

Mithibai College

Department of Computer Science

M.Sc. (Data Sci and AI)

Practical 09: CRUD operations using Neo4j

Date: 18/03/2025

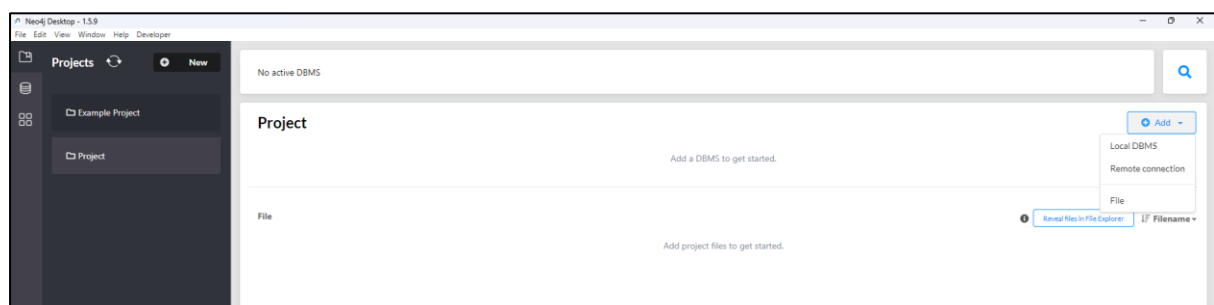
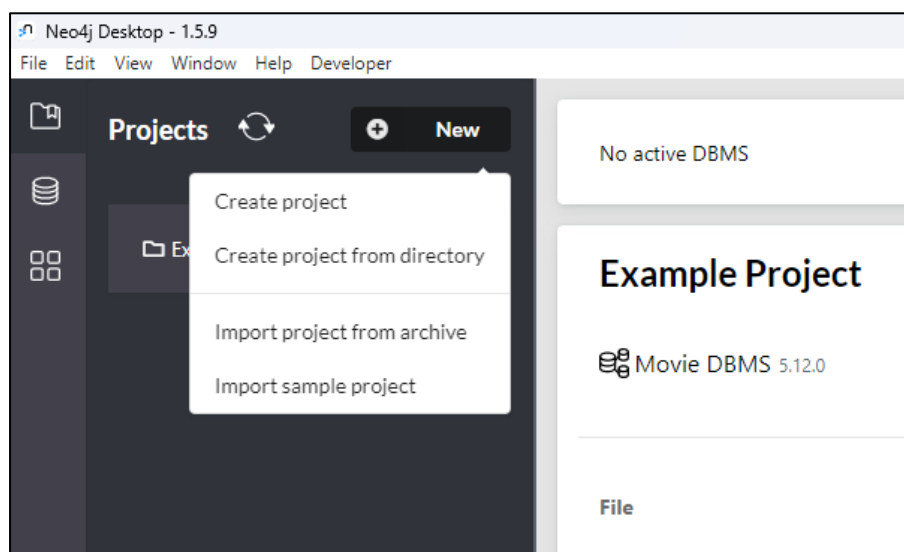
Submission Date: 21/03/2025

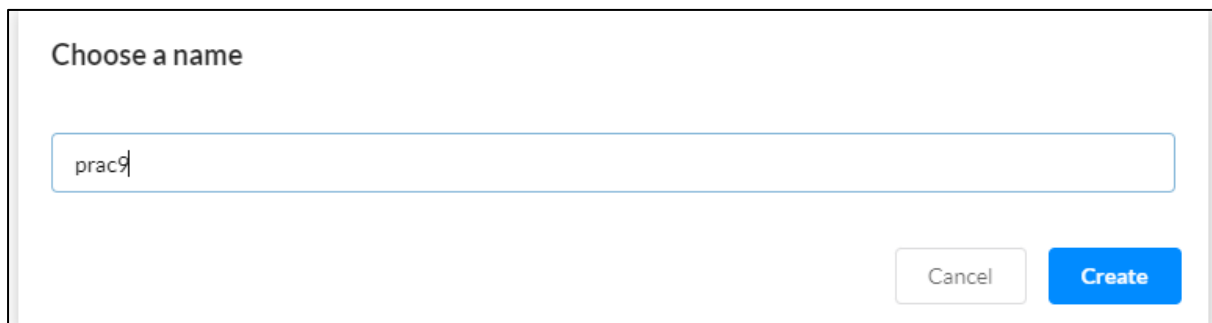
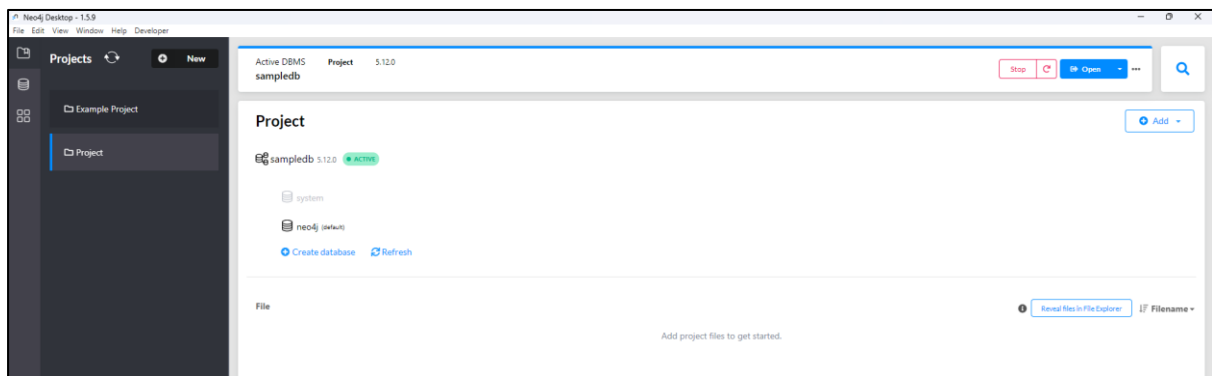
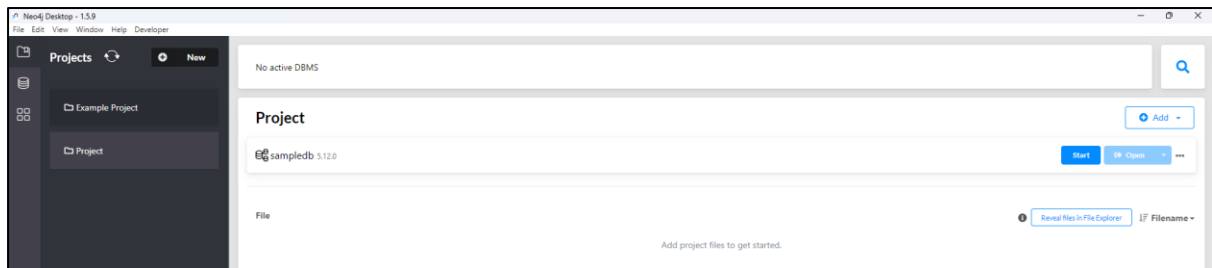
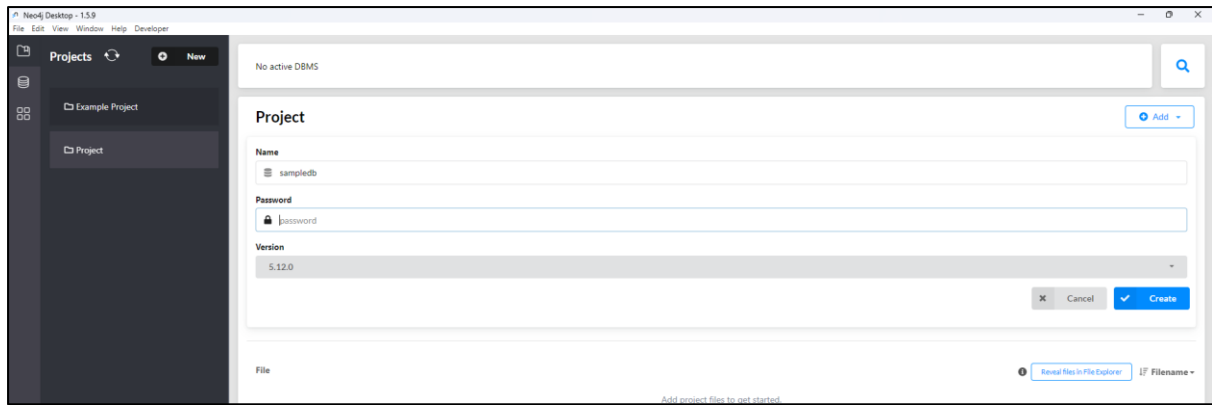
Write-up: -

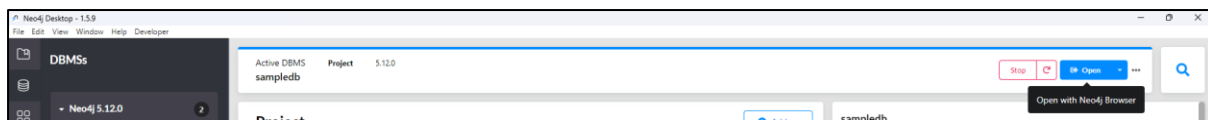
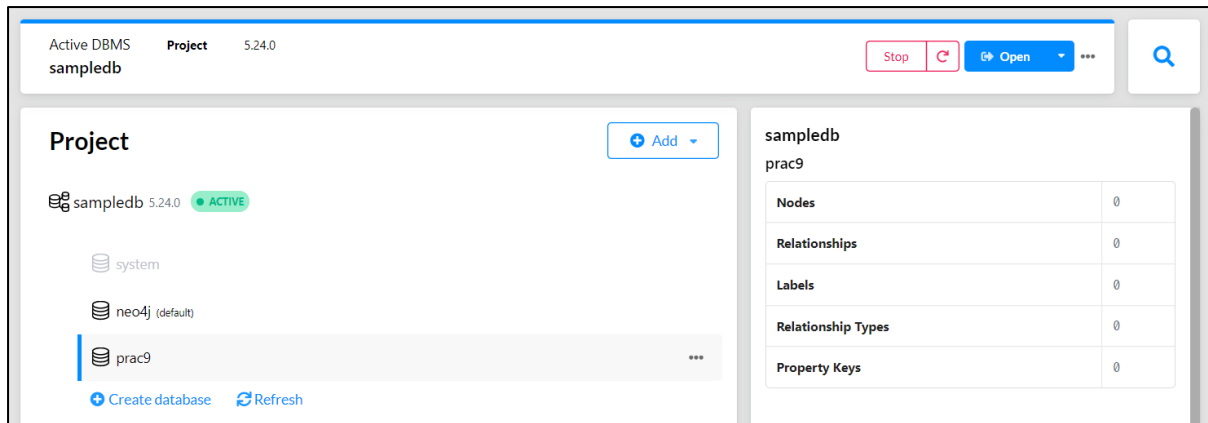
- Graph databases
- Graph databases vs Relational DBMS
- Neo4j
- Cypher query language
- Any 5 CQL constructs

Implement CRUD operations:

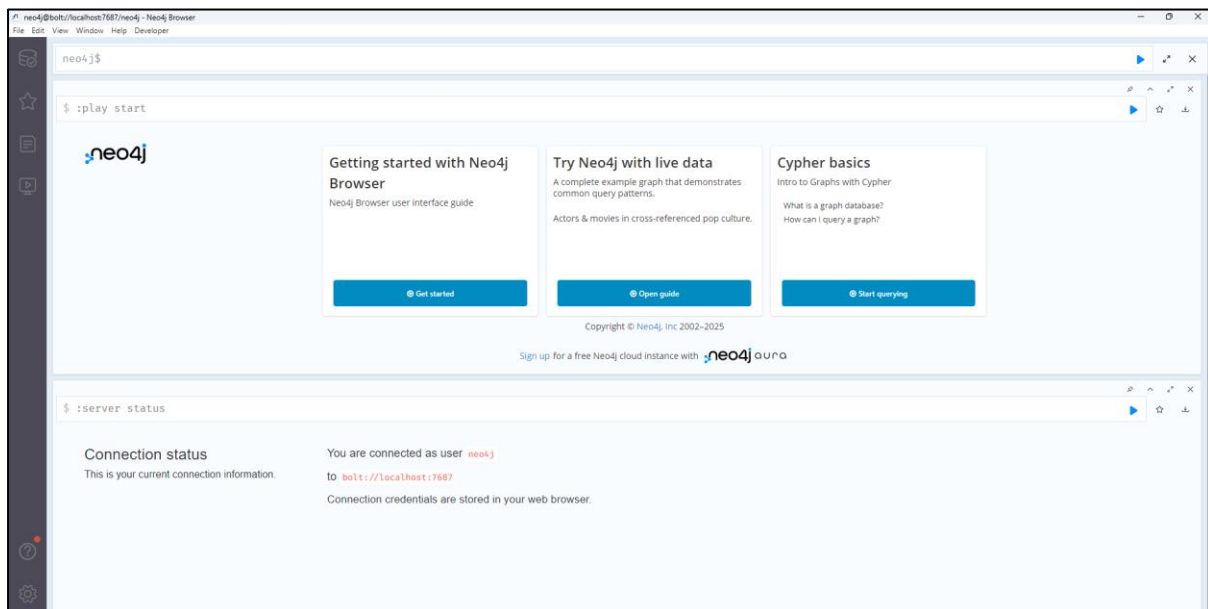
Open the Neo4j desktop App and start the Neo4j Server as shown in the following screenshot



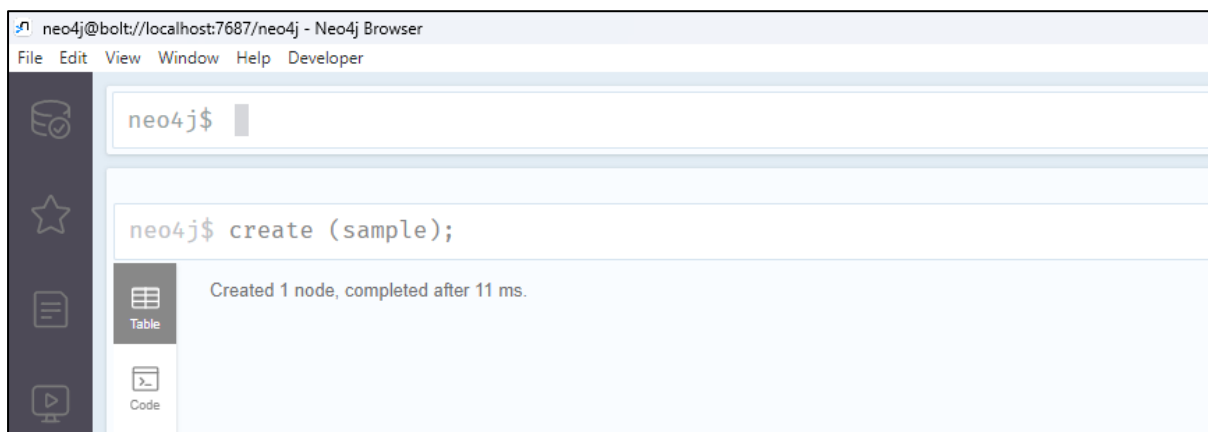




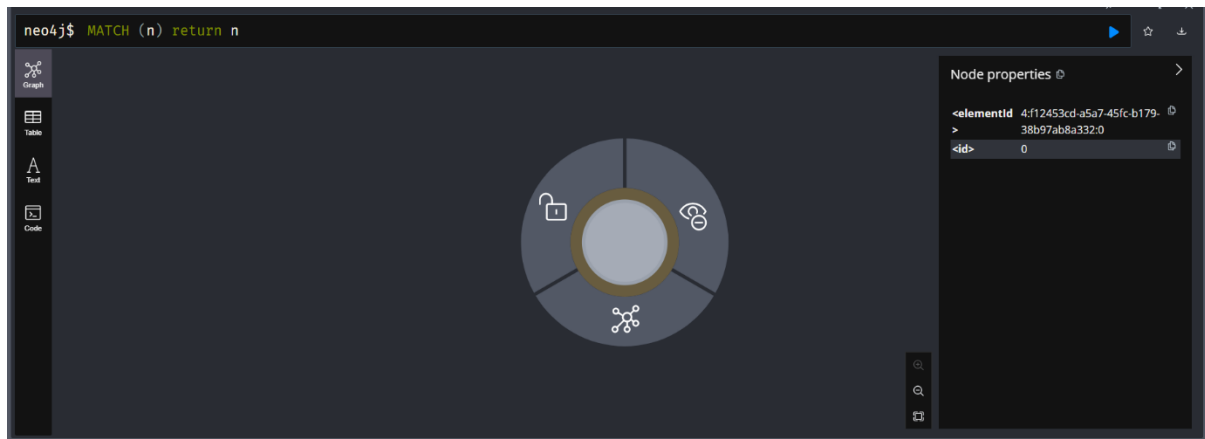
You will be getting the following interface:



1] Creating a node

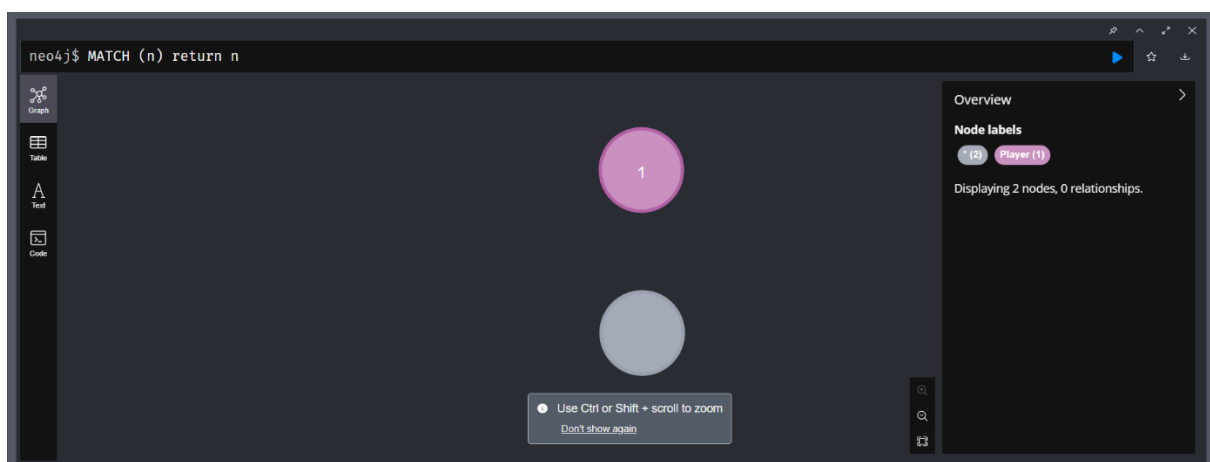
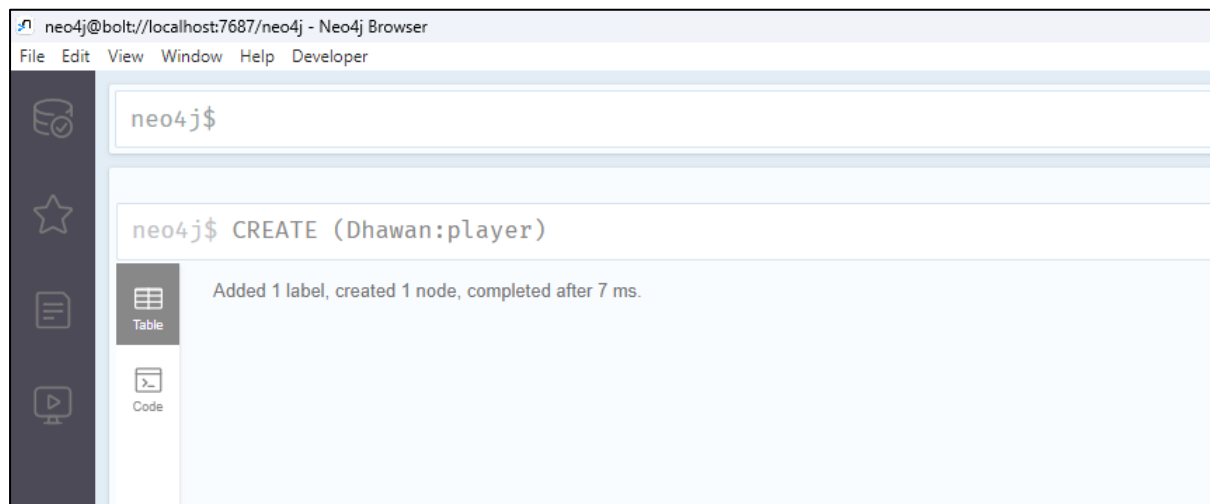


To verify the creation of the node, type and execute the following query in the dollar prompt.

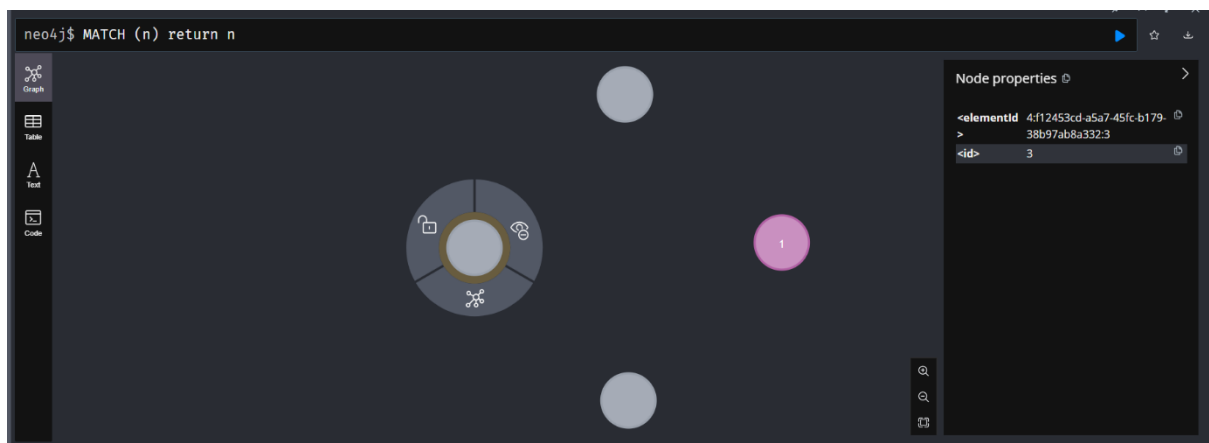
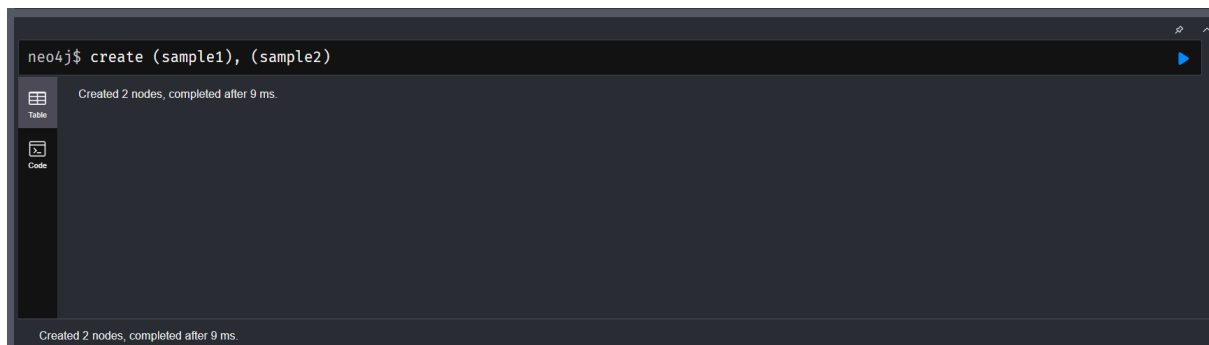


2] Creating a node with a label

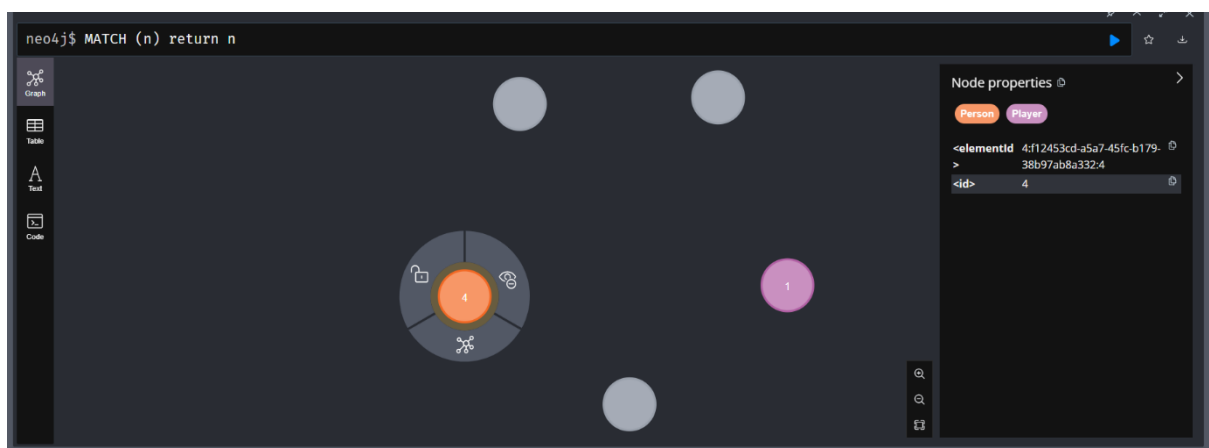
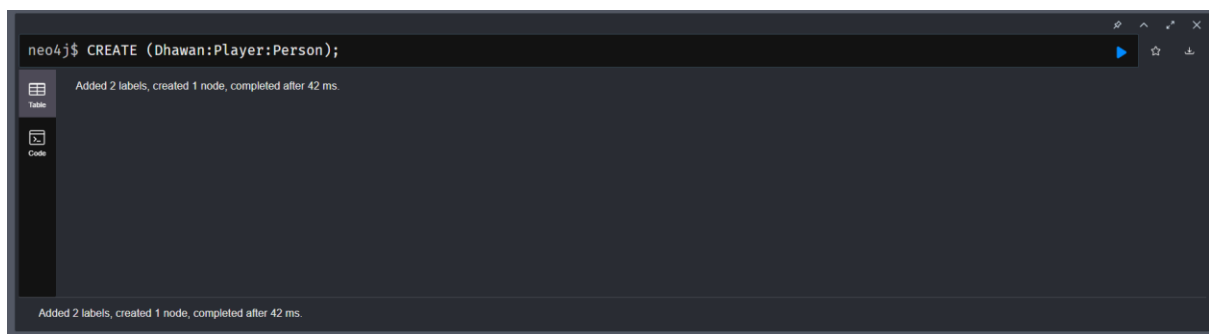
Syntax: CREATE (node:label)



3] Create multiple nodes

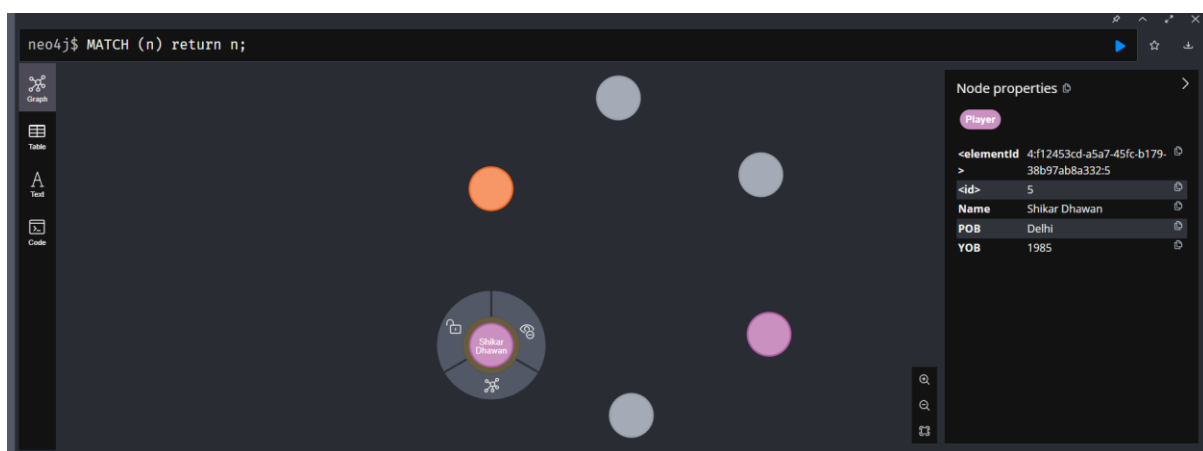


4] Creating a node with multiple labels



5] Creating a node with properties

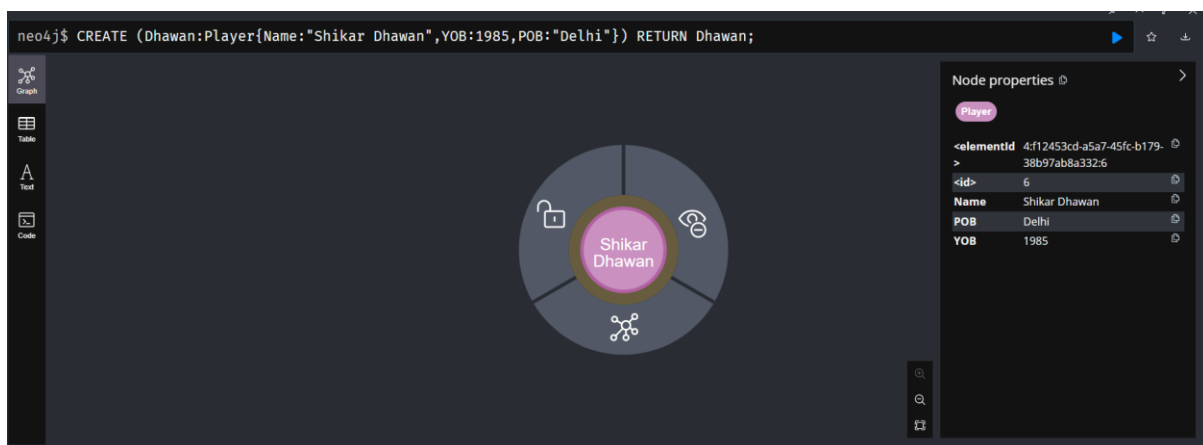
Syntax: CREATE (node:label { key1: value, key2: value, })



6] Returning the created node

Throughout the chapter, we used the **MATCH (n) RETURN n** query to view the created nodes. This query returns all the existing nodes in the database. Instead of this, we can use the **RETURN** clause with **CREATE** to view the newly created node.

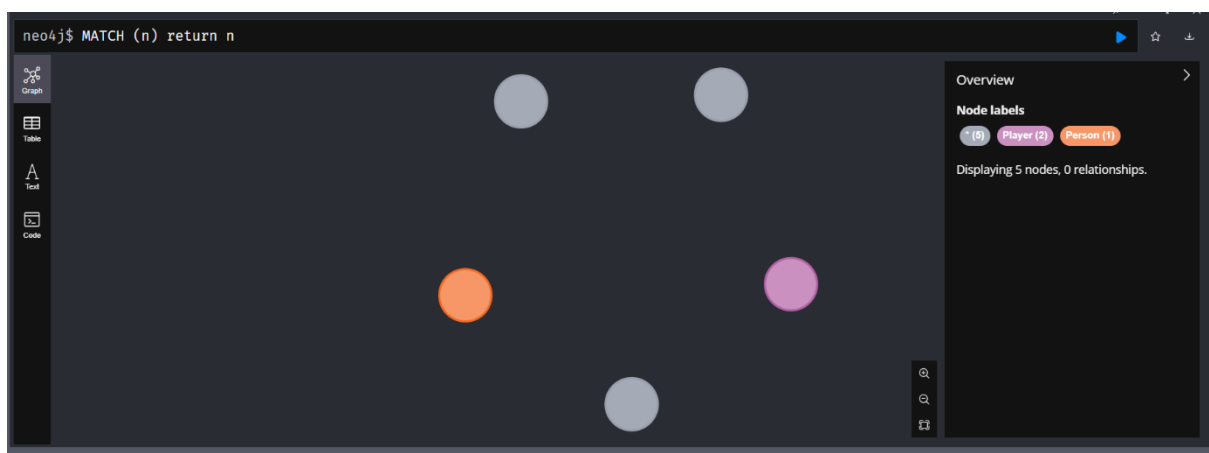
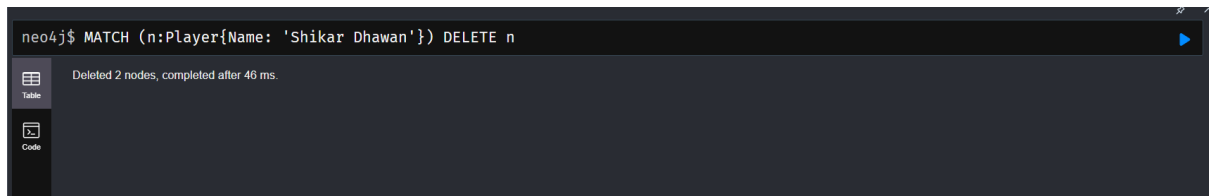
Syntax: CREATE (Node:Label{properties. . . }) RETURN Node



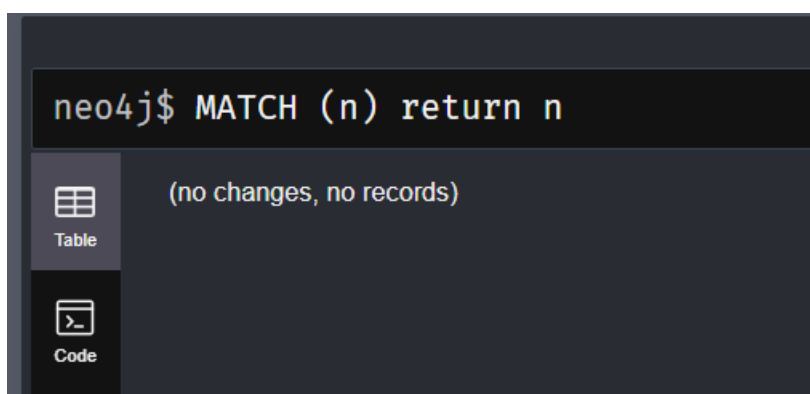
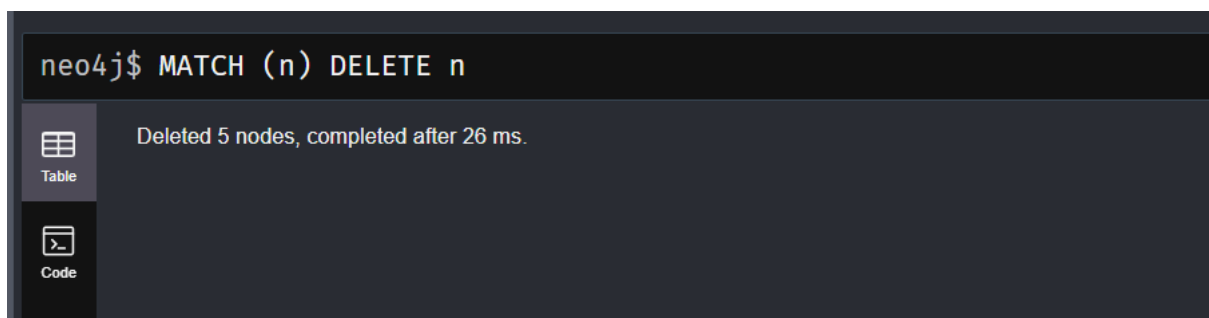
7] Deleting a node

This query is only possible to run on nodes without any relationships connected to them.

MATCH (n:Player{Name: 'Shikar Dhawan'}) DELETE n;



8] Deleting all nodes



9] Creating a relationship

Following is the syntax to create a relationship using the CREATE clause:

CREATE (node1)-[:RelationshipType]->(node2)

First of all, create two nodes Ind and Dhawan in the database, as shown below

```
1 CREATE (Dhawan:Player{name:"Shikar Dhawan", YOB:1985, POB:"Delhi"})
2 CREATE (Ind:Country{name:"India"})
```

Added 2 labels, created 2 nodes, set 4 properties, completed after 136 ms.

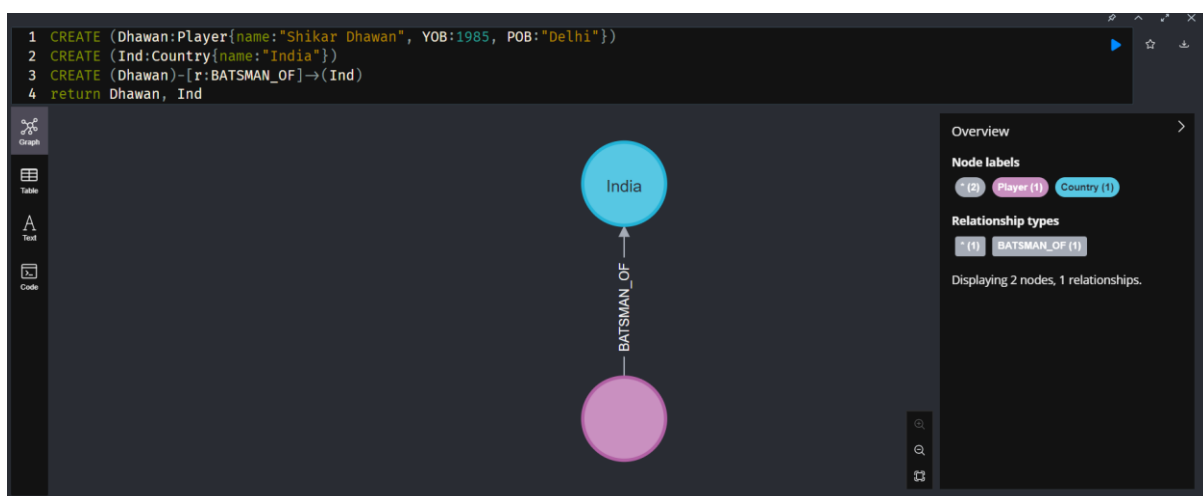
Now, create a relationship named **BATSMAN_OF** between these two nodes as

```
1 CREATE (Dhawan:Player{name:"Shikar Dhawan", YOB:1985, POB:"Delhi"})
2 CREATE (Ind:Country{name:"India"})
3 CREATE (Dhawan)-[r:BATSMAN_OF]→(Ind)
```

Finally, return both the nodes to see the created relationship.

```
1 CREATE (Dhawan:Player{name:"Shikar Dhawan", YOB:1985, POB:"Delhi"})
2 CREATE (Ind:Country{name:"India"})
3 CREATE (Dhawan)-[r:BATSMAN_OF]→(Ind)
4 return Dhawan, Ind
```

On executing, you will get the following result:



10] Creating a Relationship Between the Existing Nodes.

```
1 CREATE (Dhawan:Player{name:"Shikar Dhawan", YOB:1985, POB:"Delhi"})
2 CREATE (Ind:Country{name:"India"})
3 RETURN Dhawan, Ind
```

Overview

Node labels

- (2) Player (1) Country (1)

Displaying 2 nodes, 0 relationships.

```
1 MATCH (a:Player), (b:Country)
2 WHERE a.name="Shikar Dhawan" AND b.name="India"
3 CREATE (a)-[r:BATSMAN_OF]-(b)
4 RETURN a,b
```

Overview

Node labels

- (2) Player (1) Country (1)

Relationship types

- (1) BATSMAN_OF (1)

Displaying 2 nodes, 1 relationships.

11] Creating a Relationship with Label and Properties

```
1 CREATE (Dhawan:Player{name:"Shikar Dhawan", YOB:1985, POB:"Delhi"})
2 CREATE (Ind:Country{name:"India"})
3 RETURN Dhawan, Ind
```

Overview

Node labels

- (2) Player (1) Country (1)

Displaying 2 nodes, 0 relationships.

The screenshot shows the Neo4j Desktop interface. The top panel contains a Cypher query:

```
1 MATCH (a:Player), (b:Country)
2 WHERE a.name="Shikar Dhawan" AND b.name="India"
3 CREATE (a)-[r:BATSMAN_OF{Matches: 5, Avg:50.2}]->(b)
4 RETURN a,b
```

The middle panel displays a graph visualization with two nodes: a blue circle labeled "India" and a pink circle. A yellow arrow labeled "BATSMAN_OF" points from the pink circle to the "India" node.

The right panel shows the "Relationship properties" for the "BATSMAN_OF" relationship:

Relationship properties	
BATSMAN_OF	
<elementid>	5f12453cd-a5a7-45fc-b179-38b97ab8a332:1152921504606846977
<id>	1152921504606846977
Avg	50.2
Matches	5

12] Deleting a node that has a relationship

The screenshot shows the Neo4j Desktop interface. The top panel contains a Cypher query:

```
neo4j$ MATCH (n)-[r:BATSMAN_OF]->() DELETE r
```

The middle panel displays the result: "Deleted 1 relationship, completed after 43 ms."

The left sidebar shows the "Table" and "Code" views.

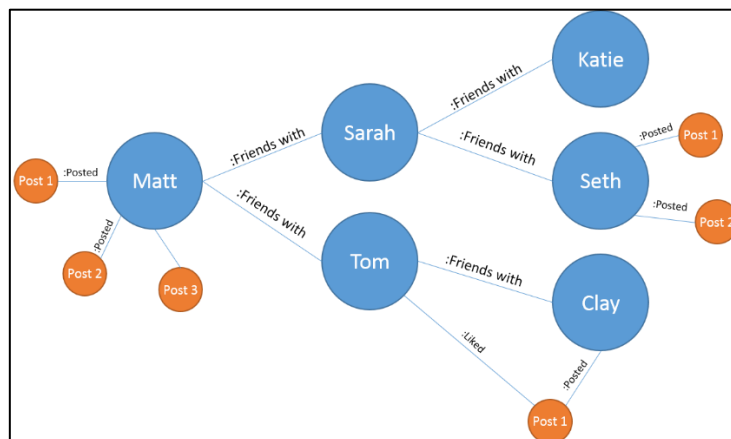
The screenshot shows the Neo4j Desktop interface. The top panel contains a Cypher query:

```
neo4j$ MATCH (n:Player{name:"Shikar Dhawan"}) DELETE n
```

The middle panel displays the result: "Deleted 1 node, completed after 15 ms."

The left sidebar shows the "Table" and "Code" views.

EVEN NUMBERS - Q21 FROM UNIT 3 ASSIGN



a. create the above database.

CREATE

```

(matt:Person{name:"Matt"}),
(sarah:Person{name:"Sarah"}),
(tom:Person{name:"Tom"}),
(katie:Person{name:"Katie"}),
(seth:Person{name:"Seth"}),
(clay:Person{name:"Clay"}),
(post1_matt:Post{name:"Post 1",author:"Matt"}),
(post2_matt:Post{name:"Post 2",author:"Matt"}),
(post3_matt:Post{name:"Post 3",author:"Matt"}),
(post1_seth:Post{name:"Post 1",author:"Seth"}),
(post2_seth:Post{name:"Post 2",author:"Seth"}),
(post1_clay:Post{name:"Post 1",author:"Clay"});

```

```

neo4j$ CREATE (matt:Person{name:"Matt"}), (sarah:Person{name:"Sarah"}), (tom:Person{name:"Tom"}), (katie:Person
Added 12 labels, created 12 nodes, set 18 properties, completed after 58 ms.

```

```

MATCH (a:Person), (b:Person)
WHERE a.name="Matt" AND b.name="Sarah"
CREATE (a)-[r:FRIENDS_WITH]->(b)
return a,b

```

```

MATCH (a:Person), (b:Person)
WHERE a.name="Matt" AND b.name="Tom"
CREATE (a)-[r:FRIENDS_WITH]->(b)
return a,b

```

```

MATCH (a:Person), (b:Person)
WHERE a.name="Sarah" AND b.name="Katie"
CREATE (a)-[r:FRIENDS_WITH]->(b)
return a,b

```

```

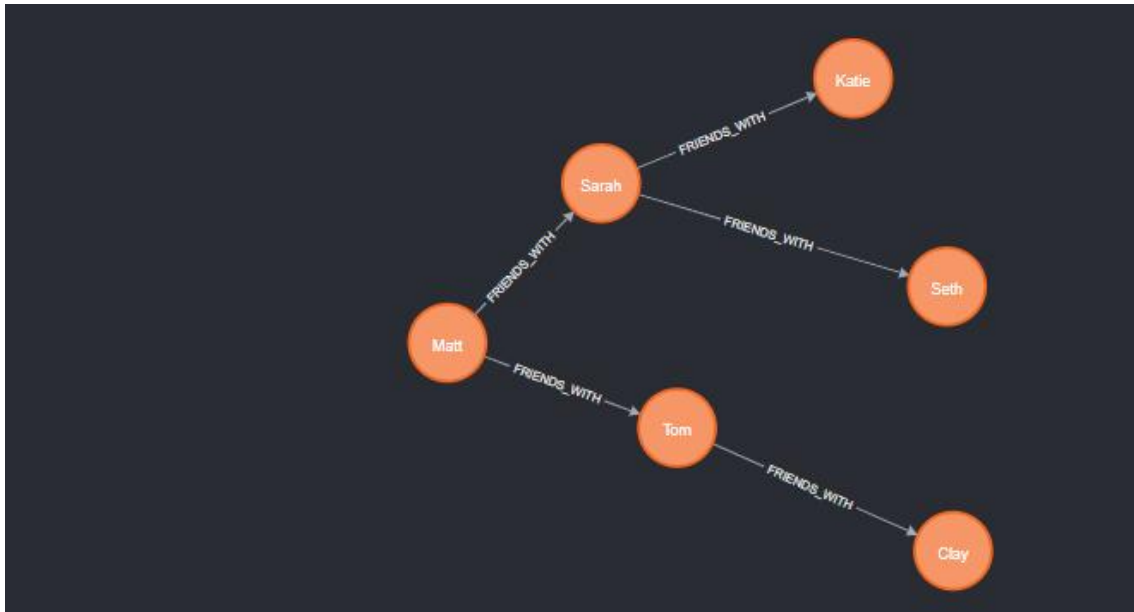
MATCH (a:Person), (b:Person)
WHERE a.name="Sarah" AND b.name="Seth"
CREATE (a)-[r:FRIENDS_WITH]->(b)
return a,b

```

```

MATCH (a:Person), (b:Person)
WHERE a.name="Tom" AND b.name="Clay"
CREATE (a)-[r:FRIENDS_WITH]->(b)
return a,b

```



```

MATCH (a:Person), (b:Post)
WHERE a.name="Matt" AND b.author="Matt"
CREATE (a)-[r:POSTED]->(b)
return a,b

```

```

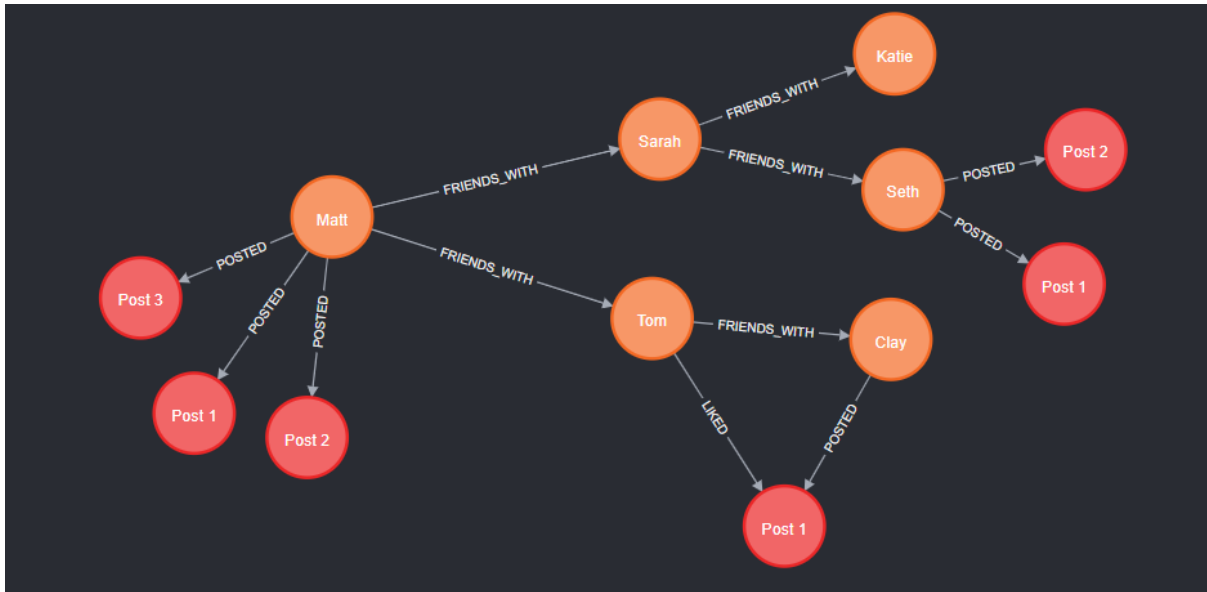
MATCH (a:Person), (b:Post)
WHERE a.name="Seth" AND b.author="Seth"
CREATE (a)-[r:POSTED]->(b)
return a,b

```

```

MATCH (a:Person), (b:Post), (c:Person)
WHERE a.name="Clay" AND b.author="Clay" AND c.name="Tom"
CREATE (a)-[:POSTED]->(b)<-[:LIKED]-(c)
return a,b,c

```



b. know that if Katie create any posts?

```

1 MATCH (katie:Person{name:"Katie"})-[:POSTED]->(post:POST)
2 RETURN katie, post

```

(no changes, no records)

Table

Warn

Code

c. Know how many friends does Seth have?

```

neo4j$ MATCH (seth:Person{name:"Seth"})-[:FRIENDS_WITH]-(friend:Person) RETURN count(friend) as number_of_friends

```

number_of_friends
1

Table

Text

Code

- d. Know that there is a path of friendship between Matt and Clay.

