

B.M.S. COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU



Lab Report

Big-Data Analytics

(20CS6PEBDA)

Submitted in partial fulfillment for the 6th Semester Laboratory

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by:

STHAVIR G SOROFF

1BM19CS161

Department of Computer Science and Engineering
B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
April-August 2022

B.M.S. COLLEGE OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the Lab work entitled “**BIG DATA ANALYTICS**” carried out by **STHAVIR G SOROFF (1BM19CS161)**, who is bonafide student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **Big Data Analytics - (20CS6PEBDA)** work prescribed for the said degree.

Antara Roy Choudhury
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

Table of Contents

1. Perform the following DB operations using Cassandra Employee.
2. Perform the following DB operations using Cassandra Library.
3. MongoDB- CRUD Demonstration.

Program - 1

1. Create a keyspace by name Employee

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

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

```
cqlsh:employee> create table employeeinfo(emp_id int primary key, emp_name text, designation text, doj timestamp, salary double, dept_name text);
```

3. Insert the values into the table in batch

```
cqlsh:employee> begin batch  
... insert into employeeinfo(emp_id, emp_name, designation, doj, salary, dept_name) values  
(1, 'Ajay', 'Data analyst', '2018-04-16', 20000, 'Corporate');  
... insert into employeeinfo(emp_id, emp_name, designation, doj, salary, dept_name) values  
(121, 'Chaitra', 'web design', '2019-08-06', 15000, 'web_designer');  
... apply batch;  
cqlsh:employee> select * from employeeinfo;
```

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

```
cqlsh:employee> update employeeinfo set emp_name = 'Joy', dept_name = 'Management' where  
emp_id = 121;  
cqlsh:employee> select * from employeeinfo;
```

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

```
cqlsh:employee> alter table employeeinfo add projects set<text>;
```

6. Update the altered table to add project names.

```
cqlsh:employee> update employeeinfo set projects = {'project1', 'project2'} where emp_id in(1,121);  
cqlsh:employee> select * from employeeinfo;
```

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

cqlsh:employee> begin batch

... insert into employeeinfo(emp_id, emp_name, designation, doj, salary, dept_name) values (121, 'Boris', 'MTO', '2001-08-05', 12212, 'Corporate') using ttl 15;

... apply batch;

cqlsh:employee> select ttl(designation) from employeeinfo where emp_id = 121;

Output :

```
Terminal +
CQL Interactive Bash Terminal.
$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 4.0-beta2 | CQL spec 3.4.5 | Native protocol v4]
Use HELP for help.
cqlsh>
cqlsh> create keyspace employee with replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
cqlsh> use employee;
cqlsh:employee> create table employeeinfo(emp_id int primary key, emp_name text, designation text, doj timestamp, salary double, dept_name text);
cqlsh:employee> begin batch
... insert into employeeinfo(emp_id, emp_name, designation, doj, salary, dept_name) values (1, 'Ajay', 'Data analyst', '2018-04-16', 20000, 'Corporate');
... insert into employeeinfo(emp_id, emp_name, designation, doj, salary, dept_name) values (121, 'Chaitra', 'web design', '2019-08-06', 15000, 'web designer');
... apply batch;
cqlsh:employee> select * from employeeinfo;

emp_id | dept_name | designation | doj | emp_name | salary
-----|-----|-----|-----|-----|-----
1 | Corporate | Data analyst | 2018-04-16 00:00:00.000000+0000 | Ajay | 20000
121 | web_designer | web design | 2019-08-06 00:00:00.000000+0000 | Chaitra | 15000
(2 rows)
cqlsh:employee> update employeeinfo set emp_name = 'Joy', dept_name = 'Management' where emp_id = 121;
cqlsh:employee> select * from employeeinfo;

emp_id | dept_name | designation | doj | emp_name | salary
-----|-----|-----|-----|-----|-----
1 | Corporate | Data analyst | 2018-04-16 00:00:00.000000+0000 | Ajay | 20000
121 | Management | web design | 2019-08-06 00:00:00.000000+0000 | Joy | 15000
(2 rows)
cqlsh:employee> alter table employeeinfo add projects set<text>;
cqlsh:employee> update employeeinfo set projects = {'project1', 'project2'} where emp_id in(1,121);
cqlsh:employee> select * from employeeinfo;

emp_id | dept_name | designation | doj | emp_name | projects | salary
-----|-----|-----|-----|-----|-----|-----
1 | Corporate | Data analyst | 2018-04-16 00:00:00.000000+0000 | Ajay | {'project1', 'project2'} | 20000
121 | Management | web design | 2019-08-06 00:00:00.000000+0000 | Joy | {'project1', 'project2'} | 15000
(2 rows)
cqlsh:employee> alter table employeeinfo add projects set<text>;
cqlsh:employee> update employeeinfo set projects = {'project1', 'project2'} where emp_id in(1,121);
cqlsh:employee> select * from employeeinfo;

emp_id | dept_name | designation | doj | emp_name | projects | salary
-----|-----|-----|-----|-----|-----|-----
1 | Corporate | Data analyst | 2018-04-16 00:00:00.000000+0000 | Ajay | {'project1', 'project2'} | 20000
121 | Management | web design | 2019-08-06 00:00:00.000000+0000 | Joy | {'project1', 'project2'} | 15000
(2 rows)
cqlsh:employee> begin batch
... insert into employeeinfo(emp_id, emp_name, designation, doj, salary, dept_name) values (121, 'Boris', 'MTO', '2001-08-05', 12212, 'Corporate') using ttl 15;
... apply batch;
cqlsh:employee> select ttl(designation) from employeeinfo where emp_id = 121;

ttl(designation)
-----
null
(1 rows)
cqlsh:employee> select * from employeeinfo;

emp_id | dept_name | designation | doj | emp_name | projects | salary
-----|-----|-----|-----|-----|-----|-----
1 | Corporate | Data analyst | 2018-04-16 00:00:00.000000+0000 | Ajay | {'project1', 'project2'} | 20000
121 | null | null | null | null | {'project1', 'project2'} | null
(2 rows)
cqlsh:employee> begin batch insert into employeeinfo(emp_id, emp_name, designation, doj, salary, dept_name) values (121, 'Boris', 'MTO', '2001-08-05', 12212, 'Corporate') using ttl 120; apply batch;
cqlsh:employee> select ttl(designation) from employeeinfo where emp_id = 121;

ttl(designation)
-----
109
(1 rows)
cqlsh:employee>
```

Program – 2

Perform the following DB operations using Cassandra.

1. Create a keyspace by name Library

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

2. Create a column family by name Library-Info with attributes

Stud_Id Primary Key,

Counter_value of type Counter,

Stud_Name, Book-Name, Book-Id, Date_of_issue

```
cqlsh:library> create table library_info( id int, counter_val counter, stud_name text, book_name text,  
book_id int, issue_date timestamp, primary key(id,stud_name,book_name,book_id,issue_date));
```

3. Insert the values into the table in batch

```
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 1 and stud_name =  
'Anand' and book_name = 'CNS' and book_id = 121 and issue_date='2020-12-31';  
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 3 and stud_name =  
'Arjun' and book_name = 'ML' and book_id = 112 and issue_date='2021-02-01';
```

```
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 5 and stud_name =  
'Chaitra' and book_name = 'Python' and book_id = 114 and issue_date='2009-08-27';  
cqlsh:library> select * from library_info;
```

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

```
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 3 and stud_name =  
'Arjun' and book_name = 'ML' and book_id = 112 and issue_date='2021-02-01';
```

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

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

6. Export the created column to a csv file

```
cqlsh:library> copy library_info(id,counter_val,stud_name,book_name,book_id,issue_date) to  
'Desktop/library_data.csv';
```

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

cqlsh:library> copy library_info(id,counter_val,stud_name,book_name,book_id,issue_date) from 'Desktop/library_data.csv';

Output :

```
Terminal +
Your Interactive Bash Terminal.
$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 4.0-beta2 | CQL spec 3.4.5 | Native protocol v4]
Use HELP for help.
cqlsh>
cqlsh> create keyspace library with replication = ( 'class' : 'SimpleStrategy','replication_factor':1);
cqlsh> use library;
cqlsh:library> create table library_info( id int, counter_val counter, stud_name text, book_name text, book_id int, issue_date timestamp,primary key(
id,stud_name,book_name,book_id,issue_date));
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 1 and stud_name = 'Anand' and book_name = 'CNS' and book_id = 121 and
issue_date='2020-12-31';
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 3 and stud_name = 'Arjun' and book_name = 'ML' and book_id = 112 and i
ssue_date='2021-02-01';
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 5 and stud_name = 'Chaitra' and book_name = 'Python' and book_id = 114
and issue_date='2009-08-27';
cqlsh:library> select * from library_info;

 id | stud_name | book_name | book_id | issue_date | counter_val
-----+-----+-----+-----+-----+-----
  5 | Chaitra | Python | 114 | 2009-08-27 00:00:00.000000+0000 | 1
  1 | Anand | CNS | 121 | 2020-12-31 00:00:00.000000+0000 | 1
  3 | Arjun | ML | 112 | 2021-02-01 00:00:00.000000+0000 | 1
(3 rows)
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 3 and stud_name = 'Arjun' and book_name = 'EDA' and book_id = 112 and
issue_date='2011-12-20';
cqlsh:library> select * from library_info where counter_val = 2 allow filtering;

 id | stud_name | book_name | book_id | issue_date | counter_val
-----+-----+-----+-----+-----+-----
(0 rows)
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 3 and stud_name = 'Arjun' and book_name = 'ML' and book_id = 112 and i
ssue_date='2011-12-20';
cqlsh:library> select * from library_info where counter_val = 2 allow filtering;

 id | stud_name | book_name | book_id | issue_date | counter_val
-----+-----+-----+-----+-----+-----
(0 rows)
cqlsh:library> update library_info SET counter_val = counter_val +1 where id = 3 and stud_name = 'Arjun' and book_name = 'ML' and book_id = 112 and i
ssue_date='2021-02-01';
cqlsh:library> select * from library_info where counter_val = 2 allow filtering;

 id | stud_name | book_name | book_id | issue_date | counter_val
-----+-----+-----+-----+-----+-----
  3 | Arjun | ML | 112 | 2021-02-01 00:00:00.000000+0000 | 2
(1 rows)
cqlsh:library> copy library_info(id,counter_val,stud_name,book_name,book_id,issue_date) to 'Desktop/library_data.csv';
Using 1 child processes

Starting copy of library.library_info with columns [id, counter_val, stud_name, book_name, book_id, issue_date].
cqlshlib.copyputil.ExportProcess.write_rows_to_csv(): writing row
cqlshlib.copyputil.ExportProcess.write_rows_to_csv(): writing row
cqlshlib.copyputil.ExportProcess.write_rows_to_csv(): writing row
cqlshlib.copyputil.ExportProcess.write_rows_to_csv(): writing row
cqlshlib.copyputil.ExportProcess.write_rows_to_csv(): writing row
Processed: 5 rows; Rate: 8 rows/s; Avg. rate: 9 rows/s
5 rows exported to 1 files in 0.555 seconds.
cqlsh:library> copy library_info(id,counter_val,stud_name,book_name,book_id,issue_date) from 'Desktop/library_data.csv';
Using 1 child processes

Starting copy of library.library_info with columns [id, counter_val, stud_name, book_name, book_id, issue_date].
Processed: 5 rows; Rate: 9 rows/s; Avg. rate: 14 rows/s
5 rows imported from 1 files in 0.365 seconds (0 skipped).
cqlsh:library>
```

Program – 3

Perform the following DB operations using MongoDB.

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

2. Insert appropriate values

```
db.student.insert({Roll: 10, Name: "suma", age: 21, contact: "7723112389", email: "suma@gmail.com"})
db.student.insert({Roll: 11, Name: "ABC", age: 20, contact: "9263532389", email: "abc@gmail.com"})
db.student.insert({Roll: 12, Name: "shek", age: 21, contact: "7788996655", email: "shek@gmail.com"})
db.student.insert({Roll: 13, Name: "raj", age: 20, contact: "1234123412", email: "raj@gmail.com"})
```

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

```
db.student.update({Roll:10}, {$set: {email: "suma123@gmail.com"}})
```

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

```
db.student.update({Roll:11}, {$set: {Name: "FEM"}})
```

5. Export the created table into local file system

```
mongoexport --db student --collection student --type csv --out D:\export.csv --fields "Roll,Name,age,contact,email"
```

6. Drop the table

```
db.student.drop()
```

7. Import a given csv dataset from the local file system into mongodb collection.

```
mongoimport --db student --collection student --type csv --file D:\export.csv --headerline
```


Output :

```
use student
db.student.insert({Roll: 10, Name: "suma", age: 21, contact: "7723112389", email: "suma@gmail.com"})
db.student.insert({Roll: 11, Name: "ABC", age: 20, contact: "9263532389", email: "abc@gmail.com"})
db.student.insert({Roll: 12, Name: "shek", age: 21, contact: "7788996655", email: "shek@gmail.com"})
db.student.insert({Roll: 13, Name: "raj", age: 20, contact: "1234123412", email: "raj@gmail.com"})

db.student.update({Roll:10}, {$set: {email: "sumal23@gmail.com"}})
db.student.update({Roll:11}, {$set: {Name: "FEM"}})

show collections
db.student.find()

mongoexport --db testdb --collection student --out C:\Users\SUMALATA\OneDrive\Desktop\output.json

Drop student

mongoimport --db testdb --collection Student C:\Users\SUMALATA\OneDrive\Desktop\output.json
```

student 0.059 sec.

Key	Value	Type
(1) ObjectId("6067b01b80764c83bffc7871")	{ 6 fields }	Object
_id	ObjectId("6067b01b80764c83bffc7871")	ObjectId
Roll	10.0	Double
Name	suma	String
age	21.0	Double
contact	7723112389	String
email	sumal23@gmail.com	String
(2) ObjectId("6067b0b980764c83bffc7872")	{ 6 fields }	Object
_id	ObjectId("6067b0b980764c83bffc7872")	ObjectId
Roll	11.0	Double
Name	FEM	String

