VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT on

Big Data Analytics (23CS6PCBDA)

Submitted by

Sufiyan Desai (1BM22CS351)

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

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

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 (23CS6PCBDA)**" carried out by **Sufiyan Desai (1BM22CS351),** 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 2025. The Lab report has been approved as it satisfies the academic requirements in respect of a **Big Data Analytics - (23CS6PCBDA)** work prescribed for the said degree.

Prof. Anusha S
Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Kavitha Sooda
Professor and Head
Department of CSE
BMSCE, Bengaluru

9	Using RDD and FlatMap count how many times each word appears in a file and write out a list of words whose count is strictly greater than 4 using Spark.	**55 - 58**
10	Write a simple streaming program in Spark to receive text data streams on a particular port, perform basic text cleaning (like white space removal, stop words removal, lemmatization, etc.), and print the cleaned text on the screen. (Open Ended Question).	**59 - 60**

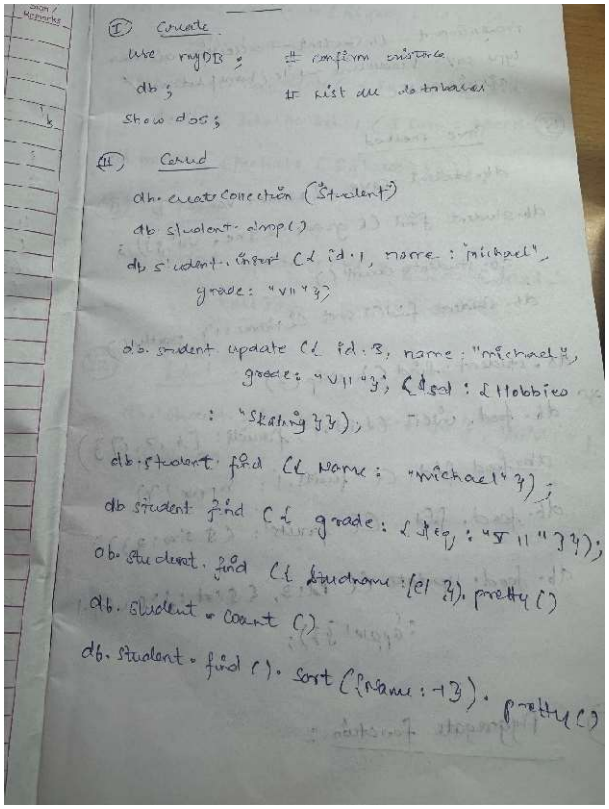
Github-link: https://github.com/sufiyandesaii/BDA_lab

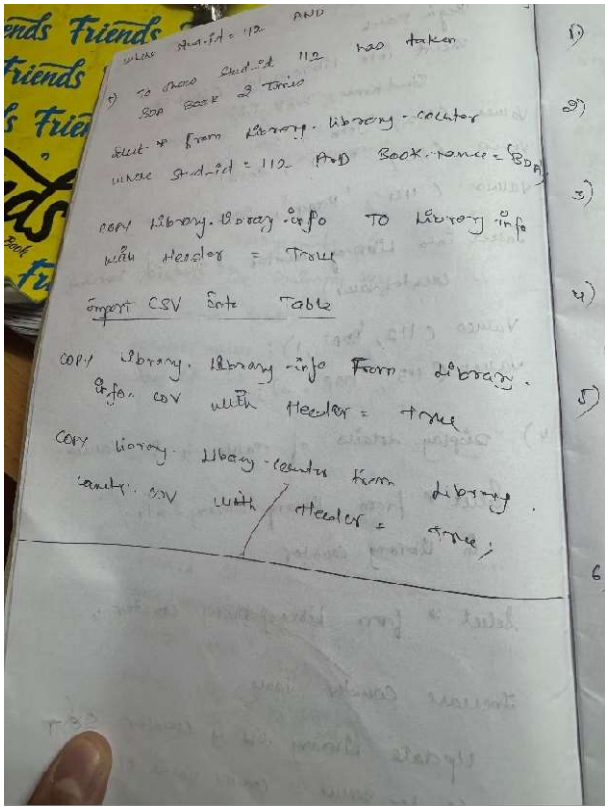
Course Outcome

CO1	Apply the concept of NoSQL, Hadoop or Spark for a given task
CO2	Analyze big data analytics mechanisms that can be applied to obtain solution for a given problem.
CO3	Design and implement solutions using data analytics mechanisms for a given problem.

Experiment-1

Q) MongoDB- CRUD Operations Demonstration (Practice and Self Study) Screenshot:





Code & 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.**

db.students.insertMany([

{ "Rollno": 10, "Name": "John", "Age": 20, "ContactNo": "1234567890", "Email-Id": "john@example.com", "grade": "A", "hobby": "Reading" },

{ "Rollno": 11, "Name": "Alice", "Age": 21, "ContactNo": "9876543210", "Email-Id": "alice@example.com", "grade": "B", "hobby": "Painting" },

{ "Rollno": 12, "Name": "Bob", "Age": 22, "ContactNo": "2345678901", "Email-Id": "bob@example.com", "grade": "C", "hobby": "Cooking" },

{ "Rollno": 13, "Name": "Eve", "Age": 23, "ContactNo": "3456789012", "Email-Id": "eve@example.com", "grade": "A" },

{ "Rollno": 14, "Name": "Charlie", "Age": 24, "ContactNo": "4567890123", "Email-Id":

```
Atlas atlas-wanmtx-shard-0 [primary] Student> use Students
switched to db Students
Atlas atlas-wanmtx-shard-0 [primary] Students> show collections

Atlas atlas-wanmtx-shard-0 [primary] Students> db.students.insertMany([
...     { "Rollno": 10, "Name": "John", "Age": 20, "ContactNo": "1234567890", "Email-Id":
"john@example.com", "grade": "A", "hobby": "Reading" },
...     { "Rollno": 11, "Name": "Alice", "Age": 21, "ContactNo": "9876543210", "Email-Id":
 "alice@example.com", "grade":
 "B", "hobby": "Painting" },
...     { "Rollno": 12, "Name": "Bob", "Age": 22, "ContactNo": "2345678901", "Email-Id": "
bob@example.com", "grade": "C", "hobby": "Cooking" },
...     { "Rollno": 13, "Name": "Eve", "Age": 23, "ContactNo": "3456789012", "Email-Id": "
eve@example.com", "grade": "A"
 },
...     { "Rollno": 14, "Name": "Charlie", "Age": 24, "ContactNo": "4567890123", "Email-Id
": "charlie@example.com", "hobby": "Gardening" }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("661ce9dc76a00ff8cc51dae1"),
    '1': ObjectId("661ce9dc76a00ff8cc51dae2"),
    '2': ObjectId("661ce9dc76a00ff8cc51dae3"),
    '3': ObjectId("661ce9dc76a00ff8cc51dae4"),
    '4': ObjectId("661ce9dc76a00ff8cc51dae5")
  }
}
```

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

db.students.updateOne(

{ "Rollno": 10 },

{ $set: { "Email-Id": "john.doe@example.com" } }

)

```
Atlas atlas-wanmtx-shard-0 [primary] Students> db.students.updateOne(
...     { "Rollno": 10 },
...     { $set: { "Email-Id": "john.doe@example.com" } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

4. **Replace the student name from “Alice” to “Alicee” of rollno 11** db.students.updateOne(

{ "Rollno": 11 },

{ $set: { "Name": "Alicee" } }

)

```
Atlas atlas-wanmtx-shard-0 [primary] Students> db.students.updateOne(
...     { "Rollno": 11 },
...     { $set: { "Name": "Alicee" } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

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

db.students.find({}, { "Name": 1, "grade": { $ifNull: ["$grade", "Not available"] }, "_id": 0 })

```
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** db.students.updateMany(

{ "Name": "Eve" },

{ $set: { "hobby": "Dancing" } }

)

```
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**

db.students.find({ "hobby": { $nin: ["Chess", "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** db.students.find({ "Name": /^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'
  }
]
```

Experiment – 2

Q) Perform the following DB operations using Cassandra

a) Create a keyspace by name **Employee**

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

c) Insert the values into the table in **batch**

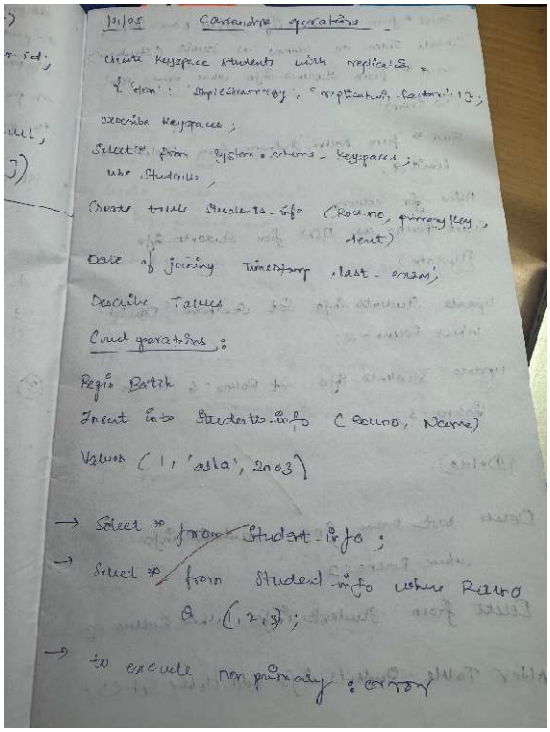
d) Update Employee name and Department of **Emp-Id 121**

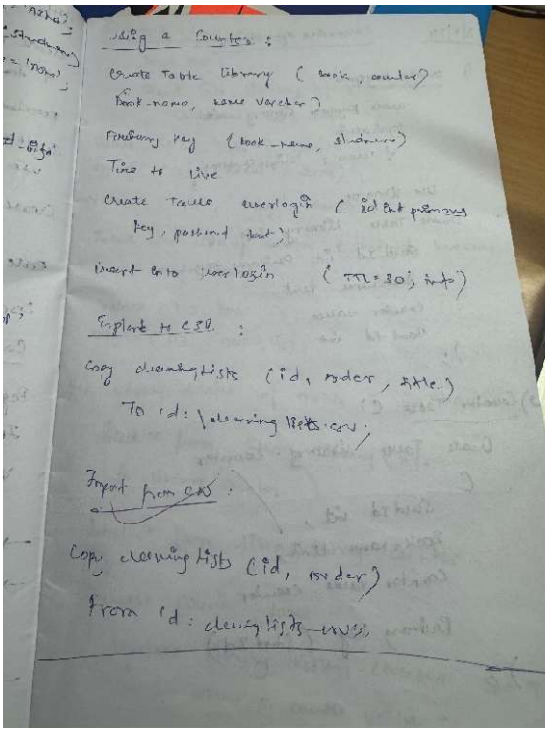
e) Sort the details of Employee records based on **salary**

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

g) Update the altered table to **add project names**

h) Create a **TTL of 15 seconds** to display the values of Employees Screen Shot:





i) Code & Output:

```
bmscecse@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.4 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> create keyspace Employee with replication = {'class':'SimpleStrategy;,;replicationfactor':1};
SyntaxException: line 1:89 mismatched input ':' expecting '}' (...with replication = {'class':'SimpleStrategy;,;replicationfactor'[:]1...)
cqlsh> create keyspace Employee WITH replication={'class':'SimpleStrategy','replicationfactor':1};
ConfigurationException: Unrecognized strategy option {replicationfactor} passed to SimpleStrategy for keyspace employee
cqlsh> create keyspace 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> CREATE TABLE IF NOT EXISTS Employee_Info(
   ... Emp_Id INT PRIMARY KEY,
   ... Emp_name TEXT,
   ... designation TEXT,
   ... date_of_joining DATE,
   ... Salary FLOAT,
   ... Dep_name TEXT,
   ... Projects SET<TEXT>);
InvalidRequest: Error from server: code=2200 [Invalid query] message="No keyspace has been specified. USE a keyspace, or explicitly specify keyspace.tablename"
cqlsh> USE eMPLOYEE
   ...
cqlsh> USE Employee
   ...
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, Dep_name TEXT, Projects SET<TEXT>);
cqlsh:employee> describe keyspace Employee

CREATE KEYSPACE employee WITH replication = {'class': 'SimpleStrategy', 'replication_factor': '1'}  AND durable_writes = true;

CREATE TABLE employee.employee_info (
    emp_id int PRIMARY KEY,
    date_of_joining date,
    dep_name text,
    designation text,
    emp_name text,
    salary float,
    projects set<text>
) 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 memtable = 'default'
    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
```

```
cqlsh:employee> update employee_info using ttl 15 set salary = 0 where emp_id = 121;
cqlsh:employee> select * from employee_info;

 emp_id | bonus | date_of_joining | dep_name    | designation | emp_name    | projects                    | salary
--------+-------+-----------------+-------------+-------------+-------------+-----------------------------+---------
    120 | 12000 |      2024-05-06 | Engineering |   Developer | Priyanka GH |  {'Project B', 'ProjectA'} |   1e+06
    123 |  null |      2024-05-07 | Engineering |    Engineer |     Sadhana | {'Project M', 'Project P'} | 1.2e+06
    122 |  null |      2024-05-06 |  Management |          HR |     Rachana | {'Project C', 'Project M'} |   9e+05
    121 | 11000 |      2024-05-06 |  Management |   Developer |      Shreya |  {'Project C', 'ProjectA'} |       0

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

 emp_id | bonus | date_of_joining | dep_name    | designation | emp_name    | projects                    | salary
--------+-------+-----------------+-------------+-------------+-------------+-----------------------------+---------
    120 | 12000 |      2024-05-06 | Engineering |   Developer | Priyanka GH |  {'Project B', 'ProjectA'} |   1e+06
    123 |  null |      2024-05-07 | Engineering |    Engineer |     Sadhana | {'Project M', 'Project P'} | 1.2e+06
    122 |  null |      2024-05-06 |  Management |          HR |     Rachana | {'Project C', 'Project M'} |   9e+05
    121 | 11000 |      2024-05-06 |  Management |   Developer |      Shreya |  {'Project C', 'ProjectA'} |    null

(4 rows)
cqlsh:employee>
```

```
   AND speculative_retry = '99p';
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining | dep_name    | designation | emp_name | projects                  | salary
--------+-----------------+-------------+-------------+----------+---------------------------+---------
    120 |      2024-05-06 | Engineering |   Developer | Priyanka |  {'Project B', 'ProjectA'} |   1e+06
    123 |      2024-05-07 | Engineering |    Engineer |  Sadhana | {'Project M', 'Project P'} | 1.2e+06
    122 |      2024-05-06 |  Management |          HR |  Rachana | {'Project C', 'Project M'} |   9e+05
    121 |      2024-05-06 |  Management |   Developer |   Shreya |  {'Project C', 'ProjectA'} |   9e+05

(4 rows)
cqlsh:employee> update employee_info set emp_name = 'Priyanka GH' Where emp_id = '120';
InvalidRequest: Error from server: code=2200 [Invalid query] message="Invalid STRING constant (120) for "emp_id" of type int"
cqlsh:employee> update employee_info set emp_name = 'Priyanka GH' Where emp_id=120;
cqlsh:employee> select * from employee_info;

 emp_id | date_of_joining | dep_name    | designation | emp_name    | projects                  | salary
--------+-----------------+-------------+-------------+-------------+---------------------------+---------
    120 |      2024-05-06 | Engineering |   Developer | Priyanka GH |  {'Project B', 'ProjectA'} |   1e+06
    123 |      2024-05-07 | Engineering |    Engineer |     Sadhana | {'Project M', 'Project P'} | 1.2e+06
    122 |      2024-05-06 |  Management |          HR |     Rachana | {'Project C', 'Project M'} |   9e+05
    121 |      2024-05-06 |  Management |   Developer |      Shreya |  {'Project C', 'ProjectA'} |   9e+05

(4 rows)
cqlsh:employee> select * from employee_info order by salary;
InvalidRequest: Error from server: code=2200 [Invalid query] message="ORDER BY is only supported when the partition key is restricted by an EQ or an IN."
cqlsh:employee> alter table employee_info add bonus INT;
cqlsh:employee> select * from employee_info;

 emp_id | bonus | date_of_joining | dep_name    | designation | emp_name    | projects                  | salary
--------+-------+-----------------+-------------+-------------+-------------+---------------------------+---------
    120 |  null |      2024-05-06 | Engineering |   Developer | Priyanka GH |  {'Project B', 'ProjectA'} |   1e+06
    123 |  null |      2024-05-07 | Engineering |    Engineer |     Sadhana | {'Project M', 'Project P'} | 1.2e+06
    122 |  null |      2024-05-06 |  Management |          HR |     Rachana | {'Project C', 'Project M'} |   9e+05
    121 |  null |      2024-05-06 |  Management |   Developer |      Shreya |  {'Project C', 'ProjectA'} |   9e+05

(4 rows)
cqlsh:employee> update employee_info set bonus = 12000 where emp_id = 120;
cqlsh:employee> select * from employee_info;

 emp_id | bonus | date_of_joining | dep_name    | designation | emp_name    | projects                  | salary
--------+-------+-----------------+-------------+-------------+-------------+---------------------------+---------
    120 | 12000 |      2024-05-06 | Engineering |   Developer | Priyanka GH |  {'Project B', 'ProjectA'} |   1e+06
    123 |  null |      2024-05-07 | Engineering |    Engineer |     Sadhana | {'Project M', 'Project P'} | 1.2e+06
    122 |  null |      2024-05-06 |  Management |          HR |     Rachana | {'Project C', 'Project M'} |   9e+05
    121 |  null |      2024-05-06 |  Management |   Developer |      Shreya |  {'Project C', 'ProjectA'} |   9e+05

(4 rows)
cqlsh:employee> update employee_info set bonus = 11000 where emp_id = 121;
cqlsh:employee> select * from employee_info using ttl 15 where emp_id = 123;
SyntaxException: line 1:28 mismatched input 'using' expecting EOF (select * from employee_info [using] ttl...)
cqlsh:employee> select * from employee_info where emp_id = 121 using ttl 15;
SyntaxException: line 1:47 no viable alternative at input 'using' (...employee_info where emp_id = 121 [using]...)
cqlsh:employee> update employee_info using ttl 15 set salary = 0 where emp_id = 121;
cqlsh:employee> select * from employee_info;
```

Experiment – 3

Q) Perform the following DB operations using Cassandra

a) Create a keyspace by name **Library**

b) 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

c) Insert the values into the table in **batch**

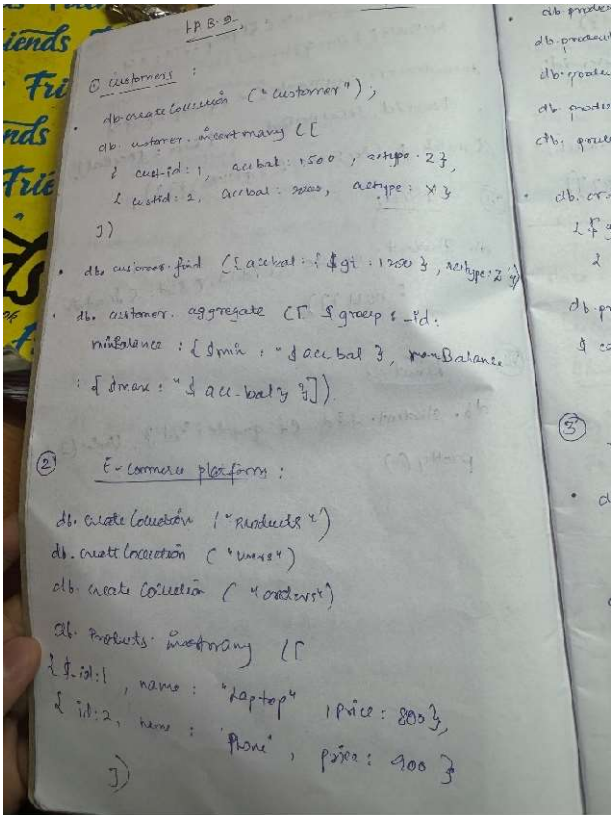
d) Display the details of the table created and **increase the value of the counter**

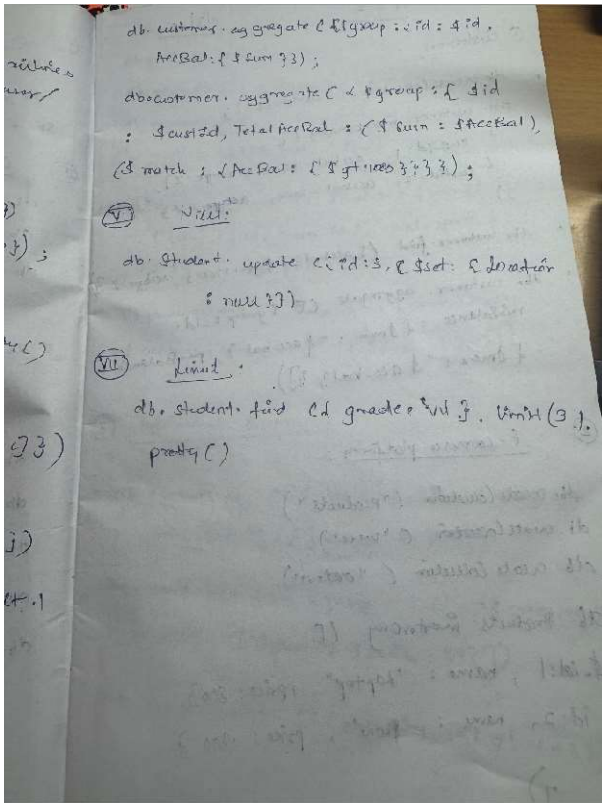
e) Write a query to show that a student with **id 112** has taken a book **"BDA" 2 times**

f) **Export** the created column to a **CSV file**

g) **Import** a given CSV dataset from **local file system** into Cassandra **column family**

Screenshot:


LAB 9.
① customers :
db.createCollection ("customer");
db.customer.insertMany ([
{ cust-id: 1, accbal: 1500, actype: 2 },
{ custid: 2, accbal: 2000, actype: X }
])
db.customers.find ({accbal: {$gt: 1500}, actype: 2})
db.customer.aggregate ([{$group: {_id:
minBalance: {$min: "$accbal"}, maxBalance
: {$max: "$accbal"}}])
② E-commerce platform :
db.createCollection ("products")
db.createCollection ("users")
db.createCollection ("orders")
db.Products.insertMany ([
{_id:1, name: "laptop", Price: 800},
{ id:2, name: "Phone", price: 400 }
])


db.customers.aggregate ([{$group: {_id: $id,
AccBal: {$sum: 1}}];
db.customer.aggregate ([{$group: {_id
: $custid, TotalAccBal: {$sum: $accbal},
{$match: {AccBal: {$gt: 1000}}}});
⑤ Null:
db.Student.update ({id:3, {$set: {branch
: null}})
⑦ Limit:
db.student.find ({grade: "VII"}).limit(3).
pretty()

Code & Output:

```
bmscecse@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.4 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> CREATE KEYSPACE Students WITH REPLICATION={
   ... 'class':'SimpleStrategy','replication_factor':1};
cqlsh> DESCRIBE KEYSPACES

students  system_auth         system_schema  system_views
system    system_distributed  system_traces  system_virtual_schema

cqlsh> SELECT * FROM system.schema_keyspaces;
InvalidRequest: Error from server: code=2200 [Invalid query] message="table schema_keyspaces does not exist"
cqlsh> use Students;
cqlsh:students> create table Students_info(Roll_No int Primary key,StudName text,DateOfJoining timestamp,last_exam_Percent double);
cqlsh:students> describe tables;

students_info

cqlsh:students> describe table students;
Table 'students' not found in keyspace 'students'
cqlsh:students> describe table students_info;

CREATE TABLE students.students_info (
    roll_no int PRIMARY KEY,
    dateofjoining timestamp,
    last_exam_percent double,
    studname text
) 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 memtable = 'default'
    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';
```

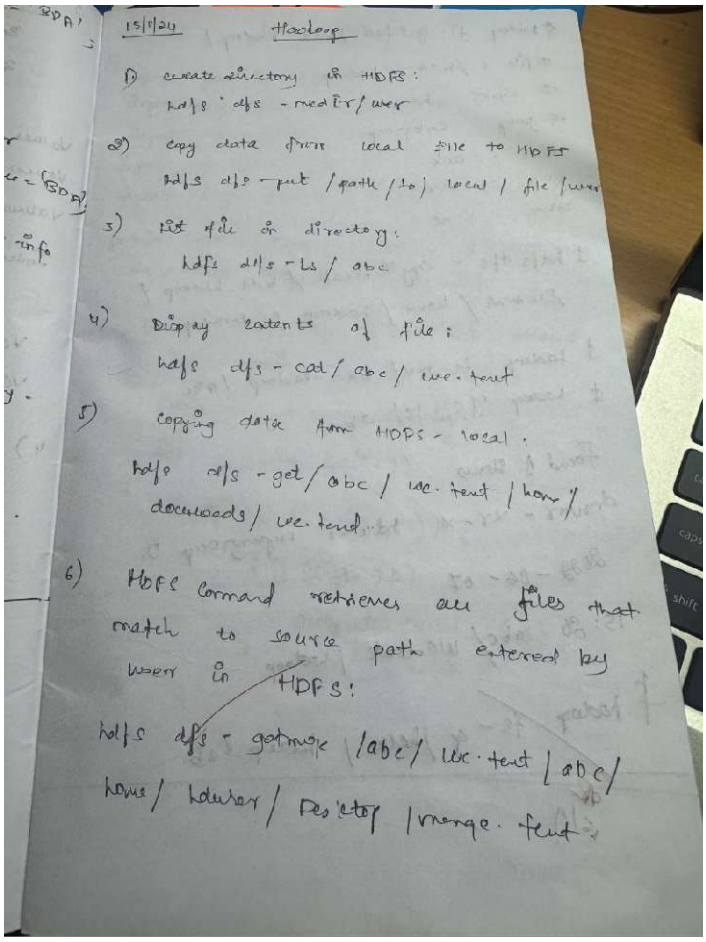
```
cqlsh:students> Begin batch insert into Students_info(Roll_no, StudName,DateOfJoining, last_exam_Percent) values(1,'Sadhana','2023-10-09', 98) insert into Students_info(Roll_no, StudName,DateOfJoining, last_exam
_Percent) values(2,'Rutu','2023-10-10', 97) insert into Students_info(Roll_no, StudName,DateOfJoining, last_exam_Percent) values(3,'Rachana','2023-10-10', 97.5) insert into Students_info(Roll_no, StudName,DateOf
Joining, last_exam_Percent) values(4,'Charu','2023-10-06', 96.5) apply batch;
cqlsh:students> select * from students_info;

 roll_no | dateofjoining                   | last_exam_percent | studname
---------+---------------------------------+-------------------+----------
       1 | 2023-10-08 18:30:00.000000+0000 |                98 |  Sadhana
       2 | 2023-10-09 18:30:00.000000+0000 |                97 |     Rutu
       4 | 2023-10-05 18:30:00.000000+0000 |              96.5 |    Charu
       3 | 2023-10-09 18:30:00.000000+0000 |              97.5 |  Rachana

(4 rows)
cqlsh:students> select * from students_info where roll_no in (1,2,3);

 roll_no | dateofjoining                   | last_exam_percent | studname
---------+---------------------------------+-------------------+----------
       1 | 2023-10-08 18:30:00.000000+0000 |                98 |  Sadhana
       2 | 2023-10-09 18:30:00.000000+0000 |                97 |     Rutu
       3 | 2023-10-09 18:30:00.000000+0000 |              97.5 |  Rachana

(3 rows)
cqlsh:students> select * from students_info where Studname='Charu';
InvalidRequest: Error from server: code=2200 [Invalid query] message="Cannot execute this query as it might involve data filtering and thus may have unpredictable performance. If you want to execute this query d
espite the performance unpredictability, use ALLOW FILTERING"
cqlsh:students> create index on Students_info(StudName);
cqlsh:students> select * from students_info where Studname='Charu';

 roll_no | dateofjoining                   | last_exam_percent | studname
---------+---------------------------------+-------------------+----------
       4 | 2023-10-05 18:30:00.000000+0000 |              96.5 |    Charu

(1 rows)
cqlsh:students> select Roll_no,StudName from students_info LIMIT 2;
```

```
(4 rows)
cqlsh:students> select * from students_info where roll_no in (1,2,3);

 roll_no | dateofjoining                   | last_exam_percent | studname
---------+---------------------------------+-------------------+----------
       1 | 2023-10-08 18:30:00.000000+0000 |                98 |  Sadhana
       2 | 2023-10-09 18:30:00.000000+0000 |                97 |     Rutu
       3 | 2023-10-09 18:30:00.000000+0000 |              97.5 |  Rachana

(3 rows)
cqlsh:students> select * from students_info where Studname='Charu';
InvalidRequest: Error from server: code=2200 [Invalid query] message="Cannot execute this query as it might involve data filtering and thus may have unpredictable performance. If you want to execute this query d
espite the performance unpredictability, use ALLOW FILTERING"
cqlsh:students> create index on Students_info(StudName);
cqlsh:students> select * from students_info where Studname='Charu';

 roll_no | dateofjoining                   | last_exam_percent | studname
---------+---------------------------------+-------------------+----------
       4 | 2023-10-05 18:30:00.000000+0000 |              96.5 |    Charu

(1 rows)
cqlsh:students> select Roll_no,StudName from students_info LIMIT 2;

 roll_no | studname
---------+----------
       1 |  Sadhana
       2 |     Rutu

(2 rows)
cqlsh:students> SELECT Roll_no as "USN" from Students_info;

 USN
-----
   1
   2
   4
   3
```

Experiment - 4

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

Screenshot:



15/9/24 Hadoop

1) create directory in HDFS:
hdfs dfs -mkdir /user

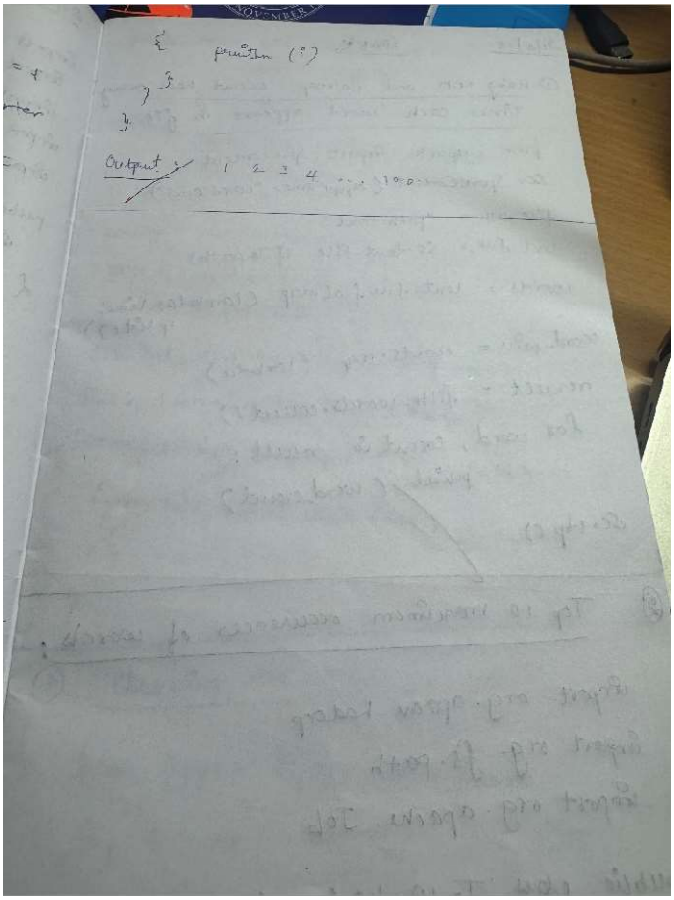
2) copy data from local file to HDFS
hdfs dfs -put /path/to/ local / file /user

3) list file or directory:
hdfs dfs -ls / abc

4) Display contents of file:
hdfs dfs - cat / abc / wc.text

5) copying data from HDFS - local.
hdfs dfs -get / abc / wc.text / home / downloads / wc.text

6) Hdfs command retrieves all files that match to source path entered by user in HDFS:
hdfs dfs - getmerge /abc/ wc.text /abc/ home/ hduser/ Desktop /merge.text



Code & Output:

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ cd ./Desktop/
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ 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:~/Desktop$ hdfs dfs -mkdir /Lab05
```

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hadoop fs -ls /Hadoop
ls: `/Hadoop': No such file or directory
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hadoop fs -ls /Lab05
```

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ touch test.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ nano text.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -put ./text.txt /Lab05/text.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hadoop fs -ls /Lab05
Found 1 items
-rw-r--r--   1 hadoop supergroup         19 2024-05-13 14:33 /Lab05/text.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -cat /Lab05/text.txt
Hello
How are you?
```

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hadoop fs -ls /Lab05
Found 2 items
-rw-r--r--   1 hadoop supergroup         15 2024-05-13 14:40 /Lab05/test.txt
-rw-r--r--   1 hadoop supergroup         19 2024-05-13 14:33 /Lab05/text.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -getmerge /Lab05 /text.txt /Lab05 /test.txt ../
Downloads/Merged.txt
getmerge: `/text.txt': No such file or directory
getmerge: `/test.txt': No such file or directory
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -getmerge /Lab05/text.txt /Lab05/test.txt ../Do
wnloads/Merged.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hadoop fs -getfacl /Lab05
# file: /Lab05
# owner: hadoop
# group: supergroup
user::rwx
group::r-x
other::r-x
```

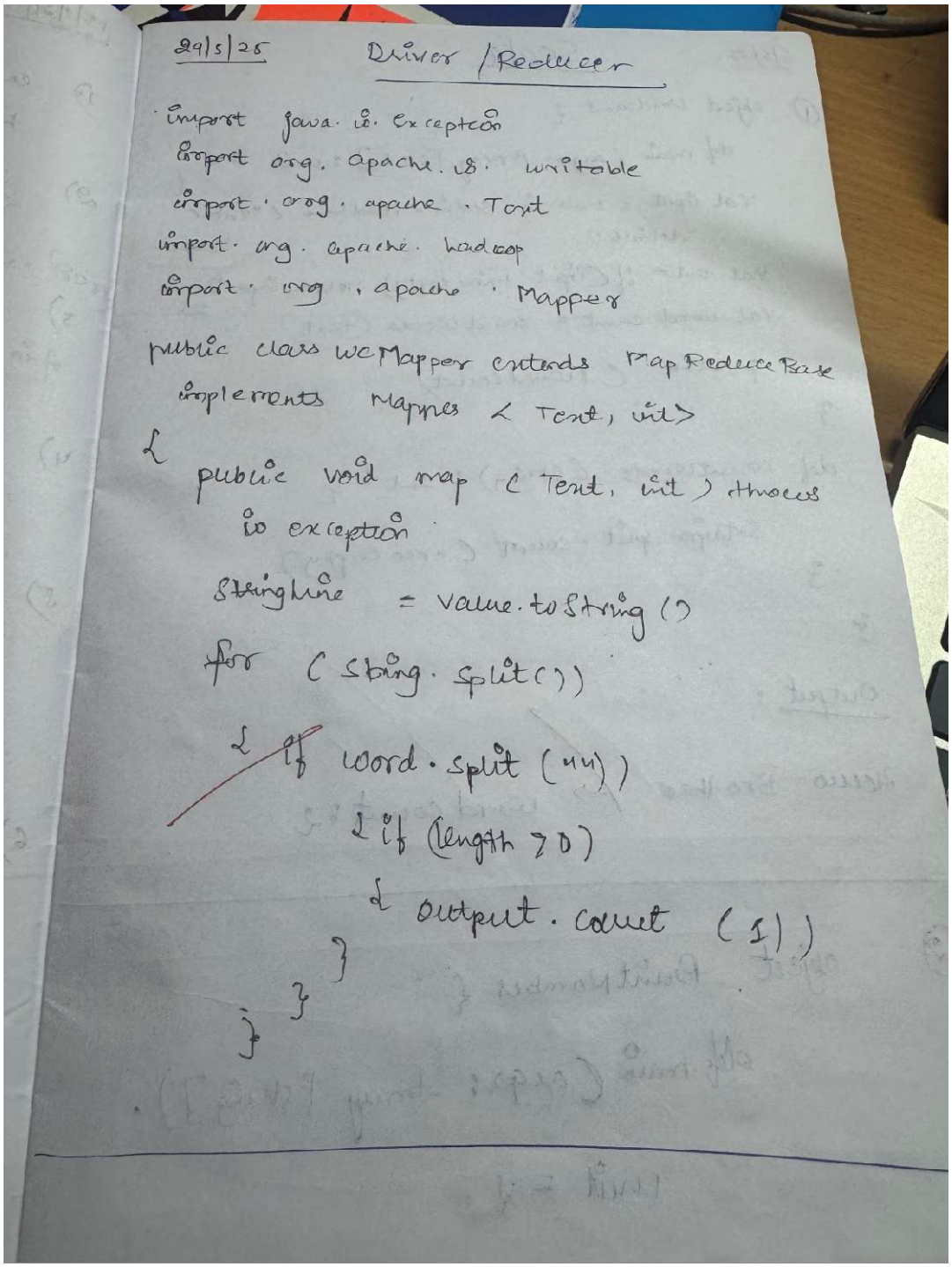
```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -copyToLocal /Lab05/text.txt ../Documents
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -copyToLocal /Lab05/test.txt ../Documents
```

```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -cat /Lab05/text.txt
Hello
How are you?
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -mv /Lab05 /test_Lab05
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -ls /test_Lab05
Found 2 items
-rw-r--r--   1 hadoop supergroup         15 2024-05-13 14:40 /test_Lab05/test.txt
-rw-r--r--   1 hadoop supergroup         19 2024-05-13 14:33 /test_Lab05/text.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -cp /test_Lab05/ /Lab05
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -ls /Lab05
Found 2 items
-rw-r--r--   1 hadoop supergroup         15 2024-05-13 14:51 /Lab05/test.txt
-rw-r--r--   1 hadoop supergroup         19 2024-05-13 14:51 /Lab05/text.txt
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~/Desktop$ hdfs dfs -ls /test_Lab05
Found 2 items
-rw-r--r--   1 hadoop supergroup         15 2024-05-13 14:40 /test_Lab05/test.txt
-rw-r--r--   1 hadoop supergroup         19 2024-05-13 14:33 /test_Lab05/text.txt
```

Experiment - 5

Q) Implement Wordcount program on Hadoop framework

Screenshot:



Code & Output:

Mapper Code: WCMapper.java java

```
CopyEdit                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: WCReducer.java java

```
CopyEdit                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> { public void reduce(Text key, Iterator<IntWritable> value, OutputCollector<Text,
IntWritable> output, Reporter rep) throws IOException { int count = 0; while (value.hasNext()) {
IntWritable i = value.next(); count += i.get();
    }
    output.collect(key, new IntWritable(count)); }
}
```

Driver Code: WCDriver.java java

CopyEdit

```
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);

    }
}
Input File -> big data hadoop big data analytics map
                reduce big data
```

Output:

(big, 1)

(data, 1)

(hadoop, 1)

(big, 1)

(data, 1)

(analytics, 1)

(map, 1)

(reduce, 1)

(big, 1)

(data, 1)

Experiment – 6

Q) From the following link extract the weather data
https://github.com/tomwhite/hadoopbook/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.

Screenshot:

Code & Output:

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

```
 GNU nano 7.2                                                    map.py *
import sys
for line in sys.stdin:
        year=line[15:19]
        temp=line[87:92]
        if temp!="+9999":
                print(f"{year}\t{temp}")
```

```
 GNU nano 7.2                                                    red.py *
import sys

current_year = None
temp_sum = 0
count = 0

for line in sys.stdin:
    year, temp = line.strip().split('\t')
    temp = int(temp)

    if current_year == year:
        temp_sum += temp
        count += 1
    else:
        if current_year:
            print(f"{current_year}\t{temp_sum/count:.2f}")
        current_year = year
        temp_sum = temp
        count = 1

# print last year
if current_year:
    print(f"{current_year}\t{temp_sum/count:.2f}")
```

```
ganajana@Anonymous:~$ hdfs dfs -mkdir /weatherdata
ganajana@Anonymous:~$ git clone https://github.com/tomwhite/hadoop-book.git
fatal: destination path 'hadoop-book' already exists and is not an empty directory.
ganajana@Anonymous:~$ git clone https://github.com/tomwhite/hadoop-book.git
Cluning into 'hadoop-book'...
remote: Enumerating objects: 4969, done.
remote: Total 4969 (delta 0), reused 0 (delta 0), pack-reused 4969 (from 1)
Receiving objects: 100% (4969/4969), 2.85 MiB | 6.78 MiB/s, done.
Resolving deltas: 100% (1945/1945), done.
ganajana@Anonymous:~$ hdfs dfs -put hadoop-book/input/ncdc/all/*/weatherdata
put: `.': No such file or directory: `hdfs://localhost:9000/user/ganajana'
ganajana@Anonymous:~$ hdfs dfs -ls /
Found 1 items
drwxr-xr-x   - ganajana supergroup          0 2025-05-26 16:27 /weatherdata
ganajana@Anonymous:~$ hdfs dfs -put hadoop-book/input/ncdc/all/*/weatherdata
put: `.': No such file or directory: `hdfs://localhost:9000/user/ganajana'
ganajana@Anonymous:~$ hdfs dfs -put hadoop-book/input/ncdc/all/* /weatherdata
ganajana@Anonymous:~$ nano map.py
ganajana@Anonymous:~$ chmod +x map.py
ganajana@Anonymous:~$ nano red.py
ganajana@Anonymous:~$ chmod +x red.py
ganajana@Anonymous:~$ hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-3.4.1.jar -input /weatherdata -output /weatherdata/out -mapper map.py
-reducer red.py
```

```
Mean Temperature: 31.18
```

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

```
 GNU nano 7.2                                                    map.py *
import sys

for line in sys.stdin:
    year = line[15:19]
    month = line[19:21]  # extract month from positions 19-20
    temp = line[87:92]
    if temp != "+9999":  # valid temperature
        print(f"{year}-{month}\t{temp}")
```

```
 GNU nano 7.2                                                    red.py *
import sys

current_year_month = None
temp_sum = 0
count = 0

for line in sys.stdin:
    year_month, temp = line.strip().split('\t')
    temp = int(temp)

    if current_year_month == year_month:
        temp_sum += temp
        count += 1
    else:
        if current_year_month:
            print(f"{current_year_month}\t{temp_sum/count:.2f}")
        current_year_month = year_month
        temp_sum = temp
        count = 1

# print last year-month
if current_year_month:
    print(f"{current_year_month}\t{temp_sum/count:.2f}")
```

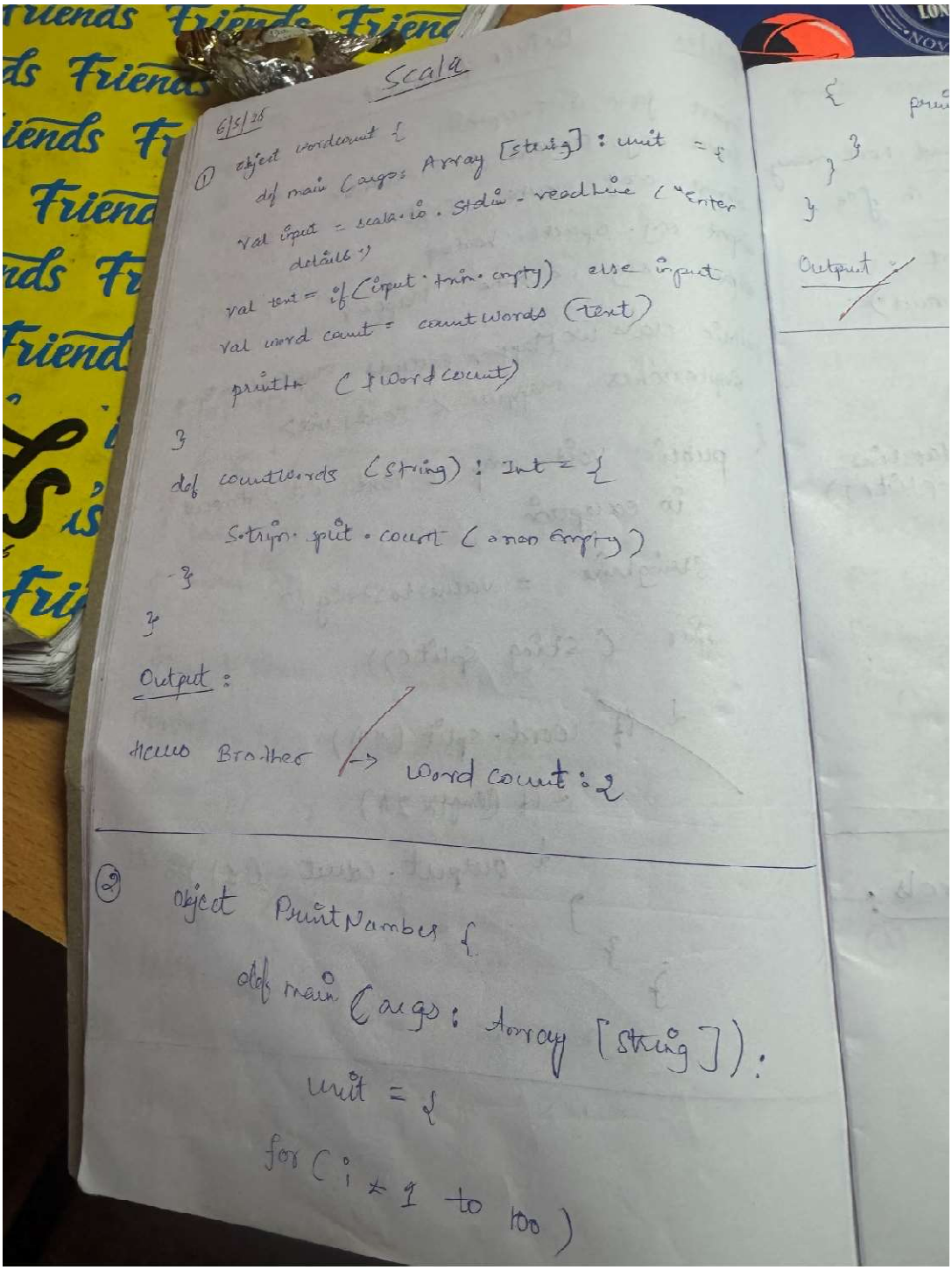
```
ganajana@Anonymous:~$ hdfs dfs -mkdir /weatherdata
ganajana@Anonymous:~$ git clone https://github.com/tomwhite/hadoop-book.git
fatal: destination path 'hadoop-book' already exists and is not an empty directory.
ganajana@Anonymous:~$ git clone https://github.com/tomwhite/hadoop-book.git
Cloning into 'hadoop-book'...
remote: Enumerating objects: 4969, done.
remote: Total 4969 (delta 0), reused 0 (delta 0), pack-reused 4969 (from 1)
Receiving objects: 100% (4969/4969), 2.85 MiB | 6.78 MiB/s, done.
Resolving deltas: 100% (1945/1945), done.
ganajana@Anonymous:~$ hdfs dfs -put hadoop-book/input/ncdc/all/*/weatherdata
put: `.': No such file or directory: `hdfs://localhost:9000/user/ganajana'
ganajana@Anonymous:~$ hdfs dfs -ls /
Found 1 items
drwxr-xr-x   - ganajana supergroup          0 2025-05-26 16:27 /weatherdata
ganajana@Anonymous:~$ hdfs dfs -put hadoop-book/input/ncdc/all/*/weatherdata
put: `.': No such file or directory: `hdfs://localhost:9000/user/ganajana'
ganajana@Anonymous:~$ hdfs dfs -put hadoop-book/input/ncdc/all/* /weatherdata
ganajana@Anonymous:~$ nano map.py
ganajana@Anonymous:~$ chmod +x map.py
ganajana@Anonymous:~$ nano red.py
ganajana@Anonymous:~$ chmod +x red.py
ganajana@Anonymous:~$ hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-3.4.1.jar -input /weatherdata -output /weatherdata/out -mapper map.py
-reducer red.py
```

```
Max Temperature: 33.50
```

Experiment – 7

Q) 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.

Screenshot:



Code & Output:

Top N Words Using MapReduce

TopN.java (Driver)

java

CopyEdit

```
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.java

java

CopyEdit

```
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.java

java

CopyEdit

```
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.java

```
java
CopyEdit
package samples.topn;

import    java.io.IOException;    import
java.util.HashMap; import java.util.Map;
import
org.apache.hadoop.io.IntWritable;
import          org.apache.hadoop.io.Text;
import
```

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

public class TopNReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
   private Map<Text, IntWritable> countMap = new HashMap<>();

   public void reduce(Text key, Iterable<IntWritable> values,
                      Reducer<Text, IntWritable, Text, IntWritable>.Context context) throws
         IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) sum
         += val.get();
      this.countMap.put(new Text(key), new IntWritable(sum));
   }

   protected void cleanup(Reducer<Text, IntWritable, Text, IntWritable>.Context context) throws
         IOException, InterruptedException {
      Map<Text, IntWritable> sortedMap =
      MiscUtils.sortByValues(this.countMap); int counter = 0; for (Text key :
      sortedMap.keySet()) { if (counter++ == 20) break;
         context.write(key,  sortedMap.get(key));
      }
   }
}
```

```
C:\hadoop-3.3.0\sbin>hadoop jar C:\sort.jar samples.topn.TopN /input_dir/input.txt /output_dir
2021-05-08 19:54:54,582 INFO client.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
```

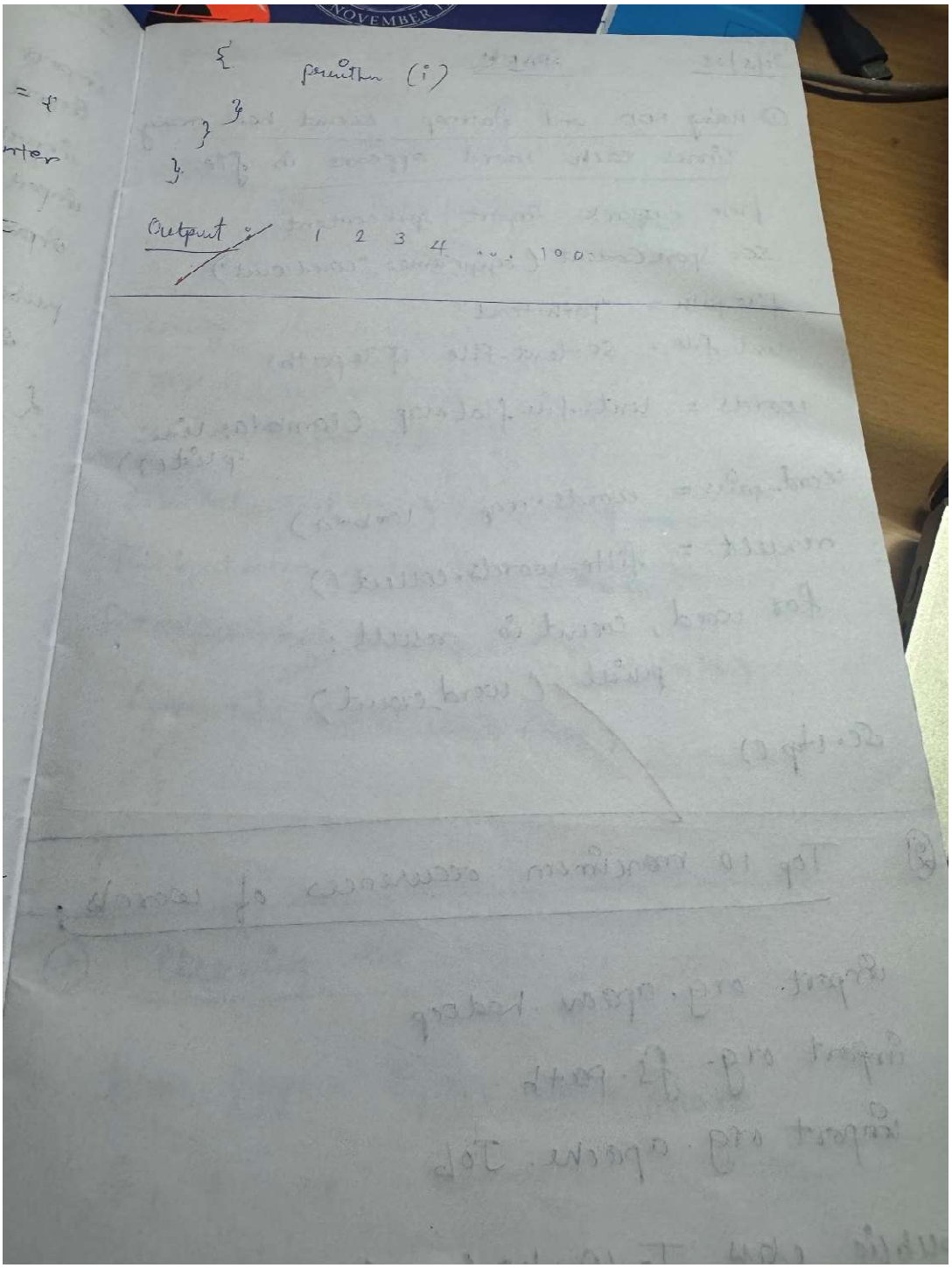
```
C:\hadoop-3.3.0\sbin>hdfs dfs -cat /output_dir/*
hello   2
hadoop  1
world   1
bye     1

C:\hadoop-3.3.0\sbin>
```

Experiment – 8

Q) Write a Scala program to print numbers from 1 to 100 using for loop.

Screenshot:

Acuus Brothes → Board count : 2
(2) object PrintNumbers {
obj main (args : Array [String]):
Unit = {
for (i ← 1 to 100)

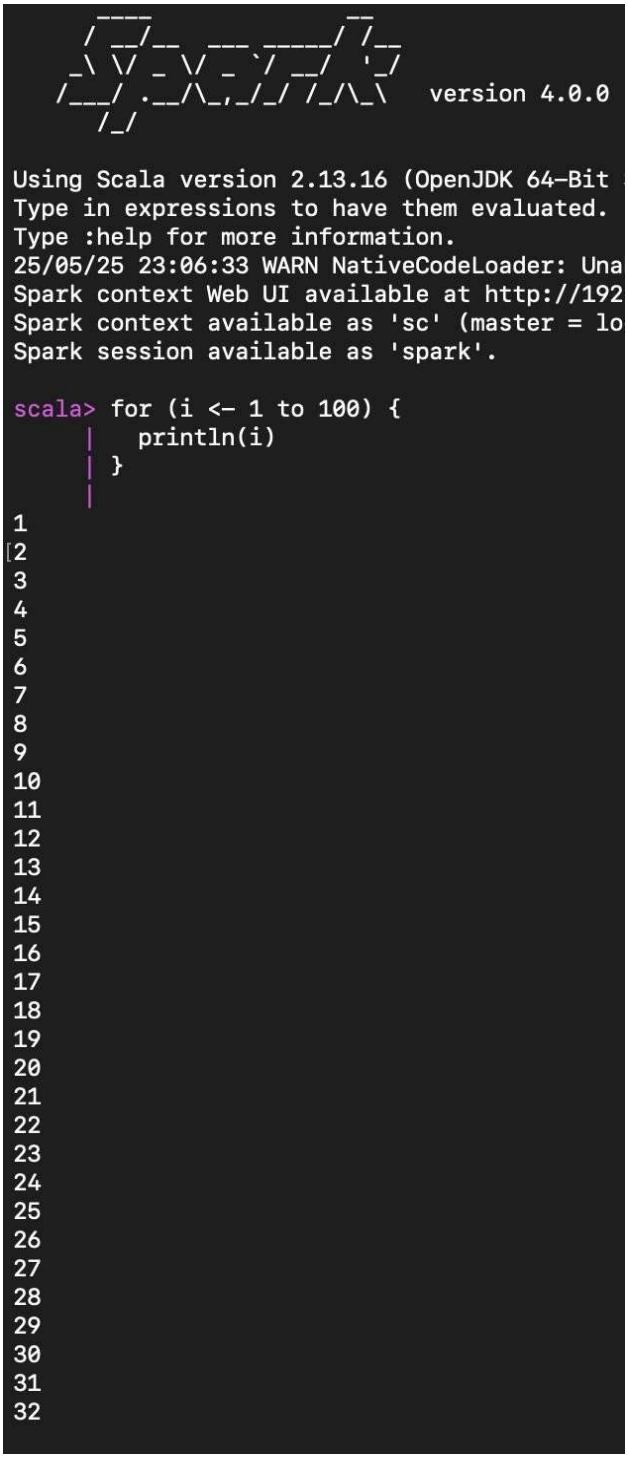


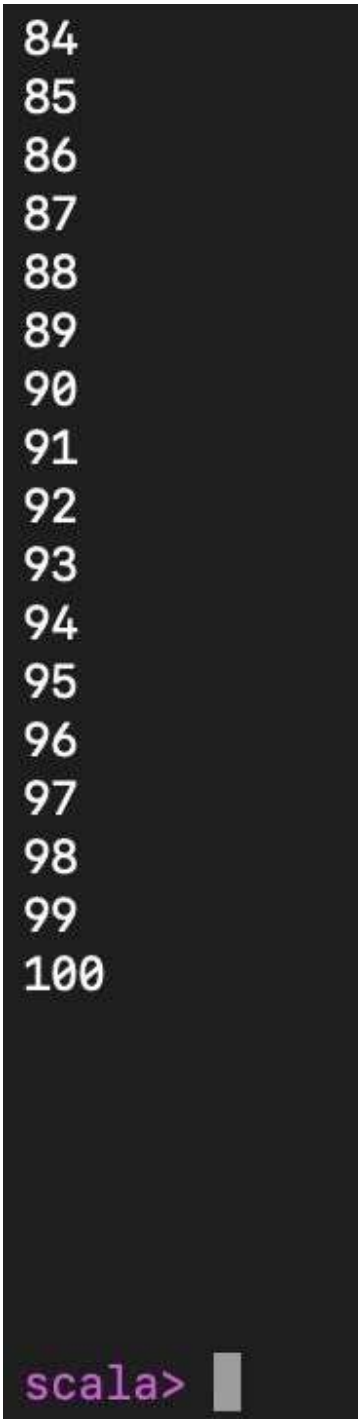
Code:

```
for (i <- 1 to 100) {
```

```
  println(i)
}
```

Output:(NEXT PAGE)

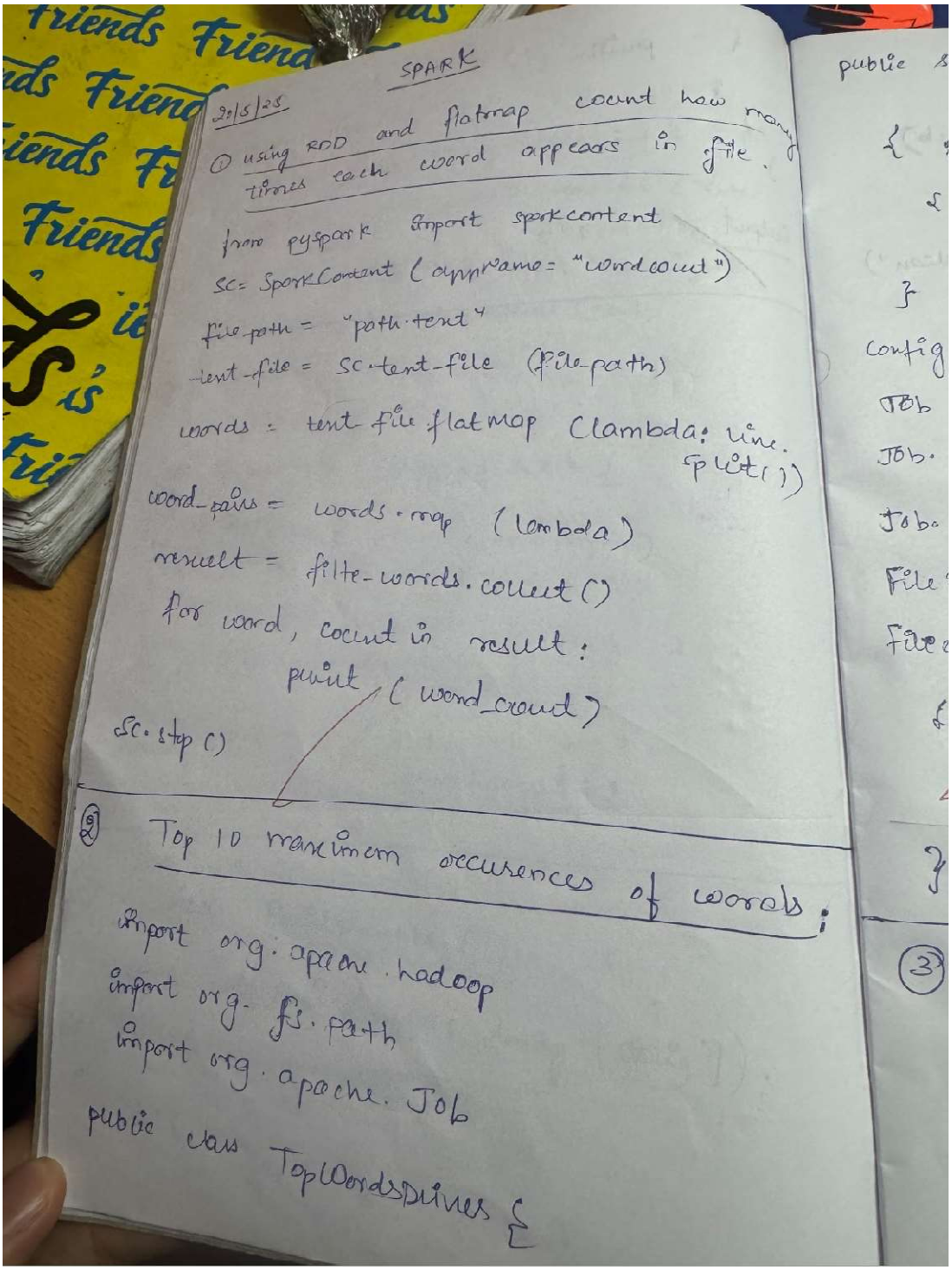

____ __
/ __/__ ___ _____/ /__
_\ \/ _ \/ _ `/ __/ '_/
/___/ .__/_,_/_/ /_/_\ version 4.0.0
/_/
Using Scala version 2.13.16 (OpenJDK 64-Bit
Type in expressions to have them evaluated.
Type :help for more information.
25/05/25 23:06:33 WARN NativeCodeLoader: Una
Spark context Web UI available at http://192
Spark context available as 'sc' (master = lo
Spark session available as 'spark'.
scala> for (i <- 1 to 100) {
| println(i)
| }
|
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32


84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
scala>

Experiment – 9

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

Screenshot:



Code:

```
val text = sc.textFile("file:///Users//Desktop/word.txt")
```

```
val words = text.flatMap(_.split("\\W+"))

val cleanedWords = words.map(_.toLowerCase).filter(_.nonEmpty)

val wordPairs = cleanedWords.map((_, 1))

val wordCounts = wordPairs.reduceByKey(_ + _)

val frequentWords = wordCounts.filter(_._2 > 4)

val wordsOnly = frequentWords.map(_._1)

wordsOnly.collect().foreach(println)
```

Input Word.txt file :

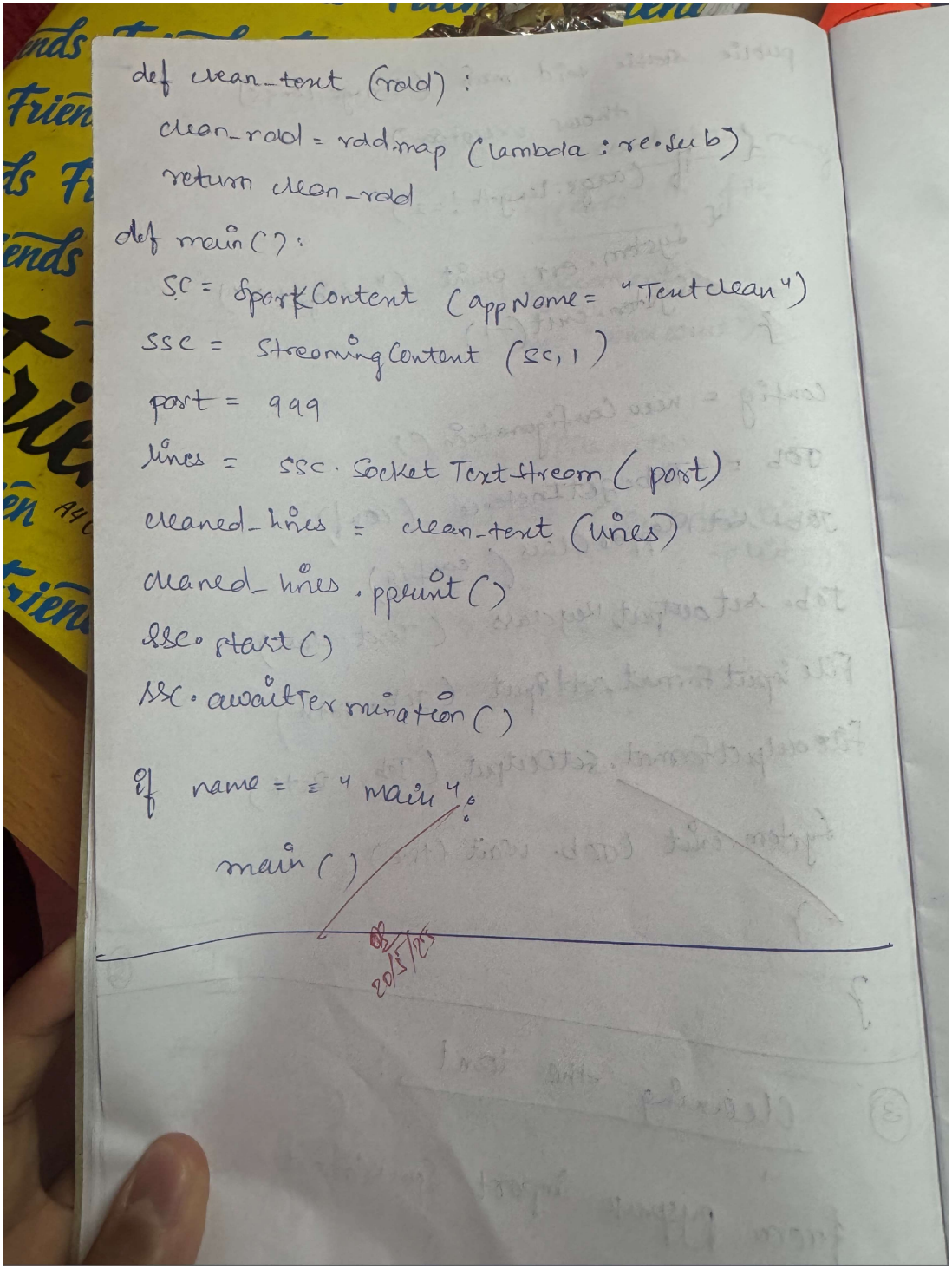
```
Apple, Apple, apple, APPLE, apple. This is an apple.
Banana orange grape spark.
Data data data data data.
Hello world. Hello Spark. Hello Scala. Hello again. Hello.
Another word, another line.
```

Output:

Experiment 10:

Q)Write a simple streaming program in Spark to receive text data streams on a particular port, perform basic text cleaning (like white space removal, stop words removal, lemmatization, etc.), and print the cleaned text on the screen. (Open Ended Question)

Screenshot:



Output:

```
Requirement already satisfied: nltk in /usr/local/lib/python3.11/dist-packages (3.9.1)
Requirement already satisfied: click in /usr/local/lib/python3.11/dist-packages (from nltk) (8.2.0)
Requirement already satisfied: joblib in /usr/local/lib/python3.11/dist-packages (from nltk) (1.5.0)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.11/dist-packages (from nltk) (2024.11.6)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from nltk) (4.67.1)
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
+------+
|word  |
+------+
|hello |
|hate  |
|hate  |
|love  |
|dont  |
|want  |
|cant  |
|put   |
|nobody|
|else  |
+------+
```