**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“JnanaSangama”, Belgaum -590014, Karnataka.**

****

**LAB REPORT**

**on**

**BIG DATA ANALYTICS**

**(20CS6PEBDA)**

***Submitted by***

**SHIVANI GAHLOT (1BM19CS150)**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**



**B.M.S. COLLEGE OF ENGINEERING**

**(Autonomous Institution under VTU)**

**BENGALURU-560019**

**May-2022 to July-2022**

**B. M. S. College of Engineering,**

**Bull Temple Road, Bangalore 560019**

(Affiliated To Visvesvaraya Technological University, Belgaum)

**Department of Computer Science and Engineering**



**CERTIFICATE**

This is to certify that the Lab work entitled “**BIG DATA ANALYTICS**” carried out by **SHIVANI GAHLOT (1BM19CS150),** who is a 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 Big Data Analytics **- (20CS6PEBDA)** work prescribed for the said degree.

Antara Roy Choudhury               **Dr. Jyothi S Nayak**

Assistant Professor Professor and Head

Department of CSE Department of CSE

BMSCE, Bengaluru BMSCE, Bengaluru

`

**Index Sheet**

| **Sl. No.** | **Experiment Title** | **Page No.** |
| --- | --- | --- |
|  | **MongoDB CRUD Demonstration** | **4-6** |
|  | **Perform the following DB operations using Cassandra- Employee** | **7-8** |
|  | **Perform the following DB operations using Cassandra- Library** | **9-10** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Course Outcome**

| CO1 | Apply the concept of NoSQL, Hadoop or Spark for a given task |
| --- | --- |
| CO2 | Analyze the Big Data and obtain insight using data analytics mechanisms. |
| CO3 | Design and implement Big data applications by applying NoSQL, Hadoop or Spark |

1. **MongoDB:**

**I. CREATE DATABASE IN MONGODB.**

use myDB;

**Confirm the existence of your database**

db;

**To list all databases**

show dbs;

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

**1. To create a collection by the name “Student”.**

db.createCollection(“Student”);

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

db.Student.drop();

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

db.Student.insert({\_id:1,StudName:”MichelleJacintha”,Grade:”VII”,Hobbies:”InternetS

urfing”});

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

db.Student.update({\_id:3,StudName:”AryanDavid”,Grade:”VII”},{$set:{Hobbies:”Skatin

g”}},{upsert:true});

**5. FIND METHOD**

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

**criteria.**

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

**B. To display only the StudName and Grade from all the documents of the Students**

**collection. The identifier\_id should be suppressed and NOT displayed.**

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

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

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

**D. To find those documents from the Students collection where the Hobbies is set to**

**either ‘Chess’ or is set to ‘Skating’.**

db.Student.find({Hobbies :{ $in: [’Chess’,’Skating’]}}).pretty ();

**E. To find documents from the Students collection where the StudName begins with “M”.**

db.Student.find({StudName:/^M/}).pretty();

**F. To find documents from the Students collection where the StudNamehas an “e” in any position.**

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

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

db.Student.count();

**H. To sort the documents from the Students collection in the descending order of**

**StudName.**

db.Student.find().sort({StudName:-1}).pretty();

**III. Import data from a CSV file**

**Given a CSV file “sample.txt” in the D:drive, import the file into the MongoDB**

**collection, “SampleJSON”. The collection is in the database “test”.**

mongoimport --db Student --collection airlines --type csv –headerline --file

/home/hduser/Desktop/airline.csv

**IV. Export data to a CSV file**

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

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

mongoexport --host localhost --db Student --collection airlines --csv --out

/home/hduser/Desktop/output.txt –fields “Year”,”Quarter”

**V. Save Method :**

*Save() method will insert a new document, if the document with the \_id does not*

*exist. If it exists it will replace the exisiting document.*

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

**VI. Add a new field to existing Document:**

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

**VII. Remove the field in an existing Document**

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

**VIII. Finding Document based on search criteria suppressing few fields**

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

**To find those documents where the Grade is not set to ‘VII’**

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

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

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

**IX. to set a particular field value to NULL**

db.Students.update({\_id:3},{$set:{Location:null}})

**X. Count the number of documents in Student Collections**

db.Students.count()

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

db.Students.count({Grade:”VII”})

**Retrieve first 3 documents**

db.Students.find({Grade:”VII”}).limit(3).pretty();

**Sort the document in Ascending order**

db.Students.find().sort({StudName:1}).pretty();

**for descending order:**

db.Students.find().sort({StudName:-1}).pretty();

**to Skip the 1st two documents from the Students Collections**

db.Students.find().skip(2).pretty()

**XII. Create a collection by name “food” and add to each document add a “fruits” array**

db.food.insert( { \_id:1, fruits:[‘grapes’,’mango’,’apple’] } )

db.food.insert( { \_id:2, fruits:[‘grapes’,’mango’,’cherry’] } )

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

**To find those documents from the “food” collection which has the “fruits array”**

**constitute of “grapes”, “mango” and “apple”.**

db.food.find ( {fruits: [‘grapes’,’mango’,’apple’] } ). pretty().

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

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

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

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

**To find the document with a particular id and display the first two elements from the**

**array “fruits”**

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

**To find all the documets from the food collection which have elements mango and**

**grapes in the array “fruits”**

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

**Update on Array:**

**Using particular id replace the element present in the 1 st index position of the fruits**

**array with apple**

db.food.update({\_id:3},{$set:{‘fruits.1’:’apple’}})

**Insert new key value pairs in the fruits array**

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

**XII. Aggregate Function :**

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

**Now group on “custID” and compute the sum of “AccBal”.**

db.Customers.aggregate ( {$group : { \_id : “$custID”,TotAccBal : {$sum:”$AccBal”} } } );

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

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

{$sum:”$AccBal”} } } );

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

**total balance greater than 1200.**

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

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

**2.     Perform the following DB operations using Cassandra.**

**1.Create a keyspace by name Employee**

CREATE KEYSPACE employee123 WITH REPLICATION = {'class':'SimpleStrategy','replication\_factor':1};

**2. Create a column family by name**

**Employee-Info with attributes**

**Emp\_Id Primary Key, Emp\_Name,**

**Designation, Date\_of\_Joining, Salary,**

**Dept\_Name**

CREATE TABLE EMPLOYEEINFO( EMPID INT PRIMARY KEY, EMPNAME TEXT, DESIGNATION TEXT, DATEOFJOINING TIMESTAMP, SALARY DOUBLE, DEPTNAME TEXT);

**3. Insert the values into the table in batch**

Begin Batch

INSERT INTO EMPLOYEEINFO (EMPID, EMPNAME, DESIGNATION, DATEOFJOINING, SALARY, DEPTNAME) VALUES(1,'ABHISHEK','ASSISTANT MANAGER', '2010-04-26', 75000, 'MARKETING')

INSERT INTO EMPLOYEEINFO (EMPID, EMPNAME, DESIGNATION, DATEOFJOINING, SALARY, DEPTNAME) VALUES(2,'BHASKAR','ASSISTANT MANAGER', '2010-04-26', 75000, 'MARKETING')

INSERT INTO EMPLOYEEINFO (EMPID, EMPNAME, DESIGNATION, DATEOFJOINING, SALARY, DEPTNAME) VALUES(3,'CHIRAG','ASSISTANT MANAGER', '2010-04-26', 75000, 'MARKETING')

INSERT INTO EMPLOYEEINFO (EMPID, EMPNAME, DESIGNATION, DATEOFJOINING, SALARY, DEPTNAME)VALUES(4,'DHANUSH','ASSISTANT MANAGER', '2010-04-26', 75000, 'MARKETING')

INSERT INTO EMPLOYEEINFO (EMPID, EMPNAME, DESIGNATION, DATEOFJOINING, SALARY, DEPTNAME) VALUES(5,'ESHAAN','ASSISTANT MANAGER', '2010-04-26', 85000, 'TECHNICAL')

INSERT INTO EMPLOYEEINFO (EMPID, EMPNAME, DESIGNATION, DATEOFJOINING, SALARY, DEPTNAME) VALUES(6,'FARAH','MANAGER', '2010-04-26', 95000, 'TECHNICAL')

INSERT INTO EMPLOYEEINFO (EMPID, EMPNAME, DESIGNATION, DATEOFJOINING, SALARY, DEPTNAME) VALUES(7,'GEMMA','MANAGER', '2010-04-26', 95000, 'PR')

INSERT INTO EMPLOYEEINFO (EMPID, EMPNAME, DESIGNATION, DATEOFJOINING, SALARY, DEPTNAME)VALUES(121,'HARRY','REGIONAL MANAGER', '2010-04-26', 99000, 'MANAGEMENT')

APPLY BATCH;

**SELECT \* FROM EMPLOYEEINFO;**

empid | dateofjoining | deptname | designation | empname | salary

-------+---------------------------------+-----------+-------------------+----------+--------

5 | 2010-04-25 18:30:00.000000+0000 | TECHNICAL | ASSISTANT MANAGER | ESHAAN | 85000

1 | 2010-04-25 18:30:00.000000+0000 | MARKETING | ASSISTANT MANAGER | ABHISHEK | 75000

2 | 2010-04-25 18:30:00.000000+0000 | MARKETING | ASSISTANT MANAGER | BHASKAR | 75000

4 | 2010-04-25 18:30:00.000000+0000 | MARKETING | ASSISTANT MANAGER | DHANUSH | 75000

121 | 2010-04-25 18:30:00.000000+0000 | MANAGEMENT | REGIONAL MANAGER | HARRY | 99000

7 | 2010-04-25 18:30:00.000000+0000 | PR | MANAGER | GEMMA | 95000

6 | 2010-04-25 18:30:00.000000+0000 | TECHNICAL | MANAGER | FARAH | 95000

3 | 2010-04-25 18:30:00.000000+0000 | MARKETING | ASSISTANT MANAGER | CHIRAG | 75000

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

UPDATE EMPLOYEEINFO SET EMPNAME='HARISH', DEPTNAME='PR' WHERE EMPID=121;

**5. Sort the details of Employee records based on salary**

SELECT \* FROM EMPLOYEE\_IN WHERE EMP\_ID IN(1,2,3,4) ORDER BY SALARY DESC ALLOW FILTERING;

**6. Alter the schema of the table Employee\_Info to add a column Projects which**

**stores a set of Projects done by the corresponding Employee.**

ALTER TABLE EMPLOYEEINFO ADD PROJECTS LIST<TEXT>;

**7. Update the altered table to add project names.**

UPDATE EMPLOYEEINFO SET PROJECTS=['FACEBOOK','SNAPCHAT'] WHERE EMPID=1;

UPDATE EMPLOYEEINFO SET PROJECTS=['FACEBOOK','SNAPCHAT'] WHERE EMPID=7;

UPDATE EMPLOYEEINFO SET PROJECTS=['PINTEREST','INSTAGRAM'] WHERE EMPID=121;

UPDATE EMPLOYEEINFO SET PROJECTS=['PINTEREST','INSTAGRAM'] WHERE EMPID=4;

UPDATE EMPLOYEEINFO SET PROJECTS=['YOUTUBE','SPOTIFY'] WHERE EMPID=2;

UPDATE EMPLOYEEINFO SET PROJECTS=['YOUTUBE','SPOTIFY'] WHERE EMPID=3;

UPDATE EMPLOYEEINFO SET PROJECTS=['YOUTUBE','SPOTIFY'] WHERE EMPID=6;

UPDATE EMPLOYEEINFO SET PROJECTS=['TWITTER','REDDIT'] WHERE EMPID=5;

**SELECT \* FROM EMPLOYEEINFO;**

empid | dateofjoining | deptname | designation | empname | projects | salary

-------+---------------------------------+-----------+-------------------+----------+----------------------------+--------

5 | 2010-04-25 18:30:00.000000+0000 | TECHNICAL | ASSISTANT MANAGER | ESHAAN | ['TWITTER', 'REDDIT'] | 85000

1 | 2010-04-25 18:30:00.000000+0000 | MARKETING | ASSISTANT MANAGER | ABHISHEK | ['FACEBOOK', 'SNAPCHAT'] | 75000

2 | 2010-04-25 18:30:00.000000+0000 | MARKETING | ASSISTANT MANAGER | BHASKAR | ['YOUTUBE', 'SPOTIFY'] | 75000

4 | 2010-04-25 18:30:00.000000+0000 | MARKETING | ASSISTANT MANAGER | DHANUSH | ['PINTEREST', 'INSTAGRAM'] | 75000

121 | 2010-04-25 18:30:00.000000+0000 | PR | REGIONAL MANAGER | HARISH | ['PINTEREST', 'INSTAGRAM'] | 99000

7 | 2010-04-25 18:30:00.000000+0000 | PR | MANAGER | GEMMA | ['FACEBOOK', 'SNAPCHAT'] | 95000

6 | 2010-04-25 18:30:00.000000+0000 | TECHNICAL | MANAGER | FARAH | ['YOUTUBE', 'SPOTIFY'] | 95000

3 | 2010-04-25 18:30:00.000000+0000 | MARKETING | ASSISTANT MANAGER | CHIRAG | ['YOUTUBE', 'SPOTIFY'] | 75000

**3. Perform the following DB operations using Cassandra.**

**1.Create a keyspace by name Library**

CREATE KEYSPACE Library WITH REPLICATION = {'class':'SimpleStrategy','replication\_factor':1};

**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**

CREATE TABLE LIBRARY\_INFO\_4 (STUD\_ID INT, COUNTER\_VALUE COUNTER, STUD\_NAME TEXT, BOOK\_NAME TEXT, BOOK\_ID INT, DATE\_OF\_ISSUE TIMESTAMP, PRIMARY KEY( STUD\_ID, STUD\_NAME, BOOK\_NAME, BOOK\_ID, DATE\_OF\_ISSUE));

**3. Insert the values into the table in batch**

UPDATE LIBRARY\_INFO\_4 SET COUNTER\_VALUE+1 WHERE STUD\_ID=121 AND STUD\_NAME=’SNEHA’ AND BOOK\_NAME=’BDA’ AND BOOK\_ID=110 AND DATE\_OF\_ISSUE=’2022-04-01’;

UPDATE LIBRARY\_INFO\_4 SET COUNTER\_VALUE+1 WHERE STUD\_ID=122 AND STUD\_NAME=’RAHUL’ AND BOOK\_NAME=’OOMD’ AND BOOK\_ID=111 AND DATE\_OF\_ISSUE=’2022-07-03’;

UPDATE LIBRARY\_INFO\_4 SET COUNTER\_VALUE+1 WHERE STUD\_ID=123 AND STUD\_NAME=’RITIKA’ AND BOOK\_NAME=’ML’ AND BOOK\_ID=112 AND DATE\_OF\_ISSUE=’2022-02-21’;

UPDATE LIBRARY\_INFO\_4 SET COUNTER\_VALUE+1 WHERE STUD\_ID=124 AND STUD\_NAME=’ISHA’ AND BOOK\_NAME=’AI’ AND BOOK\_ID=113 AND DATE\_OF\_ISSUE=’2022-09-02’;

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

SELECT \* FROM LIBRARY\_INFO\_4;

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

SELECT \* FROM LIBRARY\_INFO\_4 WHERE STUD\_ID=112;

**6. Export the created column to a csv file.**

COPY LIBRARY\_INFO\_4 (STUD\_ID, STUD\_NAME, BOOK\_NAME, BOOK\_ID, DATE\_OF\_ISSUE, COUNTER\_VALUE) TO ‘C:\Users\Admin\OneDrive\Desktop\BDA Lab\data.csv’;

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

COPY LIBRARY\_INFO\_4 (STUD\_ID, STUD\_NAME, BOOK\_NAME, BOOK\_ID, DATE\_OF\_ISSUE, COUNTER\_VALUE) FROM ‘C:\Users\Admin\OneDrive\Desktop\BDA Lab\data.csv’;