LAB 2

Perform the following DB operations using Cassandra

1. Create a keyspace by name Employee

```
cqlsh:library> CREATE KEYSPACE Employee WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 1 };
cqlsh:library>
```

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>
cqlsh:employee> CREATE TABLE Employee_Info (
           ...     Emp_Id int PRIMARY KEY,
           ...     Emp_Name text,
           ...     Designation text,
           ...     Date_of_Joining date,
           ...     Salary decimal,
           ...     Dept_Name text
           ... );
```

3. Insert the values into the table in batch

```
cqlsh:employee> BEGIN BATCH
           ... INSERT INTO Employee_Info (Emp_Id, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name)
           ... VALUES (101, 'John Doe', 'Manager', '2023-01-01', 50000, 'HR');
           ... INSERT INTO Employee_Info (Emp_Id, Emp_Name, Designation, Date_of_Joining, Salary, Dept_Name)
           ... VALUES (121, 'Jane Smith', 'Developer', '2023-02-01', 60000, 'IT');
           ... APPLY BATCH;
```

```
cqlsh:employee> UPDATE Employee_Info SET Emp_Name = 'Jane Johnson', Dept_Name = 'Engineering' WHERE Emp_Id = 121;
cqlsh:employee> SELECT * FROM Employee_Info;

 emp_id | date_of_joining | dept_name   | designation | emp_name     | salary
--------+-----------------+-------------+-------------+--------------+--------
    121 |      2023-02-01 | Engineering |   Developer | Jane Johnson |  60000
    101 |      2023-01-01 |          HR |     Manager |     John Doe |  50000

(2 rows)
```

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

5. Sort the details of Employee records based on salary

```
cqlsh:employee> paging off
Disabled Query paging.
cqlsh:employee> SELECT * FROM Employee_Info WHERE Emp_Id IN (121,101) ORDER BY Salary ALLOW FILTERING;

 emp_id | salary | date_of_joining | dept_name | designation | emp_name
cqlsh:employee> UPDATE Employee_Info SET Projects = {'ProjectA', 'ProjectB'} WHERE Emp_Id = 101 and salary=50000;
cqlsh:employee> UPDATE Employee_Info SET Projects = {'ProjectC'} WHERE Emp_Id = 121 and salary=60000;
cqlsh:employee> select * from Employee_Info;

 emp_id | salary | date_of_joining | dept_name | designation | emp_name   | projects
--------+--------+-----------------+-----------+-------------+------------+--------------------------
    121 |  60000 |      2023-02-01 |        IT |   Developer | Jane Smith |            {'ProjectC'}
    101 |  50000 |      2023-01-01 |        HR |     Manager |   John Doe | {'ProjectA', 'ProjectB'}

(2 rows)
```

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.

7. Update the altered table to add project names.

```
cqlsh:employee> UPDATE Employee_Info SET Projects = {'ProjectA', 'ProjectB'} WHERE Emp_Id = 101 and salary=50000;
cqlsh:employee> UPDATE Employee_Info SET Projects = {'ProjectC'} WHERE Emp_Id = 121 and salary=60000;
cqlsh:employee> select * from Employee_Info;

 emp_id | salary | date_of_joining | dept_name | designation | emp_name   | projects
--------+--------+-----------------+-----------+-------------+------------+--------------------------
    121 |  60000 |      2023-02-01 |        IT |   Developer | Jane Smith |            {'ProjectC'}
    101 |  50000 |      2023-01-01 |        HR |     Manager |   John Doe | {'ProjectA', 'ProjectB'}

(2 rows)
```

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