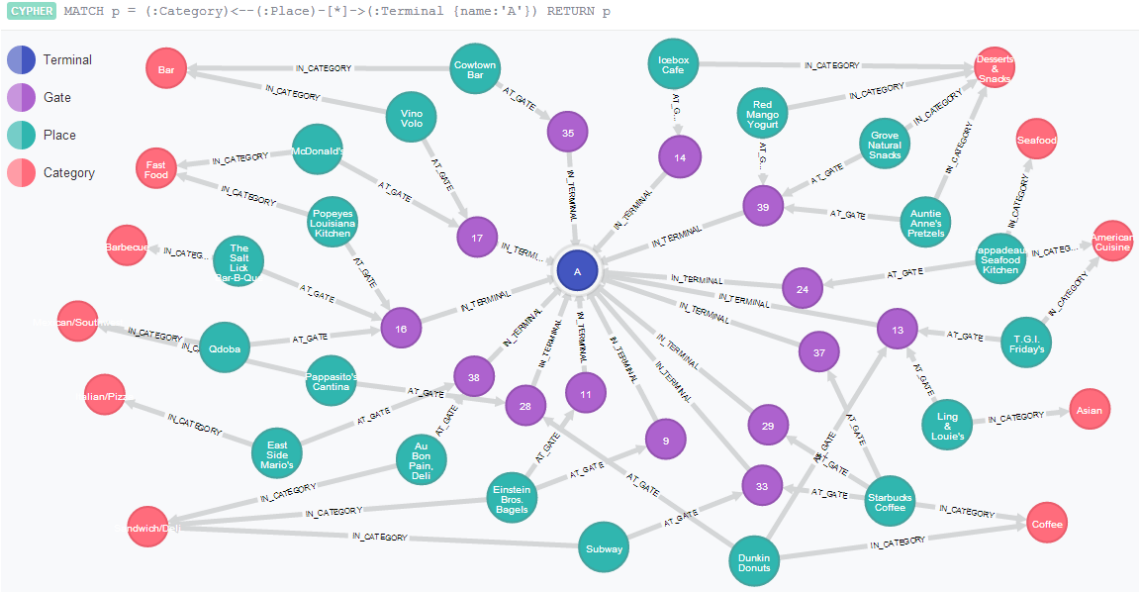
Neo4j instruction

Danchen Zhang

Catalog

- Installation
- Cypher Query Language (CQL)
- Connect to Java and Python



Tutorial materials

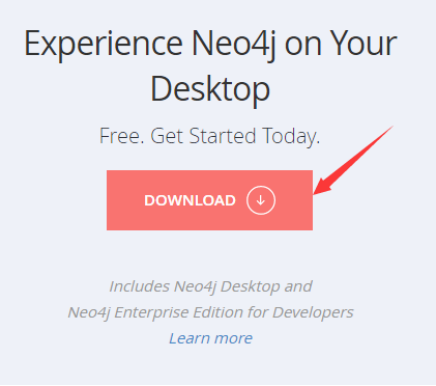
[1] https://neo4j.com/developer/get-started/

[2] https://neo4j.com/blog/neo4j-video-tutorials/

[3] https://www.tutorialspoint.com/neo4j/index.htm

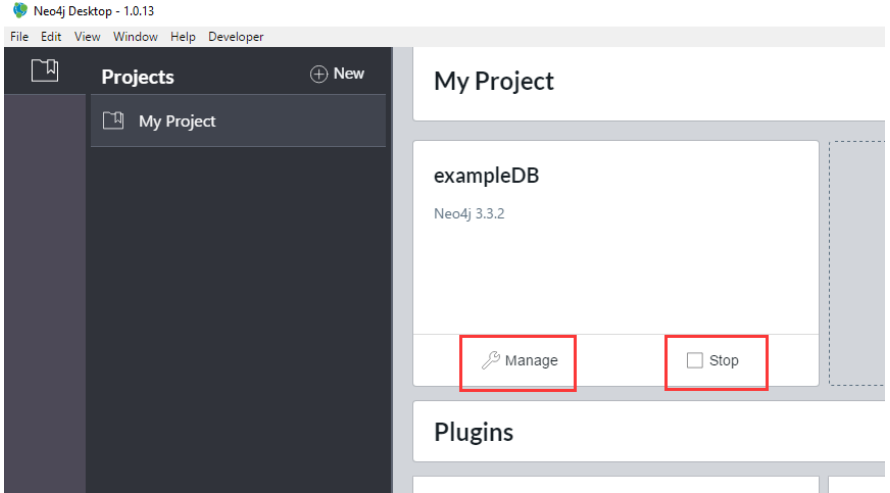
[4] https://www.quackit.com/neo4j/tutorial/

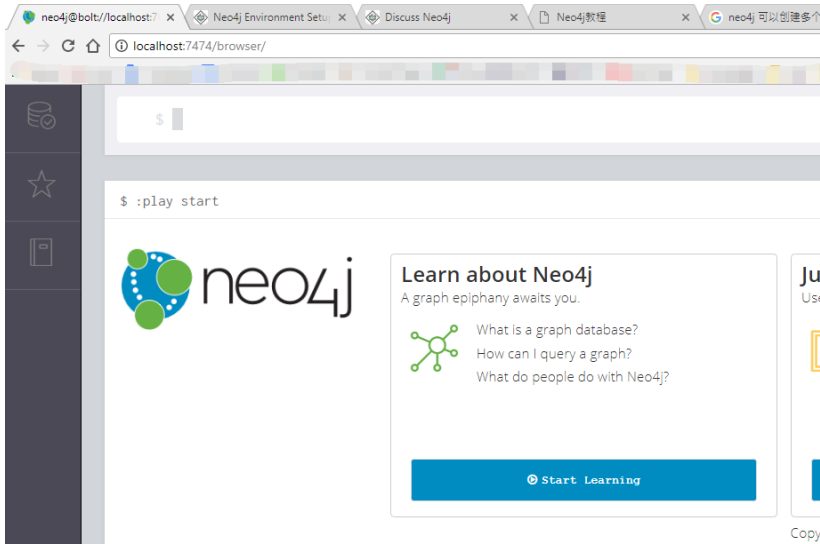
Installation & Start



- You can obtain .exe and .dmg in https://neo4j.com/download/

- Localhost:7474
- Default username: neo4j
- Default password: neo4j





Installation & Start 2

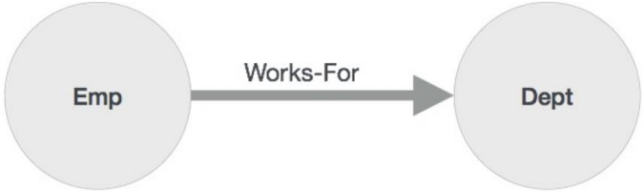
- https://www.tutorialspoint.com/neo4j/neo4j_environment_setup.htm
- Remember to setup the system path:
  - NEO4J_HOME =*******\neo4j-community-3.3.2
  - PATH = *******\neo4j-community-3.3.2\bin

Select Windows PowerShell

```
PS D:\Projects> neo4j console
WARNING: This command does not appear to be running with administra
2018-02-02 16:32:25.552+0000 INFO  ======== Neo4j 3.3.2 ========
2018-02-02 16:32:25.951+0000 INFO  Starting...
2018-02-02 16:32:32.098+0000 INFO  Bolt enabled on 127.0.0.1:7687.
2018-02-02 16:33:05.417+0000 INFO  Started.
```

Neo4j basic elements

- Node & Relationship
- Labels



- Properties

Any Q?

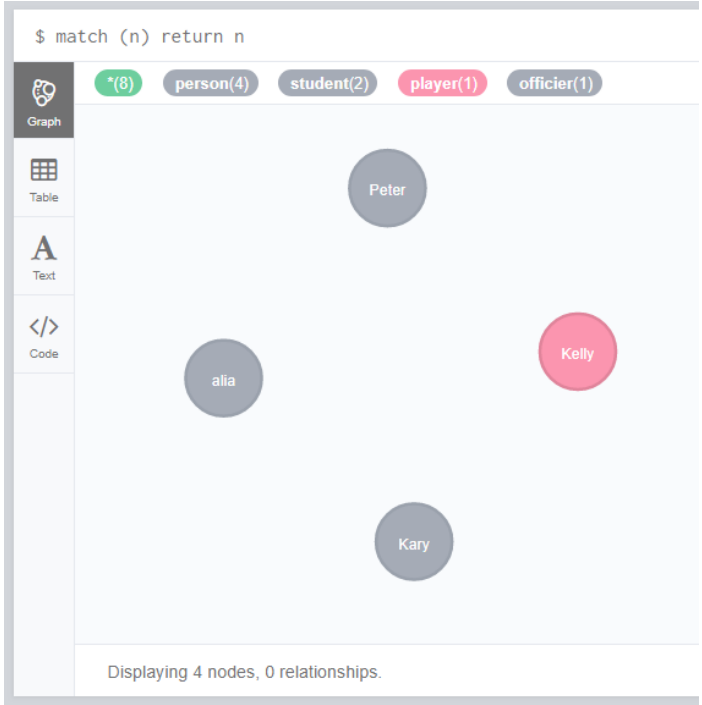
Neo4j Cypher Query Language (CQL)

- Ref: https://www.tutorialspoint.com/neo4j/neo4j_cql_introduction.htm
- Frequent CQL operations:
  - Create (unique), delete, merge
  - match, return, where + order by, limit, skip, Count, substring
  - Set, remove
  - foreach

Set 1.1: create nodes & relationships

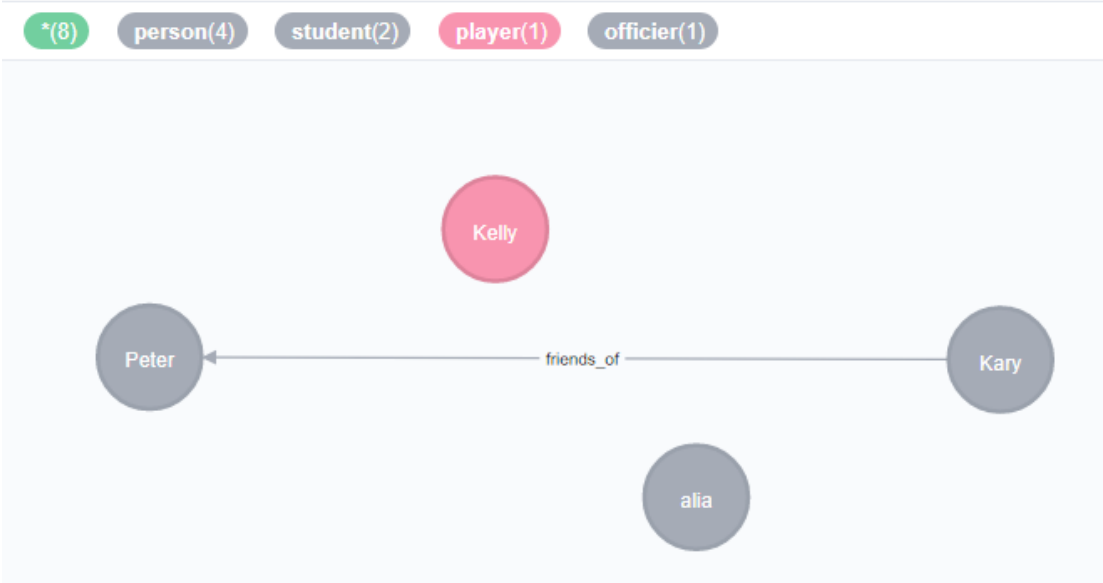
- CREATE (node_name:label {key1:value1, key2:value2,…})
- Use “match (n) return n” to check results
- CREATE CONSTRAINT ON (n:label) ASSERT n.property IS UNIQUE

- CREATE (a:person:student{name: "Kary", gender:"female", age:25})
- CREATE (b:person:player{name: "Kelly", gender:"female", age:20})
- CREATE (c:person:student{name: "Peter", gender:"male", age:25})
- CREATE (d:person:officier{name: "alia", gender:"female", age:23})



CREATE CONSTRAINT ON (n:person)
ASSERT n.name IS UNIQUE

- MATCH (a:student), (b:student)
- WHERE a.name = "Kary" AND b.name = "Peter"
- CREATE (a)-[rel:friends_of]->(b)
- RETURN a,b

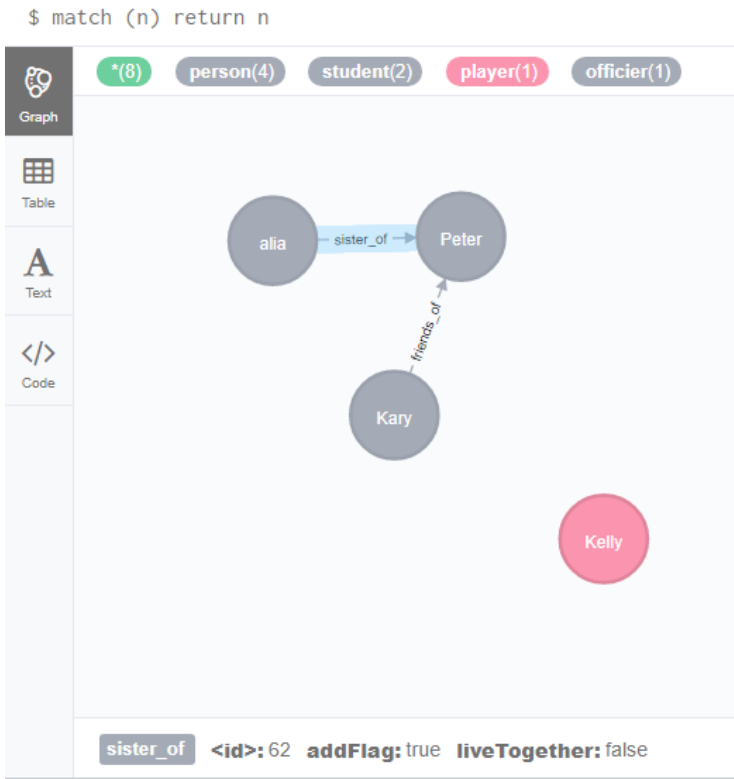


Set 1.2: Merge

- Match first, if the node is not found, create the node.
- On Create and On Match can help with extra data operation.
- Merge both nodes & relationships.

- MERGE (node:label) RETURN node
- MERGE (node a)-[rel:relationship]->(node b)

- MATCH (a:person), (b:person)
- WHERE a.name = "alia" AND b.name = "Peter"
- MERGE (a)-[relationship:sister_of{liveTogether:False}]->(b)
- on match set relationship.addFlag=False
- on create set relationship.addFlag=True
- RETURN a, b



Set 1.3: delete

- Delete selected nodes:
  - Create a node: Merge(testNode:forDelExple)
  - Delete it: MATCH (node: forDelExple) DETACH DELETE node
- Delete all nodes & relationships in the database.
  - MATCH (n) DETACH DELETE n

Set 2.1: match where return + orderby

- match (n:person)
- where n.gender = "female"
- return n.name, n.age
- order by n.age

```
$ match (n:person) where n.gender = "female" return n.name, n.age  order by n.age
```

Table

Text

Code

n.name	n.age
"Kelly"	20
"alia"	23
"Kary"	25

Set 2.2: limit, skip, Count, substring

- match (n:person) return n.name, n.age, n.gender limit 2
- match (n:person) return n.name, n.age, n.gender order by n.age skip 2
- match (n:person) return n.gender, count(*)
- match (n:person) return n.name, n.age,substring(n.gender, 0, 1)

Set3: SET + REMOVE

- SET:
  - Add a/multiple property
  - Add a/multiple labels on a node
  - Remove a property <u>= REMOVE</u>

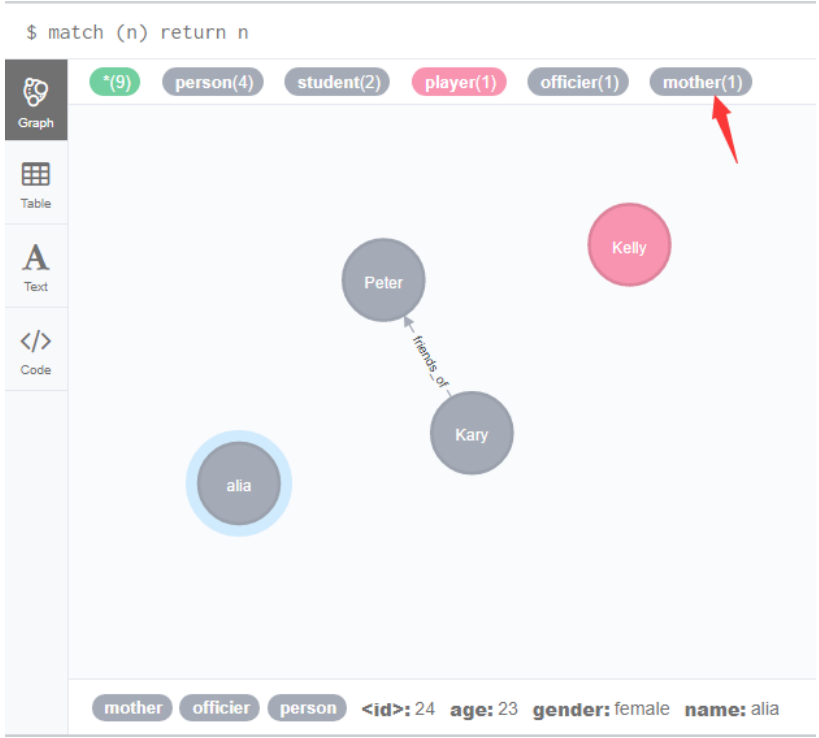
- match(n)
- where n.name="alia"
- set n.hobby="reading"
- return n

- match(n)
- where n.name="alia"
- set n.hobby=<u>null</u>
- return n

Set3: SET + REMOVE

- SET:
  - Add a/multiple property
  - Add a/multiple labels on a node
  - Remove a property <u>= REMOVE</u>

- match(n)
- where n.name="alia"
- set n:mother
- return n

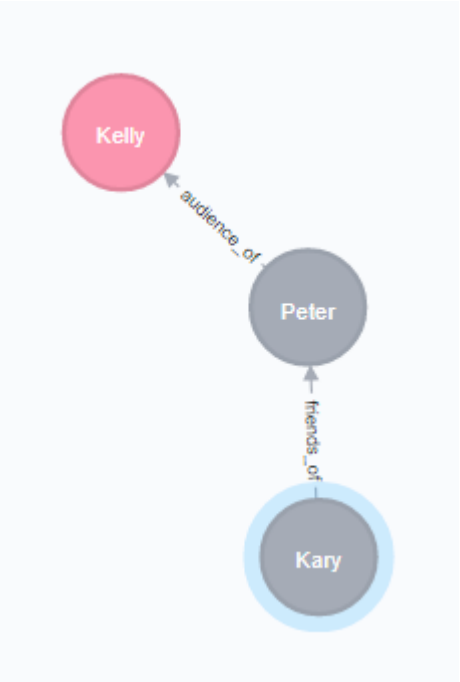


Set 4: foreach

- Traverse the nodes in the path.

- MATCH (a), (b)
- WHERE a.name = "Peter" AND b.name = "Kelly"
- CREATE (a)-[rel:audience_of]->(b)
- RETURN a,b

- MATCH p = (a:person)-[*]->(b:player)
- WHERE a.name = "Kary" AND b.name = "Kelly"
- FOREACH (n IN nodes(p)| SET n.marked = TRUE)
- return p



Practice

- Find the names of all person who is more than 21 years old.

n.name
"Kary"
"Peter"
"alia"

- Find the count of student in the current data.

count(*)
2

- List the gender of all people in the data, and replace the “a” in the output string with “A”.

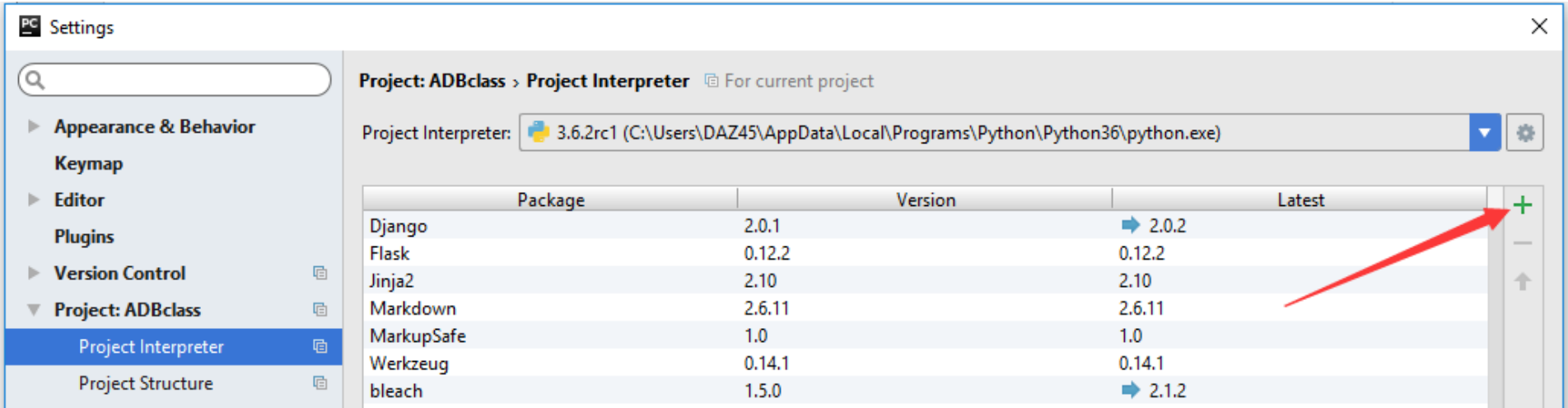
https://neo4j.com/docs/developer-manual/current/cypher/functions/string/#functions-replace

replace(n.gender, "a", "A")
"femAle"
"femAle"
"mAle"
"femAle"

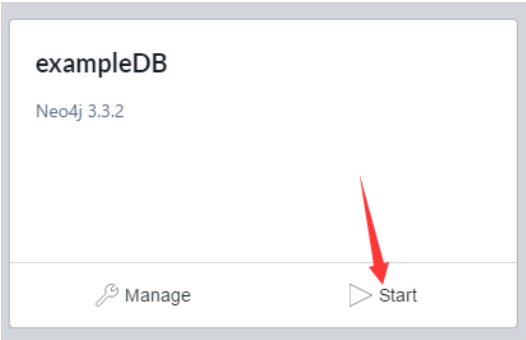
Any Q?

Control Neo4j with python

- You need py2neo
  - $ pip install py2neo
  - Pycharm=>File=>Settings=>Project:**=>Project Interpreter=>

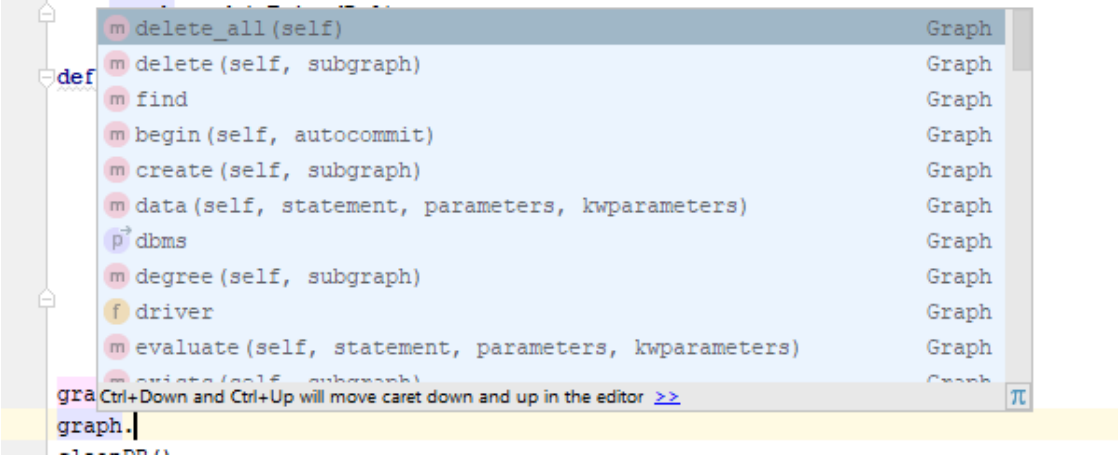


Connection to Neo4j with python



- First, build a database, and turn on the server.
- Second, code and run it.

- graph = Graph("localhost:7474", username = "neo4j", password = “111111")



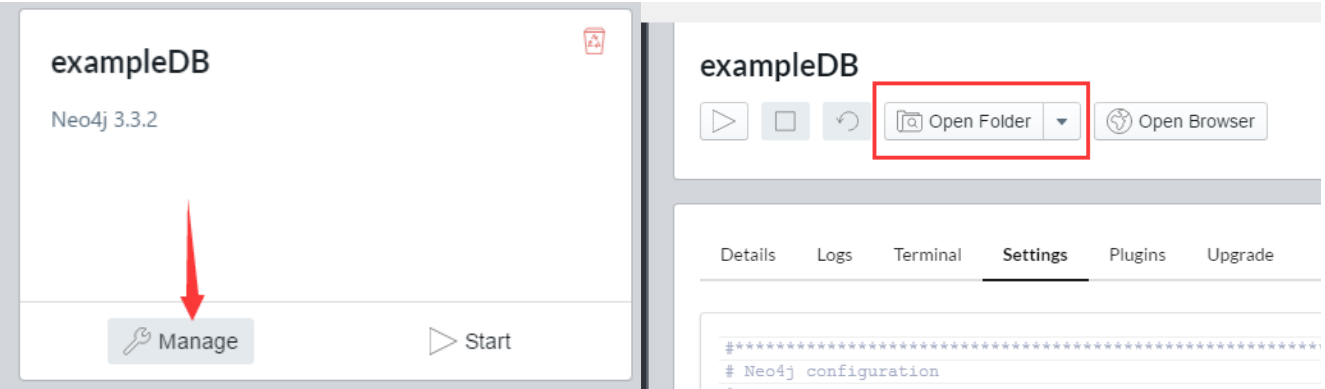
```python
from py2neo import Graph, Node, Relationship

def cleanDB():
    graph.delete_all()

def showAll():
    results = graph.find(label="Person")
    for f in results:
        print(f)
    print("========================")

def addNodeRel():
    peter=Node("Person", "student", name="Peter", age=25, gender="male")
    kelly=Node("Person", "player", name="Kelly", age=20, gender="female")
    kary=Node("Person", "student", name="Kary", age=25, gender="female")
    alia=Node("Person", "officer", name="alia", age=23, gender="female")
    graph.create(peter)
    graph.create(kelly)
    graph.create(kary)
    graph.create(alia)

    aFriendRel = Relationship(kary, 'friends_of', peter)
    aFriendRel['years'] = 2
    graph.create(aFriendRel)
```

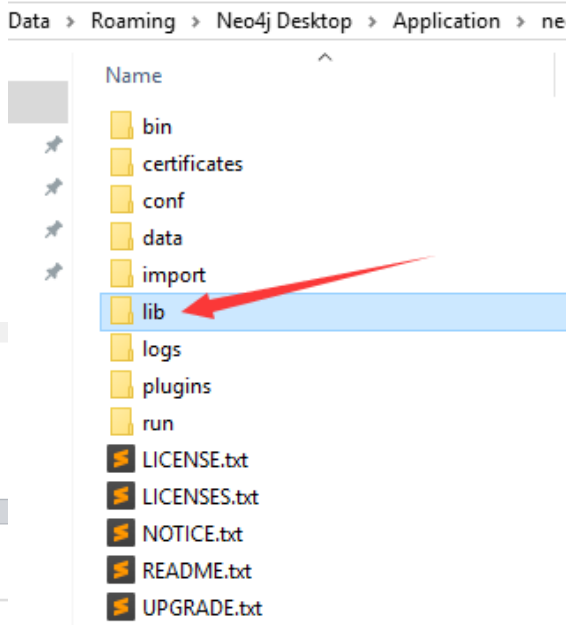
```python
def update():
    kary=Node("Person", "student", name="Kary", age=25, gender="female")
    peter=Node("Person", "student", name="Peter", age=25, gender="male")
    aFriendRel = Relationship(kary, 'friends_of', peter)
    aFriendRel['years'] =2
    graph.push(aFriendRel)

def search():
    result = graph.find_one(
        label="Person",
        property_key="gender",
        property_value="female"
    )
    print(result)
    print(result['name'])


graph = Graph("localhost:7474", username = "neo4j", password = "112358")
cleanDB()
addNodeRel()
showAll()
update()
showAll()
search()
```

Any Q?

Import jar file for Java

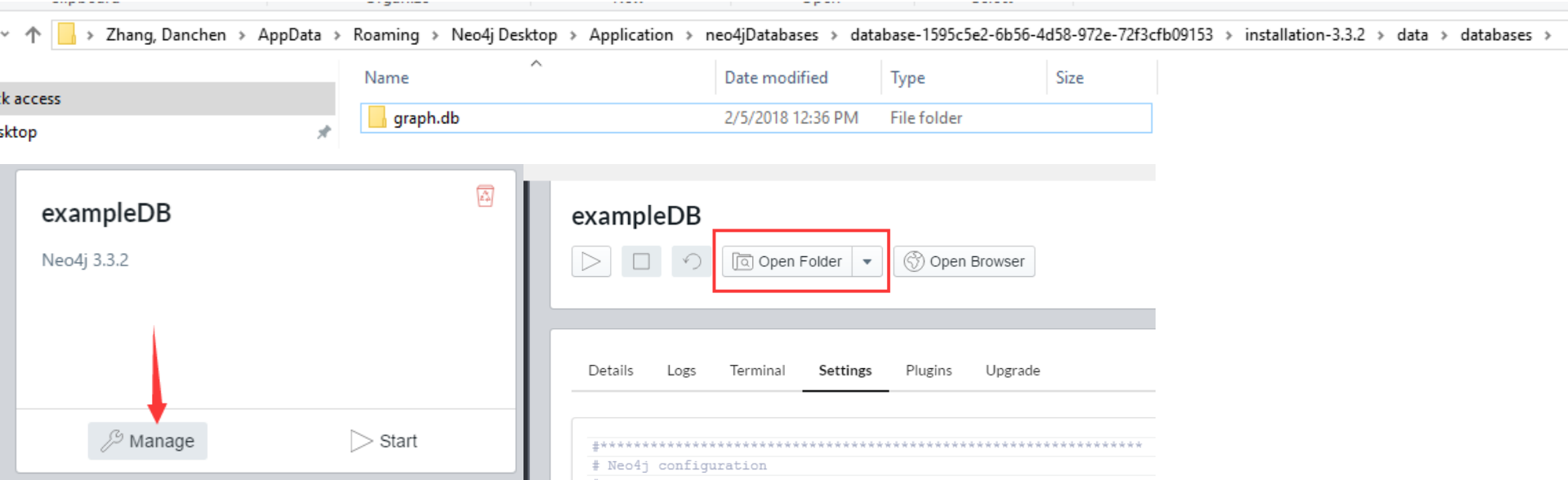
- Right click on your project in eclipse => Build Path => Add external jars => import all jar files in the lib folder.





Connect Neo4j with Java

- Close the neo4j server before running Java.
- GraphDatabaseFactory dbFactory = new GraphDatabaseFactory();
- GraphDatabaseService db = dbFactory.newEmbeddedDatabase(new File(“C:\\Users\\.....\\installation-3.3.2\\data\\databases\\graph.db"));



```java
import java.io.File;

import org.neo4j.graphdb.GraphDatabaseService;
import org.neo4j.graphdb.Label;
import org.neo4j.graphdb.Node;
import org.neo4j.graphdb.Relationship;
import org.neo4j.graphdb.RelationshipType;
import org.neo4j.graphdb.Result;
import org.neo4j.graphdb.Transaction;
import org.neo4j.graphdb.factory.GraphDatabaseFactory;

public class Neo4jExample {

    public static void main(String[] args) {
        Neo4jExample abc = new Neo4jExample();
        abc.cleanDB();
        abc.addNodeRel();
        abc.update("alia", "reading");
    }

    public enum NodeLabelSet implements Label {
        person, student, player, officer, mother;
    }

    public enum RelationshipLabelSet implements RelationshipType {
        friends_of, like;
    }

    GraphDatabaseService db;

    public Neo4jExample() {
        GraphDatabaseFactory dbFactory = new GraphDatabaseFactory();
        db = dbFactory.newEmbeddedDatabase(new File(
                "C:\\Users\\DAZ45\\AppData\\Roaming\\Neo4j Desktop\\A

    }
```

```
38  public void addNodeRel() {
39      try (Transaction tx = db.beginTx()) {
40          Node peter = db.createNode(NodeLabelSet.person, NodeLabelSet.student);
41          peter.setProperty("gender", "male");
42          peter.setProperty("name", "Peter");
43          peter.setProperty("age", 25);
44
45          Node kelly = db.createNode(NodeLabelSet.person, NodeLabelSet.player);
46          kelly.setProperty("gender", "female");
47          kelly.setProperty("name", "Kelly");
48          kelly.setProperty("age", 20);
49
50          Node kary = db.createNode(NodeLabelSet.person, NodeLabelSet.student);
51          kary.setProperty("gender", "female");
52          kary.setProperty("name", "Kary");
53          kary.setProperty("age", 25);
54
55          Node alia = db.createNode(NodeLabelSet.person, NodeLabelSet.officer);
56          alia.setProperty("gender", "female");
57          alia.setProperty("name", "alia");
58          alia.setProperty("age", 23);
59
60          Node alia2 = db.createNode(NodeLabelSet.person, NodeLabelSet.officer);
61          alia2.setProperty("gender", "female");
62          alia2.setProperty("name", "alia2222");
63          alia2.setProperty("age", 23);
64
65          Relationship relationship = kary.createRelationshipTo(peter, RelationshipLabelSet.friends_of);
66          relationship.setProperty("years", 2);
67
68          tx.success();
69      }
70      showAll();
71  }
```

```
73      public void update(String people, String hobby) {
74          showAll();
75          db.execute("match(n) where n.name=\"" + people + "\" set n.hobby=\"" + hobby + "\"");
76          showAll();
77      }
78
79      public void cleanDB() {
80          Result execResult = db.execute("match (n) detach delete n");
81          System.out.println(execResult.resultAsString());
82          showAll();
83      }
84
85      public void showAll() {
86          Result execResult = db.execute("MATCH (n) RETURN n");
87          String results = execResult.resultAsString();
88          System.out.println(results);
89      }
90
91  }
```

Any Q?