

LABORATORY PROGRAM – 3

Perform the following DB operations using Cassandra

Questions:

- Create a keyspace by name Library
- 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
- Insert the values into the table in batch
- Display the details of the table created and increase the value of the counter
- Write a query to show that a student with id 112 has taken a book “BDA” 2 times.
- Export the created column to a csv file
- Import a given csv dataset from local file system into Cassandra column family

OBSERVATION

COMMAND WITH OUTPUT

```
cqlsh:employee> CREATE KEYSPACE IF NOT EXISTS Library
... WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
cqlsh:employee> USE Library;
cqlsh:library> CREATE TABLE IF NOT EXISTS Library_Info (
...     Stud_Id INT PRIMARY KEY,
...     Stud_Name TEXT,
...     Book_Name TEXT,
...     Book_Id TEXT,
...     Date_of_issue DATE
... );
cqlsh:library> CREATE TABLE IF NOT EXISTS Book_Counter (
...     Stud_Id INT,
...     Book_Name TEXT,
...     Counter_value COUNTER,
...     PRIMARY KEY ((Stud_Id), Book_Name)
... );
cqlsh:library> BEGIN BATCH
... INSERT INTO Library_Info (Stud_Id, Stud_Name, Book_Name, Book_Id, Date_of_issue)
... VALUES (112, 'Anjali Rao', 'BDA', 'B101', '2024-10-01');
...
... INSERT INTO Library_Info (Stud_Id, Stud_Name, Book_Name, Book_Id, Date_of_issue)
... VALUES (113, 'Karthik N', 'AI', 'B102', '2024-11-11');
... APPLY BATCH;
cqlsh:library> UPDATE Book_Counter SET Counter_value = Counter_value + 1 WHERE Stud_Id = 112 AND Book_Name = 'BDA';
cqlsh:library> UPDATE Book_Counter SET Counter_value = Counter_value + 1 WHERE Stud_Id = 112 AND Book_Name = 'BDA';
cqlsh:library> SELECT * FROM Book_Counter WHERE Stud_Id = 112 AND Book_Name = 'BDA';

stud_id | book_name | counter_value
-----+-----
112 | BDA | 4
(1 rows)
```

```
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 mentable = '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 mentable_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, 'Asha', '2012-03-12', 79.9);
... INSERT INTO Students_Info (Roll_No, StudName, DateOfJoining, Last_Exam_Percent)
... VALUES (2, 'Kiran', '2012-03-12', 89.9);
... INSERT INTO Students_Info (Roll_No, StudName, DateOfJoining, Last_Exam_Percent)
... VALUES (3, 'Shanthi', '2012-03-12', 90.9);
... INSERT INTO Students_Info (Roll_No, StudName, DateOfJoining, Last_Exam_Percent)
... VALUES (4, 'Smith', '2012-03-12', 67.9);
... INSERT INTO Students_Info (Roll_No, StudName, DateOfJoining, Last_Exam_Percent)
... VALUES (5, 'Rohan', '2012-03-12', 56.9);
... APPLY BATCH;

cqlsh:students> SELECT * FROM Students_Info;

roll_no | dateofjoining | last_exam_percent | studname
-----|-----|-----|-----
5 | 2012-03-11 18:30:00.000000+0000 | 56.9 | Rohan
1 | 2012-03-11 18:30:00.000000+0000 | 79.9 | Asha
2 | 2012-03-11 18:30:00.000000+0000 | 89.9 | Kiran
4 | 2012-03-11 18:30:00.000000+0000 | 67.9 | Smith
3 | 2012-03-11 18:30:00.000000+0000 | 90.9 | Shanthi

(5 rows)
```

```
cqlsh> CREATE KEYSPACE Students WITH REPLICATION =
... {'class': 'SimpleStrategy', 'replication_factor': '1'};
cqlsh>
cqlsh> USE Students;
cqlsh:students> DESCRIBE KEYSPACES;

companies | library | products | system | system_traces
company | pro | productsss | system_auth | system_views
employe | prod | productsss | system_distributed | system_virtual_schema
employee | productname | students | system_schema

cqlsh:students> CREATE TABLE Students_Info (
... Roll_No int PRIMARY KEY,
... StudName text,
... DateOfJoining timestamp,
... last_exam_percent double
... );

cqlsh:students> SELECT * FROM system.schema_keyspaces;
InvalidRequest: Error from server: code=2200 [Invalid query] message="table schema_keyspaces does not exist"
cqlsh:students>
cqlsh:students> SELECT * FROM system_schema.keyspaces;

keyspace_name | durable_writes | replication
-----|-----|-----
companies | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
system_auth | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
system_schema | True | {'class': 'org.apache.cassandra.locator.LocalStrategy'}
library | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
products | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
system_distributed | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
system | True | {'class': 'org.apache.cassandra.locator.LocalStrategy'}
productsss | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
prod | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
pro | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
system_traces | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '2'}
students | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
company | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
employee | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
productname | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
employe | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
productsss | True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}

(17 rows)
cqlsh:students> DESCRIBE TABLES;

students_info
```

```

cqlsh:students> SELECT * FROM Students_Info WHERE Roll_No IN (1,2,3);

roll_no | dateofjoining | last_exam_percent | studname
-----+-----+-----+-----
1 | 2012-03-11 18:30:00.000000+0000 | 79.9 | Asha
2 | 2012-03-11 18:30:00.000000+0000 | 89.9 | Kiran
3 | 2012-03-11 18:30:00.000000+0000 | 90.9 | Shanthi

(3 rows)
cqlsh:students> CREATE INDEX ON Students_Info (StudName);
cqlsh:students> SELECT * FROM Students_Info WHERE StudName = 'Asha';

roll_no | dateofjoining | last_exam_percent | studname
-----+-----+-----+-----
1 | 2012-03-11 18:30:00.000000+0000 | 79.9 | Asha

(1 rows)
cqlsh:students> SELECT Roll_No, StudName FROM Students_Info LIMIT 2;

roll_no | studname
-----+-----
5 | Rohan
1 | Asha

(2 rows)
cqlsh:students> SELECT Roll_No AS USN FROM Students_Info;

usn
----
5
1
2
4
3

(5 rows)
cqlsh:students> UPDATE Students_Info
... SET StudName = 'David Sheen'
... WHERE Roll_No = 2;
cqlsh:students> UPDATE Students_Info SET Roll_No = 6 WHERE Roll_No = 3; -- ✗ ERROR!
InvalidRequest: Error from server: code=2200 [Invalid query] message="PRIMARY KEY part roll_no found in SET part"

```

```

cqlsh:students> DELETE Last_Exam_Percent FROM Students_Info WHERE Roll_No = 2;
cqlsh:students> DELETE FROM Students_Info WHERE Roll_No = 2;
cqlsh:students> ALTER TABLE Students_Info ADD hobbies SET<text>;
cqlsh:students> ALTER TABLE Students_Info ADD languages LIST<text>;
cqlsh:students> UPDATE Students_Info
... SET hobbies = hobbies + {'Chess', 'Table Tennis'}
... WHERE Roll_No = 1;
cqlsh:students> CREATE TABLE library_book (
... counter_value counter,
... book_name text,
... stud_name text,
... PRIMARY KEY(book_name, stud_name)
... );
cqlsh:students> UPDATE library_book
... SET counter_value = counter_value + 1
... WHERE book_name = 'Big Data Analytics' AND stud_name = 'Jeet';
cqlsh:students> CREATE TABLE userlogin (
... userid int PRIMARY KEY,
... password text
... );
cqlsh:students> INSERT INTO userlogin (userid, password)
... VALUES (1, 'infy') USING TTL 30;
cqlsh:students> SELECT TTL(password) FROM userlogin WHERE userid = 1;

ttl(password)
-----
20

(1 rows)
cqlsh:students> COPY Students_Info TO '/home/bmscece/Desktop/Student_Info.csv';
Using 16 child processes

Starting copy of students.students_info with columns [roll_no, dateofjoining, hobbies, languages, last_exam_percent, studname].
Processed: 4 rows; Rate: 38 rows/s; Avg. rate: 38 rows/s
4 rows exported to 1 files in 0.124 seconds.
cqlsh:students> COPY Students_Info FROM '/home/bmscece/Desktop/Student_Info.csv';
Using 16 child processes

Starting copy of students.students_info with columns [roll_no, dateofjoining, hobbies, languages, last_exam_percent, studname].
Processed: 4 rows; Rate: 7 rows/s; Avg. rate: 11 rows/s
4 rows imported from 1 files in 0.377 seconds (0 skipped).
cqlsh:students> COPY person (id, fname, lname) FROM STDIN;
column family person not found
cqlsh:students> COPY Students_Info TO STDOUT;
5,2012-03-11 18:30:00.000+0000,,,56.9,Rohan
1,2012-03-11 18:30:00.000+0000,"{'Chess', 'Table Tennis'}",,79.9,Asha
4,2012-03-11 18:30:00.000+0000,,,67.9,Smith
3,2012-03-11 18:30:00.000+0000,,,90.9,Shanthi
cqlsh:students>

```