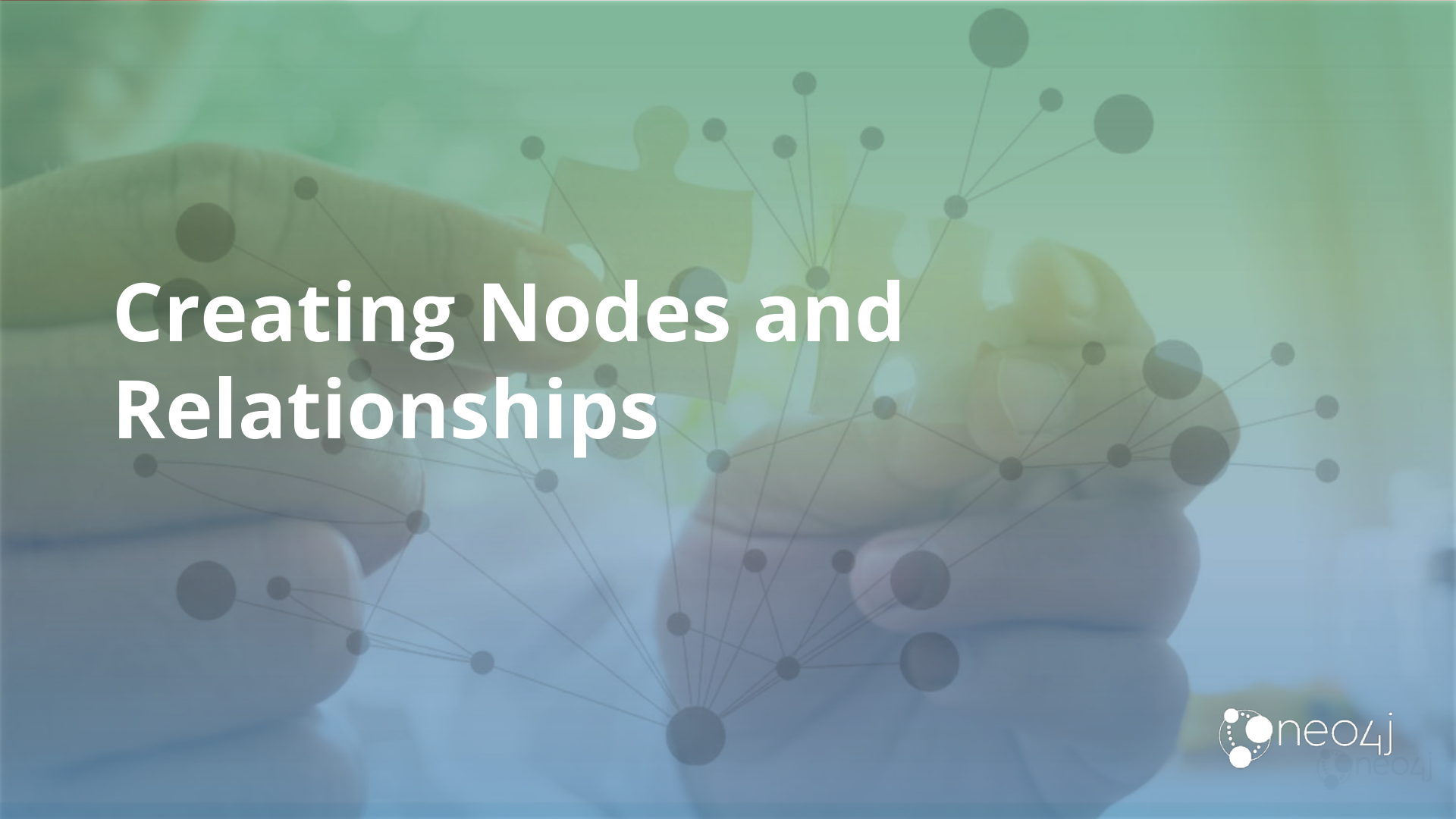
Creating Nodes and Relationships



Overview

At the end of this module, you should be able to write Cypher statements to:

- Create a node:
  - Add and remove node labels.
  - Add and remove node properties.
  - Update properties.
- Create a relationship:
  - Add and remove properties for a relationship.
- Delete a node.
- Delete a relationship.
- Merge data in a graph:
  - Creating nodes.
  - Creating relationships.

Creating a node

Create a node of type *Movie* with the *title* property set to *Batman Begins*:

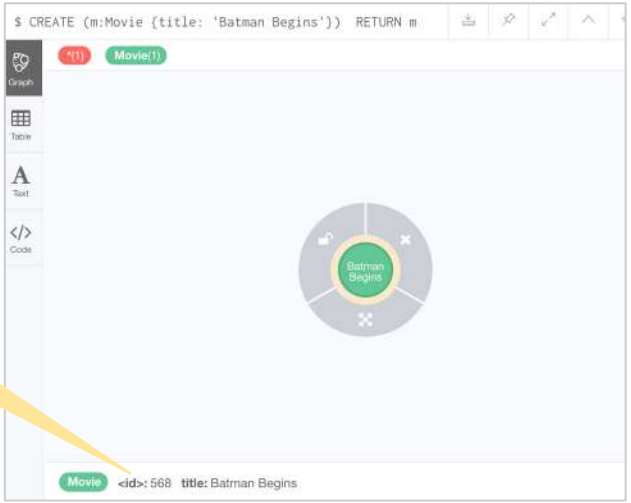
```
CREATE (:Movie {title: 'Batman Begins'})
```

Create a node of type *Movie* and *Action* with the *title* property set to *Batman Begins*:

```
CREATE (:Movie:Action {title: 'Batman Begins'})
```

Create a node of type *Movie* with the *title* property set to *Batman Begins* and return the node:

```
CREATE (m:Movie {title: 'Batman Begins'})
RETURN m
```

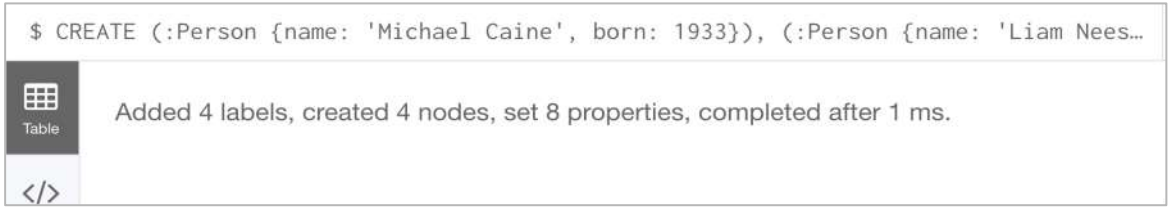


<id> is set by the graph engine

Creating multiple nodes

Create some *Person* nodes for actors and the director for the movie, *Batman Begins*:

```
CREATE (:Person {name: 'Michael Caine', born: 1933}),
    (:Person {name: 'Liam Neeson', born: 1952}),
    (:Person {name: 'Katie Holmes', born: 1978}),
    (:Person {name: 'Benjamin Melniker', born: 1913})
```



Important: The graph engine will create a node with the same properties of a node that already exists. You can prevent this from happening in one of two ways:

1. You can use `MERGE` rather than `CREATE` when creating the node.

2. You can add constraints to your graph.

Adding a label to a node

Add the *Action* label to the movie, *Batman Begins*, return all labels for this node:

```
MATCH (m:Movie)
WHERE m.title = 'Batman Begins'
SET m:Action
RETURN labels(m)
```

Removing a label from a node

Remove the *Action* label to the movie, *Batman Begins*, return all labels for this node:

```
MATCH (m:Movie:Action)
WHERE m.title = 'Batman Begins'
REMOVE m:Action
RETURN labels(m)
```

Adding or updating properties for a node

- If property does not exist for the node, it is added with the specified value.
- If property exists for the node, it is updated with the specified value

Add the properties *released* and *lengthInMinutes* to the movie *Batman Begins*:

```
MATCH (m:Movie)
WHERE m.title = 'Batman Begins'
SET m.released = 2005, m.lengthInMinutes = 140
RETURN m
```

Adding properties to a node - JSON style

Add or update <u>all</u> properties: *title*, *released*, *lengthInMinutes*, *videoFormat*, and *grossMillions* for the movie *Batman Begins*:

```
MATCH (m:Movie)
WHERE m.title = 'Batman Begins'
SET  m = {title: 'Batman Begins',
          released: 2005,
          lengthInMinutes: 140,
          videoFormat: 'DVD',
          grossMillions: 206.5}
RETURN m
```

Adding or updating properties for a node - JSON style

Add the *awards* property and update the *grossMillions* for the movie *Batman Begins*:

```
MATCH (m:Movie)
WHERE m.title = 'Batman Begins'
SET  m += { grossMillions: 300,
            awards: 66}
RETURN m
```

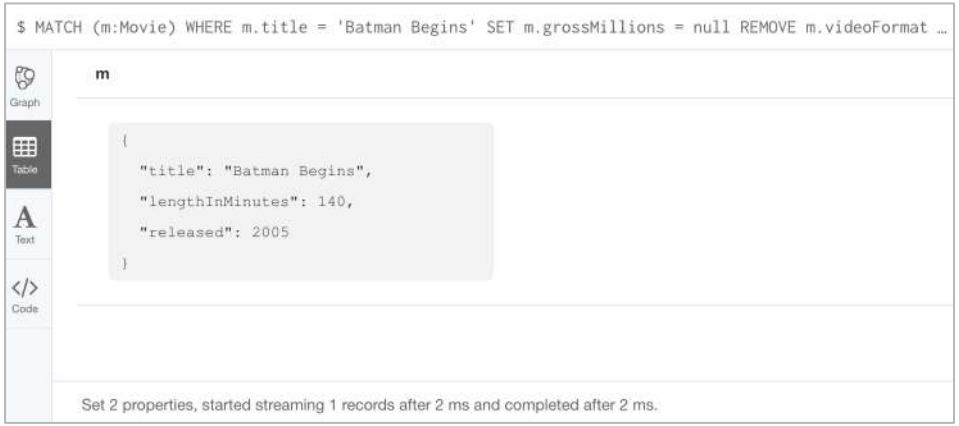
Removing properties from a node

Properties can be removed in one of two ways:

- Set the property value to null
- Use the REMOVE keyword

Remove the grossMillions and videoFormat properties:

```
MATCH (m:Movie)
WHERE m.title = 'Batman Begins'
SET m.grossMillions = null
REMOVE m.videoFormat
RETURN m
```



Exercise 6: Creating nodes

In Neo4j Browser:

:play intro-exercises

Then follow instructions for Exercise 6.

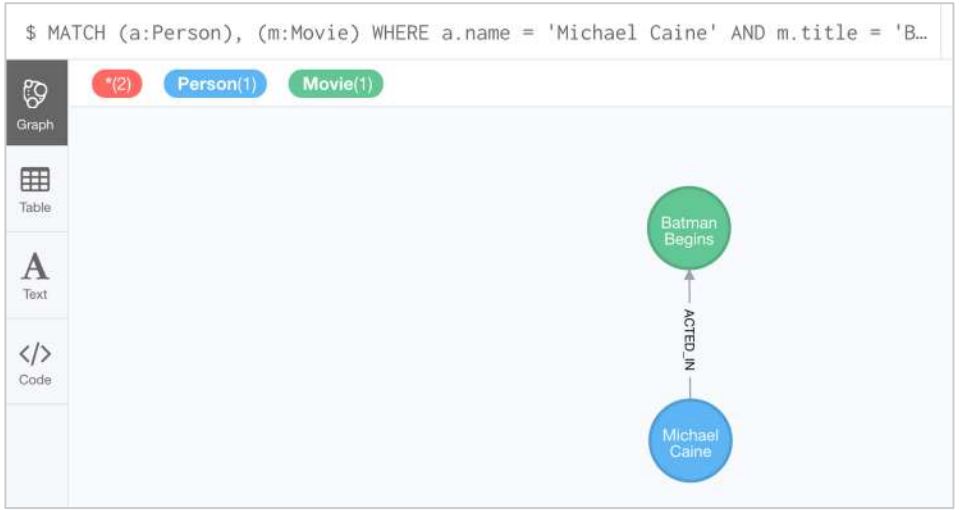
Creating a relationship

You create a relationship by:

1. Finding the “from node”.
2. Finding the “to node”.
3. Using CREATE to add the directed relationship between the nodes.

Create the *:ACTED_IN* relationship between the *Person*, *Michael Caine* and the *Movie*, *Batman Begins*:

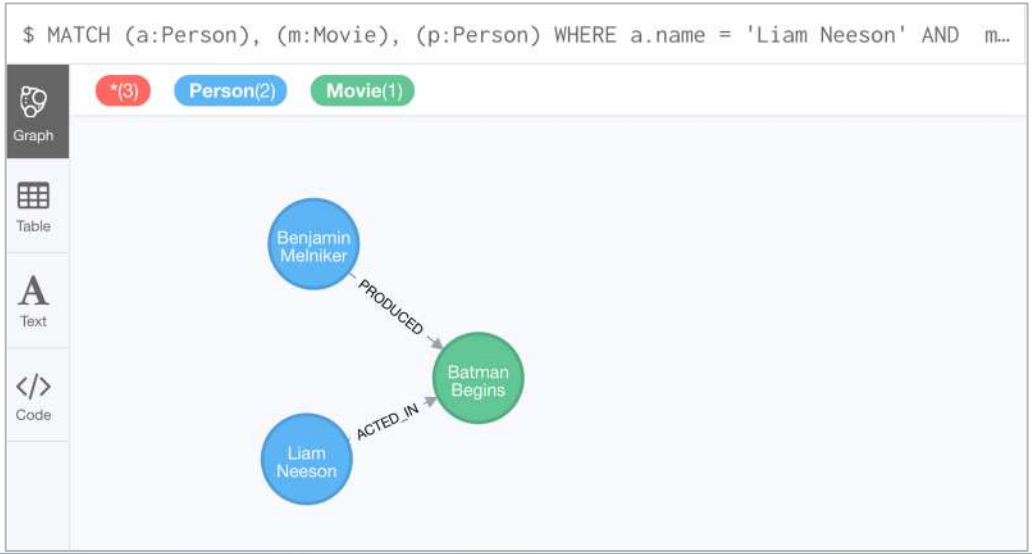
```
MATCH (a:Person), (m:Movie)
WHERE a.name = 'Michael Caine' AND
      m.title = 'Batman Begins'
CREATE (a)-[:ACTED_IN]->(m)
RETURN a, m
```



Creating multiple relationships

Create the *:ACTED_IN* relationship between the *Person*, *Liam Neeson* and the *Movie*, *Batman Begins and the :PRODUCED* relationship between the *Person*, *Benjamin Melniker* and same movie.

```
MATCH (a:Person), (m:Movie), (p:Person)
WHERE a.name = 'Liam Neeson' AND
      m.title = 'Batman Begins' AND
      p.name = 'Benjamin Melniker'
CREATE (a)-[:ACTED_IN]->(m)<-[:PRODUCED]-(p)
RETURN a, m, p
```

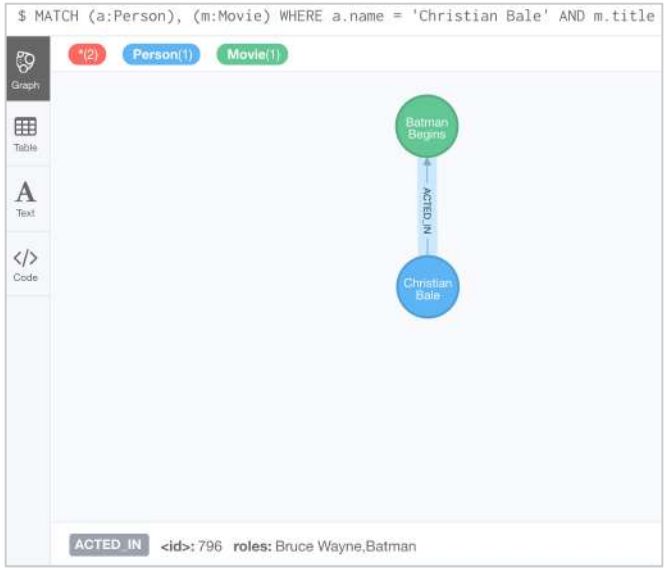


Adding properties to relationships

Same technique you use for creating and updating node properties.

Add the *roles* property to the *:ACTED_IN* relationship from Christian Bale to *Batman Begins*:

```
MATCH (a:Person), (m:Movie)
WHERE a.name = 'Christian Bale' AND
      m.title = 'Batman Begins' AND
      NOT exists((a)-[:ACTED_IN]->(m))
CREATE (a)-[rel:ACTED_IN]->(m)
SET rel.roles = ['Bruce Wayne','Batman']
RETURN a, m
```

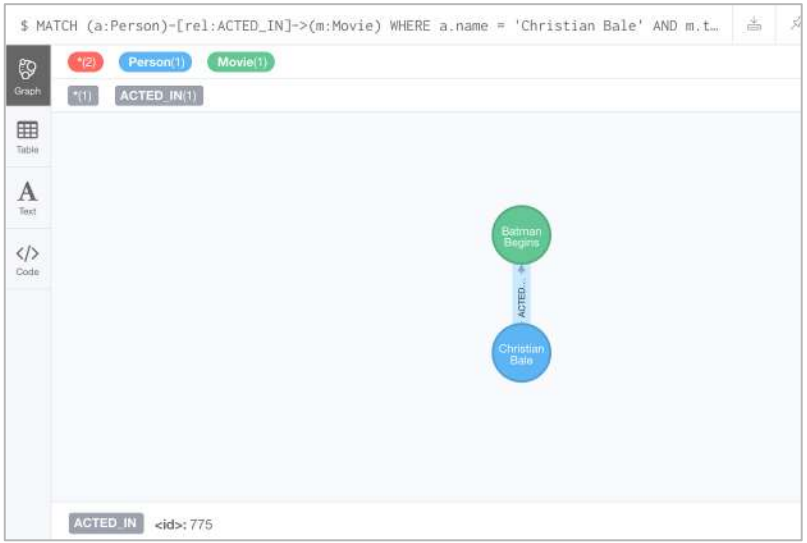


Removing properties from relationships

Same technique you use for removing node properties.

Remove the *roles* property from the *:ACTED_IN* relationship from Christian Bale to *Batman Begins*:

```
MATCH (a:Person)-[rel:ACTED_IN]->(m:Movie)
WHERE a.name = 'Christian Bale' AND
      m.title = 'Batman Begins'
REMOVE rel.roles
RETURN a, rel, m
```



Exercise 7: Creating relationships

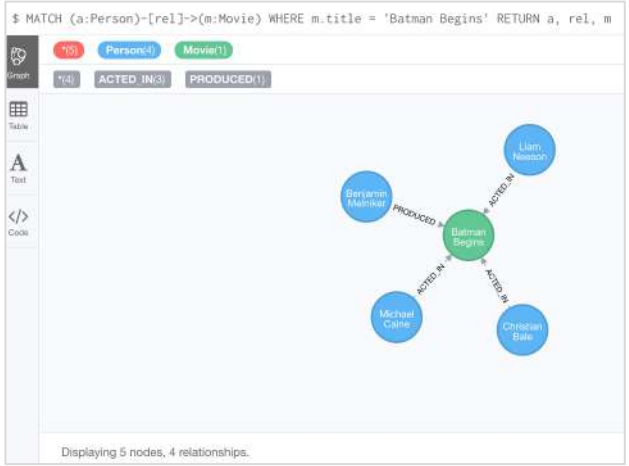
In Neo4j Browser:

:play intro-exercises

Then follow instructions for Exercise 7.

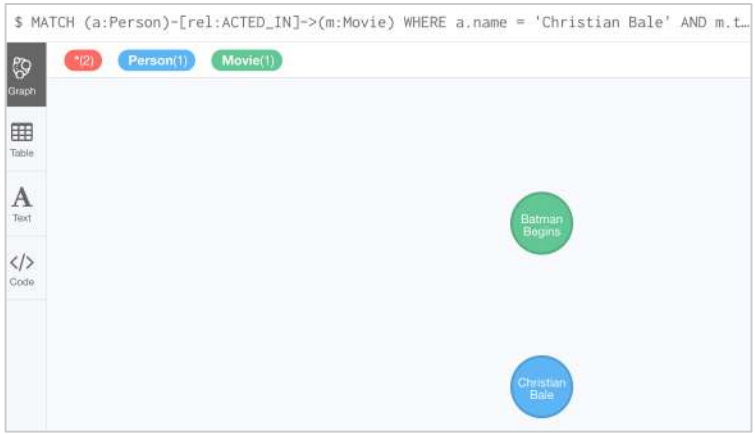
Deleting a relationship

Batman Begins relationships:



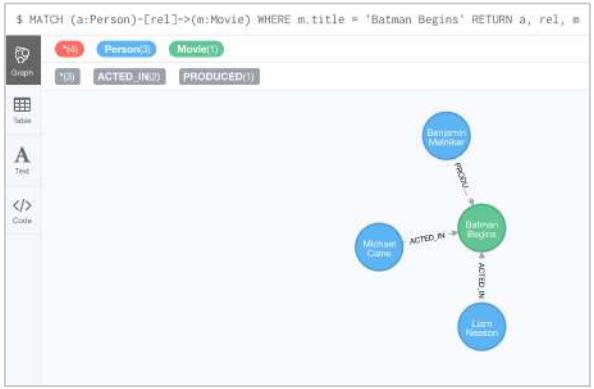
Delete the *:ACTED_IN* relationship between *Christian Bale* and *Batman Begins*:

```
MATCH (a:Person)-[rel:ACTED_IN]->(m:Movie)
WHERE a.name = 'Christian Bale' AND
      m.title = 'Batman Begins'
DELETE rel
RETURN a, m
```

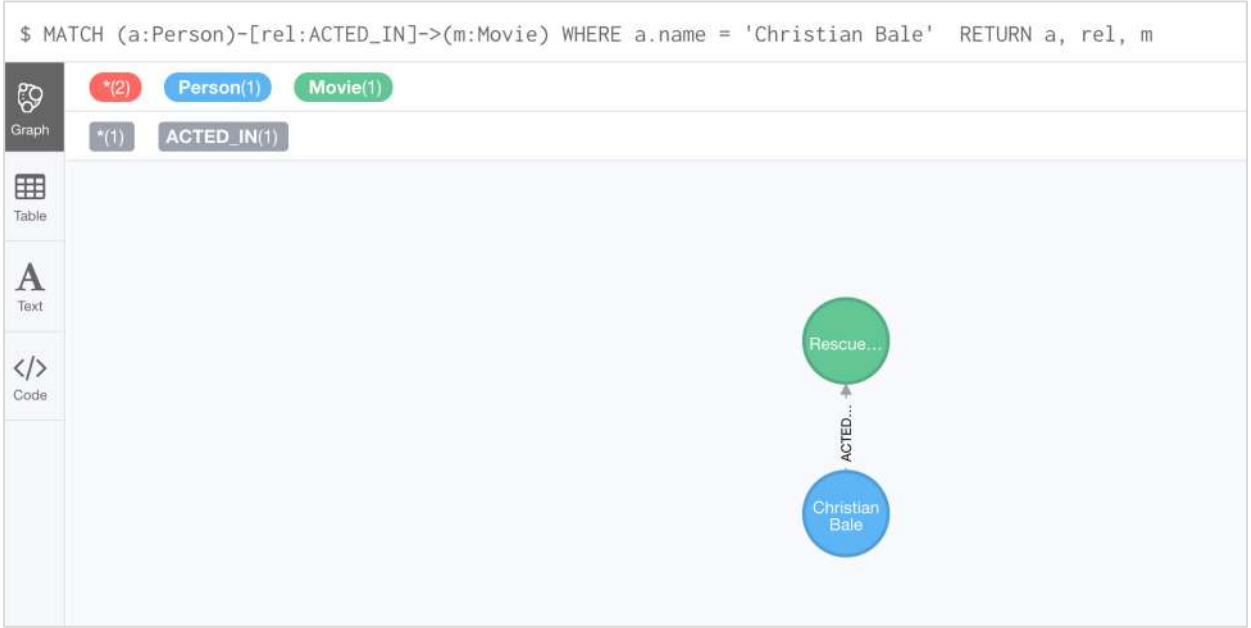


After deleting the relationship from *Christian Bale* to *Batman Begins*

Batman Begins relationships:

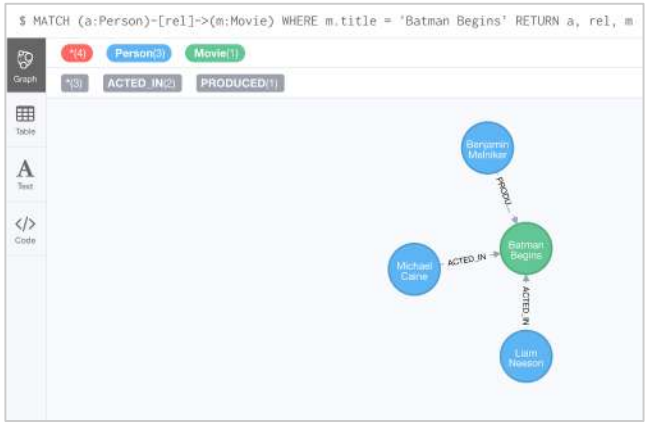


Christian Bale relationships:



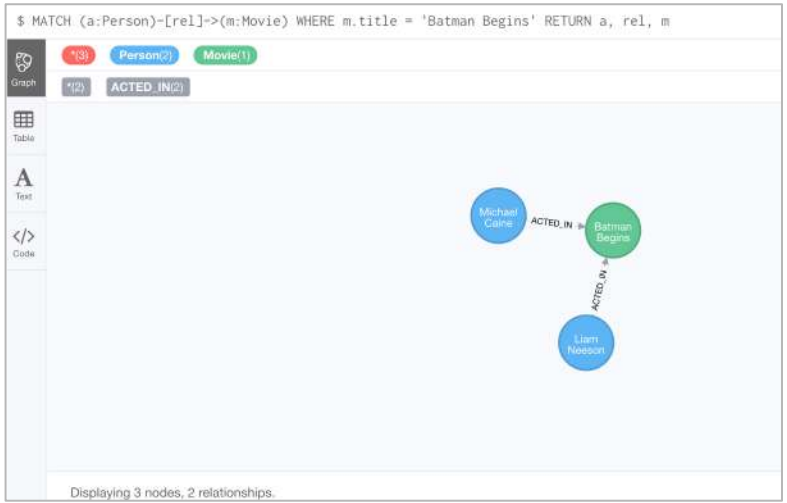
Deleting a relationship and a node - 1

Batman Begins relationships:



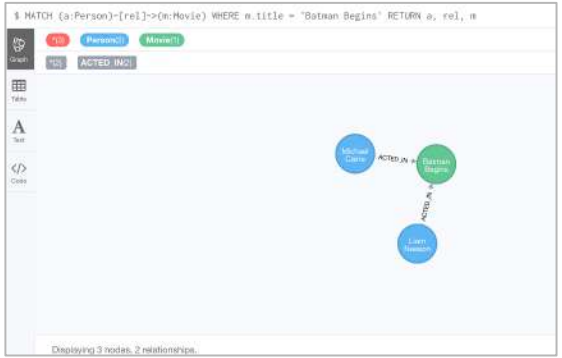
Delete the *:PRODUCED* relationship between *Benjamin Melniker* and *Batman Begins,* as well as the *Benjamin Melniker* node:

```
MATCH (p:Person)-[rel:PRODUCED]->(:Movie)
WHERE p.name = 'Benjamin Melniker'
DELETE rel, p
```



Deleting a relationship and a node - 2

Batman Begins relationships:

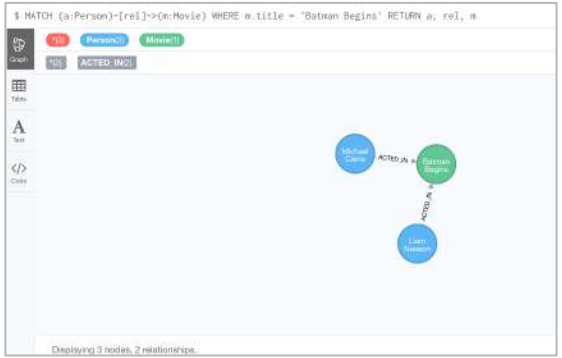


Attempt to delete *Liam Neeson* and <u>not</u> his relationships to any other nodes:

```
MATCH (p:Person)
WHERE p.name = 'Liam Neeson'
DELETE p
```

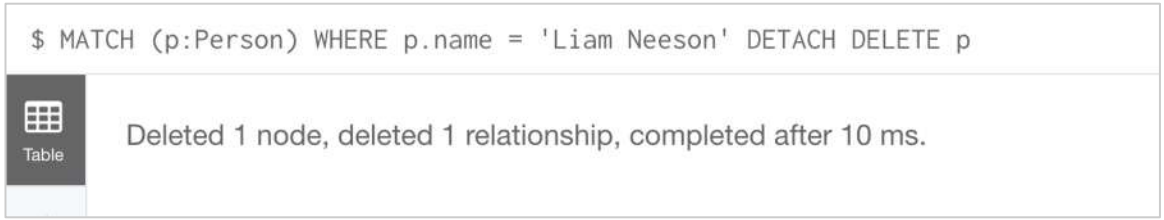
Deleting a relationship and a node - 3

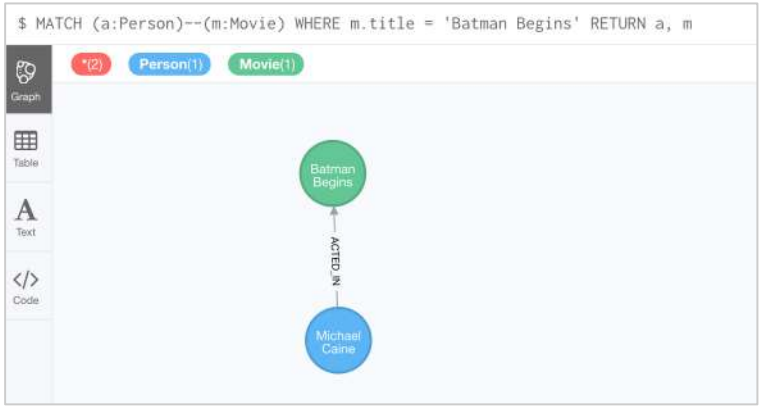
Batman Begins relationships:



Delete *Liam Neeson* and his relationships to any other nodes:

```
MATCH (p:Person)
WHERE p.name = 'Liam Neeson'
DETACH DELETE p
```





Exercise 8: Deleting nodes and relationships

In Neo4j Browser:

:play intro-exercises

Then follow instructions for Exercise 8.

Merging data in a graph

- Create a node with a different label (You do not want to add a label to an existing node.).
- Create a node with a different set of properties (You do not want to update a node with existing properties.).
- Create a unique relationship between two nodes.

Using MERGE to create nodes

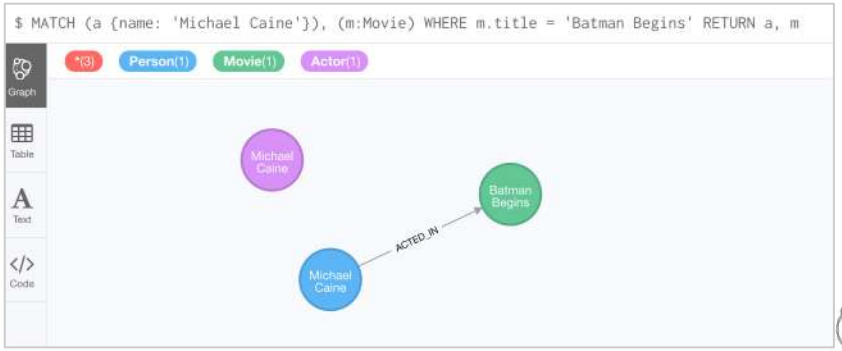
Current *Michael Caine Person* node:

Add a *Michael Caine Actor* node with a value of *1933* for *born* using MERGE. The *Actor* node is not found so a new node is created:

```
MERGE (a:Actor {name: 'Michael Caine'})
SET a.born=1933
RETURN a
```

Important: Only specify properties that will have unique keys when you merge.

Resulting *Michael Caine nodes*:



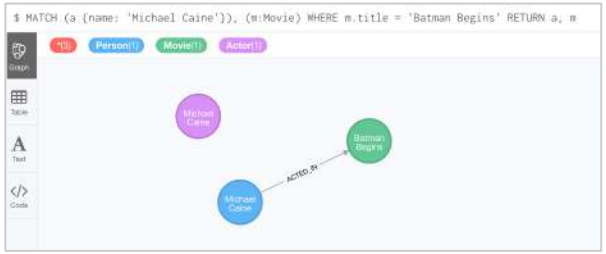
Using MERGE to create relationships

Add the relationship(s) from all *Person* nodes with a *name* property that ends with *Caine* to the *Movie* node, *Batman Begins*:

```
MATCH (p:Person), (m:Movie)
WHERE m.title = 'Batman Begins' AND
p.name ENDS WITH 'Caine'
MERGE (p)-[:ACTED_IN]->(m)
RETURN p, m
```

Specifying creation behavior for the merge

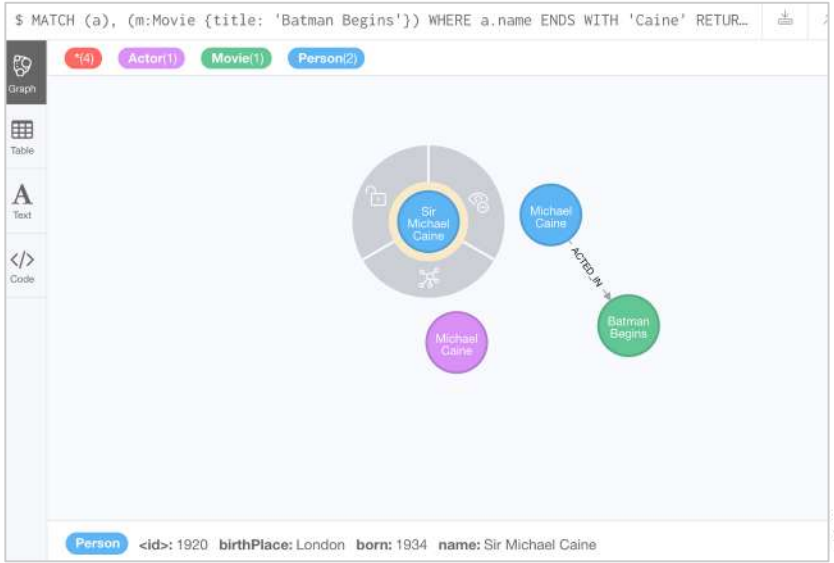
Current *Michael Caine* nodes:



Add a *Sir Michael Caine Person* node with a *born* value of *1934* for *born* using MERGE and also set the *birthPlace* property:

```
MERGE (a:Person {name: 'Sir Michael Caine'})
ON CREATE SET a.born = 1934,
              a.birthPlace = 'London'
RETURN a
```

Resulting *Michael Caine nodes*:



Specifying match behavior for the merge

Current *Michael Caine* nodes:

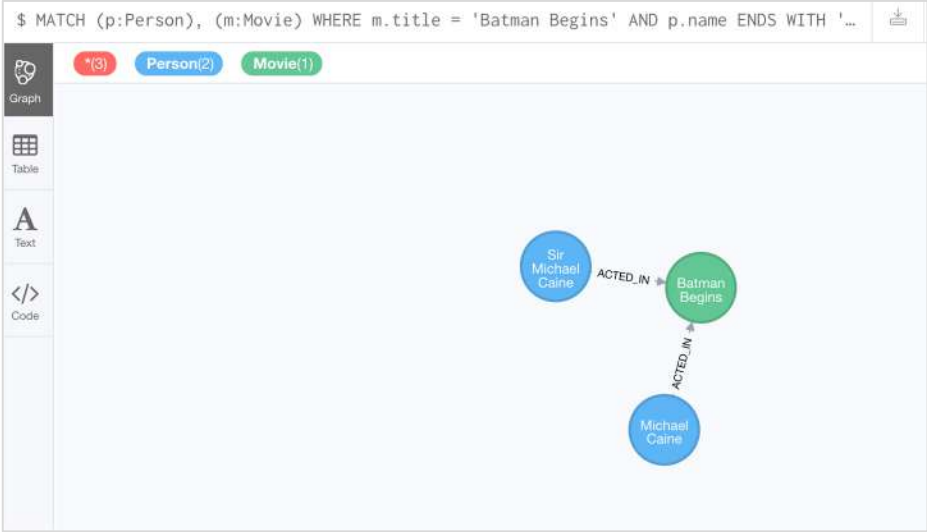
Add or update the *Michael Caine Person* node:

```
MERGE (a:Person {name: 'Sir Michael Caine'})
ON CREATE SET a.born = 1934,
              a.birthPlace = 'UK'
ON MATCH SET a.birthPlace = 'UK'
RETURN a
```

Using MERGE to create relationships

Make sure that all *Person* nodes with a person whose name ends with *Caine* are connected to the *Movie* node, *Batman Begins*.

```
MATCH (p:Person), (m:Movie)
WHERE m.title = 'Batman Begins' AND p.name ENDS WITH 'Caine'
MERGE (p)-[:ACTED_IN]->(m)
RETURN p, m
```



Exercise 9: Merging data in the graph

In Neo4j Browser:

:play intro-exercises

Then follow instructions for Exercise 9.

Accessing Neo4j resources

There are many ways that you can learn more about Neo4j. A good starting point for learning about the resources available to you is the **Neo4j Learning Resources** page at https://neo4j.com/developer/resources/.