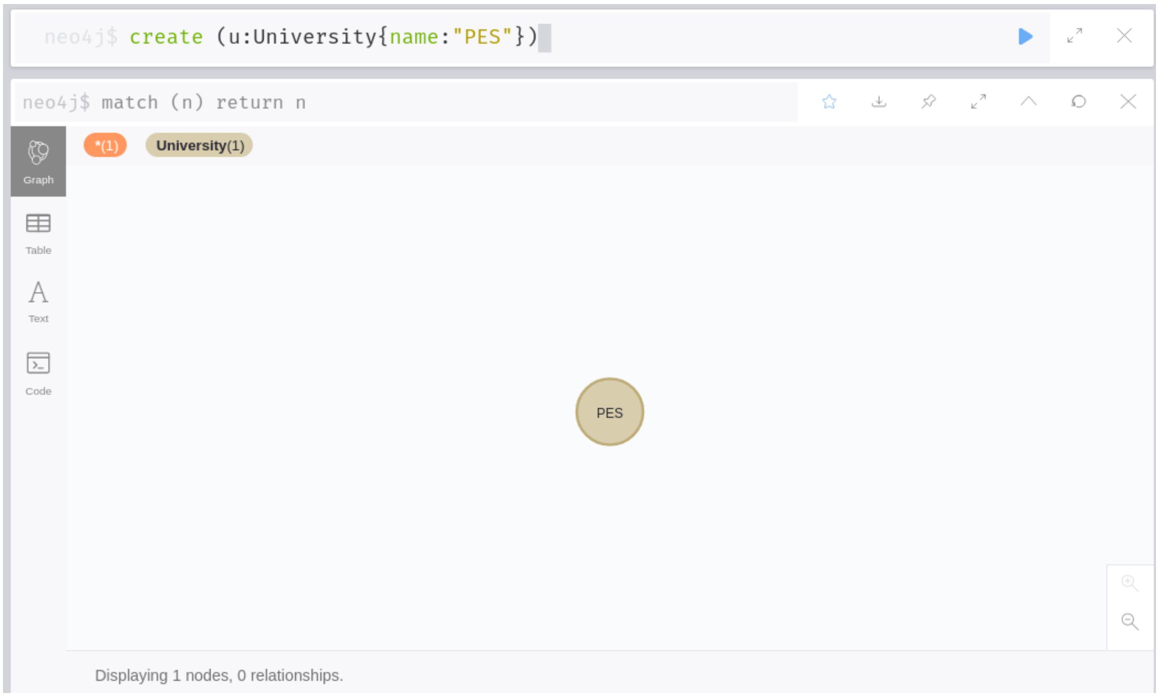
Neo4j Lab Queries Week2

Name : Chandradhar Rao
SRN : PES1UG19CS123

Week2 : Graph Based DBMS

Create Node with single label University and value name as PES

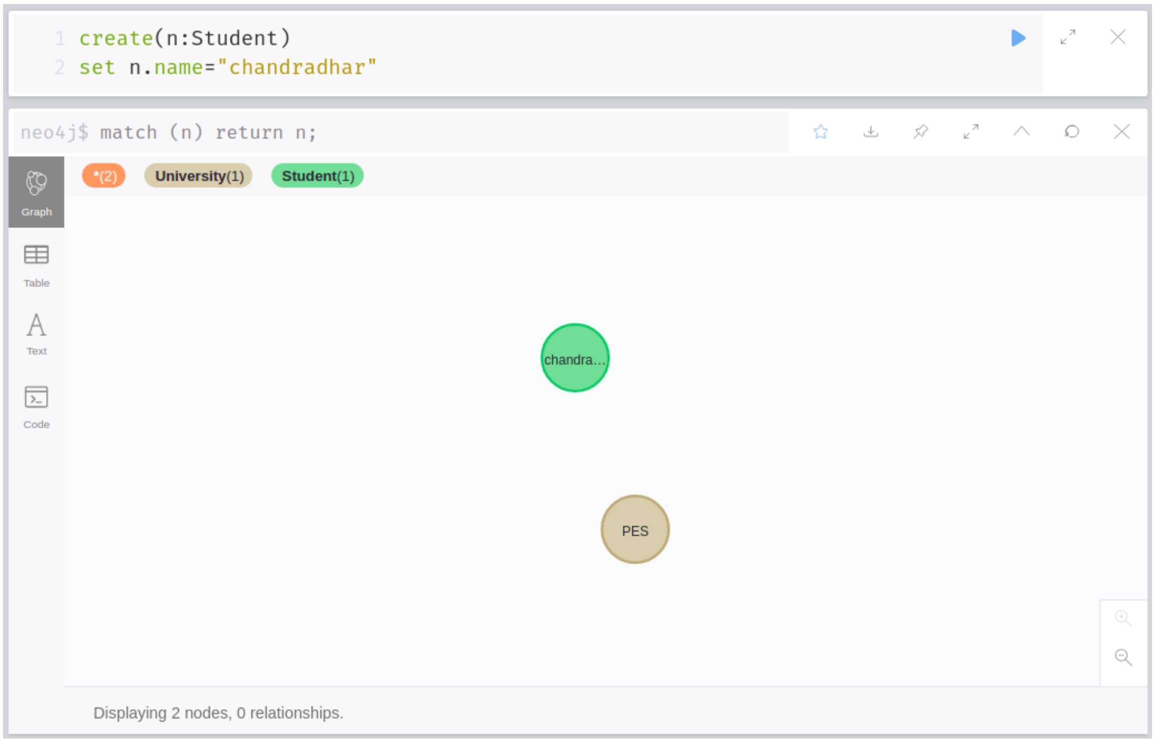


```
neo4j$ create (u:University{name:"PES"})
```

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

Displaying 1 nodes, 0 relationships.

Use set to set the key value called name with value chandradhar for a node

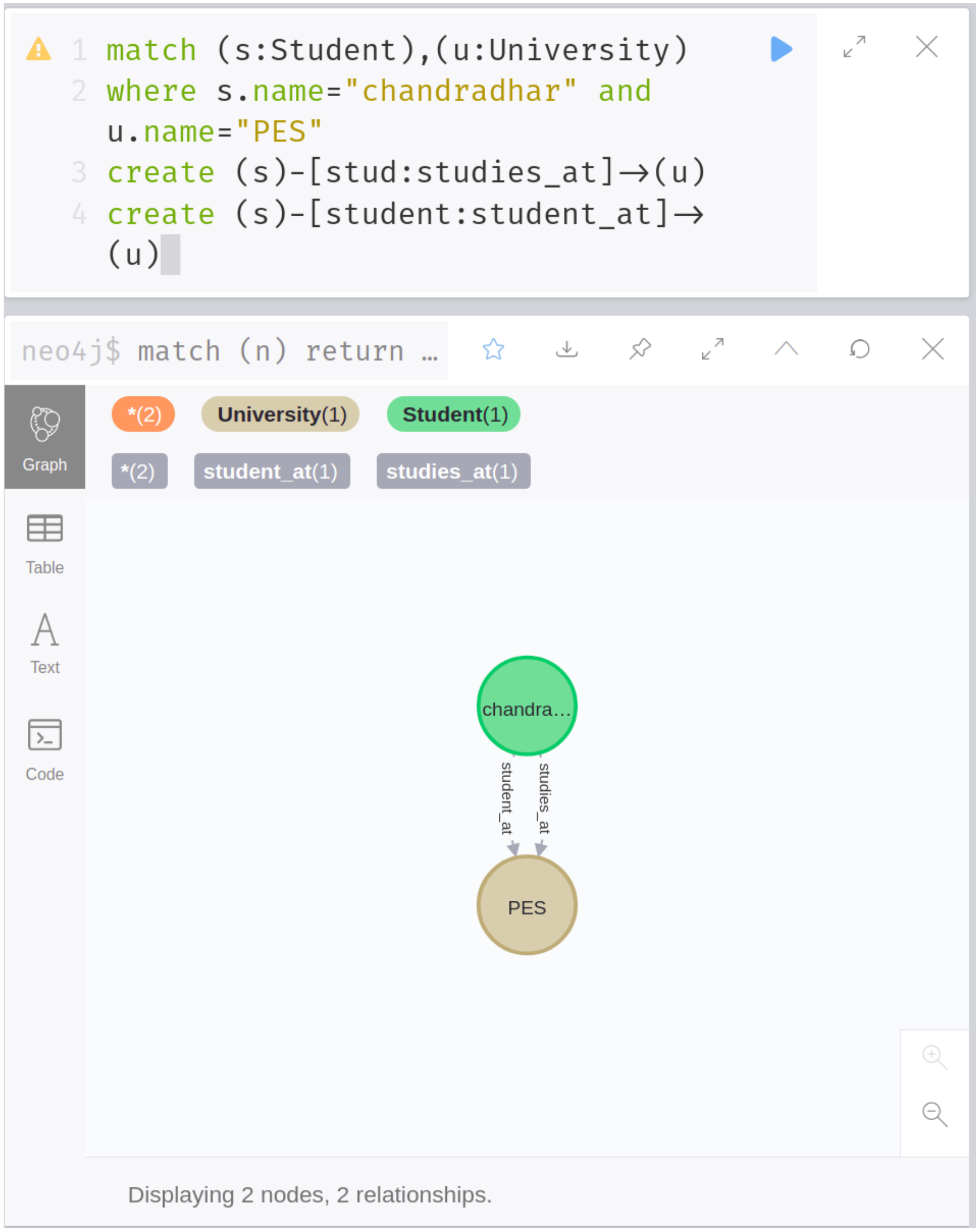


```
create(n:Student)
set n.name="chandradhar"
```

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

Displaying 2 nodes, 0 relationships.

Creating relations called studied_at and student_at between Student node chandradhar and University node pes university

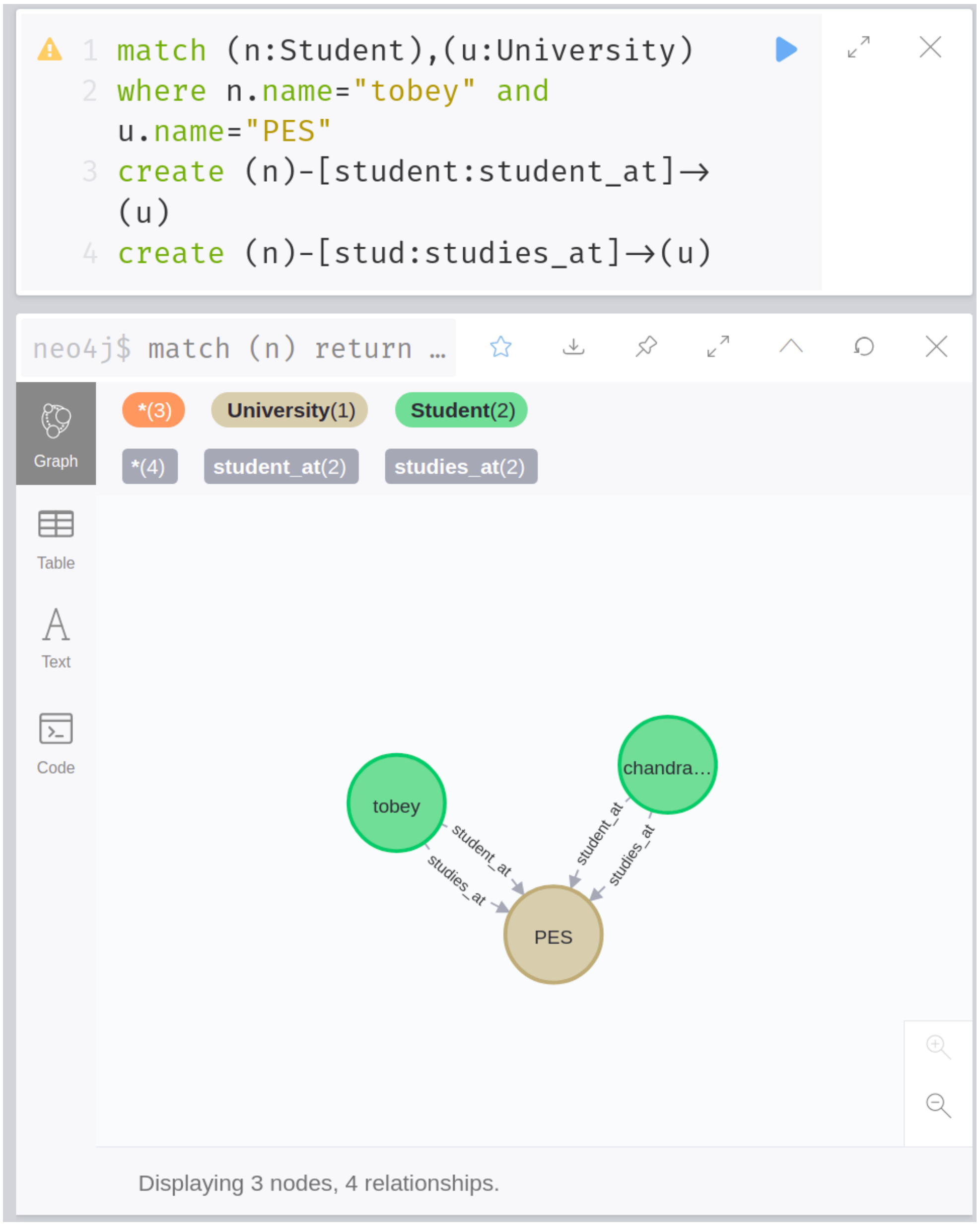


```
match (s:Student),(u:University)
where s.name="chandradhar" and
u.name="PES"
create (s)-[stud:studies_at]→(u)
create (s)-[student:student_at]→
(u)
```

```
neo4j$ match (n) return …
```

Displaying 2 nodes, 2 relationships.

Create relation student_at and studies_at between Student node with value tobey and University node PES



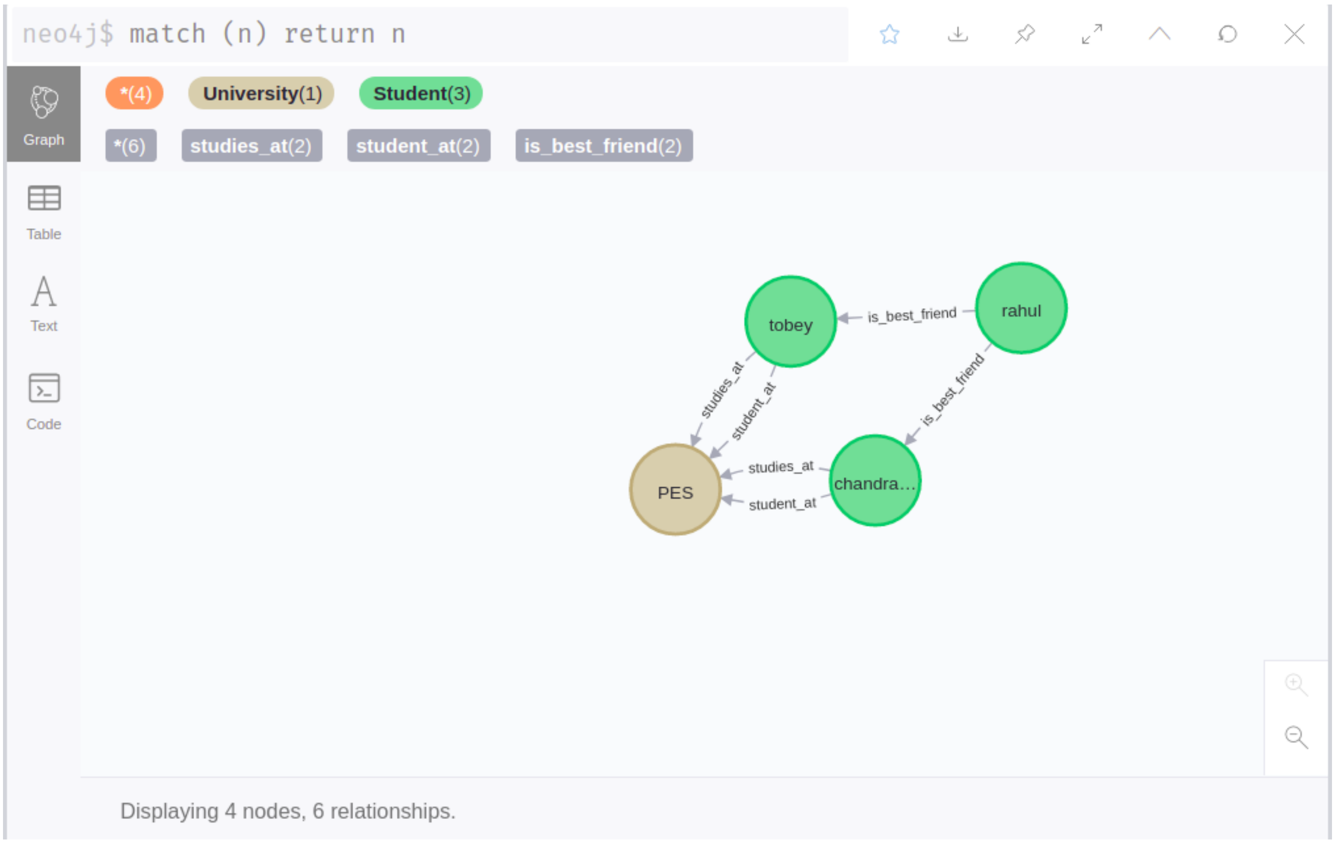
```
match (n:Student),(u:University)
where n.name="tobey" and
u.name="PES"
create (n)-[student:student_at]→
(u)
create (n)-[stud:studies_at]→(u)
```

```
neo4j$ match (n) return …
```

Displaying 3 nodes, 4 relationships.

Create a student not studying in PES University but has a best_friend relationship between tobey and chandradhar

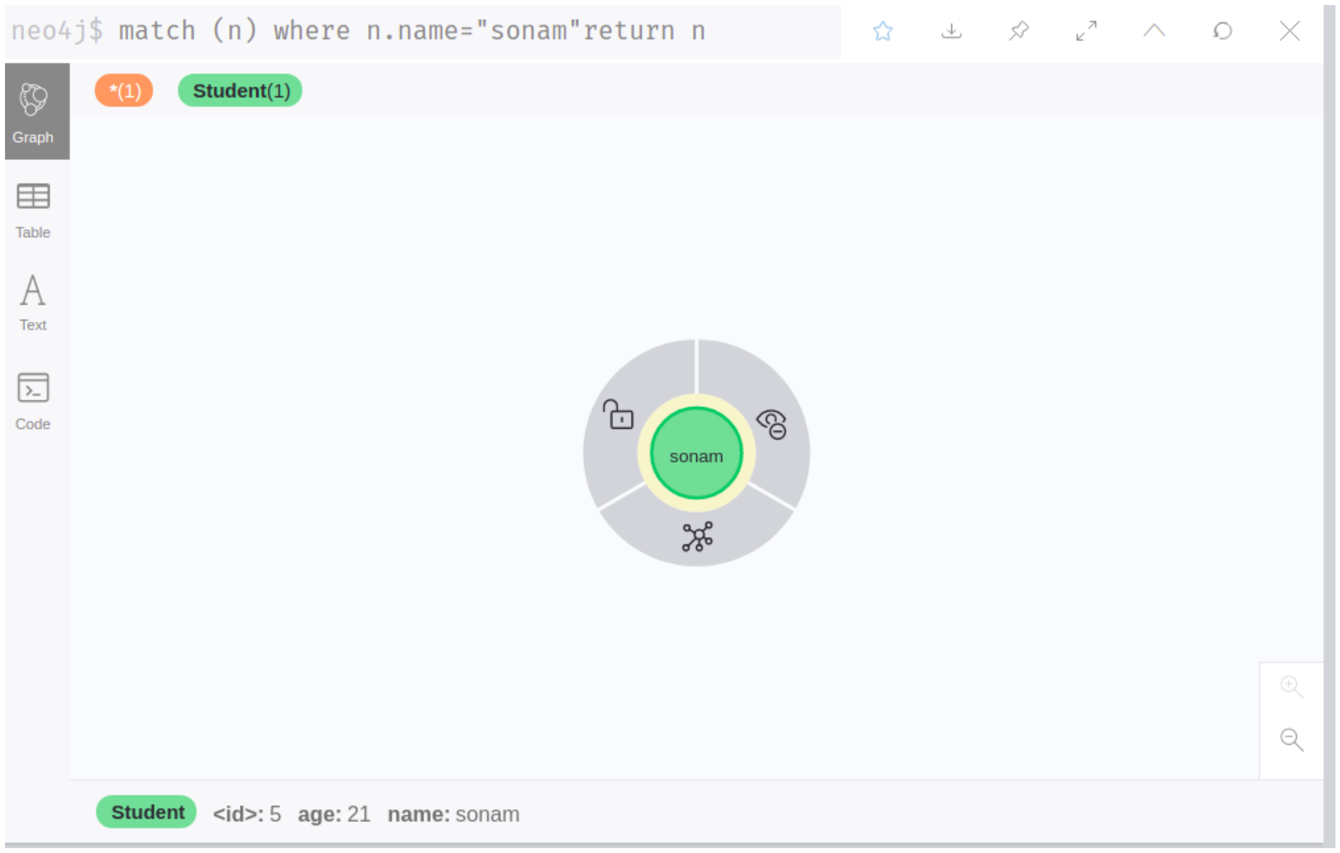
```
//create a Student not studying in pes but best_friend of tobey and chandradhar
create (n:Student{name:"rahul"})

//create relationships
match (n:Student{name:"rahul"}),(p:Student{name:"tobey"})
create (n)-[bst:is_best_friend]->(p)

match (n:Student{name:"rahul"}),(p:Student{name:"chandradhar"})
create (n)-[bst:is_best_friend]->(p)
```



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

Displaying 4 nodes, 6 relationships.

Create a node with label Student but with two key value pairs ie name=Sonum and age=21
create (n:Student{name:"sonam",age:21})



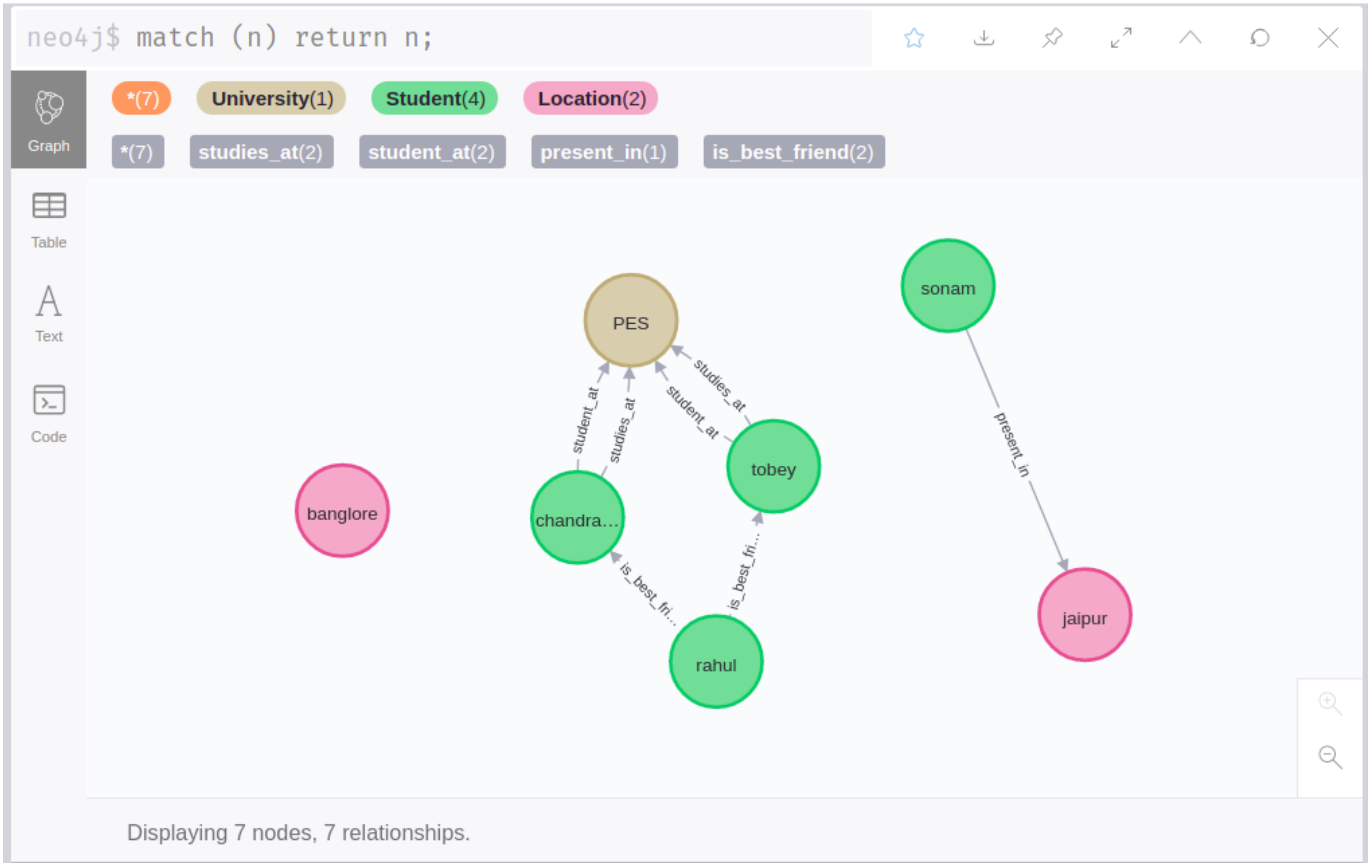
```
neo4j$ match (n) where n.name="sonam"return n
```

Student <id>: 5 age: 21 name: sonam

Create a node of label Location with value name called bangalore, another node of label location with name key having Jaipur as value.

create (l:Location{name:"banglore"})
create (l1:Location{name:"jaipur"})

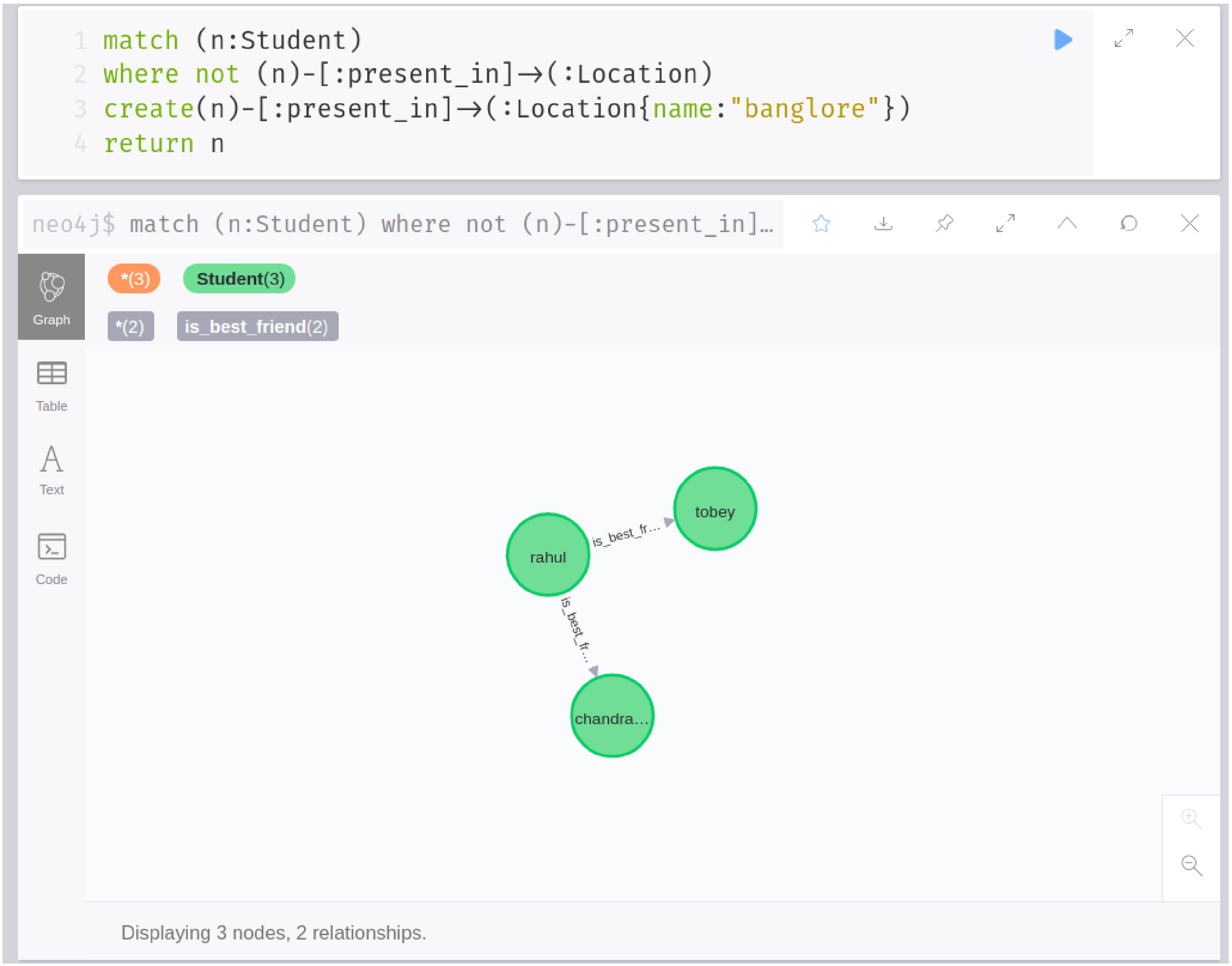
match (n:Student{name:"sonam"}),(l:Location{name:"jaipur"})
create (n)-[lives:present_in]->(l)


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

Displaying 7 nodes, 7 relationships.

Match all students without a relation present_in with the node location and set the present_in relation of those nodes with the location with name bangalore

match (n:Student)
where not (n)-[:present_in]->(:Location)
create(n)-[:present_in]->(:Location{name:"banglore"})
return n



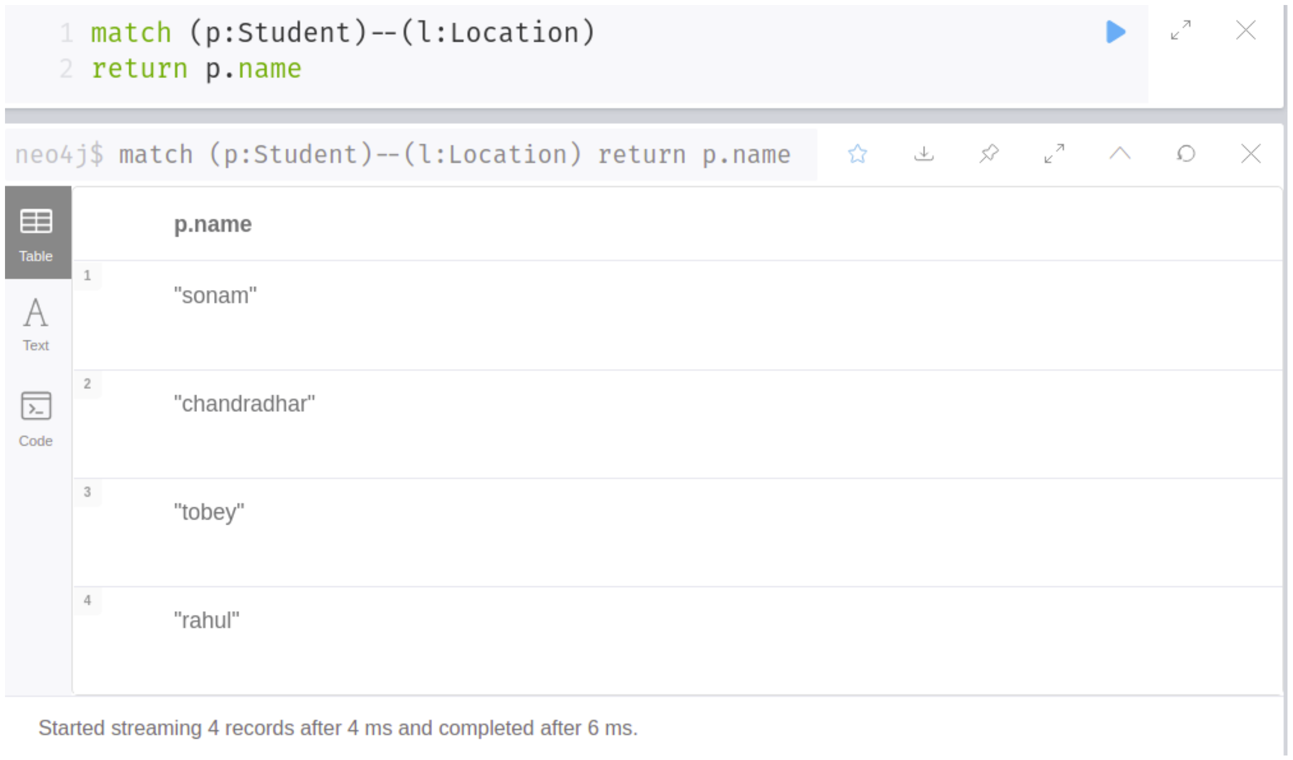
```
match (n:Student)
where not (n)-[:present_in]→(:Location)
create(n)-[:present_in]→(:Location{name:"banglore"})
return n
```

```
neo4j$ match (n:Student) where not (n)-[:present_in]…
```

Displaying 3 nodes, 2 relationships.

Retrieve the related nodes using “--”

match (p:Student)--(l:Location)
return p.name



```
match (p:Student)--(l:Location)
return p.name
```

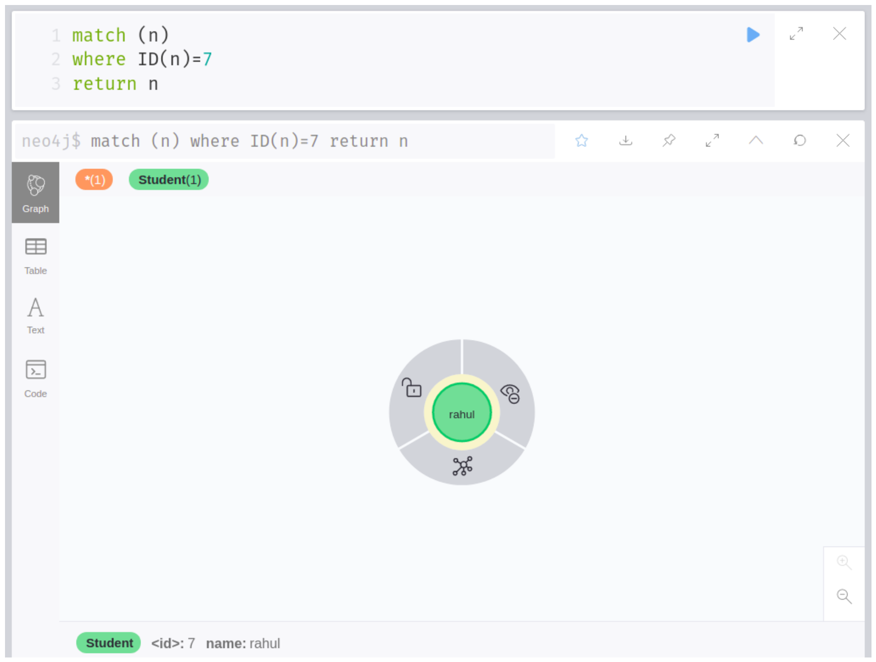
```
neo4j$ match (p:Student)--(l:Location) return p.name
```

	p.name
1	"sonam"
2	"chandradhar"
3	"tobey"
4	"rahul"

Started streaming 4 records after 4 ms and completed after 6 ms.

Retrieve nodes based on the ID

match (n)
where ID(n)=7
return n



```
match (n)
where ID(n)=7
return n
```

```
neo4j$ match (n) where ID(n)=7 return n
```

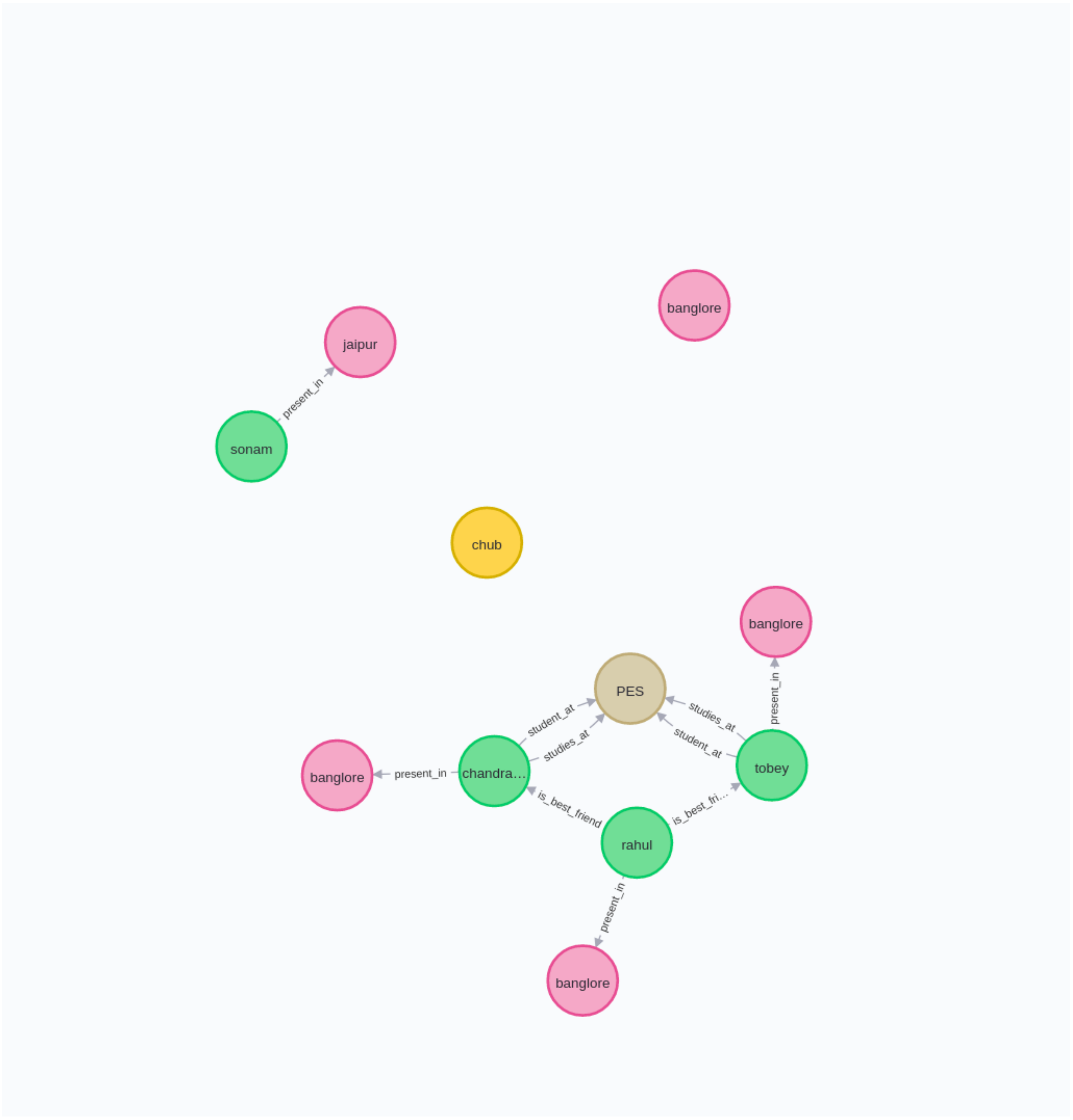
Student <id>: 7 name: rahul

Create node with multiple labels:

Here we create node with label Animal and Mammal snd give it a key called name with value chub and type as Dog.

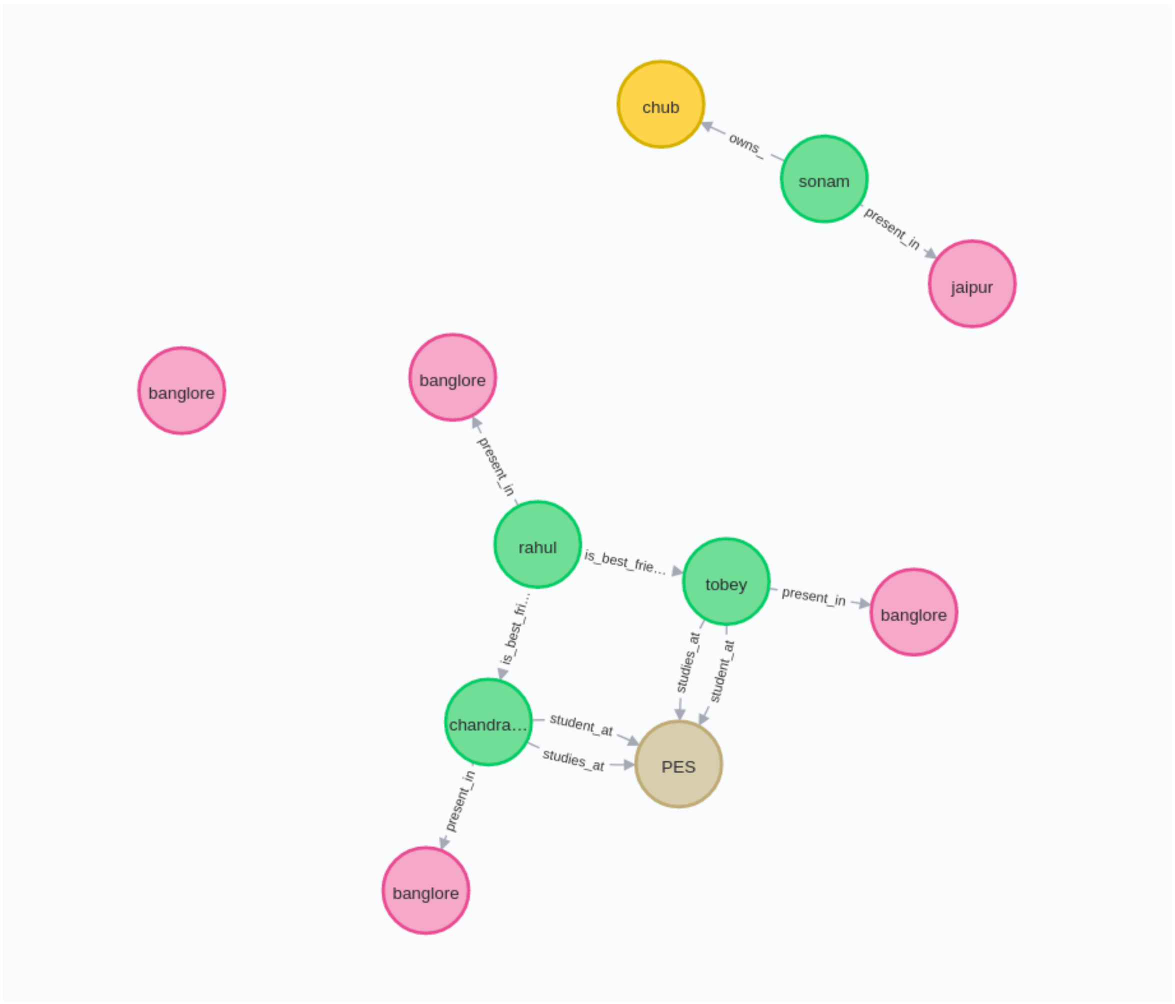
This node would be a pet owned by Sonam

```
create (n:Animal:Mammal)
set n.species="Dog"
set n.name="chub"
```



Create relationship between Sonam node and chub Anmal node where sonam owns the pet called cub:

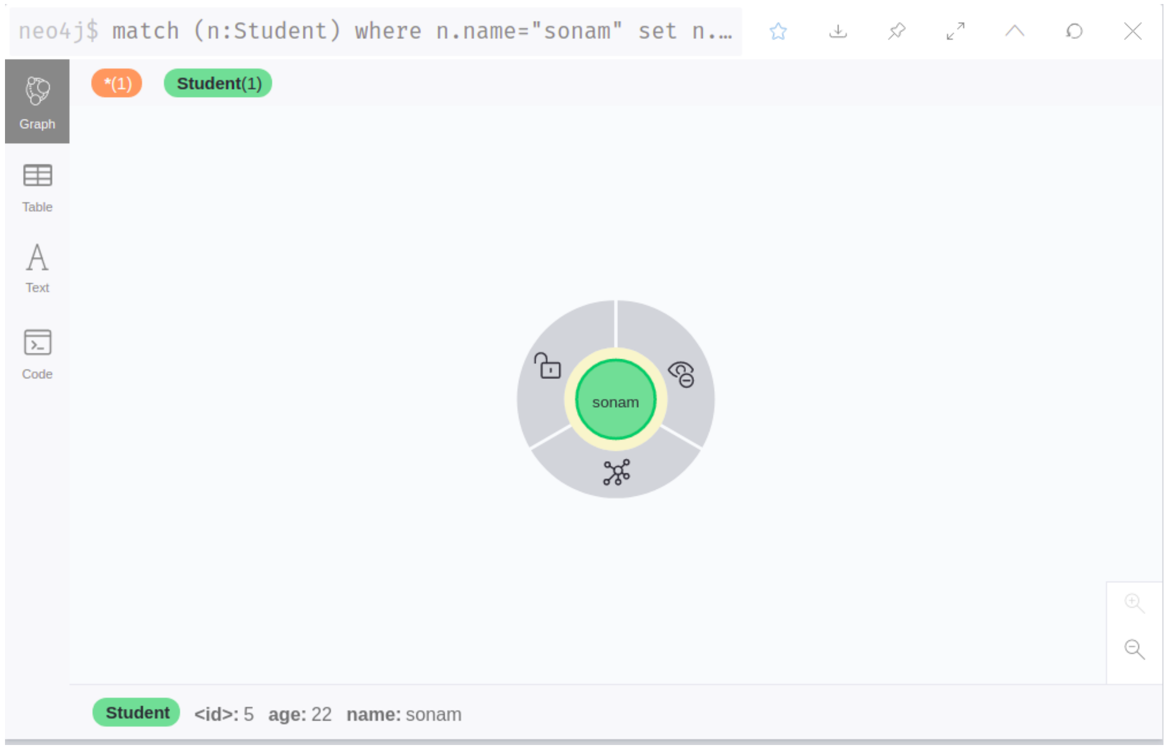
```
match (n:Animal{name:"chub"})
match (p:Student)
where p.name="sonam"
create (p)-[o:owns_]->(n)
```



Use the set command to update a value ofa key:

Here we update sonam node’s age key after her birthday

match (n:Student)
where n.name="sonam"
set n.age=22
return n

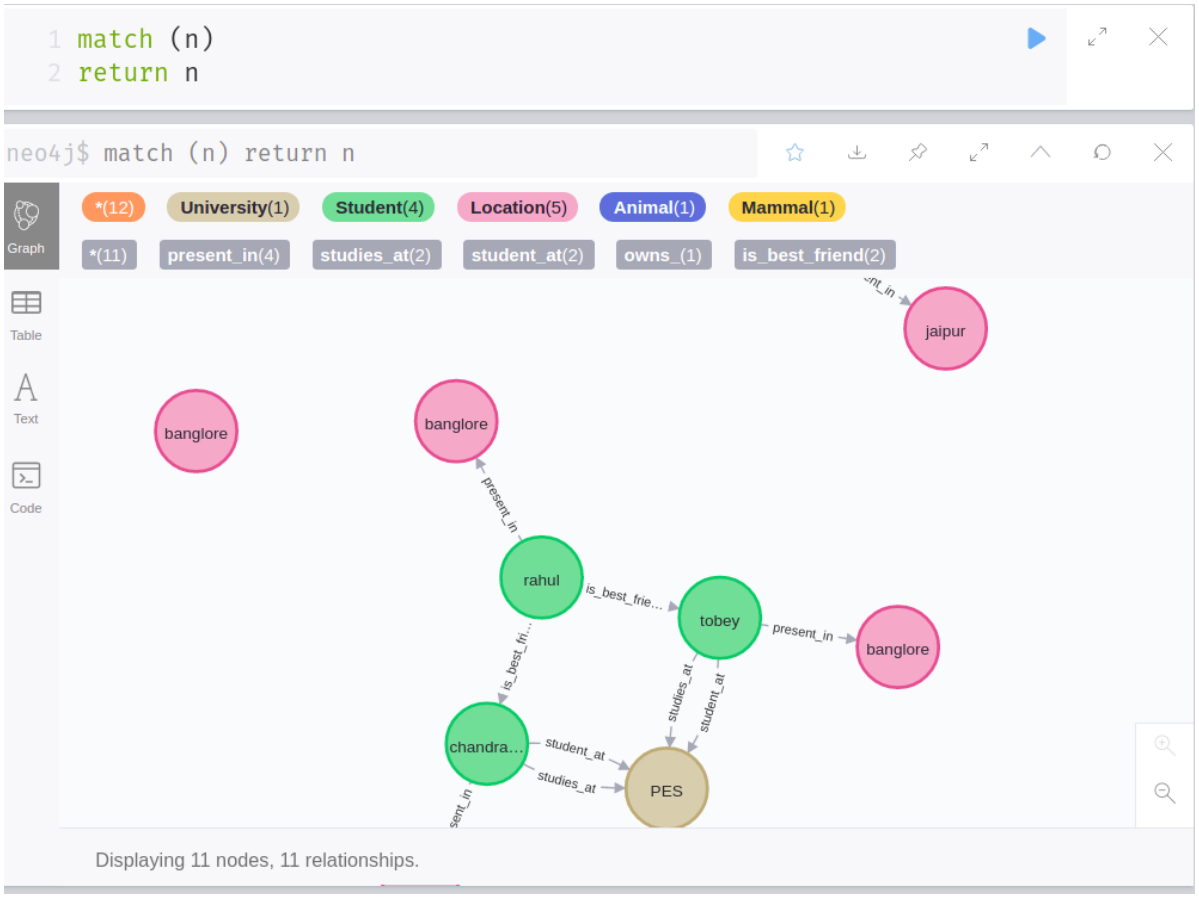


```
neo4j$ match (n:Student) where n.name="sonam" set n.…
```

Student <id>: 5 age: 22 name: sonam

Display the Entire graph:

match (n) return n



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

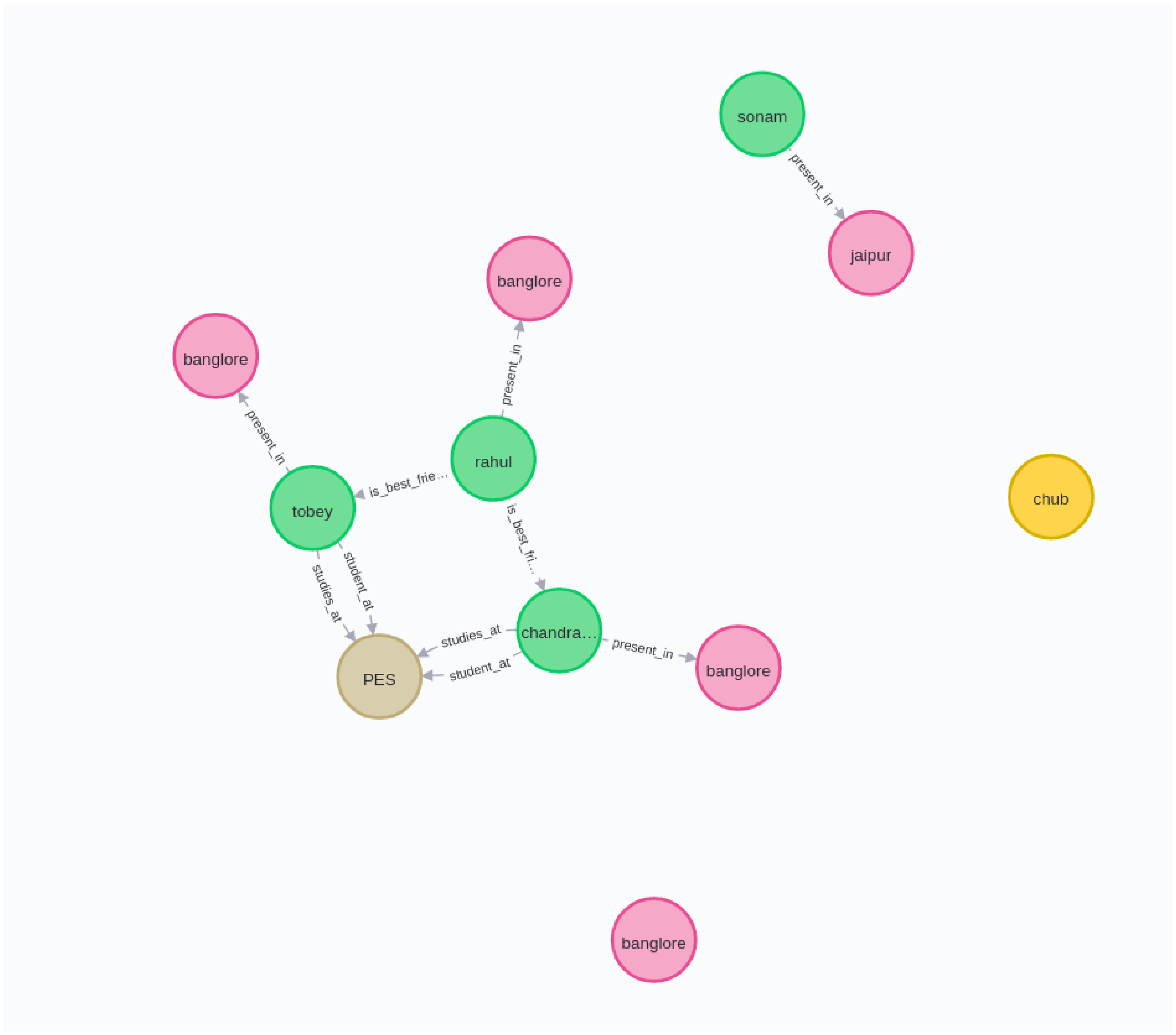
```
neo4j$ match (n) return n
```

Displaying 11 nodes, 11 relationships.

Delete Specific node:

Lets say that dog died due to an accident :(, then we need to delete the relationship between dog node and sonam node first:

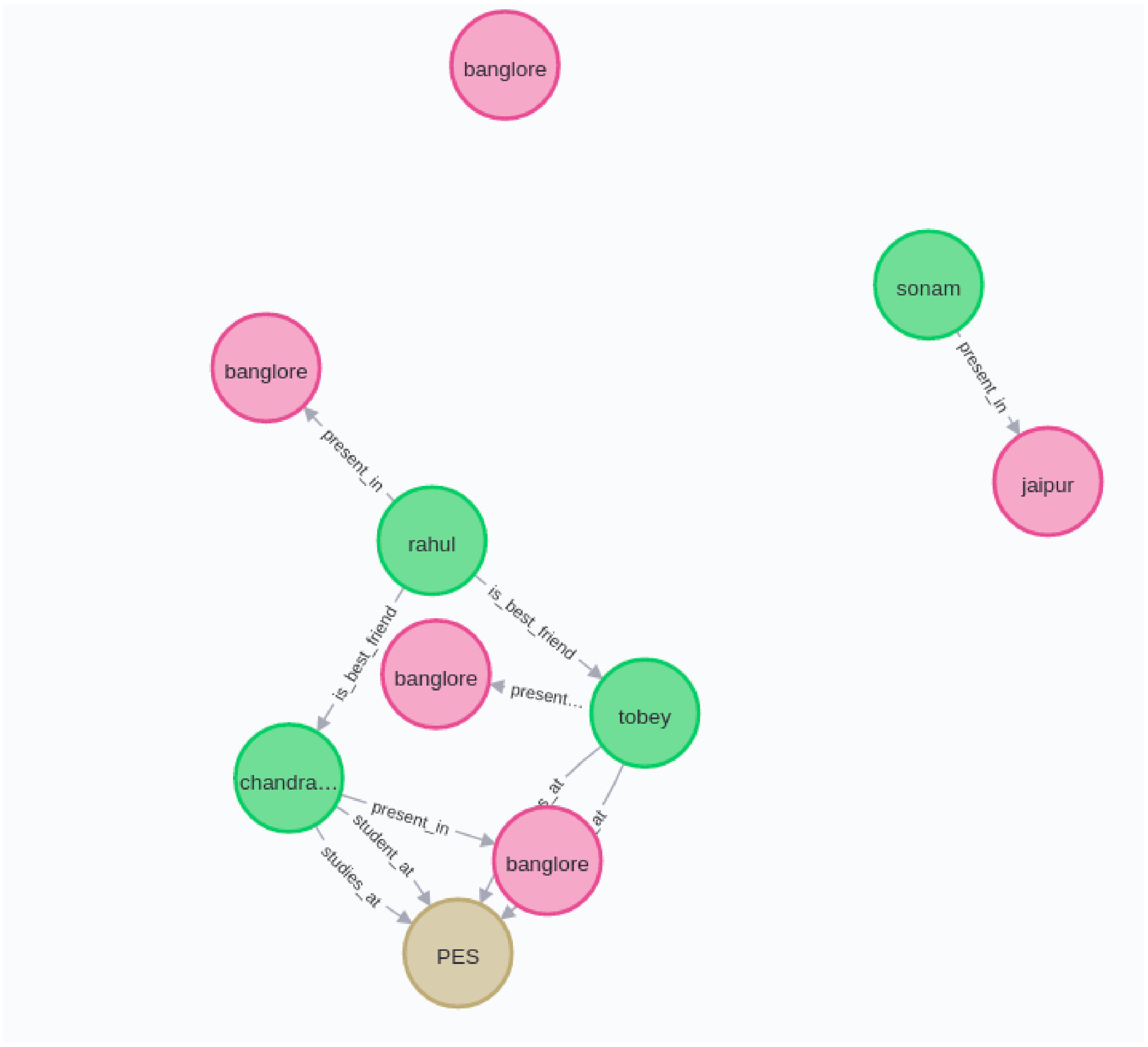
```
match
(p:Student{name:"sonam"})-[o:owns_]->(a:Animal{species:"Dog",name:"chub"})
delete o
```



Delete A specific Node:

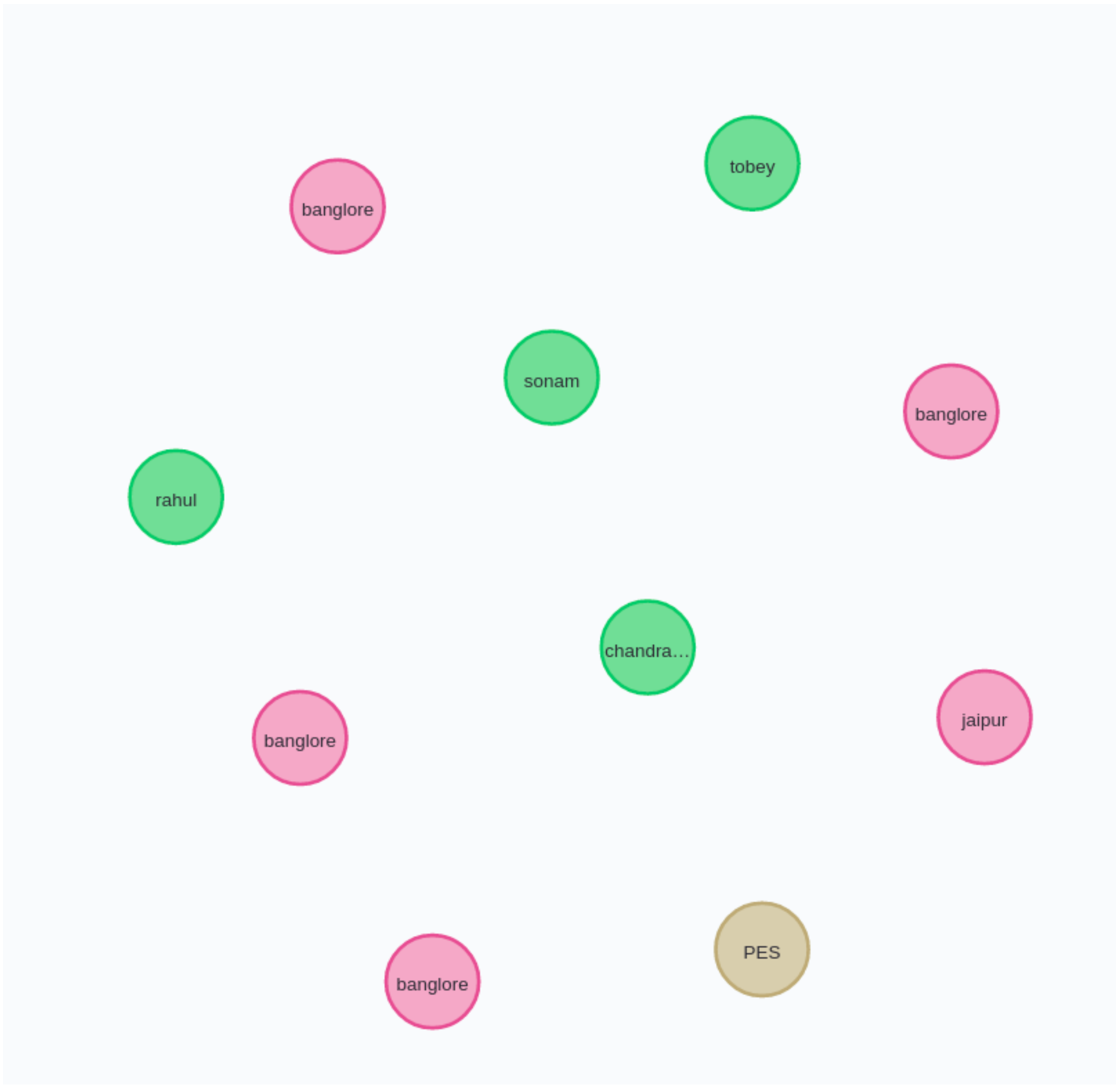
Next after detaching all relationships,we need to delete the node Dog itself:

match (d:Animal)
where d.species="Dog" and d.name="chub"
delete d



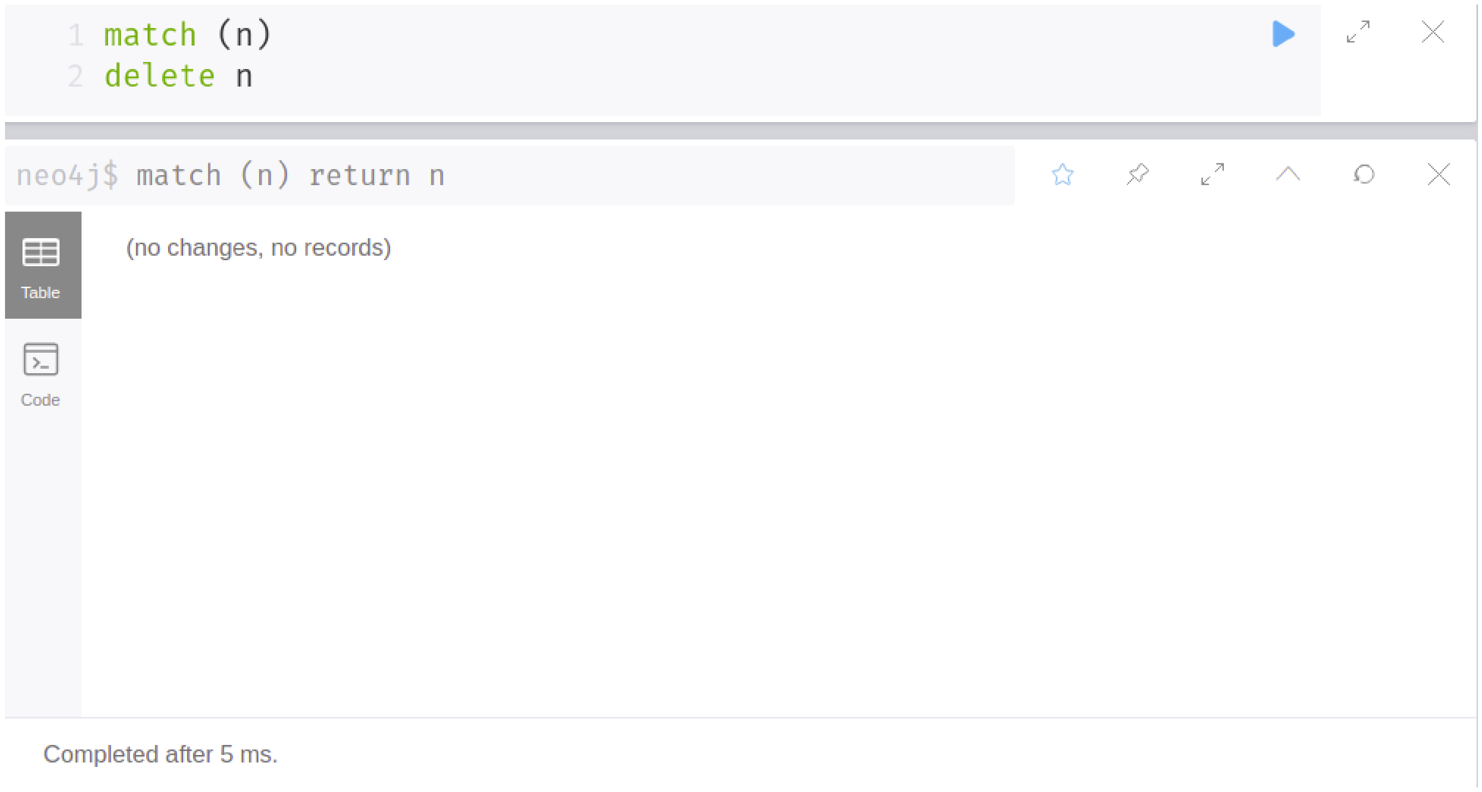
Deleting all the relationships between all the nodes:

```
match ()-[r]->()
delete r
```



Finally, Delete all nodes itself of the graph:

match (n)
delete n



```
match (n)
delete n
```

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

```
(no changes, no records)
```

Completed after 5 ms.