VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Big Data Analytics (22CS6PEBDA)

Submitted by:

Harshitha R (1BM21CS075)

Under the Guidance of
Dr.Shyamala G
Assistant Professor, BMSCE

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
March 2024 - June 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 "**Big Data Analytics**" carried out by **Harshitha R (1BM21CS075),** 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 **Big Data Analytics - (22CS6PEBDA)** work prescribed for the said degree.

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

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

Table Of Contents

S.No.	Experiment Title			Page No
1	**Course Outcomes**			
2	**Experiments**			
	2.1	**Experiment - 1**		**1**
		2.1.1	**Question:** **Perform the following DB operations using Cassandra.** • Create a keyspace by name Employee • Create a column family by name, Employee-Info with attributes Emp_Id Primary Key, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name • Insert the values into the table in batch • Update Employee name and Department of Emp-Id 121 • Sort the details of Employee records based on salary • Alter the schema of the table Employee_Info to add a column Projects which stores a • set of Projects done by the corresponding Employee. • Update the altered table to add project names. • Create a TTL of 15 seconds to display the values of Employees.	
		2.1.2	**Code with Output**	
	2.2	**Experiment - 2**		**5**
		2.2.1	**Question:** **Perform the following DB operations using Cassandra:** • Create a keyspace by name Library • 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 • Insert the values into the table in batch • Display the details of the table created and increase the value of the counter • Write a query to show that a student with id 112 has taken a book "BDA" 2 times. • Export the created column to a csv file • Import a given csv dataset from local file system into Cassandra column family**.**	
		2.2.2	**Code with Output**	
	2.3	**Experiment – 3&4**		**7**
		2.3.1	**Question:** MongoDB- CRUD Demonstration.	

1. Course Outcomes

CO1: Apply the concepts of NoSQL, Hadoop,Spark for a given task

CO2: Analyse data analytic techniques for a given problem .

CO3: Conduct experiments using data analytics mechanisms for a given problem.

2. Experiments

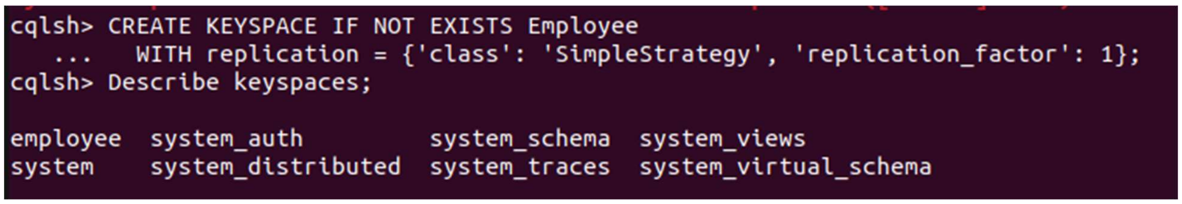
2.1 Experiment - 1

2.1.1 Question:

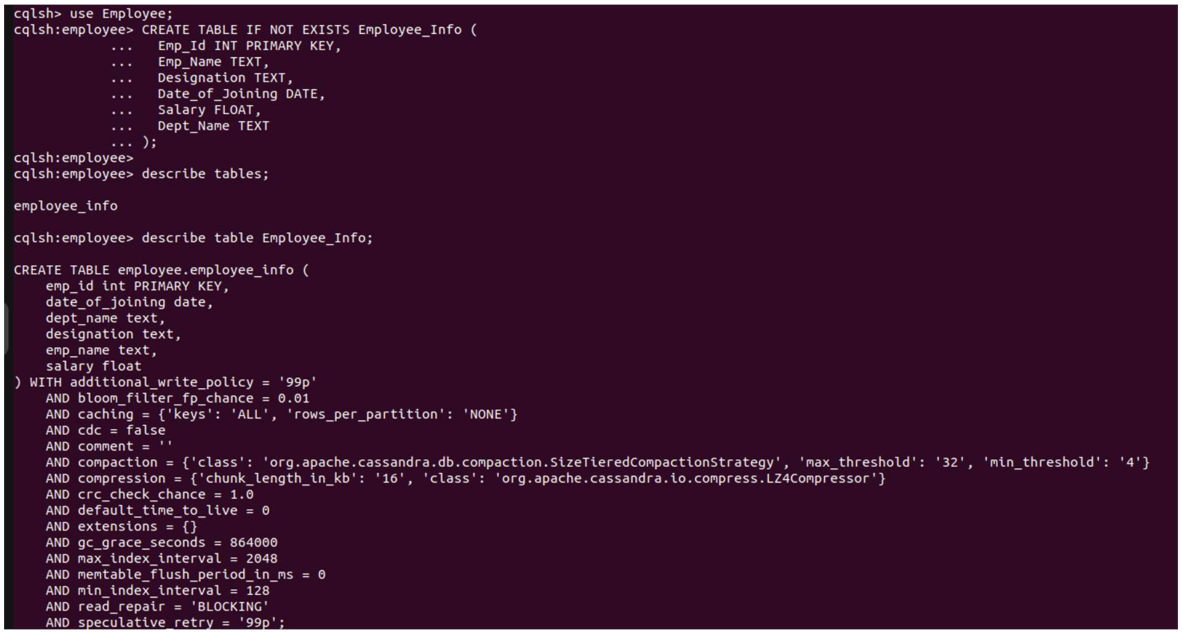
Perform the following DB operations using Cassandra.

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

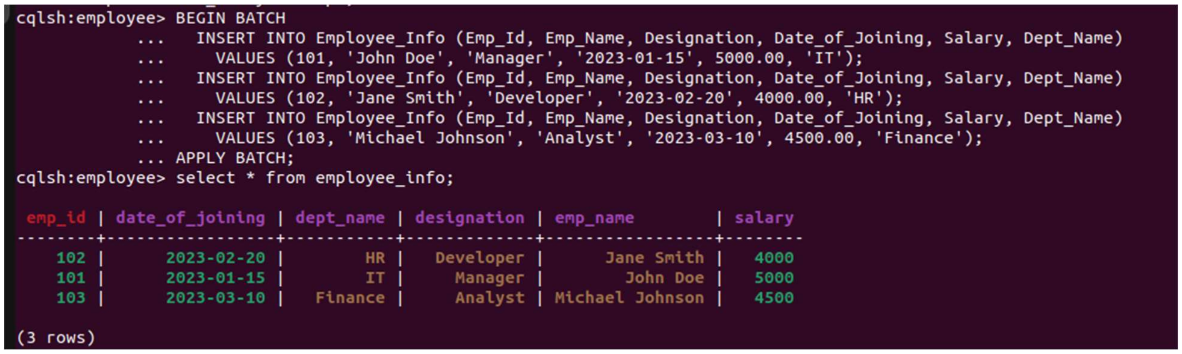
2.1.2 Code with Output:



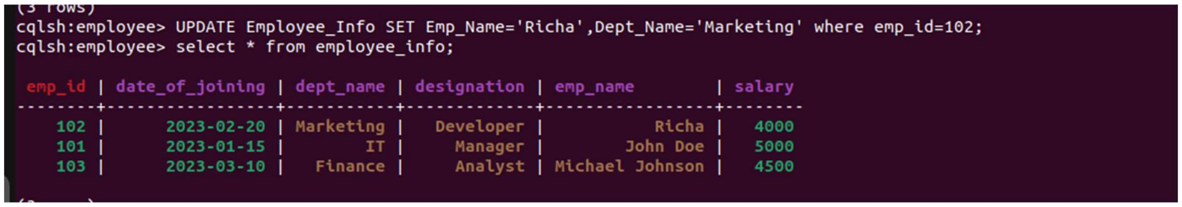
```
cqlsh> CREATE KEYSPACE IF NOT EXISTS Employee
   ...   WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
cqlsh> Describe keyspaces;

employee  system_auth         system_schema  system_views
system    system_distributed  system_traces  system_virtual_schema
```



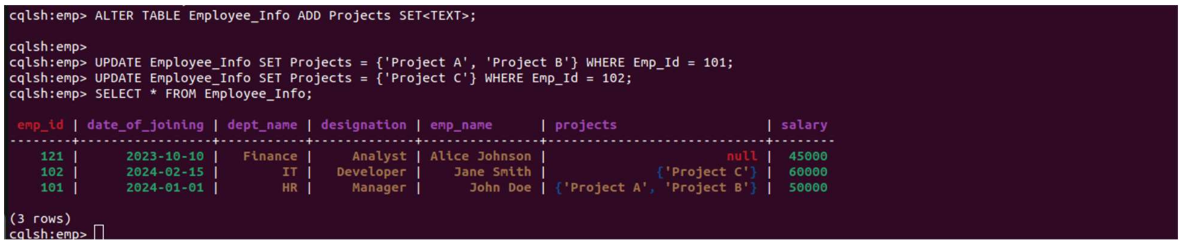
```
cqlsh> use Employee;
cqlsh:employee> CREATE TABLE IF NOT EXISTS Employee_Info (
            ...   Emp_Id INT PRIMARY KEY,
            ...   Emp_Name TEXT,
            ...   Designation TEXT,
            ...   Date_of_Joining DATE,
            ...   Salary FLOAT,
            ...   Dept_Name TEXT
            ... );
cqlsh:employee>
cqlsh:employee> describe tables;

employee_info

cqlsh:employee> describe table Employee_Info;

CREATE TABLE employee.employee_info (
    emp_id int PRIMARY KEY,
    date_of_joining date,
    dept_name text,
    designation text,
    emp_name text,
    salary float
) WITH additional_write_policy = '99p'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND cdc = false
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND extensions = {}
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99p';
```



```
cqlsh:employee> BEGIN BATCH
            ...   INSERT INTO Employee_Info (Emp_Id, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name)
            ...     VALUES (101, 'John Doe', 'Manager', '2023-01-15', 5000.00, 'IT');
            ...   INSERT INTO Employee_Info (Emp_Id, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name)
            ...     VALUES (102, 'Jane Smith', 'Developer', '2023-02-20', 4000.00, 'HR');
            ...   INSERT INTO Employee_Info (Emp_Id, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name)
            ...     VALUES (103, 'Michael Johnson', 'Analyst', '2023-03-10', 4500.00, 'Finance');
            ... APPLY BATCH;
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining | dept_name | designation | emp_name        | salary
--------+-----------------+-----------+-------------+-----------------+--------
    102 |      2023-02-20 |        HR |   Developer |      Jane Smith |   4000
    101 |      2023-01-15 |        IT |     Manager |        John Doe |   5000
    103 |      2023-03-10 |   Finance |     Analyst | Michael Johnson |   4500

(3 rows)
```



```
(3 rows)
cqlsh:employee> UPDATE Employee_Info SET Emp_Name='Richa',Dept_Name='Marketing' where emp_id=102;
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining | dept_name | designation | emp_name        | salary
--------+-----------------+-----------+-------------+-----------------+--------
    102 |      2023-02-20 | Marketing |   Developer |           Richa |   4000
    101 |      2023-01-15 |        IT |     Manager |        John Doe |   5000
    103 |      2023-03-10 |   Finance |     Analyst | Michael Johnson |   4500
```



```
cqlsh:emp> ALTER TABLE Employee_Info ADD Projects SET<TEXT>;

cqlsh:emp>
cqlsh:emp> UPDATE Employee_Info SET Projects = {'Project A', 'Project B'} WHERE Emp_Id = 101;
cqlsh:emp> UPDATE Employee_Info SET Projects = {'Project C'} WHERE Emp_Id = 102;
cqlsh:emp> SELECT * FROM Employee_Info;

 emp_id | date_of_joining | dept_name | designation | emp_name      | projects                   | salary
--------+-----------------+-----------+-------------+---------------+----------------------------+--------
    121 |      2023-10-10 |   Finance |     Analyst | Alice Johnson |                       null |  45000
    102 |      2024-02-15 |        IT |   Developer |    Jane Smith |              {'Project C'} |  60000
    101 |      2024-01-01 |        HR |     Manager |      John Doe | {'Project A', 'Project B'} |  50000

(3 rows)
cqlsh:emp>
```

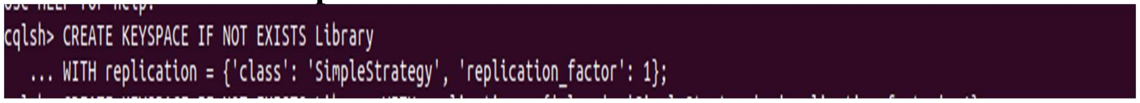
2.2 Experiment - 2

2.2.1 Question:

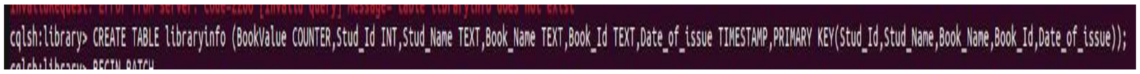
Perform the following DB operations using Cassandra:

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

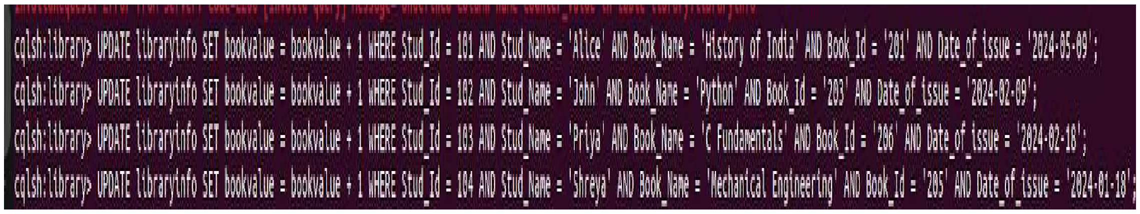
2.2.2 Code with Output:



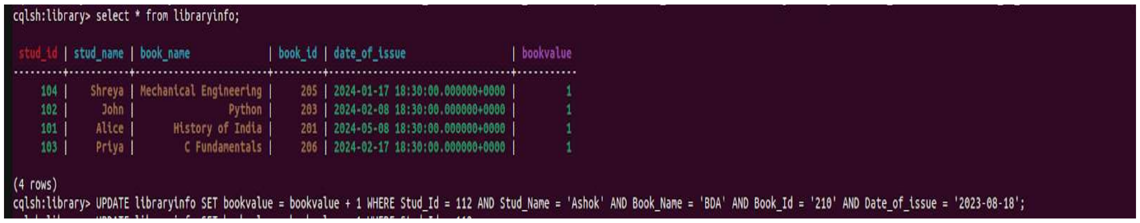
```
cqlsh> CREATE KEYSPACE IF NOT EXISTS Library
  ... WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
```



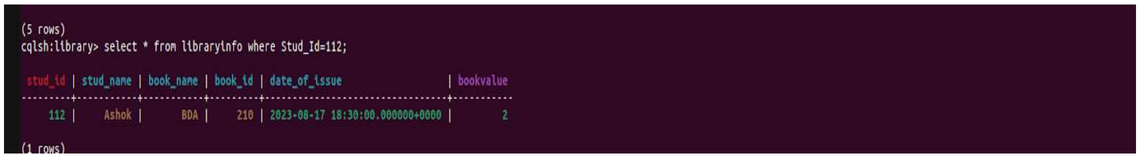
```
cqlsh:library> CREATE TABLE libraryinfo (BookValue COUNTER,Stud_Id INT,Stud_Name TEXT,Book_Name TEXT,Book_Id TEXT,Date_of_issue TIMESTAMP,PRIMARY KEY(Stud_Id,Stud_Name,Book_Name,Book_Id,Date_of_issue));
```



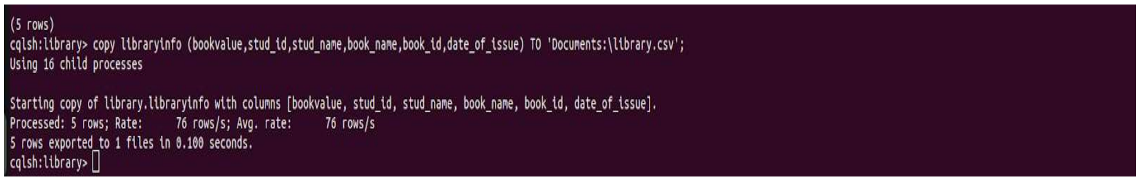
```
cqlsh:library> UPDATE libraryinfo SET bookvalue = bookvalue + 1 WHERE Stud_Id = 101 AND Stud_Name = 'Alice' AND Book_Name = 'History of India' AND Book_Id = '201' AND Date_of_issue = '2024-05-09';
cqlsh:library> UPDATE libraryinfo SET bookvalue = bookvalue + 1 WHERE Stud_Id = 102 AND Stud_Name = 'John' AND Book_Name = 'Python' AND Book_Id = '203' AND Date_of_issue = '2024-02-09';
cqlsh:library> UPDATE libraryinfo SET bookvalue = bookvalue + 1 WHERE Stud_Id = 103 AND Stud_Name = 'Priya' AND Book_Name = 'C Fundamentals' AND Book_Id = '206' AND Date_of_issue = '2024-02-18';
cqlsh:library> UPDATE libraryinfo SET bookvalue = bookvalue + 1 WHERE Stud_Id = 104 AND Stud_Name = 'Shreya' AND Book_Name = 'Mechanical Engineering' AND Book_Id = '205' AND Date_of_issue = '2024-01-18';
```



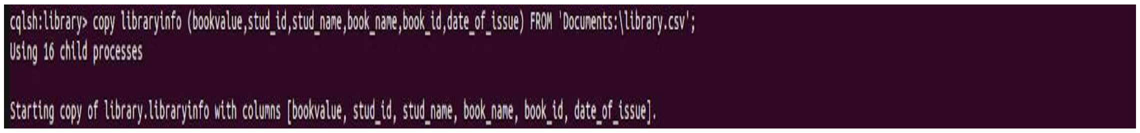
```
cqlsh:library> select * from libraryinfo;

 stud_id | stud_name | book_name              | book_id | date_of_issue                   | bookvalue
---------+-----------+------------------------+---------+---------------------------------+-----------
     104 |    Shreya | Mechanical Engineering |     205 | 2024-01-17 18:30:00.000000+0000 |         1
     102 |      John |                 Python |     203 | 2024-02-08 18:30:00.000000+0000 |         1
     101 |     Alice |       History of India |     201 | 2024-05-08 18:30:00.000000+0000 |         1
     103 |     Priya |         C Fundamentals |     206 | 2024-02-17 18:30:00.000000+0000 |         1

(4 rows)
cqlsh:library> UPDATE libraryinfo SET bookvalue = bookvalue + 1 WHERE Stud_Id = 112 AND Stud_Name = 'Ashok' AND Book_Name = 'BDA' AND Book_Id = '210' AND Date_of_issue = '2023-08-18';
```



```
(5 rows)
cqlsh:library> select * from libraryinfo where Stud_Id=112;

 stud_id | stud_name | book_name | book_id | date_of_issue                   | bookvalue
---------+-----------+-----------+---------+---------------------------------+-----------
     112 |     Ashok |       BDA |     210 | 2023-08-17 18:30:00.000000+0000 |         2

(1 rows)
```



```
(5 rows)
cqlsh:library> copy libraryinfo (bookvalue,stud_id,stud_name,book_name,book_id,date_of_issue) TO 'Documents:\library.csv';
Using 16 child processes

Starting copy of library.libraryinfo with columns [bookvalue, stud_id, stud_name, book_name, book_id, date_of_issue].
Processed: 5 rows; Rate:      76 rows/s; Avg. rate:      76 rows/s
5 rows exported to 1 files in 0.100 seconds.
cqlsh:library> 
```



```
cqlsh:library> copy libraryinfo (bookvalue,stud_id,stud_name,book_name,book_id,date_of_issue) FROM 'Documents:\library.csv';
Using 16 child processes

Starting copy of library.libraryinfo with columns [bookvalue, stud_id, stud_name, book_name, book_id, date_of_issue].
```

2.3 Experiment - 3

2.3.1 Question:

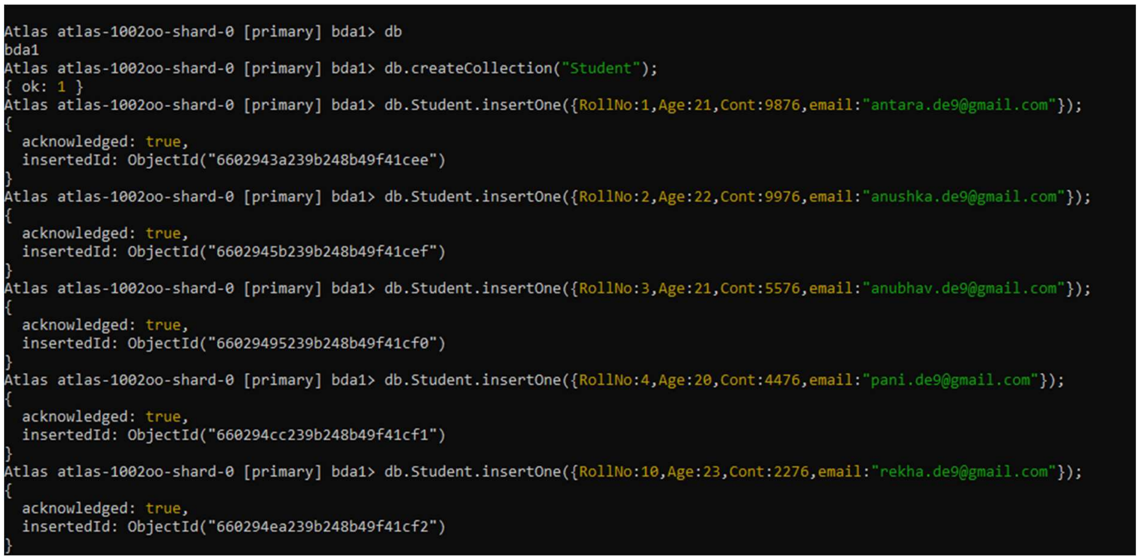
MongoDB - CRUD Demonstration.

2.3.2 Code with Output:

1.Create a database "Student" with the following attributes Rollno, Name , Age, ContactNo, Email-Id, grade, hobby:

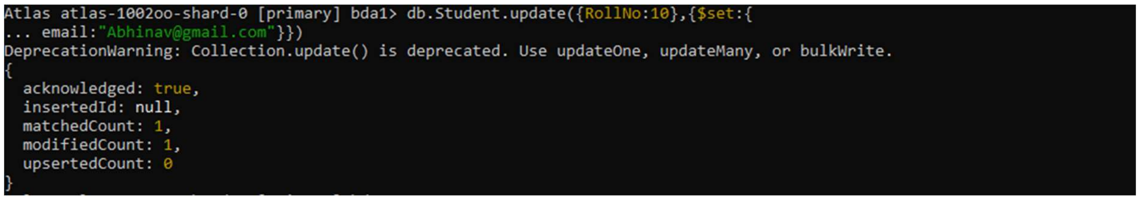
use Students

2.Insert 5 appropriate values according to the below queries.



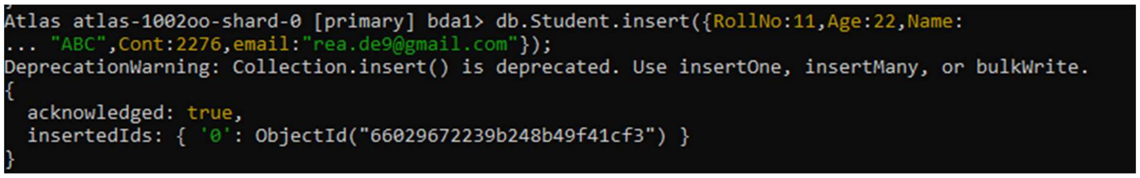
```
Atlas atlas-1002oo-shard-0 [primary] bda1> db
bda1
Atlas atlas-1002oo-shard-0 [primary] bda1> db.createCollection("Student");
{ ok: 1 }
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Student.insertOne({RollNo:1,Age:21,Cont:9876,email:"antara.de9@gmail.com"});
{
  acknowledged: true,
  insertedId: ObjectId("6602943a239b248b49f41cee")
}
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Student.insertOne({RollNo:2,Age:22,Cont:9976,email:"anushka.de9@gmail.com"});
{
  acknowledged: true,
  insertedId: ObjectId("6602945b239b248b49f41cef")
}
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Student.insertOne({RollNo:3,Age:21,Cont:5576,email:"anubhav.de9@gmail.com"});
{
  acknowledged: true,
  insertedId: ObjectId("66029495239b248b49f41cf0")
}
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Student.insertOne({RollNo:4,Age:20,Cont:4476,email:"pani.de9@gmail.com"});
{
  acknowledged: true,
  insertedId: ObjectId("660294cc239b248b49f41cf1")
}
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Student.insertOne({RollNo:10,Age:23,Cont:2276,email:"rekha.de9@gmail.com"});
{
  acknowledged: true,
  insertedId: ObjectId("660294ea239b248b49f41cf2")
}
```

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

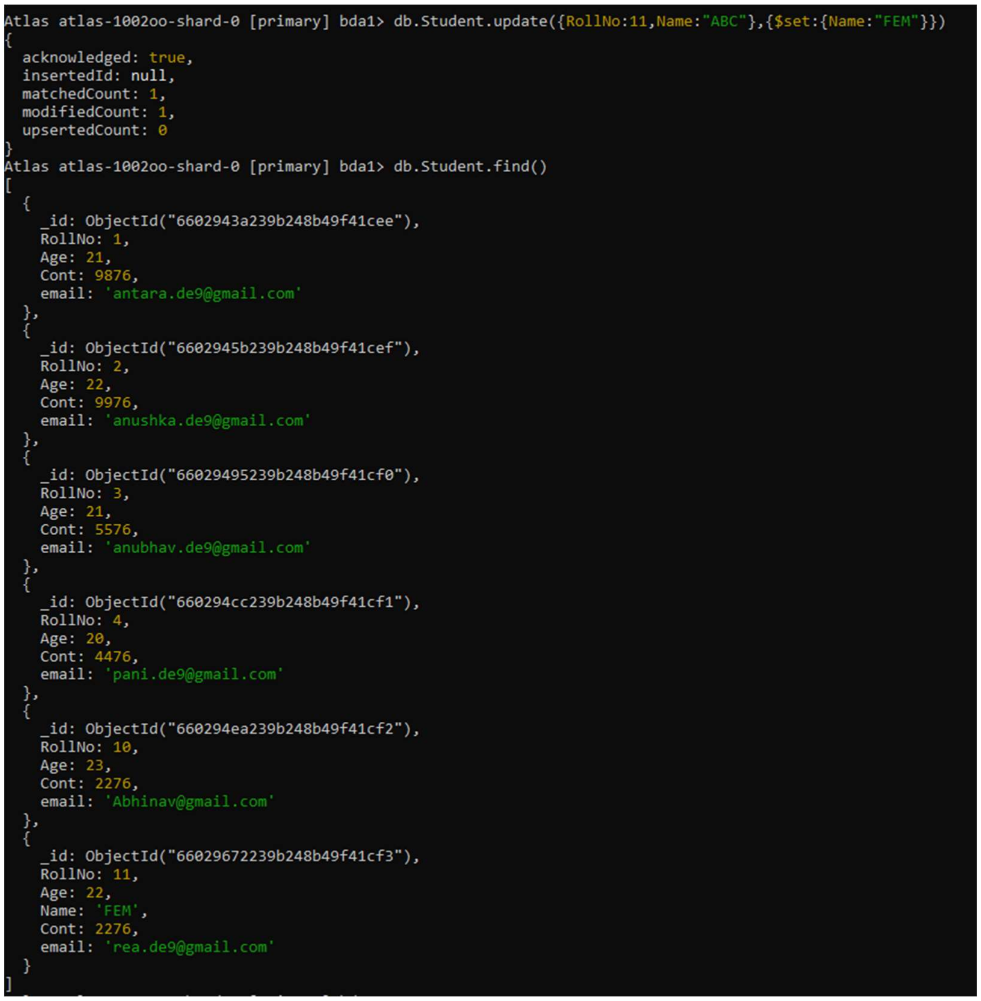


```
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Student.update({RollNo:10},{$set:{
... email:"Abhinav@gmail.com"}})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

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

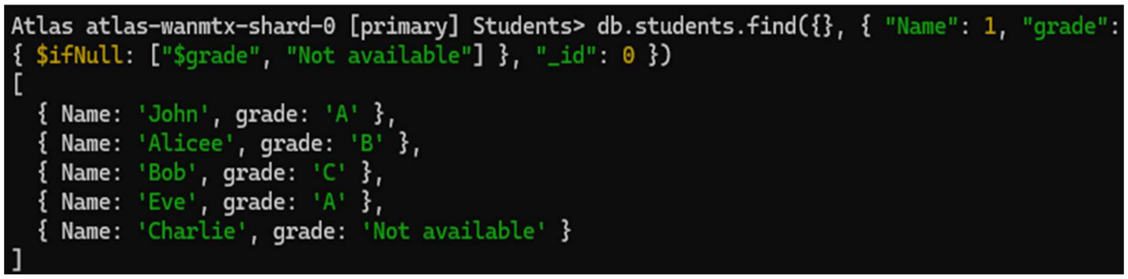


```
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Student.insert({RollNo:11,Age:22,Name:
... "ABC",Cont:2276,email:"rea.de9@gmail.com"});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("66029672239b248b49f41cf3") }
}
```



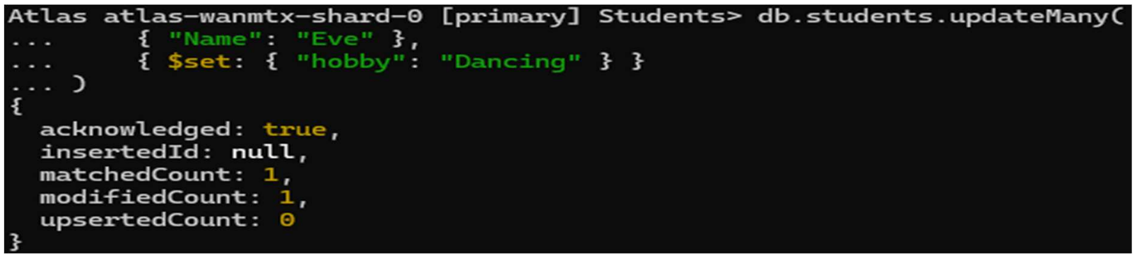
```
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Student.update({RollNo:11,Name:"ABC"},{$set:{Name:"FEM"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Student.find()
[
  {
    _id: ObjectId("6602943a239b248b49f41cee"),
    RollNo: 1,
    Age: 21,
    Cont: 9876,
    email: 'antara.de9@gmail.com'
  },
  {
    _id: ObjectId("6602945b239b248b49f41cef"),
    RollNo: 2,
    Age: 22,
    Cont: 9976,
    email: 'anushka.de9@gmail.com'
  },
  {
    _id: ObjectId("66029495239b248b49f41cf0"),
    RollNo: 3,
    Age: 21,
    Cont: 5576,
    email: 'anubhav.de9@gmail.com'
  },
  {
    _id: ObjectId("660294cc239b248b49f41cf1"),
    RollNo: 4,
    Age: 20,
    Cont: 4476,
    email: 'pani.de9@gmail.com'
  },
  {
    _id: ObjectId("660294ea239b248b49f41cf2"),
    RollNo: 10,
    Age: 23,
    Cont: 2276,
    email: 'Abhinav@gmail.com'
  },
  {
    _id: ObjectId("66029672239b248b49f41cf3"),
    RollNo: 11,
    Age: 22,
    Name: 'FEM',
    Cont: 2276,
    email: 'rea.de9@gmail.com'
  }
]
```

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



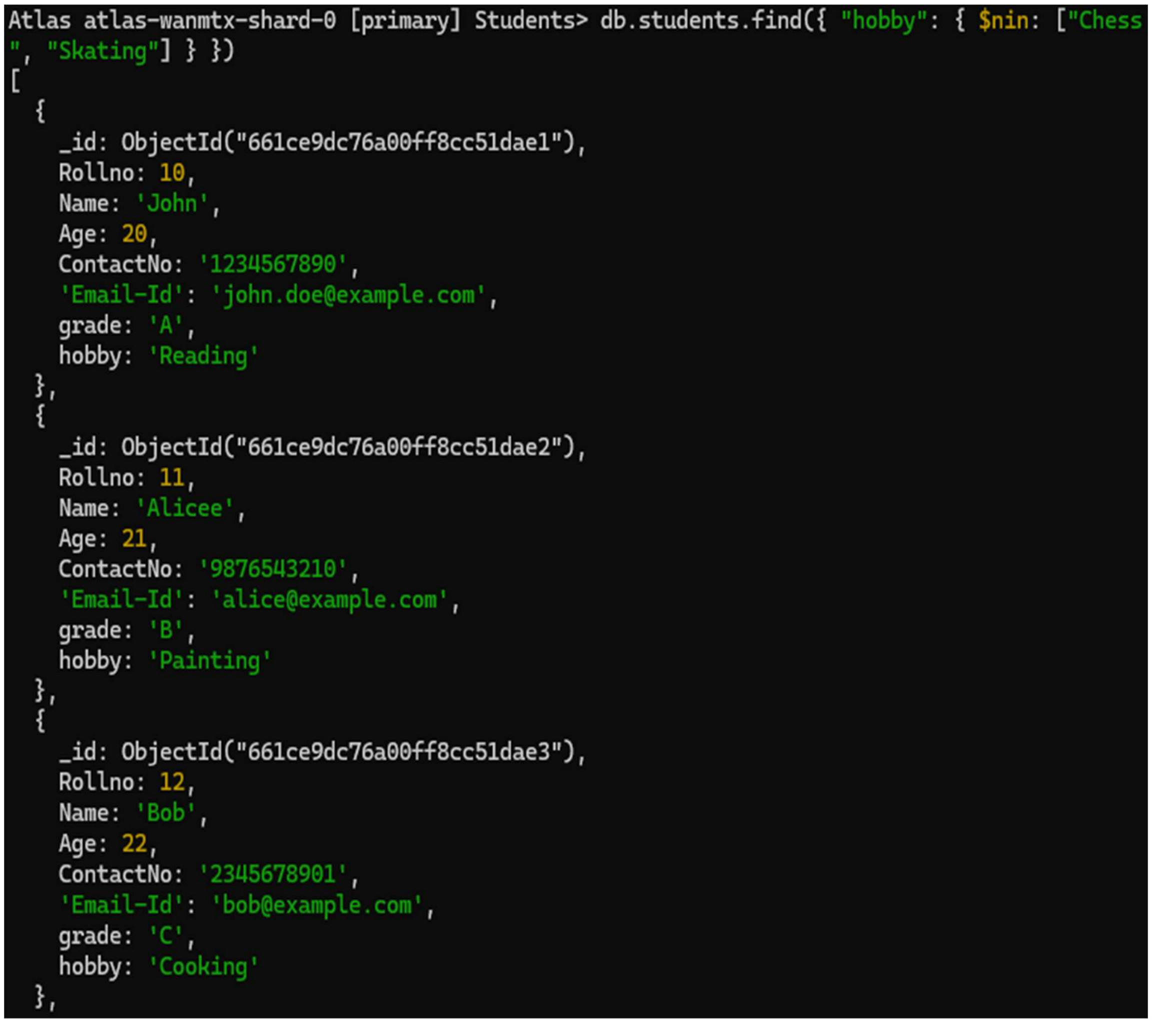
```
Atlas atlas-wanmtx-shard-0 [primary] Students> db.students.find({}, { "Name": 1, "grade":
{ $ifNull: ["$grade", "Not available"] }, "_id": 0 })
[
  { Name: 'John', grade: 'A' },
  { Name: 'Alicee', grade: 'B' },
  { Name: 'Bob', grade: 'C' },
  { Name: 'Eve', grade: 'A' },
  { Name: 'Charlie', grade: 'Not available' }
]
```

6. Update to add hobbies



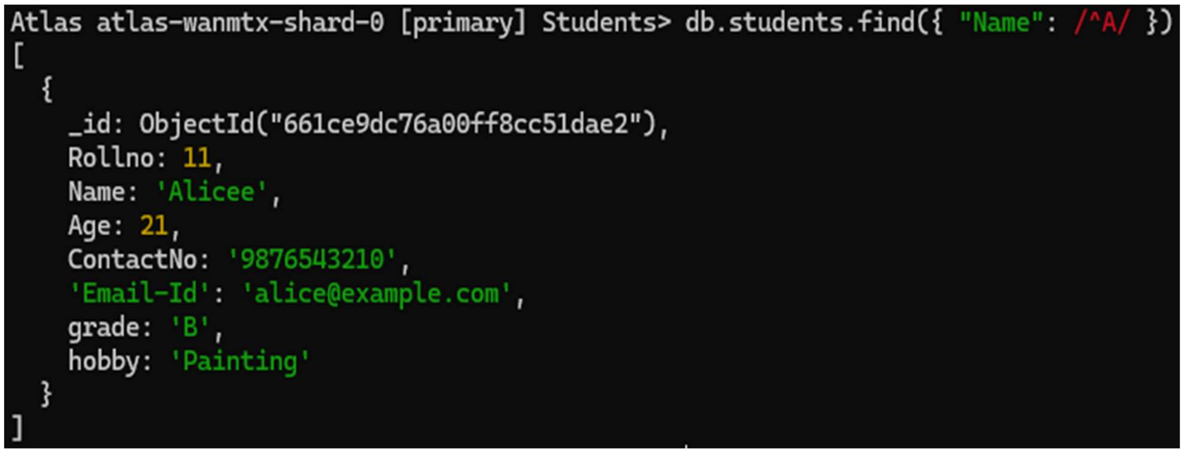
```
Atlas atlas-wanmtx-shard-0 [primary] Students> db.students.updateMany(
...      { "Name": "Eve" },
...      { $set: { "hobby": "Dancing" } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

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



```
Atlas atlas-wanmtx-shard-0 [primary] Students> db.students.find({ "hobby": { $nin: ["Chess
", "Skating"] } })
[
  {
    _id: ObjectId("661ce9dc76a00ff8cc51dae1"),
    Rollno: 10,
    Name: 'John',
    Age: 20,
    ContactNo: '1234567890',
    'Email-Id': 'john.doe@example.com',
    grade: 'A',
    hobby: 'Reading'
  },
  {
    _id: ObjectId("661ce9dc76a00ff8cc51dae2"),
    Rollno: 11,
    Name: 'Alicee',
    Age: 21,
    ContactNo: '9876543210',
    'Email-Id': 'alice@example.com',
    grade: 'B',
    hobby: 'Painting'
  },
  {
    _id: ObjectId("661ce9dc76a00ff8cc51dae3"),
    Rollno: 12,
    Name: 'Bob',
    Age: 22,
    ContactNo: '2345678901',
    'Email-Id': 'bob@example.com',
    grade: 'C',
    hobby: 'Cooking'
  },
```

8. Find documents whose name begins with A



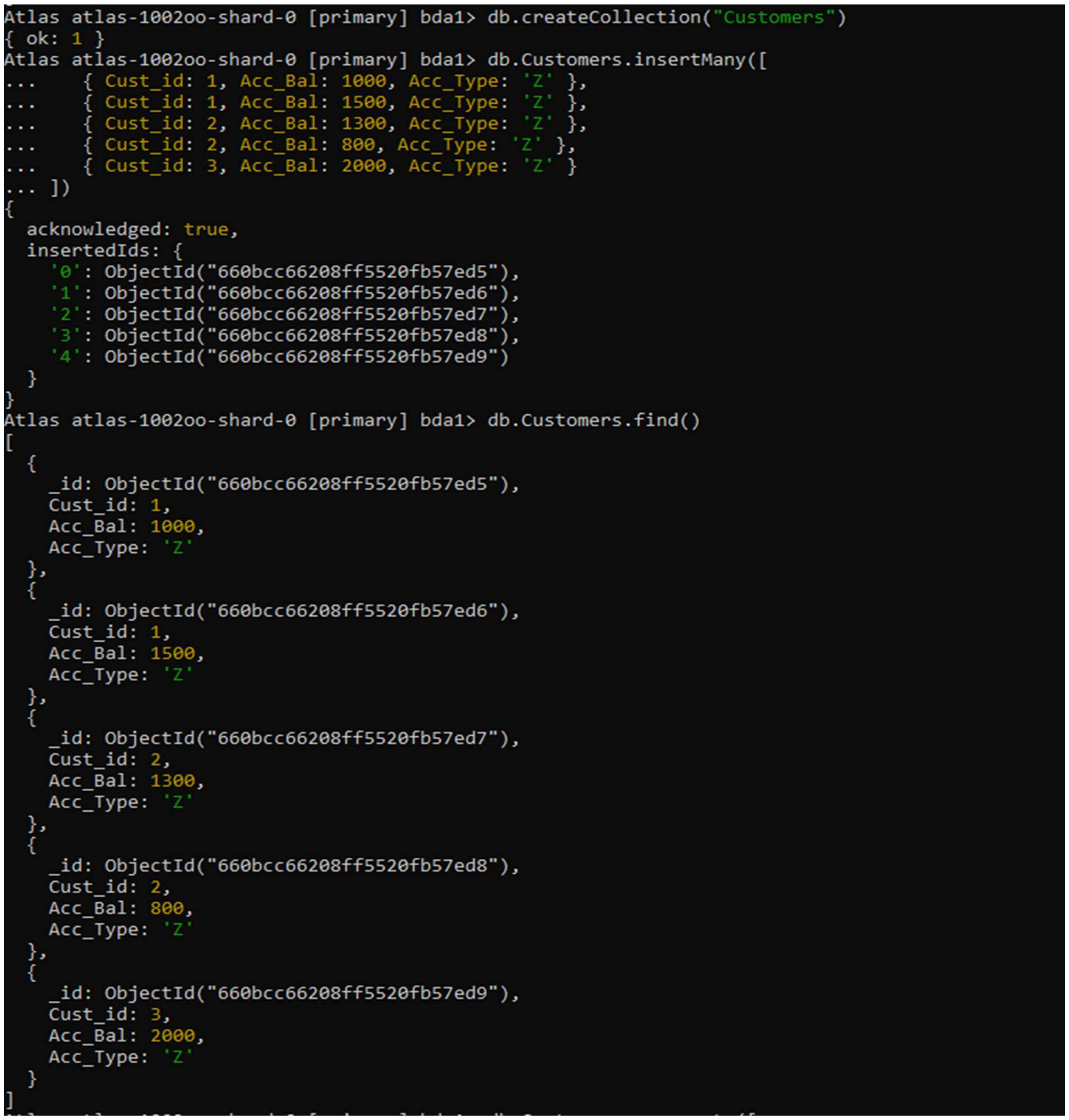
```
Atlas atlas-wanmtx-shard-0 [primary] Students> db.students.find({ "Name": /^A/ })
[
  {
    _id: ObjectId("661ce9dc76a00ff8cc51dae2"),
    Rollno: 11,
    Name: 'Alicee',
    Age: 21,
    ContactNo: '9876543210',
    'Email-Id': 'alice@example.com',
    grade: 'B',
    hobby: 'Painting'
  }
]
```

2.4 Experiment - 4

2.4.1 Question:

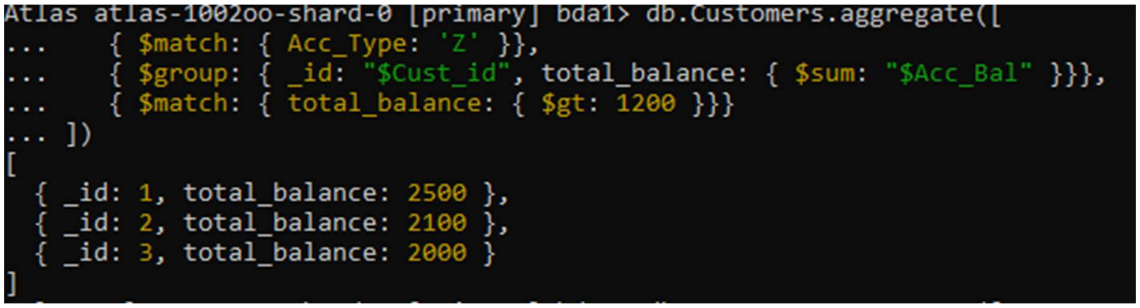
MongoDB - CRUD Demonstration.

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



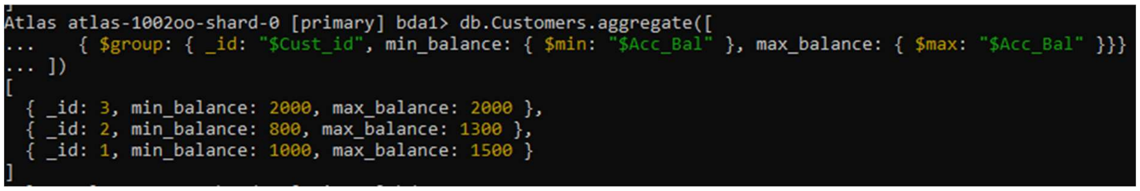
```
Atlas atlas-1002oo-shard-0 [primary] bda1> db.createCollection("Customers")
{ ok: 1 }
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Customers.insertMany([
...    { Cust_id: 1, Acc_Bal: 1000, Acc_Type: 'Z' },
...    { Cust_id: 1, Acc_Bal: 1500, Acc_Type: 'Z' },
...    { Cust_id: 2, Acc_Bal: 1300, Acc_Type: 'Z' },
...    { Cust_id: 2, Acc_Bal: 800, Acc_Type: 'Z' },
...    { Cust_id: 3, Acc_Bal: 2000, Acc_Type: 'Z' }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("660bcc66208ff5520fb57ed5"),
    '1': ObjectId("660bcc66208ff5520fb57ed6"),
    '2': ObjectId("660bcc66208ff5520fb57ed7"),
    '3': ObjectId("660bcc66208ff5520fb57ed8"),
    '4': ObjectId("660bcc66208ff5520fb57ed9")
  }
}
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Customers.find()
[
  {
    _id: ObjectId("660bcc66208ff5520fb57ed5"),
    Cust_id: 1,
    Acc_Bal: 1000,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("660bcc66208ff5520fb57ed6"),
    Cust_id: 1,
    Acc_Bal: 1500,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("660bcc66208ff5520fb57ed7"),
    Cust_id: 2,
    Acc_Bal: 1300,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("660bcc66208ff5520fb57ed8"),
    Cust_id: 2,
    Acc_Bal: 800,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("660bcc66208ff5520fb57ed9"),
    Cust_id: 3,
    Acc_Bal: 2000,
    Acc_Type: 'Z'
  }
]
```

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



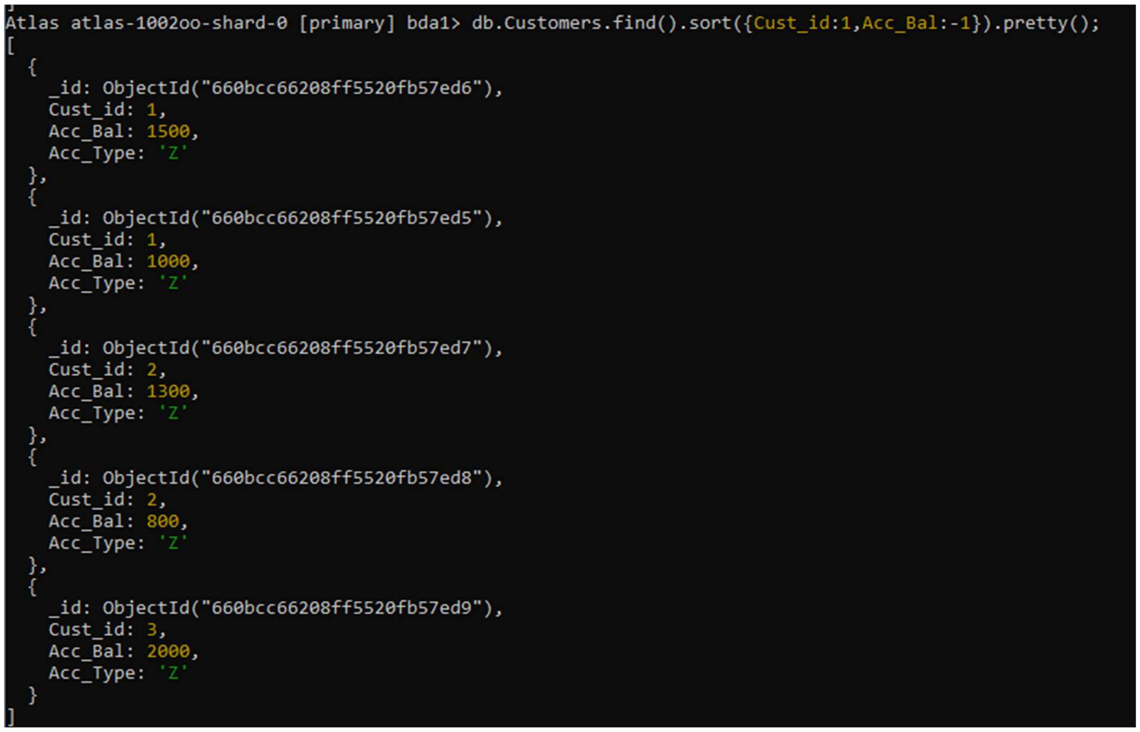
```
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Customers.aggregate([
...     { $match: { Acc_Type: 'Z' }},
...     { $group: { _id: "$Cust_id", total_balance: { $sum: "$Acc_Bal" }}},
...     { $match: { total_balance: { $gt: 1200 }}}
... ])
[
  { _id: 1, total_balance: 2500 },
  { _id: 2, total_balance: 2100 },
  { _id: 3, total_balance: 2000 }
]
```

3. Determine Minimum and Maximum account balance for each customer.



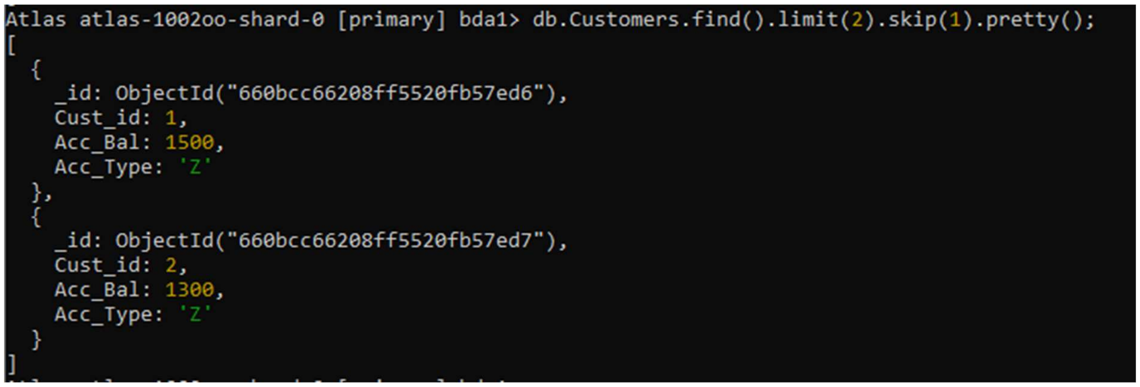
```
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Customers.aggregate([
...     { $group: { _id: "$Cust_id", min_balance: { $min: "$Acc_Bal" }, max_balance: { $max: "$Acc_Bal" }}}
... ])
[
  { _id: 3, min_balance: 2000, max_balance: 2000 },
  { _id: 2, min_balance: 800, max_balance: 1300 },
  { _id: 1, min_balance: 1000, max_balance: 1500 }
]
```

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



```
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Customers.find().sort({Cust_id:1,Acc_Bal:-1}).pretty();
[
  {
    _id: ObjectId("660bcc66208ff5520fb57ed6"),
    Cust_id: 1,
    Acc_Bal: 1500,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("660bcc66208ff5520fb57ed5"),
    Cust_id: 1,
    Acc_Bal: 1000,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("660bcc66208ff5520fb57ed7"),
    Cust_id: 2,
    Acc_Bal: 1300,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("660bcc66208ff5520fb57ed8"),
    Cust_id: 2,
    Acc_Bal: 800,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("660bcc66208ff5520fb57ed9"),
    Cust_id: 3,
    Acc_Bal: 2000,
    Acc_Type: 'Z'
  }
]
```

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



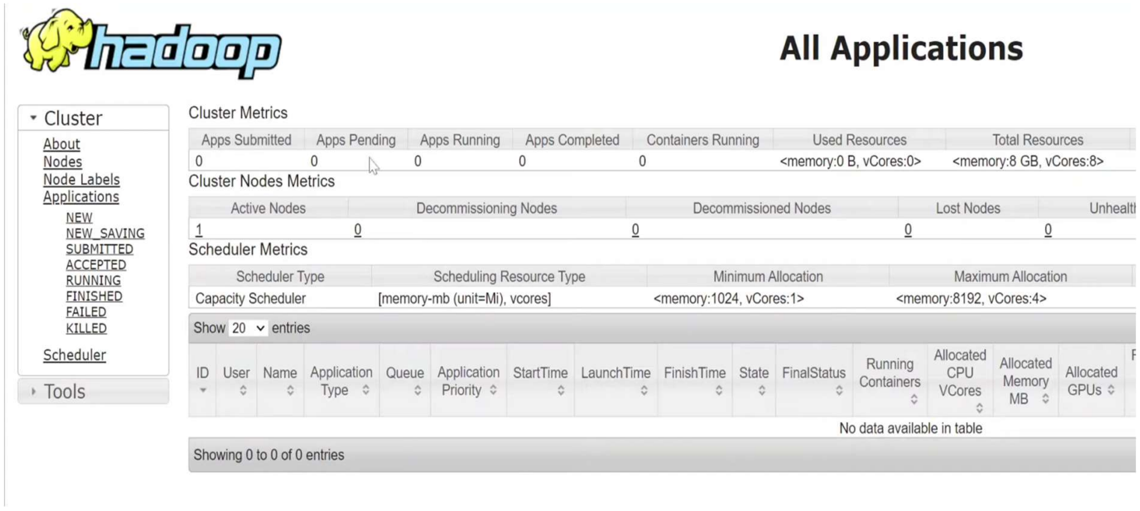
```
Atlas atlas-1002oo-shard-0 [primary] bda1> db.Customers.find().limit(2).skip(1).pretty();
[
  {
    _id: ObjectId("660bcc66208ff5520fb57ed6"),
    Cust_id: 1,
    Acc_Bal: 1500,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("660bcc66208ff5520fb57ed7"),
    Cust_id: 2,
    Acc_Bal: 1300,
    Acc_Type: 'Z'
  }
]
```

2.5 Experiment - 5

2.5.1 Question:

Hadoop Installation Screenshot

2.5.2 Screenshot:



Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Used Resources	Total Resources
0	0	0	0	0	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealt
1	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Allocated GPUs
No data available in table														

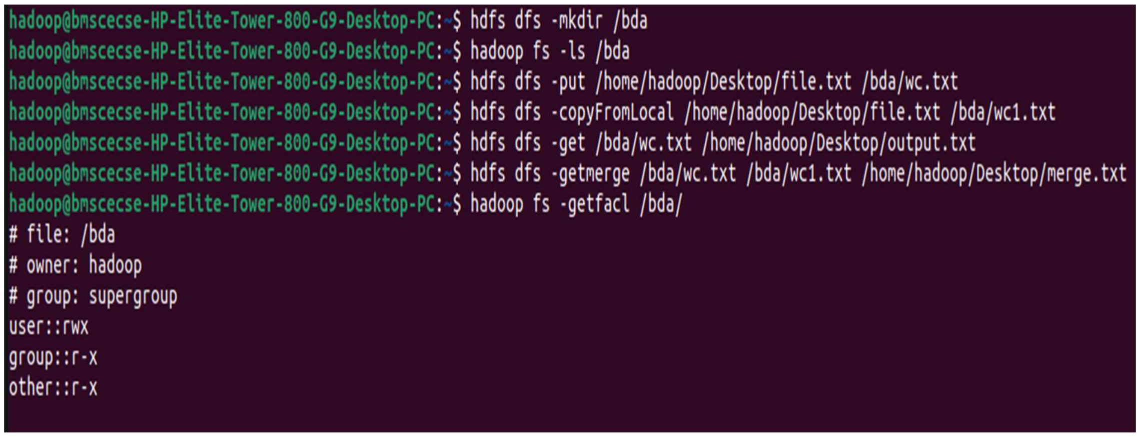
Showing 0 to 0 of 0 entries

2.6 Experiment - 6

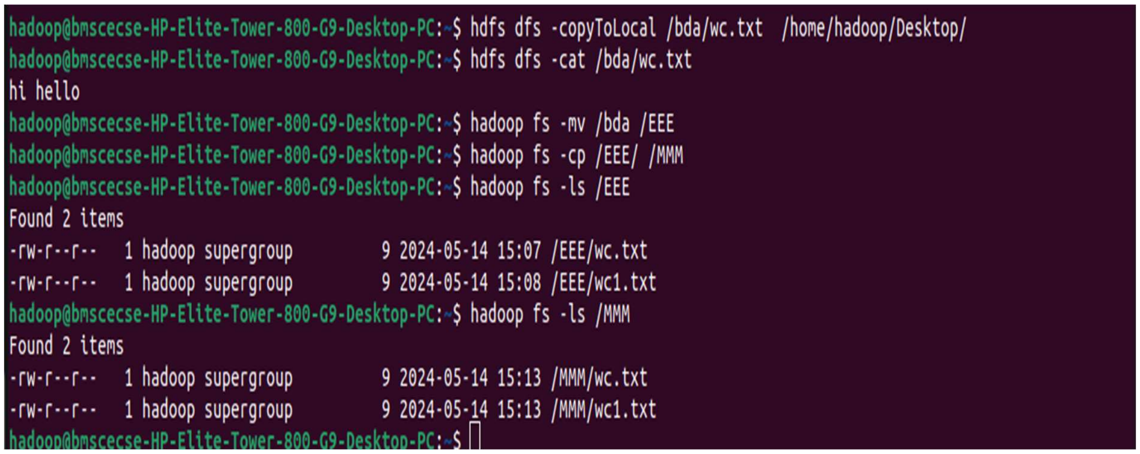
2.6.1 Question:

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

2.6.2 Code with Output:



```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -mkdir /bda
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /bda
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -put /home/hadoop/Desktop/file.txt /bda/wc.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -copyFromLocal /home/hadoop/Desktop/file.txt /bda/wc1.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -get /bda/wc.txt /home/hadoop/Desktop/output.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -getmerge /bda/wc.txt /bda/wc1.txt /home/hadoop/Desktop/merge.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -getfacl /bda/
# file: /bda
# owner: hadoop
# group: supergroup
user::rwx
group::r-x
other::r-x
```



```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -copyToLocal /bda/wc.txt  /home/hadoop/Desktop/
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hdfs dfs -cat /bda/wc.txt
hi hello
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -mv /bda /EEE
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -cp /EEE/ /MMM
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /EEE
Found 2 items
-rw-r--r--   1 hadoop supergroup          9 2024-05-14 15:07 /EEE/wc.txt
-rw-r--r--   1 hadoop supergroup          9 2024-05-14 15:08 /EEE/wc1.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /MMM
Found 2 items
-rw-r--r--   1 hadoop supergroup          9 2024-05-14 15:13 /MMM/wc.txt
-rw-r--r--   1 hadoop supergroup          9 2024-05-14 15:13 /MMM/wc1.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$
```

2.7 Experiment - 7

2.7.1 Question:

Implement Word Count Program on Hadoop framework.

2.7.2 Code with Output:

Mapper Code:

```
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> {
public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output,
Reporter rep) throws IOException
{
String line = value.toString();
for (String word : line.split(" "))
{
if (word.length() > 0)
{
output.collect(new Text(word), new IntWritable(1));
} } } }
```

Reducer Code:

```
// 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: WCDriver Java Class file.

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

2.8 Experiment - 8

2.8.1 Question:

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.
b) Find the mean max temperature for every month.

2.8.2 Code with Output:

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);
}
}
```

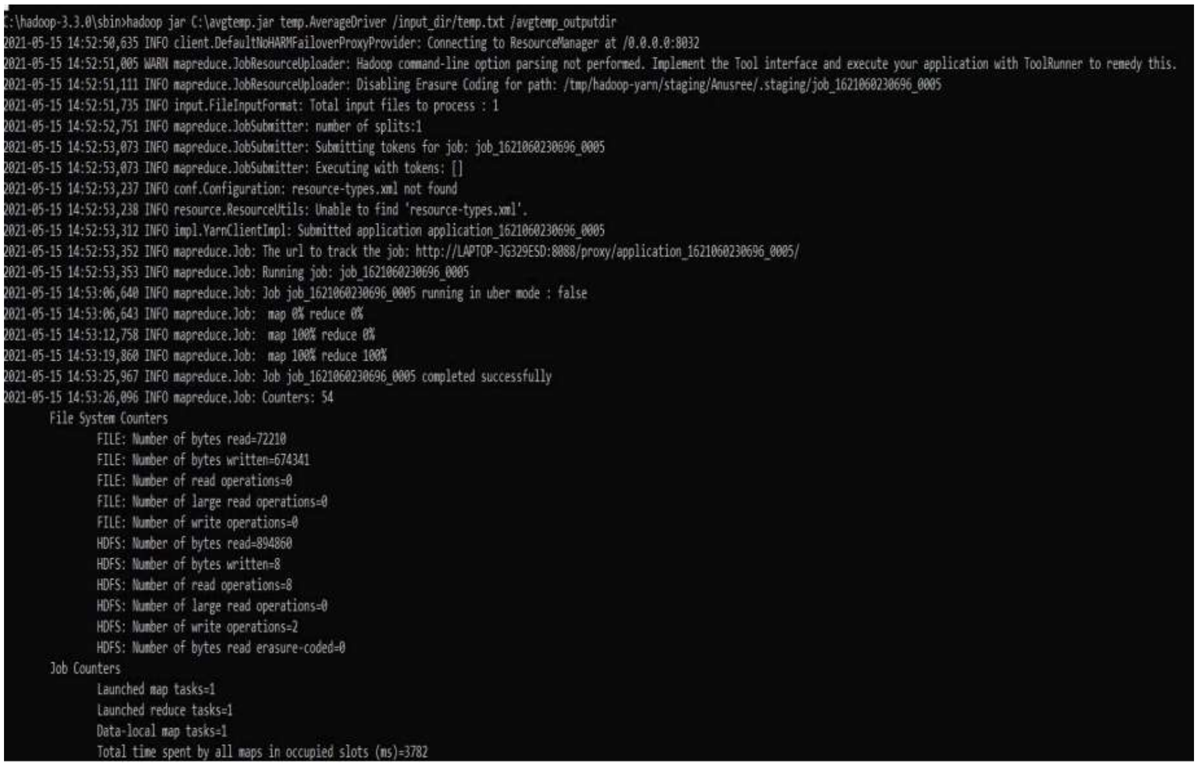
AverageMapper:

```
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 {
```

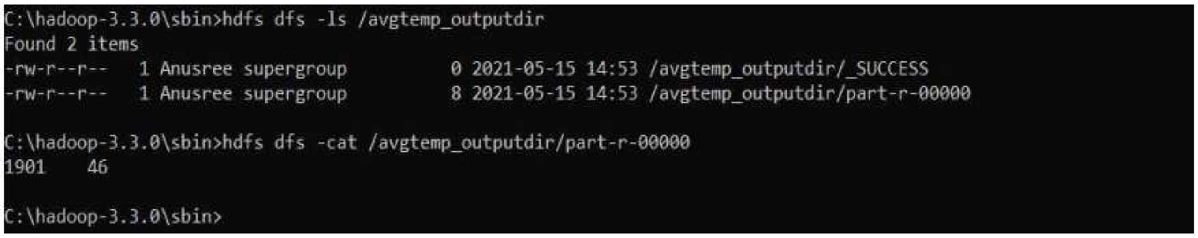
```
int temperature;
String line = value.toString();
String year = line.substring(15, 19);
if (line.charAt(87) == '+') {
temperature = Integer.parseInt(line.substring(88, 92));
} else {
temperature = Integer.parseInt(line.substring(87, 92));
}
String quality = line.substring(92, 93);
if (temperature != 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.m
reduce.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
2021-05-15 14:52:50,635 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2021-05-15 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.
2021-05-15 14:52:51,111 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/Anusree/.staging/job_1621060230696_0005
2021-05-15 14:52:51,735 INFO input.FileInputFormat: Total input files to process : 1
2021-05-15 14:52:52,751 INFO mapreduce.JobSubmitter: number of splits:1
2021-05-15 14:52:53,073 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1621060230696_0005
2021-05-15 14:52:53,073 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-05-15 14:52:53,237 INFO conf.Configuration: resource-types.xml not found
2021-05-15 14:52:53,238 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-05-15 14:52:53,312 INFO impl.YarnClientImpl: Submitted application application_1621060230696_0005
2021-05-15 14:52:53,352 INFO mapreduce.Job: The url to track the job: http://LAPTOP-JG329ESD:8088/proxy/application_1621060230696_0005/
2021-05-15 14:52:53,353 INFO mapreduce.Job: Running job: job_1621060230696_0005
2021-05-15 14:53:06,640 INFO mapreduce.Job: Job job_1621060230696_0005 running in uber mode : false
2021-05-15 14:53:06,643 INFO mapreduce.Job:  map 0% reduce 0%
2021-05-15 14:53:12,758 INFO mapreduce.Job:  map 100% reduce 0%
2021-05-15 14:53:19,860 INFO mapreduce.Job:  map 100% reduce 100%
2021-05-15 14:53:25,967 INFO mapreduce.Job: Job job_1621060230696_0005 completed successfully
2021-05-15 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
MeanMaxDriver.class

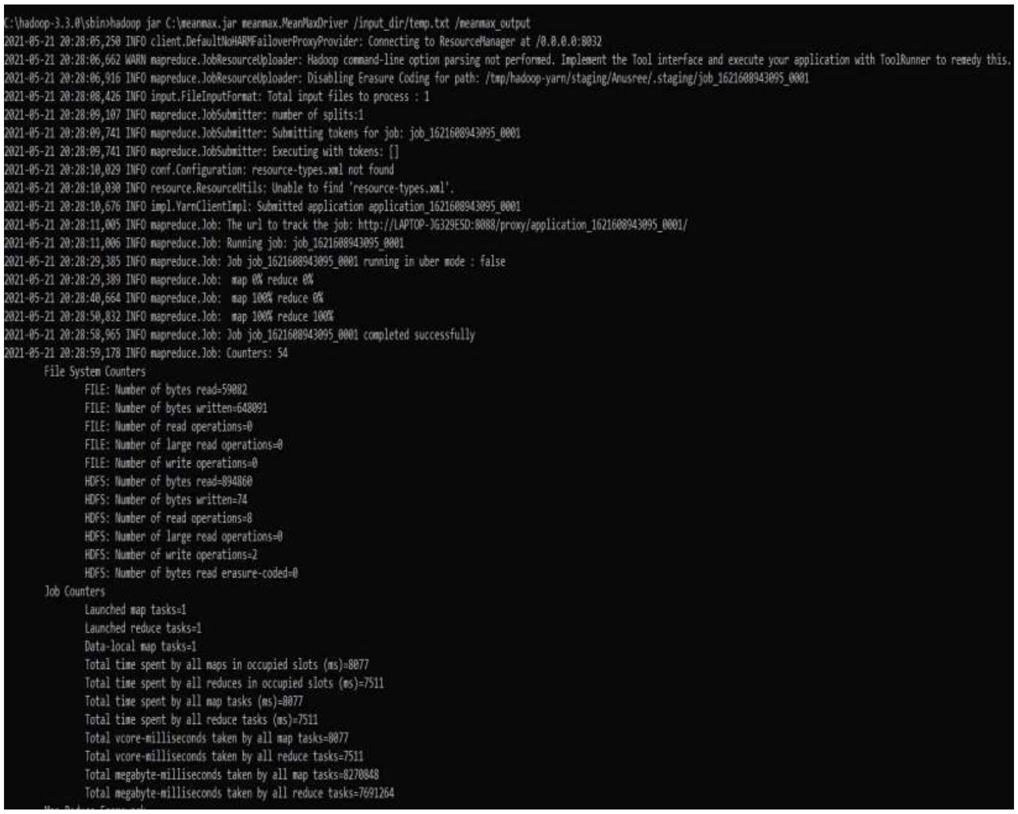
```
package meanmax;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
public 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

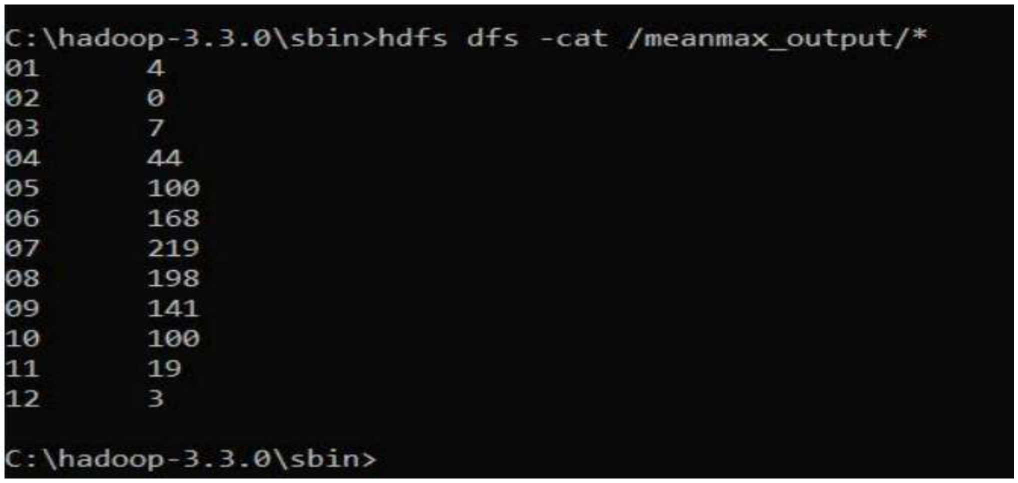
```
package meanmax;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class MeanMaxMapper extends Mapper<LongWritable, Text, Text, IntWritable> {
public static 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>
```

2.9 Experiment - 9

2.9.1 Question:

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.

2.9.2 Code with Output:

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("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

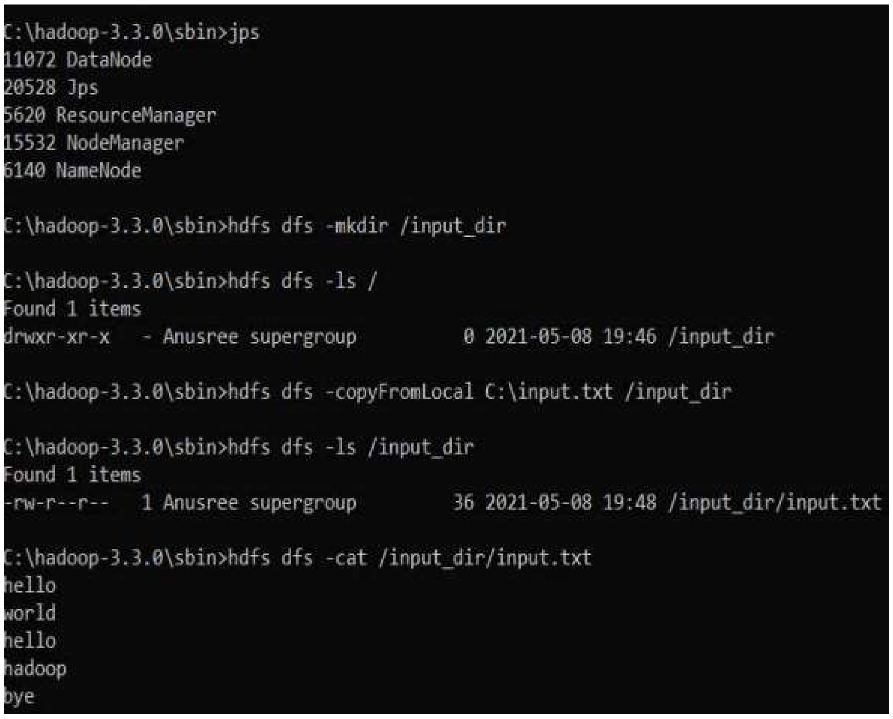
```
package samples.topn;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
public class TopNCombiner extends Reducer<Text, IntWritable, Text, IntWritable> {
public void reduce(Text key, Iterable<IntWritable> 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

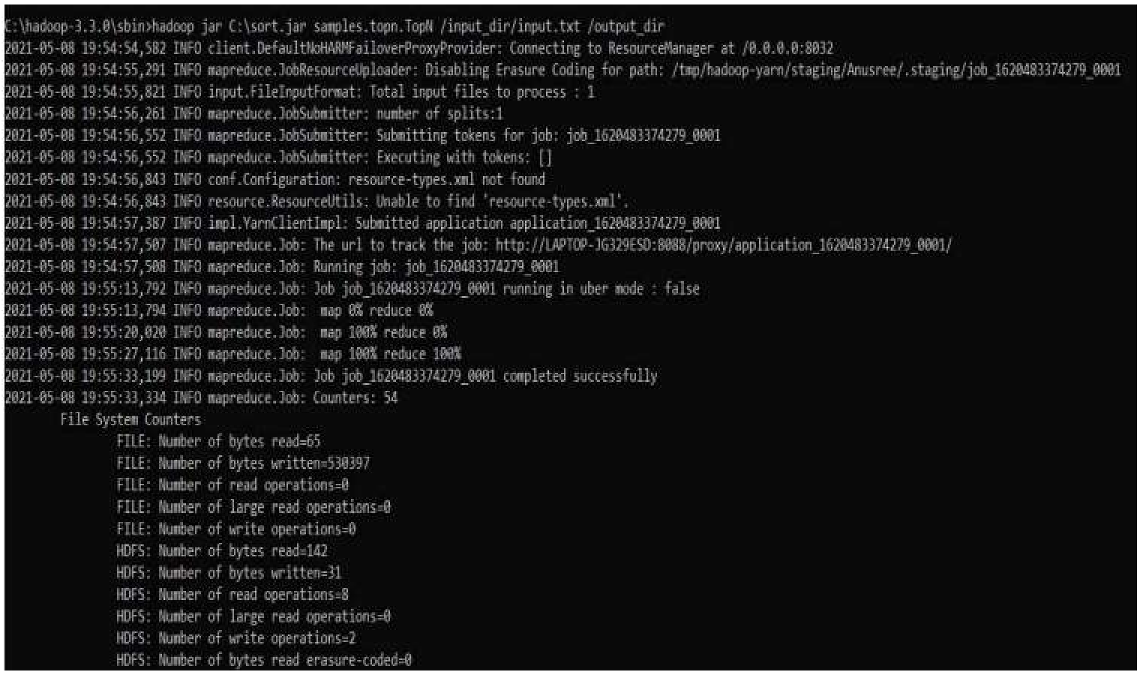
```
package samples.topn;
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class TopNMapper extends Mapper<Object, Text, Text, IntWritable> {
private static final 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

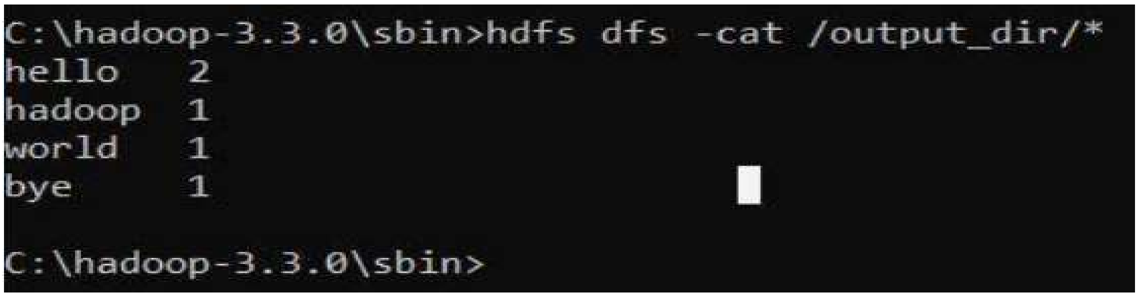
```
package samples.topn;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import utils.MiscUtils;
public class TopNReducer extends Reducer<Text, IntWritable, 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));
}
}
}
```



```
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
```



```
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.DefaultNoHARMFailoverProxyProvider: 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 19:54:57,507 INFO mapreduce.Job: The url to track the job: http://LAPTOP-JG329ESD:8088/proxy/application_1620483374279_0001/
2021-05-08 19:54:57,508 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>
```