split(" ")).map(word=>(word,1)).reduceByKey(_+_);
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[16] at reduceByKey at <console>:24

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] = ListMap(im -> 2, is -> 1, here -> 1, there -> 1
, better -> 1, khushil -> 1, lets -> 1, spark -> 1, run -> 1, hadoop -> 1, hi -> 1, to -> 1, see -> 1, w
hich -> 1, and -> 1)

scala> println(sorted);
ListMap(im -> 2, is -> 1, here -> 1, there -> 1, better -> 1, khushil -> 1, lets -> 1, spark -> 1, run -
> 1, hadoop -> 1, hi -> 1, to -> 1, see -> 1, which -> 1, and -> 1)

scala> for((k,v)<-sorted)
| {
|   if(v>4)
|   {
|     print(k+",")
|     print(v)
|     println()
|   }
| }

scala> for((k,v)<-sorted)
| {
|   println(k+",")
|   println(v)
|   println()
| }
im,
2
is,
1
here,
1
there,
1
better,
1
khushil,
1
lets,
1
spark,
1

```