

# Graphs All The Way Down

Building A GraphQL API  
Backed By A Graph Database

William Lyon  
@lyonwj  
lyonwj.com

QCon  
NEW YORK



# William Lyon

Developer Relations Engineer @neo4j

[will@neo4j.com](mailto:will@neo4j.com)

[@lyonwj](https://twitter.com/lyonwj)

[lyonwj.com](https://lyonwj.com)

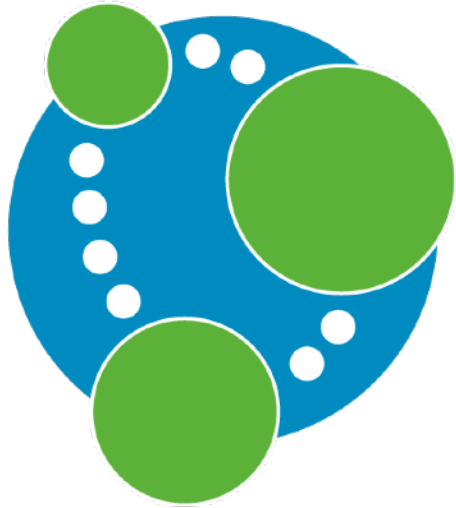


# Agenda



- Graph databases - Neo4j
- Intro to GraphQL
- Why I'm excited about GraphQL + Graph Databases
- neo4j-graphql





neo4j

[neo4j.com/developer](https://neo4j.com/developer)

# Neo4j Graph Database



**William Lyon**  
@lyonwj



#Neo4jIn140Characters

**William Lyon** @lyonwj

Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:45 PM - 23 Feb 2017



**William Lyon**

@lyonwj



Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:44 PM - 23 Feb 2017 from Dunlap, Seattle



Reply to @lyonwj



This repository

Search

Pull requests

Issues

Gist



neo4j / neo4j

Watch

326

Unstar

3,502

Fork

1,079

Code

Issues 379

Pull requests 33

Projects 0

Wiki

Pulse

Graphs

Graphs for Everyone <http://neo4j.com>

45,336 commits

18 branches

173 releases

141 contributors



**William Lyon**

@lyonwj



Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:44 PM - 23 Feb 2017 from [Dunlap, Seattle](#)



Reply to @lyonwj

talks.json

Raw

```
1 {
2   "title": "12 Years of Spring: An Open Source Journey",
3   "abstract": "Spring emerged as a core open source project in early 2003 and evolved to a broad par
4   "topics": ["keynote", "spring"],
5   "room": "Auditorium",
6   "timeslot": "Wed 29th, 09:30-10:30",
7   "speakers": {
8     "name": "Juergen Hoeller",
9     "bio": "Juergen Hoeller is co-founder of the Spring Framework open source project and has been s
10    "twitter": "https://twitter.com/springjuergen",
11    "picture": "http://www.spring.io.net/wp-content/uploads/2014/11/juergen_hoeller-220x220.jpeg"
12  }
13 }
14 }
```





**William Lyon**

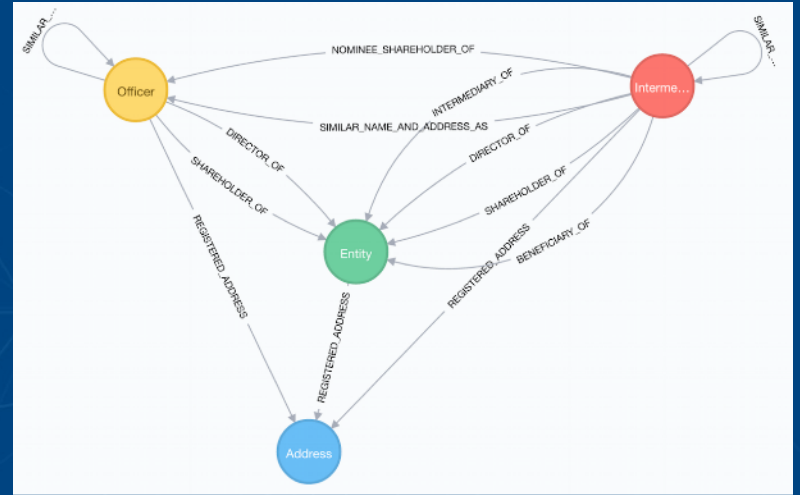
@lyonwj

Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:44 PM - 23 Feb 2017 from [Dunlap, Seattle](#)



Reply to [@lyonwj](#)







**William Lyon**

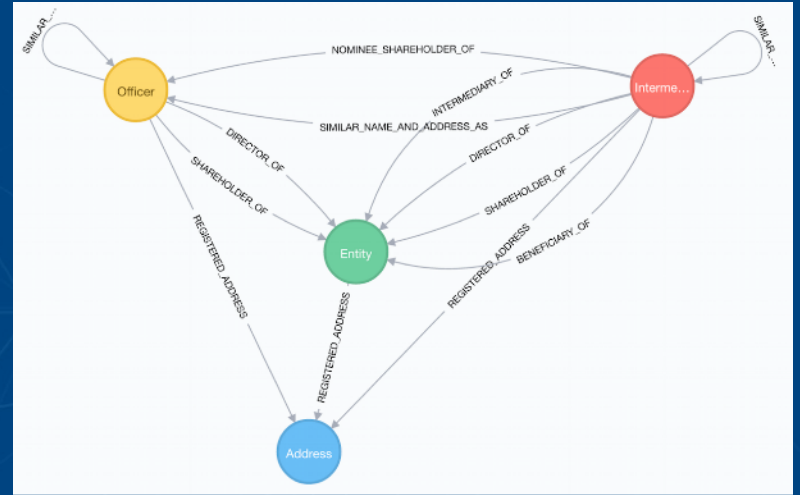
@lyonwj

Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:44 PM - 23 Feb 2017 from Dunlap, Seattle



Reply to @lyonwj



# THE PANAMA PAPERS

Continue reading... →

<https://offshoreleaks.icij.org/pages/database>



```
1 MATCH (a:Address)<-[:REGISTERED_ADDRESS]->(o:Officer)-->(e:Entity)
2 WHERE a.address CONTAINS "New York"
3 RETURN *
```



**William Lyon**

@lyonwj



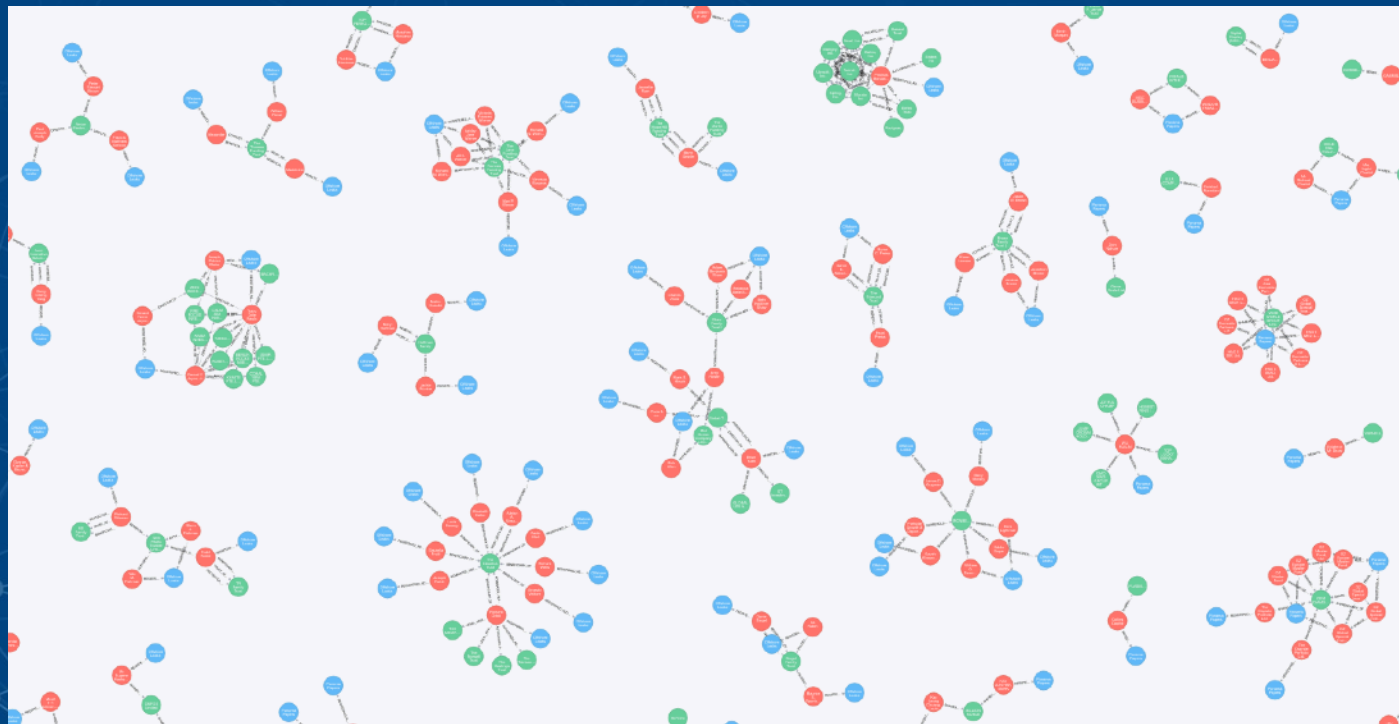
Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:44 PM - 23 Feb 2017 from [Dunlap, Seattle](#)



Reply to [@lyonwj](#)

```
1 MATCH (a:Address)<-[:REGISTERED_ADDRESS]->(o:Officer)-->(e:Entity)
2 WHERE a.address CONTAINS "New York"
3 RETURN *
```



```
1 MATCH (a:Address)<-[:REGISTERED_ADDRESS]->(o:Officer)-->(e:Entity)
2 WHERE a.address CONTAINS "New York"
3 RETURN e.jurisdiction_description AS jurisdiction, COUNT(*) AS number
4 ORDER BY number DESC LIMIT 10
```



**William Lyon**  
@lyonwj



Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:44 PM - 23 Feb 2017 from [Dunlap, Seattle](#)



Reply to [@lyonwj](#)

```
1 MATCH (a:Address)-[:REGISTERED_ADDRESS]->(o:Officer)-->(e:Entity)
2 WHERE a.address CONTAINS "New York"
3 RETURN e.jurisdiction_description AS jurisdiction, COUNT(*) AS number
4 ORDER BY number DESC LIMIT 10
```

\$ MATCH (a:Address)-[:REGIS...

	jurisdiction	number
Rows	British Virgin Islands	295
A	Cook Islands	217
Text	Undetermined	45
</>	Singapore	26
Code	Samoa	13
	Bahamas	8
	Cayman	5
	Labuan	5
	Seychelles	4
	British Anguilla	4

Started streaming 10 records after 56 ms and completed after 56 ms.



**William Lyon**  
@lyonwj

Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:44 PM - 23 Feb 2017 from Dunlap, Seattle



Reply to @lyonwj

```
1 MATCH (a:Address)<-[:REGISTERED_ADDRESS]->(o:Officer)-->(e:Entity)
2 WHERE a.address CONTAINS "New York"
3 RETURN e.jurisdiction_description AS jurisdiction, COUNT(*) AS number
4 ORDER BY number DESC LIMIT 10
```

openCypher

## What is openCypher?

The openCypher project aims to deliver a full and open specification of the industry's most widely adopted graph database query language:  
**Cypher.**

<http://www.opencypher.org/>





**William Lyon**

@lyonwj

Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:44 PM - 23 Feb 2017 from Dunlap, Seattle



Reply to @lyonwj

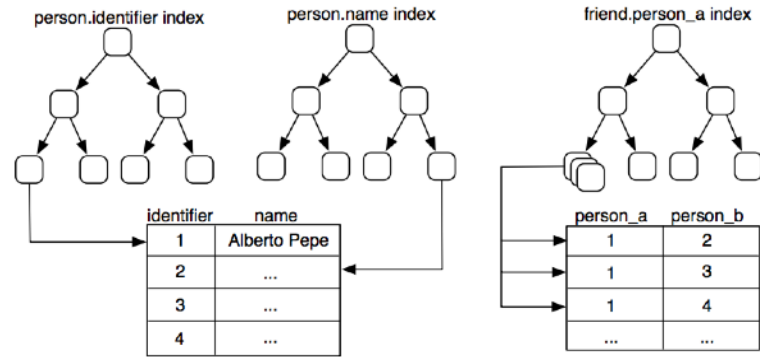


Fig. 1. A table representation of people and their friends.

# neo4jsandbox.com

Panama Papers by ICIJ

Get Started

**Details**

Data Model

Code

Advanced -

**Neo4j Browser:** <https://10-0-1-17-33122.neo4jsandbox.com/>**Direct Neo4j HTTP:** <http://34.239.248.240:33122/browser/>**Username:** neo4j**Password:** garden-recombinations-judges **IP Address:** 34.239.248.240**HTTP Port:** 33122**Bolt Port:** 33121**Expires:** 2 days, 23 hours, 59 minutes

## Launch a New Sandbox

Each sandbox includes data, interactive guides with example queries, and sample code.

### Jupyter Sandbox



Jupyter sandbox

[Launch Sandbox](#)

### Fundamentals Training



Neo4j Fundamentals classroom training with instructor-led guides

[Launch Sandbox](#)

### Neo4j 3.2



NEW Neo4j 3.2 release - Blank Sandbox

[Launch Sandbox](#)

### Neo4j GraphQL



Neo4j with GraphQL extension

[Launch Sandbox](#)

### Recommendations



Generate personalized real-time recommendations using a dataset of movie reviews.

[Launch Sandbox](#)

### NICAR 2017 Workshop



NICAR 2017 graph database workshop. Analyze campaign finance and US Congress data as a graph in Neo4j.

[Launch Sandbox](#)

### OSCON 2017 Neo4j Workshop



OSCON 2017 Neo4j workshop. Learn how to build a real time recommendation system with Neo4j

[Launch Sandbox](#)

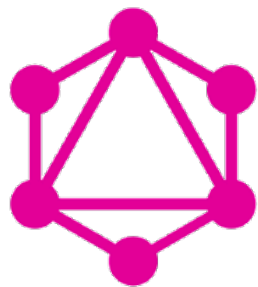
### NASA Workshop



NASA Lessons Learned knowledge graph workshop

[Launch Sandbox](#)





GraphQL

<http://graphql.org/>

# GraphQL

- “A query language for your API”
- Developed by Facebook iOS team for iOS app
  - Reduce number of round trip requests in face of low latency
- Declarative, state what fields you want
- Alternative to REST
- Self documenting (schema and types)
- Limited support for “queries”
  - Logic is implemented in server



# GraphQL

- “A query language for your API, and a server-side runtime for executing queries by using a type system you define for your data”
- “GraphQL isn't tied to any specific database or storage engine”
- “A GraphQL service is created by defining types and fields on those types, then providing functions for each field on each type”

# GraphQL Adoption



GitHub



courseera



mattermark



# GraphQL

```
{
  business(id: "YCsLfBVdLFeN2Necw1HPSA") {
    name
    address
    city
    state
    postal_code
    lat
    lon
    categories {
      name
    }
  }
}
```

GraphQL Query

```
{
  "data": {
    "business": {
      "name": "Stussy",
      "address": "1000 Queen Street W",
      "city": "Toronto",
      "state": "ON",
      "postal_code": "M6J 1H1",
      "lat": 43.644258153,
      "lon": -79.4188293813,
      "categories": [
        {
          "name": "Shopping"
        },
        {
          "name": "Fashion"
        },
        {
          "name": "Men's Clothing"
        },
        {
          "name": "Shoe Stores"
        }
      ]
    }
  }
}
```

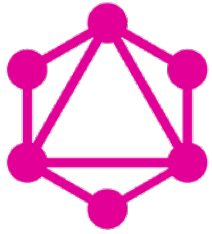
Result

# Ecosystem

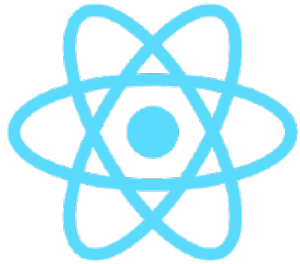
- GraphQL Clients
  - Most popular is Apollo-client
  - Also **Relay** (Relay Modern recently released at F8)
  - Apollo-client also has **iOS**, **Android** clients
  - Lokka (js)
- Frontend frameworks
  - Dominated by **React** (Fiber is react rewrite, coming soon to public)
  - Vue.js, Angular, Preact
- GraphQL-as-a-service
  - Graphcool, Scaphold, Reindex
- Language (client)
  - JavaScript dominates, but Swift (iOS) and Java (Android) emerging
- Language (server)
  - JavaScript, Ruby, Java, Scala, Go, Python, ...
- Server
  - Express.js dominates, but many other projects exist
- Tools
  - Graphiql
  - Apollo optics
  - DataLoader



# Building A Full Stack Graph Application



GraphQL

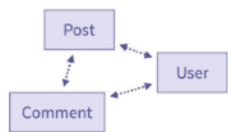


React



# GraphQL First Development

## GraphQL First Development



### 1. Design API schema

Contract between frontend and backend with a shared schema language



### 2. Build UI and backend

Parallelize with mocking, develop component-based UIs with GraphQL containers



### 3. Run in production

Static queries make loading predictable, schema tells you which fields are being used

1. Design API schema
2. Build UI and backend
3. Run in production

- Schema is your friend
- GraphQL Schema is the API spec
  - Allows for simultaneous frontend and backend development
  - Enables introspection
    - Build other tools (graphql)



# IDL Schema Syntax

```
type User {
  id: ID
  email: String!
  post(id: ID!): Post
  posts: [Post!]!
  follower(id: ID!): User
  followers: [User!]!
  followee(id: ID!): User
  followees: [User!]!
}

type Post {
  id: ID
  user: User!
  title: String!
  body: String!
  comment(id: ID!): Comment
  comments: [Comment!]!
}

type Comment {
  id: ID
  user: User!
  post: Post!
  title: String!
  body: String!
}

type Query {
  user(id: ID!): User
}

type Mutation {
  createUser(email: String!): User
  removeUser(id: ID!): Boolean
  follow(follower: ID!, followee: ID!): Boolean
  unfollow(follower: ID!, followee: ID!): Boolean
  createPost(user: ID!, title: String!, body: String!): Post
  removePost(id: ID!): Boolean
  createComment(user: ID!, post: ID!, title: String!, body: String!): Comment
  removeComment(id: ID!): Boolean
}
```

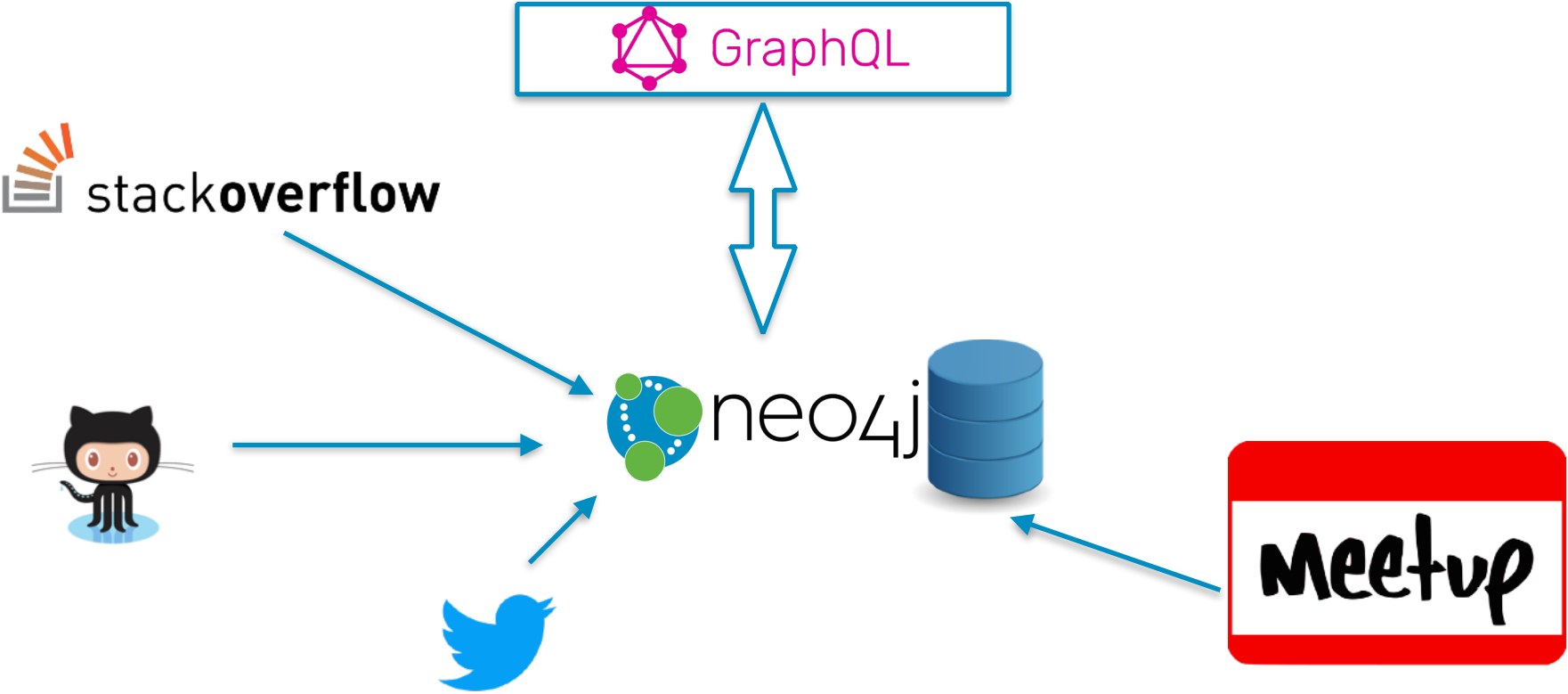
## Shorthand schema definition

- Language agnostic
- Tooling to scaffold server

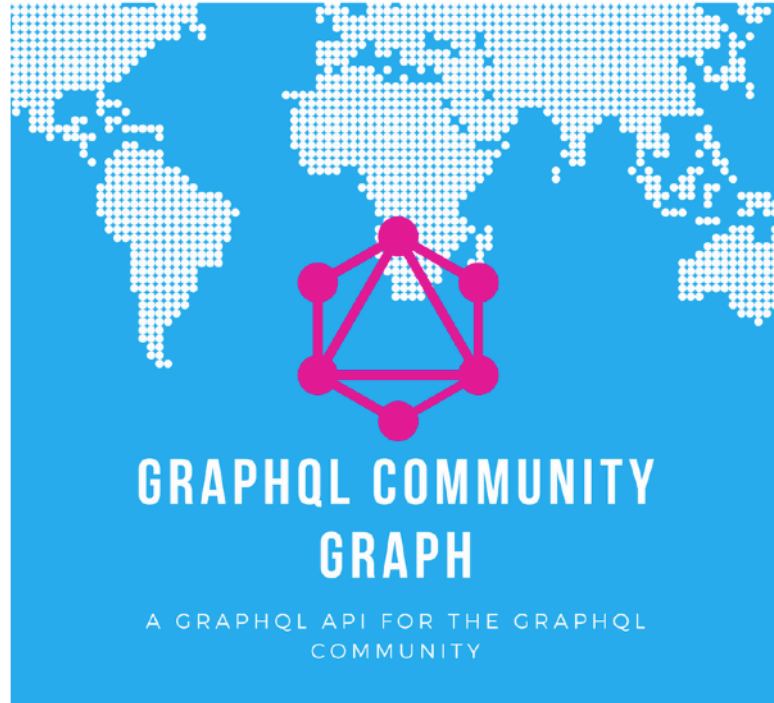


# Demo

# [graphql.community/graph.org](https://graphql.community/graph.org)



[graphql.communitygraph.org](https://graphql.community/graph.org)



POWERED BY  neo4j

```
1  
2
```

🔍 Search Schema...

A GraphQL schema provides a root type each kind of operation.

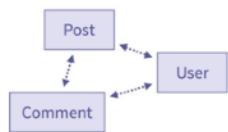
ROOT TYPES

- query: [QueryType](#)
- mutation: [MutationType](#)

# Building A GraphQL Server

# GraphQL First Development

## GraphQL First Development



### 1. Design API schema

Contract between frontend and backend with a shared schema language



### 2. Build UI and backend

Parallelize with mocking, develop component-based UIs with GraphQL containers



### 3. Run in production

Static queries make loading predictable, schema tells you which fields are being used

1. Design API schema
2. Build UI and backend
3. Run in production

- Schema is your friend
- GraphQL Schema is the API spec
  - Allows for simultaneous frontend and backend development
  - Enables introspection
    - Build other tools (graphql)

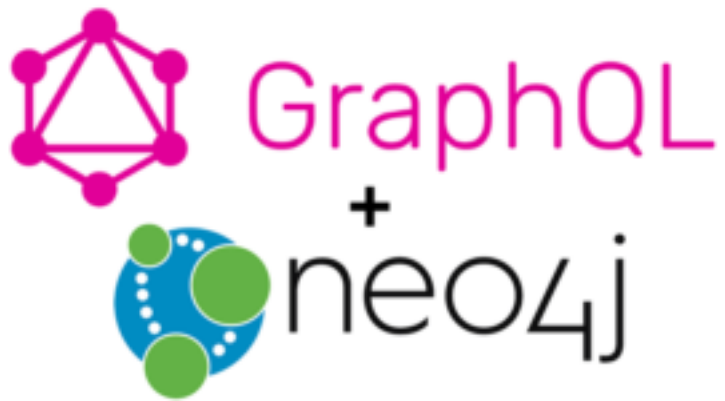
# Building A GraphQL Server

- Define GraphQL schema
- Define resolvers
  - Fetch data for a GraphQL field

```
Query: {  
  movies(_, params) {  
    let session = driver.session();  
    let query = "MATCH (movie:Movie) WHERE movie.title CONTAINS $subString RETURN movie LIMIT $limit;"  
    return session.run(query, params)  
      .then( result => { return result.records.map(record => { return record.get("movie").properties })})  
  },  
}
```







```

22 type Query {
23   movies(subString: String!, limit: Int!): [Movie]
24 }
25
26
27 schema {
28   query: Query
29 }
30
31
32 // Resolver functions query Neo4j database using Cypher query language
33 const resolvers = {
34   Query: {
35     movies(_, params) {
36       let session = driver.session();
37       let query = "MATCH (movie:Movie) WHERE movie.title CONTAINS $subString RETURN movie LIMIT $limit;";
38       return session.run(query, params)
39         .then( result => { return result.records.map(record => { return record.get("movie").properties });
40     },
41   },
42   Movie: {
43     similar(movie) {
44       let session = driver.session(),
45         params = {movieId: movie.movieId},
46         query = `
47       MATCH (m:Movie) WHERE m.movieId = $movieId
48       MATCH (m)-[:IN_GENRE]->(g:Genre)<-[:IN_GENRE]-(movie:Movie)
49       WITH movie, COUNT(*) AS score
50       RETURN movie ORDER BY score DESC LIMIT 3
51     `;
52       return session.run(query, params)
53         .then( result => { return result.records.map(record => { return record.get("movie").properties });
54     },

```

GraphQL Prettify Save Reset Headers

```

1 # Welcome to GraphQL
2
3 {
4   movies(subString:"Matrix", limit:3) {
5     title
6     movieId
7     imdbRating
8     plot
9     poster
10    similar {
11      title
12    }
13  }
14 }
15

```

---

QUERY VARIABLES

```

1 {
  "data": {
    "movies": [
      {
        "title": "Matrix, The",
        "movieId": "2571",
        "imdbRating": 8.7,
        "plot": "A computer hacker learns from mysterious
rebels about the true nature of his reality and his role in
the war against its controllers.",
        "poster": "http://ia.media-
imdb.com/images/M/MV5BMTkxNDYxOTA4M15BM15BanBkFtZTgwNTk0ZkZj
MTE@._V1_SX300.jpg",
        "similar": [
          {
            "title": "X-Men Origins: Wolverine"
          },
          {
            "title": "Who Am I? (Wo shi shei)"
          },
          {
            "title": "X-Files: Fight the Future, The"
          }
        ]
      },
      {
        "title": "Matrix Reloaded, The",
        "movieId": "6365",
        "imdbRating": 7.2,
        "plot": "Neo and the rebel leaders estimate that they
have 72 hours until 250,000 probes discover Zion and destroy

```

<https://launchpad.graphql.com/3wzp7qnjv>

```

1 // Movie recommendations with Neo4j and GraphQL!
2 // This serves a simple GraphQL endpoint backed by Neo4j
3 // See http://neo4j.com/sandbox-v2 and select "Recommendations" instance to see the dataset
4
5 // import graphql-tools and the JavaScript driver for Neo4j
6 import {v1 as neo4j} from 'neo4j-driver';
7 import { makeExecutableSchema } from 'graphql-tools';
8
9
10 // Simple Movie schema
11 const typeDefs = `
12 type Movie {
13   movieId: String!
14   title: String
15   year: Int
16   plot: String
17   poster: String
18   imdbRating: Float
19   genres: [String]
20   similar: [Movie]
21 }
22
23 type Query {
24   movies(subString: String!, limit: Int!): [Movie]
25 }
26
27 schema {
28   query: Query
29 }
30 `;
31
32 // Resolver functions query Neo4j database using Cypher query language
33 const resolvers = {
34   Query: {
35     movies(_, params) {
36       let session = driver.session();
37       let query = "MATCH (movie:Movie) WHERE movie.title CONTAINS $subString RETURN movie LIMIT $limit";
38       return session.run(query, params)
39         .then( result => { return result.records.map(record => { return record.get("movie").properties; });
40     },
41   },
42   Movie: {
43     similar(movie) {
44       let session = driver.session(),
45         params = {movieId: movie.movieId},
46         query = `
47         MATCH (m:Movie) WHERE m.movieId = $movieId
48         MATCH (m)-[:IN_GENRE]->(g:Genre)-[:IN_GENRE]-(movie:Movie)

```

```

1 {
2   movies(subString: "Matrix", limit: 3) {
3     title
4     genres
5     plot
6     year
7   }
8 }
9

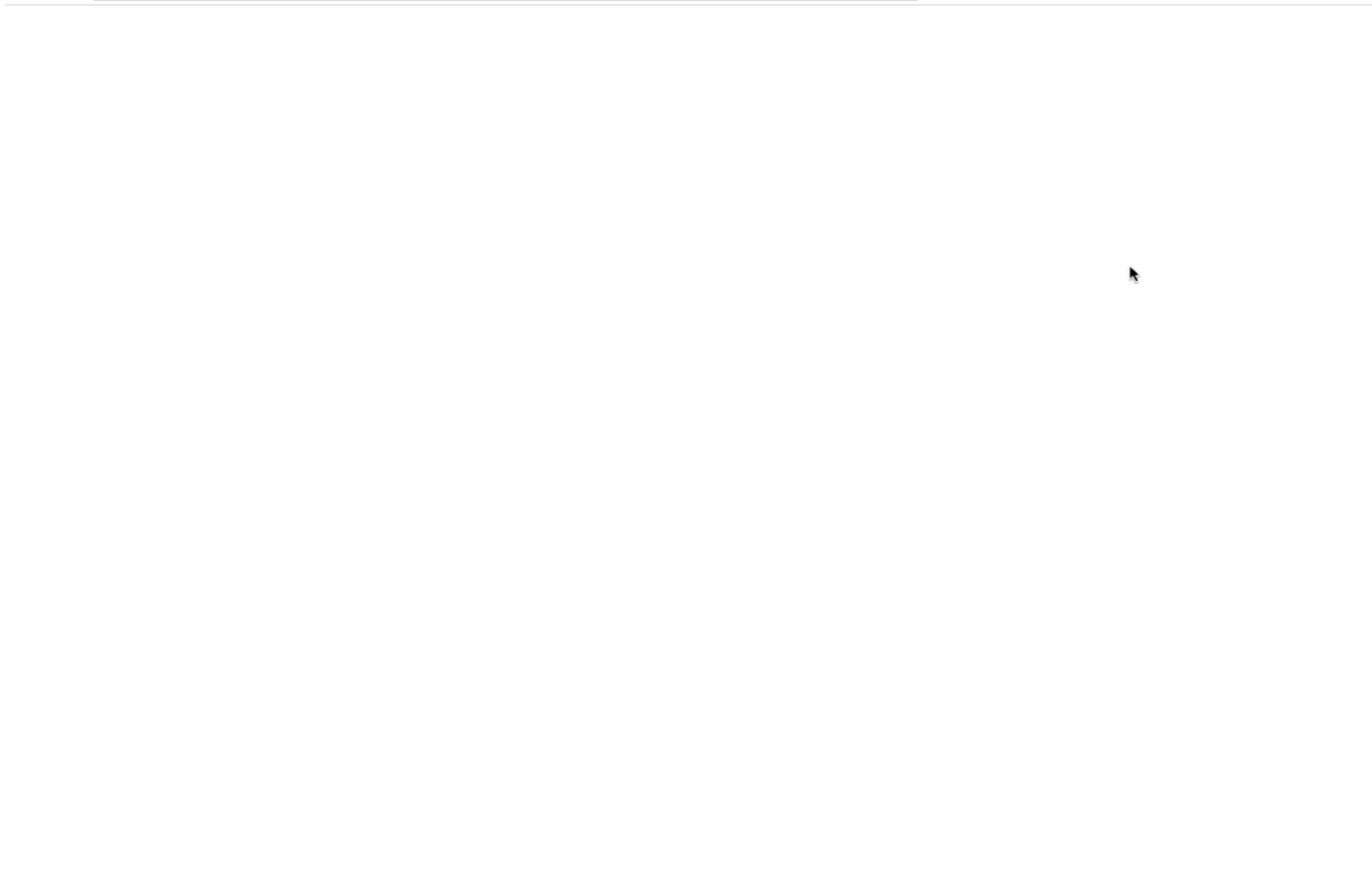
```

QUERY VARIABLES

```

1

```

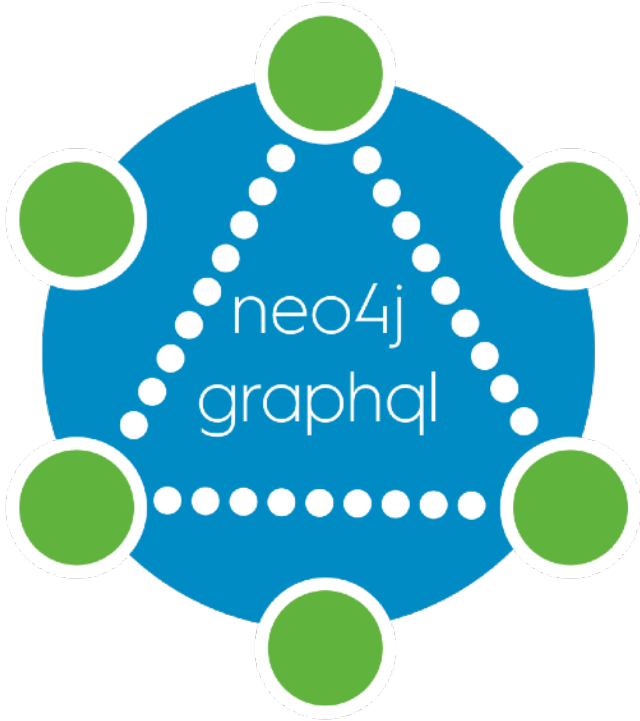


# Developer Story



- GraphQL is **not** a query language for graph databases
- Your application data is a graph, GraphQL enables you to define and query it as such (by extracting trees from the graph)
  - Implies **mapping / translation layer**
- GraphQL was designed to be bolted onto an **existing data layer** (API, database, ORM)
- **Write your own database queries, ORM, or API calls yourself** for each GraphQL field
  - One (or more) database query for each field

# Can we build a better integration?



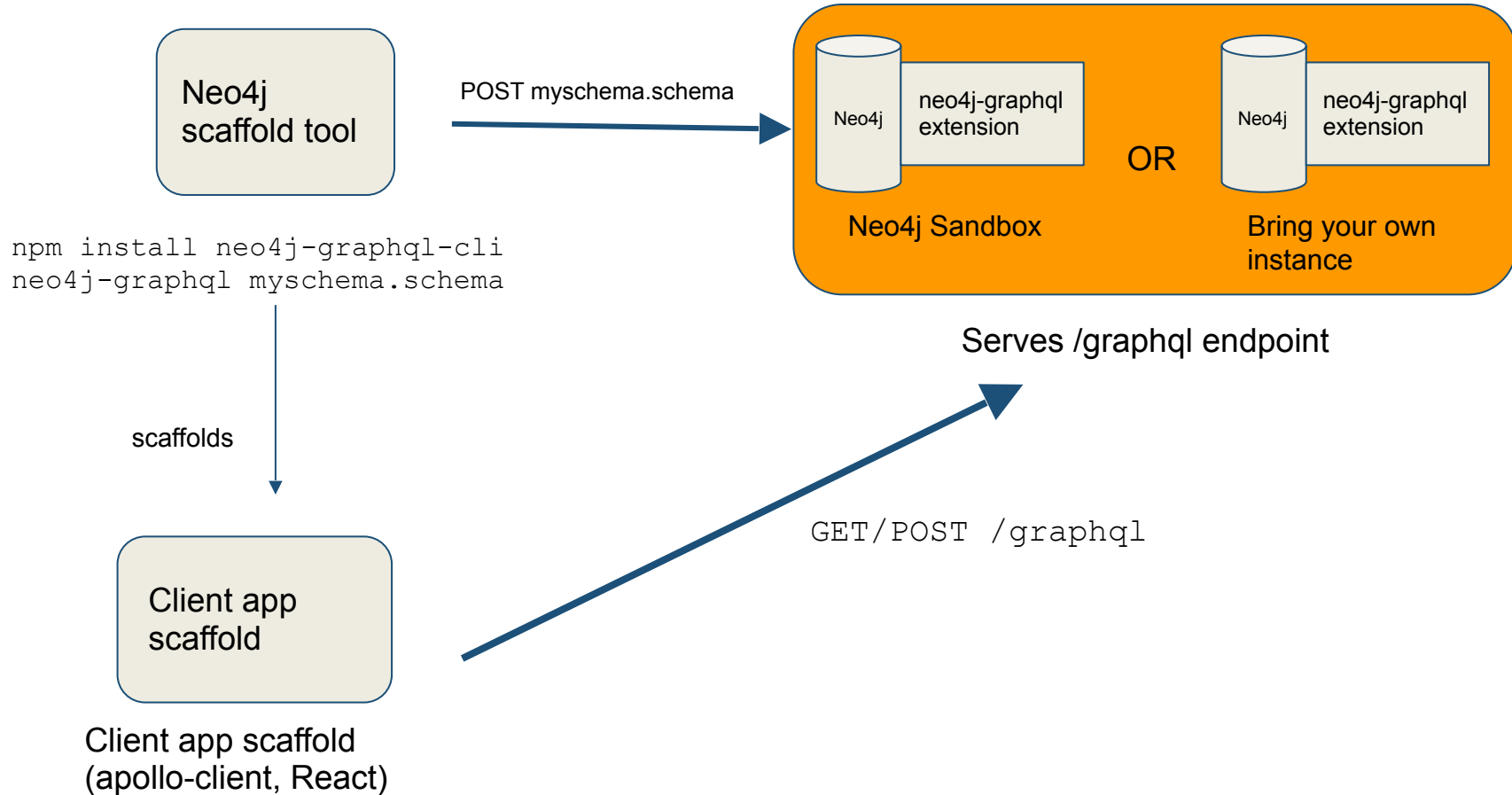
# Developer Story - A GraphQL-Neo4j integration



- If you are assuming **your application data is a graph**, why not model / store it as a graph?
- GraphQL + Neo4j **removes mapping / translation**
  - Why write SQL and GraphQL?
  - We translate GraphQL to Cypher
- **Performance** benefit of
  - storing your application data graph as a graph
  - sending a single query to the database
- **Cypher** with GraphQL takes the power of querying your application data to the next level
  - Optionally embed Cypher within GraphQL fields to move beyond just pulling trees out of your application data graph...



# neo4j-graphql





# Your Application Data Is A Graph

It's Graphs All the Way Down \*

With GraphQL, you model your business domain as a graph

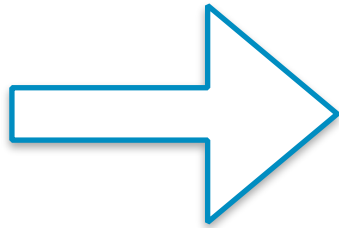
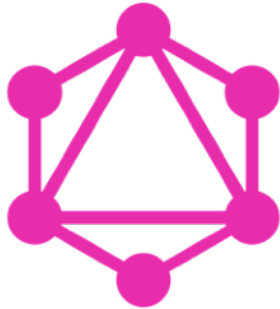
<http://graphql.org/learn/thinking-in-graphs/>

- Storing data in a graph database removes mapping / translation layer found in GraphQL resolvers



# Performance

- By translating GraphQL queries to a single Cypher query removes performance hit of N+1 database queries (1 per field)



openCypher



# Cypher

- Expose Cypher through GraphQL schema directives

```
type User {  
  name: ID!  
  address: String  
  seen: [Movie] @relation(name: "RATED")  
  recommended(first:Int = 5): [Movie] @cypher(statement:"WITH $this as u  
    MATCH (u)-->(:Movie)<--(:User)-->(reco:Movie) WHERE NOT (u)-[:RATED]->(reco)  
    RETURN reco, count(*) as score ORDER BY score DESC LIMIT $first")  
}
```





find packages



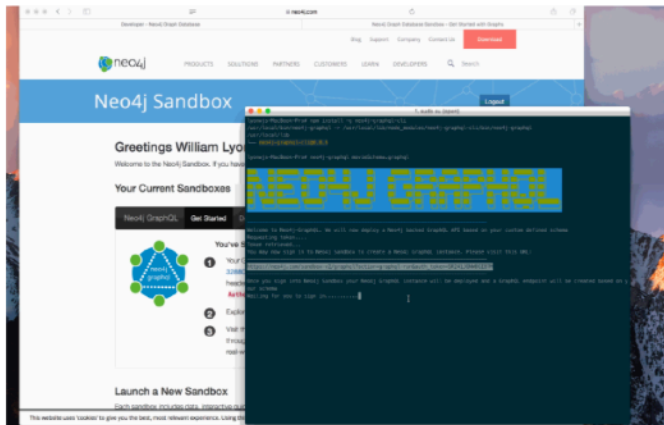
Greetings, neo4j-contrib



## ★ neo4j-graphql-cli public

Deploy Neo4j backed GraphQL APIs based on your custom GraphQL schema.

**This is a very early project, under active development. Use for prototyping and demo projects only**



### What does it do?

`neo4j-graphql-cli` allows you to deploy a Neo4j GraphQL instance on Neo4j Sandbox. This Neo4j GraphQL instance will serve a GraphQL endpoint based on a user-defined GraphQL schema.

### Unleash awesomeness

Private packages, team management tools, and powerful integrations. [Get started with npm Orgs](#)

📦 `npm i neo4j-graphql-cli`

[how? learn more](#)

🌐 neo4j-contrib published 3 weeks ago

0.0.9 is the latest of 9 releases

[github.com/neo4j-graphql/neo4j-graphql-cli](https://github.com/neo4j-graphql/neo4j-graphql-cli)

MIT

Collaborators [edit](#)



### Stats

0 downloads in the last day

0 downloads in the last week

442 downloads in the last month

<https://www.npmjs.com/package/neo4j-graphql-cli>



PRODUCTS

SOLUTIONS

PARTNERS

CUSTOMERS

LEARN

DEVELOPERS

Search

Download

# (Neo4j) – [:LOVES] – (Developers)

World's leading graph database, with native graph storage and processing.

Property graph model and Cypher query language makes it easy to understand

**Fast. Natural. Fun.**

Online Sandbox

Download

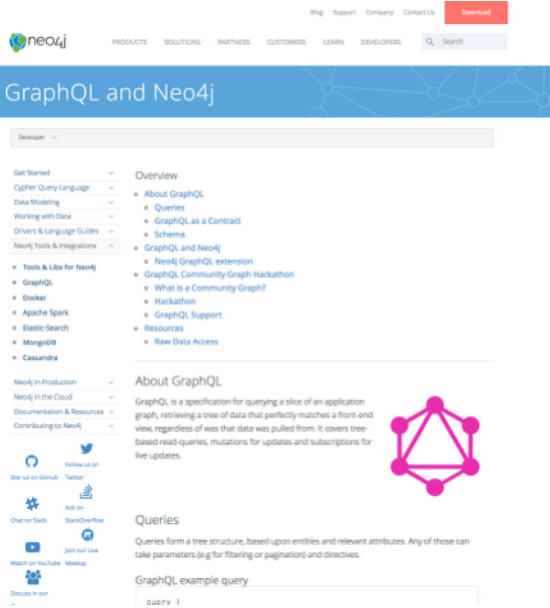
Test-Drive

Social

This website uses 'cookies' to give you the best, most relevant experience. Using the

```
1. sudo su (zsh)
```

```
npm install neo4j-graphql-cli
neo4j-graphql
```



The screenshot shows the Neo4j GraphQL Developer page. At the top, there is a navigation bar with links for 'Blog', 'Support', 'Company', 'Contact Us', and a red 'Download' button. Below this is a search bar and a blue header with the text 'GraphQL and Neo4j'. A 'Developer' dropdown menu is visible. The main content area is divided into two columns. The left column contains a 'Get Started' section with links to 'Cypher Query Language', 'Data Modeling', 'Working with Data', 'Drivers & Language Guides', and 'Neo4j Tools & Integrations'. Below this is a 'Tools & Libraries for Neo4j' section listing 'GraphQL', 'Docker', 'Apache Spark', 'Elastic Search', 'MongoDB', and 'Cassandra'. The right column features an 'Overview' section with links to 'About GraphQL', 'Queries', 'GraphQL as a Contract', 'Schema', 'GraphQL and Neo4j', 'Neo4j GraphQL extension', 'GraphQL Community Graph Hackathon', 'What is a Community Graph?', 'Hackathon', 'GraphQL Support', 'Resources', and 'Raw Data Access'. Below the overview is an 'About GraphQL' section with a description and a pink graph icon. The 'Queries' section explains that queries form a tree structure and provides an example query input field.

<https://neo4j.com/developer/graphql/>

(you) - [ :HAVE ] -> ( ? )  
( ? ) <- [ :ANSWERS ] - (will)



