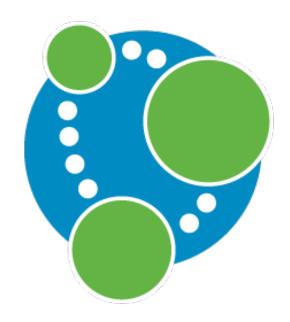
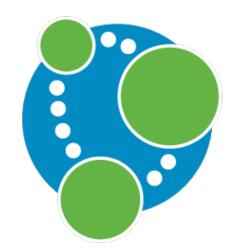


Neo4j & Cypher





- Neo4j
- Cypher
- Hands-on





- Neo4j
- Cypher
- Hands-on





Neo4j

What is Neo4j?

- Neo4j is a Database and DBMS
 use it to reliably store information and find it later
- Neo4j's data model is a Labeled Property Graph
- Neo4j is a native graph-based database
 → uses native graph storage that is specifically designed to store and manage graphs
- Neo4j has drivers for the most popular programing languages



Neo4j Desktop, the developers' mission control console

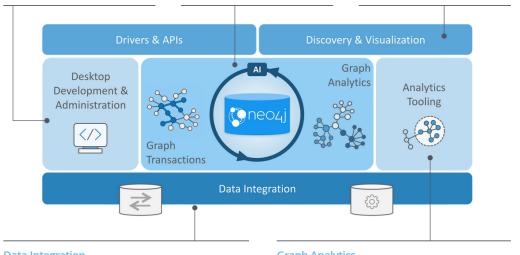
- Free, registered local license of **Enterprise Edition**
- APOC library installer
- Algorithm library installer

Neo4j 3.3 is faster & more secure than Neo4j 3.2

- 50% faster at writes
- Realtime transaction and traversal applications
- Massive scalability

Discovery & Visualization

- Integration with popular visualization vendors
- Neo4j Browser and custom visualizations allow graph exploration



Data Integration

- Neo4j ETL reveals RDBMS hidden relationships upon importing to graph
- Data Importer for fast data ingestion
- Data Lake Integrator materializes graphs from Apache Hadoop, Hive and Spark

Graph Analytics

- **Graph Algorithms** support PageRank, Centrality and Path Finding
- Cypher for Apache Spark from openCypher.org supports graph composition (sub-graphs) and algorithm chaining



Neo4j

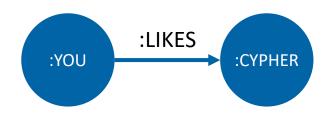
Neo4j and Python

```
# import neo4j module
import neo4j
# create connection/client/driver
driver = neo4j.GraphDatabase.driver("uri", auth=("user", "password"))
# define method to retrieve all friends of a person
def print friends of (tx, name):
    query = "MATCH (a:Person) - [:KNOWS] -> (f) WHERE a.name={name} RETURN f.name"
    for record in tx.run(query, name=name):
        print(record["f.name"])
# get all friends of "Alice"
with driver.session() as session:
    session.read transaction(print friends of, "Alice")
# close connection to the database
driver.close()
```





- Neo4j
- Cypher
- Hands-on



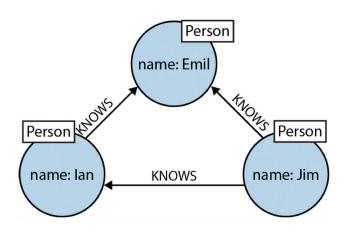
(:YOU) -[:LIKES] ->(:CYPHER)



What is Cypher?

- Cypher is Neo4j's graph query language (SQL for graphs!)
- Cypher is a declarative query language: it describes what you are interested in, not how it is acquired
- Cypher is meant to be very readable and expressive

Cypher is inspired by a number of different approaches and builds upon established practices for expressive querying. Many of the keywords like WHERE and ORDER BY are inspired by SQL. Pattern matching borrows expression approaches from SPARQL. Some of the collection semantics have been borrowed from languages such as Haskell and Python.



```
MATCH (a:Person {name:'Jim'})-[:KNOWS]->(b)-[:KNOWS]->(c), (a)-[:KNOWS]->(c)
RETURN b, c
```



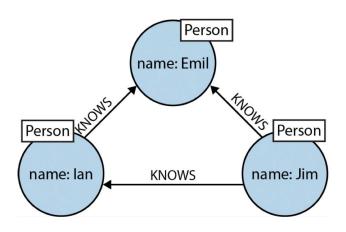
Basic Syntax

Nodes are «drawn» in parentheses:

()

• Edges are «drawn» using pairs of dashes with greater-than or less-than signs and square brackets in between:

```
-[]->
```



```
()-[]->()

(:Person)-[:KNOWS]->(:Person)

with Labels

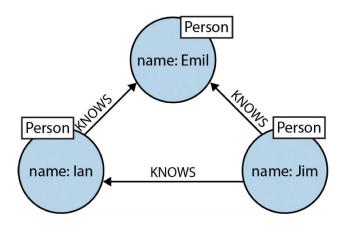
(:Person {name:'Jim'})-[:KNOWS]->(:Person {name:'Ian'})

with Labels and Properties
```



Example (1/2)

The MATCH clause



Result

b

Emil

lan

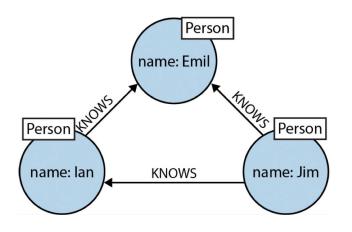
```
MATCH (a:Person {name:'Jim'})-[:KNOWS]->(b)
RETURN b
```

```
MATCH (a:Person {name:'Jim'})-[:KNOWS]->(b)-[:KNOWS]->(c), (a)-[:KNOWS]->(c) RETURN b, c
```



Example (2/2)

The MATCH clause



```
Result

b c

Ian Emil
```

```
MATCH (a:Person {name:'Jim'})-[:KNOWS]->(b)
RETURN b
```

```
MATCH (a:Person {name:'Jim'})-[:KNOWS]->(b)-[:KNOWS]->(c), (a)-[:KNOWS]->(c)
RETURN b, c
```



Neo4j

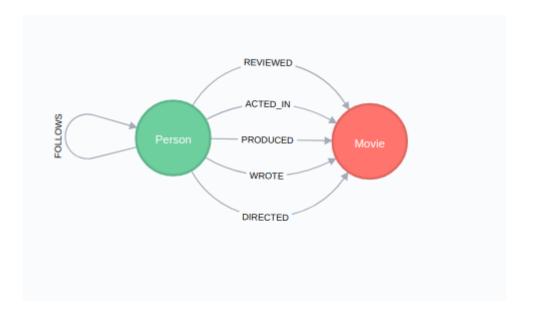
Neo4j and Python

```
# import neo4j module
import neo4j
# create connection/client/driver
driver = neo4j.GraphDatabase.driver("uri", auth=("user", "password"))
# define method to retrieve all friends of a person
def print friends of(tx, name):
    query = "MATCH (a:Person) - [:KNOWS] -> (f) WHERE a.name= { name } RETURN f.name"
    for record in tx.run(query, name=name):
        print(record["f.name"])
# get all friends of "Alice"
with driver.session() as session:
    session.read transaction(print friends of, "Alice")
# close connection to the database
driver.close()
```





- Neo4j
- Cypher
- Hands-on





Hands-on

Neo4j Browser

