

TASK 9

16/9/25

CRUD OPERATIONS IN GRAPH DATABASES

AIM: TO perform "CRUD operations" like creating, inserting, querying, finding, deleting operations on graph spaces

The steps to get started with neo4j's Aura Graph database.

Step 1: copy and paste the following link into your web browser.

<https://neo4j.com/cloud/platform/aura-graph-database/>

Step 2: click on "start free".

Step 3: choose option to "continue with google".

Step 4: click on open button.

Step 5: After clicking "open", a text file will automatically downloaded. This file contains your user ID and password details.

Step 6: copy password from download text file and paste it where required.

Step 7: Close "Get started with neo4j with beginner guides".

Step 8: you're now ready to begin practicing with graph database.

Create node with properties:

Properties are the key-value pairs using which node starts data. Create node with properties using CREATE clause need to specify these properties separated by commas within flower braces {}.

Syntax:

`CREATE n`

Creating Relationships

To create relationship using Create clause Specify relationship within square braces "[]" depending on direction of relationships placed between hyphen "-" or arrow "→" shown in following syntax

Syntax:

CREATE(node-1)-[:Relationship Type]→(node-2)

Syntax:

MATCH (a:Label of Node 1), (b:Label of Node 2) WHERE a.name = "name of node 1" AND b.name = "name of node 2"

CREATE (a)-[:Relationship]→(b) RETURN a,b

Deleting a Particular Node:

To delete particular node and need to specify details place "n" in above query

MATCH (node:Label {properties....}) DELETE node

Create graph database for student course registration,
Create student dept node insert value of properties.

Create team nodes:

Create (t₁:Team {team ID:'CC BO1', Board ID:'BID01', name:'ABS Express Coach', G.D RAMESH, captain 'SAMPATH KUMAR'}) return t₁

Create (t₂:Team {ID:'CC BO2', Board ID:'BID01', name:'Avg express', coach:T. Karthik, captain 'J. Y. JOHN') return t₂

Create Player nodes:

create (P_1 :player {player ID: '1', team ID: 'CCB01', name: 'Raj', Age: 23, Date of Birth: '02-JAN-1999', playing Role: 'Batsman', email: 'balajid@gmail.com'})
return P_1

create (P_2 :player {player ID: '33', team ID: 'CCB01', name: 'Anand', Age: 23, Date of Birth: '29-JUN-1996', playing Role: 'Batsman', email: 'sain@gmail.com'}) return P_2

create (P_3 :player {player ID: '65', team ID: 'CCB02', name: 'Kohit', Age: 33, Date of Birth: '02-JUN-1991', playing Role: 'Bowler', email: '@gmail.com'}) return P_3

Creating Relationship among Cricket Board and Teams:

match (cb:Cricket Board {Board ID: 'BID01'}, t₁: Team {team ID: 'CC B01'}) create (cb)-[r:has]→(t₁) return cb, r, t₁

match (cb:Cricket Board {Board ID: 'BID01'}, t₂: Team {team ID: 'CC B02'}) create (cb)-[r:has]→(t₂) return cb, r, t₂

Create Relationship among players and teams:

match (P_1 :player {player ID: '1'}), (t₁:team {team ID: 'CCB01'}) create (P_1)-[r]:play for→(t₁) return P_1 , r, t₁

match (P_2 :player {player ID: '33'}, t₁:team {team ID: 'CCB01'}) create (P_2)-[r₂:play for]→(t₁) return P_2 , r₂, t₁

match (P_3 :player {player ID: '65'}, t₂:team {team ID: 'CCB02'}) create (P_3)-[r₃:play for]→(t₂) return P_3 , r₃, t₂

match (P_4 :player {player ID: '75'}, t₂:team {team ID: 'CCB02'}) create (P_4)-[r₄:play for]→(t₂) return P_4 , r₄, t₂

Output:

The screenshot shows the Neo4j browser interface. On the left, the 'Database Information' sidebar lists 'Nodes (8)' (CricketTeam, Player, Team), 'Relationships (7)' (has_player), and 'Property keys'. The main area displays a graph with 8 nodes. A tooltip for one node says 'Node Details: Team'. The right side shows a 'Results Overview' panel with sections for 'Nodes (8)', '(CricketTeam (1)) -> Player (7)', and 'Team (1)'. The bottom status bar indicates 'Showing 0.111ms pre'.

OUTPUT:

The screenshot shows the Neo4j browser interface. A specific node representing a player is highlighted with a large red circle. The node has the label 'Player' and the ID '33'. A tooltip for this node says 'Node Details: Player'. The right side shows a 'Results Overview' panel with sections for 'Nodes (8)', '(CricketTeam (1)) -> Player (7)', and 'Team (1)'. The bottom status bar indicates 'Showing 0.111ms pre'.

retrieve particular player details

match(p:player {playerID : 33}) return p

The screenshot shows the Neo4j browser interface. A specific node representing a player is highlighted with a large red circle. The node has the label 'Player' and the ID '33'. A tooltip for this node says 'Node Details: Player'. The right side shows a 'Results Overview' panel with sections for 'Nodes (8)', '(CricketTeam (1)) -> Player (7)', and 'Team (1)'. The bottom status bar indicates 'Showing 0.111ms pre'.

Update particular player details:

```
match(p:Player{PlayerID:1}) set p.age=27 return p
```

Output:

The screenshot shows the Neo4j Browser interface. On the left, there's a sidebar with 'Database Information', 'Nodes & Relationships', and 'Relationships' sections. The main area displays a node labeled 'Player' with properties: id (1), PlayerID (1), FirstName (John), LastName (Doe), DOB (1990-01-01), Height (180), Weight (75), and Email (john.doe@example.com). A circular preview of the graph structure is visible at the bottom.

Delete particular player from the team:

```
match(p:Player{PlayerID:33}) delete p
```

The screenshot shows the Neo4j Browser interface. The sidebar indicates 'Nodes & Relationships' and 'Relationships'. The main area shows a node labeled 'Player' with id 33, which has been deleted, as indicated by the red circle with a slash over it. A message at the bottom states: 'The current session has destroyed node [id:33]'. A red arrow points from the text 'Result:' below to this message.

Result:

Thus the CRUD operations like creating, inserting, querying, finding, deleting operations on graph spaces were executed successfully.

~~Result: Thus CRUD operation like creating, inserting, querying, finding, deleting operations on graph space were executed successfully.~~

VEL TECH	
EX NO.	
PERFORMANCE (5)	
RESULT AND ANALYSIS (5)	
VIVA VOCE (5)	
RECORD (5)	
TOTAL (20)	
SIGN WITH DATE	