VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

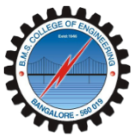
LAB REPORT
on

Big Data Analytics Lab

Submitted by

CHINMAYI(1BM21CS045)

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
Feb-2024 to July-2024

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 "LAB COURSE **Big Data Analytics**" carried out by **CHINMAYI(1BM21CS045),** 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 2024. The Lab report has been approved as it satisfies the academic requirements in respect of a **Big Data Analytics Lab - (22CS6PEBDA)** work prescribed for the said degree.

Rekha G S
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

Index Sheet

Course Outcome

CO1	Apply the concept of NoSQL, Hadoop, Spark for a given task
CO2	Analyze data analytic techniques for a given problem.
CO3	Conduct experiments using data analytics mechanisms for a given problem.

1 Perform the following DB operations using Cassandra.

1.Create a keyspace by name Employee
CREATE KEYSPACE Employee WITH REPLICATION = {'class':'SimpleStrategy', 'replication_factor':1};
use Employee;

```
cqlsh> CREATE KEYSPACE Employee WITH REPLICATION = {'class':'SimpleStrategy', 'replication_factor':1};
cqlsh> use 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
CREATE TABLE Employee_Info(Emp_id int PRIMARY KEY,Emp_Name text, Designation text, Date_of_Joining timestamp, Salary double, Dept_Name text);

```
cqlsh:employee> CREATE TABLE Employee_Info(Emp_id int PRIMARY KEY,Emp_Name text
, Designation text, Date_of_Joining timestamp, Salary double, Dept_Name text);
```

3. Insert the values into the table in batch
Begin batch
insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joining, Salary,Dept_Name)
VALUES (111,'John','Manager','2010-02-27',80000.0,'IT')
insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joining, Salary,Dept_Name)
VALUES (121,'James','Developer','2019-06-27',60000.0,'IT')
insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joining, Salary,Dept_Name)
VALUES (131,'Riya','Developer','2020-01-17',40000.0,'IT')
insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joining, Salary,Dept_Name)
VALUES (141,'Priya','Analyst','2022-02-18',50000.0,'IT')
insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joining, Salary,Dept_Name)
VALUES (151,'Davaid','Analyst','2012-02-18',70000.0,'IT')
APPLY BATCH;

```
cqlsh:employee> begin batch
            ... insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joinin
g, Salary,Dept_Name)
            ... VALUES (111,'John','Manager','2010-02-27',80000.0,'IT')
            ... insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joinin
g, Salary,Dept_Name)
            ... VALUES (121,'James','Developer','2019-06-27',60000.0,'IT')
            ... insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joinin
g, Salary,Dept_Name)
            ... VALUES (131,'Riya','Developer','2020-01-17',40000.0,'IT')
            ... insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joinin
g, Salary,Dept_Name)
            ... VALUES (141,'Priya','Analyst','2022-02-18',50000.0,'IT')
            ... insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joinin
g, Salary,Dept_Name)
            ... VALUES (151,'Davaid','Analyst','2012-02-18',70000.0,'IT')
            ... APPLY BATCH;
```

```
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining                 | dept_name | designation | emp_name | salary
--------+---------------------------------+-----------+-------------+----------+--------
    111 | 2010-02-26 18:30:00.000000+0000 |        IT |     Manager |     John |  80000
    151 | 2012-02-17 18:30:00.000000+0000 |        IT |     Analyst |   Davaid |  70000
    121 | 2019-06-26 18:30:00.000000+0000 |        IT |   Developer |    James |  60000
    141 | 2022-02-17 18:30:00.000000+0000 |        IT |     Analyst |    Priya |  50000
    131 | 2020-01-16 18:30:00.000000+0000 |        IT |   Developer |     Riya |  40000

(5 rows)
```

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

UPDATE Employee_Info SET Emp_Name = 'Jeevan', Department = 'Finance' WHERE Emp_id = 121;

```
cqlsh:employee> UPDATE Employee_Info SET Emp_Name = 'Jeevan', dept_name = 'Finance' WHERE Emp_id = 121;
cqlsh:employee> select * from Employee_info where Emp_Id = 121;

 emp_id | date_of_joining                 | dept_name | designation | emp_name | salary
--------+---------------------------------+-----------+-------------+----------+--------
    121 | 2019-06-26 18:30:00.000000+0000 |   Finance |   Developer |   Jeevan |  60000

(1 rows)
```

5. Sort the details of Employee records based on salary

select * from employye_info where emo_id in (111,121,131,141,151) order by salary desc;

```
cqlsh:employee> paging off
Disabled Query paging.
cqlsh:employee> select * from  employee_info where emp_id in (111,121,131,141,151) order by salary DESC;

 emp_id | salary | date_of_joining                 | dept_name | designation | emp_name | projects
--------+--------+---------------------------------+-----------+-------------+----------+----------------
    111 |  80000 | 2010-02-26 18:30:00.000000+0000 |        IT |     Manager |     John | {'SER', 'Studen
    151 |  70000 | 2012-02-17 18:30:00.000000+0000 |        IT |     Analyst |   Davaid |
    121 |  60000 | 2019-06-26 18:30:00.000000+0000 |        IT |   Developer |    James |
    141 |  50000 | 2022-02-17 18:30:00.000000+0000 |        IT |     Analyst |    Priya |
    131 |  40000 | 2020-01-16 18:30:00.000000+0000 |        IT |   Developer |     Riya |

(5 rows)
cqlsh:employee>
```

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.

alter table employee_info add projects set<text>;

```
(5 rows)
cqlsh:employee> alter table employee_info add projects set<text>;
cqlsh:employee> select * from  employee_info;

 emp_id | salary | date_of_joining                 | dept_name | designation | emp_name | projects
--------+--------+---------------------------------+-----------+-------------+----------+----------
    111 |  80000 | 2010-02-26 18:30:00.000000+0000 |        IT |     Manager |     John |     null
    151 |  70000 | 2012-02-17 18:30:00.000000+0000 |        IT |     Analyst |   Davaid |     null
    121 |  60000 | 2019-06-26 18:30:00.000000+0000 |        IT |   Developer |    James |     null
    141 |  50000 | 2022-02-17 18:30:00.000000+0000 |        IT |     Analyst |    Priya |     null
    131 |  40000 | 2020-01-16 18:30:00.000000+0000 |        IT |   Developer |     Riya |     null

(5 rows)
```

7. Update the altered table to add project names.

update employee_info set projects = projects + {‘emailSpamDetection’,‘StudentProtal’} where emp_id = 111 and salary = 80000;

```
cqlsh:employee> update employee_info set projects = projects + {'emailSpamDetection', 'SER', 'StudentPortal
'} where emp_id = 111 and salary = 80000 ;
cqlsh:employee> select * from  employee_info;

 emp_id | salary | date_of_joining                 | dept_name | designation | emp_name | projects
--------+--------+---------------------------------+-----------+-------------+----------+------------------
------------------------------
    111 |  80000 | 2010-02-26 18:30:00.000000+0000 |        IT |     Manager |     John | {'SER', 'StudentP
ortal', 'emailSpamDetection'}
    151 |  70000 | 2012-02-17 18:30:00.000000+0000 |        IT |     Analyst |   Davaid |
                          null
    121 |  60000 | 2019-06-26 18:30:00.000000+0000 |        IT |   Developer |    James |
                          null
    141 |  50000 | 2022-02-17 18:30:00.000000+0000 |        IT |     Analyst |    Priya |
                          null
    131 |  40000 | 2020-01-16 18:30:00.000000+0000 |        IT |   Developer |     Riya |
                          null

(5 rows)
cqlsh:employee>
```

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

```
cqlsh:employee> insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joining, Salary,Dept_Name)
           ... VALUES (161,'Chinmayi','Developer','2013-02-18',90000.0,'WEB') using ttl = 30;
SyntaxException: line 2:73 no viable alternative at input '=' (...,90000.0,'WEB') using ttl [=]...)
cqlsh:employee> insert into Employee_Info(Emp_id,Emp_Name,Designation, Date_of_Joining, Salary,Dept_Name) VALUES (161,'Chinmayi','D
eveloper','2013-02-18',90000.0,'WEB') using ttl 30;
cqlsh:employee> select * from employee;
InvalidRequest: Error from server: code=2200 [Invalid query] message="table employee does not exist"
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining                 | dept_name | designation | emp_name | salary
--------+---------------------------------+-----------+-------------+----------+--------
    111 | 2010-02-26 18:30:00.000000+0000 |        IT |     Manager |     John |  80000
    151 | 2012-02-17 18:30:00.000000+0000 |        IT |     Analyst |   Davaid |  70000
    121 | 2019-06-26 18:30:00.000000+0000 |        IT |   Developer |    James |  60000
    141 | 2022-02-17 18:30:00.000000+0000 |        IT |     Analyst |    Priya |  50000
    131 | 2020-01-16 18:30:00.000000+0000 |        IT |   Developer |     Riya |  40000
    161 | 2013-02-17 18:30:00.000000+0000 |       WEB |   Developer | Chinmayi |  90000

(6 rows)
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining                 | dept_name | designation | emp_name | salary
--------+---------------------------------+-----------+-------------+----------+--------
    111 | 2010-02-26 18:30:00.000000+0000 |        IT |     Manager |     John |  80000
    151 | 2012-02-17 18:30:00.000000+0000 |        IT |     Analyst |   Davaid |  70000
    121 | 2019-06-26 18:30:00.000000+0000 |        IT |   Developer |    James |  60000
    141 | 2022-02-17 18:30:00.000000+0000 |        IT |     Analyst |    Priya |  50000
    131 | 2020-01-16 18:30:00.000000+0000 |        IT |   Developer |     Riya |  40000

(5 rows)
cqlsh:employee>
```

2 Perform the following DB operations using Cassandra.

1.Create a keyspace by name Library

```
cqlsh> CREATE KEYSPACE IF NOT EXISTS Library WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
cqlsh> use 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:library> CREATE TABLE IF NOT EXISTS Library_Info (
           ...     Stud_Id INT,
           ...     Counter_value COUNTER,
           ...     Stud_Name TEXT,
           ...     Book_Name TEXT,
           ...     Book_Id TEXT,
           ...     Date_of_issue TIMESTAMP,
           ...     primary key (Stud_Id,Book_Id,Stud_Name,Book_Name,Date_of_issue)
           ... );
```

3. Insert the values into the table in batch

```
cqlsh:library> UPDATE Library_Info SET Counter_value = Counter_value + 1
          ... WHERE Stud_Id = 111 AND Stud_Name = 'John' AND Book_Name = 'Introduction to Cassandra' AND Book_Id = 'CASS101' AND Date_of_issue = '2024-05-07';
cqlsh:library> select * from Library_info;

 stud_id | book_id | stud_name | book_name                | date_of_issue                | counter_value
---------+---------+-----------+--------------------------+------------------------------+---------------
     111 | CASS101 |      John | Introduction to Cassandra | 2024-05-06 18:30:00.000000+0000 |             1

(1 rows)
cqlsh:library> Update Library_Info Set Counter_value = Counter_value+1 where Stud_Id = 112, Stud_Name = 'Riya',Book_Name = 'BDA', Book_Id='BDA01', Date_of_issue='2024-06-07';
SyntaxException: line 1:75 mismatched input ',' expecting EOF (...+1 where Stud_Id = 112[,] Stud_Name...)
cqlsh:library> UPDATE Library_Info SET Counter_value = Counter_value + 1  WHERE Stud_Id = 112 AND Stud_Name = 'Riya' AND Book_Name = 'BDA' AND Book_Id = 'BDA202' AND Date_of_issue = '2024-06-09';
cqlsh:library> UPDATE Library_Info SET Counter_value = Counter_value + 1  WHERE Stud_Id = 113 AND Stud_Name = 'JAMES' AND Book_Name = 'ML' AND Book_Id = 'ML303' AND Date_of_issue = '2024-06-10';
cqlsh:library> UPDATE Library_Info SET Counter_value = Counter_value + 1  WHERE Stud_Id = 114 AND Stud_Name = 'PRIYA' AND Book_Name = 'MongoDb' AND Book_Id = 'mongo404' AND Date_of_issue = '2024-06-12';
cqlsh:library> select * from Library_info;

 stud_id | book_id  | stud_name | book_name                | date_of_issue                | counter_value
---------+----------+-----------+--------------------------+------------------------------+---------------
     114 | mongo404 |     PRIYA |                  MongoDb | 2024-06-11 18:30:00.000000+0000 |             1
     111 |  CASS101 |      John | Introduction to Cassandra | 2024-05-06 18:30:00.000000+0000 |             1
     113 |    ML303 |     JAMES |                       ML | 2024-06-09 18:30:00.000000+0000 |             1
     112 |   BDA202 |      Riya |                      BDA | 2024-06-08 18:30:00.000000+0000 |             1

(4 rows)
```

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

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

 stud_id | book_id | stud_name | book_name                | date_of_issue                | counter_value
---------+---------+-----------+--------------------------+------------------------------+---------------
     111 | CASS101 |      John | Introduction to Cassandra | 2024-05-06 18:30:00.000000+0000 |             1
     113 |   ML303 |     JAMES |                       ML | 2024-06-09 18:30:00.000000+0000 |             1
     112 |  BDA202 |      Riya |                      BDA | 2024-06-08 18:30:00.000000+0000 |             2
```

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

```
cqlsh:library> UPDATE Library_Info SET Counter_value = Counter_value + 1  WHERE Stud_Id = 112 AND Stud_Name = 'Riya' AND Book_Name
= 'BDA' AND Book_Id = 'BDA202' AND Date_of_issue = '2024-06-09';
cqlsh:library> select * from Library_info;

 stud_id | book_id  | stud_name | book_name                | date_of_issue                | counter_value
---------+----------+-----------+--------------------------+------------------------------+---------------
     114 | mongo404 |     PRIYA |                  MongoDb | 2024-06-11 18:30:00.000000+0000 |             2
     111 |  CASS101 |      John | Introduction to Cassandra | 2024-05-06 18:30:00.000000+0000 |             1
     113 |   ML303 |     JAMES |                       ML | 2024-06-09 18:30:00.000000+0000 |             1
     112 |  BDA202 |      Riya |                      BDA | 2024-06-08 18:30:00.000000+0000 |             2
```

6. Export the created column to a csv file

```
(1 rows)
cqlsh:library> copy Library_info (Stud_Id,Book_Id,Stud_Name,Book_Name,Date_of_issue,Counter_value) TO '/home/bmscecse/Library_info.
csv';
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, book_id, stud_name, book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate:      56 rows/s; Avg. rate:      56 rows/s
4 rows exported to 1 files in 0.090 seconds.
cqlsh:library>
cqlsh:library> select * from Library_info;
```

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

```
cqlsh:library> COPY Library_Info (Stud_Id,Book_Id,Stud_Name,Book_Name,Date_of_issue,Counter_value) FROM '/home/bmscecse/Library_inf
o.csv' WITH HEADER = true;
Using 16 child processes

Starting copy of library.library_info with columns [stud_id, book_id, stud_name, book_name, date_of_issue, counter_value].
Processed: 3 rows; Rate:       5 rows/s; Avg. rate:       8 rows/s
3 rows imported from 1 files in 0.384 seconds (0 skipped).
cqlsh:library> select * from Library_info;

 stud_id | book_id | stud_name | book_name                | date_of_issue                   | counter_value
---------+---------+-----------+--------------------------+---------------------------------+---------------
     111 | CASS101 |      John | Introduction to Cassandra | 2024-05-06 18:30:00.000000+0000 |             1
     113 |   ML303 |     JAMES |                       ML | 2024-06-09 18:30:00.000000+0000 |             1
     112 |  BDA202 |      Riya |                      BDA | 2024-06-08 18:30:00.000000+0000 |             2
```

```
            ... APPLY BATCH;
cqlsh:employee> select * from Employee_info where emp_id in (111,121,131,141,151) order by salary desc;

 emp_id | salary | date_of_joining                 | dept_name | designation | emp_name
--------+--------+---------------------------------+-----------+-------------+----------
    111 |  80000 | 2010-02-26 18:30:00.000000+0000 |        IT |     Manager |     John
    151 |  70000 | 2012-02-17 18:30:00.000000+0000 |        IT |     Analyst |   Davaid
    121 |  60000 | 2019-06-26 18:30:00.000000+0000 |        IT |   Developer |    James
    141 |  50000 | 2022-02-17 18:30:00.000000+0000 |        IT |     Analyst |    Priya
    131 |  40000 | 2020-01-16 18:30:00.000000+0000 |        IT |   Developer |     Riya

(5 rows)
cqlsh:employee>
```

3 MongoDB- CRUD Demonstration

I. Perform the following DB operations using MongoDB.

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

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> use myDB;
already on db myDB
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.createCollection("Student");
{ ok: 1 }
```

2. Insert appropriate values

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.insert({rollno:11,age:20,phno:9887645328,email:"abc@gmail.com",name:"ABC"})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("66028ff3b495b09bc2b90d34") }
}
```

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.insert({rollno:10,age:21,phno:9848574328,email:"efg@gmail.com",name:"EFG"})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6602902db495b09bc2b90d35") }
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.insert({rollno:12,age:21,phno:8748574328,email:"hij@gmail.com",name:"HIJ"})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6602908db495b09bc2b90d36") }
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.insert({rollno:13,age:23,phno:7748574328,email:"riya@gmail.com",name:"Riya"})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("660243e7b495b09bc2b90d37") }
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.insert({rollno:14,age:22,phno:7493574395,email:"pihu@gmail.com",name:"Pihu"})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6602440eb495b09bc2b90d38") }
}
```

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.find();
[
  {
    _id: ObjectId("66028ff3b495b09bc2b90d34"),
    rollno: 11,
    age: 20,
    phno: 9887645328,
    email: 'abc@gmail.com',
    name: 'ABC'
  },
  {
    _id: ObjectId("6602902db495b09bc2b90d35"),
    rollno: 10,
    age: 21,
    phno: 9848574328,
    email: 'efg@gmail.com',
    name: 'EFG'
  },
  {
    _id: ObjectId("6602908db495b09bc2b90d36"),
    rollno: 12,
    age: 21,
    phno: 8748574328,
    email: 'hij@gmail.com',
    name: 'HIJ'
  },
  {
    _id: ObjectId("660243e7b495b09bc2b90d37"),
    rollno: 13,
    age: 23,
    phno: 7748574328,
    email: 'riya@gmail.com',
    name: 'Riya'
  },
  {
    _id: ObjectId("6602440eb495b09bc2b90d38"),
    rollno: 14,
    age: 22,
    phno: 7493574395,
    email: 'pihu@gmail.com',
    name: 'Pihu'
  }
]
```

3. Write query to update Email-Id of a student with rollno 10.

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.update({rollno:10},{$set:{email:"efg1@gmail.com"}});
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.find();
[
  {
    _id: ObjectId("66028ff3b495b09bc2b90d34"),
    rollno: 11,
    age: 20,
    phno: 9887645328,
    email: 'abc@gmail.com',
    name: 'ABC'
  },
  {
    _id: ObjectId("6602902db495b09bc2b90d35"),
    rollno: 10,
    age: 21,
    phno: 9848574328,
    email: 'efg1@gmail.com',
    name: 'EFG'
  },
  {
    _id: ObjectId("6602908db495b09bc2b90d36"),
    rollno: 12,
    age: 21,
    phno: 8748574328,
    email: 'hij@gmail.com',
    name: 'HIJ'
  },
  {
    _id: ObjectId("660243e7b495b09bc2b90d37"),
    rollno: 13,
    age: 23,
    phno: 7748574328,
    email: 'riya@gmail.com',
    name: 'Riya'
  },
  {
    _id: ObjectId("6602440eb495b09bc2b90d38"),
    rollno: 14,
    age: 22,
    phno: 7493574395,
    email: 'pihu@gmail.com',
    name: 'Pihu'
  }
]
```

4. . Replace the student name from “ABC” to “FEM” of rollno 11

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.update({rollno:11},{$set:{name:"FEM"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.find();
[
  {
    _id: ObjectId("66028ff3b495b09bc2b90d34"),
    rollno: 11,
    age: 20,
    phno: 9887645328,
    email: 'abc@gmail.com',
    name: 'FEM'
  },
  {
    _id: ObjectId("6602902db495b09bc2b90d35"),
    rollno: 10,
    age: 21,
    phno: 9848574328,
    email: 'efg1@gmail.com',
    name: 'EFG'
  },
  {
    _id: ObjectId("6602908db495b09bc2b90d36"),
    rollno: 12,
    age: 21,
    phno: 8748574328,
    email: 'hij@gmail.com',
    name: 'HIJ'
  },
  {
    _id: ObjectId("660243e7b495b09bc2b90d37"),
    rollno: 13,
    age: 23,
    phno: 7748574328,
    email: 'riya@gmail.com',
    name: 'Riya'
  },
  {
    _id: ObjectId("6602440eb495b09bc2b90d38"),
    rollno: 14,
    age: 22,
    phno: 7493574395,
    email: 'pihu@gmail.com',
    name: 'Pihu'
  }
]
```

II. Perform the following DB operations using MongoDB.

1. Create a collection by name Customers with the following attributes.
Cust_id, Acc_Bal, Acc_Type

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.createCollection("Customers");
{ ok: 1 }
```

2. Insert at least 5 values into the table

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Customers.insert({Cust_id:"100",Acc_Bal:1500,Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("66024cefb495b09bc2b90d40") }
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Customers.insert({Cust_id:"200",Acc_Bal:7000,Acc_Type:"A"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("66024cfeb495b09bc2b90d41") }
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Customers.insert({Cust_id:"200",Acc_Bal:200,Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("66024d0bb495b09bc2b90d42") }
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Customers.insert({Cust_id:"100",Acc_Bal:30000,Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("66024d29b495b09bc2b90d43") }
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Customers.insert({Cust_id:"200",Acc_Bal:5000,Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("66024d3eb495b09bc2b90d44") }
}
```

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

4. Determine Minimum and Maximum account balance for each customer_i

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Customers.aggregate([{$match: { Acc_Type: "Z" }},{$group: {_id: "$Cust_id",totalBalance: { $sum: "$Acc_Bal" }}},{$match: { totalBalance: { $gt: 1200 } } }]);
[
  { _id: '100', totalBalance: 31500 },
  { _id: '200', totalBalance: 5200 }
]
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Customers.aggregate([{$group:{_id:"$Cust_id",minimun:{$min:"$Acc_Bal"},maximum:{$max:"$Acc_Bal"}}}]);
[
  { _id: '100', minimun: 1500, maximum: 30000 },
  { _id: '200', minimun: 200, maximum: 7000 }
]
```

I. Perform the following DB operations using MongoDB.

5. Display Student Name and grade(Add if grade is not present)where the _id column is 1.

db.Student.updateMany({},{ $set: { "grade": "A" } })

db.Student.find({"rollno":11},{"name":1,"grade":1});

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.updateMany({},{ $set: { "grade": "A" } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 5,
  modifiedCount: 5,
  upsertedCount: 0
}
```

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.find();
[
  {
    _id: ObjectId("66028ff3b495b09bc2b90d34"),
    rollno: 11,
    age: 20,
    phno: 9887645328,
    email: 'abc@gmail.com',
    name: 'FEM',
    grade: 'A'
  },
  {
    _id: ObjectId("6602902db495b09bc2b90d35"),
    rollno: 10,
    age: 21,
    phno: 9848574328,
    email: 'efg1@gmail.com',
    name: 'EFG',
    grade: 'A'
  },
  {
    _id: ObjectId("6602908db495b09bc2b90d36"),
    rollno: 12,
    age: 21,
    phno: 8748574328,
    email: 'hij@gmail.com',
    name: 'HIJ',
    grade: 'A'
  },
  {
    _id: ObjectId("660243e7b495b09bc2b90d37"),
    rollno: 13,
    age: 23,
    phno: 7748574328,
    email: 'riya@gmail.com',
    name: 'Riya',
    grade: 'A'
  },
  {
    _id: ObjectId("6602440eb495b09bc2b90d38"),
    rollno: 14,
    age: 22,
    phno: 7493574395,
    email: 'pihu@gmail.com',
    name: 'Pihu',
    grade: 'A'
  }
]
```

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.find({"rollno":11},{"name":1,"grade":1});
[
  {
    _id: ObjectId("66028ff3b495b09bc2b90d34"),
    name: 'FEM',
    grade: 'A'
  }
]
```

6. Update to add hobbies

db.Student.update({"rollno":10},{$set:{"hobbies":"chess"}});
db.Student.update({"rollno":11},{$set:{"hobbies":"skating"}});
db.Student.update({"rollno":12},{$set:{"hobbies":"singing"}});
db.Student.update({"rollno":13},{$set:{"hobbies":"cricket"}});

db.Student.update({"rollno":14},{$set:{"hobbies":"painting"}});

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.update({"rollno":10},{$set:{"hobbies":"chess"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.update({"rollno":11},{$set:{"hobbies":"skating"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.update({"rollno":12},{$set:{"hobbies":"singing"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.update({"rollno":13},{$set:{"hobbies":"cricket"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.update({"rollno":14},{$set:{"hobbies":"painting"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

7. Find documents where hobbies is set neither to Chess nor to Skating

db.Student.find({"hobbies": {$nin: ["chess", "skating"] }})

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.find({"hobbies": {$nin: ["chess", "skating"] }})
[
  {
    _id: ObjectId("6602908db495b09bc2b90d36"),
    rollno: 12,
    age: 21,
    phno: 8748574328,
    email: 'hij@gmail.com',
    name: 'HIJ',
    grade: 'A',
    hobbies: 'singing'
  },
  {
    _id: ObjectId("660243e7b495b09bc2b90d37"),
    rollno: 13,
    age: 23,
    phno: 7748574328,
    email: 'riya@gmail.com',
    name: 'Riya',
    grade: 'A',
    hobbies: 'cricket'
  },
  {
    _id: ObjectId("6602440eb495b09bc2b90d38"),
    rollno: 14,
    age: 22,
    phno: 7493574395,
    email: 'pihu@gmail.com',
    name: 'Pihu',
    grade: 'A',
    hobbies: 'painting'
  }
]
```

8.Find documents whose name begins with A

db.Student.find({"name": /^A/})

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.insert({rollno:15,age:23,phno:7744567428,email:"aero@gmail.com",name:"Aero"})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("660bd039aad18859618dfa6d") }
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Student.find({"name":/^A/});
[
  {
    _id: ObjectId("660bd039aad18859618dfa6d"),
    rollno: 15,
    age: 23,
    phno: 7744567428,
    email: 'aero@gmail.com',
    name: 'Aero'
  }
]
```

II. Perform the following DB operations using MongoDB.

5. Sort the documents based on Customer ID in ascending order and Account Balance in descending order

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Customers.find().sort({"Cus_id":1,"Acc_Bal":-1}).pretty();
[
  {
    _id: ObjectId("66024d29b495b09bc2b90d43"),
    Cust_id: '100',
    Acc_Bal: 30000,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("66024cfeb495b09bc2b90d41"),
    Cust_id: '200',
    Acc_Bal: 7000,
    Acc_Type: 'A'
  },
  {
    _id: ObjectId("66024d3eb495b09bc2b90d44"),
    Cust_id: '200',
    Acc_Bal: 5000,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("66024cefb495b09bc2b90d40"),
    Cust_id: '100',
    Acc_Bal: 1500,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("66024d0bb495b09bc2b90d42"),
    Cust_id: '200',
    Acc_Bal: 200,
    Acc_Type: 'Z'
  }
]
```

6. Display only 2 nd and 3 rd records from the collection

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.Customers.find().pretty().skip(1).limit(2);
[
  {
    _id: ObjectId("66024cfeb495b09bc2b90d41"),
    Cust_id: '200',
    Acc_Bal: 7000,
    Acc_Type: 'A'
  },
  {
    _id: ObjectId("66024d0bb495b09bc2b90d42"),
    Cust_id: '200',
    Acc_Bal: 200,
    Acc_Type: 'Z'
  }
]
```

III. Perform the following DB operations using MongoDB

Create a collection by the name blogPosts and it has 3 fields id, title and comments.

In the collection the comments field is an array which consists of user details. Each collection consists of two user details inside the comments array- user name and text

Demonstrate the following

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.blogPost.insertMany([{id:1, title:"Blog 1", comments:["Nice blog","Worst blog
ever","Very concise blog"]},{id:2, title:"Blog 2", comments:["Mediocre","Very detailed information","Informative and enterta
ining"]},{id:3, title:"Blog 3", comments:["Awesome","Detailed information","Informative but boring"]},{id:4, title:"Blog 4",
 comments:["Sikkkeeeee","useless","lolll, lmao"]},{id:5, title:"Blog 5", comments:["very nice","Very cool information","Info
rmative and entertaining"]}]);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("660bd77baad18859618dfa6e"),
    '1': ObjectId("660bd77baad18859618dfa6f"),
    '2': ObjectId("660bd77baad18859618dfa70"),
    '3': ObjectId("660bd77baad18859618dfa71"),
    '4': ObjectId("660bd77baad18859618dfa72")
  }
}
```

1. Adding an element into araay

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.blogPost.update({id:1},{$push:{comments:"User 1"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-5x4rch-shard-0 [primary] myDB>  db.blogPost.update({id:2},{$push:{comments:"User 2"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.blogPost.update({id:3},{$push:{comments:"User 3"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-5x4rch-shard-0 [primary] myDB>     db.blogPost.update({id:4},{$push:{comments:"User 4"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-5x4rch-shard-0 [primary] myDB>   db.blogPost.update({id:5},{$push:{comments:"User 5"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

2. Display second element

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.blogPost.aggregate([{$project:{secondElement:{$arrayElemAt:["$comments",1]}}}]
)
[
  {
    _id: ObjectId("660bd77baad18859618dfa6e"),
    secondElement: 'Worst blog ever'
  },
  {
    _id: ObjectId("660bd77baad18859618dfa6f"),
    secondElement: 'Very detailed information'
  },
  {
    _id: ObjectId("660bd77baad18859618dfa70"),
    secondElement: 'Detailed information'
  },
  {
    _id: ObjectId("660bd77baad18859618dfa71"),
    secondElement: 'useless'
  },
  {
    _id: ObjectId("660bd77baad18859618dfa72"),
    secondElement: 'Very cool information'
  }
]
```

3. Display size of the array

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.blogPost.aggregate([{$project:{arraySize:{$size:"$comments"}}}])
[
  { _id: ObjectId("660bd77baad18859618dfa6e"), arraySize: 4 },
  { _id: ObjectId("660bd77baad18859618dfa6f"), arraySize: 4 },
  { _id: ObjectId("660bd77baad18859618dfa70"), arraySize: 4 },
  { _id: ObjectId("660bd77baad18859618dfa71"), arraySize: 4 },
  { _id: ObjectId("660bd77baad18859618dfa72"), arraySize: 4 }
]
```

4. Display first two elements of the array

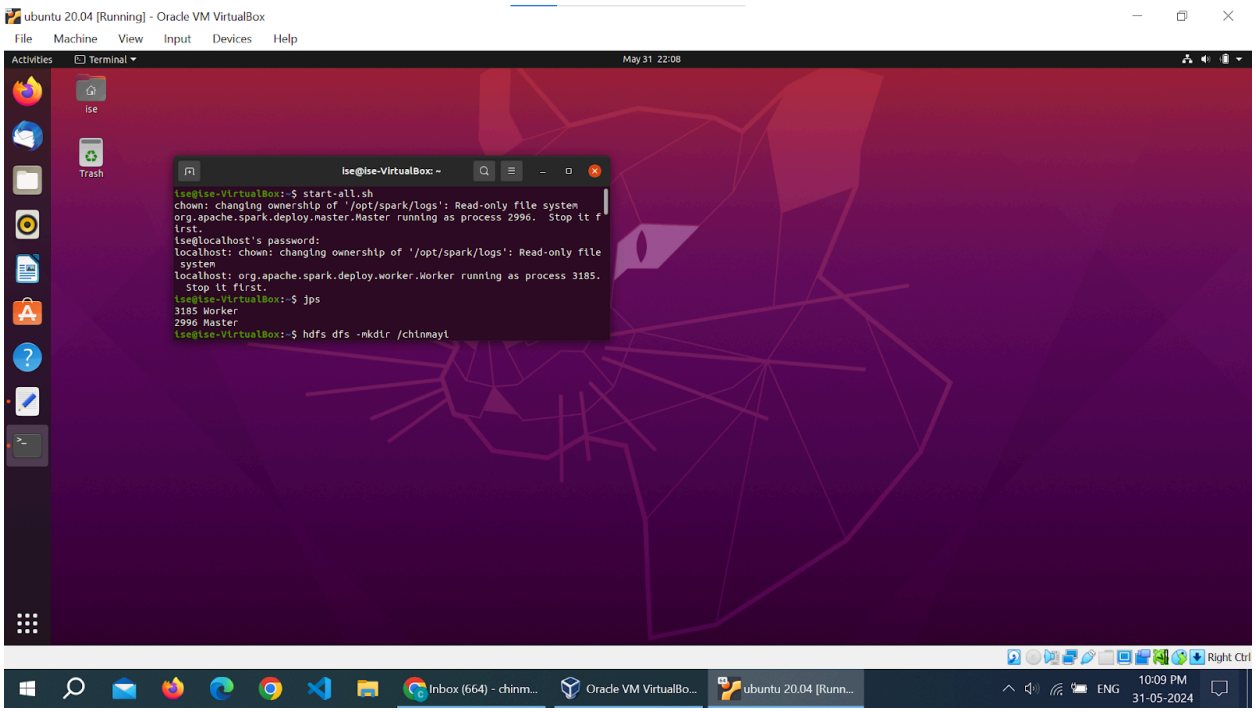
```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.blogPost.aggregate([{$project:{secondElement:{$arrayElemAt:["$comments",1]}}}]
)
[
  {
    _id: ObjectId("660bd77baad18859618dfa6e"),
    secondElement: 'Worst blog ever'
  },
  {
    _id: ObjectId("660bd77baad18859618dfa6f"),
    secondElement: 'Very detailed information'
  },
  {
    _id: ObjectId("660bd77baad18859618dfa70"),
    secondElement: 'Detailed information'
  },
  {
    _id: ObjectId("660bd77baad18859618dfa71"),
    secondElement: 'useless'
  },
  {
    _id: ObjectId("660bd77baad18859618dfa72"),
    secondElement: 'Very cool information'
  }
]
```

5. Update the document with id 4 and replace the element present in 1st index position of the array with another array

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.blogPost.update({id:4},{$set:{"comments.1":["hello","nice blog"]}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
Atlas atlas-5x4rch-shard-0 [primary] myDB> db.blogPost.find();
[
  {
    _id: ObjectId("660bd77baad18859618dfa6e"),
    id: 1,
    title: 'Blog 1',
    comments: [ 'Nice blog', 'Worst blog ever', 'Very concise blog', 'User 1' ]
  },
  {
    _id: ObjectId("660bd77baad18859618dfa6f"),
    id: 2,
    title: 'Blog 2',
    comments: [
      'Mediocre',
      'Very detailed information',
      'Informative and entertaining',
      'User 2'
    ]
  },
  {
    _id: ObjectId("660bd77baad18859618dfa70"),
    id: 3,
    title: 'Blog 3',
    comments: [
      'Awesome',
      'Detailed information',
      'Informative but boring',
      'User 3'
    ]
  },
  {
    _id: ObjectId("660bd77baad18859618dfa71"),
    id: 4,
    title: 'Blog 4',
    comments: [ 'Sikkkeeeee', [ 'hello', 'nice blog' ], 'lolll, lmao', 'User 4' ]
  },
  {
    _id: ObjectId("660bd77baad18859618dfa72"),
    id: 5,
    title: 'Blog 5',
    comments: [
      'very nice',
      'Very cool information',
      'Informative and entertaining',
      'User 5'
    ]
  }
]
```

4. Screenshot of Hadoop installed



5 Execution of HDFS Commands for interaction with Hadoop

Environment. (Minimum 10 commands to be executed).

1. mkdir: Hadoop HDFS mkdir Command Usage

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -mkdir /abcd
```

2. lsHadoop HDFS ls Command Usage

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /abcd
```

3. putHadoop HDFS put Command Usage

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -put /home/hadoop/D
esktop/Welcome.txt /abcd/WC.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /abcd
Found 1 items
-rw-r--r--   1 hadoop supergroup         19 2024-05-14 14:19 /abcd/WC.txt
```

4. copyFromLocal
Hadoop HDFS copyFromLocal Command

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -put /home/hadoop/D
esktop/Welcome.txt /abcd/WC2.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -cat /abcd/WC2.txt
Welocme to hadoop.
This is the content of Welcome file.
```

5. getHadoop HDFS get Command Usage
i.Hadoop HDFS get Command Example

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -get /abcd/WC.txt /
home/hadoop/Downloads/WWC.txt
```

ii.Hadoop HDFS get Command Example

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -getmerge /abcd/WC.
txt /abcd/WC2.txt /home/hadoop/Desktop/Merge.txt
```

iii. Hadoop HDFS get Command

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -getfacl /abcd/
# file: /abcd
# owner: hadoop
# group: supergroup
user::rwx
group::r-x
other::r-x
```

6. copyToLocalHadoop

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -copyToLocal /abcd/
WC.txt /home/hadoop/Desktop
```

7. Cat Hadoop HDFS cat Command Usage

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -cat /abcd/WC.txt
Welocme to hadoop.
```

8. mvHadoop HDFS mv Command Usage

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -mv /abcd /FFF
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /FFF
Found 2 items
-rw-r--r--   1 hadoop supergroup         19 2024-05-14 14:19 /FFF/WC.txt
-rw-r--r--   1 hadoop supergroup         56 2024-05-14 14:24 /FFF/WC2.txt
```

9. Cp Hadoop HDFS

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -mkdir /CSE
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -mkdir /CSE/InnerDi
rectory
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /CSE
Found 1 items
drwxr-xr-x   - hadoop supergroup          0 2024-05-14 14:43 /CSE/InnerDirectory
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -mkdir /LLL
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /LLL
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -cp /CSE/ /LLL
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /LLL
Found 1 items
drwxr-xr-x   - hadoop supergroup          0 2024-05-14 14:46 /LLL/CSE
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /LLL/CSE
Found 1 items
drwxr-xr-x   - hadoop supergroup          0 2024-05-14 14:46 /LLL/CSE/InnerDirec
tory
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$
```

6. Implement WordCount Program on Hadoop framework

1. Wordcount Program
Mapper Code: You have to copy paste this program into the WCMapper Java Class file.

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

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

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

                String line = value.toString();

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

Reducer Code: You have to copy paste this program into the WCReducer Java Class file

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

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

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

                int count = 0;

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

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

Driver Code: You have to copy paste this program into the WCDriver Java Class file.

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

public class WCDriver extends Configured implements Tool {

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

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

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

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ sudo snap install eclipse --classic
[sudo] password for hadoop:
eclipse 2024-03 from Snapcrafters* installed
```

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ 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 [bmscecse-HP-Elite-Tower-800-G9-Desktop-PC]
Starting resourcemanager
Starting nodemanagers
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ sudo snap install eclipse --classic
snap "eclipse" is already installed, see 'snap help refresh'
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ jps
4688 ResourceManager
5315 Jps
4405 SecondaryNameNode
3944 NameNode
4844 NodeManager
4127 DataNode
```

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ su hadoop
Password:
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ sudo update-alternatives --config java
There is only one alternative in link group java (providing /usr/bin/java): /usr/lib/jvm/java-11-openjdk-amd64/bin/java
Nothing to configure.
```

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -mkdir /rgs
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -copyFromLocal /home/hadoop/Desktop/sample.txt /rgs/test.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop jar /home/hadoop/Desktop/WC.jar WCDriver /rgs/test.txt /output
2024-05-21 15:35:13,854 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2024-05-21 15:35:13,892 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2024-05-21 15:35:13,892 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2024-05-21 15:35:13,898 WARN impl.MetricsSystemImpl: JobTracker metrics system already initialized!
2024-05-21 15:35:13,959 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2024-05-21 15:35:14,008 INFO mapred.FileInputFormat: Total input files to process : 1
2024-05-21 15:35:14,039 INFO mapreduce.JobSubmitter: number of splits:1
2024-05-21 15:35:14,100 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local311325126_0001
2024-05-21 15:35:14,100 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-05-21 15:35:14,156 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2024-05-21 15:35:14,158 INFO mapreduce.Job: Running job: job_local311325126_0001
2024-05-21 15:35:14,158 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2024-05-21 15:35:14,159 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapred.FileOutputCommitter
2024-05-21 15:35:14,161 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2024-05-21 15:35:14,161 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failures: false
2024-05-21 15:35:14,206 INFO mapred.LocalJobRunner: Waiting for map tasks
2024-05-21 15:35:14,207 INFO mapred.LocalJobRunner: Starting task: attempt_local311325126_0001_m_000000_0
2024-05-21 15:35:14,218 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2024-05-21 15:35:14,218 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failures: false
2024-05-21 15:35:14,225 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
2024-05-21 15:35:14,243 INFO mapred.MapTask: Processing split: hdfs://localhost:9000/rgs/test.txt:0+89
2024-05-21 15:35:14,253 INFO mapred.MapTask: numReduceTasks: 1
2024-05-21 15:35:14,285 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
2024-05-21 15:35:14,285 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
2024-05-21 15:35:14,285 INFO mapred.MapTask: soft limit at 83886080
2024-05-21 15:35:14,285 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600
2024-05-21 15:35:14,285 INFO mapred.MapTask: kvstart = 26214396; length = 6553600
2024-05-21 15:35:14,287 INFO mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$MapOutputBuffer
2024-05-21 15:35:14,338 INFO mapred.LocalJobRunner:
2024-05-21 15:35:14,338 INFO mapred.MapTask: Starting flush of map output
2024-05-21 15:35:14,338 INFO mapred.MapTask: Spilling map output
2024-05-21 15:35:14,338 INFO mapred.MapTask: bufstart = 0; bufend = 169; bufvoid = 104857600
2024-05-21 15:35:14,338 INFO mapred.MapTask: kvstart = 26214396(104857584); kvend = 26214320(104857280); length = 77/6553600
2024-05-21 15:35:14,341 INFO mapred.MapTask: Finished spill 0
2024-05-21 15:35:14,347 INFO mapred.Task: Task:attempt_local311325126_0001_m_000000_0 is done. And is in the process of committing
2024-05-21 15:35:14,348 INFO mapred.LocalJobRunner: hdfs://localhost:9000/rgs/test.txt:0+89
2024-05-21 15:35:14,348 INFO mapred.Task: Task 'attempt_local311325126_0001_m_000000_0' done.
2024-05-21 15:35:14,351 INFO mapred.Task: Final Counters for attempt_local311325126_0001_m_000000_0: Counters: 23
        File System Counters
                FILE: Number of bytes read=4177
                FILE: Number of bytes written=642510
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=89
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=5
```

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /output/
Found 2 items
-rw-r--r--   1 hadoop supergroup          0 2024-05-21 15:35 /output/_SUCCESS
-rw-r--r--   1 hadoop supergroup         69 2024-05-21 15:35 /output/part-00000
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -cat /output/part-00000
are     1
brother 1
family  1
hi      1
how     5
is      4
job     1
sister  1
you     1
your    4
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$
```

7. From the following link extract the weather data https://github.com/tomwhite/hadoop-

book/tree/master/input/ncdc/all

Create a Map Reduce program to

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

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));
  }
}
```

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));
  }
}
```

8. 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.

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

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

```
  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, " ");
    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));
    }
  }
}
```