

DBMS LAB

Week 2

Name : Abhishek Aditya BS

SRN : PES1UG19CS019

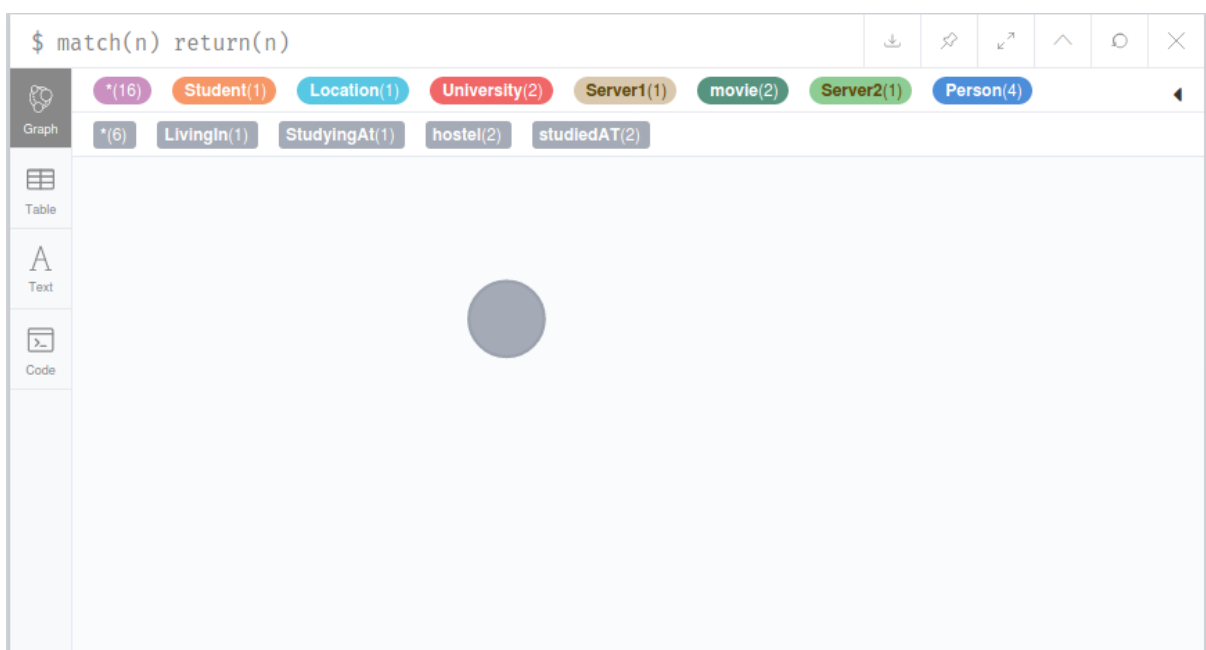
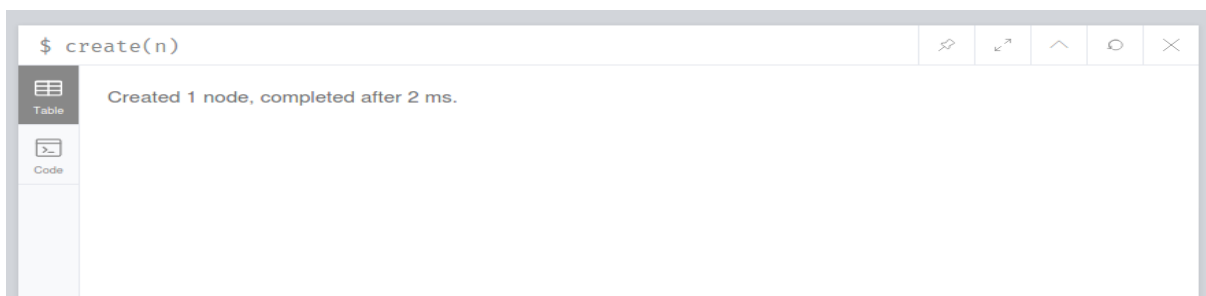
Section : A

Semester : 5th

1. Create node and relationships

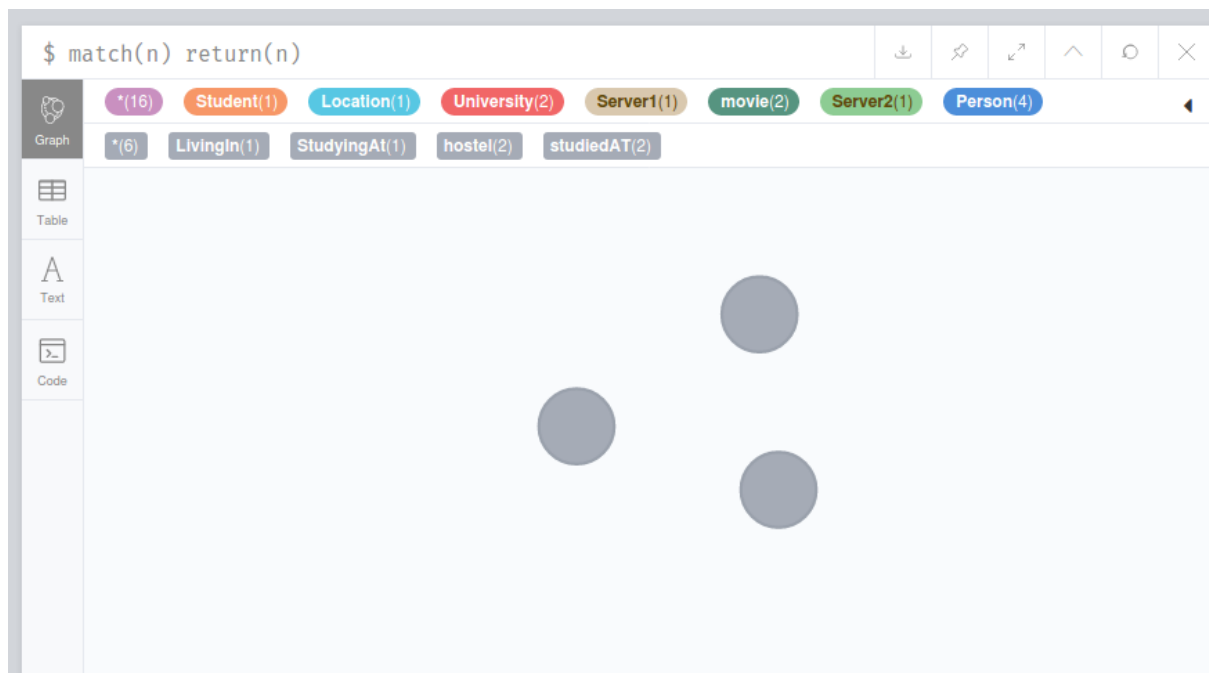
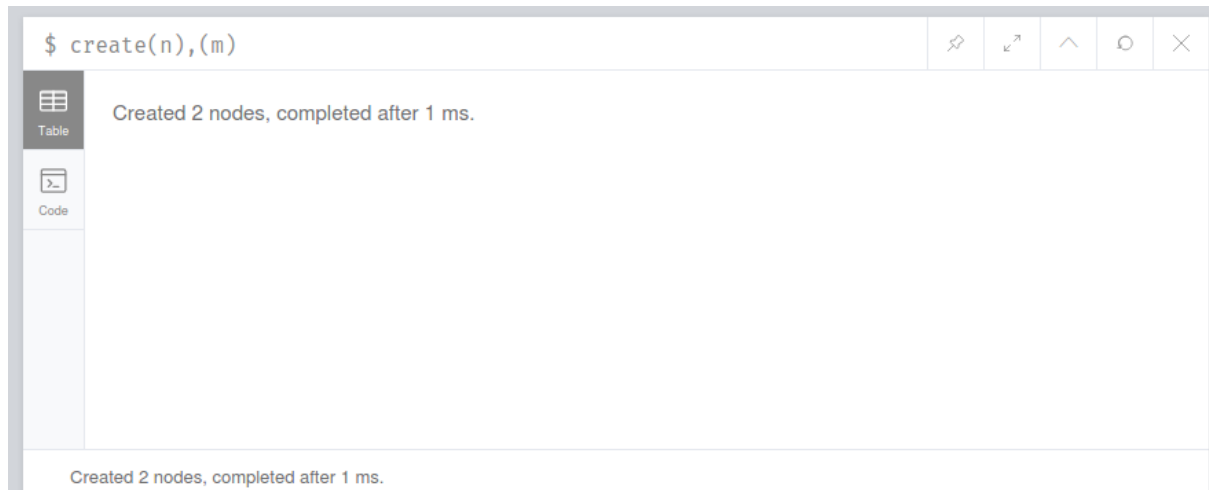
1.1 Create a single node :

Syntax: create (n)



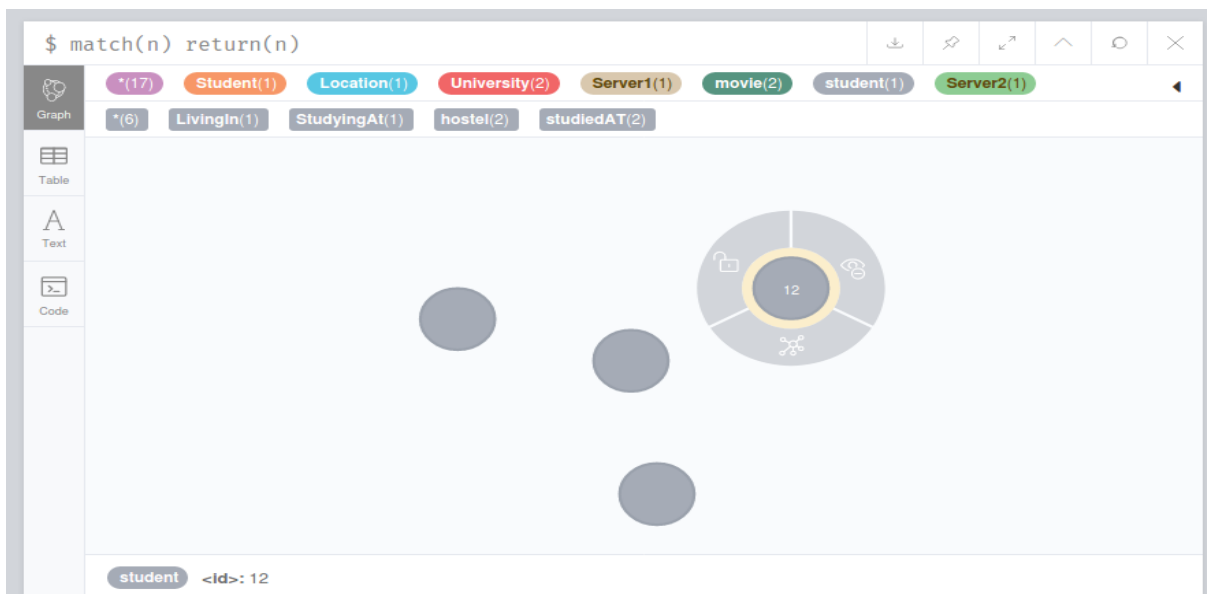
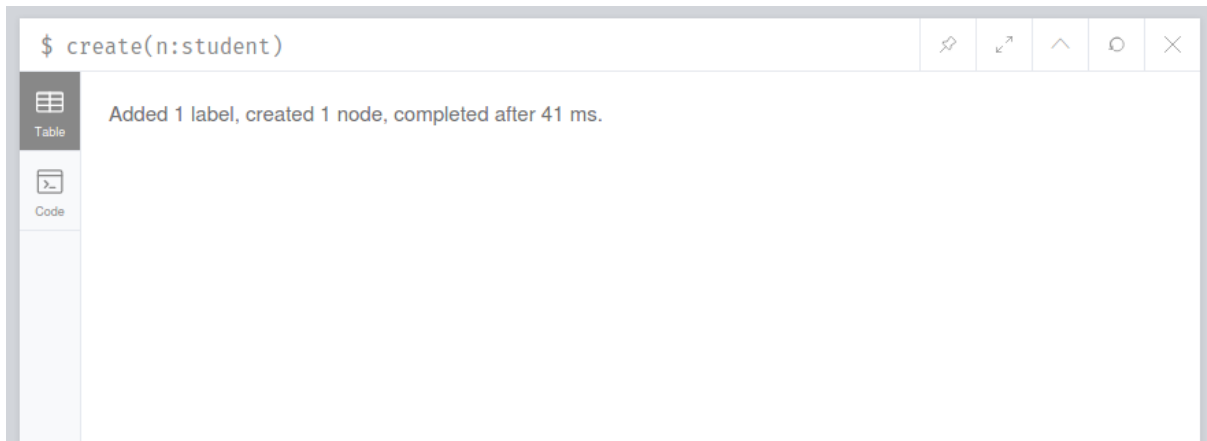
1.2 Create multiple nodes

Syntax: create(n),(m)



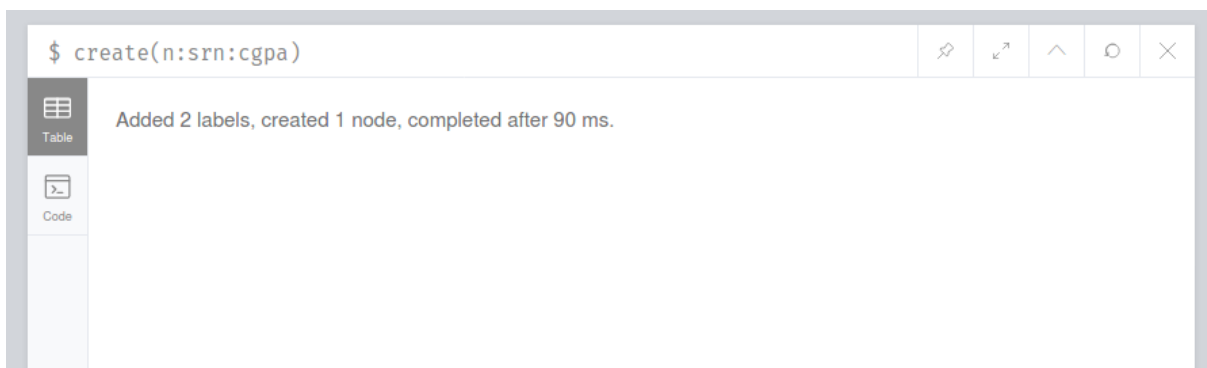
1.3 Create a node with a label

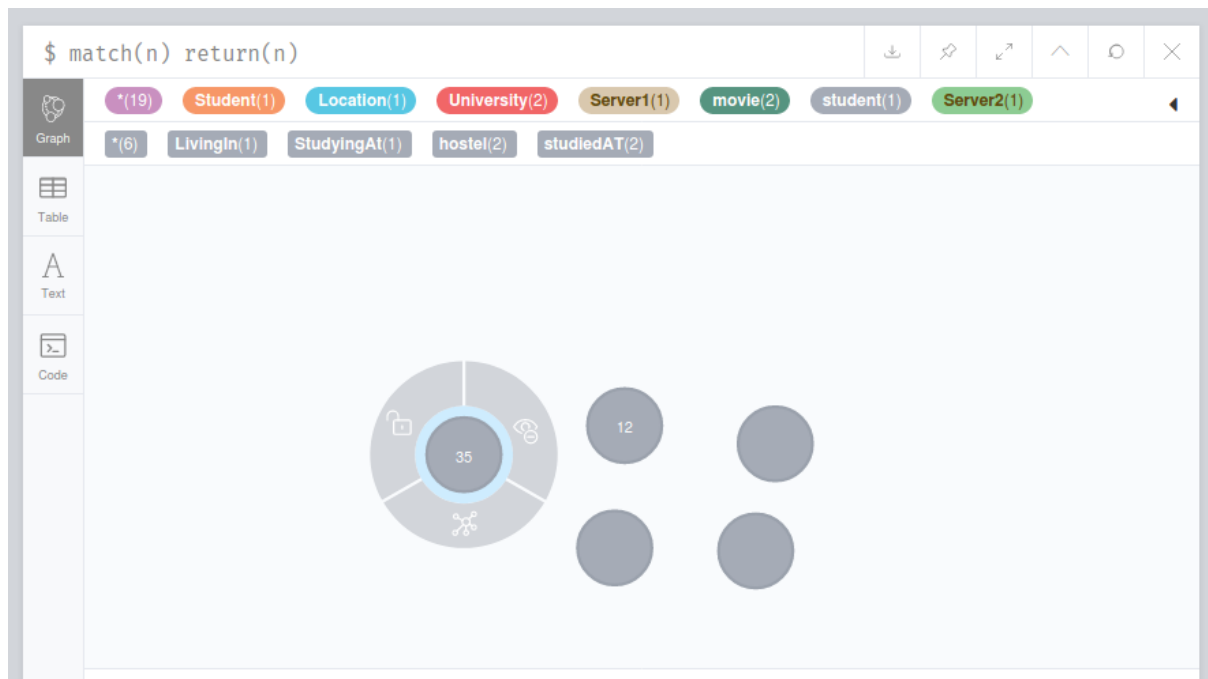
Syntax: create(n:label name)



1.4 Create a node with multiple labels

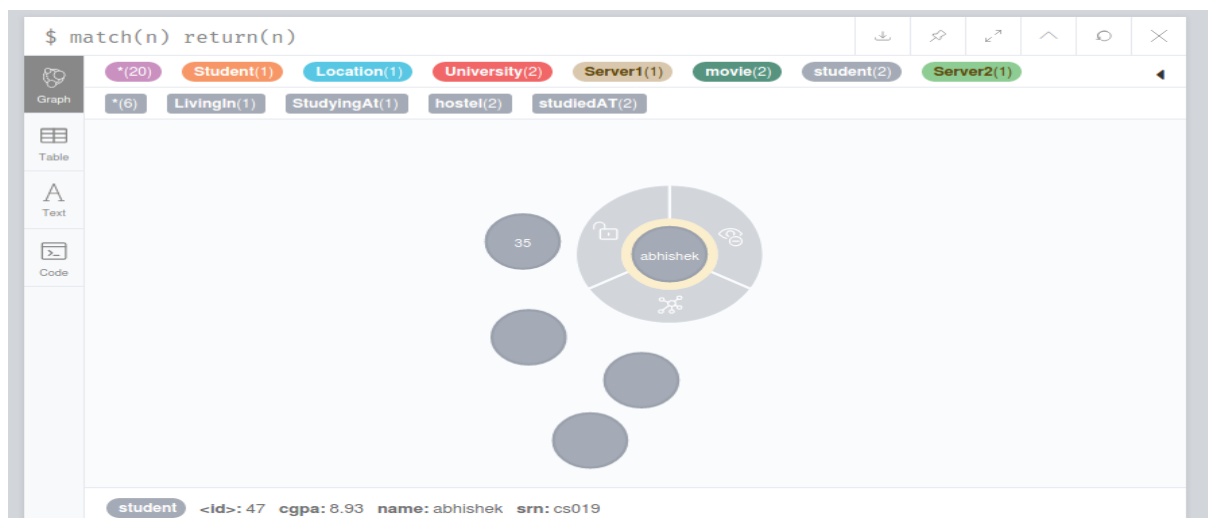
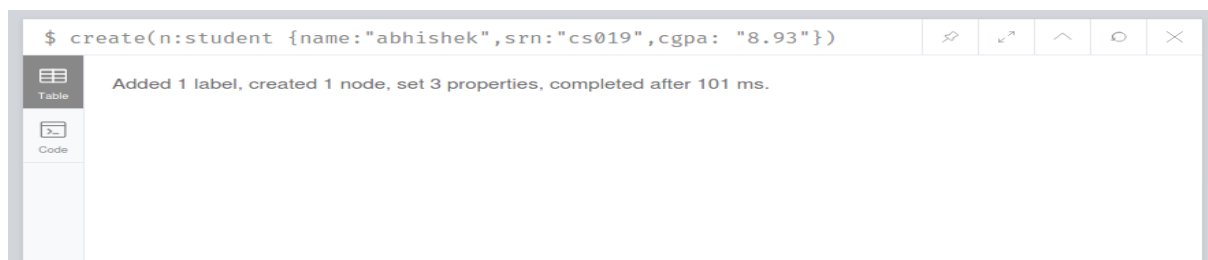
Syntax: CREATE (n:label1:label2)





1.5 Create node and add labels and properties

Syntax: Create(n:lablename {properties and values})



1.6 Create nodes with parameters as properties

Syntax: Define the property with parameter name. Add the parameter using the create clause
CREATE (n:Person \$props) RETURN n

```
$ :params {"props":{"name": "Vishal",srn:"cs571",cgpa:"9.0"}}
```

```
{  
  "props": {  
    "name": "Vishal",  
    "srn": "cs571",  
    "cgpa": "9.0"  
  }  
}
```

See [:help param](#) for usage of the :param command.

Successfully set your parameters.

```
$ create(n:student $props) return(n)
```

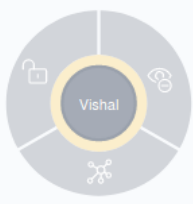
*(1) student(1)

Graph

Table

Text

Code



student <id>: 36 cgpa: 9.0 name: Vishal srn: cs571

2. Create Relationships between the nodes

Syntax:

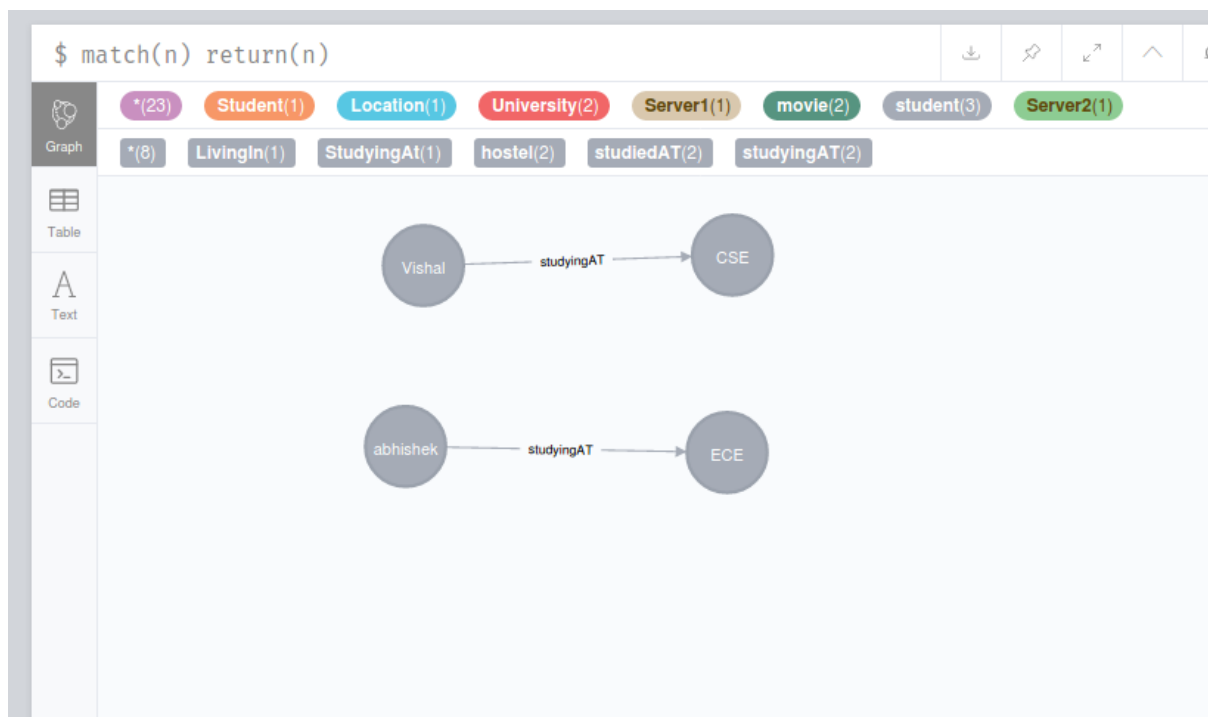
Match (node1) ,(node2)

Where condition

Create (node1) [relation type] ->(node2)

The screenshot shows a Cypher query editor with the following query: `$ match(s:student),(b:Branch) where s.name="Vishal" and b.name="CSE..."`. The interface includes a sidebar with 'Table' and 'Code' tabs. The main area displays the message 'Created 2 relationships, completed after 43 ms.' at the top and bottom.

The screenshot shows a Cypher query editor with the following query: `$ match(s:student),(b:Branch) where s.name="abhishek" and b.name="ECE" create(s)-[stu:studyingAT] -> (b)`. The interface includes a sidebar with 'Table' and 'Code' tabs. The main area displays the message 'Created 1 relationship, completed after 7 ms.'



```
$ match(s:student),(c:student) where s.name="Vishal" and c.name="abhishek" create(s)-[stu:friendOf] -> (c)
```

Created 2 relationships, completed after 5 ms.

Table

Code

3. Read nodes and attributes (Node finding)

3.1 Get all nodes

Syntax: Match(n) return(n)

\$ match(n) return(n)

Graph

Table

Text

Code

*(22) Student(1) Location(1) University(2) Server1(1) movie(2) student(3) Server2(1)

*(10) LivingIn(1) StudyingAt(1) hostel(2) studiedAT(2) friendOf(2) studyingAT(2)

```
graph TD
    Vishal((Vishal)) -- friendOf --> abhishek((abhishek))
    abhishek -- studyingAT --> ECE((ECE))
    Vishal -- studyingAT --> CSE((CSE))
    abhishek -- friendOf --> abhishek
```

3.2 Get all nodes with a label

Eg: MATCH (movie:Movie) RETURN movie.title

\$ match(n:student) return n.name

Table

Text

Code

n.name
null
"Vishal"
"abhishek"

3.3 Related nodes

Eg: MATCH (director {name: 'Oliver Stone'})--(movie)
RETURN movie.title

```
$ match(student {name: 'Vishal'})--(Branch) return Branch.year
```

Table	Branch.year
Text	null
Code	"3"

4 Update or set a value

Syntax: MATCH (n:Node)

Set n. propertyvalue = 'newvalue'

```
$ match(n:student {name:"abhishek"}) set n.cgpa=10
```

Table	Set 1 property, completed after 3 ms.
Code	

```
$ match(student {name: 'abhishek'}) return student.cgpa
```

Table	student.cgpa
Text	10
Code	

Started streaming 1 records after 1 ms and completed after 1 ms.

5. Delete operation

5.1 Delete the relationship

```
$ match(n:student {name:'Vishal'})-[r:studyingAT]→() delete r
```

Table

Code

Deleted 2 relationships, completed after 3 ms.

Deleted 2 relationships, completed after 3 ms.

```
$ match(n:student {name:'abhishek'})-[r:studyingAT]→() delete r
```

Table

Code

Deleted 1 relationship, completed after 3 ms.

```
$ match(n:student {name:'abhishek'})-[r:friendOf]→() delete r
```

Table

Code

Deleted 1 relationship, completed after 5 ms.

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

Graph

Table

Text

Code

*(23)

Student(1)

Location(1)

University(2)

Server1(1)

*(7)

LivingIn(1)

StudyingAt(1)

hostel(2)

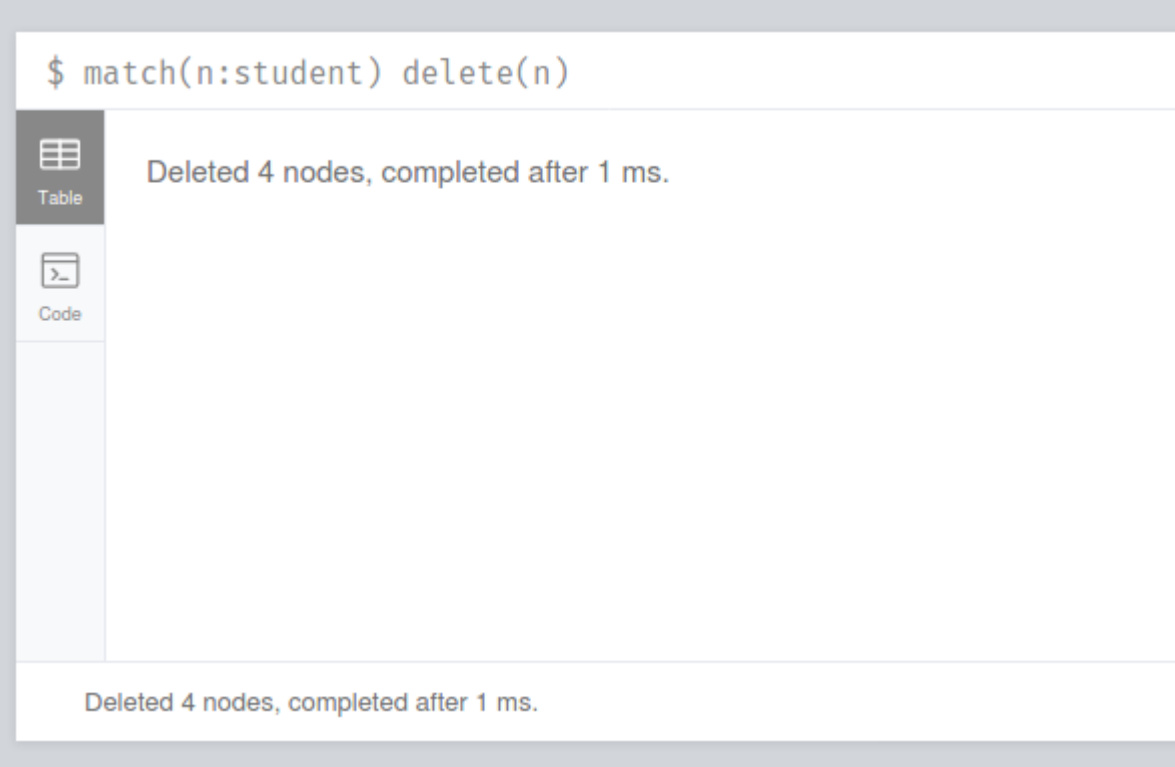
studiedAT(2)

friendOf(1)

```
graph LR; Vishal((Vishal)) -- friendOf --> abhishek((abhishek))
```

5.2 Delete the node

Syntax: Match(n) delete (n)



The screenshot shows a Neo4j Cypher query interface. At the top, a query is entered: `$ match(n:student) delete(n)`. Below the query, there is a sidebar with two tabs: "Table" (selected) and "Code". The main area displays the result: "Deleted 4 nodes, completed after 1 ms." The bottom status bar also shows: "Deleted 4 nodes, completed after 1 ms."

```
$ match(n:student) delete(n)
```

Deleted 4 nodes, completed after 1 ms.

Deleted 4 nodes, completed after 1 ms.