

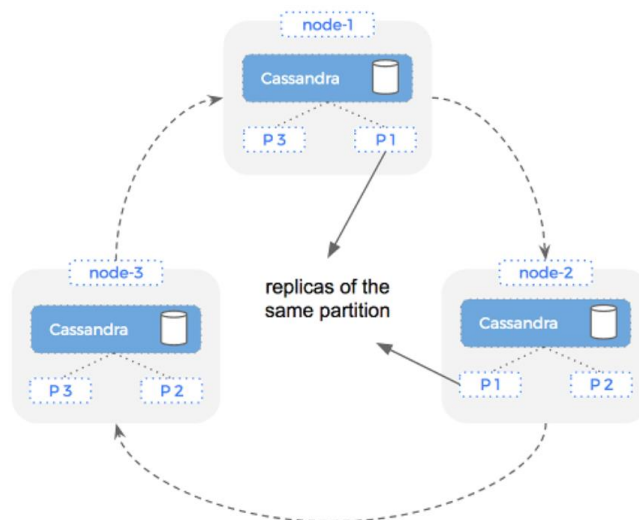
# CSE 512- Distributed Database Systems

## Group Project: Cassandrian

### Part 5: Distributed NoSQL Database Systems Implementation

Based on the sample database found on Kaggle, the most suitable NoSQL database to cater to the requirements was **Cassandra**, because of its column-based database.

We created a Cassandra cluster of three nodes using docker, namely cass1, cass2, cass3. Cass1 & cass2 are seed nodes, i.e., cass1 & cass2 will be the point of contact for the cluster and also contain all the information about the cluster. The distributed nature of Cassandra cluster ensures high availability.



```
pradn@Pradnya MINGW64 ~
$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                                                                 NAMES
e3b62b3550c8   cassandra "docker-entrypoint.s..." 15 minutes ago Up 47 seconds (health: starting) 7000-7001/tcp, 7199/tcp, 9160/tcp, 0.0.0.0:9044->9042/tcp cass3
5de5d4711d0d   cassandra "docker-entrypoint.s..." 15 minutes ago Up 47 seconds (health: starting) 7000-7001/tcp, 7199/tcp, 9160/tcp, 0.0.0.0:9043->9042/tcp cass2
a90461c8ff25   cassandra "docker-entrypoint.s..." 15 minutes ago Up 36 seconds (health: starting) 7000-7001/tcp, 7199/tcp, 9160/tcp, 0.0.0.0:9042->9042/tcp cass1
```

<input type="checkbox"/>	pradn	Running (3/3)	15.12%	3 minutes ago
<input type="checkbox"/>	<b>cass2</b> 5de5d4711d0d	cassandra	Running	6.08% 9043-9042 3 minutes ago
<input type="checkbox"/>	<b>cass1</b> a90461c8ff25	cassandra	Running	4.58% 9042-9042 3 minutes ago
<input type="checkbox"/>	<b>cass3</b> e3b62b3550c8	cassandra	Running	4.46% 9044-9042 3 minutes ago

The replication factor for the “commentary\_keyspace” is set to 2, which would create 2 replicas of each partition of the keyspace across the cluster.

## Data Schema & Data Model –

### Table – commentary

Commentary table stores commentary information for each ball in a match.

```
CREATE TABLE IF NOT EXISTS commentary (  
    Over_No TEXT,  
    Over_Score TEXT,  
    Short_comm TEXT,  
    Commentary TEXT,  
    Bold_Comm TEXT,  
    Innings_ID TEXT,  
    Ball_ID TEXT,  
    Match_ID TEXT,  
    PRIMARY KEY (Innings_ID, Ball_ID)  
);
```

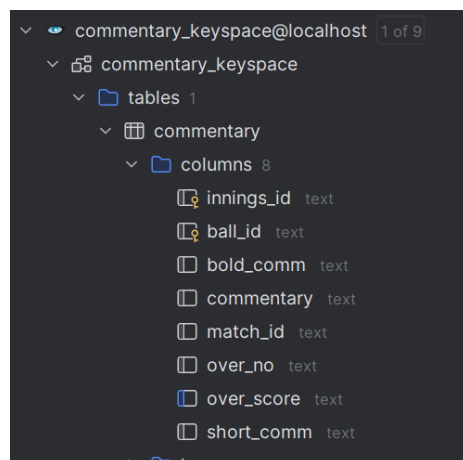
- Over\_No: Over number (1.1, 3.5, etc)
- Over\_score: Number of runs scored for that ball along with indication of wicket, byes, etc
- Short\_comm: Short Commentary
- Commentary: Long version of commentary with all details
- Bold\_comm: Comments made for entertainment
- Innings\_ID: Identifies the innings of the match. It is a combination of match\_ID & inning number eg. 1181768-1, 1181768-2
- Ball\_ID: Identifies each ball within an innings
- Match\_ID: References the match in which the commentary occurs

Innings\_ID & Ball\_ID together make the composite primary key.

## CRUD operations –

### 1. Create Commentary table:

Create\_table() method creates commentary table using the “create table” query



2. Insert the sample data:

Insert\_csv\_file\_data() method inserts the data from sample csv file – ipl2019\_final.csv into commentary table.

```
Inserted 13514 rows of data into commentary table
```

3. Sample read query:

Read\_query() method selects over\_no, over\_score, short\_comm, innings\_ID, ball\_ID of 5 entries.

```
Over_No: 19.6, Over_Score: 2, Short_comm: ['Bravo to HH Pandya, 2 runs'], Innings_ID: 1178419-2, Ball_ID: 0
Over_No: 19.5, Over_Score: 6, Short_comm: ['Bravo to HH Pandya, SIX runs'], Innings_ID: 1178419-2, Ball_ID: 1
Over_No: 18.2, Over_Score: 1, Short_comm: ['Chahar to HH Pandya, 1 run'], Innings_ID: 1178419-2, Ball_ID: 10
Over_No: 3.4, Over_Score: 0, Short_comm: ['Harbhajan Singh to Lewis, no run'], Innings_ID: 1178419-2, Ball_ID: 100
Over_No: 3.3, Over_Score: 0, Short_comm: ['Harbhajan Singh to Lewis, no run'], Innings_ID: 1178419-2, Ball_ID: 101
```

4. Update query:

Update\_query() updates over\_score & short\_comm value of the entry with innings\_ID '1181768-1' and ball\_ID '51'

```
Result before update:
Over_No: 11.4, Over_Score: 0, Short_comm: ['Chahar to Dhoni, no run'], Commentary: goes right back and bunts this to cover, Bold_Comm: [], Innings_ID: 1181768-1, Ball_ID: 51, Match_ID: 1181768
Result after update:
Over_No: 11.4, Over_Score: 1, Short_comm: ['Chahar to Dhoni, 1 run'], Commentary: goes right back and bunts this to cover, Bold_Comm: [], Innings_ID: 1181768-1, Ball_ID: 51, Match_ID: 1181768
```

5. Delete query:

Delete\_query() method deletes entry with innings\_ID '1181768-1' and ball\_ID '2'

```
Row with Innings_ID = '1181768-1' and Ball_ID = '2' deleted.
```

The screenshot shows a database interface with a query editor and a results pane. The query editor contains the following SQL query:

```
Select count(*) as count from commentary WHERE Innings_ID = '1181768-1' and Ball_ID = '2'
```

The results pane shows the output of the query:

count
1

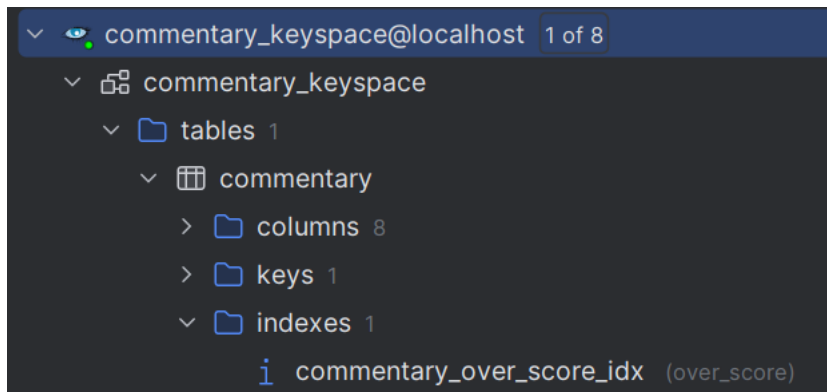
The interface also shows a database explorer on the left with the following structure:

- Database
  - commentary\_keyspace@localh
    - console\_1 204 ms
    - console\_1 204 ms

### Sample Queries to show data retrieval operations:

1. Get sixes by innings:

Get\_sixes\_by\_innings() method get the total number of sixes in every inning. As the where clause of this query contains a non-primary column over\_score, a index needs to be created on that column, so as to not affect the computation capability of Cassandra. Create\_index() is the method that creates index on over\_score column.



```
Number of sixes in every Innings -
Innings ID: 1178419-2, Sixes Count: 6
Innings ID: 1178420-2, Sixes Count: 1
Innings ID: 1175362-2, Sixes Count: 9
Innings ID: 1178431-2, Sixes Count: 10
Innings ID: 1178422-2, Sixes Count: 15
Innings ID: 1175370-2, Sixes Count: 6
Innings ID: 1175361-1, Sixes Count: 7
Innings ID: 1178408-1, Sixes Count: 3
Innings ID: 1175370-1, Sixes Count: 4
Innings ID: 1178411-2, Sixes Count: 6
Innings ID: 1181766-1, Sixes Count: 9
Innings ID: 1178412-2, Sixes Count: 8
Innings ID: 1178429-2, Sixes Count: 9
Innings ID: 1181764-2, Sixes Count: 4
Innings ID: 1178406-1, Sixes Count: 10
Innings ID: 1178417-2, Sixes Count: 12
Innings ID: 1178403-1, Sixes Count: 2
Innings ID: 1178404-2, Sixes Count: 7
```