Aim:-

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

A> 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 separed by comma with in the flower bro ces "{}",

Syntax:-

following is the syntax to create a node with properties.

CREATE (node:- label {key 1: value, key 2: value 1----y]

* Returning the created Node

To verify the creation of the node, type and execute the following query in the dollar promf - MATCH(T) RETURN.

* creating Relationship

we can create- we will specify relation step within the square braces "[]" depending on the direction of the relationship it is placed between type ten "_" and arraw "_ > as shown in the following syntax.

Syntax

Following is the syntax to create crela from stip using the CREATE clause. we will specify relation with the square braces "[]" depending on the direction of the relateonship it is placed between hypton "_" and arrow "_>" as shown it the following syntax.

Syntax:-

Following is the syntax to create a relation shop using the MATCH clause.

MATCH Ca:labe of Model). cb: Labeol Node))

WHERE a. name= "name of node1 "AND
         b. name = "Name of Node2"

CREATE (a)- [: Relation]--> (b)
RETURN a, b

A) Deleting a particular Node

(co delete a particular node, you need to specify the specify the details of the node in the place of "n" the above query.

Syntax:

Following is the syntax to delete a particular node from clleous using the DELETE clave.

```
MATCH (node: label {eroperties----------$]
DETACH DELETE node.
```

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

```
create (n: student {§d: "VTU14 so",
    {name: "John",
    dept name: "css"}
    )
```

OUTPUT:-

Added 1 label ccreated 1 note, set 3 properties (completed after 232 ms. create cn: student {sid, "VTU 1450",

```
{na: student {sid: "VTU14 501", sname: "Dhar sana"
deptname: " EEE"})
```

output :-

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

```
create (n: student {sid: "VTU14502", "sname: "vijay"
deptname: "CSE"})
```

output :- Added 1 label 1, created 1 node, sets properties completed after 12ms.
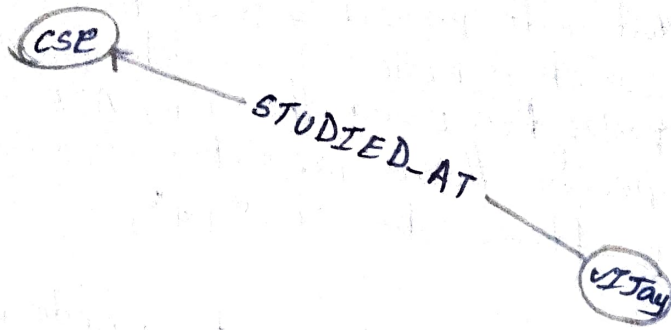
```
create (n: dept {deptname: "CSE", deptid: "dool"}
```

output :- Added 1 label, created 1 node, set 2 properties, completed after 72 ms,

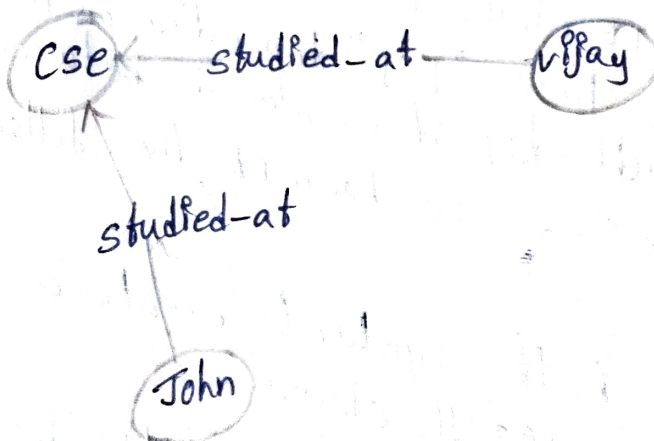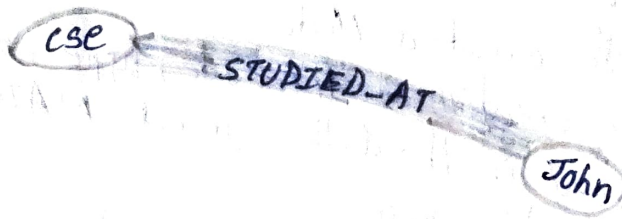select all the node nodes in your database using match connected.

* match (n) return (n).

OUTPUT:-

(CSE) ← STUDIED_AT — (VIJay)

OUTPUT:-

match

(CSE) ← STUDIED-AT — (John)

(CSE) ←── studied-at ──── (vijay)
  ↑
  │ studied-at
  │
(John)

* match (n): student) return (n)

a) create relationship b/w student and cse :-

MATCH (s: student), (d:dept) WHERE s.sname=
'vijay' AND d.deptname= 'CSE'.
  CREATE (s) - [st: STUDENTIED_AT] → (d)

return s,d.

* MATCH (s: student) (d:dept) WHERE s.sname = "John"
AND d.dept name= 'CSE'.
CREATED (s) = [st: STUDFED_AT] → (d)
returns, d.
* 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.

Result:- The implemented CRUD operation like creating, inserting, finding and removing operations using Graph DB is successfully executed.