DATA CLUB

Eric Olson – Core OS DEP Data Science

Welcome to WDG Data Club!

What is WDG Data Club?

- Training and sharing sessions on Data Science topics
- With hands-on lab or collaboration workshop
- Place for collaborative learning among early career folks and experts

What is the purpose of Data Club?

- Provide training and information on data-focused topics
  - Identify organizational knowledge gaps
  - Provide opportunities to build connections with the WDG Data Science Community (and beyond)

The 1st Rule of Data Club…

- Talk about Data Club
- Participate and learn together
- Share what you learn with others

Looking Ahead

Upcoming talks (Every 2 weeks)

- **Feb 7** Azure Ignite Impacts (tentative)
- **Feb 21** BIG (Business Intelligence Graph) updates

For information and announcements

- join 'dataclub' alias on idweb
  (http://idwebelements/GroupManagement.aspx?Group=dataclub&Operation=join)

Data Club content

- https://aka.ms/DataClub

Please give us Feedback

- Link in follow-up email following session

Introduction to Neo4j and Graph Databases

M. David Allen

Partner Solution Architect at Neo4j
@mdavidallen |david.allen@neo4j.com

What are we going to do today?

- Intro to graphs and Neo4j
- Why Graphs? What are people using Neo4j for?
- Property Graph Data Model
- Querying Graphs: Introduction to Cypher
- Data Import
- Developing Applications

Remote? Follow Along in Real Time

Search Azure Marketplace for Neo4j Enterprise and launch an empty database! (Latest: 3.5.1)

OR

https://neo4jsandbox.com

- Free temporary instance
- Get started with pre-loaded data in minutes

Intro to Graphs

What are we going to do?

- Why graphs?
- What do we use graphs for?
- Intro to the property graph model
- The whiteboard model is the model
- Querying the graph

Why graphs?

The world is a graph – everything is connected

- people, places, events
- companies, markets
- countries, history, politics
- sciences, art, teaching
- technology, networks, machines, applications, users
- software, code, dependencies, architecture, deployments
- criminals, fraudsters and their behavior

The world is a graph – everything is connected

- people, places, events
- companies, markets
- countries, history, politics
- sciences, art, teaching
- technology, networks, machines, applications, users
- software, code, dependencies, architecture, deployments
- criminals, fraudsters and their behavior

The world is a graph – everything is connected

- people, places, events
- companies, markets
- countries, history, politics
- sciences, art, teaching
- technology, networks, machines, applications, users
- software, code, dependencies, architecture, deployments
- criminals, fraudsters and their behavior

What are people using Neo4j for?

Neo4j - Transforming 100s of Large Enterprises

For Over 14 Years

Real-time promotion recommendations

FORTUNE 50 RETAIL

- Record "Cyber Monday" sales
- About 35M daily transactions
- Each transaction is 3-22 hops
- Queries executed in 4ms or less
- Replaced IBM Websphere commerce

Marriott's Real-time Pricing Engine

MARRIOTT

- 300M pricing operations per day
- 10x transaction throughput on half the hardware compared to Oracle
- Replaced Oracle database

Handling Package Routing in Real-Time

- Large postal service with over 500k employees
- Neo4j routes 7M+ packages daily at peak, with peaks of 5,000+ routing operations per second.

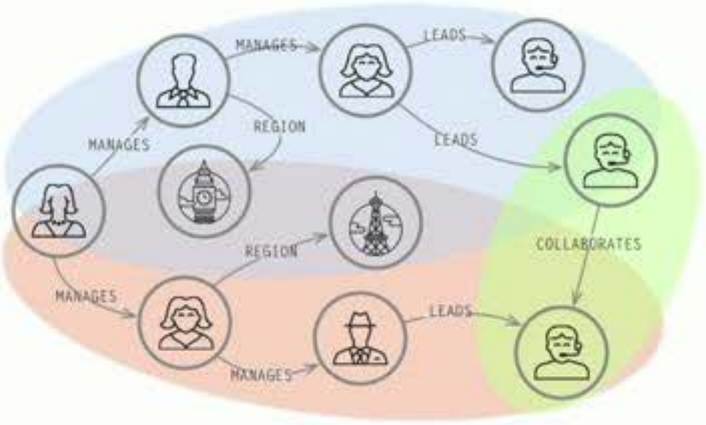
Use Cases

Internal Applications

Master Data Management

Network and
IT Operations

Fraud Detection

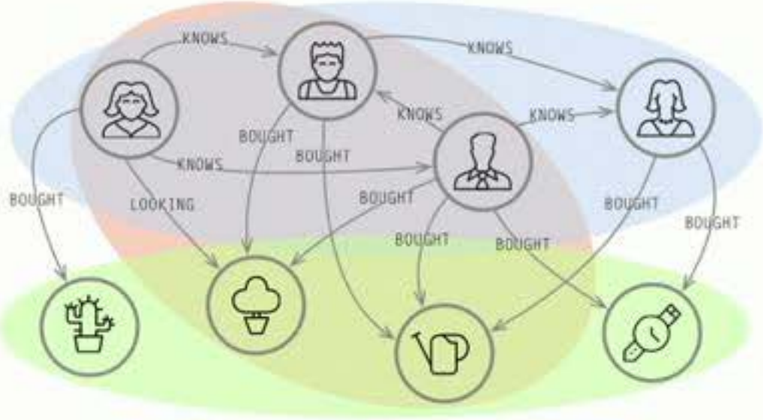


Customer-Facing Applications

Real-Time Recommendations

Graph-Based Search

Identity and
Access Management



Intro to the property graph model

Neo4j Fundamentals

- Nodes
- Relationships
- Properties
- Labels

Property Graph Model Components

Nodes

- Represent the objects in the graph
- Can be *labeled*

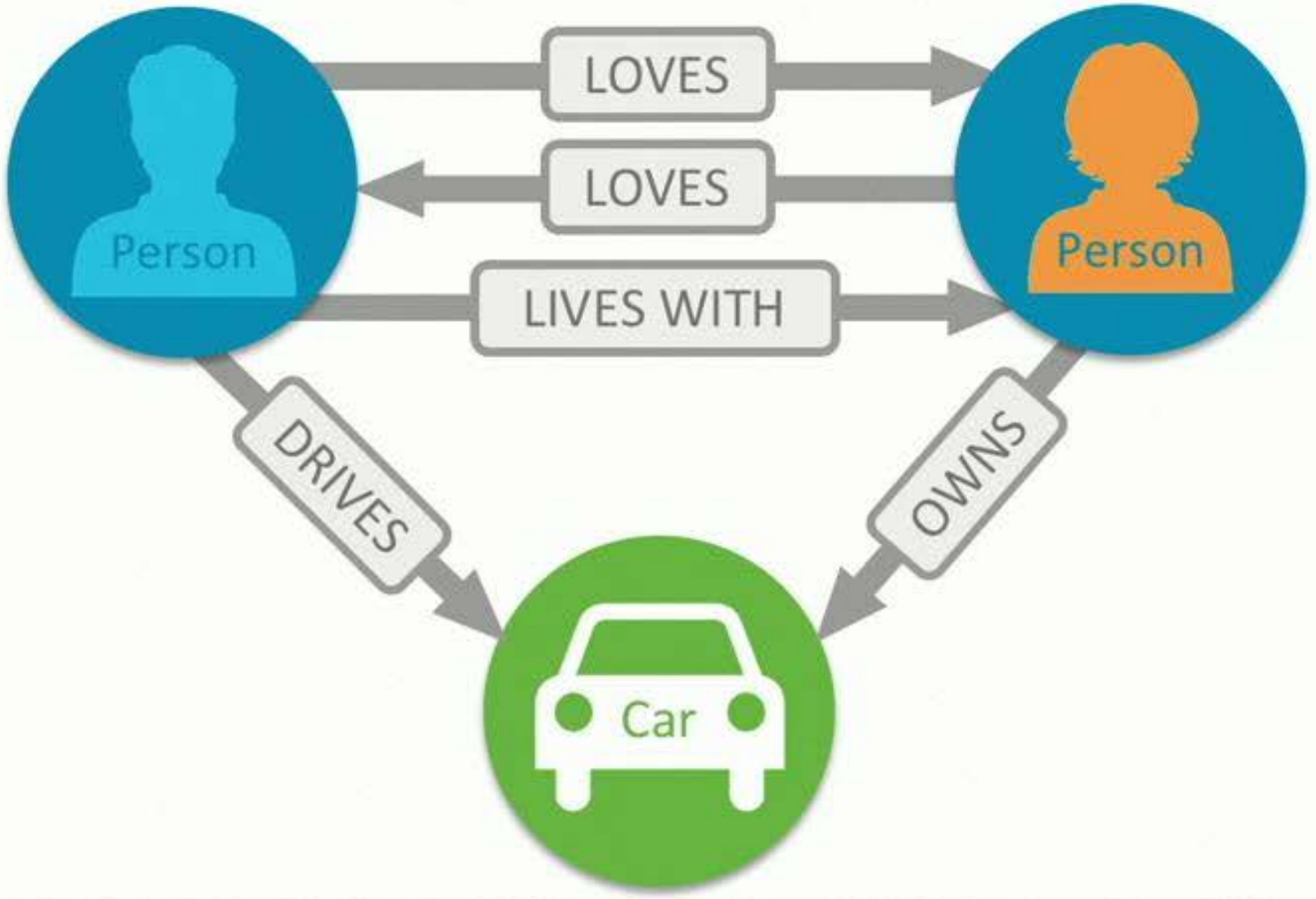
Property Graph Model Components

Nodes

- Represent the objects in the graph
- Can be *labeled*

Relationships

- Relate nodes by *type* and *direction*



Property Graph Model Components

Nodes

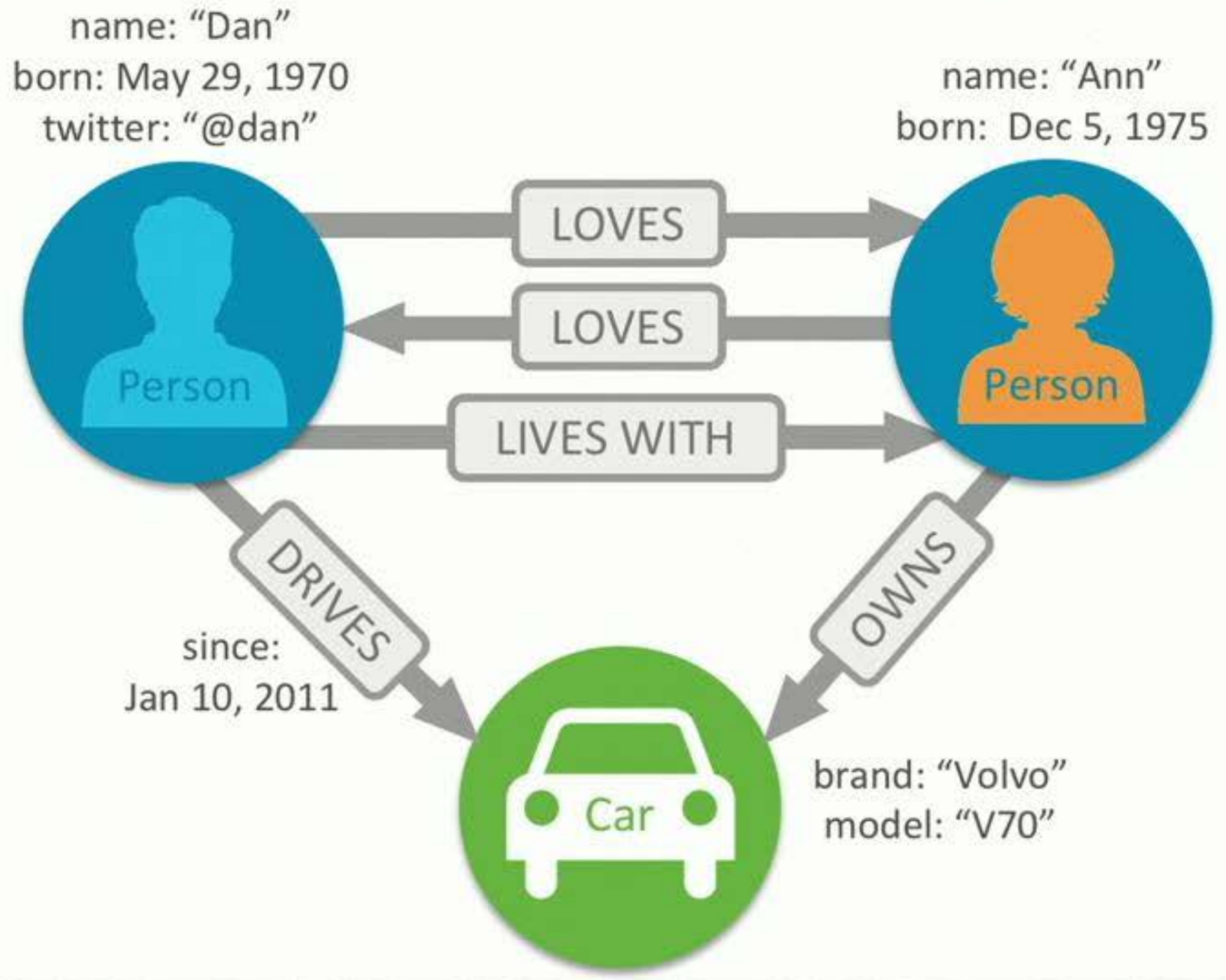
- Represent the objects in the graph
- Can be *labeled*

Relationships

- Relate nodes by *type* and *direction*

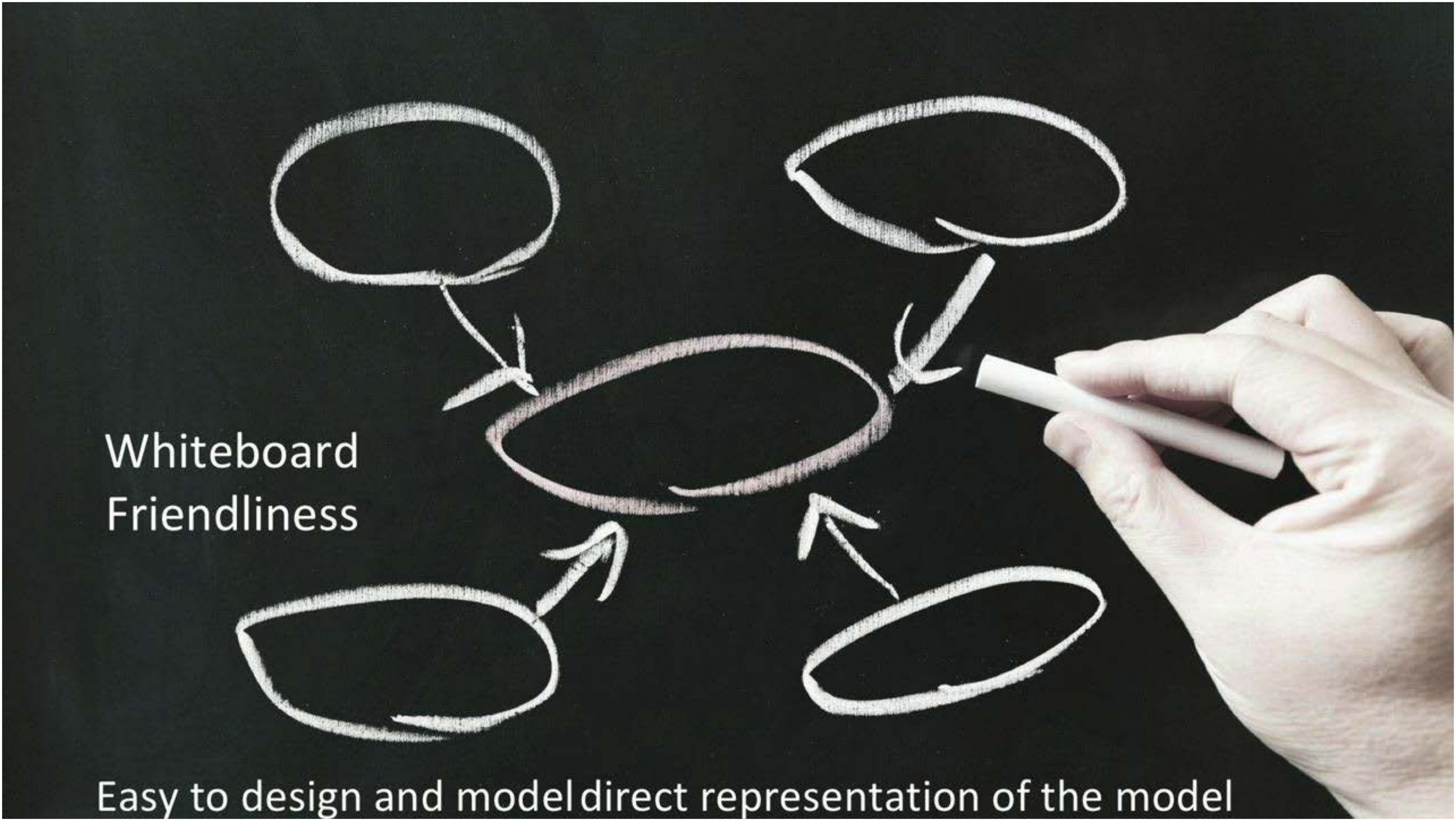
Properties

- Name-value pairs that can go on nodes and relationships.

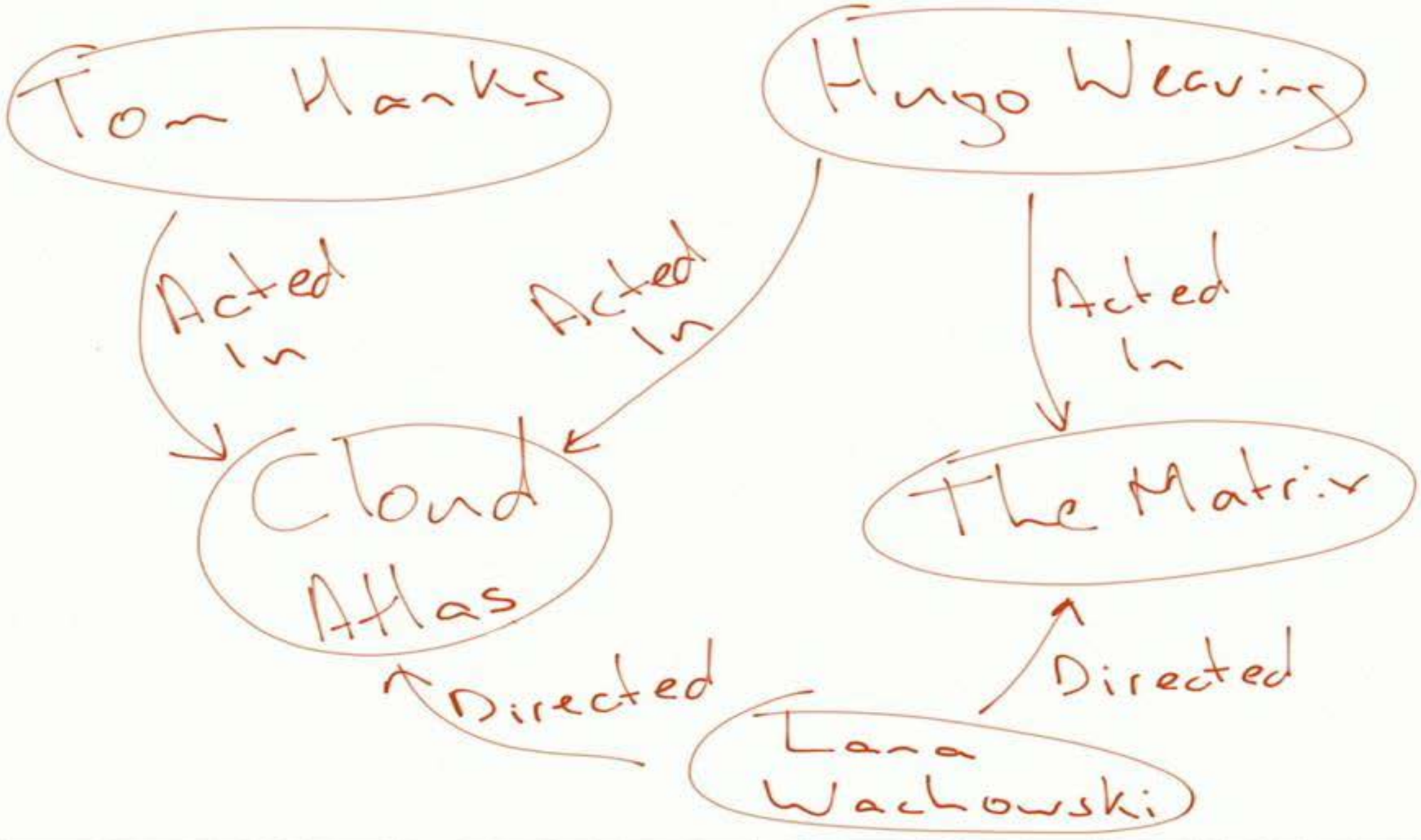


Summary of the graph building blocks

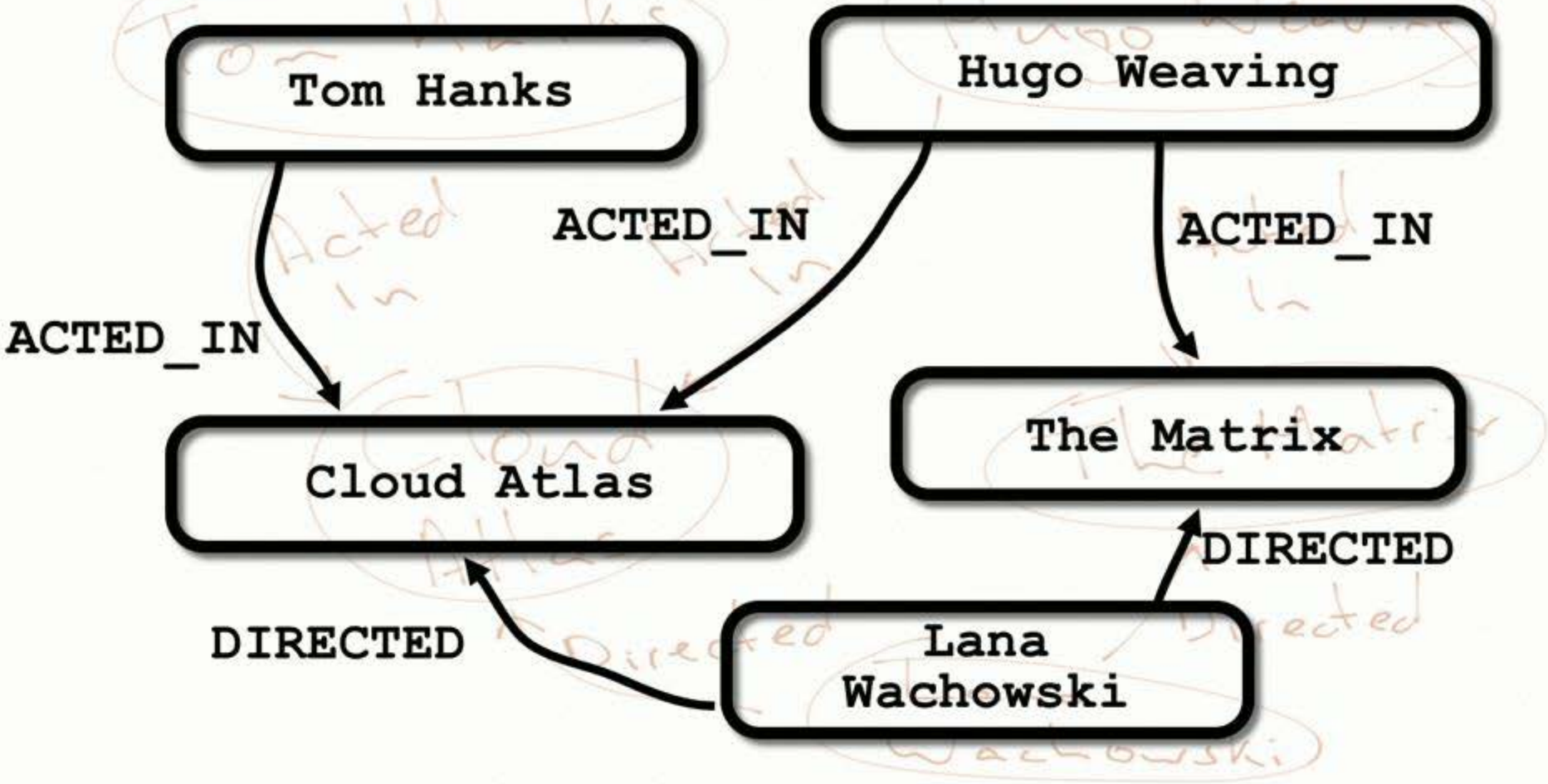
- **Nodes** - Entities and complex value types
- **Relationships** - Connect entities and structure domain
- **Properties** - Entity attributes, relationship qualities, metadata
- **Labels** - Group nodes by role


Whiteboard
Friendliness
Easy to design and model direct representation of the model

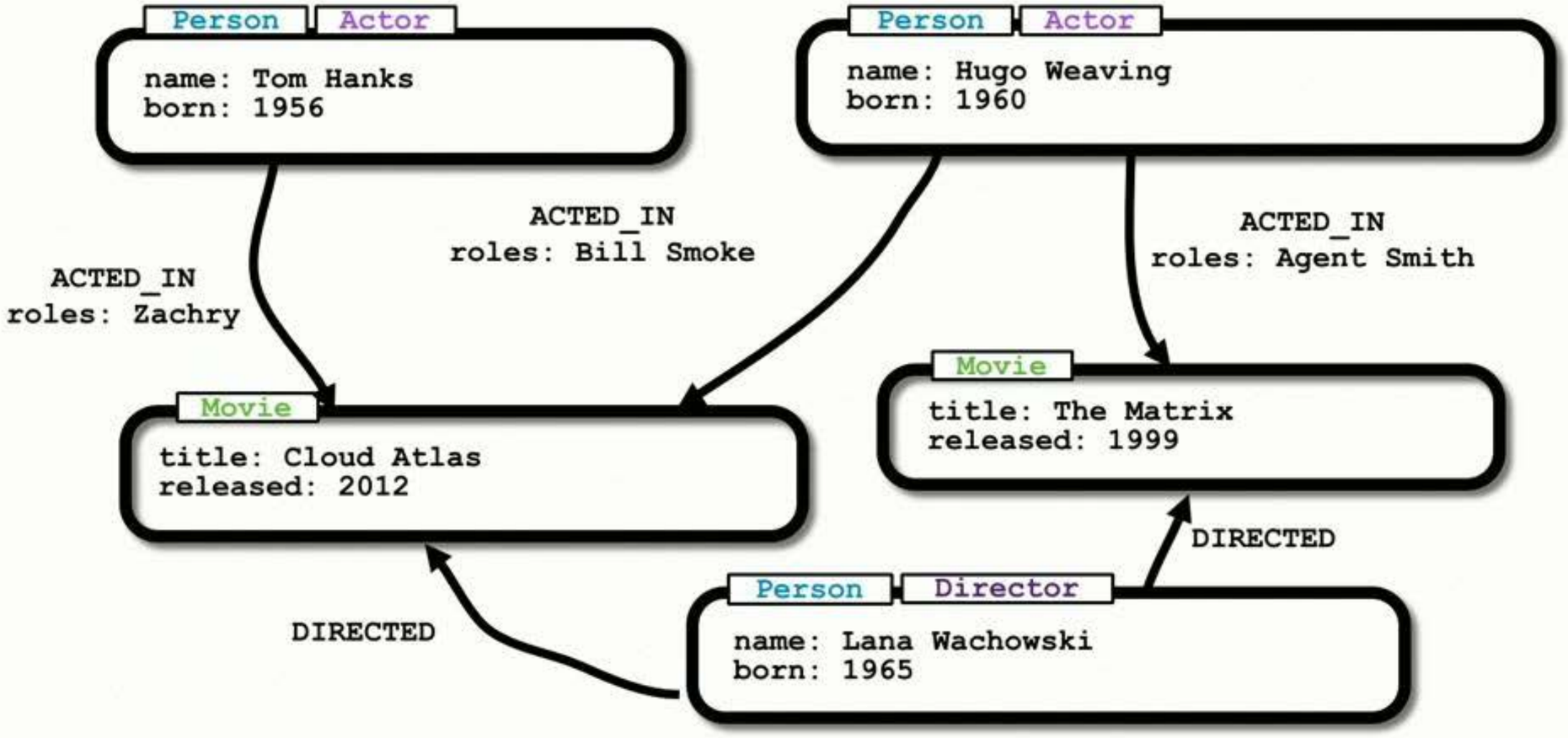
Whiteboard friendliness



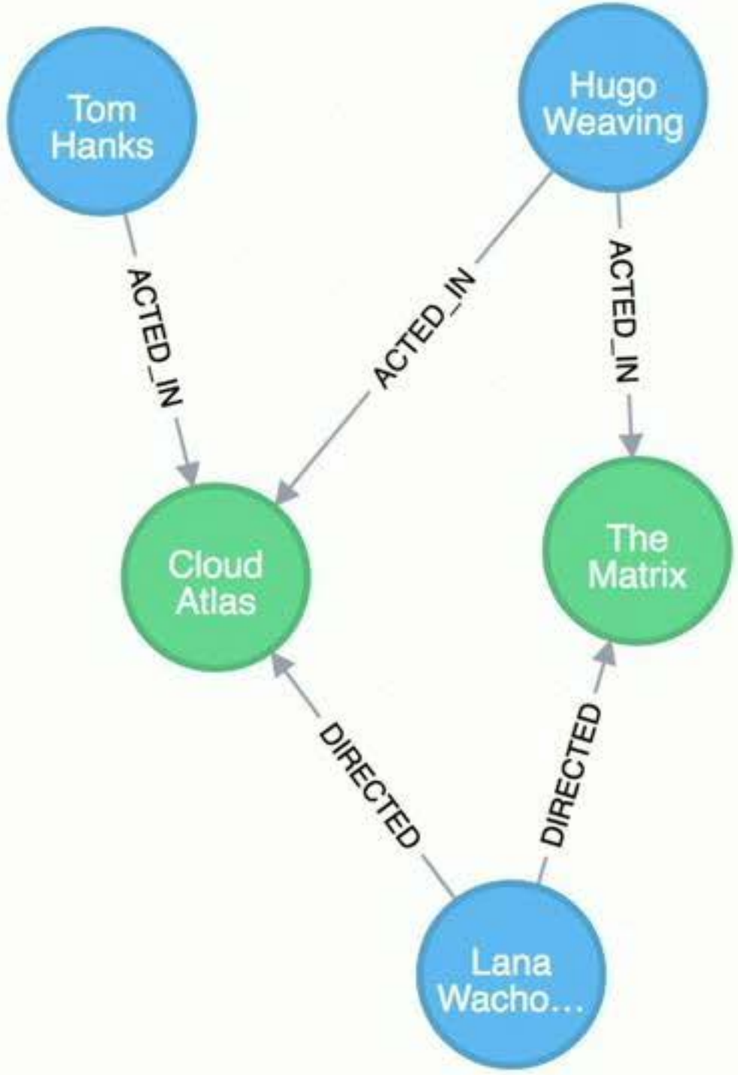
Whiteboard friendliness



Whiteboard friendliness

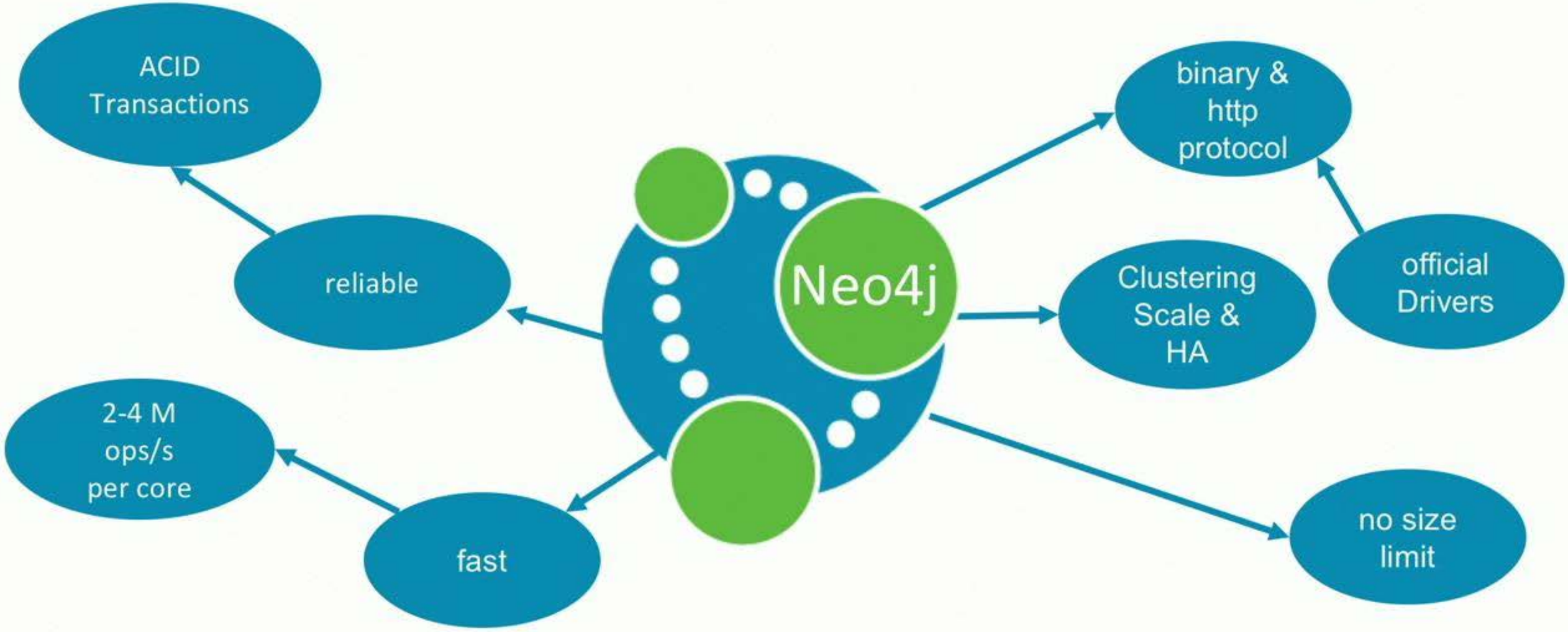


Whiteboard friendliness

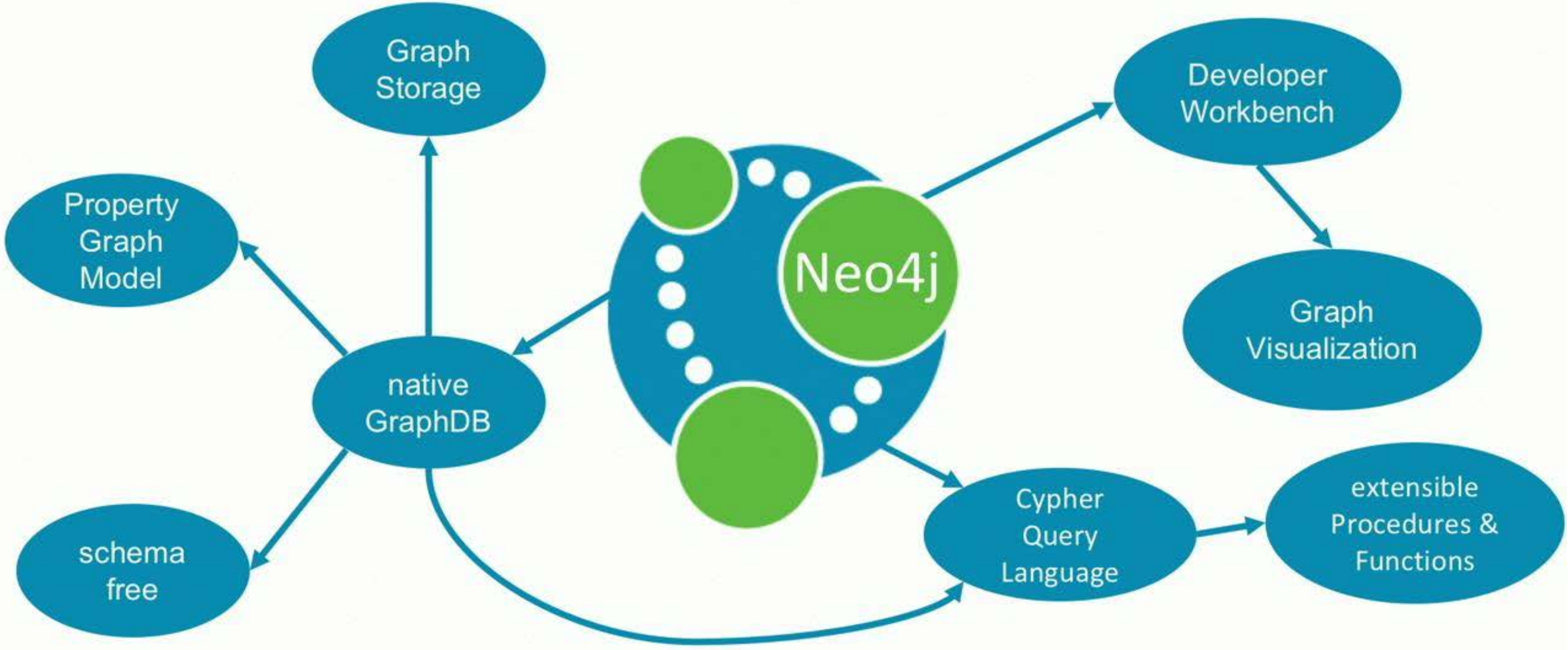


Neo4j is a Graph Database

Neo4j is a database



Neo4j is a graph database



Graph Querying

Cypher

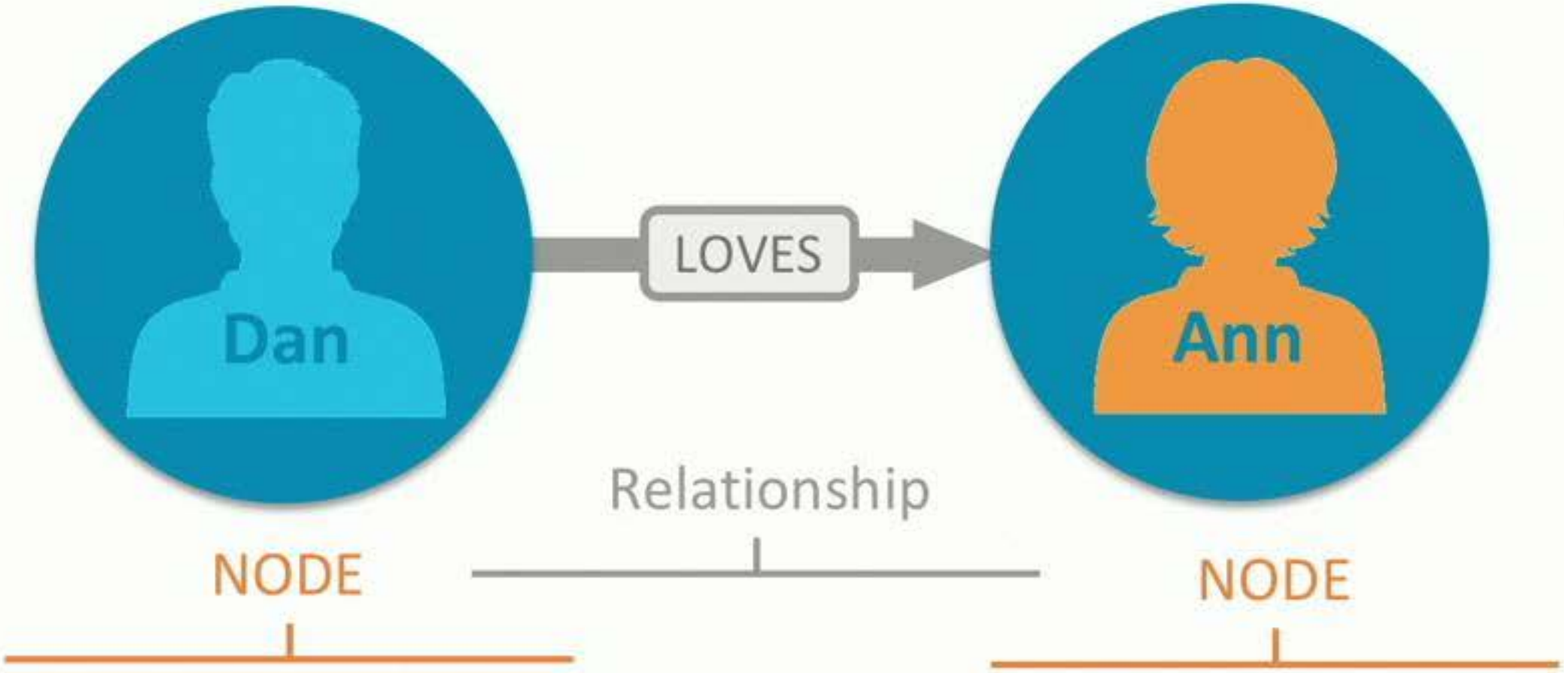
neo4j

A pattern matching query language made for graphs

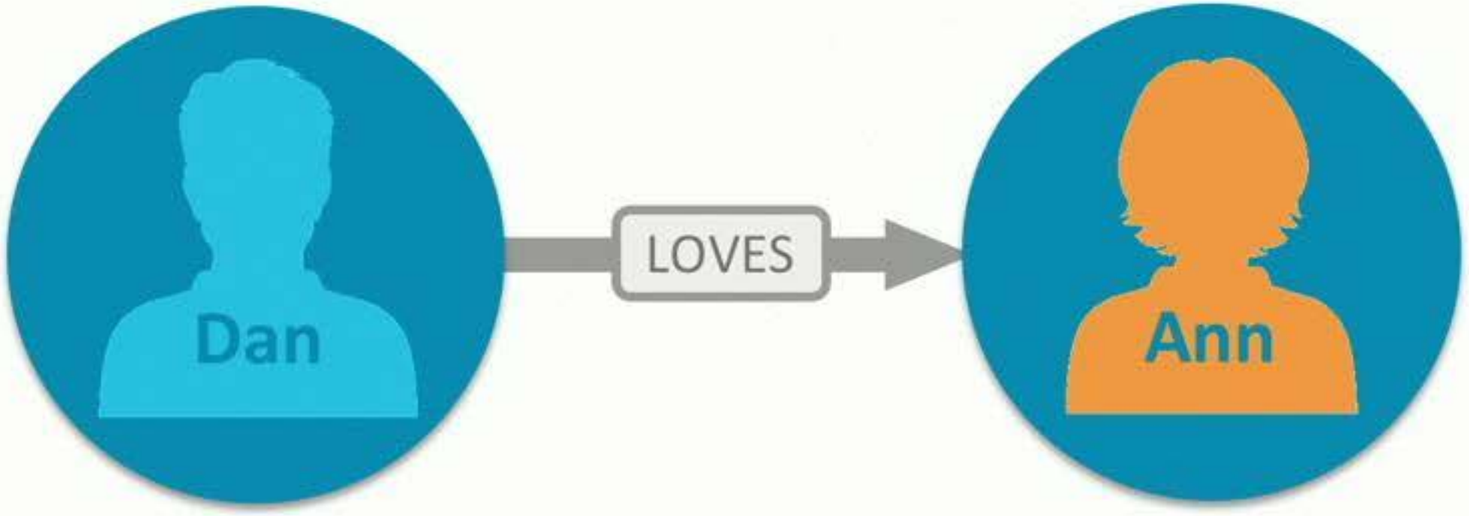
Cypher

- Declarative
- Expressive
- Pattern Matching

Pattern in our Graph Model



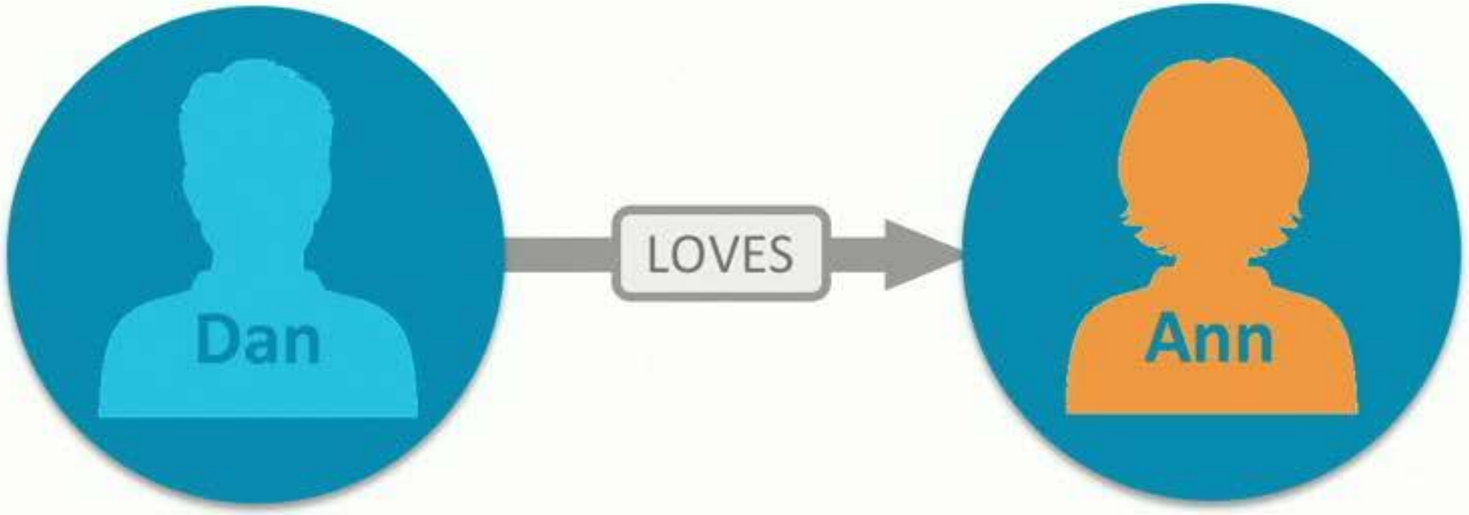
Cypher: Express Graph Patterns



Relationship

```
(:Person { name:"Dan"} ) -[:LOVES]-> (:Person { name:"Ann"} )
```

Cypher: Express Graph Patterns

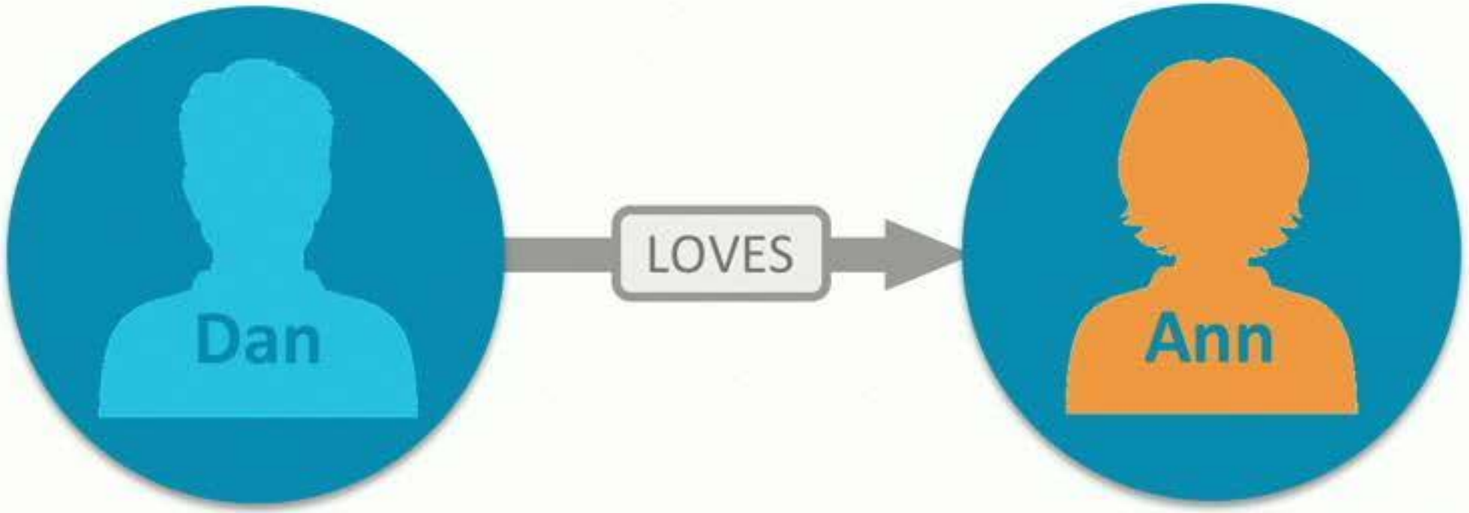


Relationship

```
(:Person { name:"Dan"} ) -[:LOVES]-> (:Person { name:"Ann"} )
```

LABEL PROPERTY LABEL PROPERTY

Cypher: Express Graph Patterns

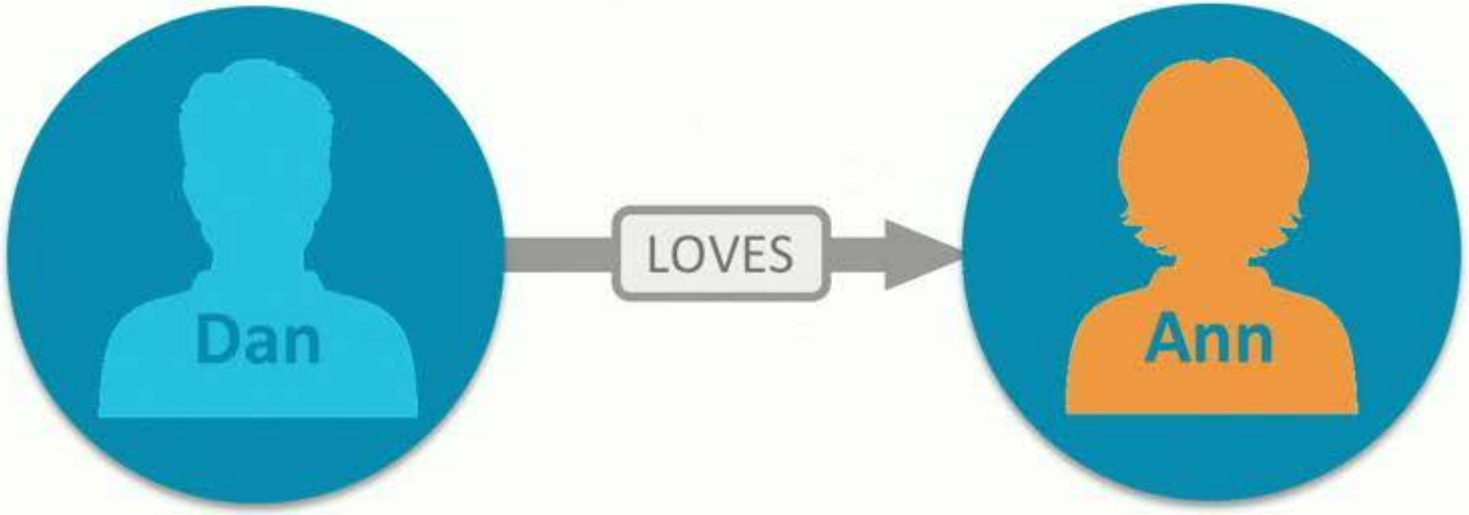


NODE Relationship NODE

```
(:Person { name:"Dan"} ) -[:LOVES]-> (:Person { name:"Ann"} )
```

LABEL PROPERTY LABEL PROPERTY

Cypher: CREATE Graph Patterns

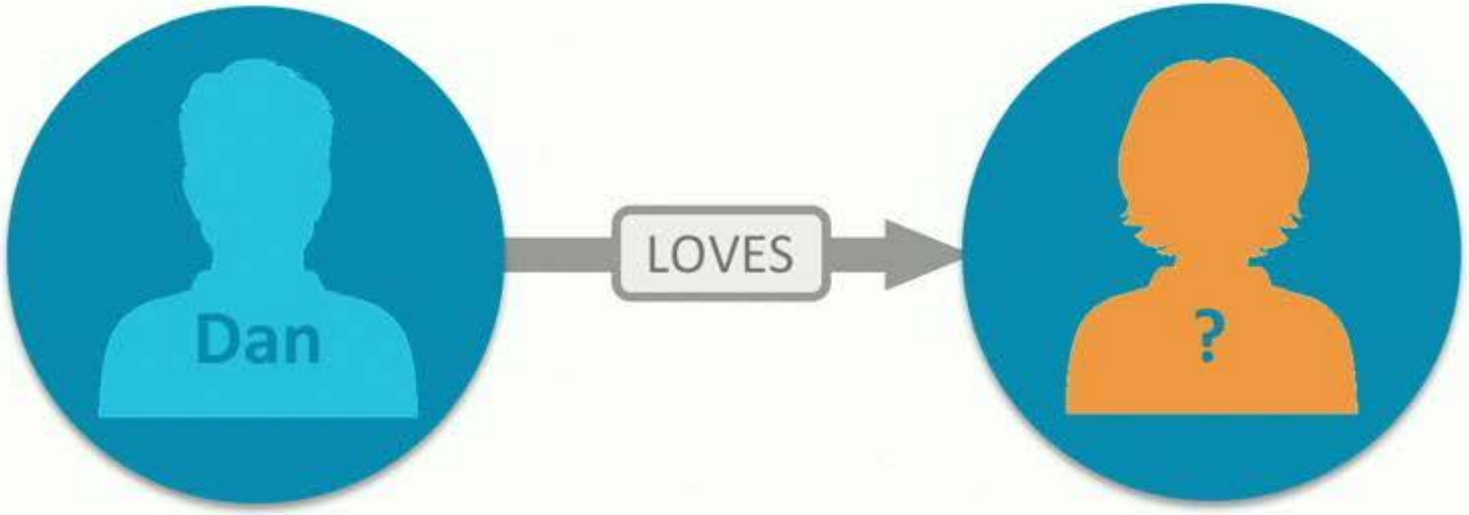


Relationship

```
CREATE (:Person { name:"Dan"} ) -[:LOVES]-> (:Person { name:"Ann"} )
```

LABEL PROPERTY LABEL PROPERTY

Cypher: MATCH Graph Patterns

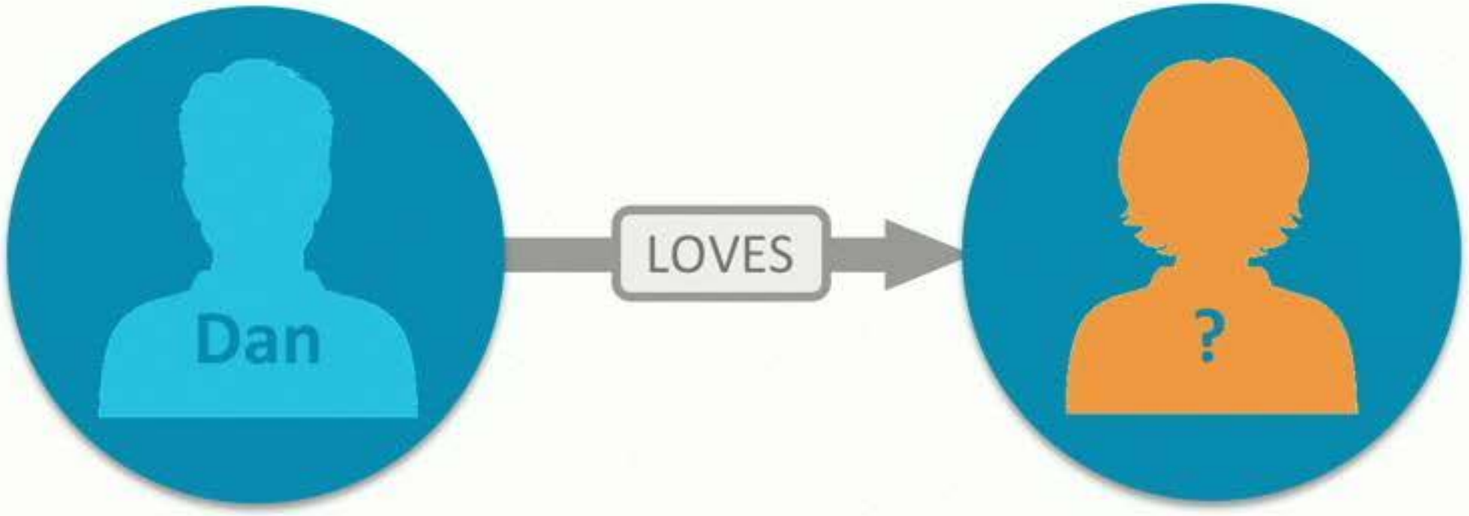


Relationship

```
MATCH (:Person { name:"Dan"} ) -[:LOVES]-> ( whom ) RETURN whom
```

LABEL PROPERTY VARIABLE

Cypher: MATCH Graph Patterns



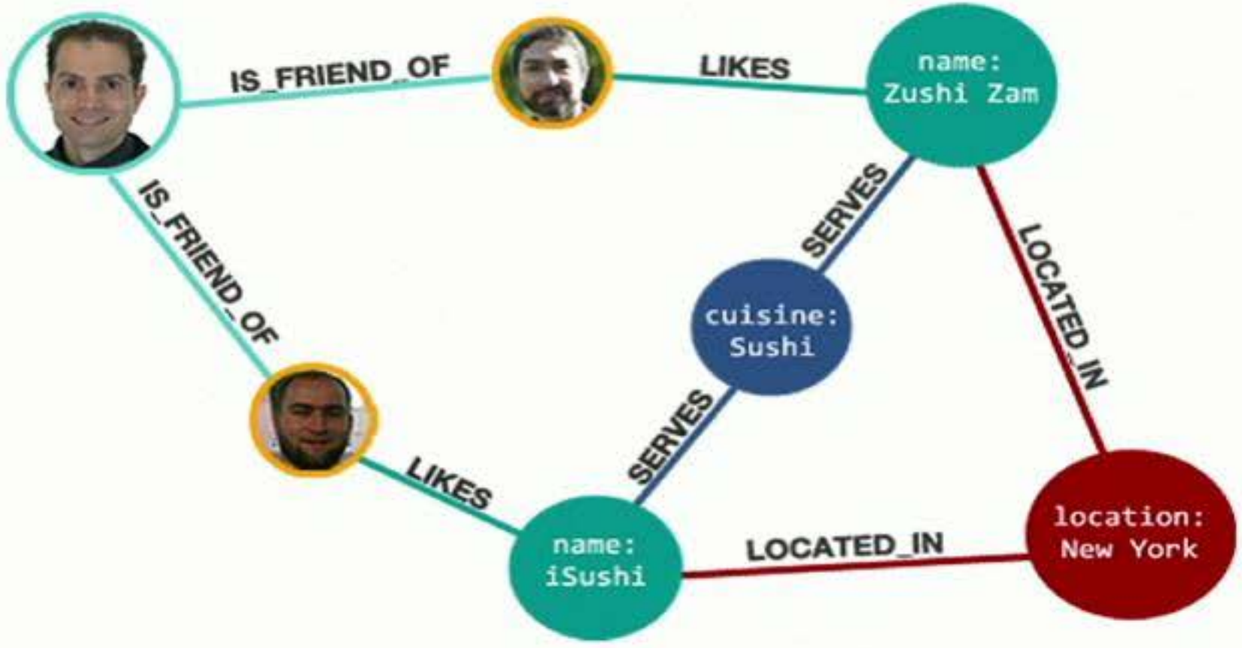
NODE Relationship NODE

```
MATCH (:Person { name:"Dan"} ) -[:LOVES]-> ( whom ) RETURN whom
```

LABEL PROPERTY VARIABLE

A graph query example

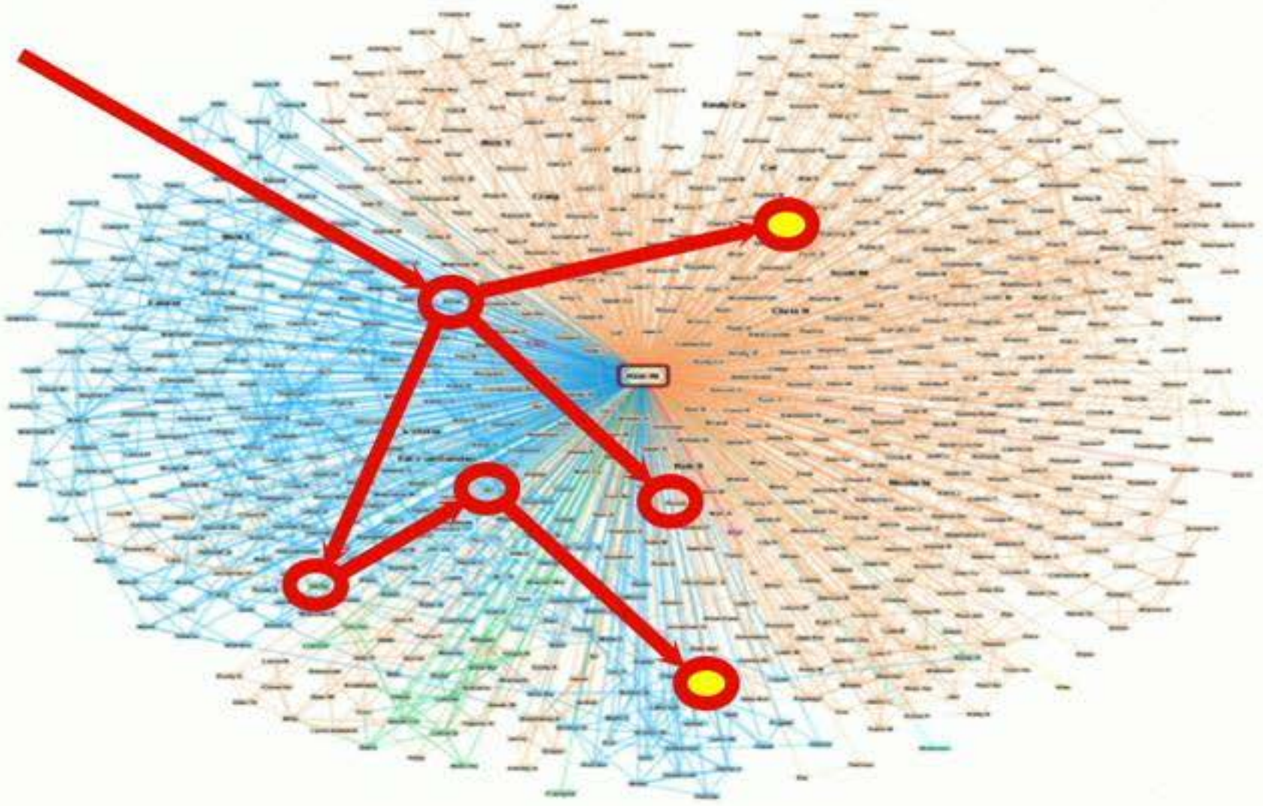
A social recommendation



A social recommendation


```
MATCH (person:Person)-[:IS_FRIEND_OF]->(friend),
      (friend)-[:LIKES]->(restaurant),
      (restaurant)-[:LOCATED_IN]->(loc:Location),
      (restaurant)-[:SERVES]->(type:Cuisine)
WHERE person.name = 'Philip'
AND loc.location='New York'
AND type.cuisine='Sushi'
RETURN restaurant.name
```

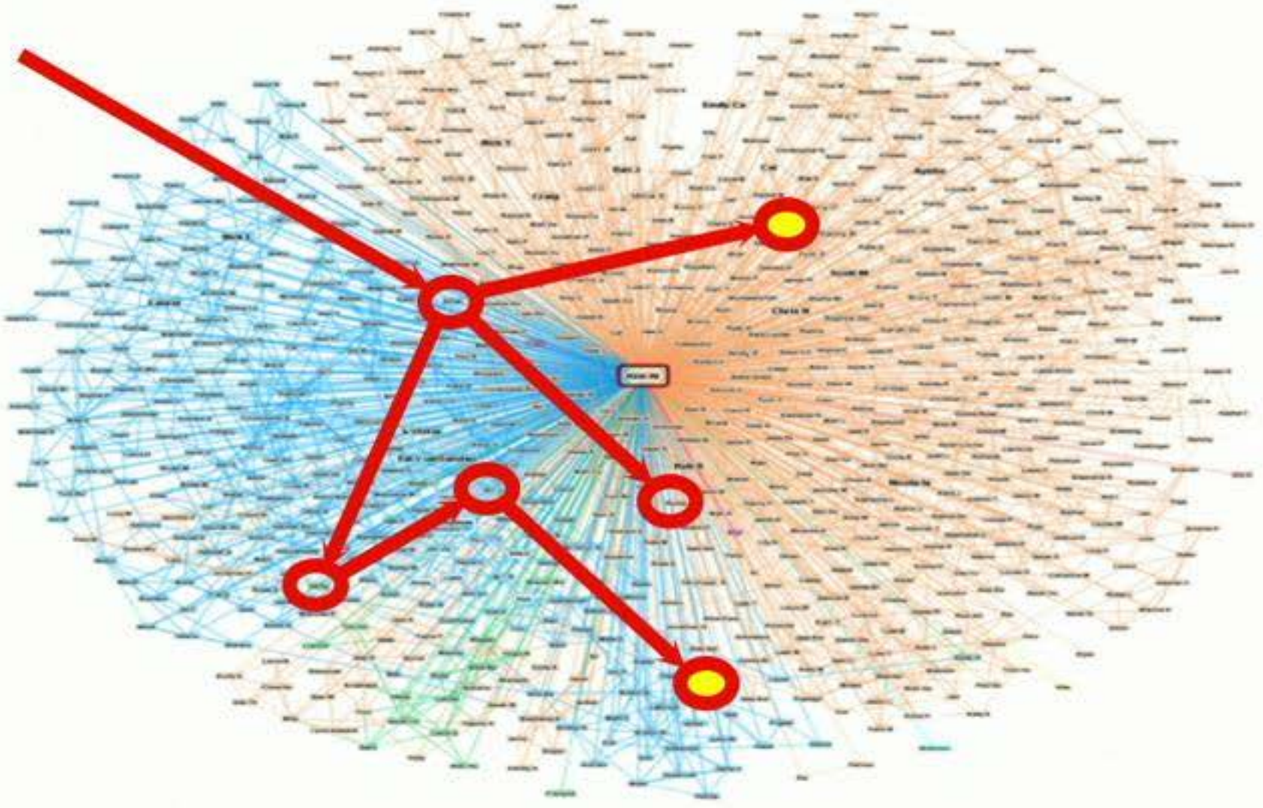
A social recommendation



A social recommendation


```
MATCH (person:Person)-[:IS_FRIEND_OF]->(friend),
      (friend)-[:LIKES]->(restaurant),
      (restaurant)-[:LOCATED_IN]->(loc:Location),
      (restaurant)-[:SERVES]->(type:Cuisine)
WHERE person.name = 'Philip'
AND loc.location='New York'
AND type.cuisine='Sushi'
RETURN restaurant.name
```

A social recommendation



Cypher: Declarative Query for Graphs

Cypher is just ASCII Art

Jay Hartford
@HL_Jayford

Follow

What I learned in Neo4j training today. You draw ascii art to code.

RETWEET 1 LIKE 1

11:15 PM - 15 Sep 2015

1 1

Nodes

Nodes are drawn with parentheses.

( )

Relationships

Relationships are drawn as arrows, with additional detail in brackets.

```
-->
-[:DIRECTED]->
```

Patterns

Patterns are drawn by connecting nodes and relationships with hyphens, optionally specifying a direction with **>** and **<** signs.

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

The components of a Cypher query

```
MATCH (m:Movie)
RETURN m
```

MATCH and **RETURN** are Cypher keywords

m is a variable

:Movie is a node label

The components of a Cypher query

```
MATCH (p:Person)-[r:ACTED_IN]->(m:Movie)
RETURN p, r, m
```

MATCH and **RETURN** are Cypher keywords

p,**r**, and **m** are variables

:Movie is a node label

:ACTED_IN is a relationship type

The components of a Cypher query

```
MATCH path = (:Person)-[:ACTED_IN]->(:Movie)
RETURN path
```

MATCH and **RETURN** are Cypher keywords

path is a variable

:Movie is a node label

:ACTED_IN is a relationship type

Graph versus Tabular results

```
MATCH (m:Movie)
RETURN m
```

Graph versus Tabular results

```
MATCH (m:Movie)
RETURN m.title, m.released
```

Properties are accessed with `{variable}.{property_key}`

Case sensitivity

Case sensitive

Node labels

Relationship types

Property keys

Case insensitive

Cypher keywords

Case sensitivity

Case sensitive	Case insensitive
:Person	MaTcH
:ACTED_IN	return
name	

Aggregates

Aggregates

Aggregate queries in Cypher are a little bit different than in SQL as we don’t need to specify a grouping key.

Aggregates

We implicitly group by any non aggregate fields in the **RETURN** statement.

```
// implicitly groups by p.name
MATCH (p:Person)-[:ACTED_IN]->(m:Movie)
RETURN p.name, count(*) AS numberOfMovies
```

Other aggregate functions

There are other aggregate functions as well.

You can see the full list on the Cypher refcard:
neo4j.com/docs/cypher-refcard/current/

Aggregation
`count(*)` The number of matching rows.
`count(variable)` The number of non-`NULL` values.
`count(DISTINCT variable)` All aggregation functions also take the `DISTINCT` modifier, which removes duplicates from the values.
`collect(n.property)` List from the values, ignores `NULL`.
`sum(n.property)` Sum numerical values. Similar functions are `avg`, `min`, `max`.
`percentileDisc(n.property, {percentile})` Discrete percentile. Continuous percentile is `percentileCont`. The `percentile` argument is from `0.0` to `1.0`.
`stdev(n.property)` Standard deviation for a sample of a population. For an entire population use `stdevp`.

Constraints and Indexes

Unique Constraints

We create unique constraints to:

- ensure uniqueness
- allow fast lookup of nodes which match label-property pairs.

```
CREATE CONSTRAINT ON (label:Label)
ASSERT label.property IS UNIQUE
```

Unique Constraints

There are three types of unique constraints:

- Unique node property constraint

```
CREATE CONSTRAINT ON (label:Label)
ASSERT label.property IS UNIQUE
```

Unique Constraints

There are three types of unique constraints:

- Unique node property constraint
- Node property existence constraint

```
CREATE CONSTRAINT ON (label:Label)
ASSERT EXISTS(label.name)
```

Unique Constraints

There are three types of unique constraints:

- Unique node property constraint
- Node property existence constraint
- Relationship property existence constraint

```
CREATE CONSTRAINT ON ()-[rel:REL_TYPE]->()
ASSERT EXISTS(rel.name)
```

Indexes

We create indexes to:

- allow fast lookup of nodes which match label-property pairs.

```
CREATE INDEX ON :Label(property)
```

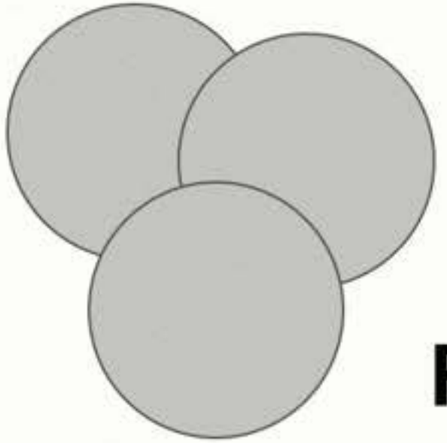
What are these fast lookups?

The following predicates use indexes:

- Equality
- `STARTS WITH`
- `CONTAINS`
- `ENDS WITH`
- Range searches
- (Non-) existence checks

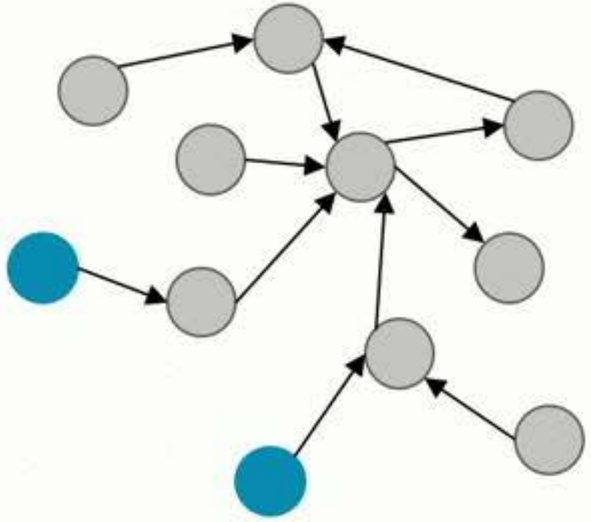
How are indexes used in neo4j?

Indexes are **only** used to find the **starting points** for queries.



Relational

Use index scans to look up rows in tables and join them with rows from other tables



Graph

Use indexes to find the starting points for a query.

The MERGE Clause

```
MERGE (p:Person {name: 'Tom Hanks', oscar: true})
RETURN p
```

There is not a `:Person` node with `name:'Tom Hanks'` and `oscar:true` in the graph, but there is a `:Person` node with `name:'Tom Hanks'`.

What do you think will happen here?

Write queries

The CREATE Clause

```
CREATE (m:Movie {title:'Mystic River', released:2003})
RETURN m
```

The SET Clause


```
MATCH (m:Movie {title: 'Mystic River'})
SET m.tagline = 'We bury our sins here, Dave. We wash them clean.'
RETURN m
```

The CREATE Clause


```
MATCH (m:Movie {title: 'Mystic River'})
MATCH (p:Person {name: 'Kevin Bacon'})
CREATE (p)-[r:ACTED_IN {roles: ['Sean']}]->(m)
RETURN p, r, m
```

The MERGE Clause

```
MERGE (p:Person {name: 'Tom Hanks'})
RETURN p
```

The MERGE Clause

```
MERGE (p:Person {name: 'Tom Hanks', oscar: true})
RETURN p
```

The MERGE Clause

```
MERGE (p:Person {name: 'Tom Hanks'})
SET p.oscar = true
RETURN p
```

The MERGE Clause

```
MERGE (p:Person {name: 'Tom Hanks', oscar: true})
RETURN p
```

The MERGE Clause

```
MERGE (p:Person {name: 'Tom Hanks'})
SET p.oscar = true
RETURN p
```

The MERGE Clause

```
MERGE (p:Person {name: 'Tom Hanks', oscar: true})
RETURN p
```

The MERGE Clause

```
MERGE (p:Person {name: 'Tom Hanks'})
SET p.oscar = true
RETURN p
```

ON CREATE and ON MATCH


```
MERGE (p:Person {name: 'Your Name'})
ON CREATE SET p.created = timestamp(), p.updated = 0
ON MATCH SET p.updated = p.updated + 1
RETURN p.created, p.updated;
```

Data Import with Cypher

ON CREATE and ON MATCH

```
MERGE (p:Person {name: 'Your Name'})
ON CREATE SET p.created = timestamp(), p.updated = 0
ON MATCH SET p.updated = p.updated + 1
RETURN p.created, p.updated;
```

The MERGE Clause

```
MERGE (p:Person {name: 'Tom Hanks'})
SET p.oscar = true
RETURN p
```

What are these fast lookups?

The following predicates use indexes:

- Equality
- `STARTS WITH`
- `CONTAINS`
- `ENDS WITH`
- Range searches
- (Non-) existence checks

Unique Constraints

There are three types of unique constraints:

- Unique node property constraint
- Node property existence constraint
- Relationship property existence constraint

```
CREATE CONSTRAINT ON ()-[rel:REL_TYPE]->()
ASSERT EXISTS(rel.name)
```

Unique Constraints

There are three types of unique constraints:

- Unique node property constraint
- Node property existence constraint

```
CREATE CONSTRAINT ON (label:Label)
ASSERT EXISTS(label.name)
```

The CREATE Clause

```
CREATE (m:Movie {title:'Mystic River', released:2003})
RETURN m
```

LOAD CSV is an ETL Power Tool

- Load CSV data from a **http(s)** or **file** URL
- Provides a **stream** of records (lists or maps)
- Create, update or extend graph structures with the full power of Cypher
- Transactional operations on existing graph in running server
- Transform and convert CSV values
- Allows us to import **into** our graph model
- Up to 10M nodes & relationships work well

This is our CSV Data

`movies.csv`

title	released	tagline
The Matrix	1999	Welcome to the Real World
Cloud Atlas	2012	Everything is connected
Sleepless in Seattle	1993	"What if someone you never met, someone you never saw, someone you never knew was the only someone for you?"
Apollo 13	1995	"Houston, we have a problem."

This is our CSV Data

`people.csv`

name	born
Tom Hanks	1956
Keanu Reeves	1964
Meg Ryan	1961
Clint Eastwood	1930

This is our CSV Data

neo4j

actors.csv

movie	roles	person
Sleepless in Seattle	Sam Baldwin	Tom Hanks
Cloud Atlas	Zachry;Dr. Henry Goose;Isaac Sachs;Dermot Hoggins	Tom Hanks
The Matrix	Neo	Keanu Reeves
Sleepless in Seattle	Annie Reed	Meg Ryan

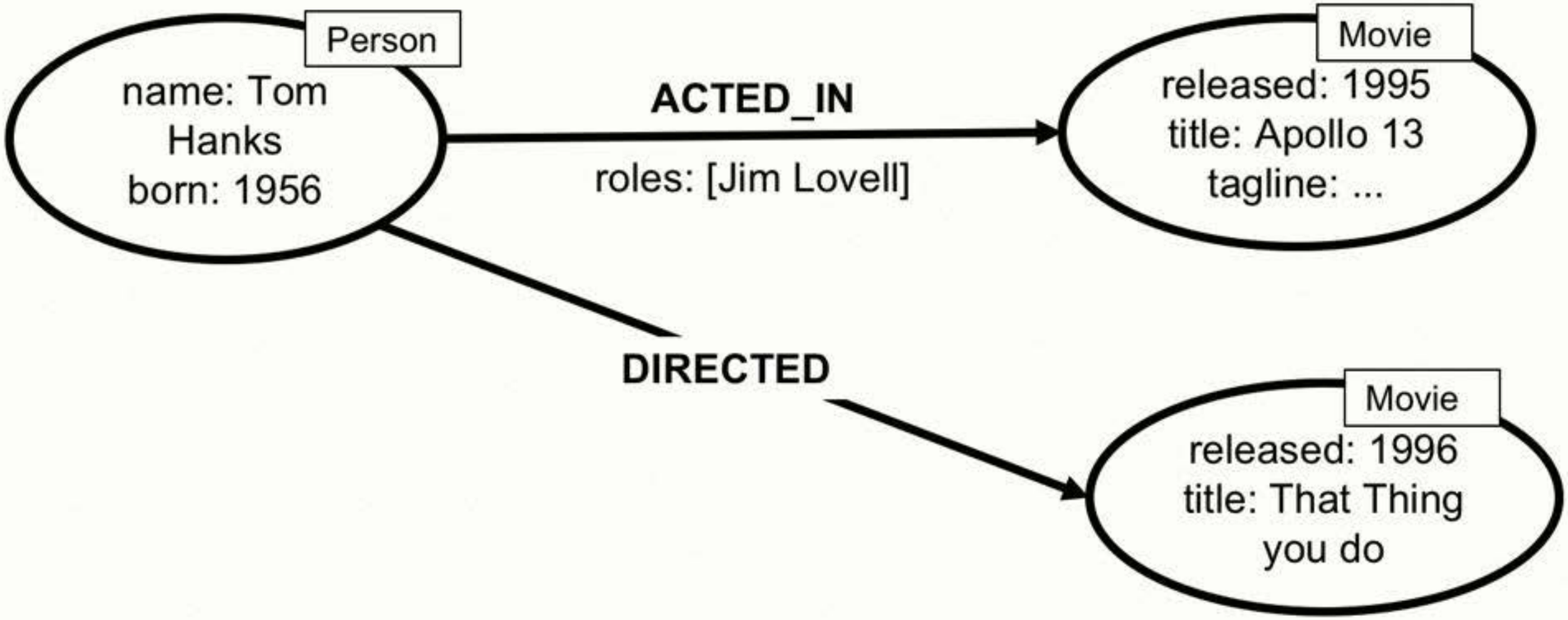
This is our CSV Data

neo4j

`directors.csv`

movie	person
Sleepless in Seattle	Nora Ephron
Cloud Atlas	Lilly Wachowski
Cloud Atlas	Tom Tykwer
The Matrix	Lana Wachowski

Our Data Model: Movies and Actors



Tabular CSV records to Graph structure

for each record from a file:

- either **Create a Node** (Person, Movie)

```
CREATE (:Person {name:row.name, born: toInt(row.born)});
```

- **Find** start **and** end node (Person & Movie)

  Create a Relationship

```
MATCH (m:Movie {title:row.movie}),
      (p:Person {name:row.person})
CREATE (p)-[:ACTED_IN {roles:split(r.roles)} ]->(m);
```

Reading data from CSV with Cypher


```
[USING PERIODIC COMMIT]          // optional transaction batching

LOAD CSV                                          // load csv data

WITH HEADERS     // optionally use first header row as keys in "row" map

FROM "url"       // file:// URL relative to $NEO4J_HOME/import or http://
AS row           // return each row of the CSV as list of strings or map

FIELDTERMINATOR ";" // alternative field delimiter

  ... rest of the Cypher statement ...
```

Tabular CSV records to Graph structure

Type the following command into the query pane in the browser:

```
:play http://guides.neo4j.com/fundamentals/import.html
```

Developing Applications

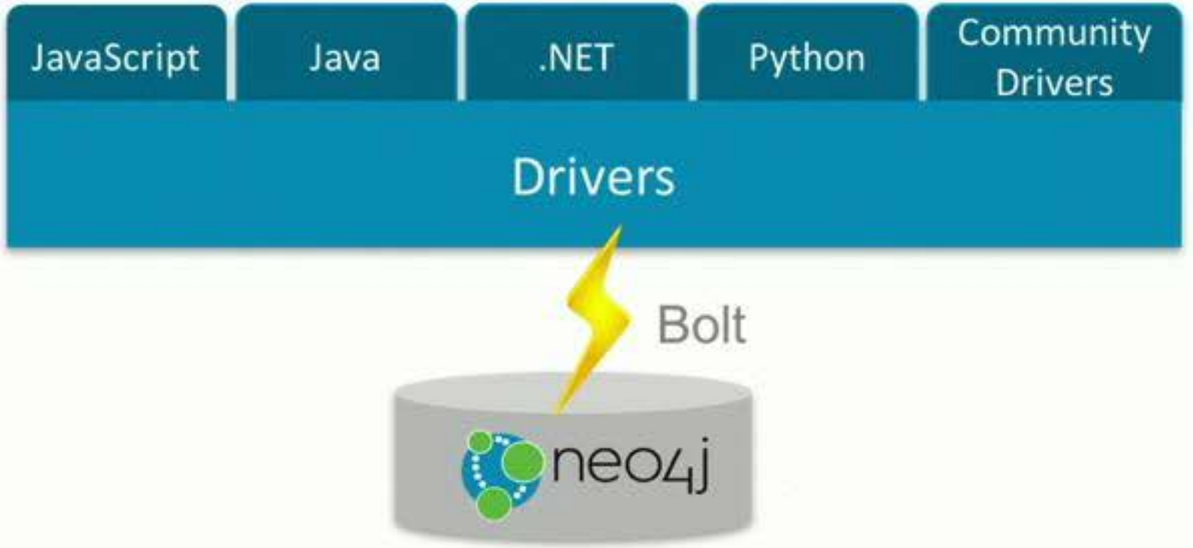
with Neo4j and Cypher

Available APIs

- Remote with Drivers
  - Bolt Binary Protocol (since Neo4j 3.0)
  - Officially Supported Drivers (Go, Java, Python, JS, & others)
  - Cypher transactional HTTP Endpoint
  - neo4j.com/developer/language-guides
- Native Java API
  - User Defined Procedures
  - Execute Cypher
  - Core Java API

Bolt Binary Protocol

- High performance protocol
- Versioned Protocol and API
- Based on Packstream
- Auth and TLS
- Pooling



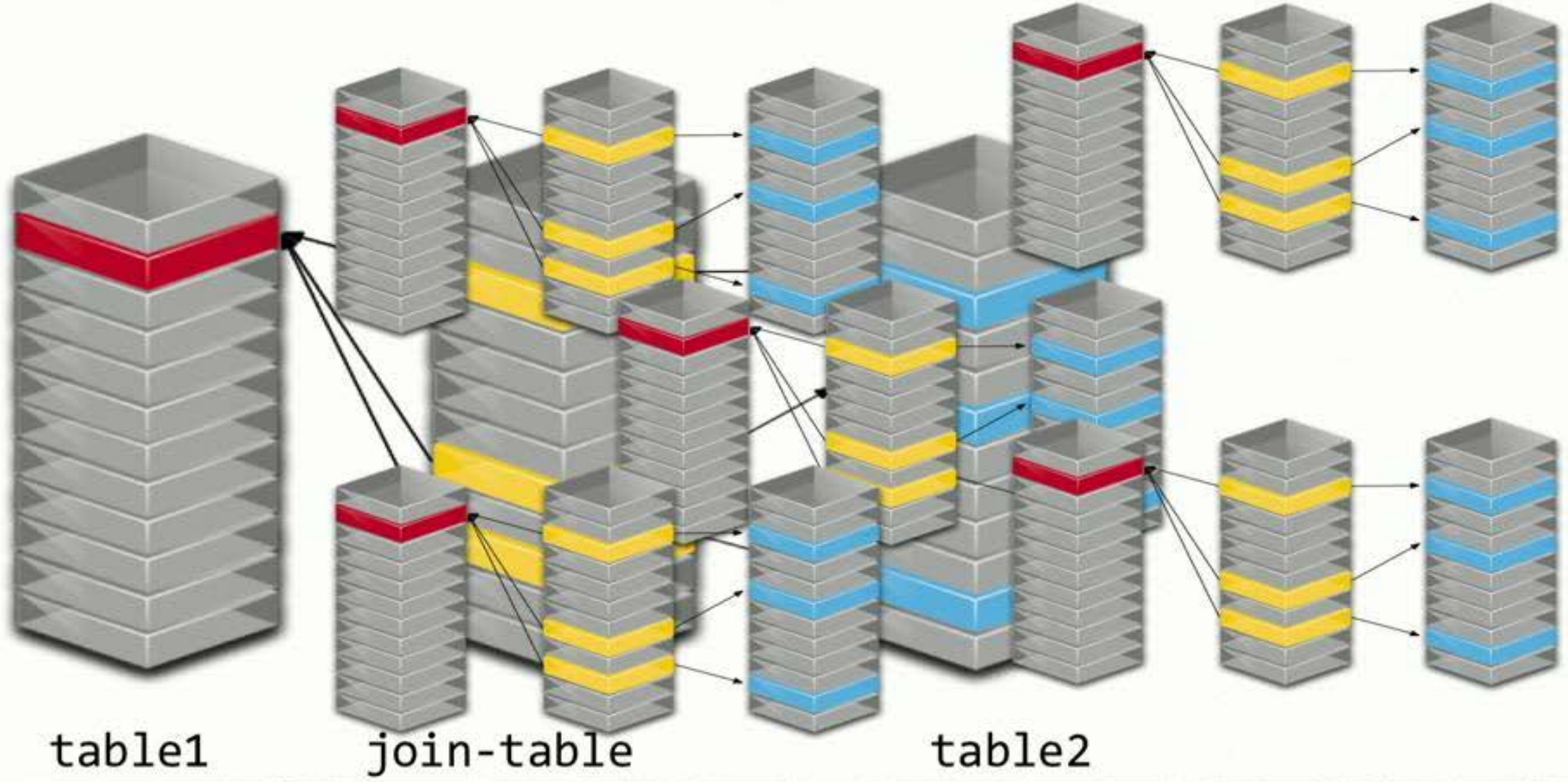
Extend Cypher - User Defined Procedures

- User Defined Procedures
- 80:20 high performance operations
- Methods annotated with `@Procedure`, `@UserFunction (>= 3.1)`, `@UserAggregateFunction (>= 3.2)`
- Procedures Returns Stream of DTOs
- Automatic Transaction Scope
- Callable within Cypher
- Deploy jar in `$NEO4J_HOME/plugins` directory
- Use of embedded Java APIs

Relational to Graph

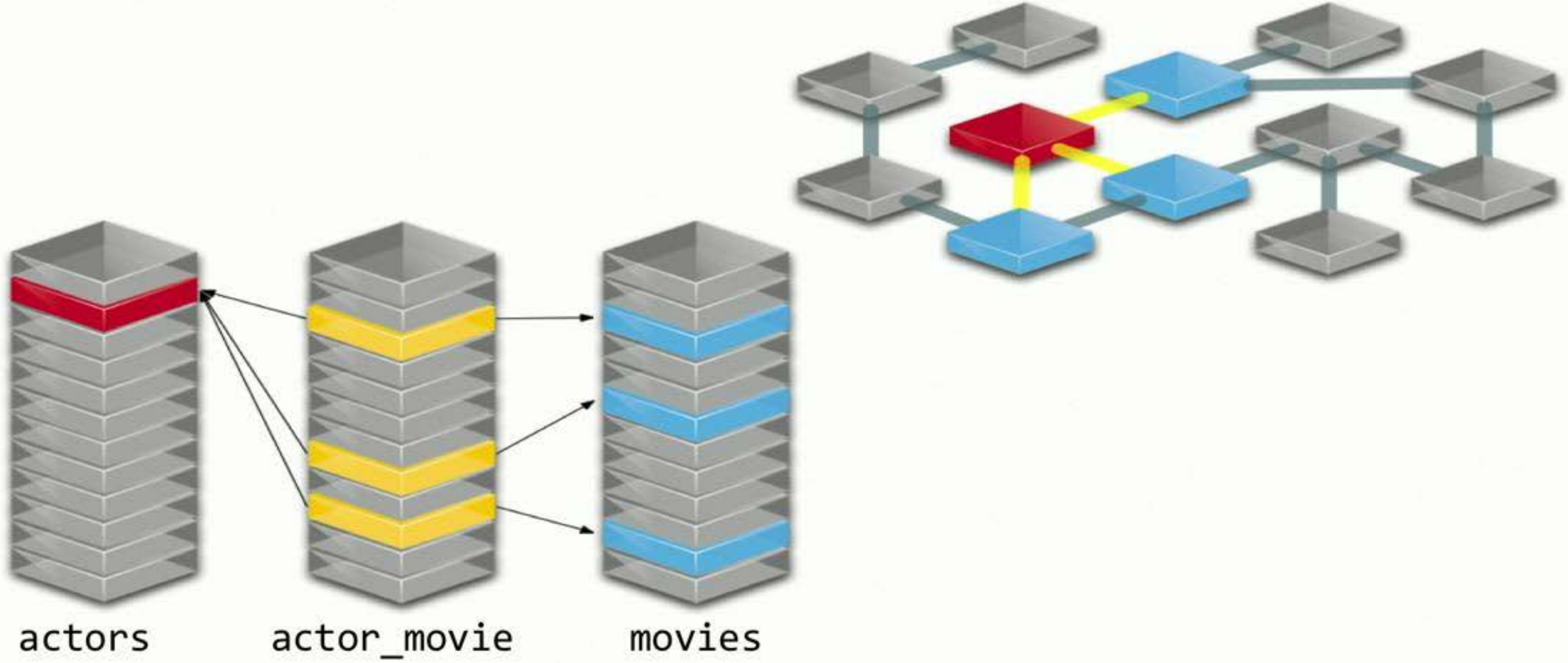
Relational to Graph

Relational is simple...until it gets complicated...



Relational to Graph

You know relational...now consider relationships...



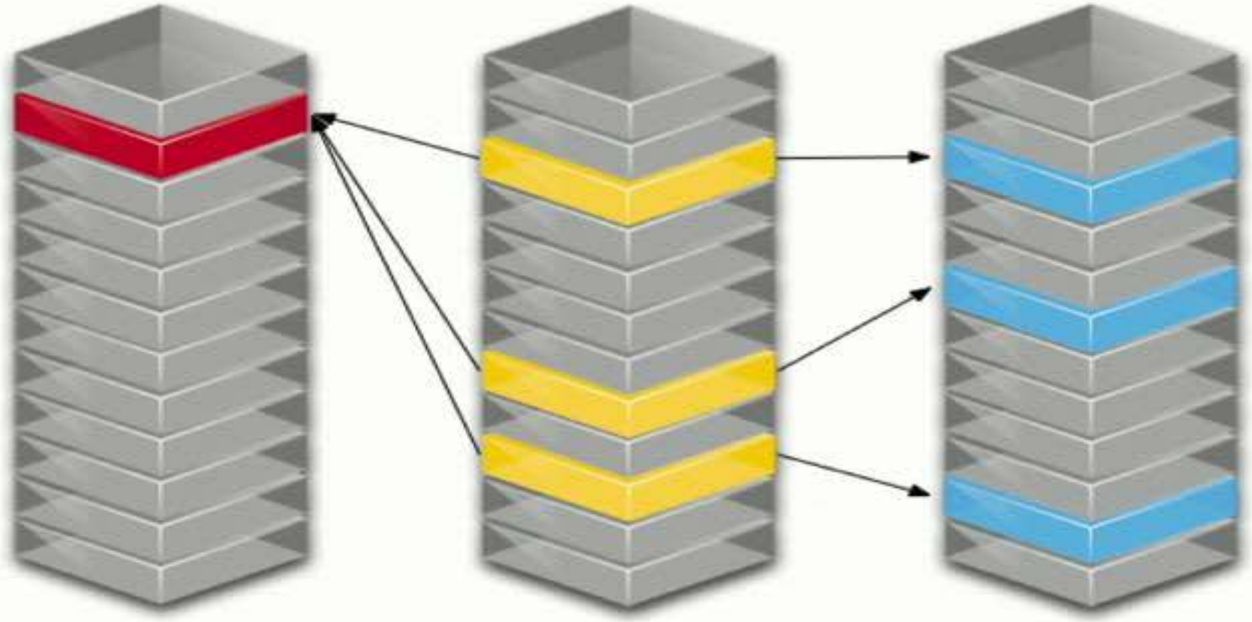
Normalized Relational Model to Graph Model

- Entity-Tables become Nodes
- Foreign Keys become Relationships
- Link Tables become Relationships
- Remove artificial Primary Keys and Foreign Keys

We already know how to query a relational database

Just use SQL

```
select movie.title
from actors join actor_movie on (...) join movies on (...)
where actors.name = "Andreas"
```

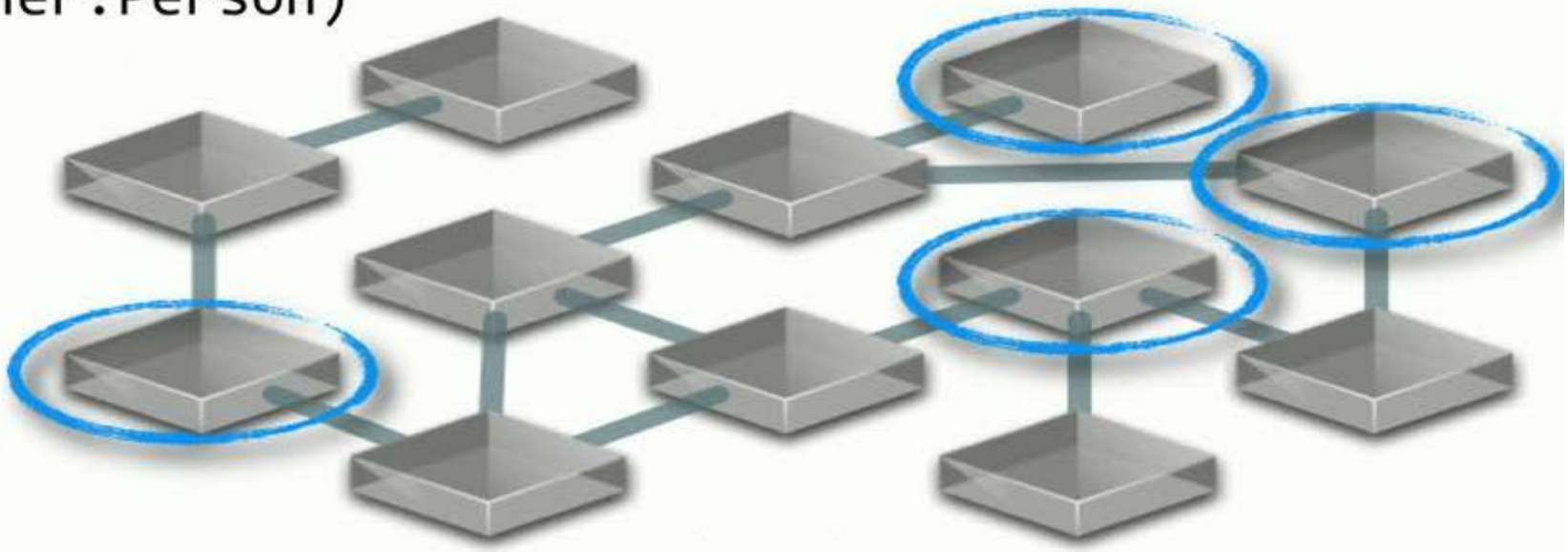


neo4j

How do we query a graph?

We traverse the graph

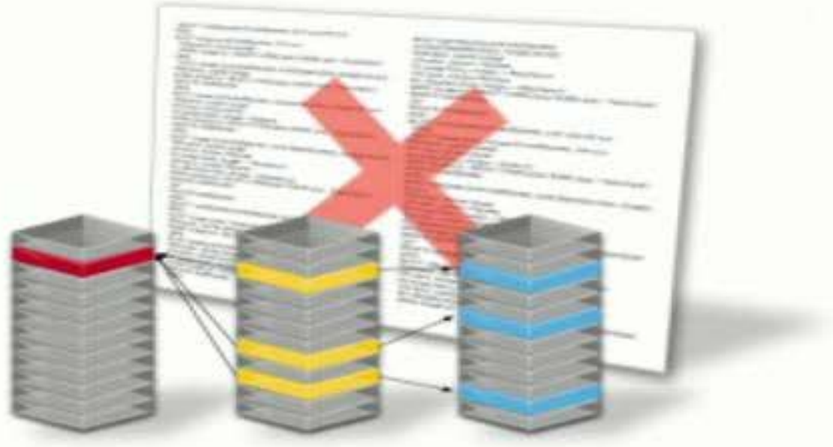
```
// find starting points
MATCH (:Person {name:'Andreas'})
// then traverse the relationships
      -[:ACTED_IN]->(movie:Movie)
     <-[:ACTED_IN]- (other:Person)
// and return results
RETURN other
```



RDBMS can't handle relationships well

neo4j

- Cannot model or store data and relationships without complexity
- Performance degrades with number and levels of relationships, and database size
- Query complexity grows with need for JOINs
- Adding new types of data and relationships requires schema redesign, increasing time to market



RDBMS can’t handle relationships well

neo4j

When **data relationships are valuable in real-time** traditional databases aren’t the best choice.

Express Complex Queries Easily with Cypher

Find all reports and how many people they manage, up to 3 levels down.

Cypher

```
MATCH (boss)-[:MANAGES*0..3]->(mgr)

WHERE boss.name = "John Doe"
AND (mgr)-[:MANAGES]->()

RETURN mgr.name AS Manager,
       size((mgr)-[:MANAGES*1..3]->())
       AS Total
```

SQL

```
(SELECT T.directReportees AS directReportees, sum(T.count) AS count
FROM (
SELECT manager.pid AS directReportees, 0 AS count
  FROM person_reportee manager
  WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
UNION
  SELECT manager.pid AS directReportees, count(manager.directly_manages) AS count
FROM person_reportee manager
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
UNION
SELECT manager.pid AS directReportees, count(reportee.directly_manages) AS count
FROM person_reportee manager
JOIN person_reportee reportee
ON manager.directly_manages = reportee.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
UNION
SELECT manager.pid AS directReportees, count(L2Reportees.directly_manages) AS count
FROM person_reportee manager
JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
) AS T
GROUP BY directReportees)
UNION
(SELECT T.directReportees AS directReportees, sum(T.count) AS count
FROM (
SELECT manager.directly_manages AS directReportees, 0 AS count
FROM person_reportee manager
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
UNION
SELECT reportee.pid AS directReportees, count(reportee.directly_manages) AS count
FROM person_reportee manager
JOIN person_reportee reportee
ON manager.directly_manages = reportee.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
UNION

SELECT depth1Reportees.pid AS directReportees,
count(depth2Reportees.directly_manages) AS count
FROM person_reportee manager
JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
) AS T
GROUP BY directReportees)
UNION
(SELECT T.directReportees AS directReportees, sum(T.count) AS count
  FROM(
  SELECT reportee.directly_manages AS directReportees, 0 AS count
FROM person_reportee manager
JOIN person_reportee reportee
ON manager.directly_manages = reportee.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
UNION
SELECT L2Reportees.pid AS directReportees, count(L2Reportees.directly_manages)
AS count
FROM person_reportee manager
JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
) AS T
GROUP BY directReportees)
UNION
(SELECT L2Reportees.directly_manages AS directReportees, 0 AS count
FROM person_reportee manager
JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
)
```

Unlocking Value from Your Data Relationships

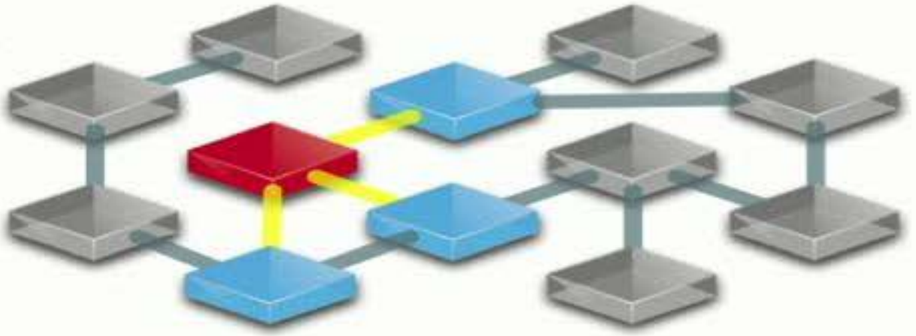
- Model your data **naturally** as a graph of data and relationships
- Drive graph model **from domain and use-cases**
- Use relationship information in **real-time** to transform your business
- Add new relationships **on the fly** to adapt to your changing requirements

High Query Performance with a Native Graph DB

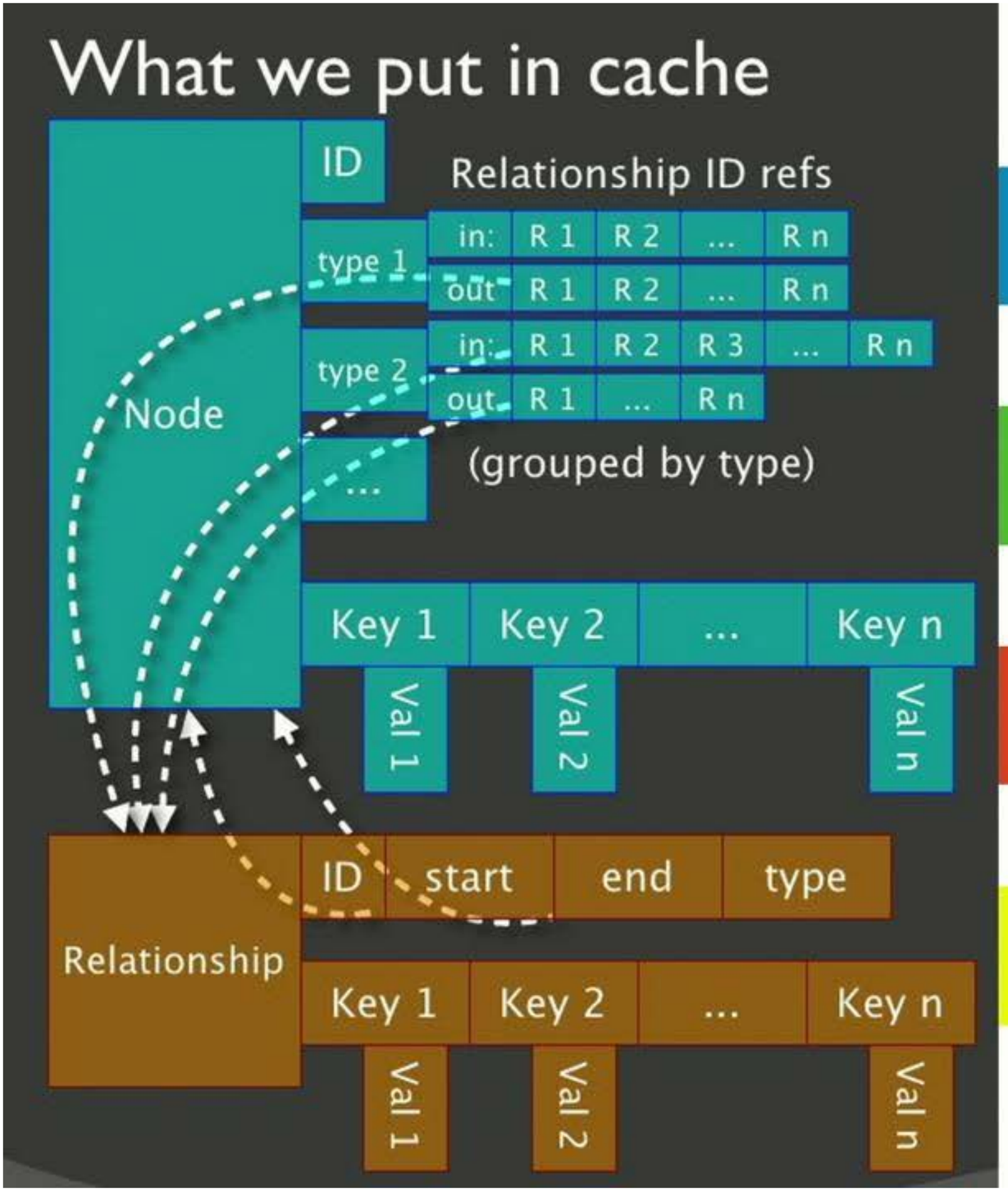
- Relationships are **first class citizen**
- No need for joins, just follow pre-materialized relationships of nodes
- Query & Data-**locality** – navigate out from your starting points
- Only load what's needed
- Aggregate and project results as you go
- Optimized disk and memory model for graphs

Looks different. Who cares?

- a sample social graph with ~1,000 persons
- average 50 friends per person
- pathExists(a,b) limited to depth 4
- caches warmed up to eliminate disk I/O



	# persons	query time
Relational database	1,000	2000ms
Neo4j	1,000	2ms
Neo4j	1,000,000	2ms



Neo4j Secret Sauce

1 Pointers instead of Lookups

2 Fixed Sized Records

3 “Joins” on Creation

4 Spin Spin Spin through this data structure

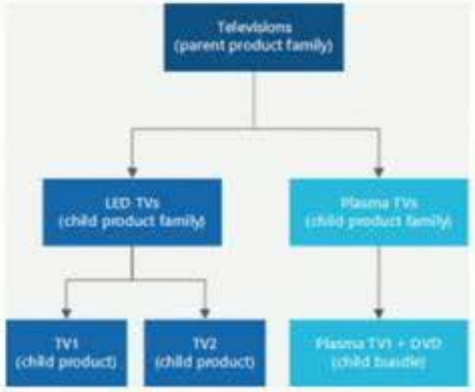
When to Choose Neo4j over Relational
Sub-second Response Time on n-way Joins + Agility

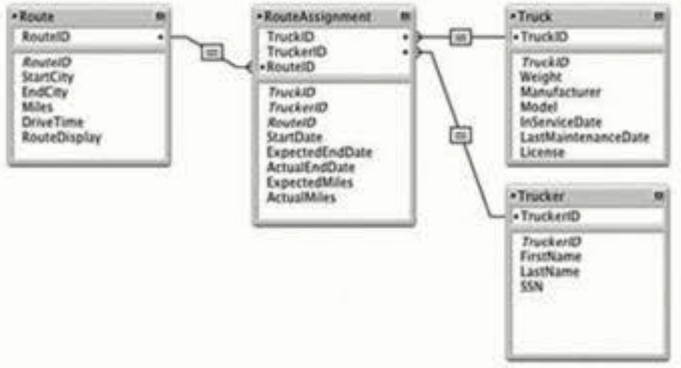
Neo4j Secret Sauce is "Index Free Adjacency"

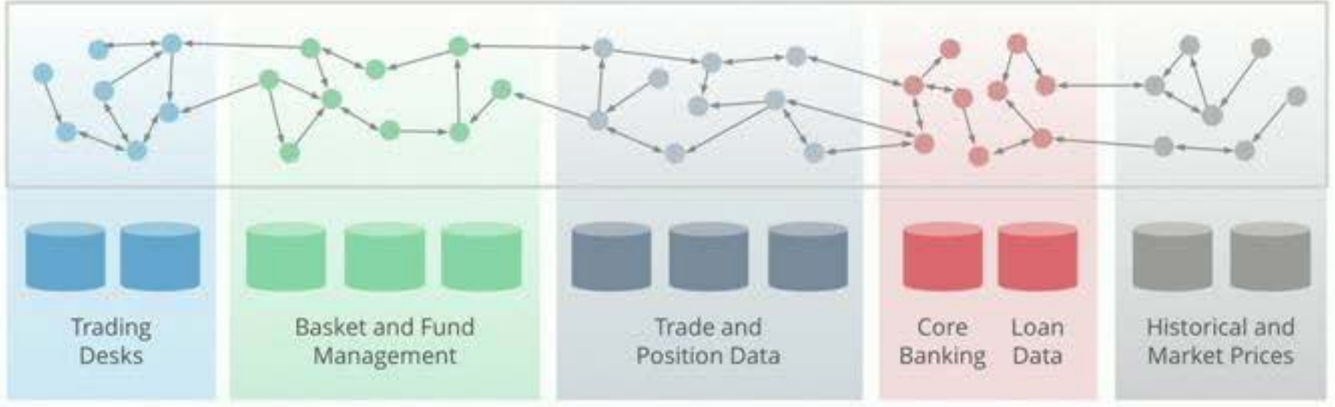
Relational Cannot Respond Sub-Second for 2+Way Joins

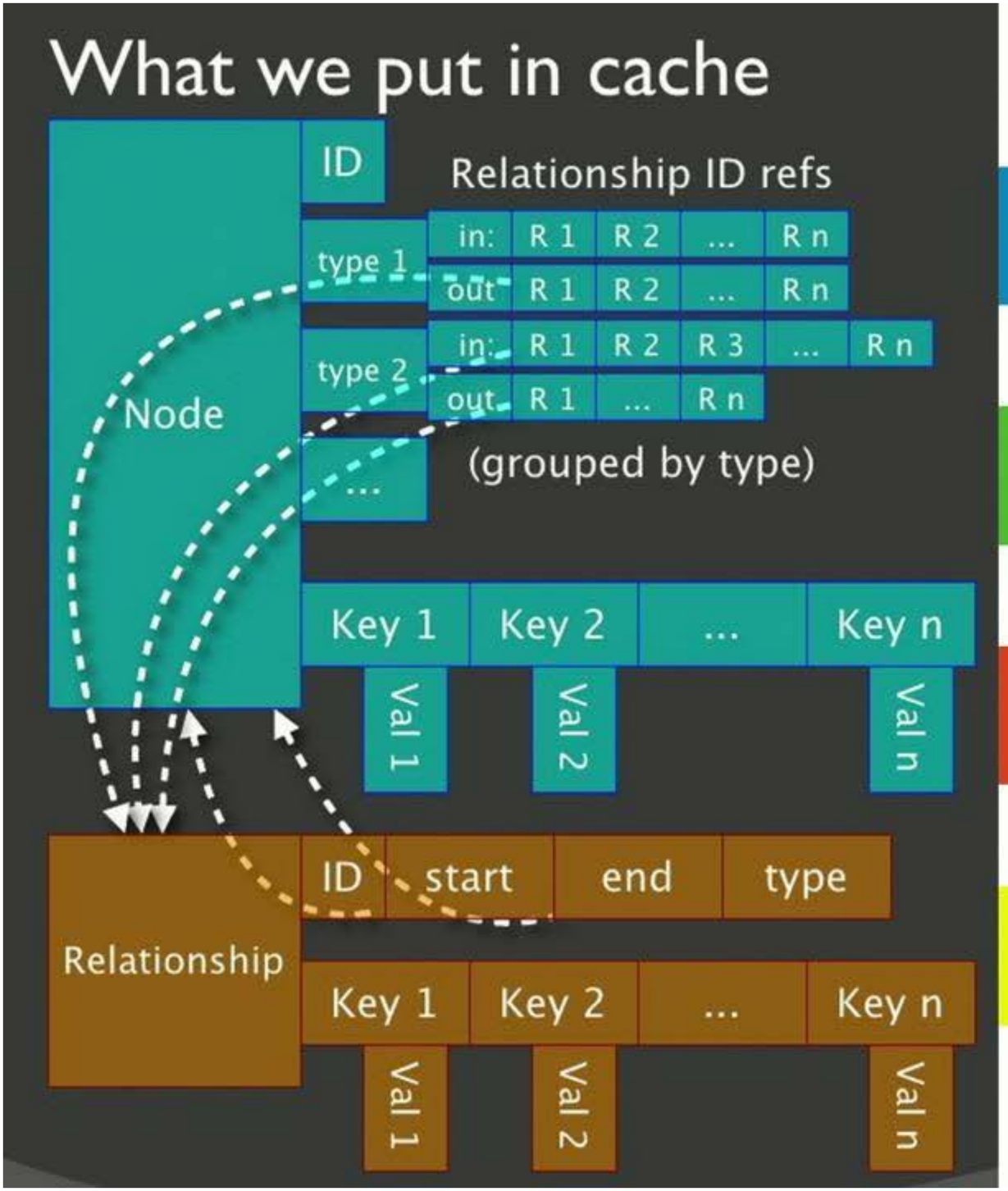
Relational Is Not Agile

- **Changes to queries**
- **New data feeds**









Neo4j Secret Sauce

1 Pointers instead of Lookups

2 Fixed Sized Records

3 “Joins” on Creation

4 Spin Spin Spin through this data structure

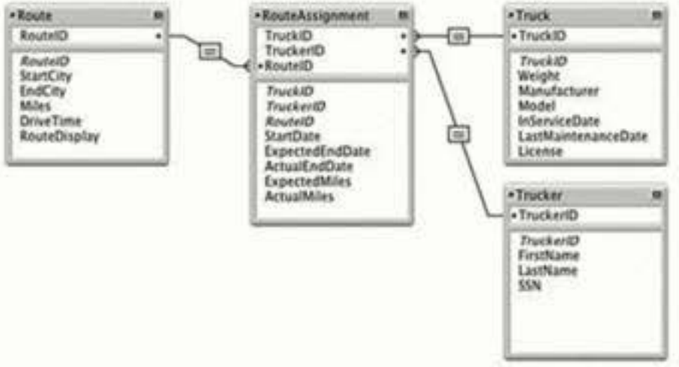
When to Choose Neo4j over Relational
Sub-second Response Time on n-way Joins + Agility

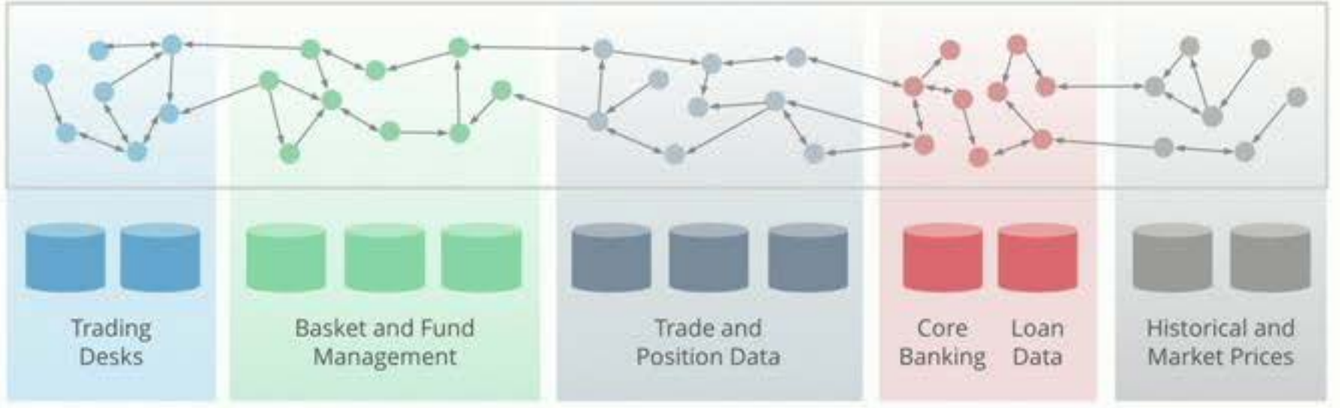
Neo4j Secret Sauce is "Index Free Adjacency"

Relational Cannot Respond Sub-Second for 2+Way Joins

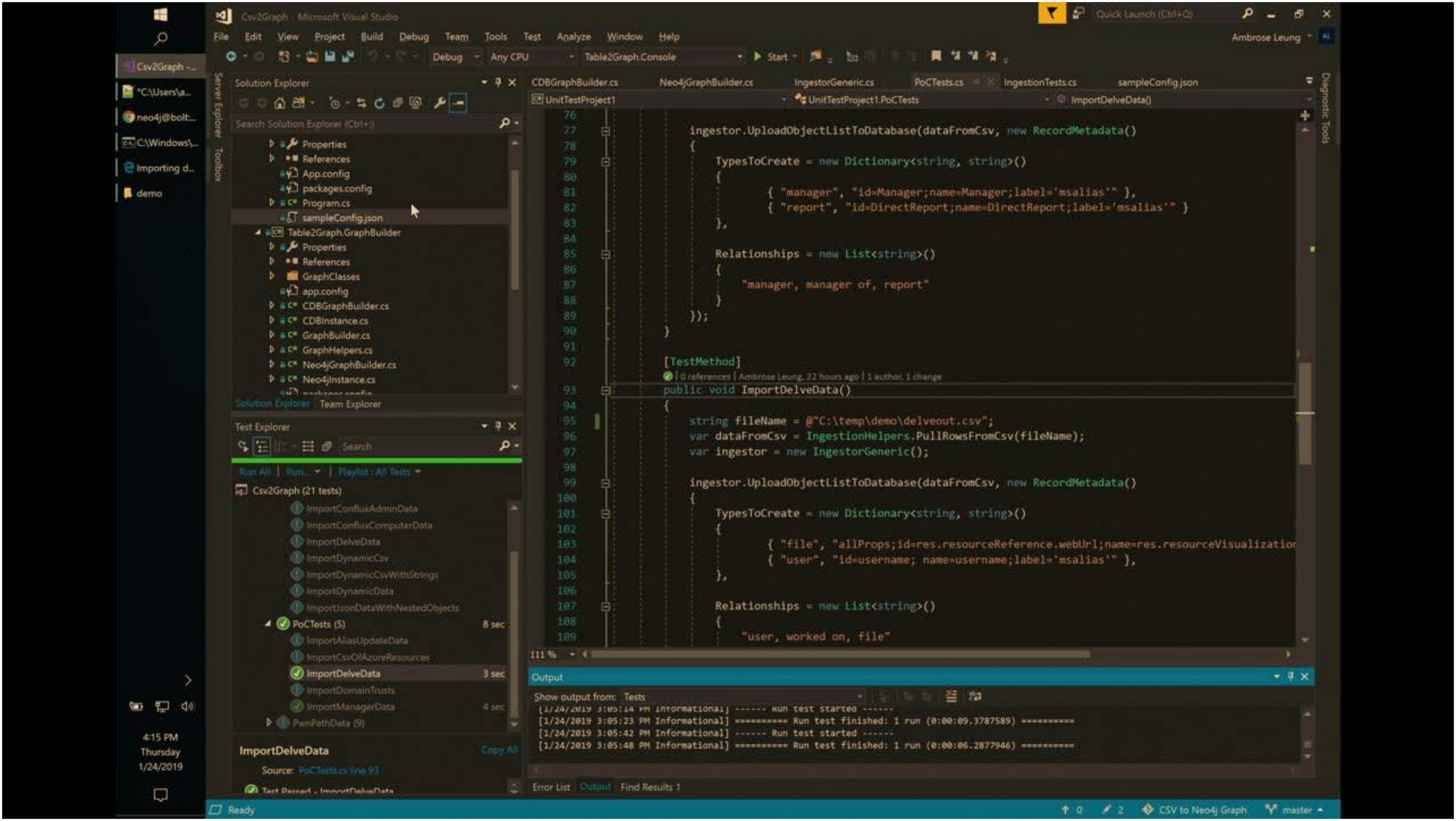
Relational Is Not Agile

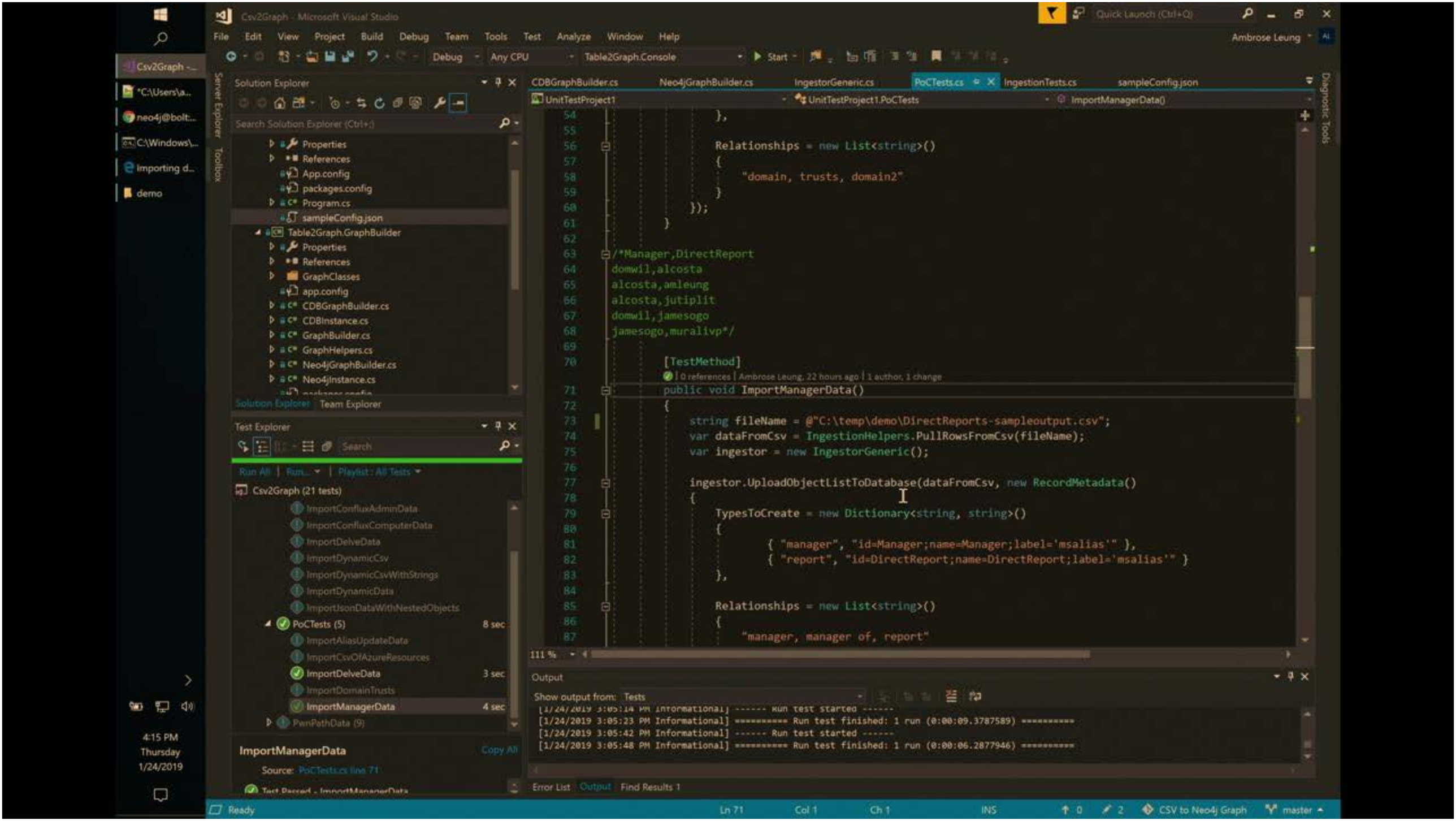
- **Changes to queries**
- **New data feeds**

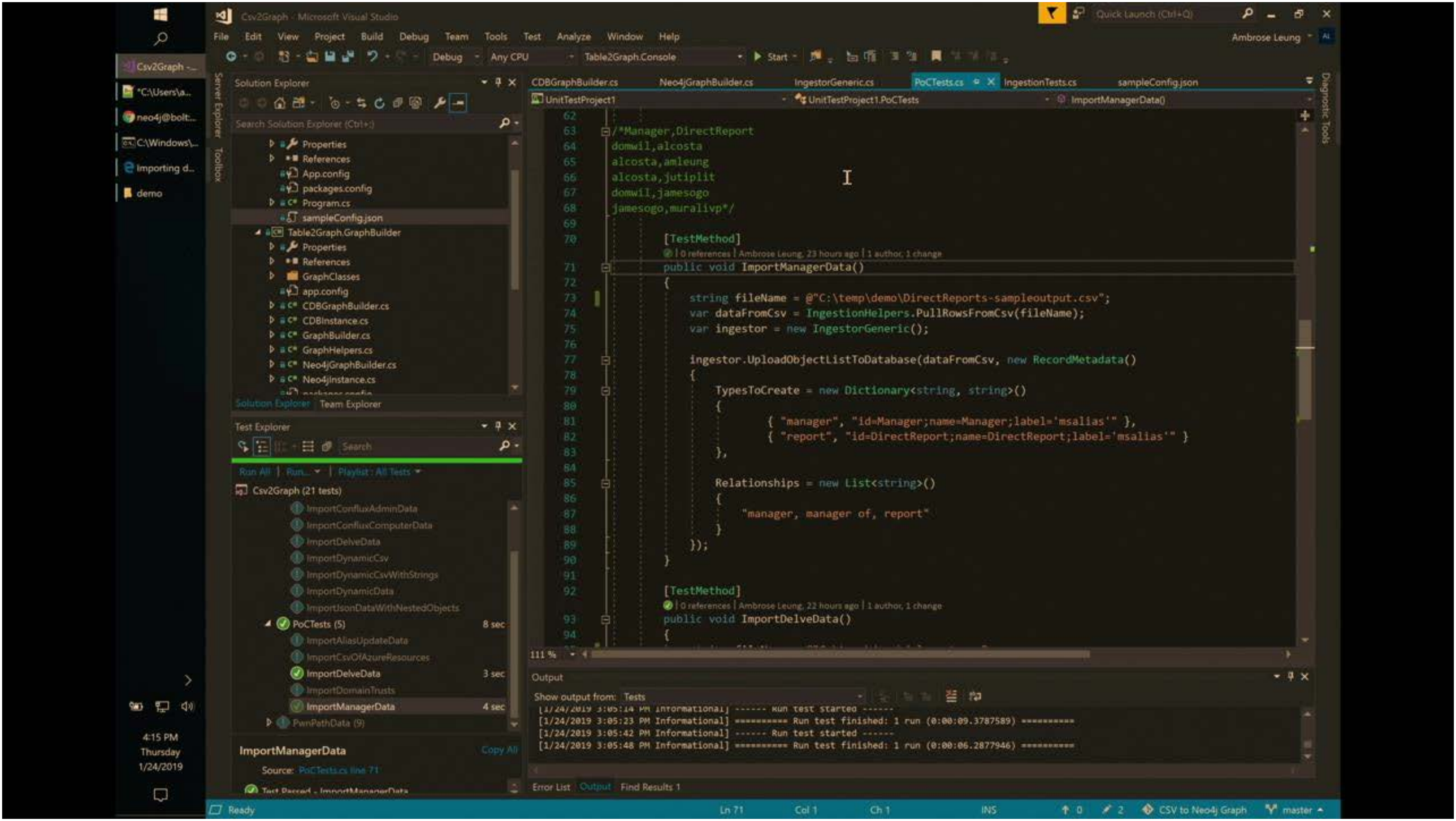


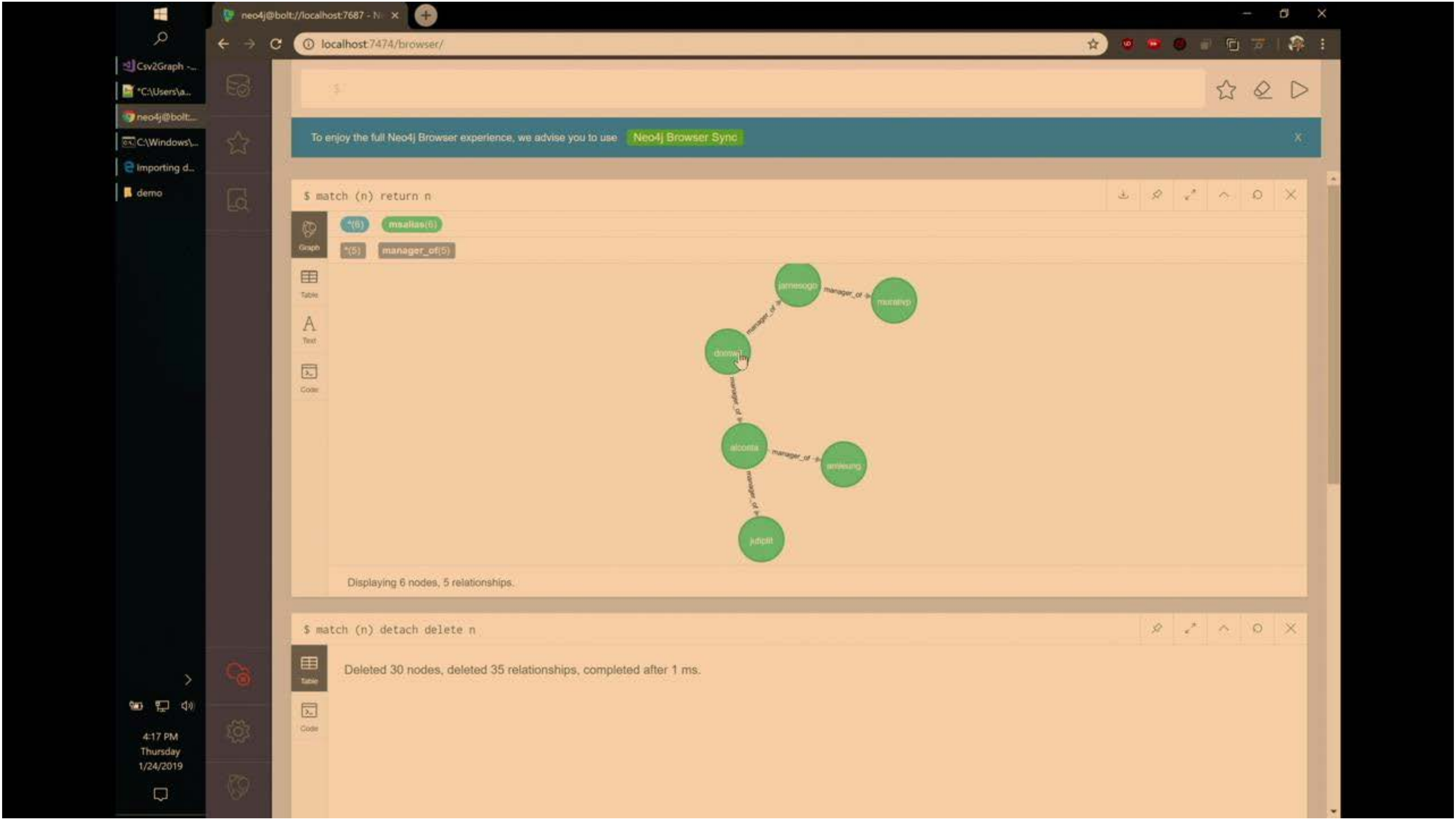


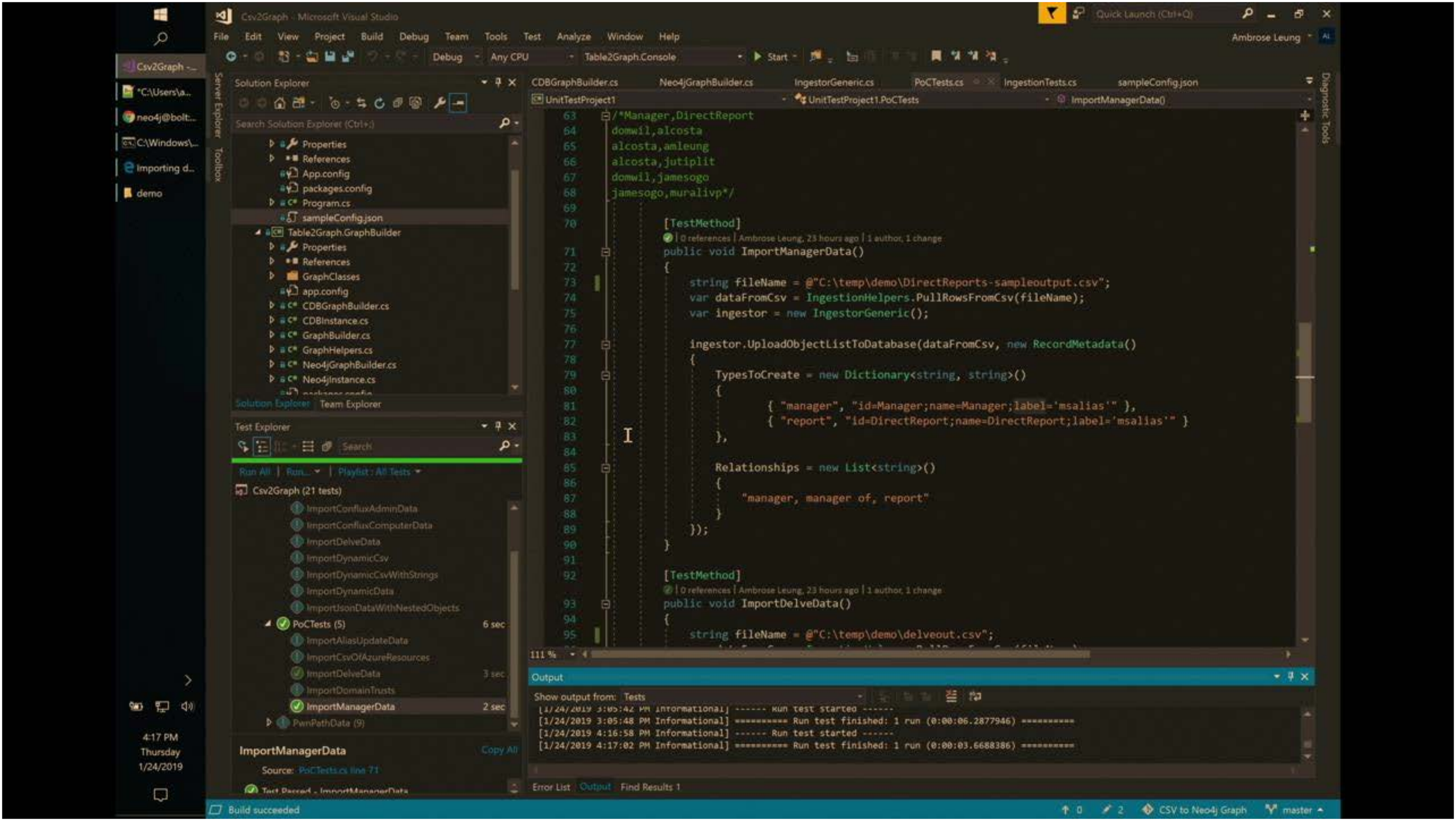
More ways to learn about Neo4j

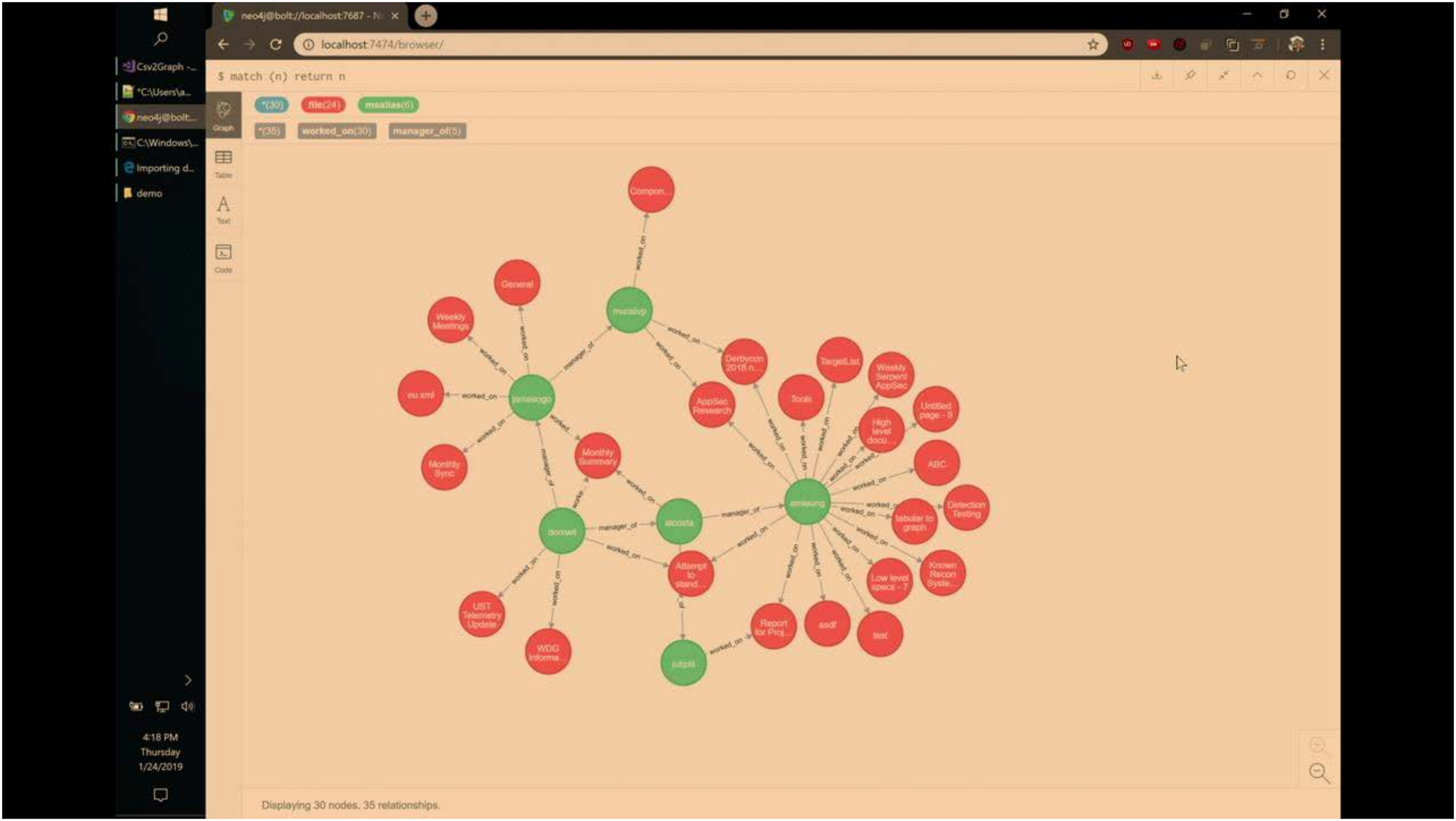

Csv2Graph - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Ambrose Leung
Debug Any CPU Table2Graph.Console Start
Solution Explorer
Search Solution Explorer (Ctrl+;)
Properties
References
App.config
packages.config
Program.cs
sampleConfig.json
Table2Graph.GraphBuilder
Properties
References
GraphClasses
app.config
CDBGraphBuilder.cs
CDBInstance.cs
GraphBuilder.cs
GraphHelpers.cs
Neo4jGraphBuilder.cs
Neo4jInstance.cs
Solution Explorer Team Explorer
Test Explorer
Search
Run All | Run... | Playlist: All Tests
Csv2Graph (21 tests)
ImportConfluxAdminData
ImportConfluxComputerData
ImportDelveData
ImportDynamicCsv
ImportDynamicCsvWithStrings
ImportDynamicData
ImportJsonDataWithNestedObjects
PoCTests (5) 8 sec
ImportAliasUpdateData
ImportCsvOfAzureResources
ImportDelveData 3 sec
ImportDomainTrusts
ImportManagerData 4 sec
PwnPathData (9)
ImportDelveData
Copy All
Source: PoCTests.cs line 93
CDBGraphBuilder.cs
Neo4jGraphBuilder.cs
IngestorGeneric.cs
PoCTests.cs
IngestionTests.cs
sampleConfig.json
UnitTestProject1
UnitTestProject1.PoCTests
ImportDelveData()
ingestor.UploadObjectListToDatabase(dataFromCsv, new RecordMetadata()
{
TypesToCreate = new Dictionary<string, string>()
{
{ "manager", "id=Manager;name=Manager;label='msalias'" },
{ "report", "id=DirectReport;name=DirectReport;label='msalias'" }
},
Relationships = new List<string>()
{
"manager, manager of, report"
}
});
}
[TestMethod]
0 references | Ambrose Leung, 22 hours ago | 1 author, 1 change
public void ImportDelveData()
{
string fileName = @"C:\temp\demo\delveout.csv";
var dataFromCsv = IngestionHelpers.PullRowsFromCsv(fileName);
var ingestor = new IngestorGeneric();
ingestor.UploadObjectListToDatabase(dataFromCsv, new RecordMetadata()
{
TypesToCreate = new Dictionary<string, string>()
{
{ "file", "allProps;id=res.resourceReference.webUrl;name=res.resourceVisualizatior
{ "user", "id=username; name=username;label='msalias'" },
},
Relationships = new List<string>()
{
"user, worked on, file"
111 %
Output
Show output from: Tests
[1/24/2019 3:05:14 PM Informational] ------ Run test started ------
[1/24/2019 3:05:23 PM Informational] ========== Run test finished: 1 run (0:00:09.3787589) ==========
[1/24/2019 3:05:42 PM Informational] ------ Run test started ------
[1/24/2019 3:05:48 PM Informational] ========== Run test finished: 1 run (0:00:06.2877946) ==========
Error List Output Find Results 1
Ready
0 2 CSV to Neo4j Graph master
4:15 PM
Thursday
1/24/2019


Csv2Graph - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Ambrose Leung
Debug Any CPU Table2Graph.Console Start
Solution Explorer
Search Solution Explorer (Ctrl+;)
Properties
References
App.config
packages.config
Program.cs
sampleConfig.json
Table2Graph.GraphBuilder
Properties
References
GraphClasses
app.config
CDBGraphBuilder.cs
CDBInstance.cs
GraphBuilder.cs
GraphHelpers.cs
Neo4jGraphBuilder.cs
Neo4jInstance.cs
Solution Explorer Team Explorer
Test Explorer
Search
Run All | Run... | Playlist: All Tests
Csv2Graph (21 tests)
ImportConfluxAdminData
ImportConfluxComputerData
ImportDelveData
ImportDynamicCsv
ImportDynamicCsvWithStrings
ImportDynamicData
ImportJsonDataWithNestedObjects
PoCTests (5) 8 sec
ImportAliasUpdateData
ImportCsvOfAzureResources
ImportDelveData 3 sec
ImportDomainTrusts
ImportManagerData 4 sec
PwnPathData (9)
ImportManagerData
Copy All
Source: PoCTests.cs line 71
CDBGraphBuilder.cs
Neo4jGraphBuilder.cs
IngestorGeneric.cs
PoCTests.cs
IngestionTests.cs
sampleConfig.json
UnitTestProject1
UnitTestProject1.PoCTests
ImportManagerData()
},
Relationships = new List<string>()
{
"domain, trusts, domain2"
}
});
}
/*Manager,DirectReport
domwil,alcosta
alcosta,amleung
alcosta,jutiplit
domwil,jamesogo
jamesogo,muralivp*/
[TestMethod]
0 references | Ambrose Leung, 22 hours ago | 1 author, 1 change
public void ImportManagerData()
{
string fileName = @"C:\temp\demo\DirectReports-sampleoutput.csv";
var dataFromCsv = IngestionHelpers.PullRowsFromCsv(fileName);
var ingestor = new IngestorGeneric();
ingestor.UploadObjectListToDatabase(dataFromCsv, new RecordMetadata()
{
TypesToCreate = new Dictionary<string, string>()
{
{ "manager", "id=Manager;name=Manager;label='msalias'" },
{ "report", "id=DirectReport;name=DirectReport;label='msalias'" }
},
Relationships = new List<string>()
{
"manager, manager of, report"
111 %
Output
Show output from: Tests
[1/24/2019 3:05:14 PM Informational] ------ Run test started ------
[1/24/2019 3:05:23 PM Informational] ========== Run test finished: 1 run (0:00:09.3787589) ==========
[1/24/2019 3:05:42 PM Informational] ------ Run test started ------
[1/24/2019 3:05:48 PM Informational] ========== Run test finished: 1 run (0:00:06.2877946) ==========
Error List Output Find Results 1
Ready Ln 71 Col 1 Ch 1 INS CSV to Neo4j Graph master
4:15 PM
Thursday
1/24/2019


Csv2Graph - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Ambrose Leung
Debug Any CPU Table2Graph.Console Start
Solution Explorer
Search Solution Explorer (Ctrl+;)
Properties
References
App.config
packages.config
Program.cs
sampleConfig.json
Table2Graph.GraphBuilder
Properties
References
GraphClasses
app.config
CDBGraphBuilder.cs
CDBInstance.cs
GraphBuilder.cs
GraphHelpers.cs
Neo4jGraphBuilder.cs
Neo4jInstance.cs
Solution Explorer Team Explorer
Test Explorer
Search
Run All | Run... | Playlist: All Tests
Csv2Graph (21 tests)
ImportConfluxAdminData
ImportConfluxComputerData
ImportDelveData
ImportDynamicCsv
ImportDynamicCsvWithStrings
ImportDynamicData
ImportJsonDataWithNestedObjects
PoCTests (5) 8 sec
ImportAliasUpdateData
ImportCsvOfAzureResources
ImportDelveData 3 sec
ImportDomainTrusts
ImportManagerData 4 sec
PwnPathData (9)
ImportManagerData
Copy All
Source: PoCTests.cs line 71
CDBGraphBuilder.cs
Neo4jGraphBuilder.cs
IngestorGeneric.cs
PoCTests.cs
IngestionTests.cs
sampleConfig.json
UnitTestProject1
UnitTestProject1.PoCTests
ImportManagerData()
/*Manager,DirectReport
domwil,alcosta
alcosta,amleung
alcosta,jutiplit
domwil,jamesogo
jamesogo,muralivp*/
[TestMethod]
0 references | Ambrose Leung, 23 hours ago | 1 author, 1 change
public void ImportManagerData()
{
string fileName = @"C:\temp\demo\DirectReports-sampleoutput.csv";
var dataFromCsv = IngestionHelpers.PullRowsFromCsv(fileName);
var ingestor = new IngestorGeneric();
ingestor.UploadObjectListToDatabase(dataFromCsv, new RecordMetadata()
{
TypesToCreate = new Dictionary<string, string>()
{
{ "manager", "id=Manager;name=Manager;label='msalias'" },
{ "report", "id=DirectReport;name=DirectReport;label='msalias'" }
},
Relationships = new List<string>()
{
"manager, manager of, report"
}
});
}
[TestMethod]
0 references | Ambrose Leung, 22 hours ago | 1 author, 1 change
public void ImportDelveData()
{
111 %
Output
Show output from: Tests
[1/24/2019 3:05:14 PM Informational] ------ Run test started ------
[1/24/2019 3:05:23 PM Informational] ========== Run test finished: 1 run (0:00:09.3787589) ==========
[1/24/2019 3:05:42 PM Informational] ------ Run test started ------
[1/24/2019 3:05:48 PM Informational] ========== Run test finished: 1 run (0:00:06.2877946) ==========
Error List Output Find Results 1
Ready Ln 71 Col 1 Ch 1 INS CSV to Neo4j Graph master
4:15 PM
Thursday
1/24/2019


neo4j@bolt://localhost:7687
localhost:7474/browser/
Csv2Graph
*C:\Users\a...
neo4j@bolt:...
C:\Windows\...
Importing d...
demo
To enjoy the full Neo4j Browser experience, we advise you to use
Neo4j Browser Sync
$ match (n) return n
*(6)
msalias(6)
*(5)
manager_of(5)
Graph
Table
Text
Code
manager_of
Displaying 6 nodes, 5 relationships.
$ match (n) detach delete n
Deleted 30 nodes, deleted 35 relationships, completed after 1 ms.
4:17 PM
Thursday
1/24/2019


Csv2Graph - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Ambrose Leung
Debug Any CPU Table2Graph.Console Start
Solution Explorer
Search Solution Explorer (Ctrl+;)
Properties
References
App.config
packages.config
Program.cs
sampleConfig.json
Table2Graph.GraphBuilder
Properties
References
GraphClasses
app.config
CDBGraphBuilder.cs
CDBInstance.cs
GraphBuilder.cs
GraphHelpers.cs
Neo4jGraphBuilder.cs
Neo4jInstance.cs
Solution Explorer Team Explorer
Test Explorer
Search
Run All Run... Playlist: All Tests
Csv2Graph (21 tests)
ImportConfluxAdminData
ImportConfluxComputerData
ImportDelveData
ImportDynamicCsv
ImportDynamicCsvWithStrings
ImportDynamicData
ImportJsonDataWithNestedObjects
PoCTests (5) 6 sec
ImportAliasUpdateData
ImportCsvOfAzureResources
ImportDelveData 3 sec
ImportDomainTrusts
ImportManagerData 2 sec
PwnPathData (9)
ImportManagerData
Copy All
Source: PoCTests.cs line 71
CDBGraphBuilder.cs
Neo4jGraphBuilder.cs
IngestorGeneric.cs
PoCTests.cs
IngestionTests.cs
sampleConfig.json
UnitTestProject1
UnitTestProject1.PoCTests
ImportManagerData()
/*Manager,DirectReport
domwil,alcosta
alcosta,ameleung
alcosta,jutiplit
domwil,jamesogo
jamesogo,muralivp*/
[TestMethod]
0 references | Ambrose Leung, 23 hours ago | 1 author, 1 change
public void ImportManagerData()
{
string fileName = @"C:\temp\demo\DirectReports-sampleoutput.csv";
var dataFromCsv = IngestionHelpers.PullRowsFromCsv(fileName);
var ingestor = new IngestorGeneric();
ingestor.UploadObjectListToDatabase(dataFromCsv, new RecordMetadata()
{
TypesToCreate = new Dictionary<string, string>()
{
{ "manager", "id=Manager;name=Manager;label='msalias'" },
{ "report", "id=DirectReport;name=DirectReport;label='msalias'" }
},
Relationships = new List<string>()
{
"manager, manager of, report"
}
});
}
[TestMethod]
0 references | Ambrose Leung, 23 hours ago | 1 author, 1 change
public void ImportDelveData()
{
string fileName = @"C:\temp\demo\delveout.csv";
111 %
Output
Show output from: Tests
[1/24/2019 3:05:42 PM Informational] ------ Run test started ------
[1/24/2019 3:05:48 PM Informational] ========== Run test finished: 1 run (0:00:06.2877946) ==========
[1/24/2019 4:16:58 PM Informational] ------ Run test started ------
[1/24/2019 4:17:02 PM Informational] ========== Run test finished: 1 run (0:00:03.6688386) ==========
Error List Output Find Results 1
Build succeeded
CSV to Neo4j Graph
master
Csv2Graph -...
*C:\Users\a...
neo4j@bolt:...
C:\Windows\...
Importing d...
demo
4:17 PM
Thursday
1/24/2019


neo4j@bolt://localhost:7687 - N
localhost:7474/browser/
$ match (n) return n
Graph
Table
Text
Code
*(30)
file(24)
msalias(0)
*(35)
worked_on(30)
manager_of(5)
Compon...
General
Weekly Meetings
muralivp
Derbycon 2018 n...
TargetList
Weekly Serpent AppSec
Untitled page - 9
High level docu...
ABC
Detection Testing
tabular to graph
Known Recon Syste...
Low level specs - 7
test
asdf
Report for Proj...
AppSec Research
Tools
esi.xml
jamesago
Monthly Sync
Monthly Summary
dominell
alcosta
Attempt to stand...
UST Telemetry Update
WDG Informa...
jutpal
smlsung
worked_on
manager_of
4:18 PM
Thursday
1/24/2019
Displaying 30 nodes, 35 relationships.

Importing data into a g

https://msblox-03.visualstudio.com/CSV%20to%20Neo4j%20Graph/_wiki/wikis/CSV-to-Neo4j-Graph.wiki?wikiVersion=GBwikiMaster&pagePath=%2FImporting%2

Azure DevOps | msblox-03 / CSV to Neo4j Graph / Wiki | Search

CSV to Neo4j Graph

- Overview
- Summary
- Dashboards
- Analytics views
- Wiki
- Project Online
- Boards
- Repos
- Pipelines
- Test Plans
- Artifacts
- Project settings

Wikis > CSV-to-Neo4j-Graph.wiki

Importing data into a graph - Graph Builder

Ambrose Leung 7 minutes ago Revisions | Edit page | New page | More

Filter pages b

Importing data into a graph - Graph Builder

Importing data...

Table2Graph.exe Co...

aka.ms/csv2graph

Why the need for yet another way to add data into a graph?

There is currently no straightforward way of adding arbitrarily formatted csv data to a graph.
The API shown here makes it intuitive to define objects (vertices) and relationships (edges) as metadata.

Example: You have a table of records with the format

ManagerName	ReportName	IdManager	IdReport
Alice	Bob	1	2
Alice	Cindy	1	3
Cindy	David	3	4

You first define the type of objects as a dictionary of types (in this case, the types are "manager" and "report"):

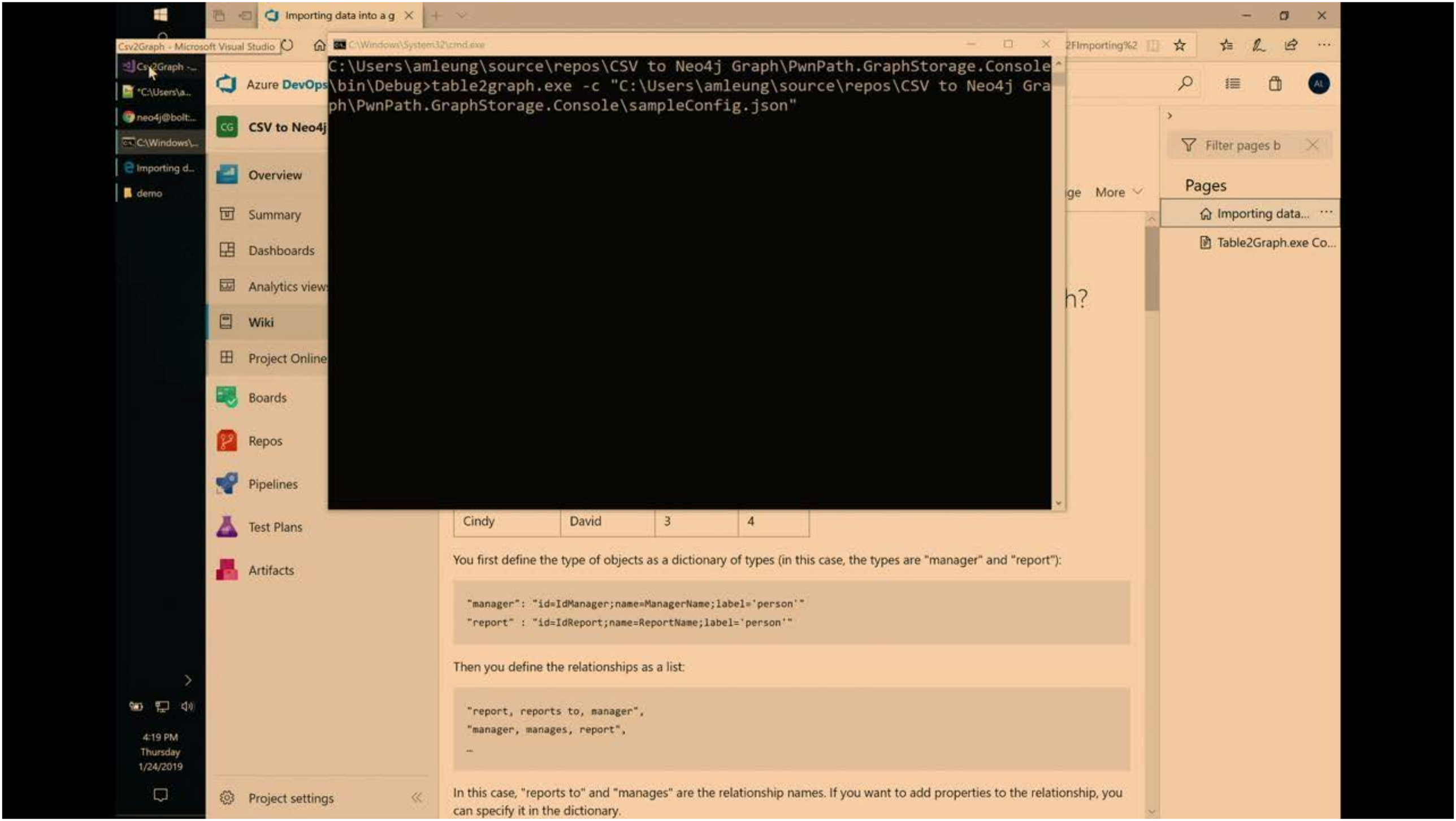
```
"manager": "id=IdManager;name=ManagerName;label='person'"
"report" : "id=IdReport;name=ReportName;label='person'"
```

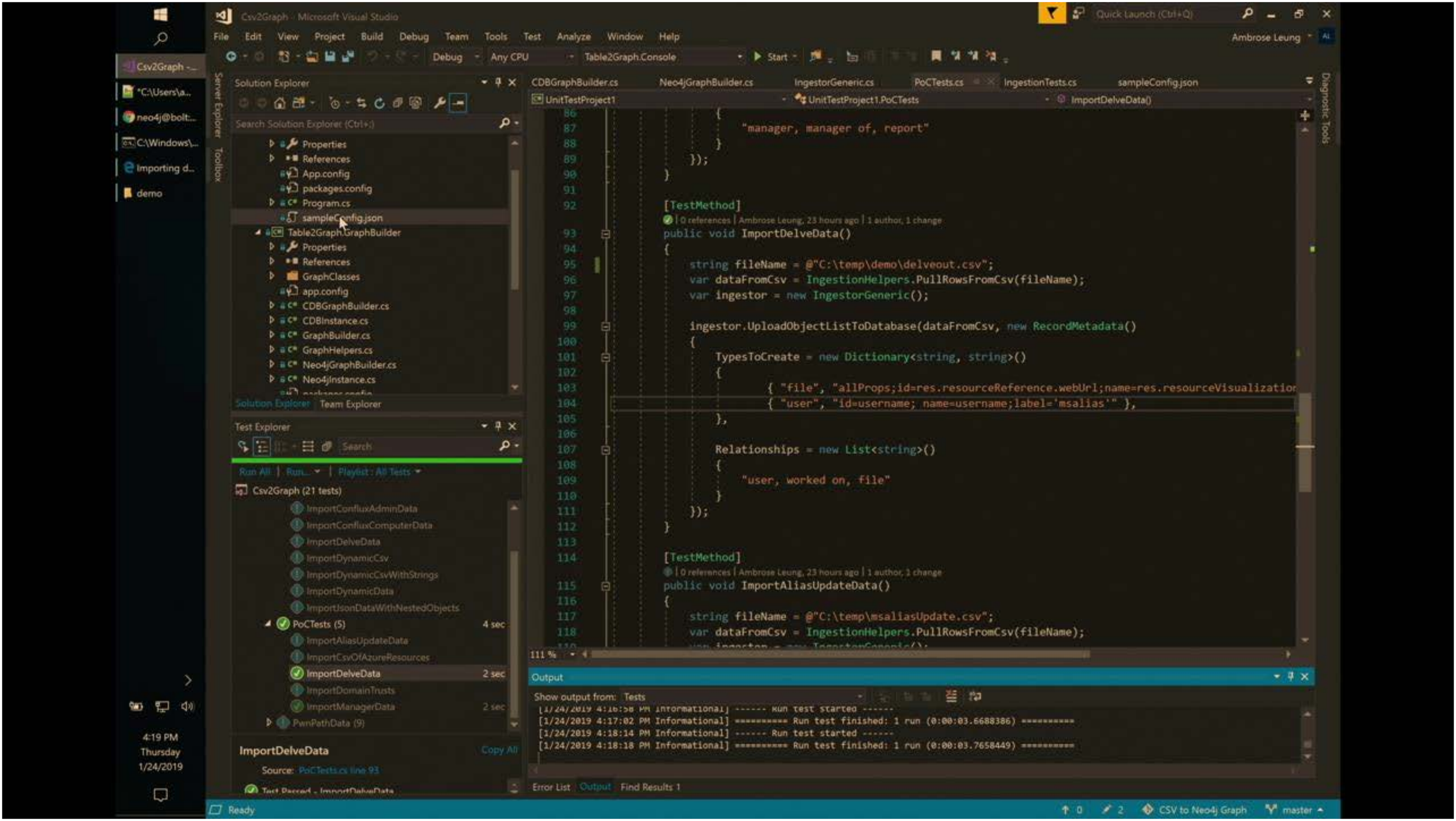
Then you define the relationships as a list:

```
"report, reports to, manager",
"manager, manages, report",
…
```

In this case, "reports to" and "manages" are the relationship names. If you want to add properties to the relationship, you can specify it in the dictionary.

4:18 PM
Thursday
1/24/2019


Importing data into a g
Csv2Graph - Microsoft Visual Studio
C:\Windows\System32\cmd.exe
C:\Users\amleung\source\repos\CSV to Neo4j Graph\PwnPath.GraphStorage.Console\bin\Debug>table2graph.exe -c "C:\Users\amleung\source\repos\CSV to Neo4j Graph\PwnPath.GraphStorage.Console\sampleConfig.json"
Azure DevOps
CSV to Neo4j
Overview
Summary
Dashboards
Analytics views
Wiki
Project Online
Boards
Repos
Pipelines
Test Plans
Artifacts
Project settings
Filter pages b
Pages
Importing data...
Table2Graph.exe Co...
More
Cindy
David
3
4
You first define the type of objects as a dictionary of types (in this case, the types are "manager" and "report"):
"manager": "id=IdManager;name=ManagerName;label='person'"
"report" : "id=IdReport;name=ReportName;label='person'"
Then you define the relationships as a list:
"report, reports to, manager",
"manager, manages, report",
…
In this case, "reports to" and "manages" are the relationship names. If you want to add properties to the relationship, you can specify it in the dictionary.
4:19 PM
Thursday
1/24/2019


Csv2Graph - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Ambrose Leung
Debug Any CPU Table2Graph.Console Start
Solution Explorer
Search Solution Explorer (Ctrl+;)
Properties
References
App.config
packages.config
Program.cs
sampleConfig.json
Table2Graph.GraphBuilder
Properties
References
GraphClasses
app.config
CDBGraphBuilder.cs
CDBInstance.cs
GraphBuilder.cs
GraphHelpers.cs
Neo4jGraphBuilder.cs
Neo4jInstance.cs
Solution Explorer Team Explorer
Test Explorer
Run All | Run... | Playlist: All Tests
Csv2Graph (21 tests)
ImportConfluxAdminData
ImportConfluxComputerData
ImportDelveData
ImportDynamicCsv
ImportDynamicCsvWithStrings
ImportDynamicData
ImportJsonDataWithNestedObjects
PoCTests (5) 4 sec
ImportAliasUpdateData
ImportCsvOfAzureResources
ImportDelveData 2 sec
ImportDomainTrusts
ImportManagerData 2 sec
PwnPathData (9)
ImportDelveData
Copy All
Source: PoCTests.cs line 93
CDBGraphBuilder.cs
Neo4jGraphBuilder.cs
IngestorGeneric.cs
PoCTests.cs
IngestionTests.cs
sampleConfig.json
UnitTestProject1
UnitTestProject1.PoCTests
ImportDelveData()
"manager, manager of, report"
[TestMethod]
0 references | Ambrose Leung, 23 hours ago | 1 author, 1 change
public void ImportDelveData()
{
string fileName = @"C:\temp\demo\delveout.csv";
var dataFromCsv = IngestionHelpers.PullRowsFromCsv(fileName);
var ingestor = new IngestorGeneric();
ingestor.UploadObjectListToDatabase(dataFromCsv, new RecordMetadata()
{
TypesToCreate = new Dictionary<string, string>()
{
{ "file", "allProps;id=res.resourceReference.webUrl;name=res.resourceVisualizatior
{ "user", "id=username; name=username;label='msalias'" },
},
Relationships = new List<string>()
{
"user, worked on, file"
}
});
}
[TestMethod]
0 references | Ambrose Leung, 23 hours ago | 1 author, 1 change
public void ImportAliasUpdateData()
{
string fileName = @"C:\temp\msaliasUpdate.csv";
var dataFromCsv = IngestionHelpers.PullRowsFromCsv(fileName);
111 %
Output
Show output from: Tests
[1/24/2019 4:17:02 PM Informational] ========== Run test finished: 1 run (0:00:03.6688386) ==========
[1/24/2019 4:18:14 PM Informational] ------ Run test started ------
[1/24/2019 4:18:18 PM Informational] ========== Run test finished: 1 run (0:00:03.7658449) ==========
Error List Output Find Results 1
Ready
CSV to Neo4j Graph
master
4:19 PM
Thursday
1/24/2019

Importing data into a g

https://msblox-03.visualstudio.com/CSV%20to%20Neo4j%20Graph/_wiki/wikis/CSV-to-Neo4j-Graph.wiki?wikiVersion=GBwikiMaster&pagePath=%2FImporting%2

Azure DevOps | msblox-03 / CSV to Neo4j Graph / Wiki | Search

CSV to Neo4j Graph

- Overview
- Summary
- Dashboards
- Analytics views
- Wiki
- Project Online
- Boards
- Repos
- Pipelines
- Test Plans
- Artifacts
- Project settings

Wikis > CSV-to-Neo4j-Graph.wiki

Importing data into a graph - Graph Builder

Ambrose Leung 7 minutes ago Revisions | Edit page | New page | More

Filter pages b

Importing data into a graph - Graph Builder

Importing data...

Table2Graph.exe Co...

aka.ms/csv2graph

Why the need for yet another way to add data into a graph?

There is currently no straightforward way of adding arbitrarily formatted csv data to a graph.
The API shown here makes it intuitive to define objects (vertices) and relationships (edges) as metadata.

Example: You have a table of records with the format

ManagerName	ReportName	IdManager	IdReport
Alice	Bob	1	2
Alice	Cindy	1	3
Cindy	David	3	4

You first define the type of objects as a dictionary of types (in this case, the types are "manager" and "report"):

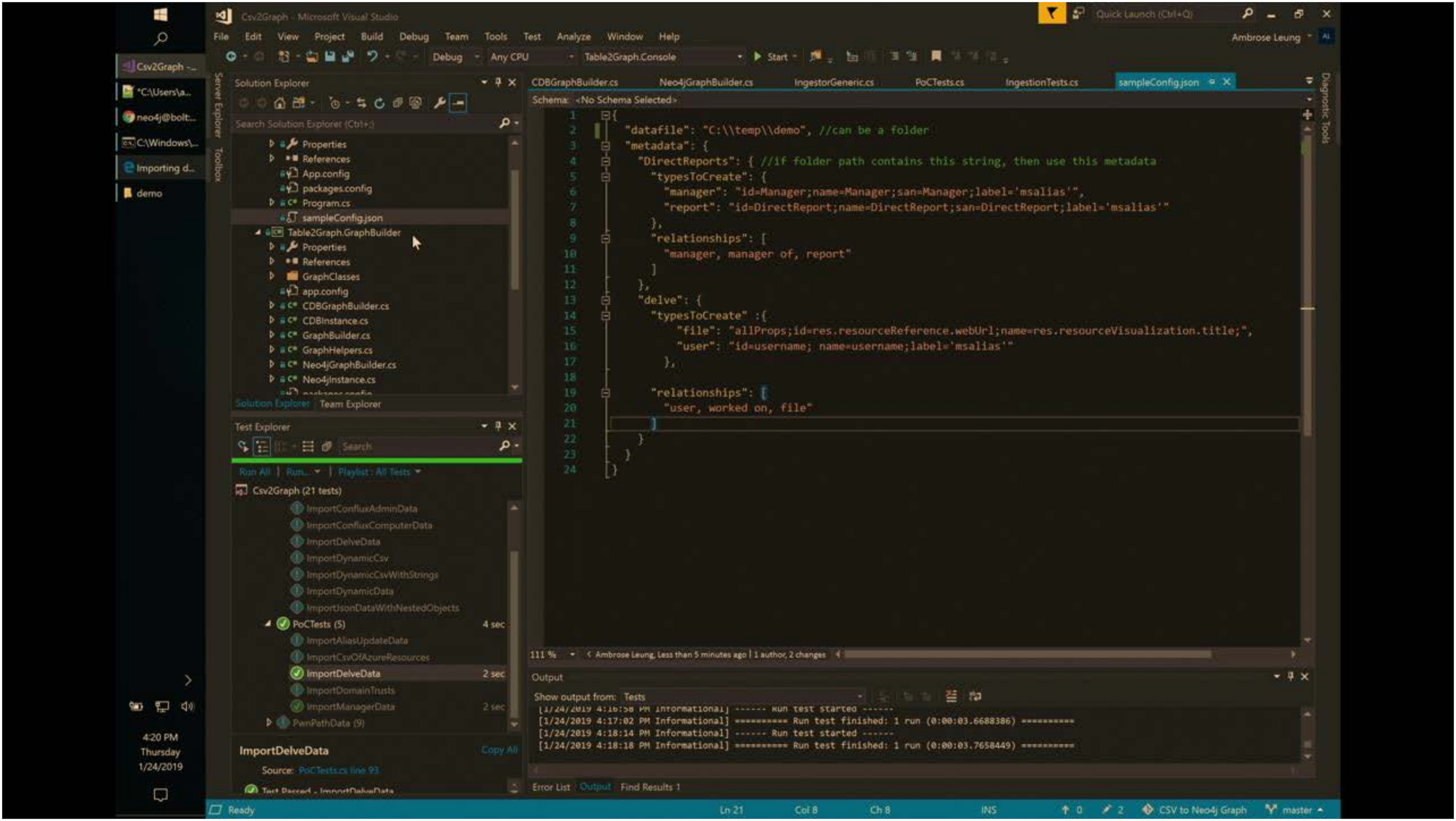
```
"manager": "id=IdManager;name=ManagerName;label='person'"
"report" : "id=IdReport;name=ReportName;label='person'"
```

Then you define the relationships as a list:

```
"report, reports to, manager",
"manager, manages, report",
…
```

In this case, "reports to" and "manages" are the relationship names. If you want to add properties to the relationship, you can specify it in the dictionary.

4:19 PM
Thursday
1/24/2019


Csv2Graph - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Ambrose Leung
Debug Any CPU Table2Graph.Console Start
Solution Explorer
Search Solution Explorer (Ctrl+;)
Properties
References
App.config
packages.config
Program.cs
sampleConfig.json
Table2Graph.GraphBuilder
Properties
References
GraphClasses
app.config
CDBGraphBuilder.cs
CDBInstance.cs
GraphBuilder.cs
GraphHelpers.cs
Neo4jGraphBuilder.cs
Neo4jInstance.cs
Solution Explorer Team Explorer
Test Explorer
Search
Run All | Run... | Playlist : All Tests
Csv2Graph (21 tests)
ImportConfluxAdminData
ImportConfluxComputerData
ImportDelveData
ImportDynamicCsv
ImportDynamicCsvWithStrings
ImportDynamicData
ImportJsonDataWithNestedObjects
PoCTests (5) 4 sec
ImportAliasUpdateData
ImportCsvOfAzureResources
ImportDelveData 2 sec
ImportDomainTrusts
ImportManagerData 2 sec
PwnPathData (9)
ImportDelveData
Copy All
Source: PoCTests.cs line 93
CDBGraphBuilder.cs
Neo4jGraphBuilder.cs
IngestorGeneric.cs
PoCTests.cs
IngestionTests.cs
sampleConfig.json
Schema: <No Schema Selected>
{
"datafile": "C:\\temp\\demo", //can be a folder
"metadata": {
"DirectReports": { //if folder path contains this string, then use this metadata
"typesToCreate": {
"manager": "id=Manager;name=Manager;san=Manager;label='msalias'",
"report": "id=DirectReport;name=DirectReport;san=DirectReport;label='msalias'"
},
"relationships": [
"manager, manager of, report"
]
},
"delve": {
"typesToCreate" :{
"file": "allProps;id=res.resourceReference.webUrl;name=res.resourceVisualization.title;",
"user": "id=username; name=username;label='msalias'"
},
"relationships": [
"user, worked on, file"
]
}
}
}
111 %
Ambrose Leung, Less than 5 minutes ago | 1 author, 2 changes
Output
Show output from: Tests
[1/24/2019 4:17:02 PM Informational] ========== Run test finished: 1 run (0:00:03.6688386) ==========
[1/24/2019 4:18:14 PM Informational] ------ Run test started ------
[1/24/2019 4:18:18 PM Informational] ========== Run test finished: 1 run (0:00:03.7658449) ==========
Error List Output Find Results 1
Ready Ln 21 Col 8 Ch 8 INS CSV to Neo4j Graph master
4:20 PM
Thursday
1/24/2019

Importing data into a g

https://msblox-03.visualstudio.com/CSV%20to%20Neo4j%20Graph/_wiki/wikis/CSV-to-Neo4j-Graph.wiki?wikiVersion=GBwikiMaster&pagePath=%2FImporting%2

Azure DevOps | msblox-03 / CSV to Neo4j Graph / Wiki | Search

CSV to Neo4j Graph

- Overview
- Summary
- Dashboards
- Analytics views
- Wiki
- Project Online
- Boards
- Repos
- Pipelines
- Test Plans
- Artifacts
- Project settings

Wikis > CSV-to-Neo4j-Graph.wiki

Importing data into a graph - Graph Builder

Ambrose Leung 7 minutes ago Revisions | Edit page | New page | More

Filter pages b

Importing data into a graph - Graph Builder

Importing data...

Table2Graph.exe Co...

aka.ms/csv2graph

Why the need for yet another way to add data into a graph?

There is currently no straightforward way of adding arbitrarily formatted csv data to a graph.
The API shown here makes it intuitive to define objects (vertices) and relationships (edges) as metadata.

Example: You have a table of records with the format

ManagerName	ReportName	IdManager	IdReport
Alice	Bob	1	2
Alice	Cindy	1	3
Cindy	David	3	4

You first define the type of objects as a dictionary of types (in this case, the types are "manager" and "report"):

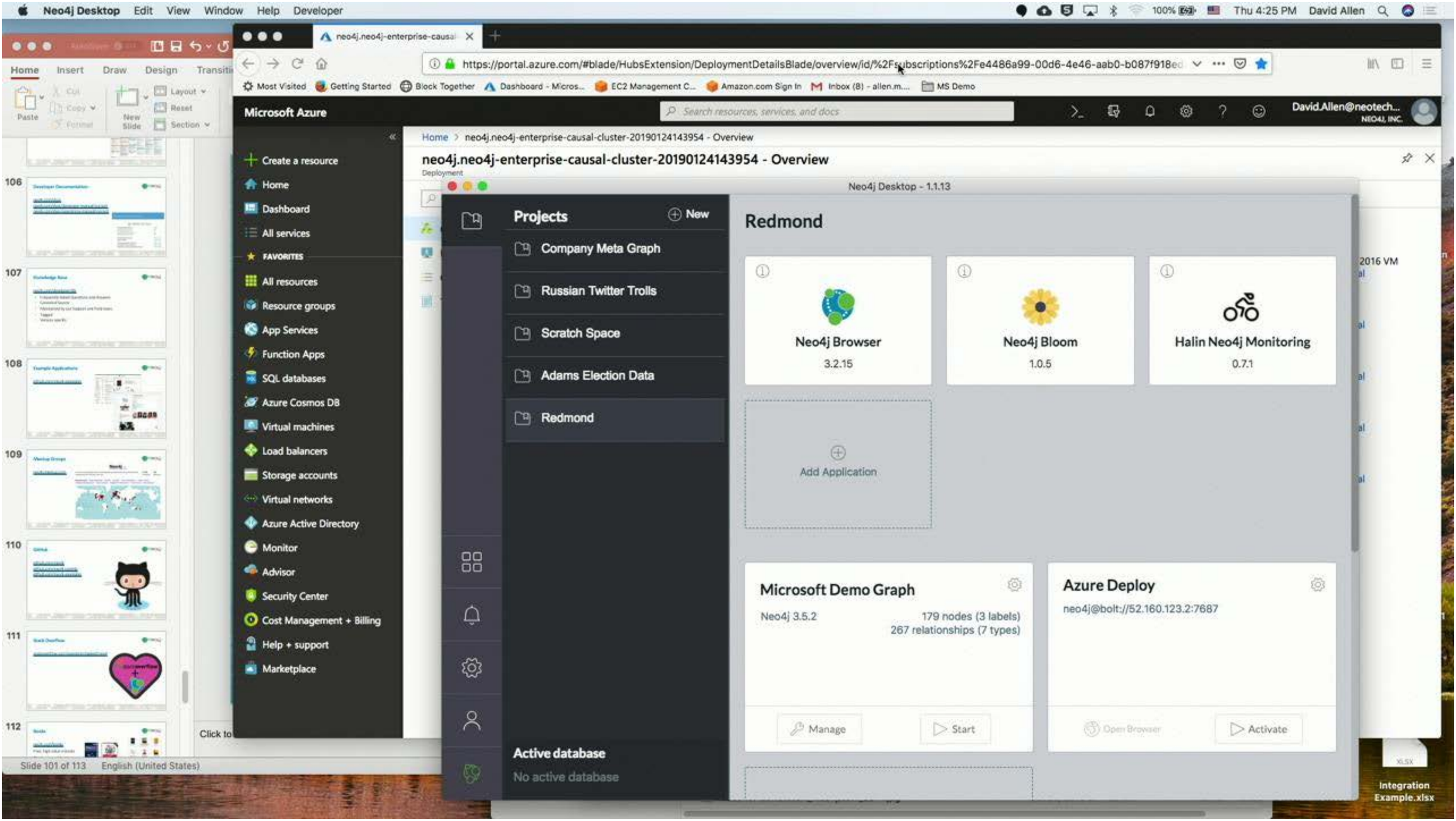
```
"manager": "id=IdManager;name=ManagerName;label='person'"
"report" : "id=IdReport;name=ReportName;label='person'"
```

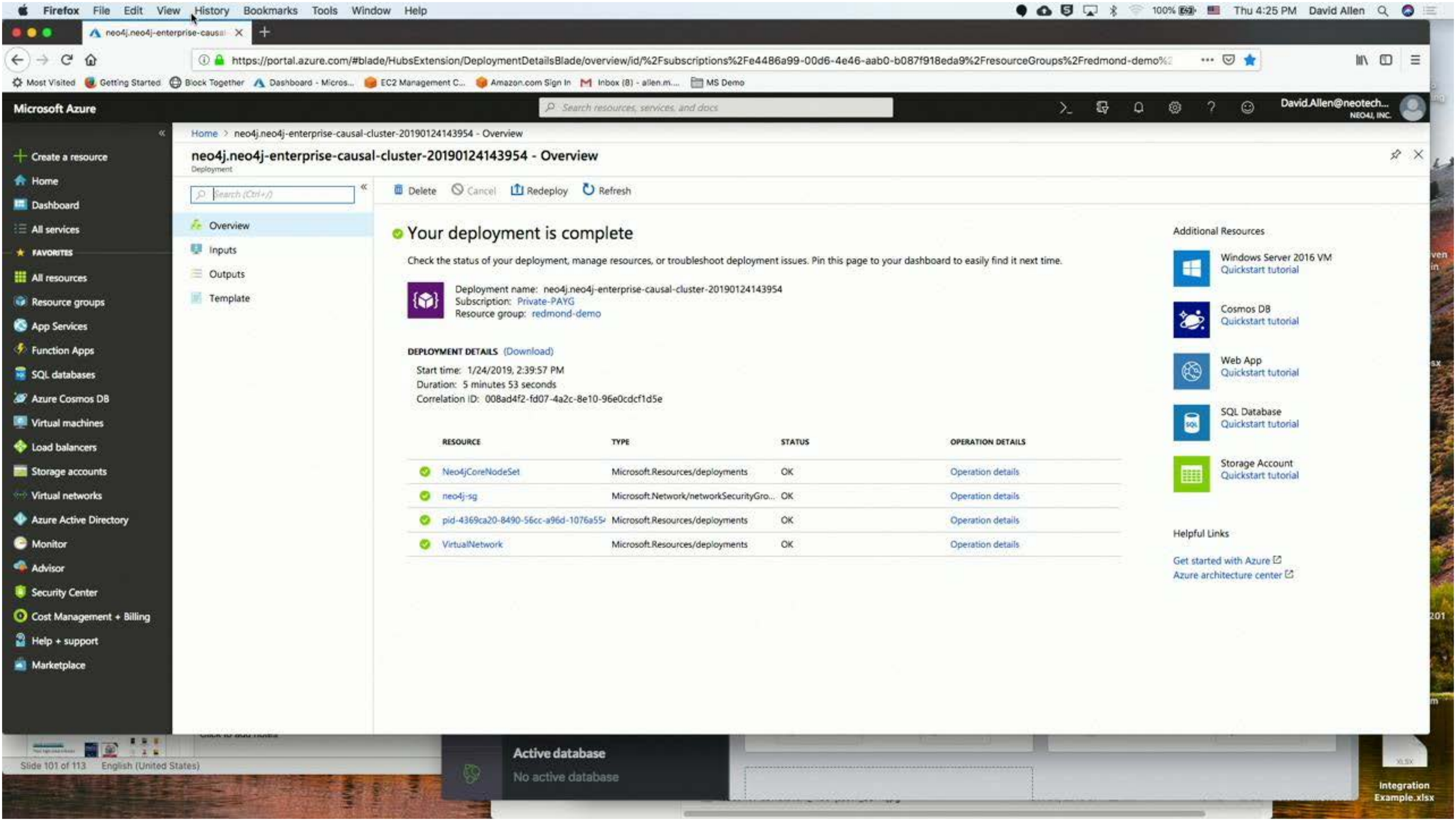
Then you define the relationships as a list:

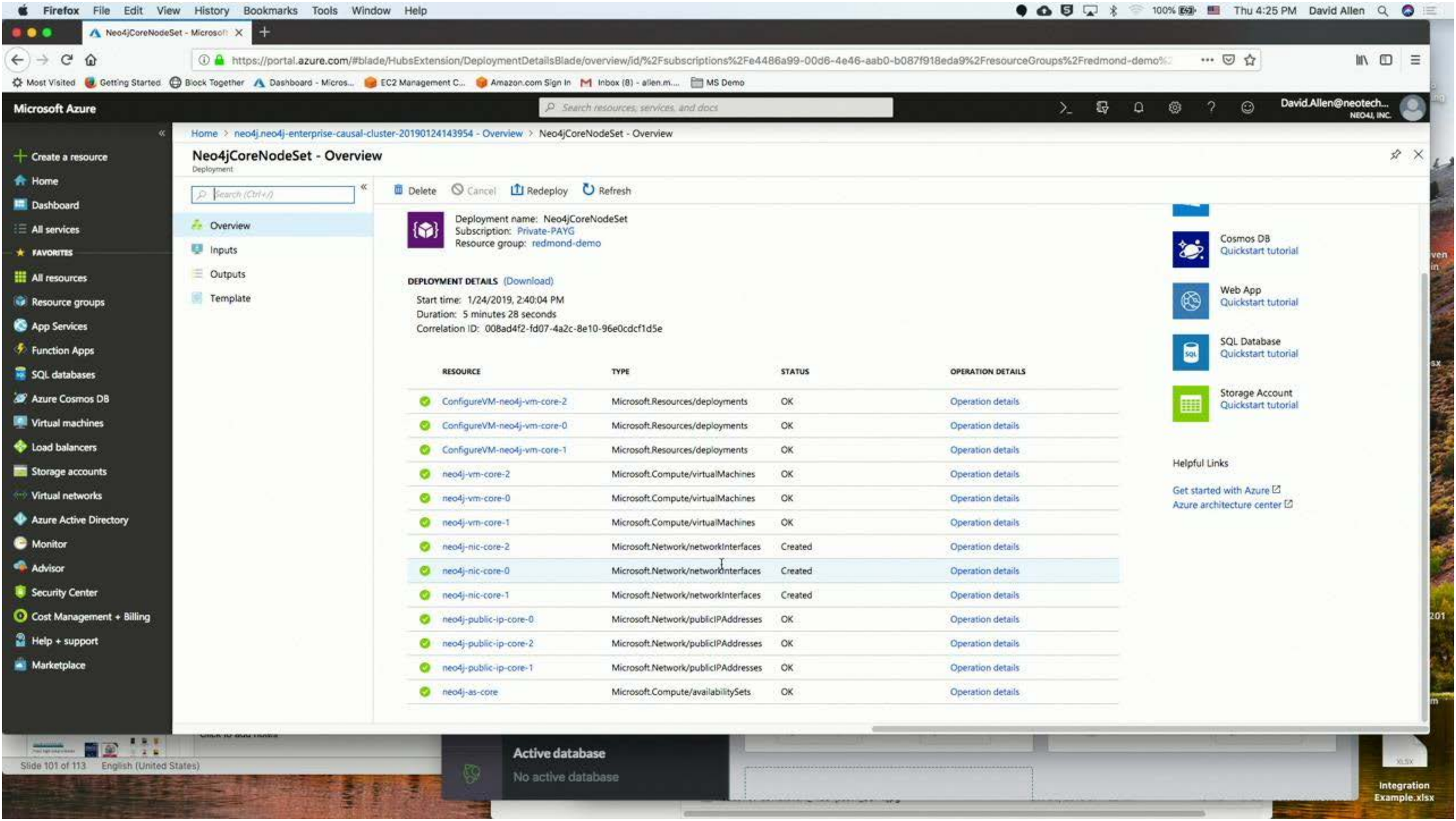
```
"report, reports to, manager",
"manager, manages, report",
…
```

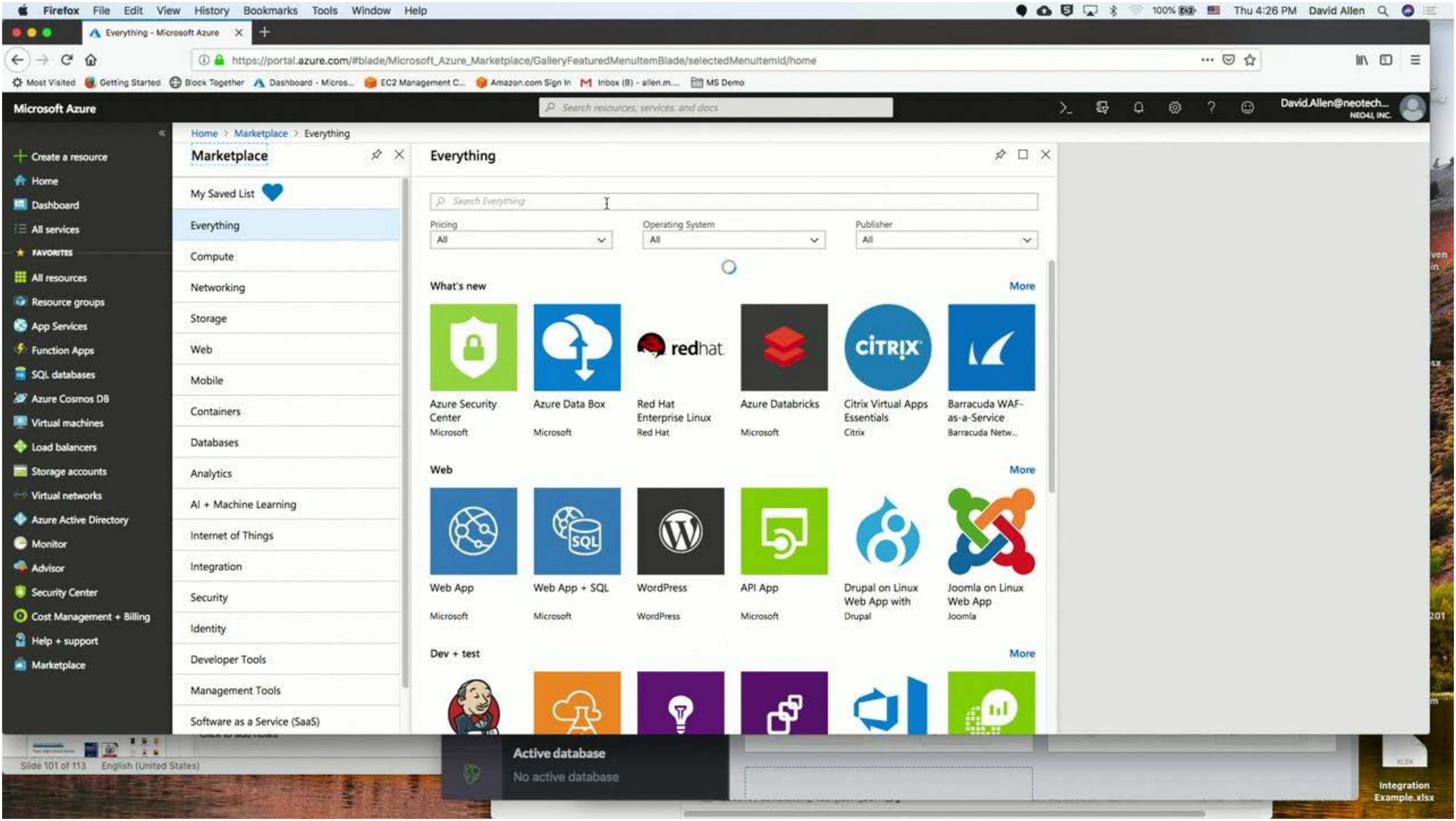
In this case, "reports to" and "manages" are the relationship names. If you want to add properties to the relationship, you can specify it in the dictionary.

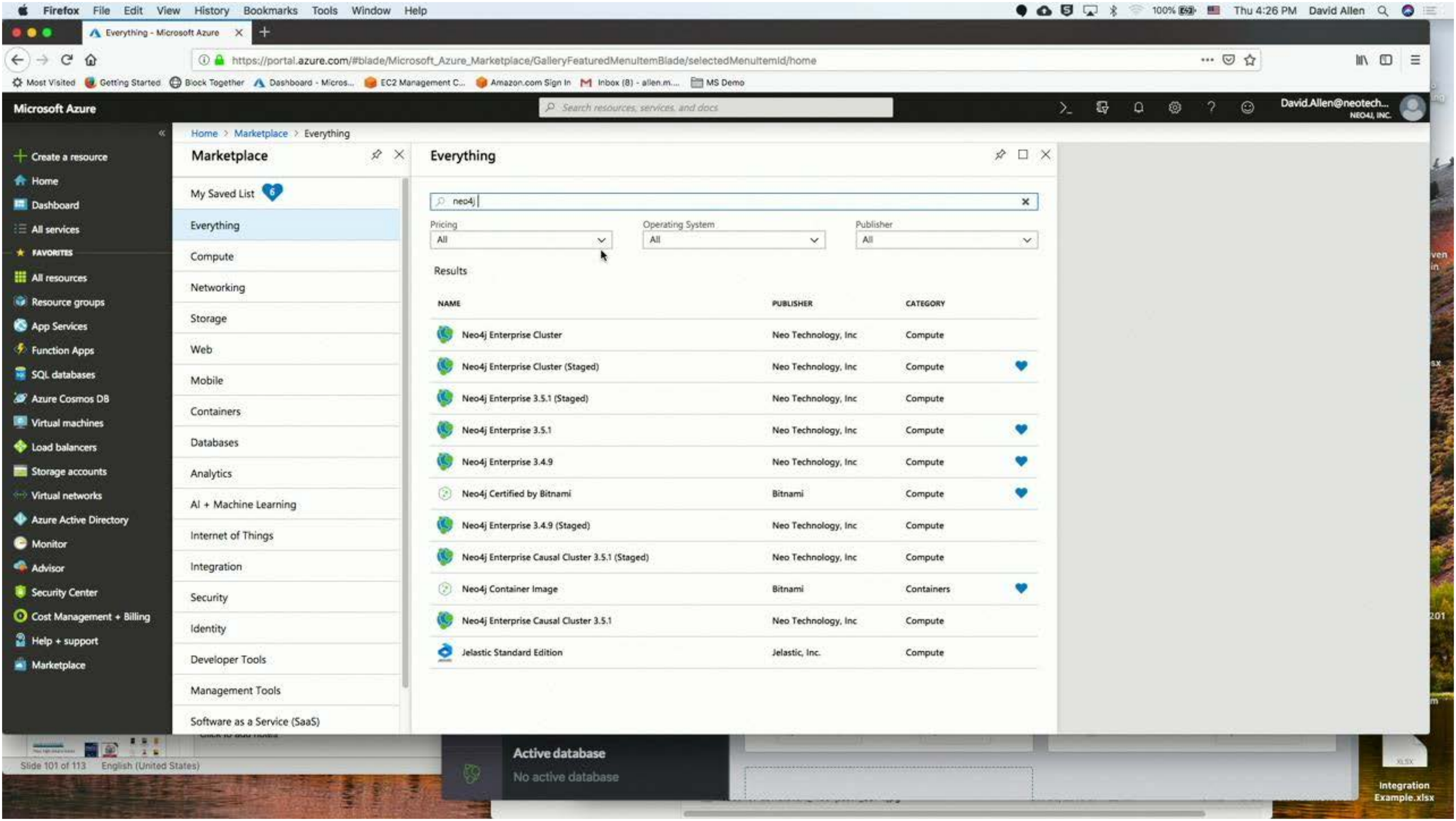
4:20 PM
Thursday
1/24/2019

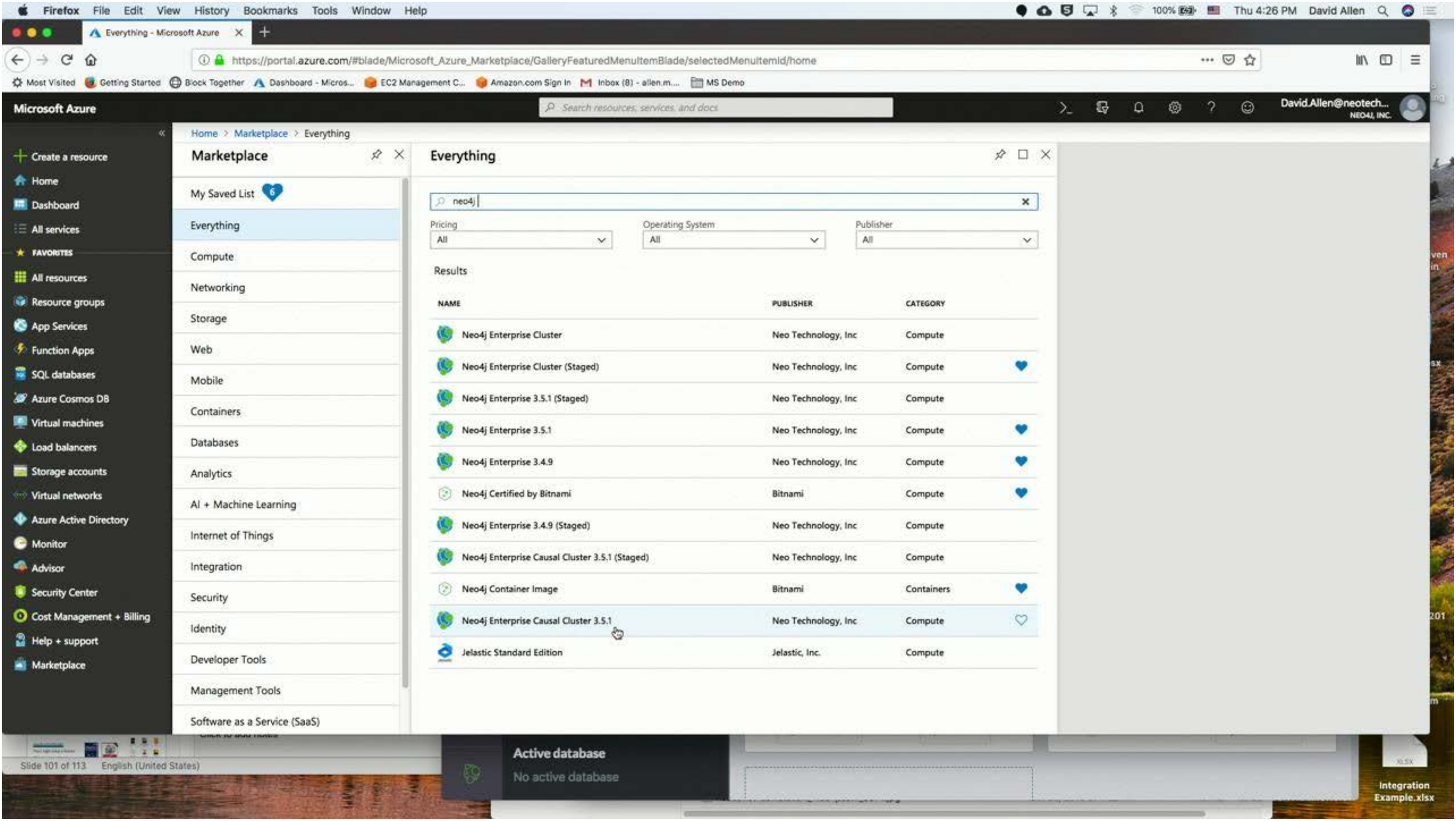

Neo4j Desktop Edit View Window Help Developer
Thu 4:25 PM David Allen
Microsoft Azure
Search resources, services, and docs
David.Allen@neotech...
NEO4J, INC.
Home > neo4j.neo4j-enterprise-causal-cluster-20190124143954 - Overview
neo4j.neo4j-enterprise-causal-cluster-20190124143954 - Overview
Deployment
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
Neo4j Desktop - 1.1.13
Projects
New
Company Meta Graph
Russian Twitter Trolls
Scratch Space
Adams Election Data
Redmond
Redmond
Neo4j Browser
3.2.15
Neo4j Bloom
1.0.5
Halin Neo4j Monitoring
0.7.1
Add Application
Microsoft Demo Graph
Neo4j 3.5.2
179 nodes (3 labels)
267 relationships (7 types)
Manage
Start
Azure Deploy
neo4j@bolt://52.160.123.2:7687
Open Browser
Activate
Active database
No active database
Slide 101 of 113
English (United States)
integration Example.xlsx

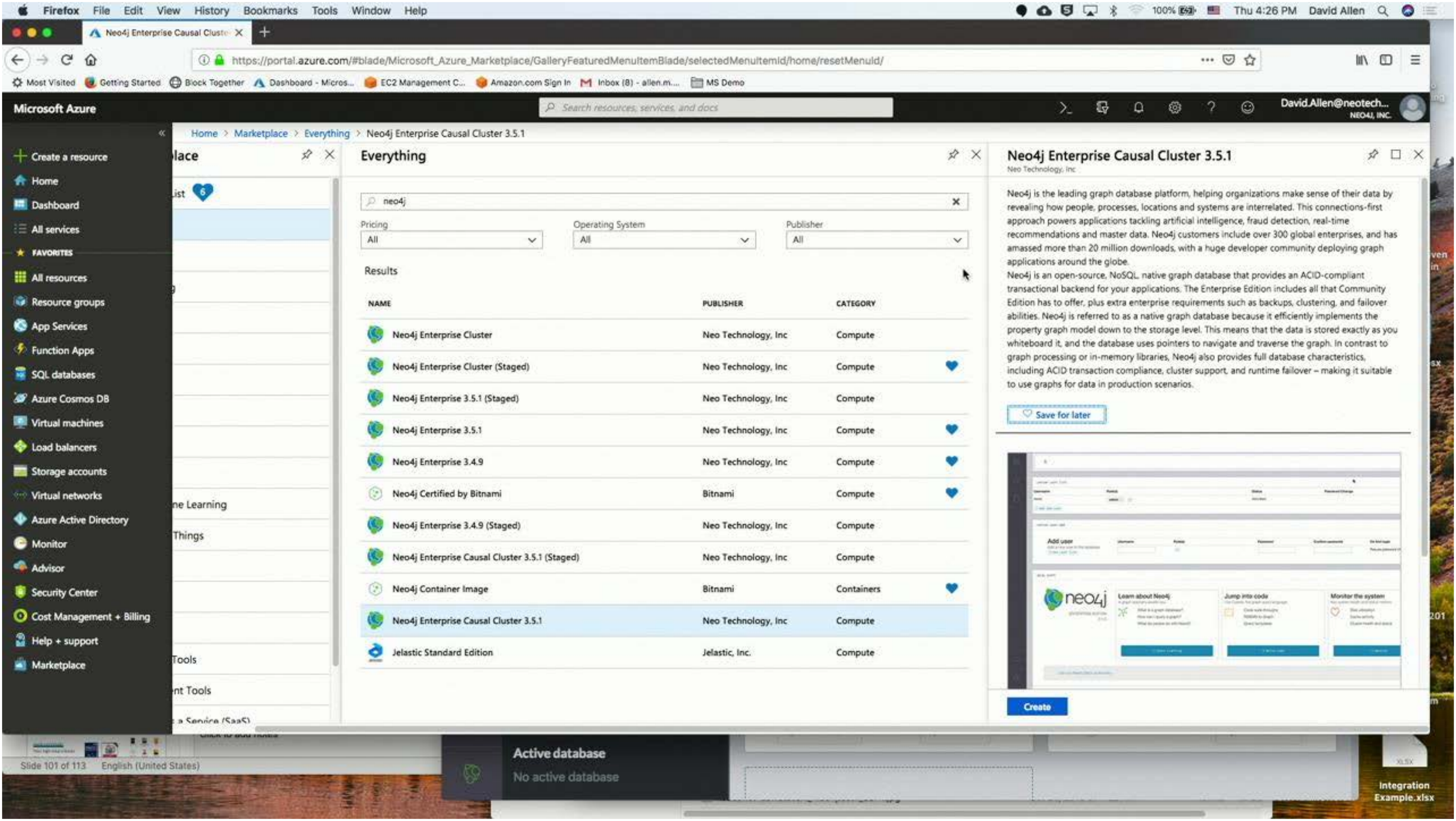

Firefox File Edit View History Bookmarks Tools Window Help
Thu 4:25 PM David Allen
neo4j.neo4j-enterprise-causal
https://portal.azure.com/#blade/HubsExtension/DeploymentDetailsBlade/overview/id/%2Fsubscriptions%2Fe4486a99-00d6-4e46-aab0-b087f918eda9%2FresourceGroups%2Fredmond-demo%2
Most Visited Getting Started Block Together Dashboard - Micros... EC2 Management C... Amazon.com Sign In Inbox (8) - allen.m.... MS Demo
Microsoft Azure
Search resources, services, and docs
David.Allen@neotech...
NEO4J, INC.
Home > neo4j.neo4j-enterprise-causal-cluster-20190124143954 - Overview
neo4j.neo4j-enterprise-causal-cluster-20190124143954 - Overview
Deployment
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
Search (Ctrl+/)
Overview
Inputs
Outputs
Template
Delete
Cancel
Redeploy
Refresh
Your deployment is complete
Check the status of your deployment, manage resources, or troubleshoot deployment issues. Pin this page to your dashboard to easily find it next time.
Deployment name: neo4j.neo4j-enterprise-causal-cluster-20190124143954
Subscription: Private-PAYG
Resource group: redmond-demo
DEPLOYMENT DETAILS (Download)
Start time: 1/24/2019, 2:39:57 PM
Duration: 5 minutes 53 seconds
Correlation ID: 008ad4f2-fd07-4a2c-8e10-96e0cdcf1d5e
RESOURCE
TYPE
STATUS
OPERATION DETAILS
Neo4jCoreNodeSet
Microsoft.Resources/deployments
OK
Operation details
neo4j-sg
Microsoft.Network/networkSecurityGro...
OK
Operation details
pid-4369ca20-8490-56cc-a96d-1076a55
Microsoft.Resources/deployments
OK
Operation details
VirtualNetwork
Microsoft.Resources/deployments
OK
Operation details
Additional Resources
Windows Server 2016 VM
Quickstart tutorial
Cosmos DB
Quickstart tutorial
Web App
Quickstart tutorial
SQL Database
Quickstart tutorial
Storage Account
Quickstart tutorial
Helpful Links
Get started with Azure
Azure architecture center
Active database
No active database
Slide 101 of 113 English (United States)
integration Example.xlsx

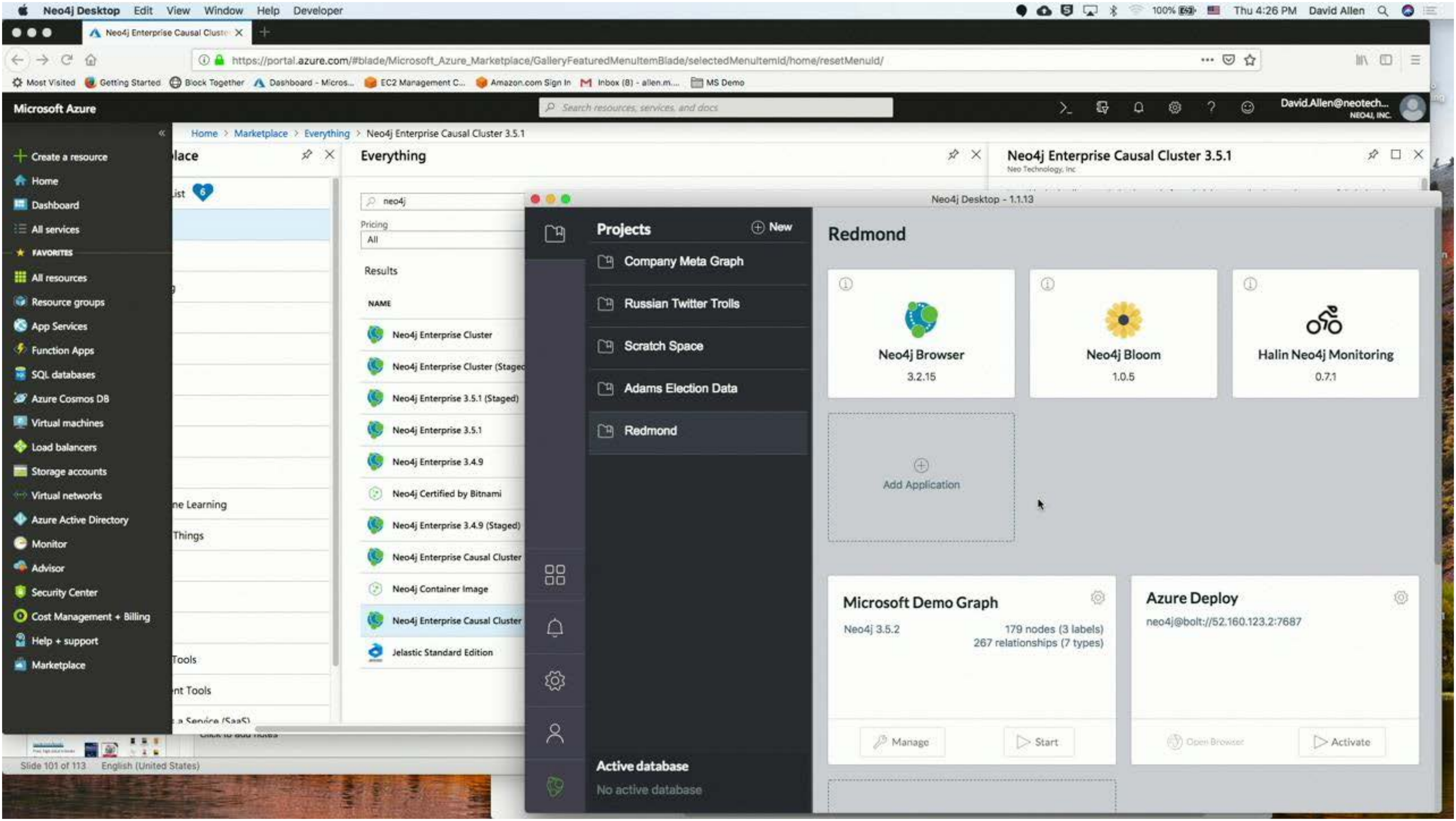

Firefox File Edit View History Bookmarks Tools Window Help
Thu 4:25 PM David Allen
Neo4jCoreNodeSet - Microsoft
https://portal.azure.com/#blade/HubsExtension/DeploymentDetailsBlade/overview/id/%2Fsubscriptions%2Fe4486a99-00d6-4e46-aab0-b087f918eda9%2FresourceGroups%2Fredmond-demo%
Most Visited Getting Started Block Together Dashboard - Micros... EC2 Management C... Amazon.com Sign In Inbox (8) - allen.m.... MS Demo
Microsoft Azure
Search resources, services, and docs
David.Allen@neotech...
NEO4J, INC.
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
Home > neo4j.neo4j-enterprise-causal-cluster-20190124143954 - Overview > Neo4jCoreNodeSet - Overview
Neo4jCoreNodeSet - Overview
Deployment
Search (Ctrl+/)
Overview
Inputs
Outputs
Template
Delete
Cancel
Redeploy
Refresh
Deployment name: Neo4jCoreNodeSet
Subscription: Private-PAYG
Resource group: redmond-demo
DEPLOYMENT DETAILS (Download)
Start time: 1/24/2019, 2:40:04 PM
Duration: 5 minutes 28 seconds
Correlation ID: 008ad4f2-fd07-4a2c-8e10-96e0cdcf1d5e
RESOURCE TYPE STATUS OPERATION DETAILS
ConfigureVM-neo4j-vm-core-2 Microsoft.Resources/deployments OK Operation details
ConfigureVM-neo4j-vm-core-0 Microsoft.Resources/deployments OK Operation details
ConfigureVM-neo4j-vm-core-1 Microsoft.Resources/deployments OK Operation details
neo4j-vm-core-2 Microsoft.Compute/virtualMachines OK Operation details
neo4j-vm-core-0 Microsoft.Compute/virtualMachines OK Operation details
neo4j-vm-core-1 Microsoft.Compute/virtualMachines OK Operation details
neo4j-nic-core-2 Microsoft.Network/networkInterfaces Created Operation details
neo4j-nic-core-0 Microsoft.Network/networkInterfaces Created Operation details
neo4j-nic-core-1 Microsoft.Network/networkInterfaces Created Operation details
neo4j-public-ip-core-0 Microsoft.Network/publicIPAddresses OK Operation details
neo4j-public-ip-core-2 Microsoft.Network/publicIPAddresses OK Operation details
neo4j-public-ip-core-1 Microsoft.Network/publicIPAddresses OK Operation details
neo4j-as-core Microsoft.Compute/availabilitySets OK Operation details
Cosmos DB Quickstart tutorial
Web App Quickstart tutorial
SQL Database Quickstart tutorial
Storage Account Quickstart tutorial
Helpful Links
Get started with Azure
Azure architecture center
Active database
No active database
Slide 101 of 113 English (United States)
integration Example.xlsx

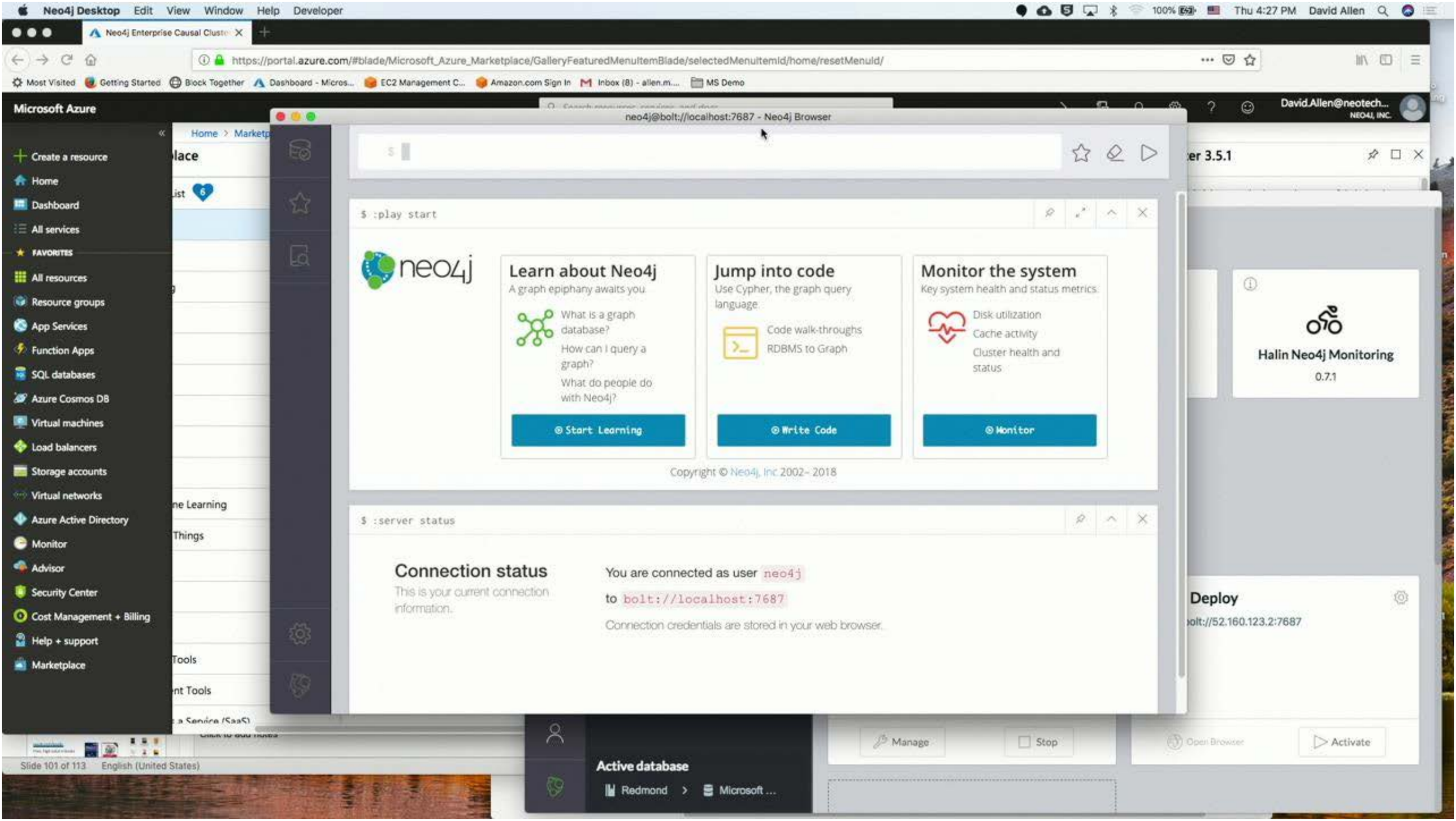

Firefox File Edit View History Bookmarks Tools Window Help
Thu 4:26 PM David Allen
Everything - Microsoft Azure
https://portal.azure.com/#blade/Microsoft_Azure_Marketplace/GalleryFeaturedMenuItemBlade/selectedMenuItemId/home
Most Visited Getting Started Block Together Dashboard - Micros... EC2 Management C... Amazon.com Sign In Inbox (8) - allen.m.... MS Demo
Microsoft Azure
Search resources, services, and docs
David.Allen@neotech...
NEO4J, INC.
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
Home > Marketplace > Everything
Marketplace
My Saved List
Everything
Compute
Networking
Storage
Web
Mobile
Containers
Databases
Analytics
AI + Machine Learning
Internet of Things
Integration
Security
Identity
Developer Tools
Management Tools
Software as a Service (SaaS)
Everything
Search Everything
Pricing
All
Operating System
All
Publisher
All
What's new
More
Azure Security Center
Microsoft
Azure Data Box
Microsoft
Red Hat Enterprise Linux
Red Hat
Azure Databricks
Microsoft
Citrix Virtual Apps Essentials
Citrix
Barracuda WAF-as-a-Service
Barracuda Netw...
Web
More
Web App
Microsoft
Web App + SQL
Microsoft
WordPress
WordPress
API App
Microsoft
Drupal on Linux Web App with
Drupal
Joomla on Linux Web App
Joomla
Dev + test
More
Active database
No active database
Slide 101 of 113 English (United States)
integration Example.xlsx

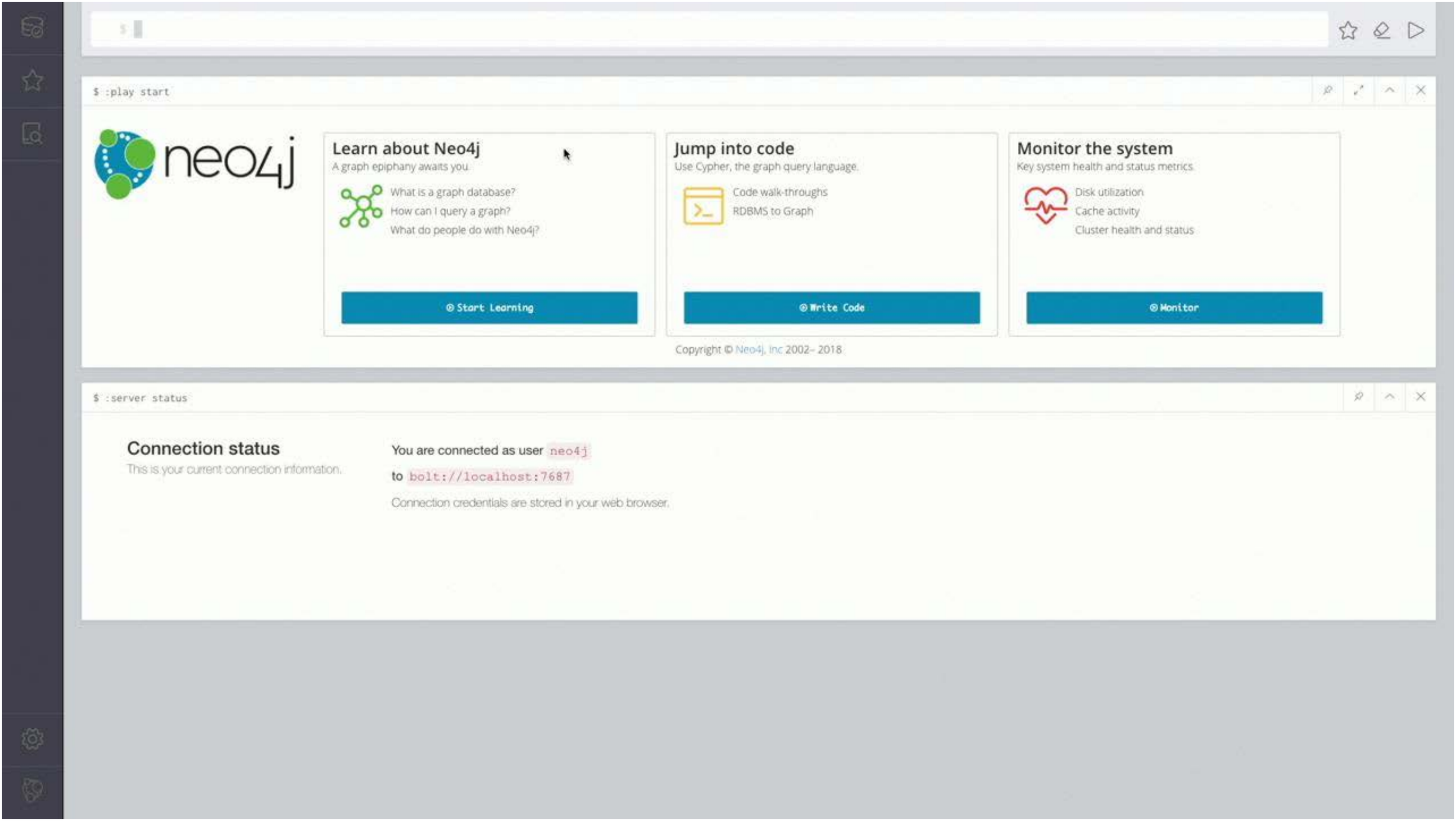

Firefox File Edit View History Bookmarks Tools Window Help
Thu 4:26 PM David Allen
Everything - Microsoft Azure
https://portal.azure.com/#blade/Microsoft_Azure_Marketplace/GalleryFeaturedMenuItemBlade/selectedMenuItemId/home
Most Visited Getting Started Block Together Dashboard - Micros... EC2 Management C... Amazon.com Sign In Inbox (8) - allen.m.... MS Demo
Microsoft Azure
Search resources, services, and docs
David.Allen@neotech...
NEO4J, INC.
Home > Marketplace > Everything
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
Marketplace
My Saved List 6
Everything
Compute
Networking
Storage
Web
Mobile
Containers
Databases
Analytics
AI + Machine Learning
Internet of Things
Integration
Security
Identity
Developer Tools
Management Tools
Software as a Service (SaaS)
Everything
neo4j
Pricing
All
Operating System
All
Publisher
All
Results
NAME
PUBLISHER
CATEGORY
Neo4j Enterprise Cluster Neo Technology, Inc Compute
Neo4j Enterprise Cluster (Staged) Neo Technology, Inc Compute
Neo4j Enterprise 3.5.1 (Staged) Neo Technology, Inc Compute
Neo4j Enterprise 3.5.1 Neo Technology, Inc Compute
Neo4j Enterprise 3.4.9 Neo Technology, Inc Compute
Neo4j Certified by Bitnami Bitnami Compute
Neo4j Enterprise 3.4.9 (Staged) Neo Technology, Inc Compute
Neo4j Enterprise Causal Cluster 3.5.1 (Staged) Neo Technology, Inc Compute
Neo4j Container Image Bitnami Containers
Neo4j Enterprise Causal Cluster 3.5.1 Neo Technology, Inc Compute
Jelastic Standard Edition Jelastic, Inc. Compute
Active database
No active database
Slide 101 of 113 English (United States)
integration Example.xlsx


Firefox File Edit View History Bookmarks Tools Window Help
Thu 4:26 PM David Allen
Everything - Microsoft Azure
https://portal.azure.com/#blade/Microsoft_Azure_Marketplace/GalleryFeaturedMenuItemBlade/selectedMenuItemId/home
Most Visited Getting Started Block Together Dashboard - Micros... EC2 Management C... Amazon.com Sign In Inbox (8) - allen.m.... MS Demo
Microsoft Azure
Search resources, services, and docs
David.Allen@neotech...
NEO4J, INC.
Home > Marketplace > Everything
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
Marketplace
My Saved List 6
Everything
Compute
Networking
Storage
Web
Mobile
Containers
Databases
Analytics
AI + Machine Learning
Internet of Things
Integration
Security
Identity
Developer Tools
Management Tools
Software as a Service (SaaS)
Everything
neo4j
Pricing
All
Operating System
All
Publisher
All
Results
NAME
PUBLISHER
CATEGORY
Neo4j Enterprise Cluster Neo Technology, Inc Compute
Neo4j Enterprise Cluster (Staged) Neo Technology, Inc Compute
Neo4j Enterprise 3.5.1 (Staged) Neo Technology, Inc Compute
Neo4j Enterprise 3.5.1 Neo Technology, Inc Compute
Neo4j Enterprise 3.4.9 Neo Technology, Inc Compute
Neo4j Certified by Bitnami Bitnami Compute
Neo4j Enterprise 3.4.9 (Staged) Neo Technology, Inc Compute
Neo4j Enterprise Causal Cluster 3.5.1 (Staged) Neo Technology, Inc Compute
Neo4j Container Image Bitnami Containers
Neo4j Enterprise Causal Cluster 3.5.1 Neo Technology, Inc Compute
Jelastic Standard Edition Jelastic, Inc. Compute
Active database
No active database
Slide 101 of 113 English (United States)
integration Example.xlsx


Firefox File Edit View History Bookmarks Tools Window Help
Thu 4:26 PM David Allen
Neo4j Enterprise Causal Cluster
https://portal.azure.com/#blade/Microsoft_Azure_Marketplace/GalleryFeaturedMenuItemBlade/selectedMenuItemId/home/resetMenuId/
Most Visited Getting Started Block Together Dashboard - Micros... EC2 Management C... Amazon.com Sign In Inbox (8) - allen.m.... MS Demo
Microsoft Azure
Search resources, services, and docs
David.Allen@neotech...
NEO4J, INC.
Home > Marketplace > Everything > Neo4j Enterprise Causal Cluster 3.5.1
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
Everything
neo4j
Pricing
All
Operating System
All
Publisher
All
Results
NAME
PUBLISHER
CATEGORY
Neo4j Enterprise Cluster
Neo Technology, Inc
Compute
Neo4j Enterprise Cluster (Staged)
Neo Technology, Inc
Compute
Neo4j Enterprise 3.5.1 (Staged)
Neo Technology, Inc
Compute
Neo4j Enterprise 3.5.1
Neo Technology, Inc
Compute
Neo4j Enterprise 3.4.9
Neo Technology, Inc
Compute
Neo4j Certified by Bitnami
Bitnami
Compute
Neo4j Enterprise 3.4.9 (Staged)
Neo Technology, Inc
Compute
Neo4j Enterprise Causal Cluster 3.5.1 (Staged)
Neo Technology, Inc
Compute
Neo4j Container Image
Bitnami
Containers
Neo4j Enterprise Causal Cluster 3.5.1
Neo Technology, Inc
Compute
Jelastic Standard Edition
Jelastic, Inc.
Compute
Neo4j Enterprise Causal Cluster 3.5.1
Neo Technology, Inc
Neo4j is the leading graph database platform, helping organizations make sense of their data by revealing how people, processes, locations and systems are interrelated. This connections-first approach powers applications tackling artificial intelligence, fraud detection, real-time recommendations and master data. Neo4j customers include over 300 global enterprises, and has amassed more than 20 million downloads, with a huge developer community deploying graph applications around the globe.
Neo4j is an open-source, NoSQL, native graph database that provides an ACID-compliant transactional backend for your applications. The Enterprise Edition includes all that Community Edition has to offer, plus extra enterprise requirements such as backups, clustering, and failover abilities. Neo4j is referred to as a native graph database because it efficiently implements the property graph model down to the storage level. This means that the data is stored exactly as you whiteboard it, and the database uses pointers to navigate and traverse the graph. In contrast to graph processing or in-memory libraries, Neo4j also provides full database characteristics, including ACID transaction compliance, cluster support, and runtime failover – making it suitable to use graphs for data in production scenarios.
Save for later
Create
Active database
No active database
Slide 101 of 113 English (United States)
integration Example.xlsx


Neo4j Desktop Edit View Window Help Developer
Thu 4:26 PM David Allen
Neo4j Enterprise Causal Cluster
https://portal.azure.com/#blade/Microsoft_Azure_Marketplace/GalleryFeaturedMenuItemBlade/selectedMenuItemId/home/resetMenuId/
Most Visited Getting Started Block Together Dashboard - Micros... EC2 Management C... Amazon.com Sign In Inbox (8) - allen.m.... MS Demo
Microsoft Azure
Search resources, services, and docs
David.Allen@neotech...
NEO4J, INC.
Home > Marketplace > Everything > Neo4j Enterprise Causal Cluster 3.5.1
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
Everything
neo4j
Pricing
All
Results
NAME
Neo4j Enterprise Cluster
Neo4j Enterprise Cluster (Staged)
Neo4j Enterprise 3.5.1 (Staged)
Neo4j Enterprise 3.5.1
Neo4j Enterprise 3.4.9
Neo4j Certified by Bitnami
Neo4j Enterprise 3.4.9 (Staged)
Neo4j Enterprise Causal Cluster
Neo4j Container Image
Neo4j Enterprise Causal Cluster
Jelastic Standard Edition
Neo4j Enterprise Causal Cluster 3.5.1
Neo Technology, Inc
Neo4j Desktop - 1.1.13
Projects
New
Company Meta Graph
Russian Twitter Trolls
Scratch Space
Adams Election Data
Redmond
Redmond
Neo4j Browser
3.2.15
Neo4j Bloom
1.0.5
Halin Neo4j Monitoring
0.7.1
Add Application
Microsoft Demo Graph
Neo4j 3.5.2
179 nodes (3 labels)
267 relationships (7 types)
Manage
Start
Azure Deploy
neo4j@bolt://52.160.123.2:7687
Open Browser
Activate
Active database
No active database
Slide 101 of 113 English (United States)


Neo4j Desktop Edit View Window Help Developer
Thu 4:27 PM David Allen
Neo4j Enterprise Causal Cluster
https://portal.azure.com/#blade/Microsoft_Azure_Marketplace/GalleryFeaturedMenuItemBlade/selectedMenuItemId/home/resetMenuId/
Most Visited Getting Started Block Together Dashboard - Micros... EC2 Management C... Amazon.com Sign In Inbox (8) - allen.m.... MS Demo
Microsoft Azure
David.Allen@neotech...
NEO4J, INC.
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
neo4j@bolt://localhost:7687 - Neo4j Browser
$:play start
neo4j
Learn about Neo4j
A graph epiphany awaits you
What is a graph database?
How can I query a graph?
What do people do with Neo4j?
Start Learning
Jump into code
Use Cypher, the graph query language
Code walk-throughs
RDBMS to Graph
Write Code
Monitor the system
Key system health and status metrics
Disk utilization
Cache activity
Cluster health and status
Monitor
Copyright © Neo4j, Inc 2002- 2018
$:server status
Connection status
This is your current connection information.
You are connected as user neo4j
to bolt://localhost:7687
Connection credentials are stored in your web browser.
Halin Neo4j Monitoring
0.7.1
Deploy
Active database
Redmond
Microsoft ...
Manage
Stop
Open Browser
Activate
Slide 101 of 113
English (United States)


$:play start
neo4j
Learn about Neo4j
A graph epiphany awaits you.
What is a graph database?
How can I query a graph?
What do people do with Neo4j?
Start Learning
Jump into code
Use Cypher, the graph query language.
Code walk-throughs
RDBMS to Graph
Write Code
Monitor the system
Key system health and status metrics.
Disk utilization
Cache activity
Cluster health and status
Monitor
Copyright © Neo4j, Inc 2002– 2018
$:server status
Connection status
This is your current connection information.
You are connected as user neo4j
to bolt://localhost:7687
Connection credentials are stored in your web browser.

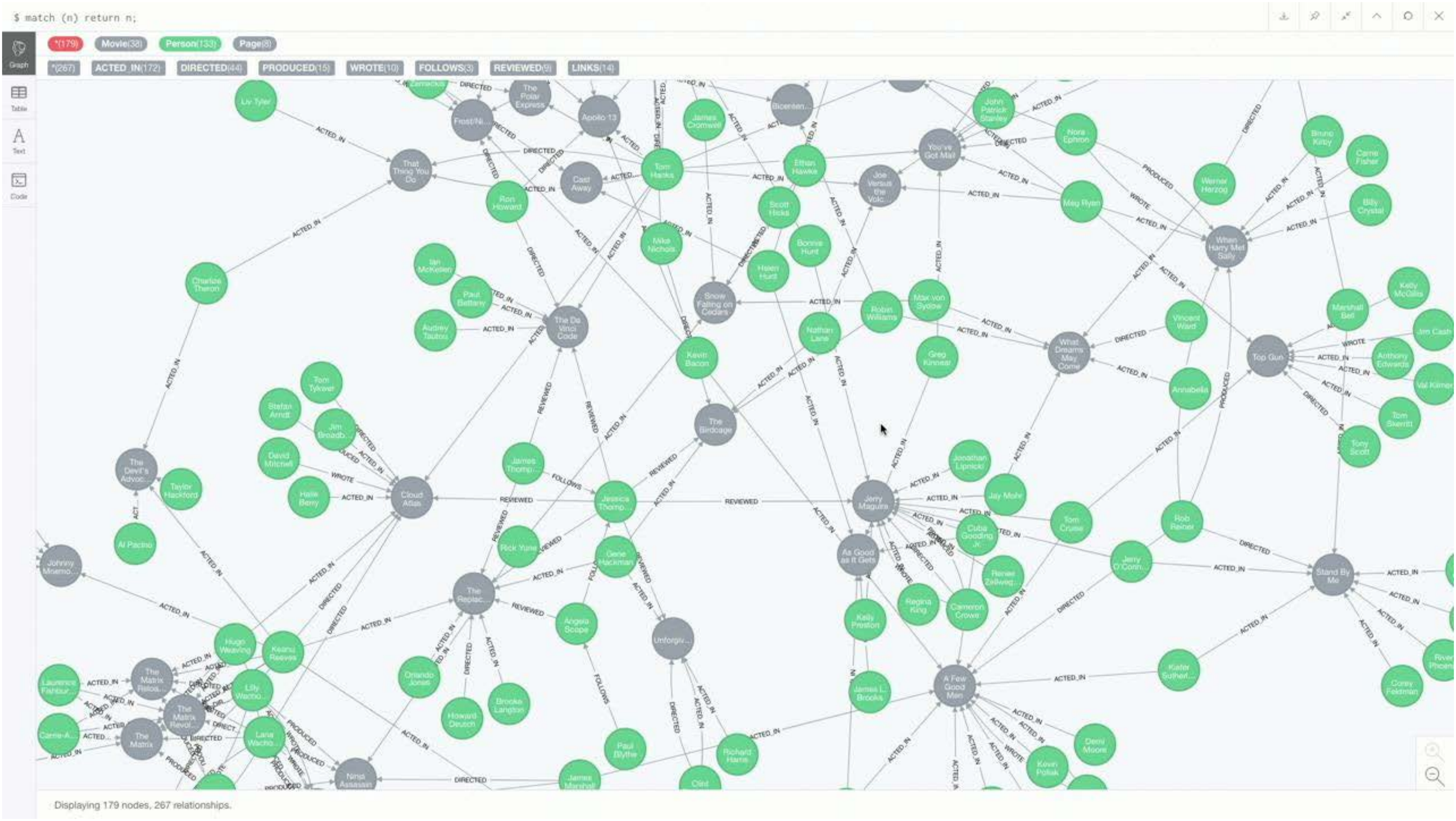

$ match (n) return n;
*(179)
Movie(38)
Person(133)
Page(8)
*(267)
ACTED_IN(172)
DIRECTED(44)
PRODUCED(15)
WROTE(10)
FOLLOWS(3)
REVIEWED(9)
LINKS(14)
Graph
Table
Text
Code
Displaying 179 nodes, 267 relationships.

Table of Contents

1. Introduction
   1.1. Algorithms
   1.2. Installation
   1.3. Usage
   1.4. Graph loading
   1.5. Building locally
2. The Yelp example
   2.1. The Yelp Open Dataset
   2.2. Data
   2.3. Graph model
   2.4. Import
   2.5. Networks
3. Procedures
4. Centrality algorithms
   4.1. The PageRank algorithm
   4.2. The Betweenness Centrality algorithm
   4.3. The Closeness Centrality algorithm
   4.4. The Harmonic Centrality algorithm
5. Community detection algorithms
   5.1. The Louvain algorithm
   5.2. The Label Propagation algorithm
   5.3. The Connected Components algorithm
   5.4. The Strongly Connected Components algorithm
   5.5. The Triangle Counting / Clustering Coefficient algorithm
6. Path finding algorithms
   6.1. The Minimum Weight Spanning Tree algorithm
   6.2. The Shortest Path algorithm
   6.3. The Single Source Shortest Path algorithm
   6.4. The All Pairs Shortest Path

The Neo4j Graph Algorithms User Guide v3.5

This is the user guide for Neo4j Graph Algorithms version 3.5, authored by the Neo4j Team.

The guide covers the following areas:

- Chapter 1, *Introduction* — An introduction to Neo4j Graph Algorithms.
- Chapter 2, *The Yelp example* — An illustration of how to use graph algorithms on a social network of friends.
- Chapter 3, *Procedures* — A list of Neo4j Graph Algorithm procedures.
- Chapter 4, *Centrality algorithms* — A detailed guide to each of the centrality algorithms, including use-cases and examples.
- Chapter 5, *Community detection algorithms* — A detailed guide to each of the community detection algorithms, including use-cases and examples.
- Chapter 6, *Path finding algorithms* — A detailed guide to each of the path finding algorithms, including use-cases and examples.
- Chapter 7, *Similarity algorithms* — A detailed guide to each of the similarity algorithms, including use-cases and examples.
- Chapter 8, *Preprocessing functions and procedures* — A detailed guide to each of the preprocessing functions and procedures.

Table of Contents

1. Introduction
   1.1. Algorithms
   1.2. Installation
   1.3. Usage
   1.4. Graph loading
   1.5. Building locally
2. The Yelp example
   2.1. The Yelp Open Dataset
   2.2. Data
   2.3. Graph model
   2.4. Import
   2.5. Networks
3. Procedures
4. Centrality algorithms
   4.1. The PageRank algorithm
   4.2. The Betweenness Centrality algorithm
   4.3. The Closeness Centrality algorithm
   4.4. The Harmonic Centrality algorithm
5. Community detection algorithms
   5.1. The Louvain algorithm
   5.2. The Label Propagation algorithm
   5.3. The Connected Components algorithm
   5.4. The Strongly Connected Components algorithm
   5.5. The Triangle Counting / Clustering Coefficient algorithm
6. Path finding algorithms
   6.1. The Minimum Weight Spanning Tree algorithm
   6.2. The Shortest Path algorithm
   6.3. The Single Source Shortest Path algorithm
   6.4. The All Pairs Shortest Path

The Neo4j Graph Algorithms User Guide v3.5

This is the user guide for Neo4j Graph Algorithms version 3.5, authored by the Neo4j Team.

The guide covers the following areas:

- Chapter 1, *Introduction* — An introduction to Neo4j Graph Algorithms.
- Chapter 2, *The Yelp example* — An illustration of how to use graph algorithms on a social network of friends.
- Chapter 3, *Procedures* — A list of Neo4j Graph Algorithm procedures.
- Chapter 4, *Centrality algorithms* — A detailed guide to each of the centrality algorithms, including use-cases and examples.
- Chapter 5, *Community detection algorithms* — A detailed guide to each of the community detection algorithms, including use-cases and examples.
- Chapter 6, *Path finding algorithms* — A detailed guide to each of the path finding algorithms, including use-cases and examples.
- Chapter 7, *Similarity algorithms* — A detailed guide to each of the similarity algorithms, including use-cases and examples.
- Chapter 8, *Preprocessing functions and procedures* — A detailed guide to each of the preprocessing functions and procedures.

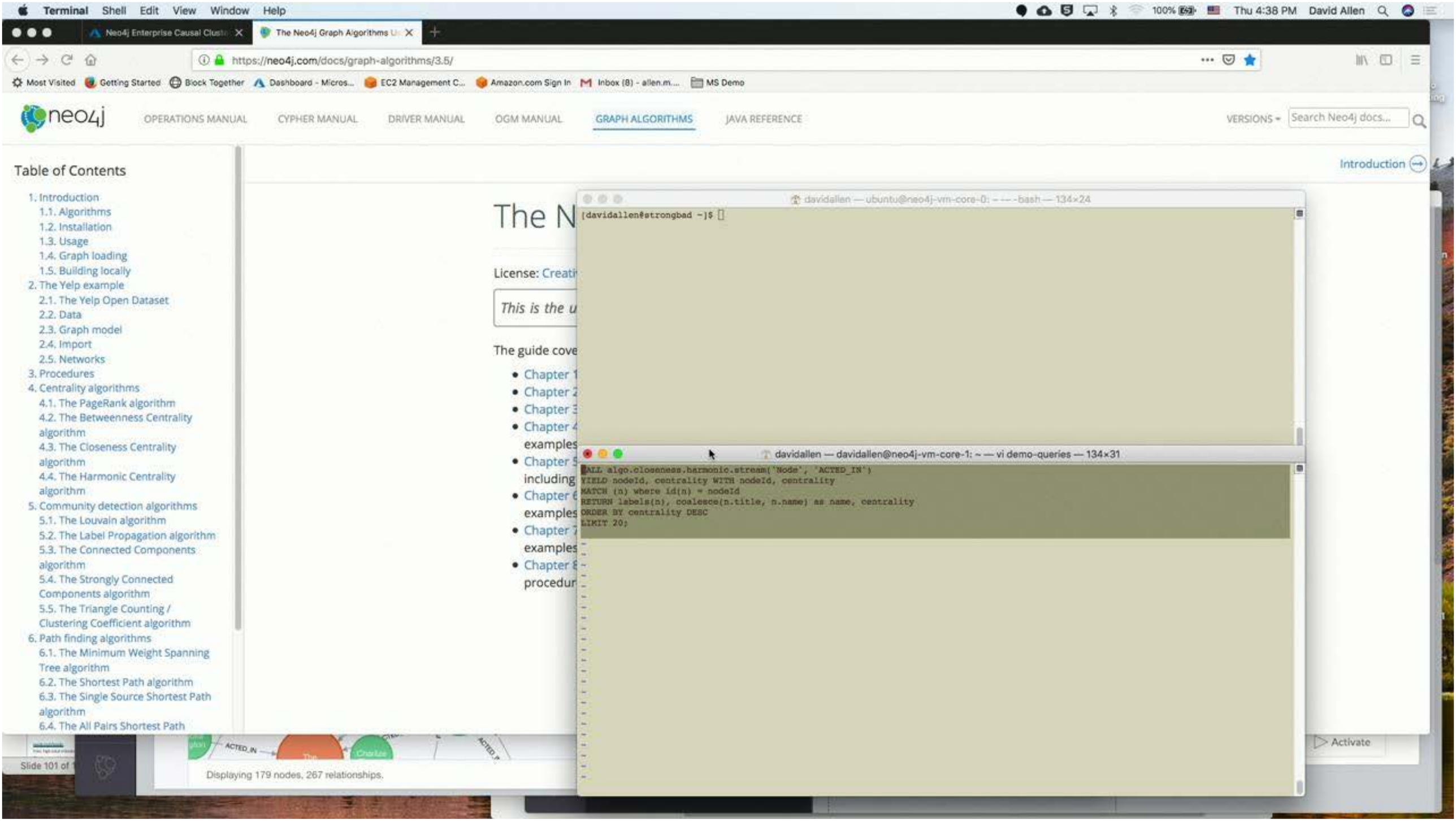

Terminal Shell Edit View Window Help
Thu 4:38 PM David Allen
https://neo4j.com/docs/graph-algorithms/3.5/
neo4j
OPERATIONS MANUAL
CYPHER MANUAL
DRIVER MANUAL
OGM MANUAL
GRAPH ALGORITHMS
JAVA REFERENCE
VERSIONS
Search Neo4j docs...
Table of Contents
1. Introduction
1.1. Algorithms
1.2. Installation
1.3. Usage
1.4. Graph loading
1.5. Building locally
2. The Yelp example
2.1. The Yelp Open Dataset
2.2. Data
2.3. Graph model
2.4. Import
2.5. Networks
3. Procedures
4. Centrality algorithms
4.1. The PageRank algorithm
4.2. The Betweenness Centrality algorithm
4.3. The Closeness Centrality algorithm
4.4. The Harmonic Centrality algorithm
5. Community detection algorithms
5.1. The Louvain algorithm
5.2. The Label Propagation algorithm
5.3. The Connected Components algorithm
5.4. The Strongly Connected Components algorithm
5.5. The Triangle Counting / Clustering Coefficient algorithm
6. Path finding algorithms
6.1. The Minimum Weight Spanning Tree algorithm
6.2. The Shortest Path algorithm
6.3. The Single Source Shortest Path algorithm
6.4. The All Pairs Shortest Path
Introduction
The N
License: Creati
This is the u
The guide cove
Chapter 1
Chapter 2
Chapter 3
Chapter 4
examples
Chapter 5
including
Chapter 6
examples
Chapter 7
examples
Chapter 8
procedur
davidallen — ubuntu@neo4j-vm-core-0: ~ — -bash — 134×24
[davidallen@strongbad ~]$
davidallen — davidallen@neo4j-vm-core-1: ~ — vi demo-queries — 134×31
CALL algo.closeness.harmonic.stream('Node', 'ACTED_IN')
YIELD nodeId, centrality WITH nodeId, centrality
MATCH (n) where id(n) = nodeId
RETURN labels(n), coalesce(n.title, n.name) as name, centrality
ORDER BY centrality DESC
LIMIT 20;
ACTED_IN
Displaying 179 nodes, 267 relationships.
Activate
Slide 101 of

Table of Contents

1. Introduction
   1.1. Algorithms
   1.2. Installation
   1.3. Usage
   1.4. Graph loading
   1.5. Building locally
2. The Yelp example
   2.1. The Yelp Open Dataset
   2.2. Data
   2.3. Graph model
   2.4. Import
   2.5. Networks
3. Procedures
4. Centrality algorithms
   4.1. The PageRank algorithm
   4.2. The Betweenness Centrality algorithm
   4.3. The Closeness Centrality algorithm
   4.4. The Harmonic Centrality algorithm
5. Community detection algorithms
   5.1. The Louvain algorithm
   5.2. The Label Propagation algorithm
   5.3. The Connected Components algorithm
   5.4. The Strongly Connected Components algorithm
   5.5. The Triangle Counting / Clustering Coefficient algorithm
6. Path finding algorithms
   6.1. The Minimum Weight Spanning Tree algorithm
   6.2. The Shortest Path algorithm
   6.3. The Single Source Shortest Path algorithm
   6.4. The All Pairs Shortest Path

The Neo4j Graph Algorithms User Guide v3.5

This is the user guide for Neo4j Graph Algorithms version 3.5, authored by the Neo4j Team.

The guide covers the following areas:

- Chapter 1, *Introduction* — An introduction to Neo4j Graph Algorithms.
- Chapter 2, *The Yelp example* — An illustration of how to use graph algorithms on a social network of friends.
- Chapter 3, *Procedures* — A list of Neo4j Graph Algorithm procedures.
- Chapter 4, *Centrality algorithms* — A detailed guide to each of the centrality algorithms, including use-cases and examples.
- Chapter 5, *Community detection algorithms* — A detailed guide to each of the community detection algorithms, including use-cases and examples.
- Chapter 6, *Path finding algorithms* — A detailed guide to each of the path finding algorithms, including use-cases and examples.
- Chapter 7, *Similarity algorithms* — A detailed guide to each of the similarity algorithms, including use-cases and examples.
- Chapter 8, *Preprocessing functions and procedures* — A detailed guide to each of the preprocessing functions and procedures.

Neo4j Desktop Edit View Window Help Developer

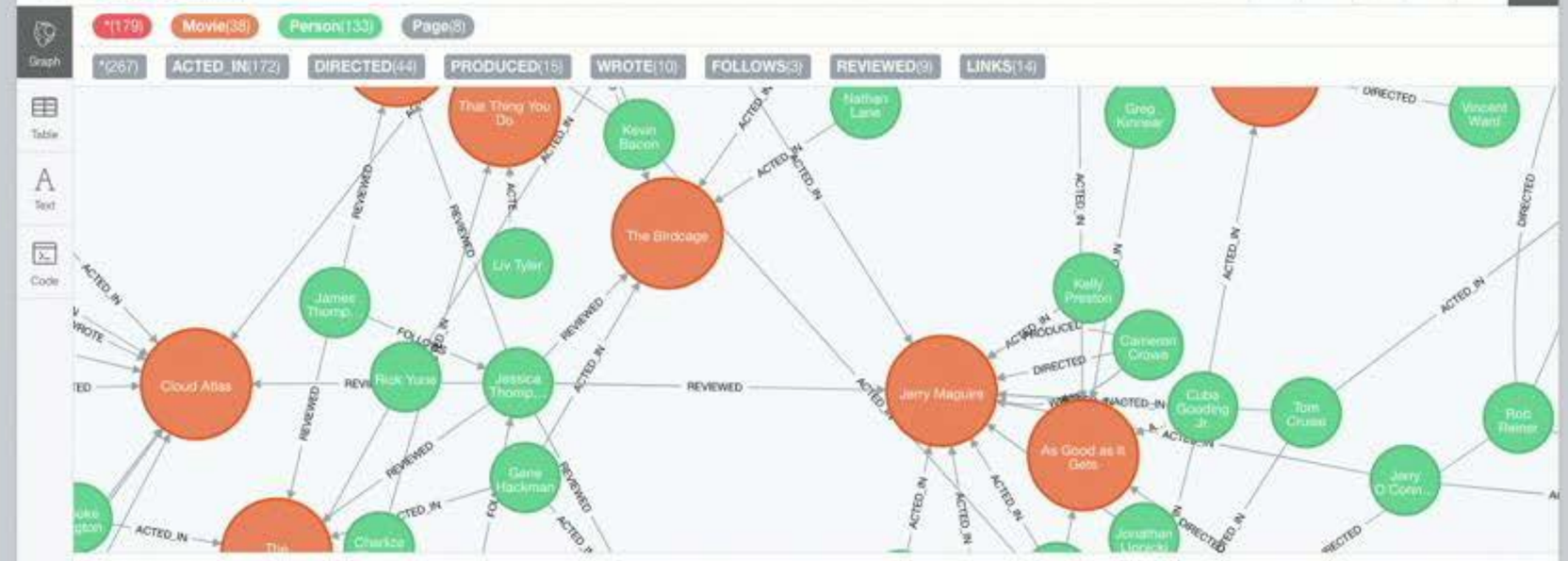
Thu 4:38 PM David Allen

Neo4j Browser

```
YIELD nodeId, centrality WITH nodeId, centrality
MATCH (n) where id(n) = nodeId
RETURN labels(n), coalesce(n.title, n.name) as name, centrality
ORDER BY centrality DESC
LIMIT 20;
```

Database access not available. Please use `:server connect` to establish connection. There's a graph waiting for you.

$ match (n) return n;



$:play movies

The Movie Graph

Create

To the right is a giant code block containing a single Cypher query statement composed of multiple CREATE clauses. This will create the

```
CREATE (TheMatrix:Movie {title:'The Matrix', released:1999, tagline:'Welcome to the Real World'})
CREATE (Keanu:Person {name:'Keanu Reeves', born:1964})
CREATE (Carrie:Person {name:'Carrie-Anne Moss', born:1967})
CREATE (Laurence:Person {name:'Laurence Fishburne', born:1961})
CREATE (Hugo:Person {name:'Hugo Weaving', born:1960})
CREATE (LillyW:Person {name:'Lilly Wachowski', born:1967})
```

Table of C

1. Introdu
1.1. Alg
1.2. Inst
1.3. Us
1.4. Gra
1.5. Bui
2. The Ye
2.1. The
2.2. Da
2.3. Gra
2.4. Imp
2.5. Ne
3. Proced
4. Central
4.1. The
4.2. The
algorith
4.3. The
algorith
4.4. The
algorith
5. Commu
5.1. The
5.2. The
5.3. The
algorith
5.4. The
Compo
5.5. The
Cluster
6. Path fin
6.1. The
Tree alg
6.2. The
6.3. The
algorith
6.4. The

Slide 101 of

VERSIONS

Search Neo4j docs...

Introduction

4j Bloom
1.0.5

Halin Neo4j Monitoring
0.7.1

Azure Deploy
neo4j@bolt://52.160.123.2:7687

Open Browser

Activate

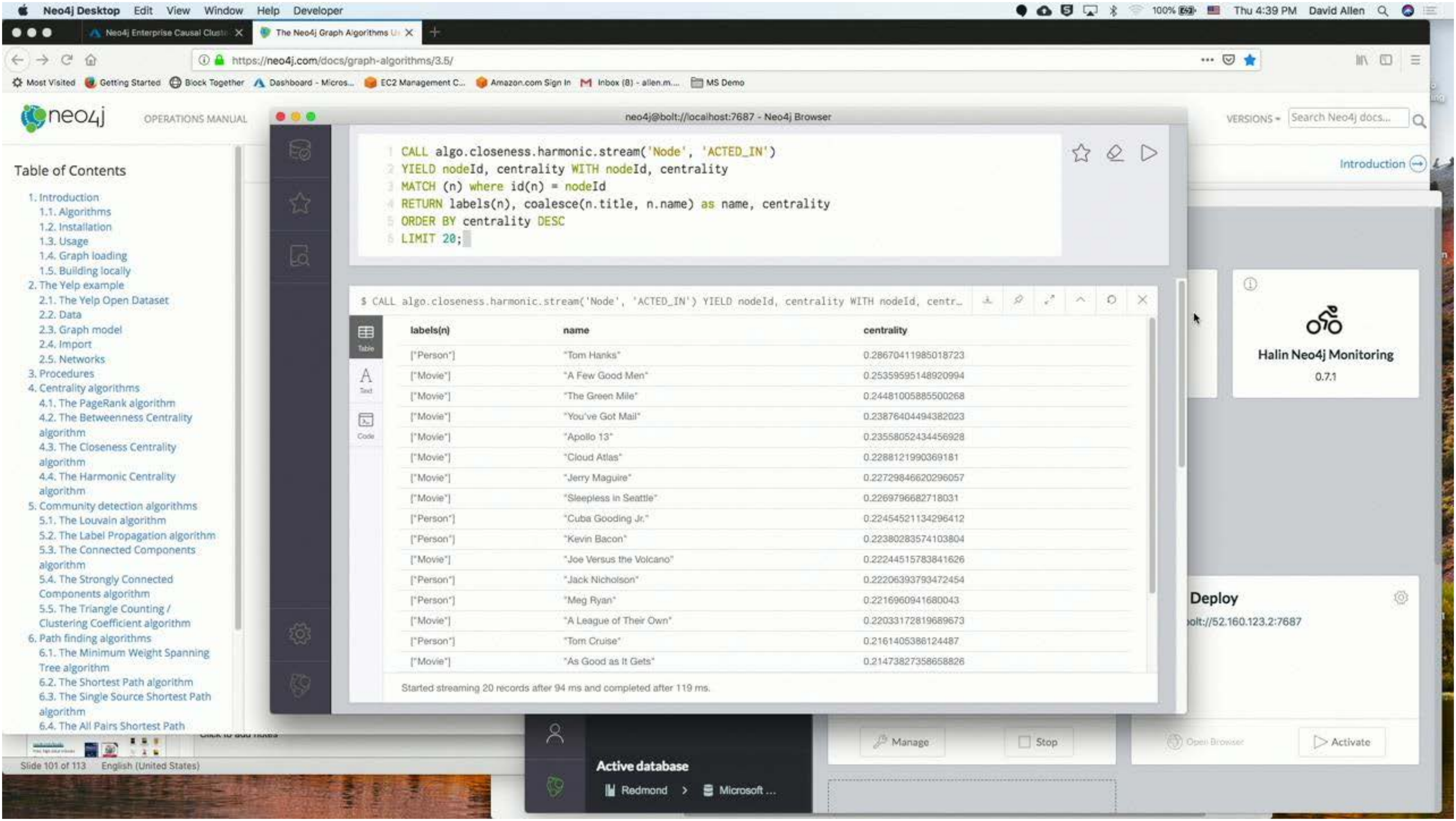

Neo4j Desktop Edit View Window Help Developer
Thu 4:39 PM David Allen
https://neo4j.com/docs/graph-algorithms/3.5/
neo4j OPERATIONS MANUAL
Table of Contents
1. Introduction
1.1. Algorithms
1.2. Installation
1.3. Usage
1.4. Graph loading
1.5. Building locally
2. The Yelp example
2.1. The Yelp Open Dataset
2.2. Data
2.3. Graph model
2.4. Import
2.5. Networks
3. Procedures
4. Centrality algorithms
4.1. The PageRank algorithm
4.2. The Betweenness Centrality algorithm
4.3. The Closeness Centrality algorithm
4.4. The Harmonic Centrality algorithm
5. Community detection algorithms
5.1. The Louvain algorithm
5.2. The Label Propagation algorithm
5.3. The Connected Components algorithm
5.4. The Strongly Connected Components algorithm
5.5. The Triangle Counting / Clustering Coefficient algorithm
6. Path finding algorithms
6.1. The Minimum Weight Spanning Tree algorithm
6.2. The Shortest Path algorithm
6.3. The Single Source Shortest Path algorithm
6.4. The All Pairs Shortest Path
neo4j@bolt://localhost:7687 - Neo4j Browser
CALL algo.closeness.harmonic.stream('Node', 'ACTED_IN')
YIELD nodeId, centrality WITH nodeId, centrality
MATCH (n) where id(n) = nodeId
RETURN labels(n), coalesce(n.title, n.name) as name, centrality
ORDER BY centrality DESC
LIMIT 20;
$ CALL algo.closeness.harmonic.stream('Node', 'ACTED_IN') YIELD nodeId, centrality WITH nodeId, centr…
Table
Text
Code
labels(n) name centrality
["Person"] "Tom Hanks" 0.28670411985018723
["Movie"] "A Few Good Men" 0.25359595148920994
["Movie"] "The Green Mile" 0.24481005885500268
["Movie"] "You've Got Mail" 0.23876404494382023
["Movie"] "Apollo 13" 0.23558052434456928
["Movie"] "Cloud Atlas" 0.2288121990369181
["Movie"] "Jerry Maguire" 0.22729846620296057
["Movie"] "Sleepless in Seattle" 0.2269796682718031
["Person"] "Cuba Gooding Jr." 0.22454521134296412
["Person"] "Kevin Bacon" 0.22380283574103804
["Movie"] "Joe Versus the Volcano" 0.22244515783841626
["Person"] "Jack Nicholson" 0.22206393793472454
["Person"] "Meg Ryan" 0.2216960941680043
["Movie"] "A League of Their Own" 0.22033172819689673
["Person"] "Tom Cruise" 0.2161405386124487
["Movie"] "As Good as It Gets" 0.21473827358658826
Started streaming 20 records after 94 ms and completed after 119 ms.
Introduction
Halin Neo4j Monitoring
0.7.1
Deploy
Manage
Stop
Open Browser
Activate
Active database
Redmond
Microsoft …
Slide 101 of 113 English (United States)

Table of Contents

1. Introduction
   1.1. Algorithms
   1.2. Installation
   1.3. Usage
   1.4. Graph loading
   1.5. Building locally
2. The Yelp example
   2.1. The Yelp Open Dataset
   2.2. Data
   2.3. Graph model
   2.4. Import
   2.5. Networks
3. Procedures
4. Centrality algorithms
   4.1. The PageRank algorithm
   4.2. The Betweenness Centrality algorithm
   4.3. The Closeness Centrality algorithm
   4.4. The Harmonic Centrality algorithm
5. Community detection algorithms
   5.1. The Louvain algorithm
   5.2. The Label Propagation algorithm
   5.3. The Connected Components algorithm
   5.4. The Strongly Connected Components algorithm
   5.5. The Triangle Counting / Clustering Coefficient algorithm
6. Path finding algorithms
   6.1. The Minimum Weight Spanning Tree algorithm
   6.2. The Shortest Path algorithm
   6.3. The Single Source Shortest Path algorithm
   6.4. The All Pairs Shortest Path

The Neo4j Graph Algorithms User Guide v3.5

This is the user guide for Neo4j Graph Algorithms version 3.5, authored by the Neo4j Team.

The guide covers the following areas:

- Chapter 1, *Introduction* — An introduction to Neo4j Graph Algorithms.
- Chapter 2, *The Yelp example* — An illustration of how to use graph algorithms on a social network of friends.
- Chapter 3, *Procedures* — A list of Neo4j Graph Algorithm procedures.
- Chapter 4, *Centrality algorithms* — A detailed guide to each of the centrality algorithms, including use-cases and examples.
- Chapter 5, *Community detection algorithms* — A detailed guide to each of the community detection algorithms, including use-cases and examples.
- Chapter 6, *Path finding algorithms* — A detailed guide to each of the path finding algorithms, including use-cases and examples.
- Chapter 7, *Similarity algorithms* — A detailed guide to each of the similarity algorithms, including use-cases and examples.
- Chapter 8, *Preprocessing functions and procedures* — A detailed guide to each of the preprocessing functions and procedures.

Table of Contents

1. Introduction
   1.1. Algorithms
   1.2. Installation
   1.3. Usage
   1.4. Graph loading
   1.5. Building locally
2. The Yelp example
   2.1. The Yelp Open Dataset
   2.2. Data
   2.3. Graph model
   2.4. Import
   2.5. Networks
3. Procedures
4. Centrality algorithms
   4.1. The PageRank algorithm
   4.2. The Betweenness Centrality algorithm
   4.3. The Closeness Centrality algorithm
   4.4. The Harmonic Centrality algorithm
5. Community detection algorithms
   5.1. The Louvain algorithm
   5.2. The Label Propagation algorithm
   5.3. The Connected Components algorithm
   5.4. The Strongly Connected Components algorithm
   5.5. The Triangle Counting / Clustering Coefficient algorithm
6. Path finding algorithms
   6.1. The Minimum Weight Spanning Tree algorithm
   6.2. The Shortest Path algorithm
   6.3. The Single Source Shortest Path algorithm
   6.4. The All Pairs Shortest Path

The Neo4j Graph Algorithms User Guide v3.5

This is the user guide for Neo4j Graph Algorithms version 3.5, authored by the Neo4j Team.

The guide covers the following areas:

- Chapter 1, *Introduction* — An introduction to Neo4j Graph Algorithms.
- Chapter 2, *The Yelp example* — An illustration of how to use graph algorithms on a social network of friends.
- Chapter 3, *Procedures* — A list of Neo4j Graph Algorithm procedures.
- Chapter 4, *Centrality algorithms* — A detailed guide to each of the centrality algorithms, including use-cases and examples.
- Chapter 5, *Community detection algorithms* — A detailed guide to each of the community detection algorithms, including use-cases and examples.
- Chapter 6, *Path finding algorithms* — A detailed guide to each of the path finding algorithms, including use-cases and examples.
- Chapter 7, *Similarity algorithms* — A detailed guide to each of the similarity algorithms, including use-cases and examples.
- Chapter 8, *Preprocessing functions and procedures* — A detailed guide to each of the preprocessing functions and procedures.

Table of Contents

1. Introduction
   1.1. Algorithms
   1.2. Installation
   1.3. Usage
   1.4. Graph loading
   1.5. Building locally
2. The Yelp example
   2.1. The Yelp Open Dataset
   2.2. Data
   2.3. Graph model
   2.4. Import
   2.5. Networks
3. Procedures
4. Centrality algorithms
   4.1. The PageRank algorithm
   4.2. The Betweenness Centrality algorithm
   4.3. The Closeness Centrality algorithm
   4.4. The Harmonic Centrality algorithm
5. Community detection algorithms
   5.1. The Louvain algorithm
   5.2. The Label Propagation algorithm
   5.3. The Connected Components algorithm
   5.4. The Strongly Connected Components algorithm
   5.5. The Triangle Counting / Clustering Coefficient algorithm
6. Path finding algorithms
   6.1. The Minimum Weight Spanning Tree algorithm
   6.2. The Shortest Path algorithm
   6.3. The Single Source Shortest Path algorithm
   6.4. The All Pairs Shortest Path

The Neo4j Graph Algorithms User Guide v3.5

This is the user guide for Neo4j Graph Algorithms version 3.5, authored by the Neo4j Team.

The guide covers the following areas:

- Chapter 1, *Introduction* — An introduction to Neo4j Graph Algorithms.
- Chapter 2, *The Yelp example* — An illustration of how to use graph algorithms on a social network of friends.
- Chapter 3, *Procedures* — A list of Neo4j Graph Algorithm procedures.
- Chapter 4, *Centrality algorithms* — A detailed guide to each of the centrality algorithms, including use-cases and examples.
- Chapter 5, *Community detection algorithms* — A detailed guide to each of the community detection algorithms, including use-cases and examples.
- Chapter 6, *Path finding algorithms* — A detailed guide to each of the path finding algorithms, including use-cases and examples.
- Chapter 7, *Similarity algorithms* — A detailed guide to each of the similarity algorithms, including use-cases and examples.
- Chapter 8, *Preprocessing functions and procedures* — A detailed guide to each of the preprocessing functions and procedures.

Table of Contents

1. Introduction
   1.1. Algorithms
   1.2. Installation
   1.3. Usage
   1.4. Graph loading
   1.5. Building locally
2. The Yelp example
   2.1. The Yelp Open Dataset
   2.2. Data
   2.3. Graph model
   2.4. Import
   2.5. Networks
3. Procedures
4. Centrality algorithms
   4.1. The PageRank algorithm
   4.2. The Betweenness Centrality algorithm
   4.3. The Closeness Centrality algorithm
   4.4. The Harmonic Centrality algorithm
5. Community detection algorithms
   5.1. The Louvain algorithm
   5.2. The Label Propagation algorithm
   5.3. The Connected Components algorithm
   5.4. The Strongly Connected Components algorithm
   5.5. The Triangle Counting / Clustering Coefficient algorithm
6. Path finding algorithms
   6.1. The Minimum Weight Spanning Tree algorithm
   6.2. The Shortest Path algorithm
   6.3. The Single Source Shortest Path algorithm
   6.4. The All Pairs Shortest Path

The Neo4j Graph Algorithms User Guide v3.5

This is the user guide for Neo4j Graph Algorithms version 3.5, authored by the Neo4j Team.

The guide covers the following areas:

- Chapter 1, *Introduction* — An introduction to Neo4j Graph Algorithms.
- Chapter 2, *The Yelp example* — An illustration of how to use graph algorithms on a social network of friends.
- Chapter 3, *Procedures* — A list of Neo4j Graph Algorithm procedures.
- Chapter 4, *Centrality algorithms* — A detailed guide to each of the centrality algorithms, including use-cases and examples.
- Chapter 5, *Community detection algorithms* — A detailed guide to each of the community detection algorithms, including use-cases and examples.
- Chapter 6, *Path finding algorithms* — A detailed guide to each of the path finding algorithms, including use-cases and examples.
- Chapter 7, *Similarity algorithms* — A detailed guide to each of the similarity algorithms, including use-cases and examples.
- Chapter 8, *Preprocessing functions and procedures* — A detailed guide to each of the preprocessing functions and procedures.

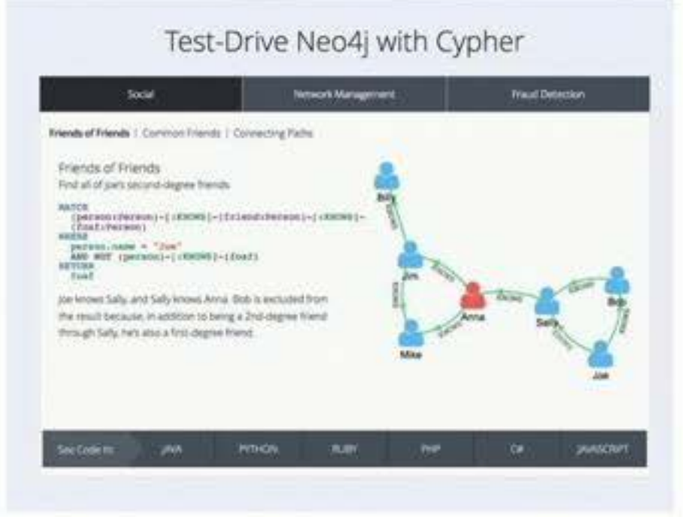
More ways to learn about Neo4j

Developer Pages

neo4j

neo4j.com/developer

- Intro to Graphs
- RDBMS to Graphs
- Data Import & Data Modeling
- Language Guides & Drivers
- Visualization
- Neo4j Ecosystem



Graph Academy

neo4j

neo4j.com/graphacademy

- Online & Classroom Training
- University Program
- Certification
- Webinars
- Graph Days, Graph Talks

Become a Neo4j Certified Professional

neo4j

neo4j.com/graphacademy/neo4j-certification/

1 hour certification exam covers:

- Cypher
- Data Modeling
- Clustering
- and more...

Cypher Reference Card

neo4j.com/docs/cypher-refcard/current/

Developer Documentation

neo4j

neo4j.com/docs

neo4j.com/docs/developer-manual/current

neo4j.com/docs/operations-manual/current

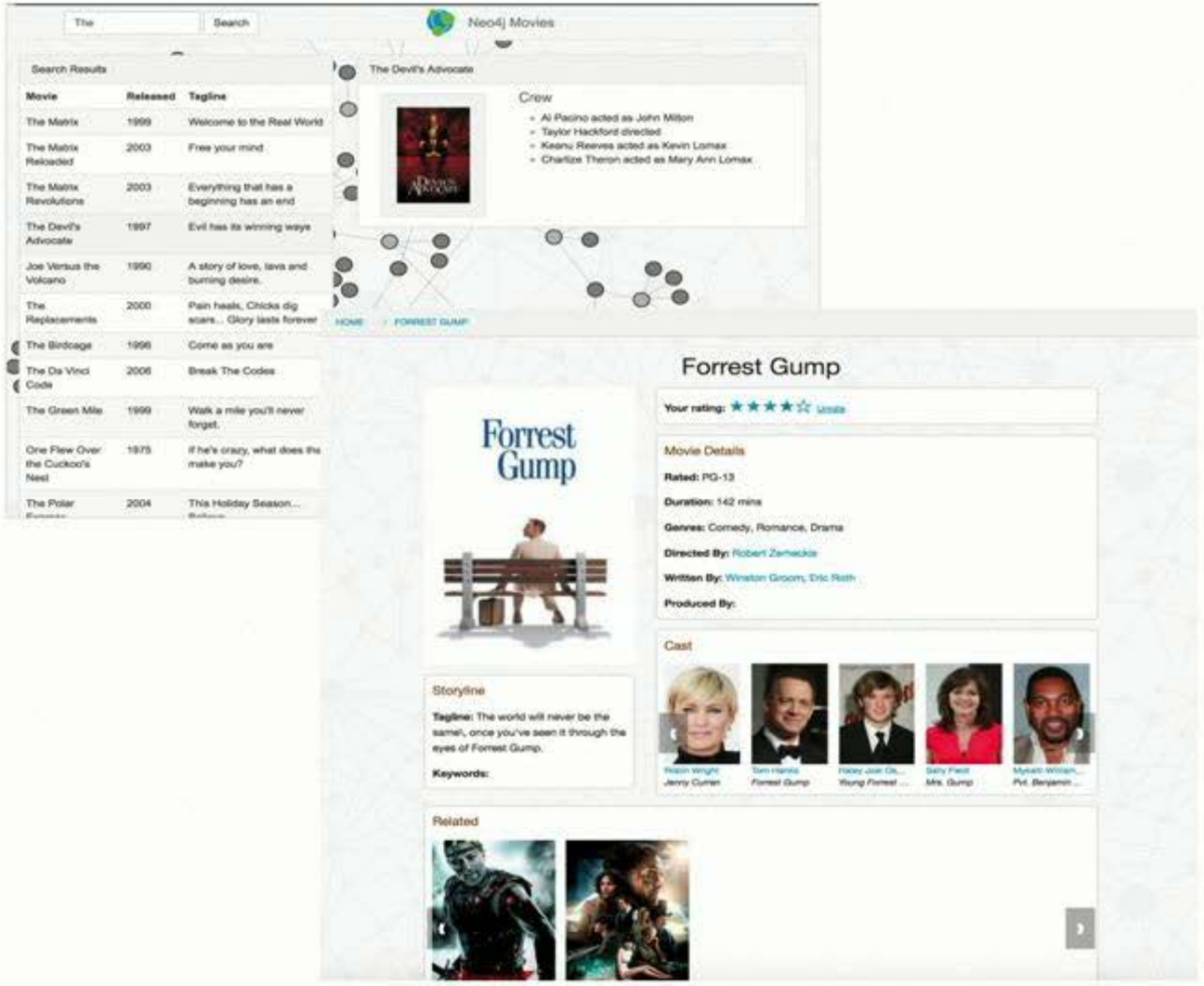
Knowledge Base

neo4j.com/developer/kb

- Frequently Asked Questions and Answers
- Canonical Source
- Maintained by our Support and Field team
- Tagged
- Version specific

Example Applications

github.com/neo4j-examples



Meetup Groups

neo4j

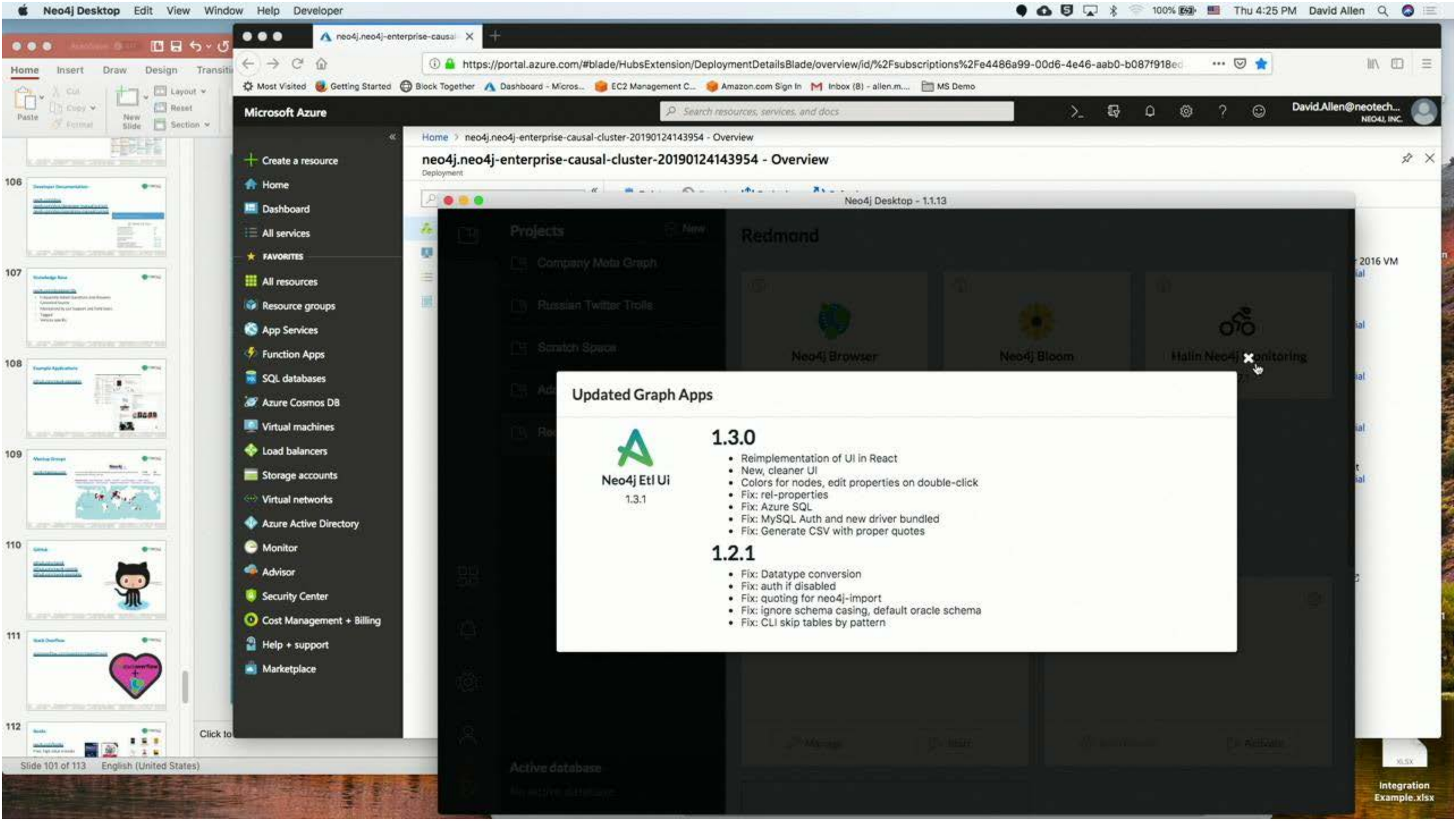
neo4j.meetup.com

Neo4j

Find out what's happening in Neo4j Meetup groups around the world and start meeting up with the ones near you.

41,729 members | **108** Meetups

Related topics: Graph Databases · NoSQL · Big Data · Data Visualization · Graph Theory · Database Development · Data Analytics · Database Professionals · Open Source · Neo4j-Social


Neo4j Desktop Edit View Window Help Developer
Thu 4:25 PM David Allen
Microsoft Azure
Search resources, services, and docs
David.Allen@neotech...
NEO4J, INC.
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function Apps
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Billing
Help + support
Marketplace
Home > neo4j.neo4j-enterprise-causal-cluster-20190124143954 - Overview
neo4j.neo4j-enterprise-causal-cluster-20190124143954 - Overview
Deployment
Neo4j Desktop - 1.1.13
Projects
Redmond
Neo4j Browser
Neo4j Bloom
Updated Graph Apps
Neo4j Etl Ui
1.3.1
1.3.0
Reimplementation of UI in React
New, cleaner UI
Colors for nodes, edit properties on double-click
Fix: rel-properties
Fix: Azure SQL
Fix: MySQL Auth and new driver bundled
Fix: Generate CSV with proper quotes
1.2.1
Fix: Datatype conversion
Fix: auth if disabled
Fix: quoting for neo4j-import
Fix: ignore schema casing, default oracle schema
Fix: CLI skip tables by pattern
Slide 101 of 113
English (United States)
integration Example.xlsx

GitHub

github.com/neo4j

github.com/neo4j-contrib

github.com/neo4j-examples

Stack Overflow

neo4j

stackoverflow.com/questions/tagged/neo4j

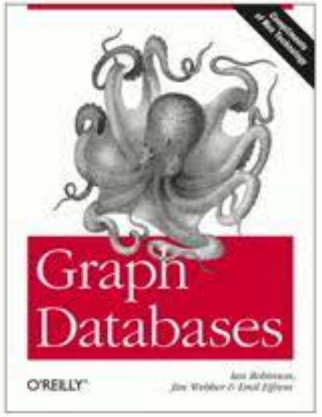
Books

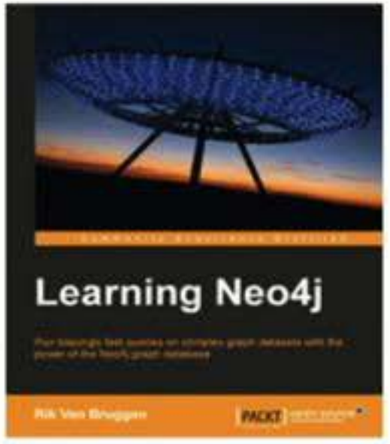
neo4j

neo4j.com/books

Free, high value e-books

Third party publications





Q&A

Table of Contents

1. Introduction
   1.1. What is Cypher?
   1.2. Querying and updating the graph
   1.3. Transactions
   1.4. Uniqueness
2. Syntax
   2.1. Values and types
   2.2. Naming rules and recommendations
   2.3. Expressions
   2.4. Variables
   2.5. Reserved keywords
   2.6. Parameters
   2.7. Operators
   2.8. Comments
   2.9. Patterns
   2.10. Temporal (Date/Time) values
   2.11. Lists
   2.12. Maps
   2.13. Spatial values
   2.14. Working with `null`
3. Clauses
   3.1. MATCH
   3.2. OPTIONAL MATCH
   3.3. START
   3.4. RETURN
   3.5. WITH
   3.6. UNWIND
   3.7. WHERE
   3.8. ORDER BY
   3.9. SKIP
   3.10. LIMIT
   3.11. CREATE
   3.12. DELETE
   3.13. SET
   3.14. REMOVE

2.10.3.1. Specifying temporal instants

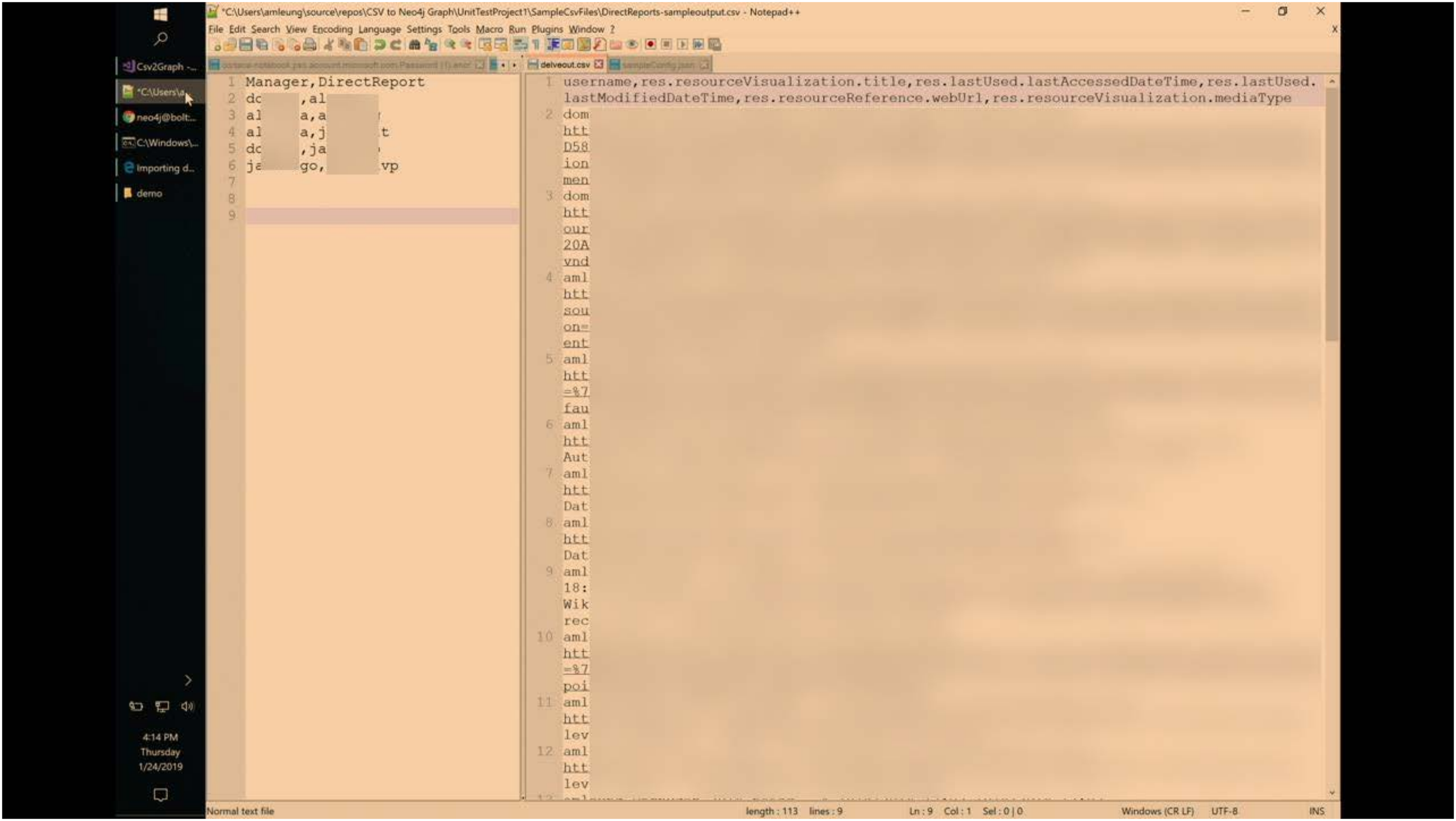
A temporal instant consists of three parts; the `date`, the `time`, and the `timezone`. These parts may then be combined to produce the various temporal value types. Literal characters are denoted in **`bold`**.

Temporal instant type	Composition of parts
Date	`<date>`
Time	`<time><timezone>` or `T` `<time><timezone>`
LocalTime	`<time>` or `T` `<time>`
*DateTime**	`<date>` `T` `<time><timezone>`
*LocalDateTime**	`<date>` `T` `<time>`

*When `date` and `time` are combined, `date` must be complete; i.e. fully identify a particular day.

Specifying dates

Component	Format	Description
Year	`YYYY`	Specified with at least four digits (special rules apply in certain cases)
Month	`MM`	Specified with a double digit number from `01` to `12`
Week	`ww`	Always prefixed with `W` and specified with a double digit number from `01` to `53`
Quarter	`q`	Always prefixed with `Q` and specified with a single digit number from `1` to `4`
Day of the month	`DD`	Specified with a double digit number from `01` to `31`


*C:\Users\amleung\source\repos\CSV to Neo4j Graph\UnitTestProject1\SampleCsvFiles\DirectReports-sampleoutput.csv - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
delveout.csv
Manager,DirectReport
username,res.resourceVisualization.title,res.lastUsed.lastAccessedDateTime,res.lastUsed.
lastModifiedDateTime,res.resourceReference.webUrl,res.resourceVisualization.mediaType
Normal text file
length : 113 lines : 9
Ln : 9 Col : 1 Sel : 0 | 0
Windows (CR LF)
UTF-8
INS
Csv2Graph -...
*C:\Users\a...
neo4j@bolt:...
C:\Windows\...
Importing d...
demo
4:14 PM
Thursday
1/24/2019