PRACTICAL-7

AIM: WORKING WITH Neo4j DATABASE

Neo4j is a graph database management system that stores and manages data as nodes, edges, and properties.

Use :

- Neo4j is used in a wide range of applications, such as social networking, recommendation engines, fraud detection, and supply chain management.
- It is particularly useful for applications that involve complex relationships between data points.

Pros:

- Neo4j's strength is its ability to query complex relationships and quickly traverse large datasets.
- It is a highly scalable database system and can handle large amounts of data with ease.
- Its query language, Cypher, is easy to learn and use.

Cons:

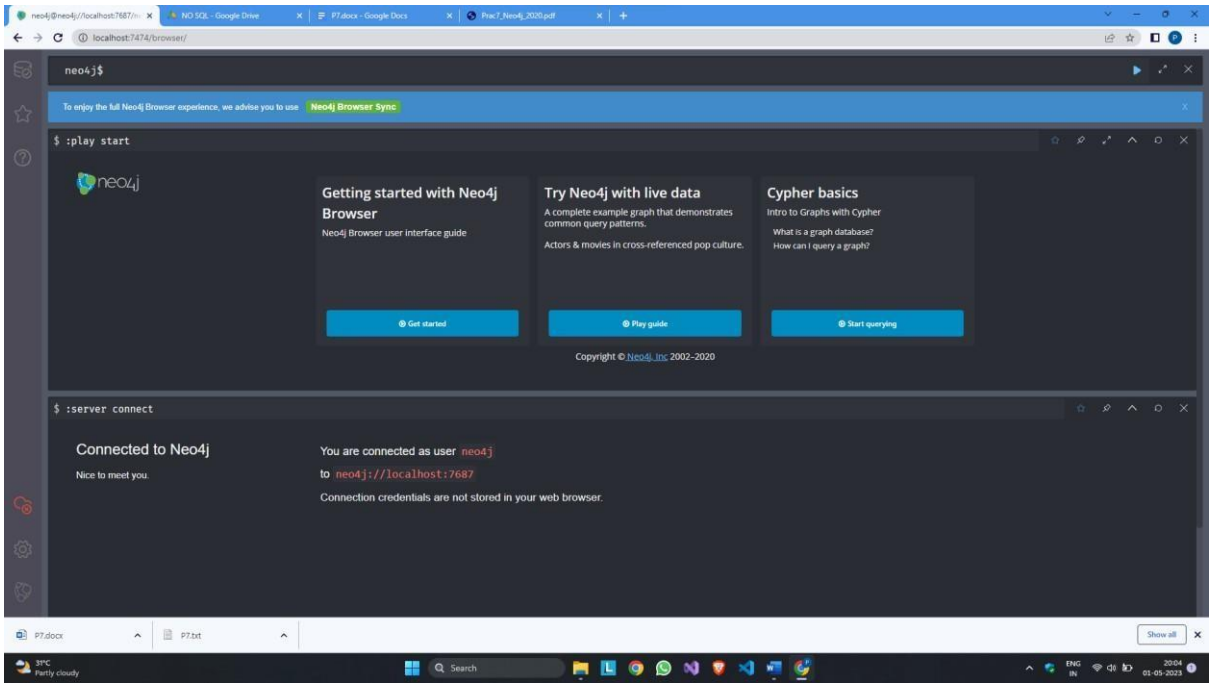
- Neo4j can be more complex to set up and manage compared to traditional relational databases.
- It may not be suitable for applications that require heavy write operations.

Installation:

- set NEO4J_HOME=C:\neo4j-community-3.5.15
- set PATH=C:\neo4j-community-3.5.15\bin;%PATH%
- set JAVA_HOME= C:\Program Files\Java\jdk-11.0.10
- set PATH= C:\Program Files\Java\jdk-11.0.10\bin;%PATH%
- neo4j.bat install-service
- neo4j.bat start

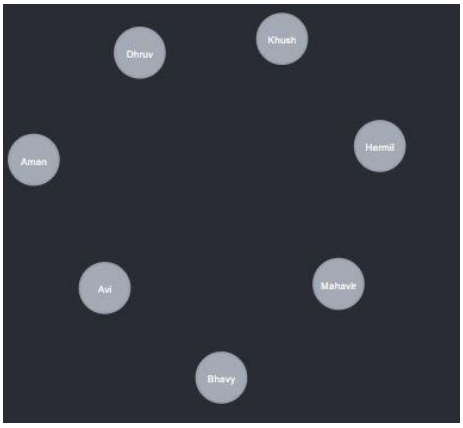
```
C:\Windows>cd..

C:\>set NEO4J_HOME=C:\neo4j-community-4.2.3

C:\>set PATH=C:\neo4j-community-4.2.3\bin;%PATH%

C:\>set Java_HOME=C:\Program Files\Java\jdk-11.0.10

C:\>set PATH=C:\Program Files\Java\jdk-11.0.10\bin;%PATH%

C:\>neo4j.bat
Usage: neo4j { console | start | stop | restart | status | install-service | uninstall-service | update-service } < -Verbose >

C:\>neo4j.bat start
Service start failed - service 'neo4j' not found

C:\>neo4j.bat install-service
Neo4j service installed

C:\>neo4j.bat start
Neo4j service started

C:\>_
```

Type in browser: **http://localhost:7474**



1. **Create 7 nodes in people label having properties name, department, address, age.**

   Query:
   CREATE (n1:People_068 {Name:"Bhavy",Dept:"CE", Addr:"Vijapur", Age:19}),
   (n2:People_068 {Name:"Mahavir",Dept:"CE", Addr:"Unava", Age:20}),
   (n3:People_068 {Name:"Hermil",Dept:"CE", Addr:"Gandhinagar", Age:18}),
   (n4:People_068 {Name:"Khush",Dept:"CE", Addr:"Mehsana", Age:22}),
   (n5:People_068 {Name:"Dhruv",Dept:"CE", Addr:"Unava", Age:19}),
   (n6:People_068 {Name:"Aman",Dept:"IT", Addr:"Mehsana", Age:20}),
   (n7:People_068 {Name:"Avi",Dept:"IT", Addr:"Gandhinagar", Age:20})
   RETURN n1,n2,n3,n4,n5,n6,n7 **Output:**



2. **Create 7 nodes in film label having properties title, releasedyear,directorname.**

   Query:

CREATE (n1:Film_068 {title:"Lagaan",RelYear:2001 , Director:"Ashutosh Gowariker"}),

(n2:Film_068 {title:"3 Idiots",RelYear:2009 , Director:"Rajkumar Hirani"}),

(n3:Film_068 {title:"Bajirao Mastani",RelYear:2015 , Director:"Sanjay Leela Bhansali"}),

(n4:Film_068 {title:"Dangal",RelYear:2016 , Director:"Nitesh Tiwari"}),

(n5:Film_068 {title:"Taare Zameen Par",RelYear:2007 , Director:"Aamir Khan"}),

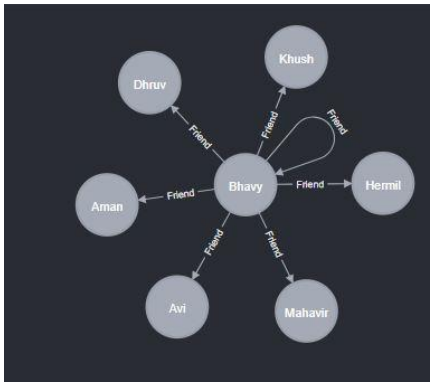
(n6:Film_068 {title:"Chhichhore",RelYear:2019 , Director:"Nitesh Tiwari"}),

(n7:Film_068 {title:"Munna Bhai M.B.B.S.",RelYear:2003 , Director:"Rajkumar Hirani"})

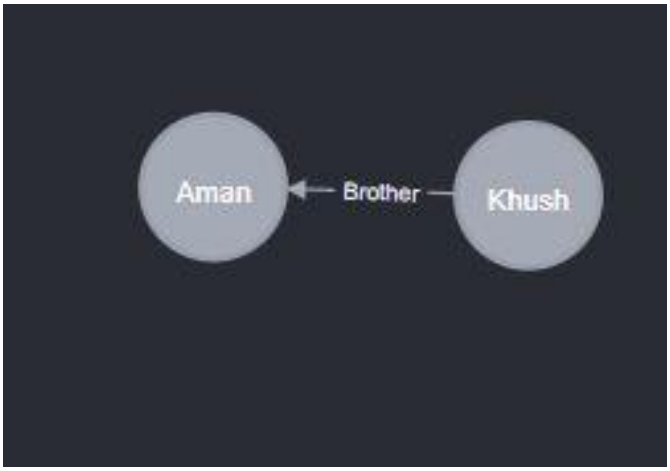
RETURN n1,n2,n3,n4,n5,n6,n7 **Output:**

3. **Create relationships called friend, brother, and favourite nodes.9**

Query: MATCH(n:People_068 {Name:"Bhavy"}),(m:People_068) CREATE (n)[r:Friend]->(m) RETURN n,m,r **Output:**

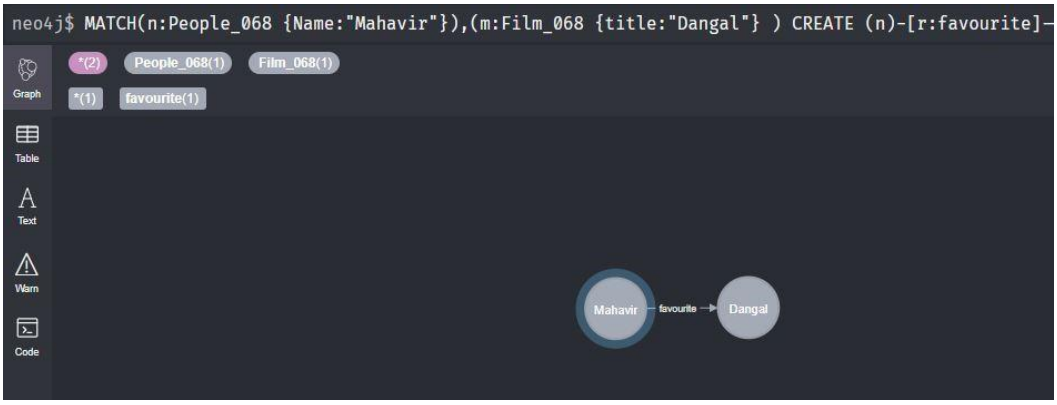


Query: MATCH(n:People_068 {Name:"Khush"}),(m:People_068 {Name:"Aman"}) CREATE (n)-[r:Brother]->(m) RETURN n,m,r **Output:**



Query:

Output:

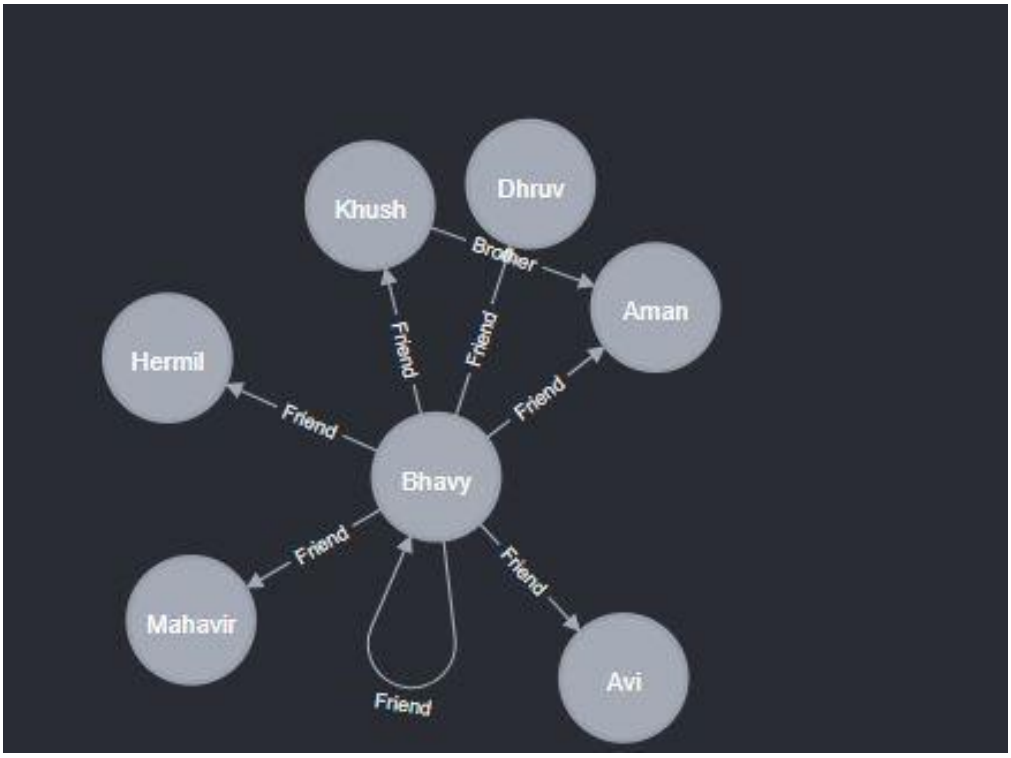


4. **What are the favourite movies of Mahavir.**

Query: MATCH(n:People_068 {Name:"Mahavir"})-[r:Favourite]->(m:Film_068) RETURN m,r,n **Output:**

5. **Who are friends of Bhavy**

Query: MATCH(n:People_068 {Name:"Bhavy"})-[r:Friend]>(m:People_068) RETURN m,r,n **Output:**

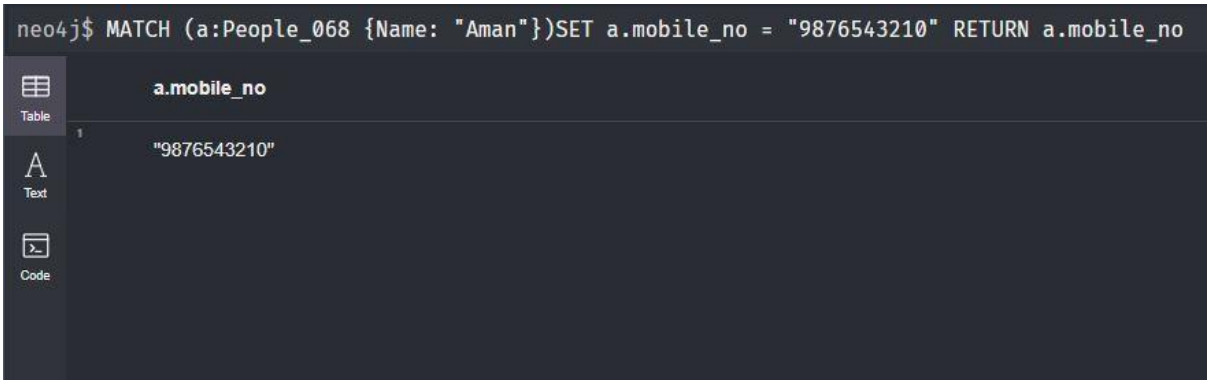


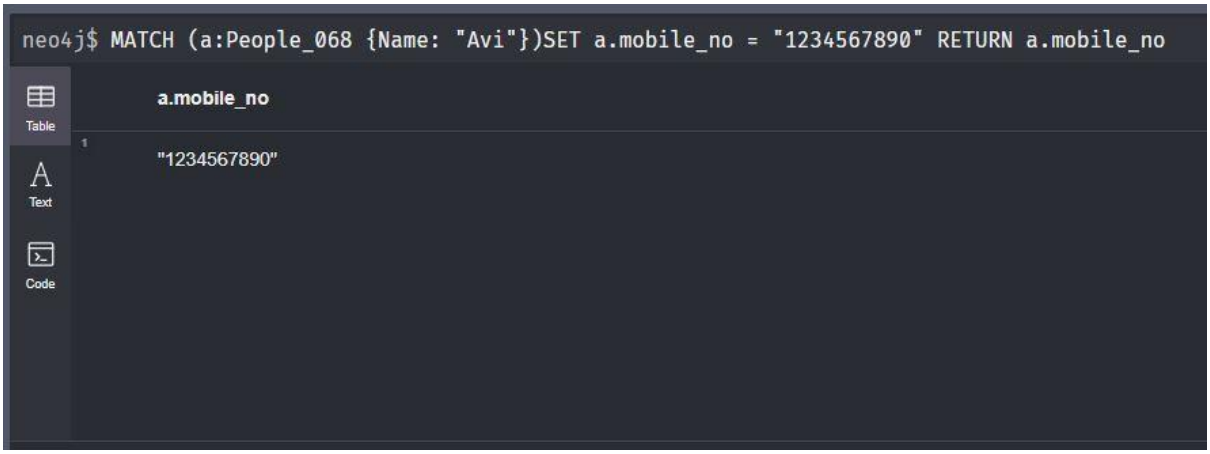
6. **Add a property called mobile_no for person Aman and Avi Query:**

MATCH (a:People_068 {Name: "Aman"})SET a.mobile_no = "9876543210"
RETURN a.mobile_no

MATCH (a:People_068 {Name: "Avi"})SET a.mobile_no = "1234567890"

RETURN a.mobile_no **Output:**

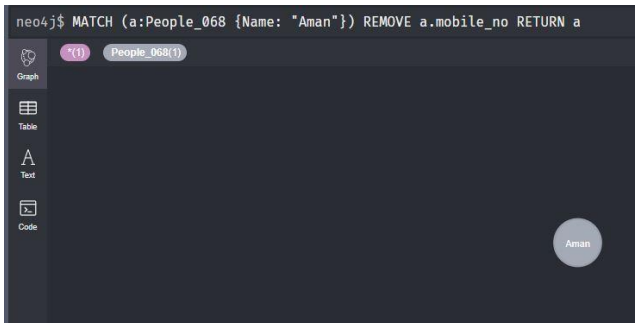




7. **Remove mobile _ no property for person Aman**

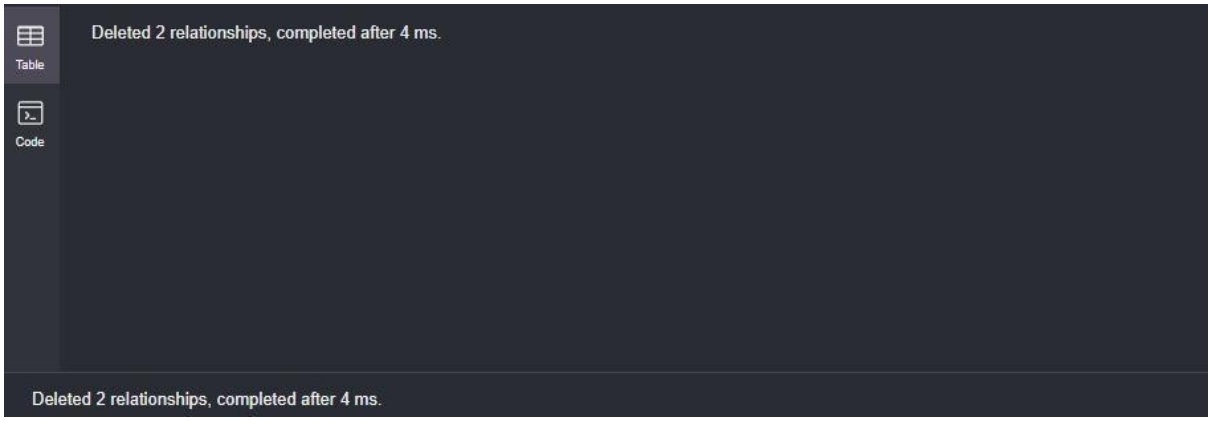
Query:MATCH (a:People_068 {Name: "Aman"}) REMOVE a.mobile_no RETURN a

Output:



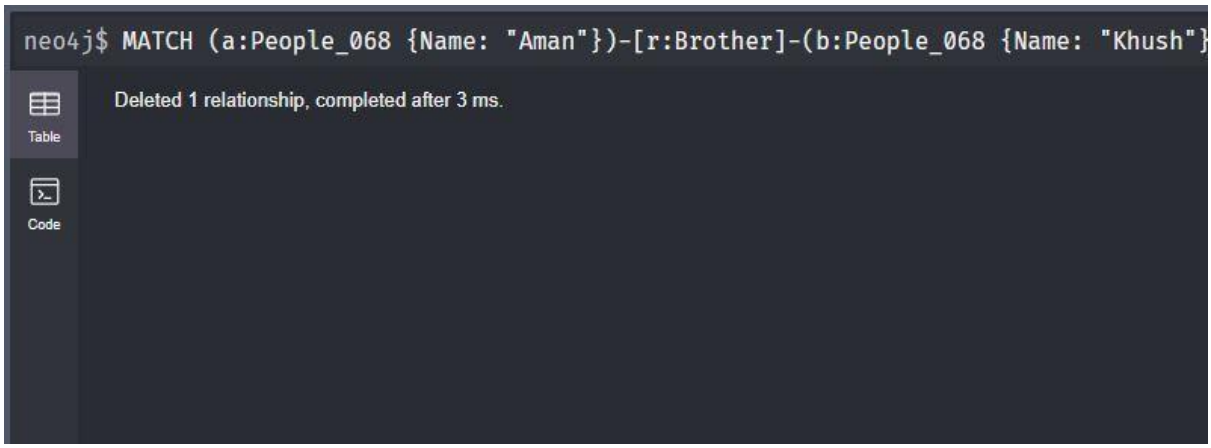
8. **Delete a relationship called brother**

Query: MATCH (a:People_068 {Name: "Aman"})-[r:Brother]-(b:People_068 {Name: "Khush"}) DELETE r **Output:**



9. **Delete a node having age 25**

Query: MATCH (n:People_068 {Age:25})-[r]-(m:People_068) DELETE r DELETE n **Output:**



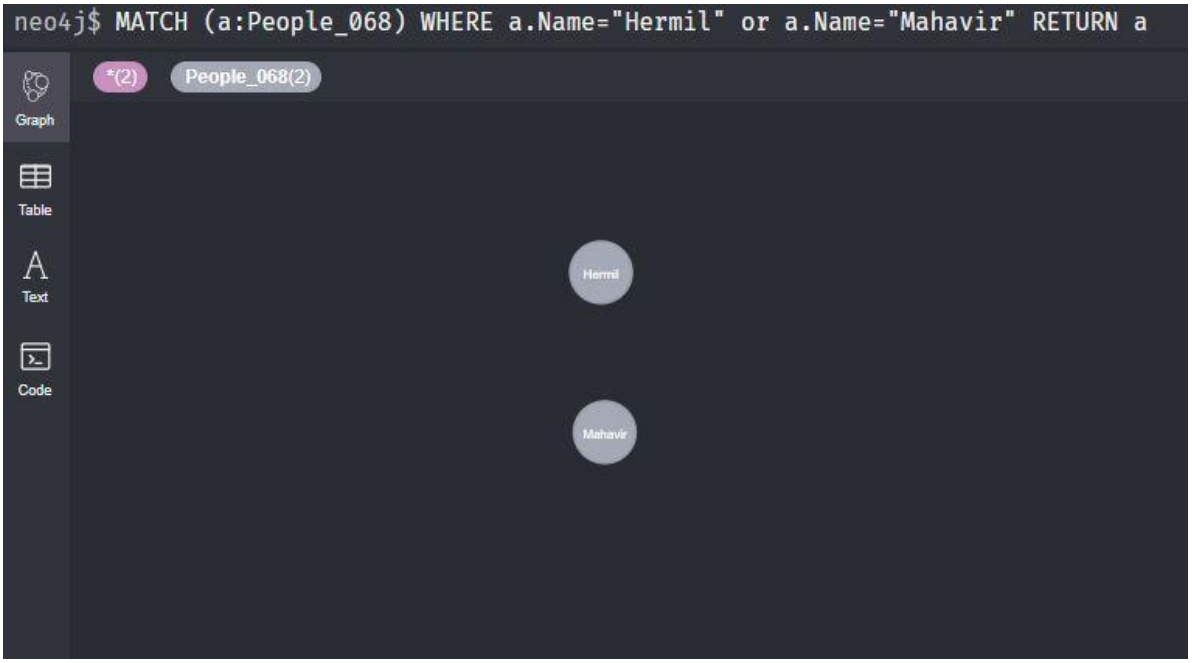
10. **Display details of node whose name is Dhruv and age is 19 Query:** MATCH

MATCH(n:People_068 {Name:"Dhruv",Age:19}) RETURN n **Output:**

11. **display details of node having name Mahavir or Aman**

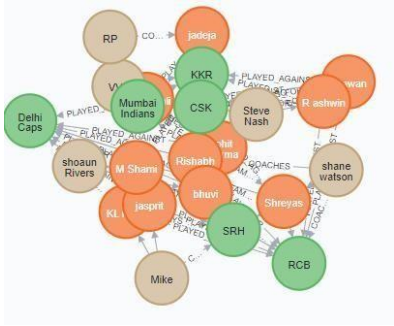
Query: MATCH (a:People_068) WHERE a.Name="Hermil" or a.Name="Mahavir" RETURN a

Output:



Exercise 2:

MATCH (n) RETURN n

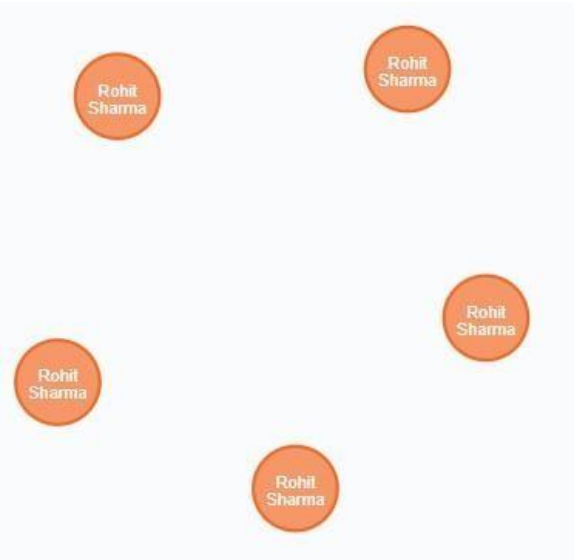


MATCH (player:PLAYER) RETURN player.name, player.height

	player.name	player.height
1	"Virat Kohli"	170
2	"Rohit Sharma"	171
3	"Ajinkya rahane"	169
4	"jadeja"	183

Q1))// Nodes where name is Rohit Sharma //

MATCH (player:PLAYER) WHERE player.name = "Rohit Sharma" RETURN player

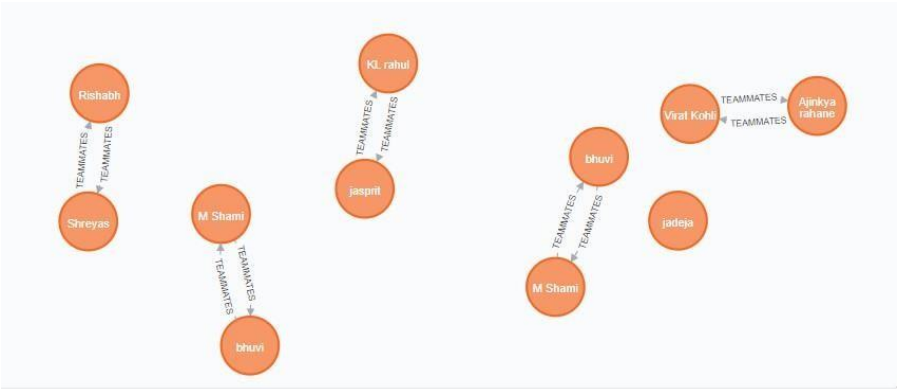


Q2)// Nodes where name is Rohit Sharma //

MATCH (player:PLAYER {name: "Rohit Sharma"}) RETURN player

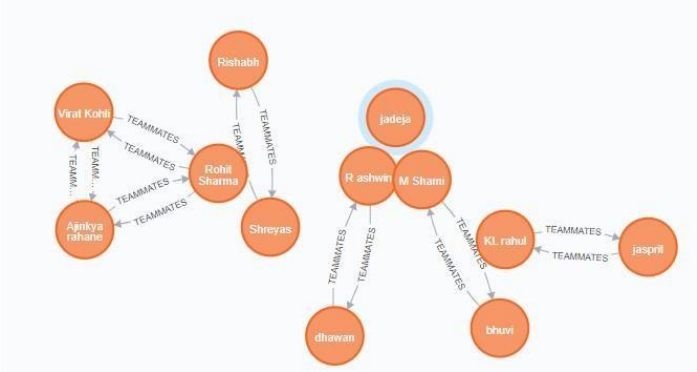
Q3)// Nodes where name is not Rohit Sharma

MATCH (player:PLAYER) WHERE player.name <> "Rohit Sharma"RETURN player



Nodes where height is greater than or equal to 2

Match(n:PLAYER) WHERE n.height>=2 return n



Q5// Nodes where height is less than 2

Match(n:PLAYER) WHERE n.height<2 return n

(no changes, no records)

Q6// Nodes with a BMI larger than 25

Match(n:PLAYER) where n.BMI>25 return n

```
match (n:PLAYER) where n.BMI>25 return n
```

(no changes, no records)

Q7// Nodes with a BMI not larger than 25

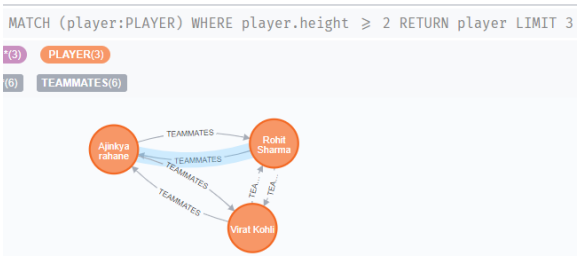
Match(n:PLAYER) where n.BMI<= 25 return n

```
match (n:PLAYER) where n.BMI ≤ 25 return n
```

(no changes, no records)

// Limit

MATCH (player:PLAYER) WHERE player.height >= 2RETURN player LIMIT 3



// Skip

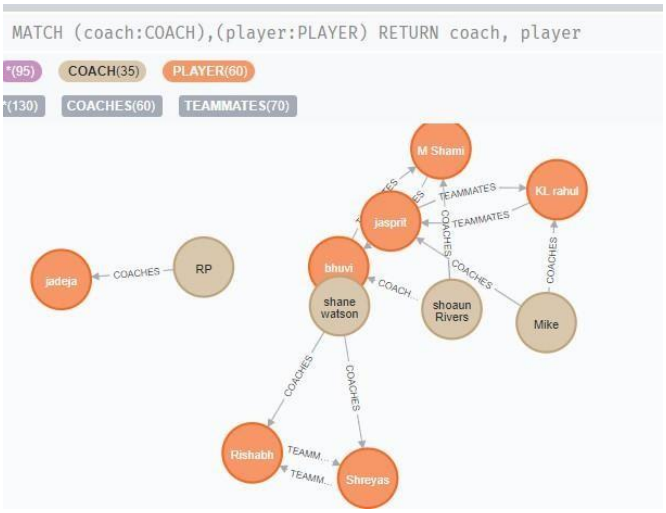
MATCH (player:PLAYER) WHERE player.height >= 2RETURN playerSKIP 1 LIMIT 3

// Orderby

MATCH (player:PLAYER) WHERE player.height >= 2RETURN playerSKIP 1 ORDER BY player.height DESC LIMIT 3

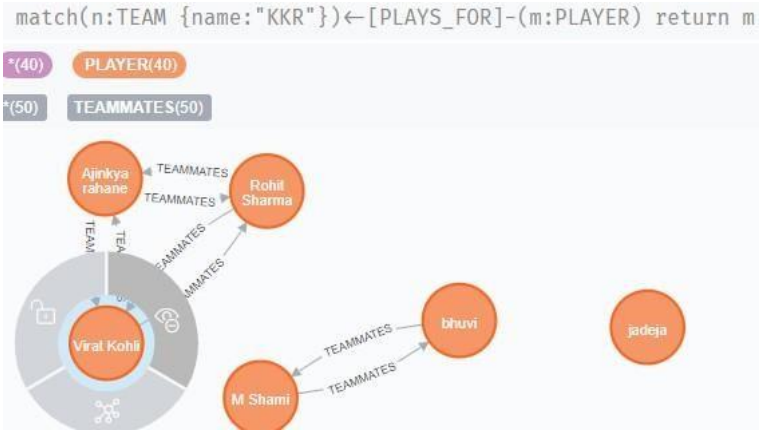
// Query for multiple nodes

MATCH (coach:COACH), (player:PLAYER) RETURN coach, player



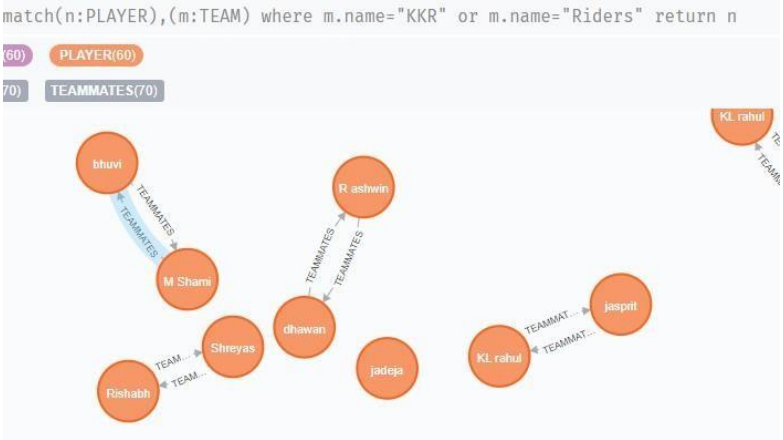
// GET ALL KKR PLAYERS //

Match(n:TEAM {name:”KKR”}<-[P LAYS_FOR]-(m:PLAYER) return m

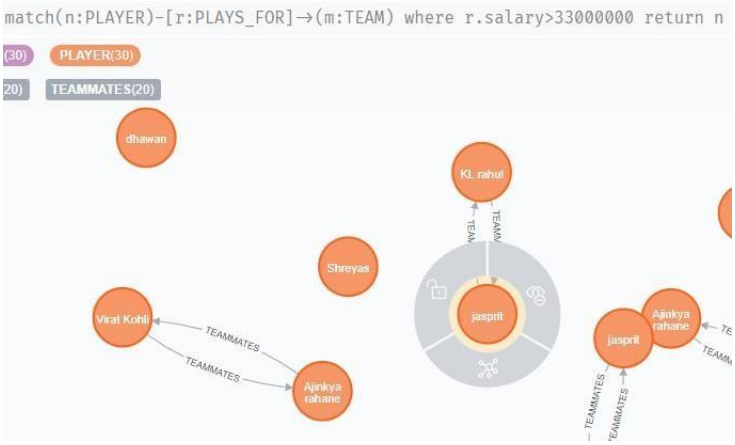


// GET ALL KKR OR royal PLAYERS //

Match(N:PLAYER),(m:TEAM) where m.name=”KKR” or m.name=”Riders” return n



// GET ALL PLAYERS THAT MAKE MORE THE 35M //
Mathc(n:PLAYER)[r:PLAYS_FOR]->(m:TEAM) where r.salary>33000000 return n



// GET ALL OF viratS TEAMMATES THAT MAKE MORE THAN 40M //

Mathc(n:PLAYER)-[r:TEAMMATES]->(m:PLAYER) where n.name="virat" return m.salary>40000000

```
match(n:PLAYER)-[r:TEAMMATES]→(m:PLAYER) where n.name="virat" return m.salary>40000000
```

(no changes, no records)

// GET PLAYERS AND NUMBER OF GAMES PLAYED // match(n:PLAYER) OPTIONAL MATCH (n)-[r:PLAYED_AGAINST]->() RETURN n.name,count(r) as num_games

```
neo4j$ match(n:PLAYER) OPTIONAL MATCH (n)-[r:PLAYED_AGAINST]→() RETURN n.name,count(r) as num_games
```

Table | Text | Code

	n.name	num_games
1	"Virat Kohli"	4
2	"Rohit Sharma"	4
3	"Ajinkya rahane"	4

// GET PLAYERS AND runs PER GAME // match(n:PLAYER) OPTIONAL MATCH (n)-[r:PLAYED_AGAINST]->() RETURN n.name,sum(r.runs) as num_runs

```
neo4j$ match(n:PLAYER) OPTIONAL MATCH (n)-[r:PLAYED_AGAINST]→() RETURN n.name,sum(r.runs) as num_runs
```

	n.name	num_runs
1	"Virat Kohli"	972
2	"Rohit Sharma"	512
3	"Ajinkya rahane"	2077
4	"jadeja"	4879

// GET HIGHEST SCORING PLAYER IN THE Riders // match(n:TEAM{name:"KKR"})optional match (n)<-[r:PLAYS_FOR]-(m:PLAYER)[p:PLAYED_AGAINST]->() RETURN m.name,sum(p.runs) as num_runs order by num_runs DESC LIMIT 1

```
neo4j$ match(n:TEAM{name:"KKR"})optional match (n)←[r:PLAYS_FOR]-(m:PLAYER)-[p:PLAYED_AGAINST]→() RETURN m.name,sum(p.r…
```

	m.name	num_runs
1	"Ajinkya rahane"	2077

// Delete relationship

MATCH(n:PLAYER)-[r:TEAMMATES]->(m:PLAYER) delete r

```
neo4j$ MATCH(n:PLAYER)-[r:TEAMMATES]→(m:PLAYER) delete r
```

Deleted 14 relationships, completed after 6 ms.

MATCH(n:COACH)-[r:COACHES]->(m) delete r

```
neo4j$ MATCH(n:COACH)-[r:COACHES]→(m) delete r
```

Deleted 12 relationships, completed after 2 ms.

MATCH(n:COACH)-[r:COACHES_FOR]->(m) delete r

```
neo4j$ MATCH(n:COACH)-[r:COACHES_FOR]→(m) delete r
```

Deleted 6 relationships, completed after 2 ms.

MATCH(n:PLAYER)-[r:PLAYS_FOR]->(m) delete r

```
neo4j$ MATCH(n:PLAYER)-[r:PLAYS_FOR]→(m) delete r
```

Table

Deleted 12 relationships, completed after 2 ms.

MATCH(n:PLAYER)-[r:PLAYED_AGAINST]->(m) delete r

```
neo4j$ MATCH(n:PLAYER)-[r:PLAYED_AGAINST]→(m) delete r
```

Table

Deleted 37 relationships, completed after 3 ms.

// Delete Node

MATCH(n:PLAYER) delete n

```
neo4j$ MATCH(n:PLAYER) delete n
```

Table

Deleted 12 nodes, completed after 2 ms.

MATCH(n:TEAM) delete n

```
neo4j$ MATCH(n:TEAM) delete n
```

Table

Deleted 6 nodes, completed after 1 ms.

MATCH(n:COACH) delete n

```
neo4j$ MATCH(n:COACH) delete n\
```

Deleted 7 nodes, completed after 3 ms.

CREATE (virat:PLAYER:COACH:GENERAL_MANAGER { name: "Rohit Sharma", height: 2.01 })

```
neo4j$ CREATE (virat:PLAYER:COACH:GENERAL_MANAGER { name: "Rohit Sharma", height: 2.01 })
```

Table

Added 3 labels, created 1 node, set 2 properties, completed after 16 ms.

CREATE (virat:PLAYER:COACH:GENERAL_MANAGER { name: "Rohit Sharma", height: 2.01 }) - [:PLAYS_FOR {salary: 40000000}] -> (:TEAM {name: "KKR"})

```
neo4j$ CREATE (virat:PLAYER:COACH:GENERAL_MANAGER { name: "Rohit Sharma", height: 2.01 }) - [:PLAYS_FOR {salary: 40000000}] …
```

Table

Added 4 labels, created 2 nodes, set 4 properties, created 1 relationship, completed after 2 ms.

CREATE (virat:PLAYER:COACH:GENERAL_MANAGER { name: "Rohit Sharma", height: 2.01 })

```
neo4j$ CREATE (virat:PLAYER:COACH:GENERAL_MANAGER { name: "Rohit Sharma", height: 2.01 })
```

Added 3 labels, created 1 node, set 2 properties, completed after 1 ms.

CREATE (:TEAM {name: "KKR"})

```
neo4j$ CREATE (:TEAM {name: "KKR"})
```

Table

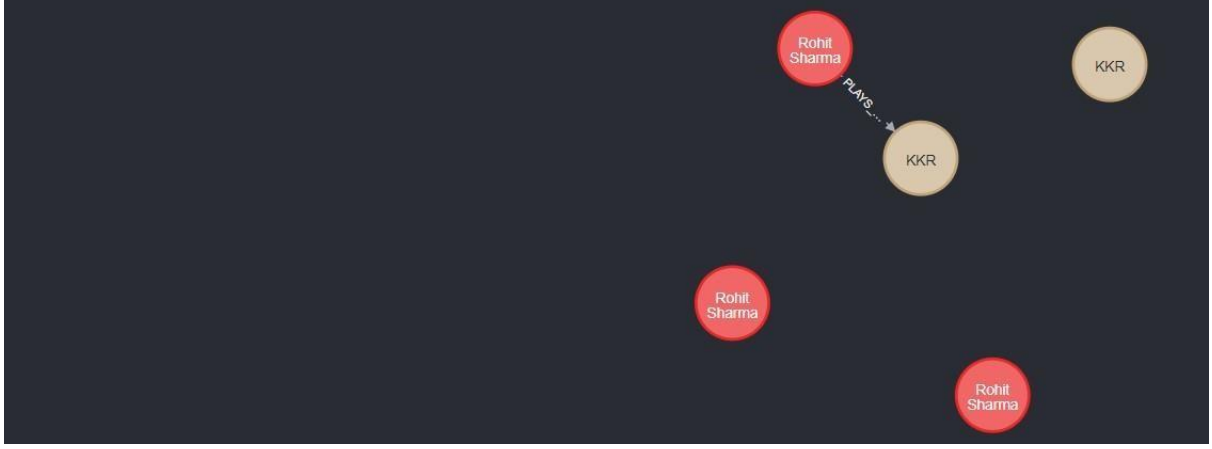
Added 1 label, created 1 node, set 1 property, completed after 1 ms.

MATCH (virat:PLAYER {name: "Rohit Sharma"}), (Riders:TEAM {name: "KKR"}) return virat,Riders

```
MATCH (virat:PLAYER {name: "Rohit Sharma"}), (Riders:TEAM {name: "KKR"}) return virat,Riders
```

*(11) COACH(3) GENERAL_MANAGER(3) PLAYER(3) TEAM(2)

*(1) PLAYS_FOR(1)



CREATE (virat) - [:PLAYS_FOR] -> (Riders)

```
neo4j$ CREATE (virat) - [:PLAYS_FOR] → (Riders)
```

Created 2 nodes, created 1 relationship, completed after 2 ms.

MATCH (n) WHERE ID(n) = 3 SET n.age = 24, n.height = 2.02 RETURN n

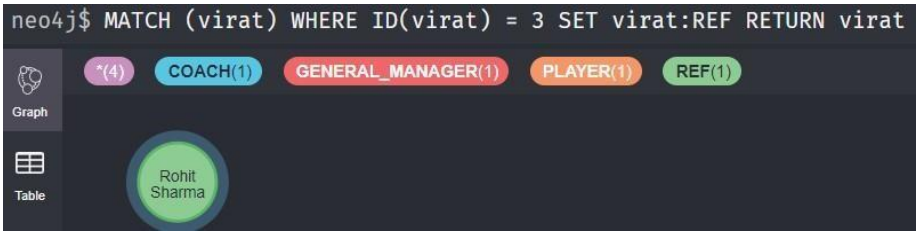
```
neo4j$ MATCH (n) WHERE ID(n) = 3 SET n.age = 24, n.height = 2.02 RETURN n
```

Graph

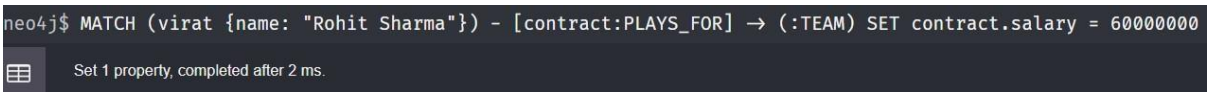
*(3) COACH(1) GENERAL_MANAGER(1) PLAYER(1)

Table

MATCH (virat) WHERE ID(virat) = 3 SET virat:REF RETURN virat



MATCH (virat {name: "Rohit Sharma"}) - [contract:PLAYS_FOR] -> (:TEAM) SET contract.salary = 60000000



MATCH (virat) WHERE ID(virat) = 3 REMOVE virat:REF RETURN virat

MATCH (virat) WHERE ID(virat) = 3 REMOVE virat.age RETURN virat