WEEK-2

Perform the following DB operations using MongoDB

Create a collection by the name **blogPosts** and it has **3 fields id, title and comments**.
In the collection the **comments** field is an array which consists of user details. Each collection consists of two user details inside the **comments array**- user name and text

db.createCollection("blogPosts")

```
Atlas atlas-axcx6s-shard-0 [primary] lab3> db.createCollection("blogPosts")
{ ok: 1 }
Atlas atlas-axcx6s-shard-0 [primary] lab3> show collections
blogPosts
Atlas atlas-axcx6s-shard-0 [primary] lab3> db.blogPosts.insertOne({
...     id: 1,
...     title: "Sample Title",
...     comments: [
...         { user: "User1", text: "Comment1" }
...     ]
... })
{
  acknowledged: true,
  insertedId: ObjectId("660bc8a8c0cf4e885fcbb2e3")
}
```

Demonstrate the following

1. Adding an element into array

db.blogPosts.insertOne({
id: 1,
title: "Sample Title",
comments: [
{ user: "User1", text: "Comment1" }
]
})

(Similarly, Insert 4 ids)

```
}
Atlas atlas-57yq38-shard-0 [primary] test> db.blogPosts.find()
[
  {
    _id: ObjectId('660bcdd48adf462691a26f41'),
    id: 1,
    title: 'Sample Title',
    comments: [ { user: 'User1', text: 'Comment1' } ]
  }
]
```

```
Atlas atlas-axcx6s-shard-0 [primary] lab3> db.blogPosts.find()
[
  {
    _id: ObjectId("660bc8a8c0cf4e885fcbb2e3"),
    id: 1,
    title: 'Sample Title',
    comments: [ { user: 'User1', text: 'Comment1' } ]
  },
  {
    _id: ObjectId("660bc98ac0cf4e885fcbb2e4"),
    id: 2,
    title: 'Title2',
    comments: [ { user: 'User2', text: 'Comment2' } ]
  },
  {
    _id: ObjectId("660bc9cdc0cf4e885fcbb2e5"),
    id: 3,
    title: 'Title3',
    comments: [ { user: 'User3', text: 'Comment3' } ]
  },
  {
    _id: ObjectId("660bc9dec0cf4e885fcbb2e6"),
    id: 4,
    title: 'Title4',
    comments: [ { user: 'User4', text: 'Comment4' } ]
  }
```

2. Display second element

db.blogPosts.find().skip(1).limit(1)

```
Atlas atlas-axcx6s-shard-0 [primary] lab3> db.blogPosts.find().skip(1).limit(1)
[
  {
    _id: ObjectId("660bc98ac0cf4e885fcbb2e4"),
    id: 2,
    title: 'Title2',
    comments: [ { user: 'User2', text: 'Comment2' } ]
  }
]
```

3. Display size of the array

db.blogPosts.aggregate([
{ $project: { commentsCount: { $size: "$comments" } } }
])

```
Atlas atlas-axcx6s-shard-0 [primary] lab3> db.blogPosts.aggregate([
...     { $project: { commentsCount: { $size: "$comments" } } }
... ])
[
  { _id: ObjectId("660bc8a8c0cf4e885fcbb2e3"), commentsCount: 1 },
  { _id: ObjectId("660bc98ac0cf4e885fcbb2e4"), commentsCount: 1 },
  { _id: ObjectId("660bc9cdc0cf4e885fcbb2e5"), commentsCount: 1 },
  { _id: ObjectId("660bc9dec0cf4e885fcbb2e6"), commentsCount: 2 }
]
```

4. Display first two elements of the array

db.blogPosts.aggregate([
{ $project: { firstTwoComments: { $slice: ["$comments", 2] } } }
])

```
Atlas atlas-axcx6s-shard-0 [primary] lab3> db.blogPosts.aggregate([
...     { $project: { firstTwoComments: { $slice: ["$comments", 2] } } }
... ])
[
  {
    _id: ObjectId("660bc8a8c0cf4e885fcbb2e3"),
    firstTwoComments: [ { user: 'User1', text: 'Comment1' } ]
  },
  {
    _id: ObjectId("660bc98ac0cf4e885fcbb2e4"),
    firstTwoComments: [ { user: 'User2', text: 'Comment2' } ]
  },
  {
    _id: ObjectId("660bc9cdc0cf4e885fcbb2e5"),
    firstTwoComments: [ { user: 'User3', text: 'Comment3' } ]
  },
  {
    _id: ObjectId("660bc9dec0cf4e885fcbb2e6"),
    firstTwoComments: [
      { user: 'User4', text: 'Comment4' },
      [ { user: 'NewUser', text: 'NewComment' } ]
    ]
  }
]
```

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

db.blogPosts.updateOne(
{ id: 4 },
{ $set: { "comments.1": [{ user: "NewUser", text: "NewComment" }] } }
)

```
Atlas atlas-axcx6s-shard-0 [primary] lab3> db.blogPosts.updateOne( { id: 4 }, { $set: { "comments.1": [{ user: "NewUser"
, text: "NewComment" }] } } )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
Atlas atlas-axcx6s-shard-0 [primary] lab3> db.blogPosts.find()
[
  {
    _id: ObjectId("660bc8a8c0cf4e885fcbb2e3"),
    id: 1,
    title: 'Sample Title',
    comments: [ { user: 'User1', text: 'Comment1' } ]
  },
  {
    _id: ObjectId("660bc98ac0cf4e885fcbb2e4"),
    id: 2,
    title: 'Title2',
    comments: [ { user: 'User2', text: 'Comment2' } ]
  },
  {
    _id: ObjectId("660bc9cdc0cf4e885fcbb2e5"),
    id: 3,
    title: 'Title3',
    comments: [ { user: 'User3', text: 'Comment3' } ]
  },
  {
    _id: ObjectId("660bc9dec0cf4e885fcbb2e6"),
    id: 4,
    title: 'Title4',
    comments: [
      { user: 'User4', text: 'Comment4' },
      [ { user: 'NewUser', text: 'NewComment' } ]
    ]
  }
```

Cassandra basics

Learn Cassandra Tutorial - javatpoint

6. Execute the following

Create KeySpace :

CREATE KEYSPACE Students WITH REPLICATION =
{'class':'SimpleStrategy','replication_factor':1};

Describe the existing Keyspaces:

DESCRIBE KEYSPACES;

For More details on existing keyspaces:

SELECT * FROM system.schema_keyspaces;

use the keyspace "Students":

USE Students;

To create table (column family) by name Student_Info:

CREATE TABLE Students_Info (Roll_No int PRIMARY KEY, StudName text, DateOfJoining timestamp, last_exam_Percent double);

Lookup the names of all tables in the current keyspaces
DESCRIBE TABLES;

Describe the table information

DESCRIBE TABLE <Table_Name>;

CRUD

Insert :

BEGIN BATCH
INSERT INTO Students_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (1,'Asha','2012-03-12',79.9)
INSERT INTO Students_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (1,'Krian','2012-03-12',89.9)
INSERT INTO Students_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (1,'Tarun','2012-03-12',78.9)
INSERT INTO Students_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (1,'Samrth','2012-03-12',90.9)
INSERT INTO Students_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)

VALUES (1,'Smitha','2012-03-12',67.9)
INSERT INTO Students_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (1,'Rohan','2012-03-12',56.9)
APPLY BATCH;

View data from the table "Students_Info"

SELECT * FROM Students_Info;

View data from the table "Students_Info" where RoolNo column either has a value 1 or 2 or 3

SELECT * FROM Students_Info WHERE Roll_No IN (1,2,3);

To execute a non primary key - will throw an error
select * from students_info where Studname= 'Asha';

So create an INDEX on the Column as below:
To create an INDEX on StudName Column of the Students_Info column family

CREATE INDEX ON Students_Info (StudName);

Now execute the query based on the INDEXED Column:
select * from students_info where Studname= 'Asha';

To specify the number of rows retured in the output
select Roll_No, StudName from students_info LIMIT 2;

Alias for Column:

Select Roll_No as "USN" from Students_info;

UPDATE

UPDATE students_info SET StudName='David Sheen' WHERE RollNo=2;

Lets try to update the primary key

UPDATE students_info SET rollno=6 WHERE rollno=3;

DELETE
DELETE LastExamPercent FROM students_info WHERE RollNo=2;

Delete a Row
DELETE FROM student_info WHERE RollNo=2;