

**Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
(Deemed to be University Estd. u/s 3 of UGC Act, 1956)**



**School of Computing
B.Tech. – Information Technology**

VTR UGE2021- (CBCS)



Academic Year: 2025–2026

SUMMER SEMESTER - SS2526

Course Code : 10211IT201

Course Name : Database System Concepts

Slot No : S12L5

DBMS TASK - 9 REPORT

Submitted by:

VTUNO	REGISTER NUMBER	STUDENT NAME
VTU28684	24UEIT0058	YEKAMBARAM NANDHA KISHORE

ABSTRACT

The objective of this task is to perform **CRUD operations** using **graph databases** such as Neo4j.

Graph databases use **nodes** and **relationships** to represent and query complex data relationships efficiently.

AIM:

To perform CRUD operations like creating, inserting, querying, finding, deleting operations on graph spaces.

❖ Create Node with Properties

Properties are the key-value pairs using which a node stores data. You can create a node with properties using the CREATE clause. You need to specify these properties separated by commas within the flower braces "{}".

Syntax

Following is the syntax to create a node with properties.

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

❖ Returning the Created Node

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

```
MATCH (n) RETURN n
```

❖ Creating Relationships

We can create a relationship using the CREATE clause. We will specify relationship within the square braces “[]” depending on the direction of the relationship it is placed between hyphen “ - ” and arrow “ → ” as shown in the following syntax.

Syntax

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

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

❖ Creating a Relationship Between the Existing Nodes

You can also create a relationship between the existing nodes using the MATCH clause.

Syntax

Following is the syntax to create a relationship using the MATCH clause.

```
MATCH (a:LabeofNode1), (b:LabeofNode2)
```

```
WHERE a.name = "nameofnode1" AND b.name = " nameofnode2"
```

```
CREATE (a)-[: Relation]->(b)
```

```
RETURN a,b
```

❖ Deleting a Particular Node

To delete a particular node, you need to specify the details of the node in the place of “n” in the above query.

Syntax

Following is the syntax to delete a particular node from Neo4j using the DELETE clause.

```
MATCH (node:label {properties..... })
```

```
DETACH DELETE node
```

Create a graph database for student course registration, create student and dept node and insert values of properties.

```
create(n:student{Sid: "VTU14500",  
Sname:"John",  
deptname:"CSE" }  
)
```

OUTPUT

Added 1 label, created 1 node, set 3 properties, completed after 232 ms.

```
Create(n:student {Sid: "VTU14501",  
Sname:"Dharsana", deptname:"EEE"})
```

OUTPUT

Added 1 label, created 1 node, set 3 properties, completed after 16 ms.

```
Create(n:student { Sid: "VTU14502",  
Sname:"vijay",  
deptname:"CSE"  
})
```

OUTPUT

Added 1 label, created 1 node, set 3 properties, completed after 12 ms.

```
Create(n:dept{deptname:"cse",deptid:"d001"})
```

OUTPUT:

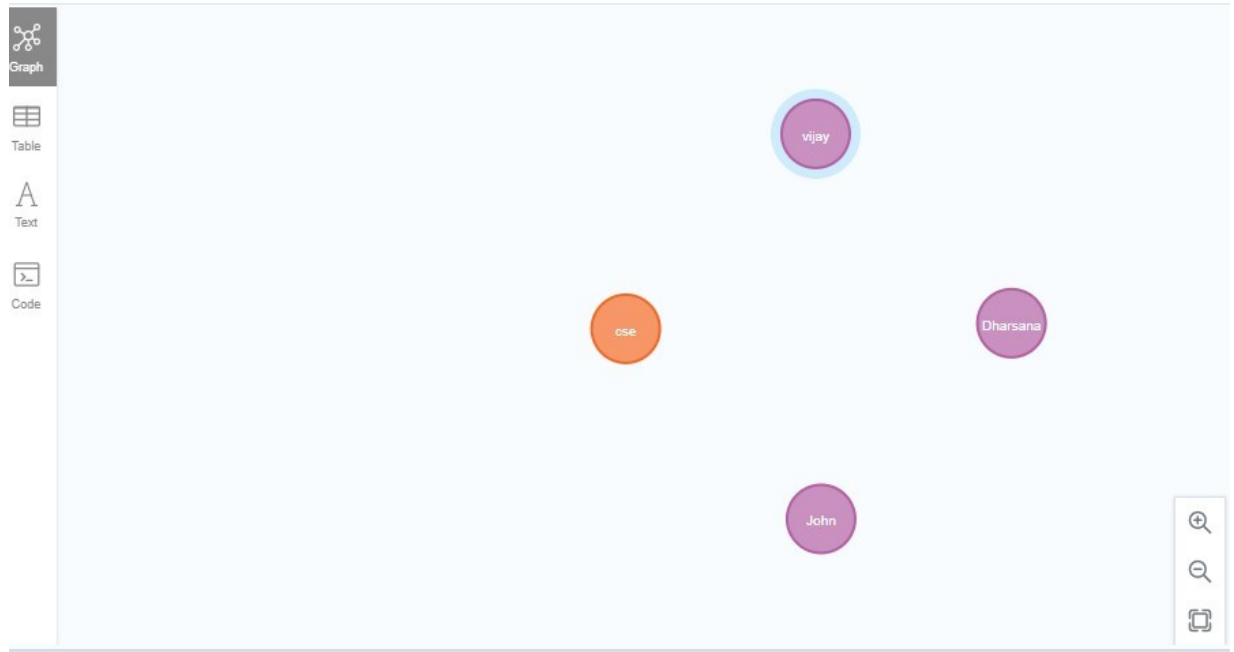
Added 1 label, created 1 node, set 2 properties, completed after 72 ms.

Select all the nodes in your database using match command

✧ **match(n) return(n)**

OUTPUT

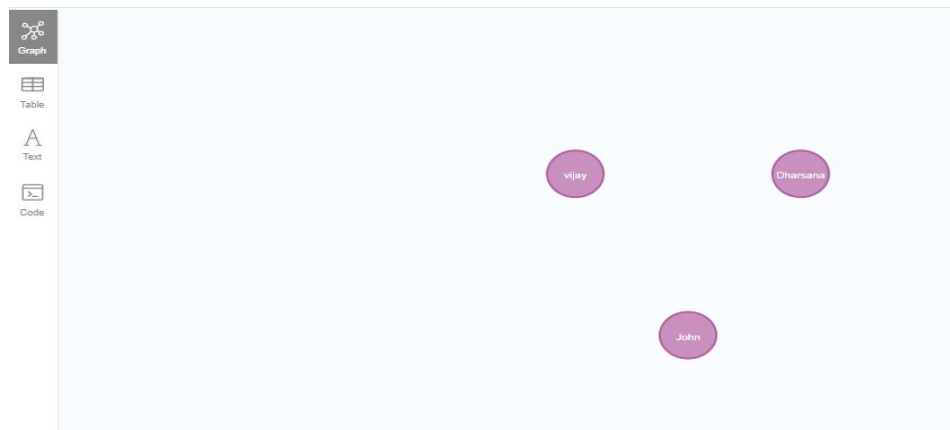
```
neo4j$ match(n) return(n)
```



✧ **match(n:student) return(n)**

OUTPUT:

```
neo4j$ match(n:student) return(n)
```



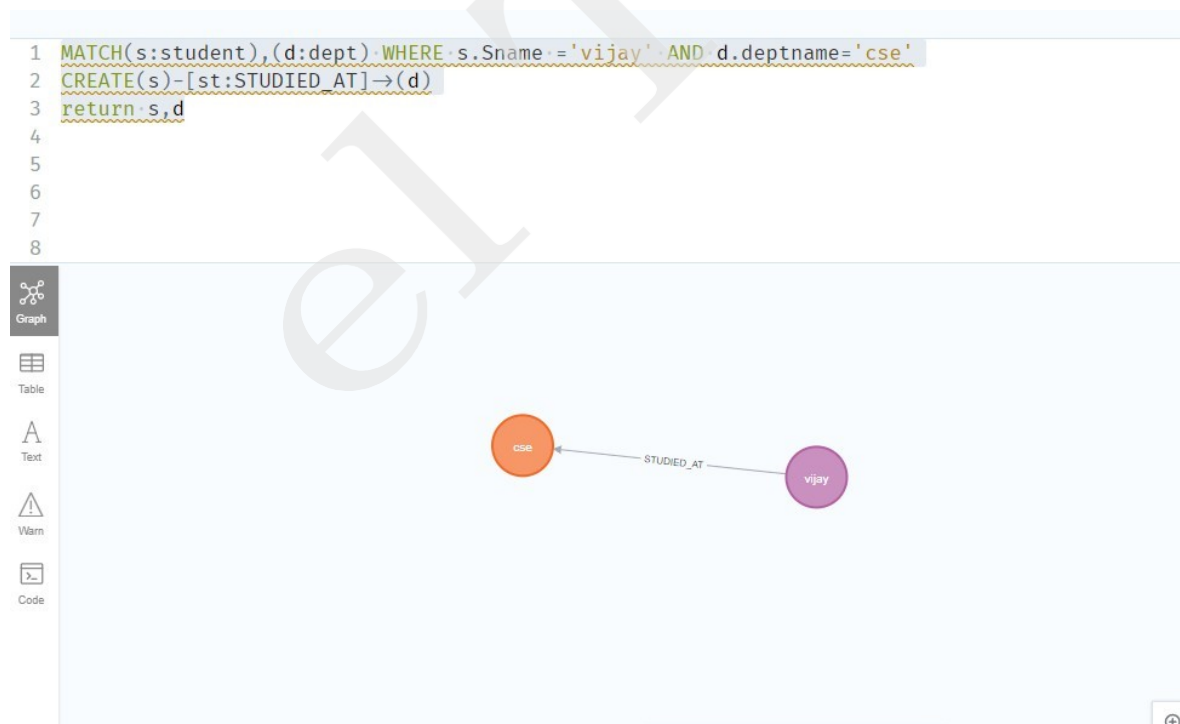
a) **Create relationship between student and cse .**

```
MATCH(s:student),(d:dept) WHERE s.Sname ='vijay' AND d.deptname='cse'
```

```
CREATE(s)-[st:STUDIED_AT]-(d)
```

```
return s,d
```

OUTPUT:



```
MATCH(s:student),(d:dept) WHERE s.Sname ='John' AND d.deptname='  
cse'
```

```
CREATE(s)-[st:STUDIED_AT]->(d)
```

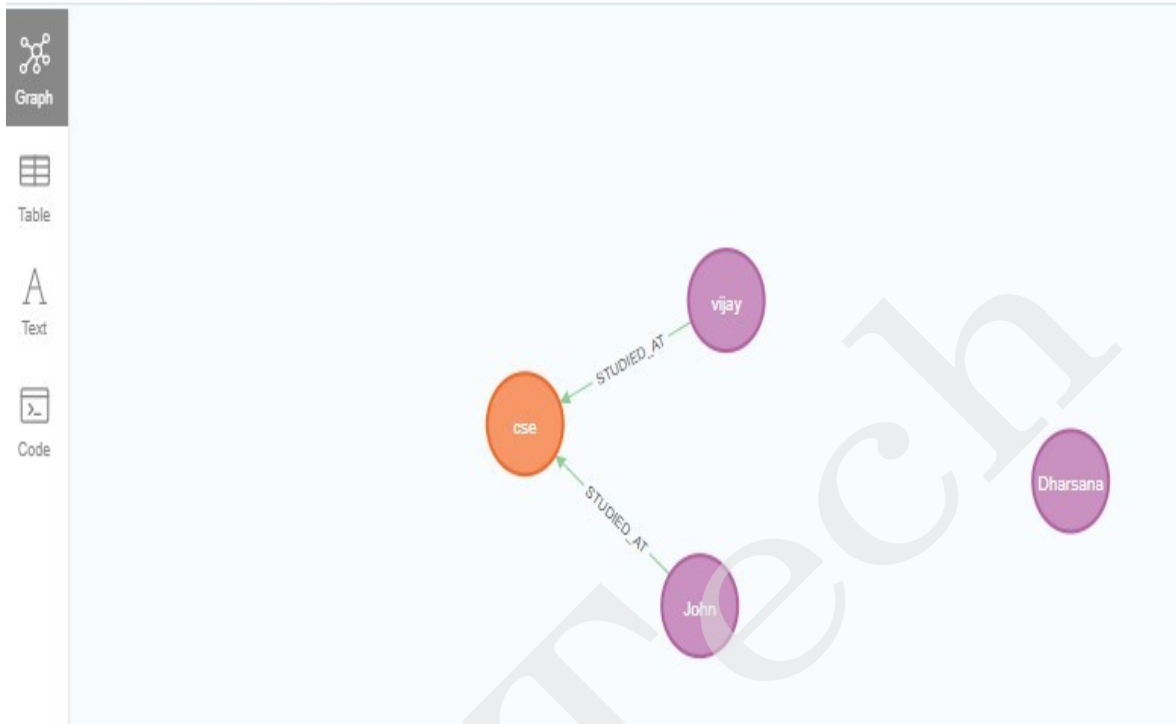
```
return s,d
```

OUTPUT:



```
match(n) return(n)
```

```
neo4j$ match(n) return(n)
```



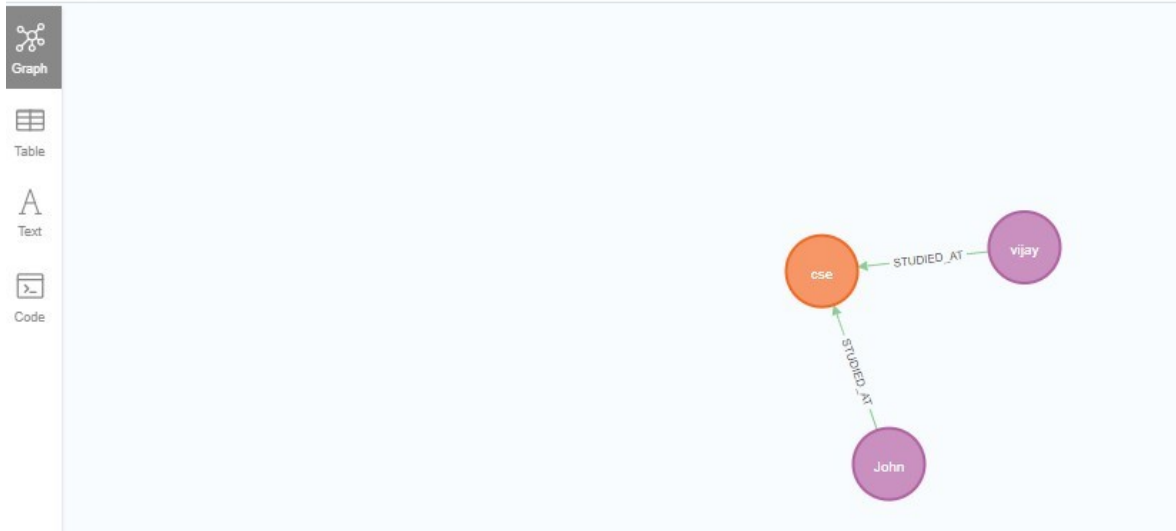
b) Delete a node from student

```
match(n:student{Sname:'Dharsana'}) DELETE(n)
```

OUTPUT:

Deleted 1 node, completed after 10834 ms.

```
neo4j$ match(n)return(n)
```



Result

The implementation of CRUD operations like creating, inserting, finding and removing operations using GraphDB is successfully executed.