

Homework 8

1. Create 10 students (Alex, Andrew, Arnold, Ally, Bob, Brad, Bran, Chris, Charles, David) nodes with names, ages, and addresses. Display all nodes

```
CREATE (Alex:Student {name: 'Alex', age: 25, address: '123 Main
Street, Apt 4B, Anytown'}),
      (Andrew:Student {name: 'Andrew', age: 22, address: '456 Elm
Street, Apt 3A, Othertown'}),
      (Arnold:Student {name: 'Arnold', age: 21, address: '789 Oak
Street, Apt 2C, Another town'}),
      (Ally:Student {name: 'Ally', age: 23, address: '321 Pine
Street, Apt 1D, Differenttown'}),
      (Bob:Student {name: 'Bob', age: 24, address: '654 Birch
Street, Apt 5E, Newtown'}),
      (Brad:Student {name: 'Brad', age: 26, address: '987 Cedar
Street, Apt 6F, Nearbytown'}),
      (Bran:Student {name: 'Bran', age: 20, address: '234 Maple
Street, Apt 7G, Nearbytown'}),
      (Chris:Student {name: 'Chris', age: 27, address: '567 Walnut
Street, Apt 8H, Nearbytown'}),
      (Charles:Student {name: 'Charles', age: 22, address: '876
Redwood Street, Apt 9I, Nearbytown'}),
      (David:Student {name: 'David', age: 28, address: '123 Spruce
Street, Apt 10J, Nearbytown'});
```

The screenshot shows the Neo4j Desktop interface. On the left is a sidebar with 'Database Information'. The 'Use database' section shows 'neo4j' selected. The 'Node labels' section shows a red 'Student' label. The 'Relationship types' section shows 'No relationships in database'. The 'Property keys' section shows 'address', 'age', and 'name'. The 'Connected as' section shows 'Username: neo4j', 'Roles: admin, PUBLIC', and 'Admin: :server user list'. The 'DBMS' section shows 'Cluster role: primary', 'Version: 5.12.0', 'Edition: Enterprise', 'Name: neo4j', 'Databases: :dbs', 'Information: :sysinfo', and 'Query List: :queries'. The main area is a query editor with a prompt 'neo4j\$'. Below the prompt, a query is entered: 'neo4j\$ create (Alex: Student {name: 'Alex', age: 25, address: '123 Main Street, Apt 4B, Anytown...'. The status bar at the bottom of the query editor says 'Added 10 labels, created 10 nodes, set 30 properties, completed after 21 ms.'

The screenshot shows the Neo4j Desktop interface with a graph visualization. The sidebar on the left has a 'Graph' icon selected. The main area displays a graph with 10 nodes, each labeled with a name: Alex, Andrew, Bob, Brenda, Charlie, David, Emily, Henry, Ivy, and Jack. The nodes are connected by lines representing relationships. The query editor at the top shows the query 'neo4j\$ match (s:Student) return s'. The status bar at the bottom of the query editor says 'Added 10 labels, created 10 nodes, set 30 properties, completed after 21 ms.'

2. Display min age, max-age, and average age among all students.

The image shows the Neo4j Desktop interface with two queries executed.

Query 1: `neo4j$ match (s: Student) return min(s.age), max(s.age), avg(s.age);`

min(s.age)	max(s.age)	avg(s.age)
22	45	31.400000000000002

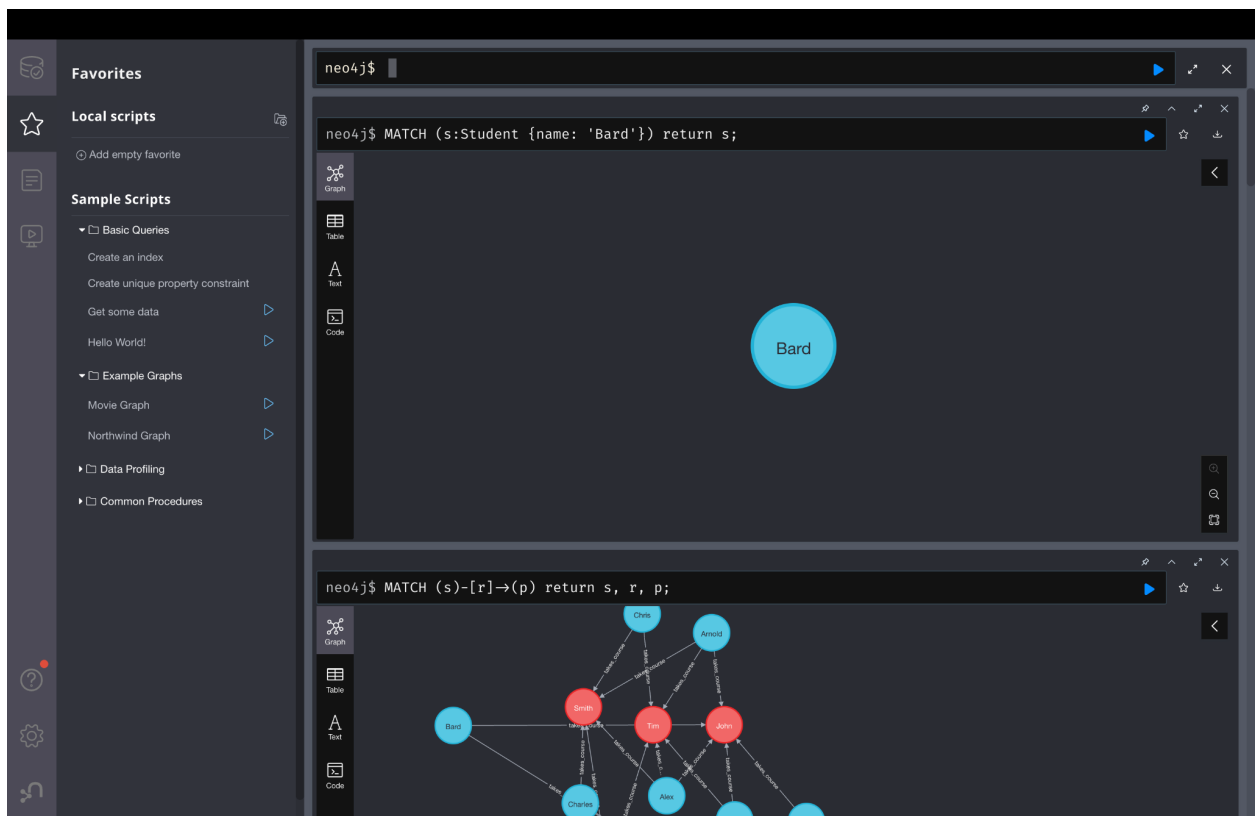
Started streaming 1 records after 6 ms and completed after 7 ms.

Query 2: `neo4j$ match (s:Student) return s`

The graph visualization shows several nodes representing students: Brad, Bard, Bob, Abby, Olivia, Charles, Aaron, Amy, and Andrew.

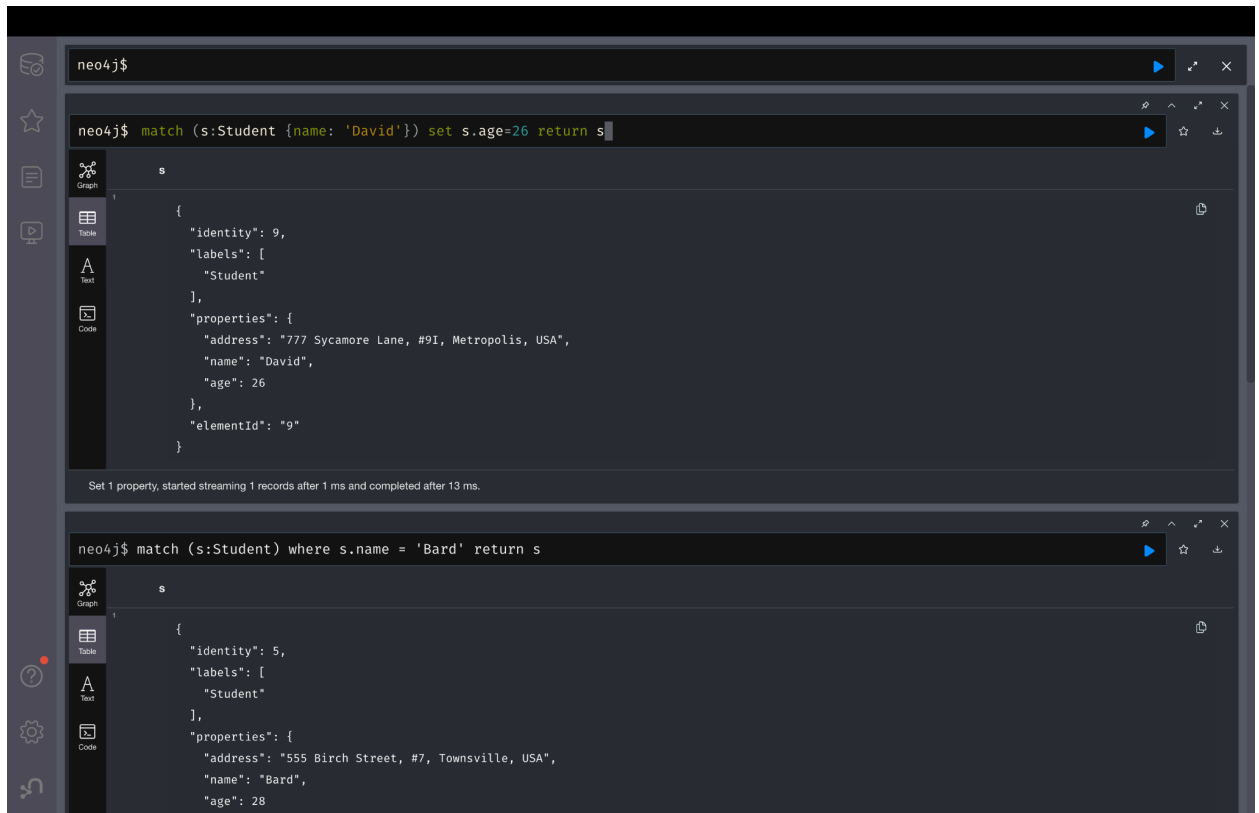
3. Display only Brad node.

```
MATCH (s:Student {name: 'Brad'}) return s;
```



4. Update age of David to be 26

```
match (s:Student {name: 'David'}) set s.age=26 return s;
```



5. Create 5 Professor (Smith, John, Taylor, James, Tim) nodes with name, and course title (201,202,203,204,205). Display all professor nodes.

```
create
(p1:Professor {name: 'Smith', course_title: 201}),
(p2:Professor {name: 'John', course_title: 202}),
(p3:Professor {name: 'Taylor', course_title: 203}),
(p4:Professor {name: 'James', course_title: 204}),
(p5:Professor {name: 'Tim', course_title: 205})
```

Database Information

Use database
neo4j

Node labels
Professor Student

Relationship types
No relationships in database

Property keys
address age course_title name

Connected as
Username: neo4j
Roles: admin, PUBLIC
Admin: :server user list
:server user add
Disconnect: :server disconnect

DBMS
Cluster role: primary
Version: 5.12.0
Edition: Enterprise
Name: neo4j
Databases: :dbs
Information: :sysinfo
Query List: :queries

neo4j\$

```
1 create
2   (p1:Professor {name: 'Smith', course_title: 201}),
3   (p2:Professor {name: 'John', course_title: 202}),
4   (p3:Professor {name: 'Taylor', course_title: 203}),
5   (p4:Professor {name: 'James', course_title: 204}),
6   (p5:Professor {name: 'Tim', course_title: 205})
```

Added 5 labels, created 5 nodes, set 10 properties, completed after 88 ms.

Favorites

Local scripts

Sample Scripts

Basic Queries

Create an index

Create unique property constraint

Get some data

Hello World!

Example Graphs

Movie Graph

Northwind Graph


Data Profiling

Common Procedures

neo4j\$

Created 3 relationships, completed after 2 ms.

```
neo4j$ match (p:Professor) return p;
```



```
neo4j$ MATCH (s:Student) return s;
```

6. Create 'takes_course' relationships (eg; **from:** Semester) between all students and any professor. (Make sure each student takes at least 3 courses) (Eg: Arnold takes_course John) (Arnold takes_course Tim)

MATCH

```
(s: Student {name: 'Bran'}),  
(p1: Professor {name: 'Tim'}),  
(p2: Professor {name: 'John'}),  
(p3: Professor {name: 'Smith'})
```

```
CREATE (s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)
```

MATCH

```
(s: Student {name: 'Chris'}),  
(p1: Professor {name: 'Tim'}),  
(p2: Professor {name: 'James'}),  
(p3: Professor {name: 'Smith'})
```

```
CREATE (s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)
```

MATCH

```
(s: Student {name: 'Charles'}),  
(p1: Professor {name: 'Taylor'}),  
(p2: Professor {name: 'James'}),  
(p3: Professor {name: 'Smith'})
```

```
CREATE (s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)
```

MATCH

```
(s: Student {name: 'David'}),  
(p1: Professor {name: 'James'}),  
(p2: Professor {name: 'John'}),  
(p3: Professor {name: 'Taylor'})
```

```
CREATE (s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)
```

MATCH

```
(s: Student {name: 'Alex'}),  
(p1:Professor {name: 'Tim'}),  
(p2:Professor {name: 'John'}),  
(p3:Professor {name: 'Smith'})
```

CREATE

```
(s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)
```

MATCH

```
(s: Student {name: 'Andrew'}),  
(p1:Professor {name: 'Tim'}),  
(p2:Professor {name: 'Taylor'}),  
(p3:Professor {name: 'Smith'})
```

CREATE

```
(s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)
```

MATCH

```
(s: Student {name: 'Arnold'}),  
(p1:Professor {name: 'Tim'}),  
(p2:Professor {name: 'John'}),  
(p3:Professor {name: 'Smith'})
```

CREATE

```
(s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)
```

MATCH

```
(s: Student {name: 'Ally'}),  
(p1:Professor {name: 'Tim'}),  
(p2:Professor {name: 'John'}),  
(p3:Professor {name: 'James'})
```

CREATE

```
(s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)
```

MATCH

```
(s: Student {name: 'Bob'}),  
(p1:Professor {name: 'James'})
```



```

(p2:Professor {name: 'John'}),
(p3:Professor {name: 'Smith'})
CREATE
(s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)

MATCH
(s: Student {name: 'Bran'}),
(p1:Professor {name: 'Tim'}),
(p2:Professor {name: 'John'}),
(p3:Professor {name: 'James'})
CREATE
(s)-[:takes_course]->(p1),(s)-[:takes_course]->(p2),(s)-[:takes_course]->(p3)

```

The screenshot displays the Neo4j Desktop interface. On the left, the 'Database Information' sidebar shows the database name 'neo4j', node labels '(t1s)', 'Professor', and 'Student', relationship types '(t1s) takes_course', and property keys 'address', 'age', 'course_title', and 'name'. The 'Connected as' section shows the user 'neo4j' with roles 'admin, PUBLIC'. The 'DBMS' section shows the cluster role 'primary', version '5.12.0', edition 'Enterprise', name 'neo4j', and databases 'dbs', 'sysinfo', and 'queries'.

The main query editor shows two queries being executed. The first query is:

```

1 MATCH (s:Student {name: 'Bard'}), (p1:Professor {name: 'Taylor'}), (p2:Professor {name: 'John'})
2 CREATE (s)-[:takes_course]->(p1), (s)-[:takes_course]->(p2);
3

```

The execution result shows 'Created 2 relationships, completed after 9 ms.'.

The second query is:

```

neo4j$ Match (s:Student {name: 'Bard'}), (p:Professor {name: 'James'}) CREATE (s) -[:takes_cou...

```

The execution result shows 'Created 1 relationship, completed after 32 ms.'.

The screenshot displays the Neo4j Desktop application. On the left sidebar, there are sections for 'Favorites', 'Local scripts', and 'Sample Scripts'. The 'Sample Scripts' section is expanded, showing various query templates like 'Basic Queries', 'Example Graphs', 'Data Profiling', and 'Common Procedures'. The main workspace is divided into two panes. The top pane shows a Cypher query being executed:

```
neo4j$ ;  
1 MATCH  
2 (s: Student {name: 'Bran'}),  
3 (p1: Professor {name: 'Tim'}),  
4 (p2: Professor {name: 'John'}),  
5 (p3: Professor {name: 'Smith'})  
6 CREATE (s)-[:takes_course]->(p1), (s)-[:takes_course]->(p2), (s)-[:takes_course]->(p3)
```

 Below the query, a status message indicates 'Created 3 relationships, completed after 2 ms.' The bottom pane shows the graph visualization of the query result. It features four red circular nodes labeled 'James', 'Taylor', 'Tim', and 'John'. The nodes are arranged in a square pattern, with 'James' and 'Taylor' at the top and 'Tim' and 'John' at the bottom. The interface includes a toolbar on the left with icons for 'Table', 'Warn', and 'Code', and a toolbar on the right with icons for 'Graph', 'Table', 'Text', and 'Code'.

7. Display courses Ally took.

```
MATCH (s:Student {name: 'Ally'}) -[:takes_course]-> (p:Professor) return  
p.course_title;
```

The screenshot shows the Neo4j Desktop interface. On the left is a sidebar with 'Favorites', 'Local scripts', and 'Sample Scripts'. The main area displays two queries in the 'neo4j\$' console.

Query 1:

```
neo4j$ MATCH (s:Student {name: 'Ally'}) -[:takes_course]→ (p:Professor) return p.course_title;
```

Results:

	p.course_title
1	204
2	202
3	205

Started streaming 3 records after 5 ms and completed after 6 ms.

Query 2:

```
neo4j$ MATCH (s)-[r]→(p) return s, r, p;
```

Results:

The graph shows 15 nodes: Smith, Bob, Taylor, David, David, James, Ally, John, Chris, Arnold, Bran, Alex, Tim, and two unlabeled nodes. Relationships are labeled 'takes_course'.

8. Display all 15 nodes with relationships

```
MATCH (s:Student)-[r:takes_course]→(p:Professor) return s, r, p;
```


The screenshot displays the Neo4j Desktop application. On the left sidebar, there are sections for 'Favorites', 'Local scripts', and 'Sample Scripts'. The 'Sample Scripts' section is expanded, showing various query templates like 'Basic Queries', 'Example Graphs', 'Data Profiling', and 'Common Procedures'. The main workspace is divided into two panes. The top pane shows a Cypher query being executed:

```
1 MATCH
2 (s1:Student {name: 'Bob'}),
3 (s2:Student {name: 'Bran'}),
4 (p1:Professor {name: 'James'})
5 DETACH_DELETE s1, s2, p1;
```

 Below the query, a status message indicates: 'Deleted 3 nodes, deleted 11 relationships, completed after 9 ms.' The bottom pane shows the same query being executed:

```
neo4j$ MATCH (s:Student)-[r:takes_course]-(p:Professor) return s, r, p;
```

 Below the query, a graph visualization is displayed, showing a network of nodes and relationships. The nodes are colored red and blue, and the relationships are labeled with 'takes_course'.

10. Display the graph

```
MATCH (s)-[r]->(p) return s, r, p;
```

Favorites

Local scripts

Sample Scripts

Basic Queries

Create an index

Create unique property constraint

Get some data

Hello World!

Example Graphs

Movie Graph

Northwind Graph

Data Profiling

Common Procedures

neo4j\$

neo4j\$ MATCH (s)-[r]→(p) return s, r, p;

Graph

Table

Text

Code

```
graph TD; Bob((Bob)) -- takes_course --> Chris((Chris)); Bob -- takes_course --> Arnold((Arnold)); Bob -- takes_course --> Tim((Tim)); Bob -- takes_course --> Abby((Abby)); Bob -- takes_course --> Adam((Adam)); Bob -- takes_course --> Taylor((Taylor)); Bob -- takes_course --> David((David)); Bob -- takes_course --> Brad((Brad)); Chris -- takes_course --> Arnold; Arnold -- takes_course --> Tim; Tim -- takes_course --> Abby; Abby -- takes_course --> Adam; Adam -- takes_course --> Taylor; Taylor -- takes_course --> David; David -- takes_course --> Brad; Brad -- takes_course --> Taylor; Taylor -- takes_course --> Bob; Chris -- takes_course --> Bob; Arnold -- takes_course --> Bob; Tim -- takes_course --> Bob; Abby -- takes_course --> Bob; Adam -- takes_course --> Bob; David -- takes_course --> Bob; Brad -- takes_course --> Bob;
```

neo4j\$ MATCH (s1:Student {name: 'Bob'}), (s2:Student {name: 'Bran'}), (p1:Professor {name: 'Ja...})

Table

Warn

Code

Deleted 3 nodes, deleted 11 relationships, completed after 9 ms.