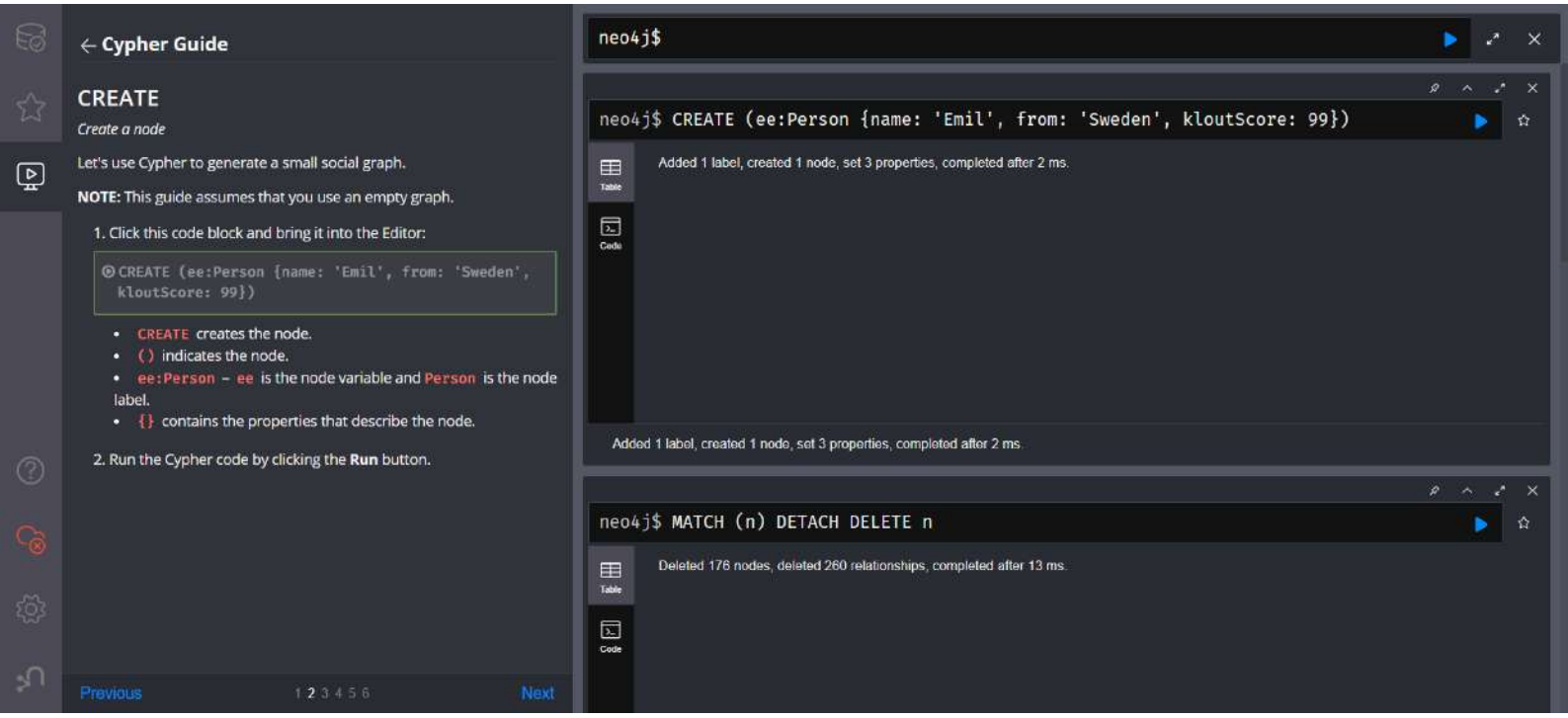
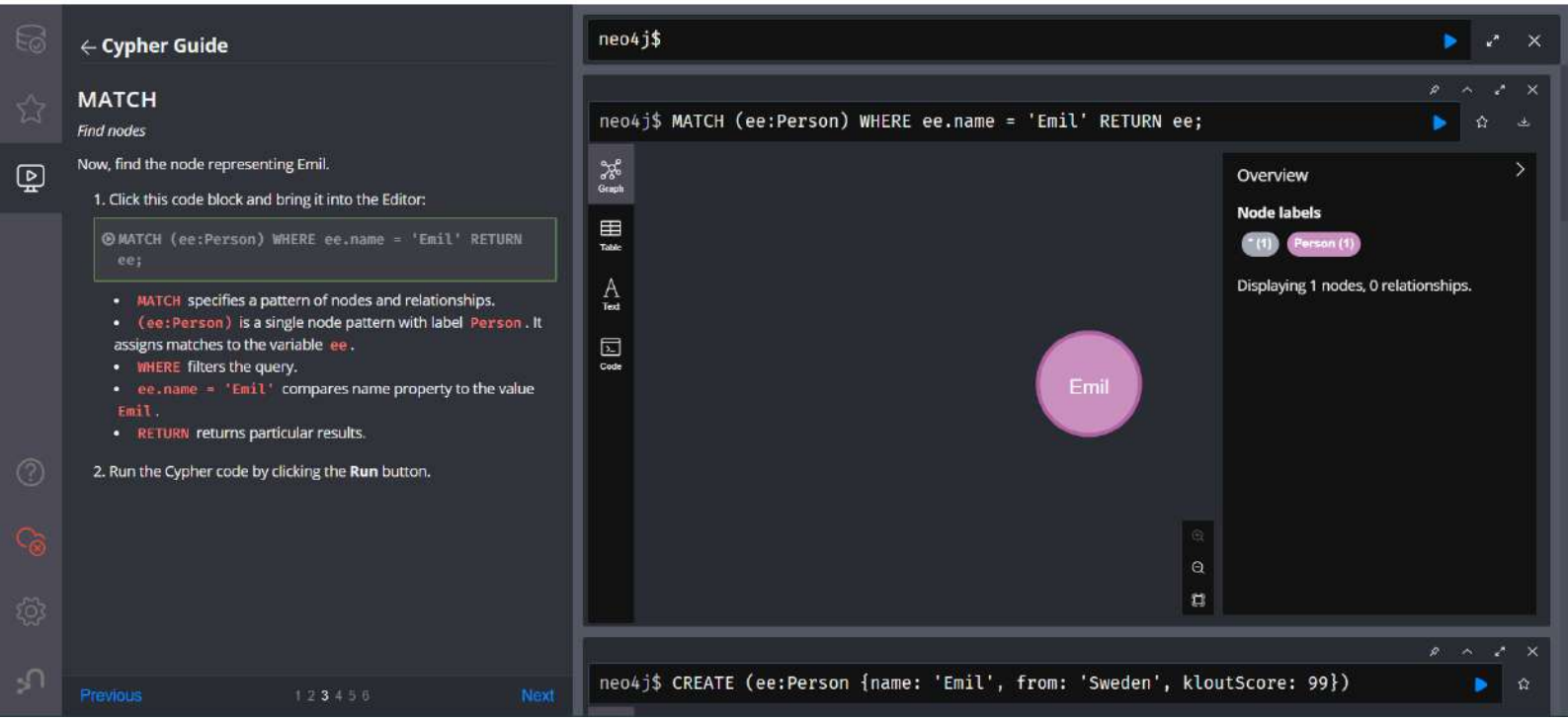
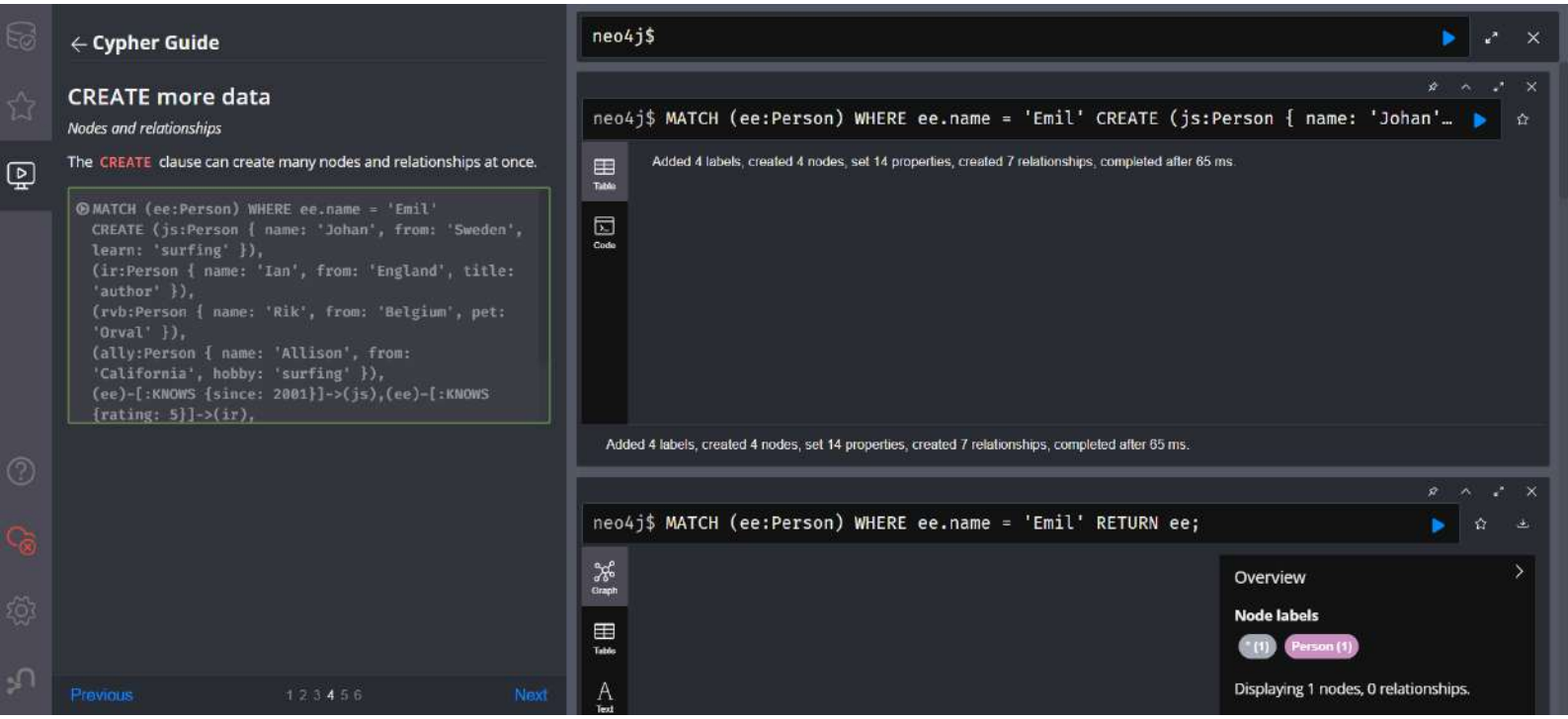
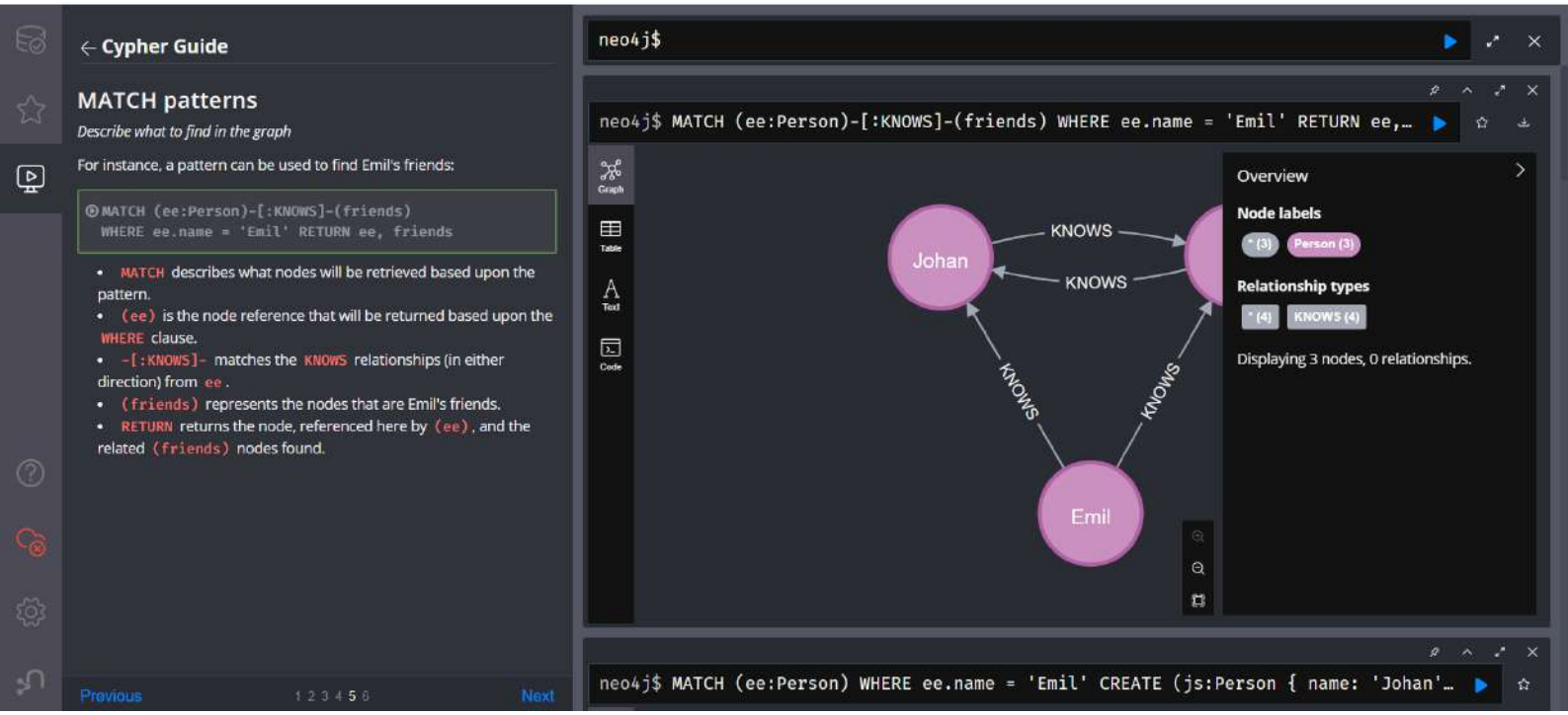

← Cypher Guide
CREATE
Create a node
Let's use Cypher to generate a small social graph.
NOTE: This guide assumes that you use an empty graph.
1. Click this code block and bring it into the Editor:
CREATE (ee:Person {name: 'Emil', from: 'Sweden', kloutScore: 99})
CREATE creates the node.
() indicates the node.
ee:Person – ee is the node variable and Person is the node label.
{} contains the properties that describe the node.
2. Run the Cypher code by clicking the Run button.
Previous
1 2 3 4 5 6
Next
neo4j$
neo4j$ CREATE (ee:Person {name: 'Emil', from: 'Sweden', kloutScore: 99})
Table
Code
Added 1 label, created 1 node, set 3 properties, completed after 2 ms.
Added 1 label, created 1 node, set 3 properties, completed after 2 ms
neo4j$ MATCH (n) DETACH DELETE n
Table
Code
Deleted 176 nodes, deleted 260 relationships, completed after 13 ms.


Cypher Guide
MATCH
Find nodes
Now, find the node representing Emil.
1. Click this code block and bring it into the Editor:
MATCH (ee:Person) WHERE ee.name = 'Emil' RETURN ee;
MATCH specifies a pattern of nodes and relationships.
(ee:Person) is a single node pattern with label Person. It assigns matches to the variable ee.
WHERE filters the query.
ee.name = 'Emil' compares name property to the value Emil.
RETURN returns particular results.
2. Run the Cypher code by clicking the Run button.
Previous
1 2 3 4 5 6
Next
neo4j$
neo4j$ MATCH (ee:Person) WHERE ee.name = 'Emil' RETURN ee;
Graph
Table
Text
Code
Emil
Overview
Node labels
* (1)
Person (1)
Displaying 1 nodes, 0 relationships.
neo4j$ CREATE (ee:Person {name: 'Emil', from: 'Sweden', kloutScore: 99})


← Cypher Guide
CREATE more data
Nodes and relationships
The CREATE clause can create many nodes and relationships at once.
MATCH (ee:Person) WHERE ee.name = 'Emil'
CREATE (js:Person { name: 'Johan', from: 'Sweden',
learn: 'surfing' }),
(ir:Person { name: 'Ian', from: 'England', title:
'author' }),
(rvb:Person { name: 'Rik', from: 'Belgium', pet:
'Orval' }),
(ally:Person { name: 'Allison', from:
'California', hobby: 'surfing' }),
(ee)-[:KNOWS {since: 2001}]->(js),(ee)-[:KNOWS
{rating: 5}]->(ir),
Previous
1 2 3 4 5 6
Next
neo4j$
neo4j$ MATCH (ee:Person) WHERE ee.name = 'Emil' CREATE (js:Person { name: 'Johan'…
Table
Code
Added 4 labels, created 4 nodes, set 14 properties, created 7 relationships, completed after 65 ms.
Added 4 labels, created 4 nodes, set 14 properties, created 7 relationships, completed after 65 ms.
neo4j$ MATCH (ee:Person) WHERE ee.name = 'Emil' RETURN ee;
Graph
Table
Text
Overview
Node labels
* (1)
Person (1)
Displaying 1 nodes, 0 relationships.


Cypher Guide
MATCH patterns
Describe what to find in the graph
For instance, a pattern can be used to find Emil's friends:
MATCH (ee:Person)-[:KNOWS]-(friends)
WHERE ee.name = 'Emil' RETURN ee, friends
MATCH describes what nodes will be retrieved based upon the pattern.
(ee) is the node reference that will be returned based upon the WHERE clause.
-[:KNOWS]- matches the KNOWS relationships (in either direction) from ee.
(friends) represents the nodes that are Emil's friends.
RETURN returns the node, referenced here by (ee), and the related (friends) nodes found.
Previous
1 2 3 4 5 6
Next
neo4j$
neo4j$ MATCH (ee:Person)-[:KNOWS]-(friends) WHERE ee.name = 'Emil' RETURN ee,…
Graph
Table
Text
Code
Johan
KNOWS
KNOWS
KNOWS
KNOWS
Emil
Overview
Node labels
* (3)
Person (3)
Relationship types
* (4)
KNOWS (4)
Displaying 3 nodes, 0 relationships.
neo4j$ MATCH (ee:Person) WHERE ee.name = 'Emil' CREATE (js:Person { name: 'Johan'…