**DBMS Laboratory**

**UE19CS304**

**5th Semester, Academic Year 2021-22**

Week #: 2 - Operation on Neo4j GraphDB

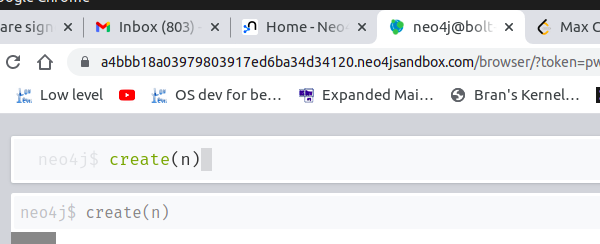
Date: 14/9/2021

|  |  |  |
| --- | --- | --- |
| Name :  SUMUKH RAJU BHAT | SRN :  PES1UG19CS519 | Section :  H |

1. **Create a single node**

*create (n)*

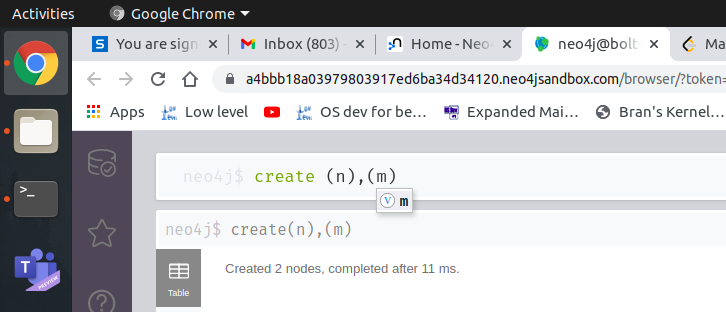
*-* Create a node with no props or labels



1. **Create multiple nodes**

*create (n), (m)*

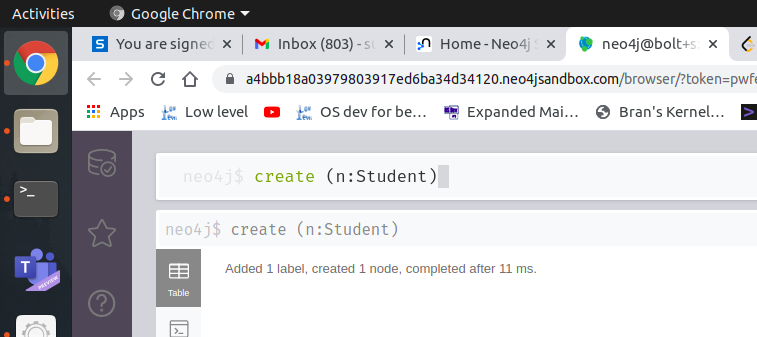
*-* Create multiple nodes with no labels or props



1. **Create a node with label**

*create (n:Student)*

*-* Create a node with Student label and no props



1. **Create a node with label and properties**

*create (n:Student {name: “ABC”, SRN: “123”})*

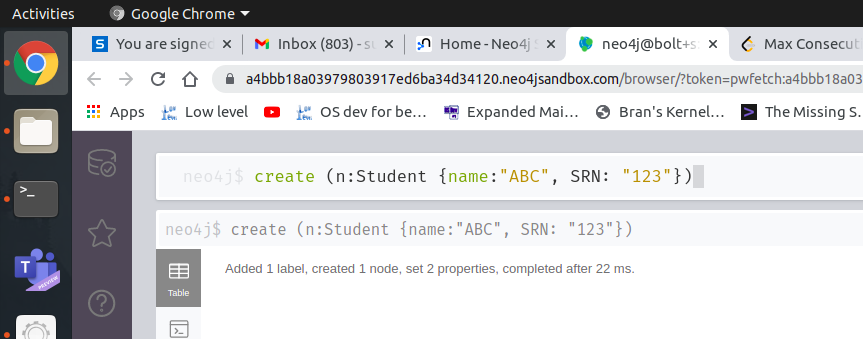
*create (n:Student {name: “LMN”, SRN: “456”})*

*create (n:Student {name: “JHI”, SRN: “789”})*

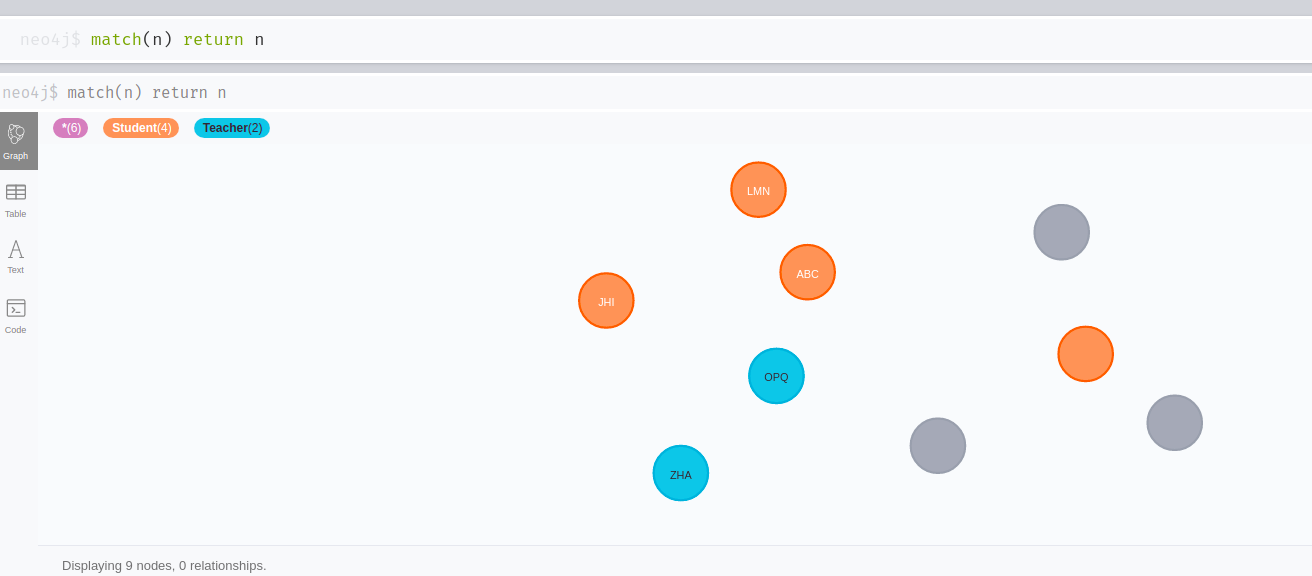
*create (n:Teacher{name: “ZHA”, FID: “439”})*

*create (n:Teacher{name: “OPQ”, FID: “731”})*

- Create 3 nodes with label Student and props name, SRN and 2 nodes with label Teacher with props name, FID.



Result:



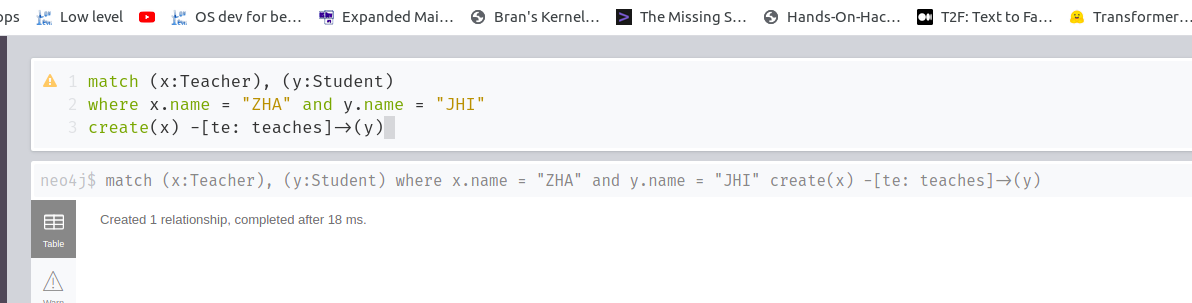
1. **Create relation between specific nodes**

*match (x:Teacher), (y:Student)*

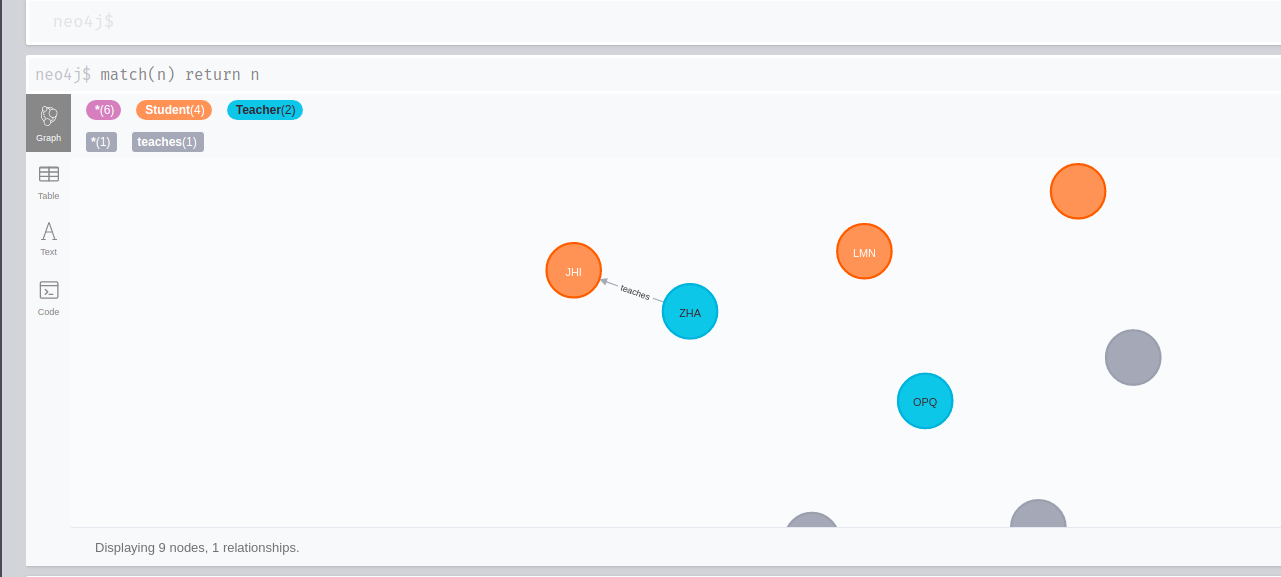
*where x.name = “ZHA” and y.name = “JHI”*

*create (x)-[tea:teaches]->(y)*

*-* Creates a “teaches” relationship between a teacher JHI and student ZHA.



Result:

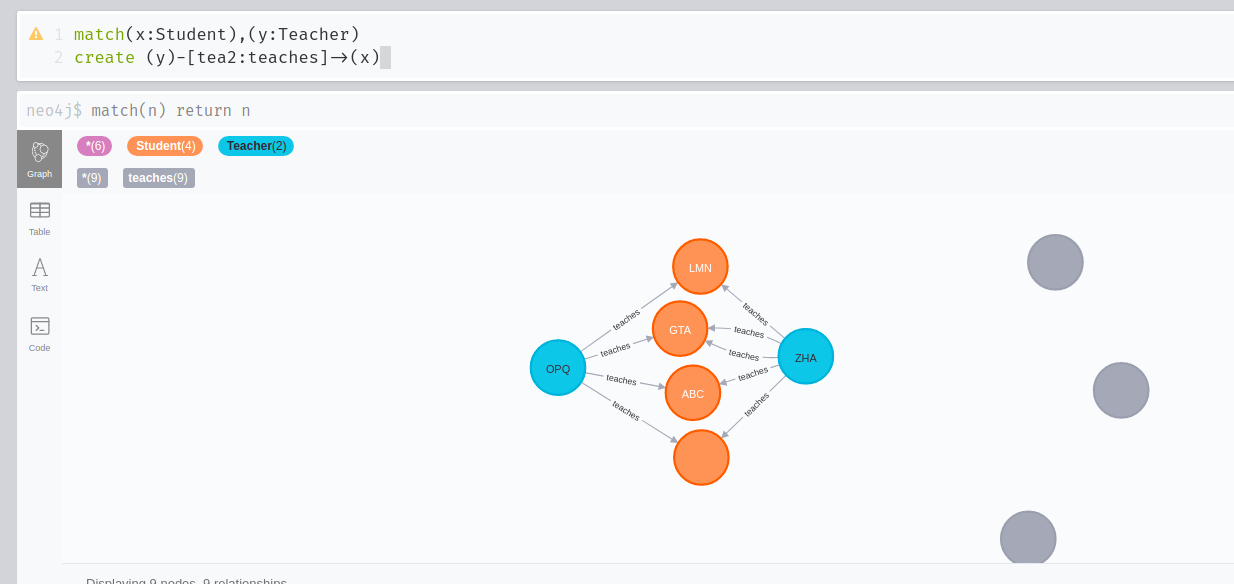


1. **Create relation between nodes**

*match(x:Student), (y:Teacher)*

*create (y)-[]->(x)*

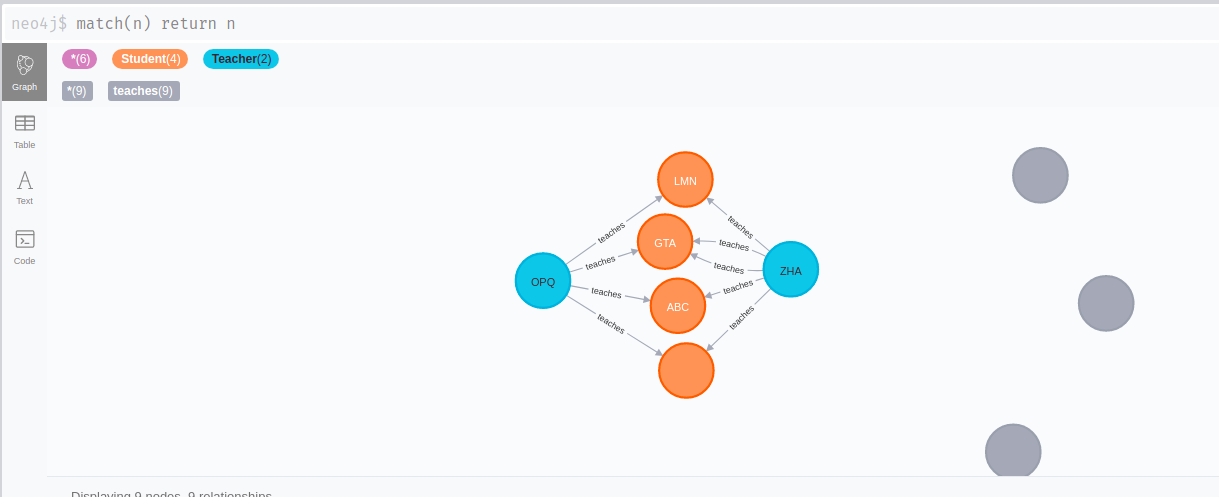
- Create a relationship between all Teacher and Student nodes as “teaches”



1. **Display the graph**

*match(n) return n*

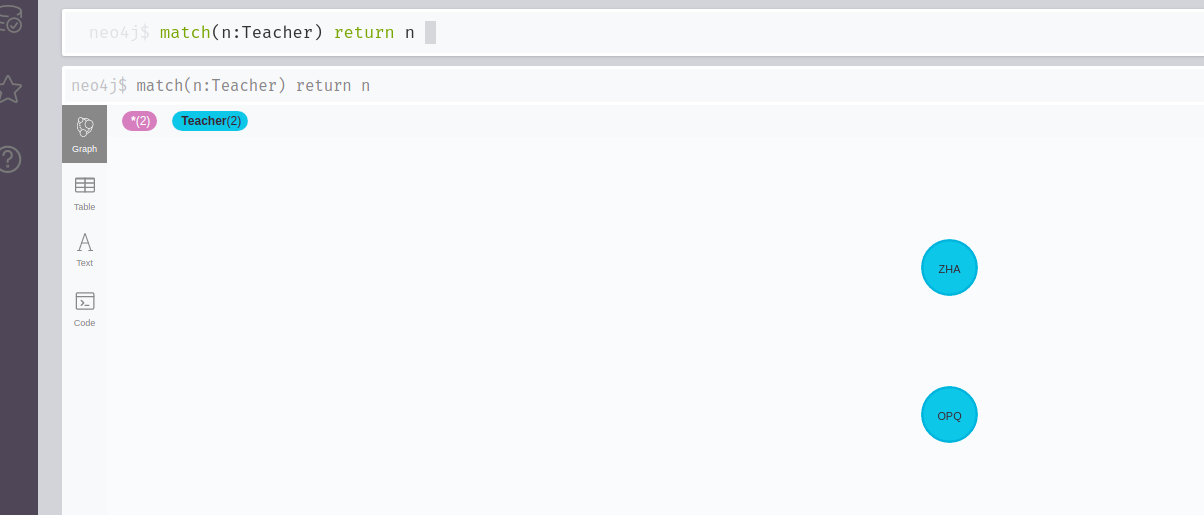
- Displays all the nodes



1. **Display specific nodes based on label and ids**

*match(n:Teacher) return n*

- Displays only the nodes with label Teacher



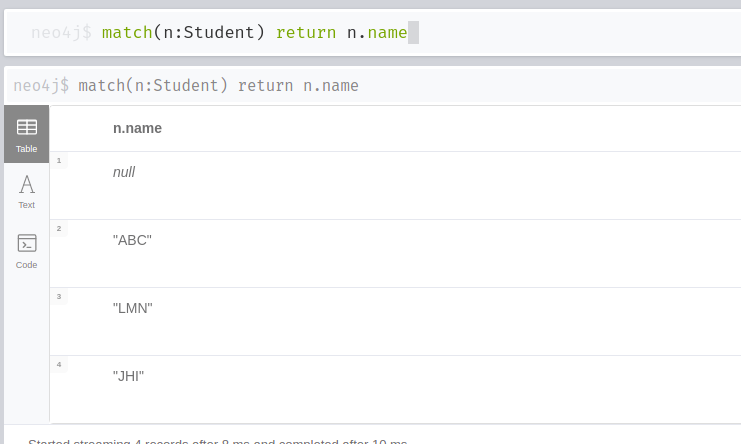
*match(n:Teacher) where id(n) = 8 return n*

*-* Displays the Teacher node with condition that id = 8



*match(n:Student) return n.name*

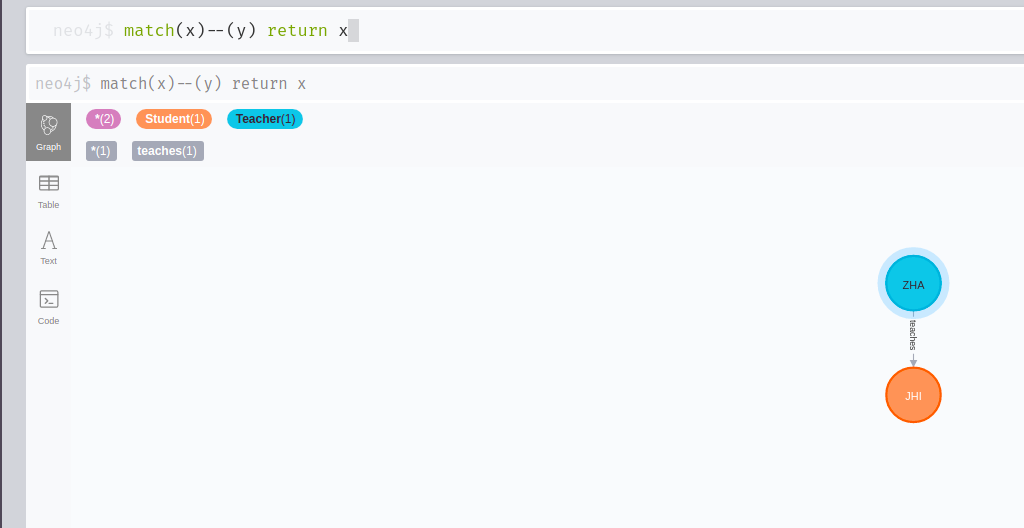
- Returns the name prop of all the Student labeled nodes.



1. **Display Relationships**

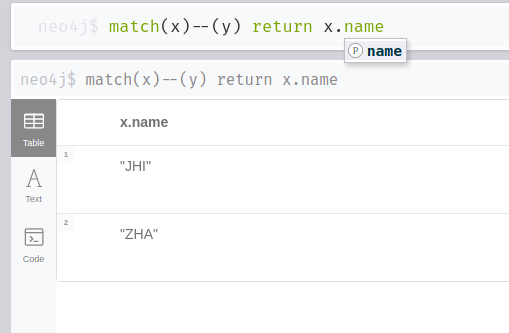
*match(x)--(y) return x*

- Return all the relationships in the db. The output showcases the output only when relationship was added.



*match(x) -- (y) return x.name*

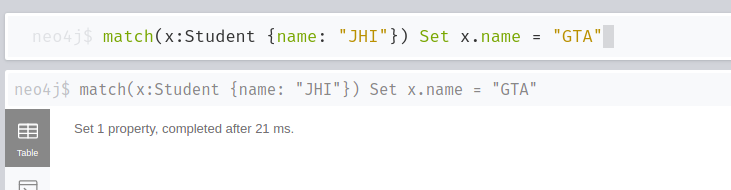
- Return the name prop of all the nodes who have a outgoing(In the example x is related to y implies x has a outgoing relationship and y has incoming relationship) relationships in the db. (The output showcases the output only when relationship was added.)



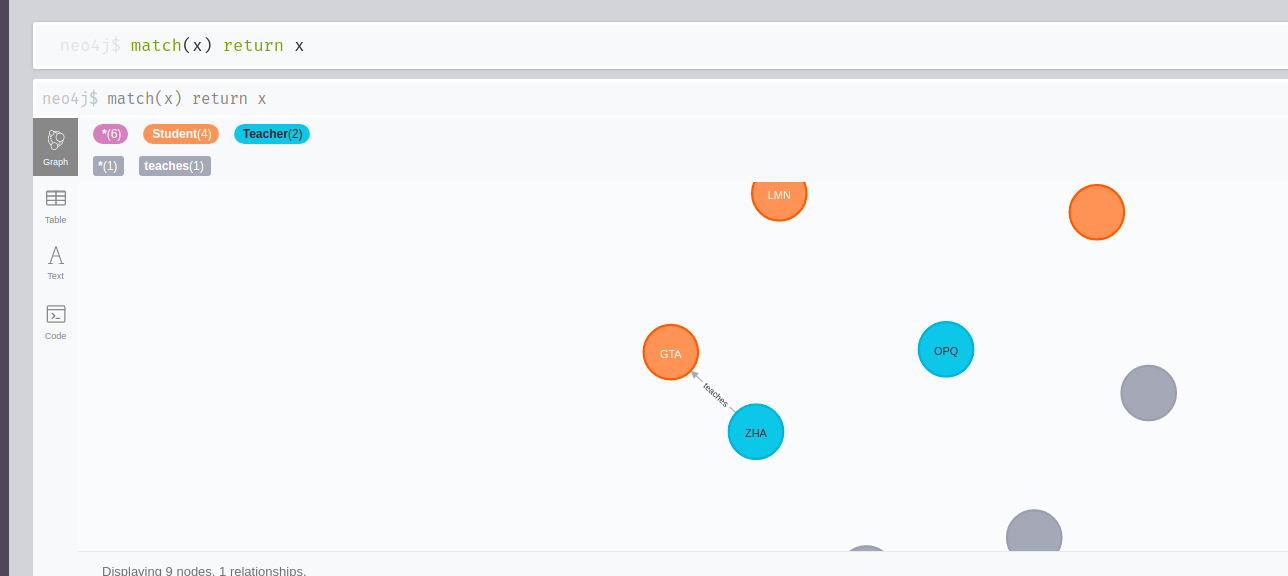
1. **Update specific node**

*match(x:Student {name: “JHI”}) Set x.name = “GTA”*

- Update the name prop of the Student node where name was originally JHI. (The output showcases the output only when relationship was added.)



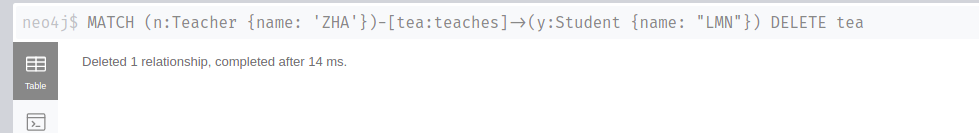
Result:



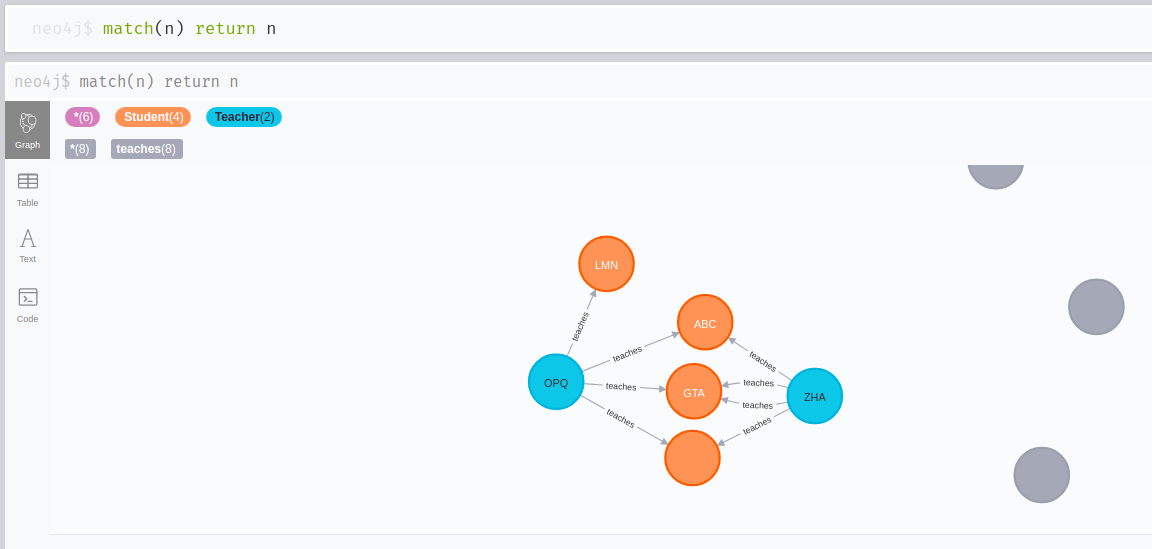
1. **Delete certain relationship**

*match(n:Teacher {name: “ZHA”})-[tea:teaches]->(y:Student {name: “LMN”}) delete tea*

- Delete the teaches relationship between Teacher node with name prop LMN and Student node with name prop LMN.



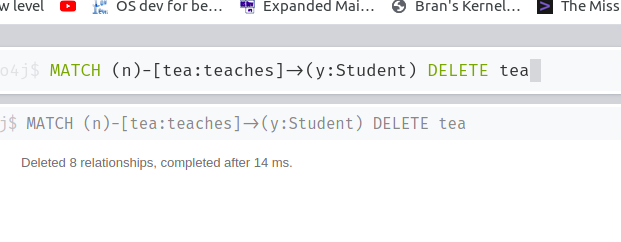
Result:



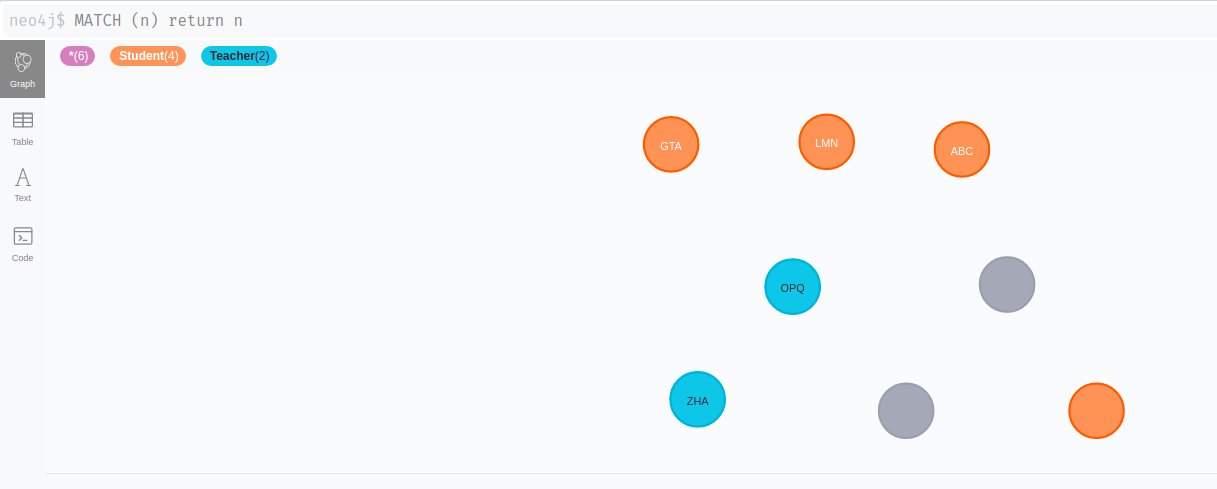
1. **Delete all relationship**

*match (n)-[tea:teaches]->(y:Student) delete tea*

- Deletes all the teaches relationship between any node to Student nodes.



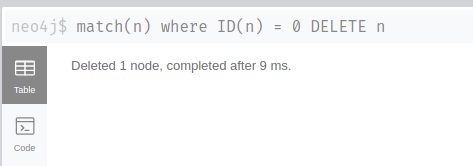
Result:



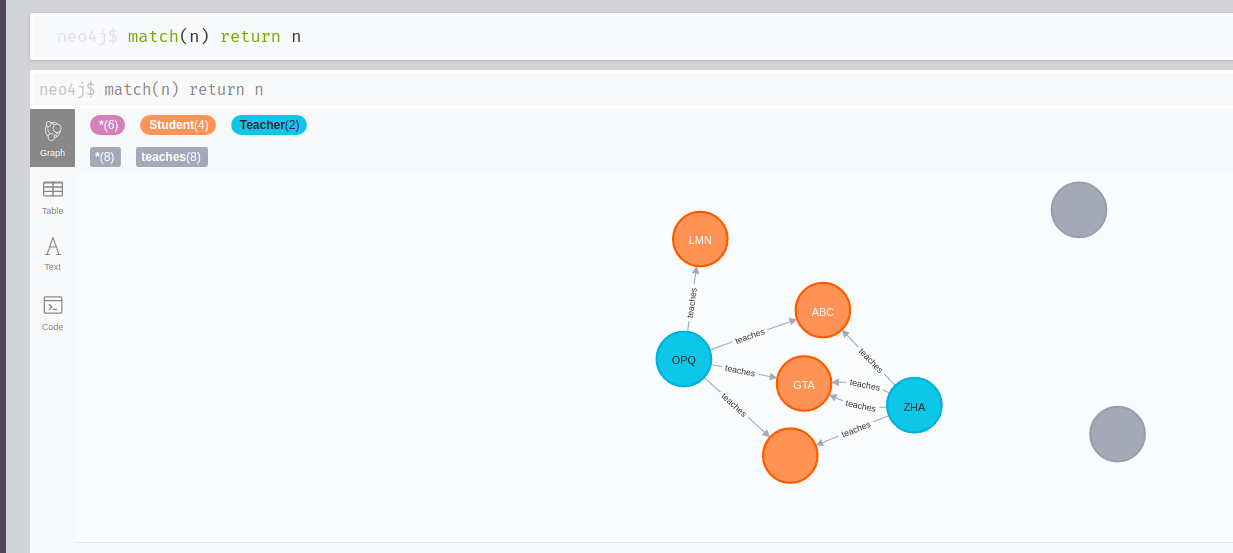
1. ***Delete certain node***

*match(n) where ID(n) = 0 delete n*

- Delete the node with id = 0

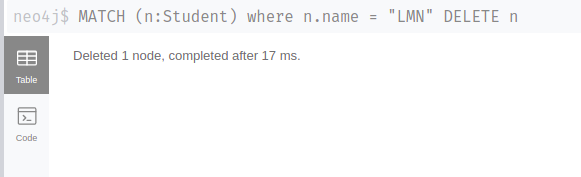


Result:

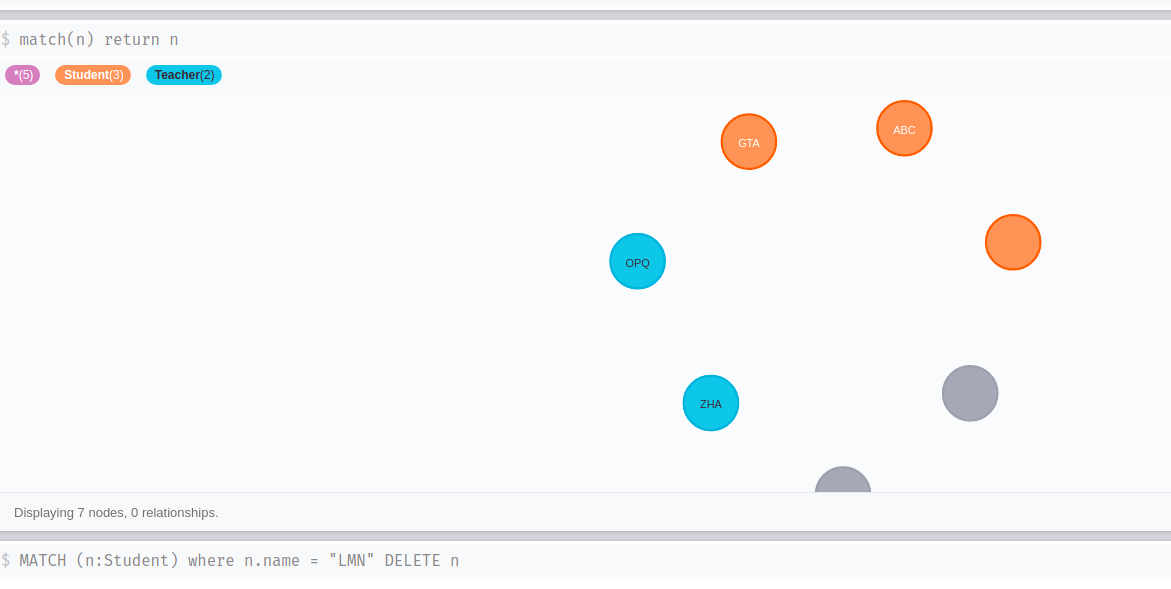


*match(n:Student) where n.name = “LMN” delete n*

- Delete the Student node with name prop as LMN



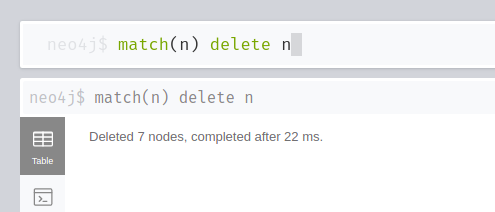
Result:



1. **Delete all nodes**

*match(n) delete n*

- After all relationships are deleted, delete all the nodes in the db



Result:

