

# Lab - 10

NEO4J

Name	Anirudh
SRN	PES1UG23CS077
Sec	B
Date	31-10-2025

## Part A: Create Nodes:

1. Create Student Nodes with name, age and major. We have Alice, 21, CSE & Bob, 22, ECE and Charlie, 20, CSE.

The screenshot shows the Neo4j browser interface. At the top, a command line window displays the query: `neo4j$ MATCH (n:Student) RETURN n LIMIT 25;`. Below this, a results overview panel indicates there are 3 nodes and 3 student records. The main area shows three circular nodes representing students: Alice (bottom center), Bob (top right), and Charlie (top left). The interface includes tabs for Graph, Table, and RAW, and various navigation and search tools.

2.Create Professor Nodes, Dr. Smith for the CSE department and Dr. Jones for the ECE department

```
neo4j$ MATCH (n:Professor) RETURN n LIMIT 25;
```

The screenshot shows the Neo4j browser interface. At the top, there is a command line input field with the query: "MATCH (n:Professor) RETURN n LIMIT 25;". Below the input field are three tabs: "Graph" (which is selected), "Table", and "RAW". To the right of the tabs are several icons for saving, copying, and navigating. The main area displays two circular nodes representing professors. The top node is labeled "Dr. Jones" and the bottom node is labeled "Dr. Smith". On the right side of the screen, there is a "Results overview" panel. It shows a summary of the results: "Nodes (2)", "\* (2)", and "Professor (2)". Below this panel, a message indicates: "Started streaming 2 records after 38 ms and completed after 44 ms."

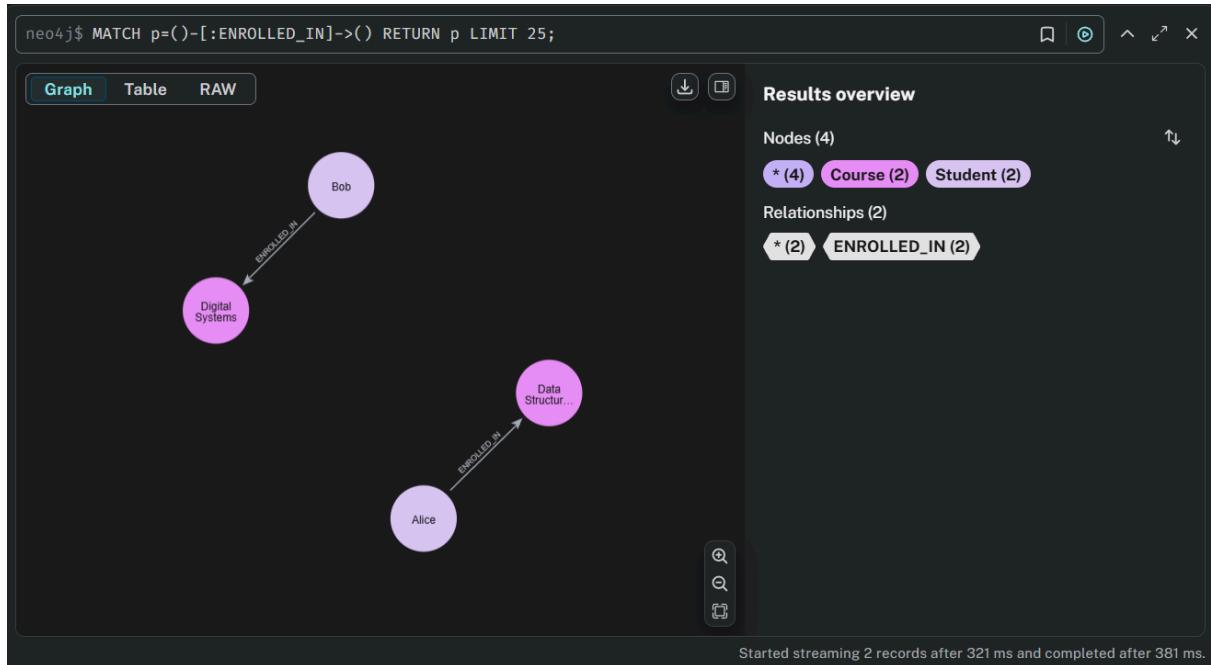
3.Create Course Nodes (CODE: CS101, DATA STRUCTURES & CODE: EC202, DIGITAL SYSTEMS)

```
neo4j$ MATCH (n:Course) RETURN n LIMIT 25;
```

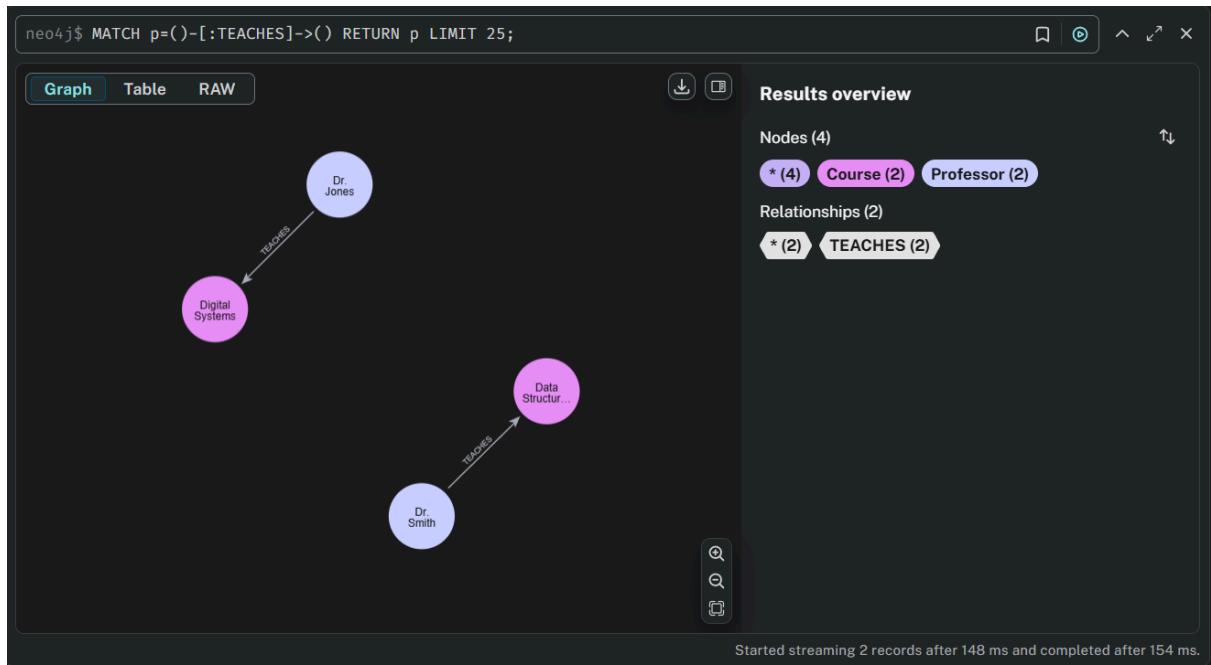
The screenshot shows the Neo4j browser interface. At the top, there is a command line input field with the query: "MATCH (n:Course) RETURN n LIMIT 25;". Below the input field are three tabs: "Graph" (selected), "Table", and "RAW". To the right of the tabs are icons for saving, copying, and navigating. The main area displays two circular nodes representing courses. The top node is labeled "Digital Systems" and the bottom node is labeled "Data Structur...". On the right side of the screen, there is a "Results overview" panel. It shows a summary of the results: "Nodes (2)", "\* (2)", and "Course (2)". Below this panel, a message indicates: "Started streaming 2 records after 7 ms and completed after 10 ms."

## Part B: Create Relationships

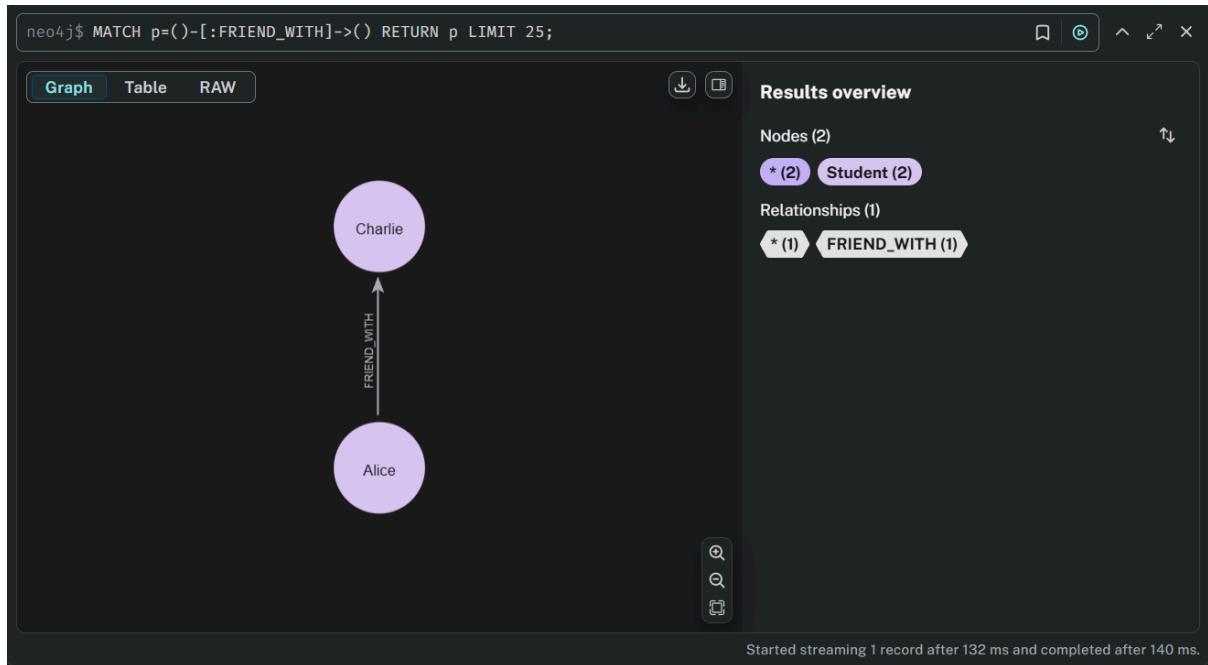
4. Show that Alice has enrolled into “CS101” and Bob has enrolled into “EC202”



5. Show that professor Dr. Smith teaches “CS101” and Dr. Jones teaches “EC202”



## 6.Create friendship between Alice and Charlie



## Created Nodes and relationships:

The screenshot shows the Neo4j browser history panel. It lists several database operations with their results and execution times. The operations include creating nodes for Alice, Charlie, Dr. Jones, Dr. Smith, and Bob, and creating relationships such as FRIEND\_WITH, TEACHES, and ENROLLED\_IN. The history shows the creation of 9 nodes and 9 relationships.

- `neo4j$ MATCH (a:Student {name: "Alice"}), (c:Student {name: "Charlie"}) CREATE (a)-[:FRIEND_WITH]->(c);` Completed after 137 ms
- `neo4j$ MATCH (j:Professor {name: "Dr. Jones"}), (ec:Course {code: "EC202"}) CREATE (j)-[:TEACHES]->(ec);` Completed after 141 ms
- `neo4j$ MATCH (s:Professor {name: "Dr. Smith"}), (cs:Course {code: "CS101"}) CREATE (s)-[:TEACHES]->(cs);` Completed after 147 ms
- `neo4j$ MATCH (b:Student {name: "Bob"}), (ec:Course {code: "EC202"}) CREATE (b)-[:ENROLLED_IN]->(ec);` Completed after 149 ms
- `neo4j$ MATCH (a:Student {name: "Alice"}), (cs:Course {code: "CS101"}) CREATE (a)-[:ENROLLED_IN]->(cs);` Completed after 311 ms
- `neo4j$ CREATE (cs:Course {code: "CS101", title: "Data Structures"}), (ec:Course {code: "EC202", title: "Digital Systems"});` Completed after 98 ms
- `neo4j$ CREATE (s:Professor {name: "Dr. Smith", department: "CSE"}), (j:Professor {name: "Dr. Jones", department: "ECE"});` Completed after 90 ms
- `neo4j$ CREATE (a:Student {name: "Alice", age: 21, major: "CSE"}), (b:Student {name: "Bob", age: 22, major: "ECE"}), (c:Student {name: "Charlie", age: 20, major: "CSE"});` Completed after 184 ms

## Part C: Query the Graph:

### 7. List All Students:

```
neo4j$ MATCH (s:Student) RETURN s.name AS Student, s.age AS Age, s.major AS Major;
```

Table RAW

Student	Age	Major
1 "Alice"	21	"CSE"
2 "Bob"	22	"ECE"
3 "Charlie"	20	"CSE"

Started streaming 3 records after 18 ms and completed after 36 ms.

### 8. Find Courses Taught by Dr. Smith

```
neo4j$ MATCH (p:Professor {name: "Dr. Smith"})-[:TEACHES]->(c:Course) RETURN c.title AS CourseTaughtByDrSmith
```

Table RAW

CourseTaughtByDrSmith
1 "Data Structures"

Started streaming 1 record after 1,850 ms and completed after 1,967 ms.

### 9. Find Friends of Charlie

```
neo4j$ MATCH (s:Student {name: "Charlie"})-[:FRIEND_WITH]-(friend) RETURN friend.name AS FriendsOfCharlie
```

Table RAW

FriendsOfCharlie
1 "Alice"

Started streaming 1 record after 268 ms and completed after 273 ms.

### 10. List All Students in the Same Course

```
neo4j$ MATCH (s1:Student)-[:ENROLLED_IN]->(c:Course)<-[ENROLLED_IN]-(s2:Student) WHERE s1.name < s2.name
```

No changes, no records

Completed after 786 ms

## 11.Find Professors Who Teach Alice's Courses

```
neo4j$ MATCH (a:Student {name: "Alice"})-[:ENROLLED_IN]->(c:Course)<-[TEACHES]-(p:Professor) RETURN p.name  
^ ↴ ×  
Table RAW  
Professor CourseTaughtToAlice  
"Dr. Smith" "Data Structure"  
s"  
Started streaming 1 record after 302 ms and completed after 306 ms
```

## 12.Find Students Who Are Friends and Enrolled in the Same Course

```
neo4j$ MATCH (s1:Student)-[:FRIEND_WITH]-(s2:Student), (s1)-[:ENROLLED_IN]->(c:Course), (s2)-[:ENROLLED_IN]->(c)  
No changes, no records  
Completed after 537 ms
```

## 13.Find Courses with More Than One Student Enrolled

```
neo4j$ MATCH (s:Student)-[:ENROLLED_IN]->(c:Course) WITH c, COUNT(s) AS num_students WHERE num_students > 1  
No changes, no records  
Completed after 530 ms
```

## 14.Count how many students each professor teaches.

```
neo4j$ MATCH (p:Professor)-[:TEACHES]->(c:Course)<-[ENROLLED_IN]-(s:Student) RETURN p.name AS Professor, COUNT(s) AS TotalStudents  
^ ↴ ×  
Table RAW  
Professor TotalStudents  
"Dr. Smith" 1  
"Dr. Jones" 1  
Started streaming 2 records after 527 ms and completed after 529 ms.
```