Exercise: 6

Table and Batch Operations in Cassandra with DataStax Astra

Objectives:

- **To understand the basic operations related to tables in Cassandra.**
- **To practice batch operations for efficient data manipulation.**
- **To gain proficiency in querying and updating data in Cassandra tables.**

Experiment:

1. Display the set of key spaces

```
select * from system.schema_keyspaces;
```

(OR)

```
describe keyspaces;
```

2. Connect with 'default_keyspace' key space

```
use default_keyspace;
```

3. Create a table named users with columns for user_id, fname, and lname.

```
CREATE TABLE users (user_id int PRIMARY KEY,fname text,lname text);
```

4. Insert data into the users table one by one

```
INSERT INTO users (user_id, fname, lname) VALUES (1745, 'John', 'Smith');

INSERT INTO users (user_id, fname, lname) VALUES (1746, 'Jane', 'Doe');

INSERT INTO users (user_id, fname, lname) VALUES (1747, 'Alice', 'Johnson');
```

5. Insert multiple rows into the users table in Cassandra using a batch operation.

```
BEGIN BATCH
   INSERT INTO users (user_id, fname, lname) VALUES (1745, 'John', 'Smith');
   INSERT INTO users (user_id, fname, lname) VALUES (1746, 'Jane', 'Doe');
APPLY BATCH;
```

6. Display all data from the users table.

```
SELECT * FROM users;
```

7. Display only the user_id and fname columns from the users table

SELECT user_id, fname FROM users;

8. Display data from the users table where lname is 'Smith'.

SELECT * FROM users WHERE lname = 'Smith';

9.Display data from the users table sorted by lname in descending order.

SELECT * FROM users ORDER BY lname DESC;

10.Display the first two rows from the users table.

SELECT * FROM users LIMIT 2;

11.Count the number of users with the same lname.

SELECT lname, COUNT(*) as count FROM users GROUP BY lname;

12. Display unique last names from the users table.

SELECT DISTINCT lname FROM users;

13. Update the telephone field for the user with user_id = 1745 to '21212121'.

UPDATE users SET telephone = '21212121' WHERE user_id = 1745;

14.Update both fname and lname of the user with user_id 1746.

UPDATE users SET fname = 'Jane', lname = 'Smith' WHERE user_id = 1746;

15. Increment the login_attempts counter for the user with user_id 1747 by 1.

UPDATE users SET login_attempts = login_attempts + 1 WHERE user_id = 1747;

16.Update the email of the user with user_id 1748 only if it's currently null.

UPDATE users SET email = 'example@example.com' WHERE user_id = 1748 IF email IS NULL;

17.Update the password of the user with user_id 1745 only if the current password matches a specific value.

UPDATE users SET password = 'new_password' WHERE user_id = 1745 IF password = 'old_password';

18. Use different types of data modification operations within a single batch operation in Cassandra.

BEGIN BATCH

```
    INSERT INTO users (user_id, fname, lname) VALUES (1745, 'John', 'Smith');
    UPDATE users SET fname = 'Jonathan' WHERE user_id = 1746;
    DELETE FROM users WHERE user_id = 1747;
APPLY BATCH;
```

19.Update multiple rows in a batch operation.

```
BEGIN BATCH
    UPDATE users SET status = 'active' WHERE user_id = 1745;
    UPDATE users SET status = 'inactive' WHERE user_id = 1746;
APPLY BATCH;
```

20. Perform conditional updates on multiple rows using a batch operation

```
BEGIN BATCH
    UPDATE users SET fname = 'Jonathan' WHERE user_id = 1745 IF lname = 'Smith';
    UPDATE users SET lname = 'Doe' WHERE user_id = 1746 IF fname = 'Jane';
APPLY BATCH;
```

21. Add a new field named telephone to the users table.

```
ALTER TABLE users ADD telephone text;
```

22.Delete the user with user_id 1745 from the users table.

```
DELETE FROM users WHERE user_id = 1745;
```

23. Delete the email column value for the user with user_id 1746.

```
DELETE email FROM users WHERE user_id = 1746;
```

24.Delete multiple users in a batch operation.

```
BEGIN BATCH
    DELETE FROM users WHERE user_id = 1747;
    DELETE FROM users WHERE user_id = 1748;
APPLY BATCH;
```

25. Remove the users table.

DROP TABLE users;

26. Remove all data from the users table.

TRUNCATE users;

27. Create a table with a composite key

CREATE TABLE tab2 (id1 int, id2 int, first_name varchar, last_name varchar, PRIMARY KEY (id1, id2));

Web References:

1. https://www.collegesidekick.com/study-docs/691456
2. https://www.tutorialspoint.com/cassandra/index.htm

Video References:

1. https://youtu.be/-h0vQVREfXs
2. https://www.youtube.com/watch?v=Mh1q0Yldo3Y&list=PL_YfpwdeYvKgcfuCyPX_NPwqDoXm83AnC
3. https://www.youtube.com/watch?v=YjYWsN1vek8&list=PL2g2h-wyI4SqCdxdiyi8enEyWvACcUa9R

Exercise: 7

Working with Collections in Cassandra using DataStax Astra

Objectives:

- **To understand and implement SET collections in Cassandra.**
- **To explore and practice LIST collections in Cassandra.**
- **To learn and apply MAP collections in Cassandra.**
- **To gain proficiency in data modeling using collections in Cassandra.**

Experiment:

Collections - SETS

A set is an unordered set of values. It is possible to solve the problem of multi-valued fields, like emails.

1. create a table named users in Cassandra with a column named emails as a set of text values

```
CREATE TABLE users (
   user_id int PRIMARY KEY,
   fname text,
   lname text,
   emails set<text>
);
```

2. insert data into the emails set for a user with user_id 1234.

```
INSERT INTO users (user_id, fname, lname, emails)
VALUES(1234, 'Frodo', 'Baggins', {'f@baggins.com', 'baggins@gmail.com'});
```

3. Retrieve the user_id and emails for the user with user_id 1234.

```
SELECT user_id, emails FROM users WHERE user_id = 1234;
```

4. Add the email address 'fb@friendsofmordor.org' to the set of emails for the user with user_id 1234

```
UPDATE users SET emails = emails + {'fb@friendsofmordor.org'} WHERE user_id = 1234;
```

5. Retrieve the user_id and updated emails for the user with user_id 1234 after adding the new email.

SELECT user_id, emails FROM users WHERE user_id = 1234;

6.Remove the email address 'fb@friendsofmordor.org' from the set of emails for the user with user_id 1234

UPDATE users SET emails = emails - {'fb@friendsofmordor.org'} WHERE user_id = 1234;

7.Remove all email addresses from the set for the user with user_id 1234.

UPDATE users SET emails = {} WHERE user_id = 1234;

(OR)

DELETE emails FROM users WHERE user_id = 1234;

Collections - LISTS

A list is a sorted collection of non-unique values where elements are ordered by their positions in the list.

1. Alter the users table to include a column named top_places as a list of text values

ALTER TABLE users ADD top_places list<text>;

2. Insert a list of the best places visited into the top_places column for the user with user_id 1234.

UPDATE users SET top_places = ['rivendell', 'rohan'] WHERE user_id = 1234;

3.Add the place 'mordor' to the end of the list of top_places for the user with user_id 1234

UPDATE users SET top_places = top_places + ['mordor'] WHERE user_id = 1234;

4. Update the second place in the list top_places to 'riddermark' for the user with user_id 1234.

UPDATE users SET top_places[1] = 'riddermark' WHERE user_id = 1234;

5.Remove the third place from the top_places list for the user with user_id 1234

DELETE top_places[3] FROM users WHERE user_id = 1234;

6.Remove all occurrences of 'riddermark' from the top_places list for the user with user_id 1234.

UPDATE users SET top_places = top_places - ['riddermark'] WHERE user_id = 1234;

7.Retrieve the user_id and top_places list for the user with user_id 1234.

SELECT user_id, top_places FROM users WHERE user_id = 1234;

Collections - MAP

A map allows us to associate two elements, in the form of a key / value. It can be used, for example, to save the schedules of different events in a user profile. Using Cassandra, each element in a Map is stored as a column that we can modify, replace and query.

1.Alter the users table to include a column named todo as a map with timestamp keys and text values

ALTER TABLE users ADD todo map<timestamp, text>;

2. Insert tasks into the todo map for the user with user_id 1234.

UPDATE users SET todo = { '2012-9-24' : 'enter mordor', '2012-10-2 12:00' : 'throw ring into mount doom' } WHERE user_id = 1234;

3. Add a task with the timestamp '2012-10-2 12:00' to the todo map for the user with user_id 1234

UPDATE users SET todo['2012-10-2 12:00'] = 'throw my precious into mount doom' WHERE user_id = 1234;

4. Insert tasks into the todo map for the user with user_id 1234 using the INSERT statement.

INSERT INTO users (user_id, todo) VALUES (1234, { '2013-9-22 12:01' : 'birthday wishes to Bilbo', '2013-10-1 18:00' : 'Check into Inn of Prancing Pony' });

5. Remove the task with the timestamp '2012-9-24' from the todo map for the user with user_id 1234

DELETE todo['2012-9-24'] FROM users WHERE user_id = 1234;

6.Retrieve the user_id and todo map for the user with user_id 1234.

SELECT user_id, todo FROM users WHERE user_id = 1234;

Web References:

1. https://www.collegesidekick.com/study-docs/691456
2. https://www.tutorialspoint.com/cassandra/index.htm

Video References:

1. https://youtu.be/cFpfRTe5u20
2. https://youtu.be/uGUgqZQD6FU
3. https://youtu.be/2YxV1Ia0evc