



Графовая СУБД Neo4j

Автор: Кутуев Владимир Александрович,

Санкт-Петербургский государственный университет
Кафедра системного программирования

20 ноября 2021г.

- **Neo4j** — одна из наиболее популярных графовых СУБД
- Официальный сайт — <https://neo4j.com/>
- Документация — <https://neo4j.com/docs/>



Где применяется Neo4j

<https://neo4j.com/use-cases/>

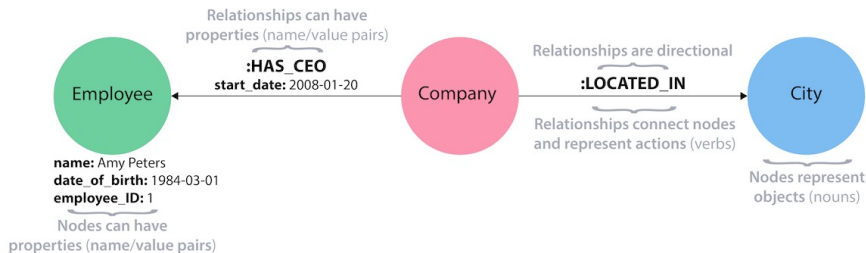
- Fraud Detection & Analytics
- Network and Database Infrastructure Monitoring for IT Operations
- Recommendation Engine & Product Recommendation System
- Master Data Management
- Social Media and Social Network Graphs
- Identity & Access Management
- Retail
- Telecommunications
- Data Privacy, Risk and Compliance
- Artificial Intelligence and Analytics
- Financial Services
- ...

Как поработать с Neo4j

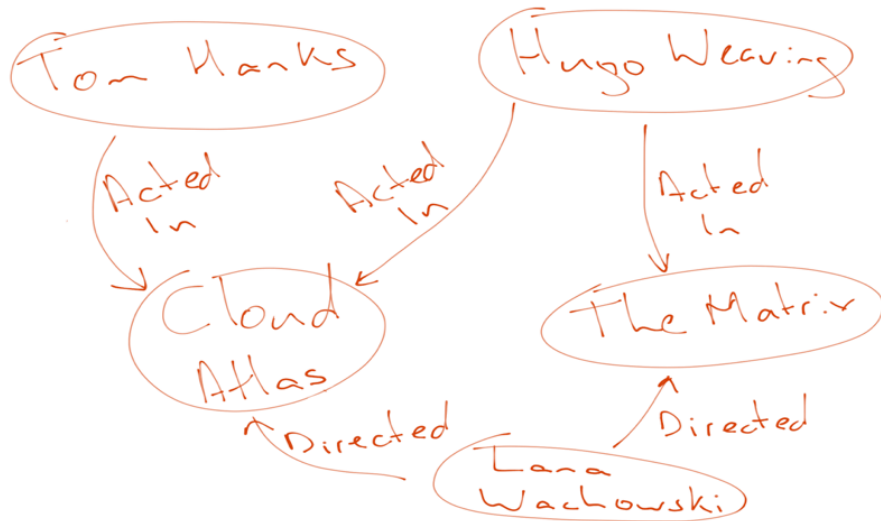
- Скачать и установить: <https://neo4j.com/download-center/#community>
- Docker: https://hub.docker.com/_/neo4j
- Neo4j AuraDB: <https://neo4j.com/cloud/aura/>
- Официальная песочница: <https://neo4j.com/sandbox/>
- Web-консоль: <http://console.neo4j.org/>

Графовая модель данных

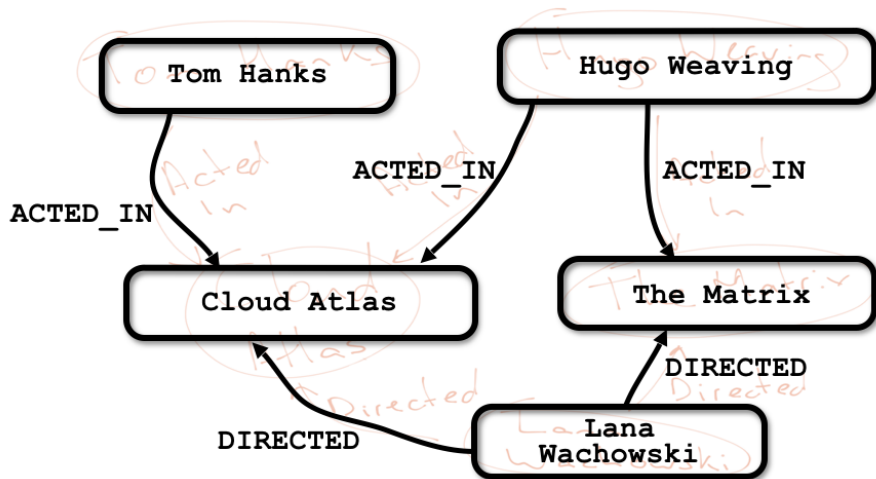
- Вершины (Nodes)
 - ▶ Вершина может содержать несколько меток (Labels)
- Рёбра (Relationships)
 - ▶ Направленные
 - ▶ У каждого ребра есть тип (Type)
- Вершины и рёбра могут хранить свойства (Properties)



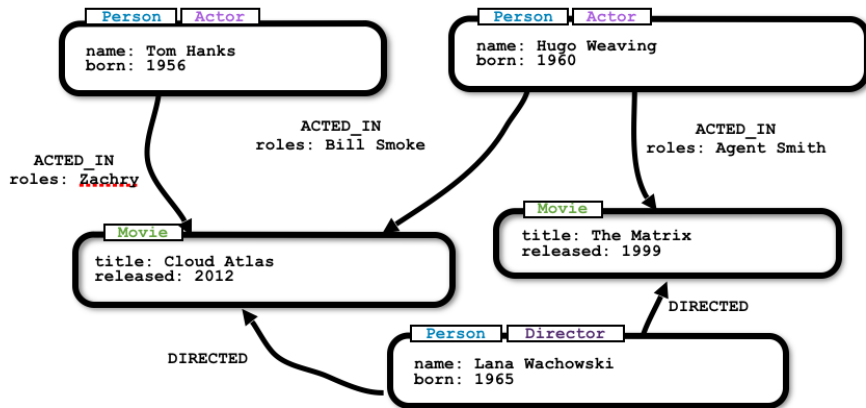
Графовая модель данных



Графовая модель данных



Графовая модель данных



- Cypher Query Language

- ▶ Спецификация: <http://opencypher.org/>
- ▶ Cypher Manual: <https://neo4j.com/docs/cypher-manual/current/>
- ▶ Cypher Refcard: <https://neo4j.com/docs/cypher-refcard/current/>

Пример работы с графовой базой данных


Возьмём демонстрационную базу и напишем к ней запросы (Подробнее можно посмотреть в tutorialе <https://neo4j.com/developer/cypher/guide-cypher-basics/>)


Select a project



☐ For Developers (9) ☐ For Data Scientists (7)

Featured Dataset



Beginner For Developers 

Movies

A guide to common graph query patterns involving connections between movies, actors, and directors.





For Developers 

OpenStreetMap

A graph solution to the shortest-path problem with Cypher involving points of interest and routing of Central Park in New York City.

Libraries Enabled: GraphQL



Beginner For Data Scientists 

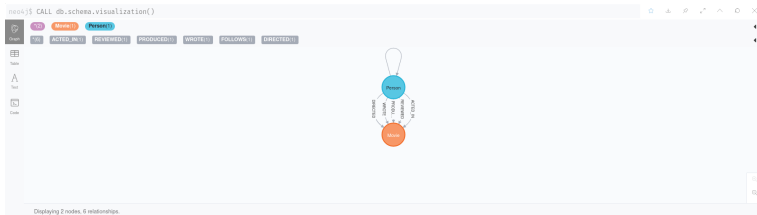
Graph Data Science

Leverage Neo4j Graph Data Science library to explore graph algorithms for analytics and feature engineering.

Libraries Enabled: Graph Data Science

Информация о схеме

```
CALL db.schema.visualization()
```



```
CALL db.schema.nodeTypeProperties()
```

The image shows the Neo4j schema node type properties interface. At the top, there's a command bar with the text 'neo4j\$ CALL db.schema.nodeTypeProperties()'. Below it, a toolbar contains icons for 'Schema', 'Tools', 'Text', and 'Code'. The main area displays a table with 5 columns: 'nodeType', 'nodeLabels', 'propertyName', 'propertyTypes', and 'mandatory'. The table contains 5 rows of data. A status bar at the bottom indicates 'Started streaming 5 records after 8 ms and completed after 28 ms.'

	nodeType	nodeLabels	propertyName	propertyTypes	mandatory
1	"Movie"	["Movie"]	"title"	["String"]	true
2	"Movie"	["Movie"]	"released"	["Long"]	true
3	"Movie"	["Movie"]	"tagline"	["String"]	false
4	"Person"	["Person"]	"name"	["String"]	true
5	"Person"	["Person"]	"born"	["Long"]	false

Информация о схеме

- Эту информацию можно обрабатывать

```
CALL db.schema.relTypeProperties()
YIELD relType, propertyName, propertyTypes
WHERE relType=":REVIEWED"
RETURN relType, propertyName, propertyTypes
```

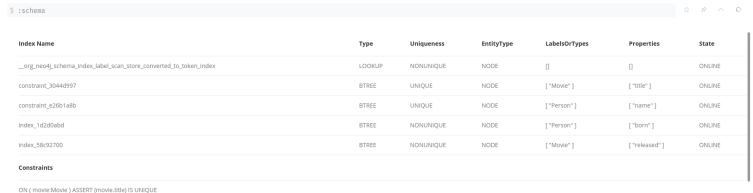


neo4j\$ CALL db.schema.relTypeProperties() YIELD relType, propertyName, propertyTypes WHERE relType=":REVIEWED" RETURN relType, propertyName, propertyTypes

relType	propertyName	propertyTypes
":REVIEWED"	"summary"	["String"]
":REVIEWED"	"rating"	["Long"]

- Индексы и ограничения целостности

:schema



\$:schema

Index Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
...org.neo4j.schema_index_label_scan_store_converted_to_token_index	LOOKUP	NONUNIQUE	NODE	[]	[]	ONLINE
constraint_3044d997	BTREE	UNIQUE	NODE	["Movie"]	["title"]	ONLINE
constraint_e26b1a8b	BTREE	UNIQUE	NODE	["Person"]	["name"]	ONLINE
index_1d2d5ebd	BTREE	NONUNIQUE	NODE	["Person"]	["born"]	ONLINE
index_58c92700	BTREE	NONUNIQUE	NODE	["Movie"]	["released"]	ONLINE

Constraints

ON (:movie:Movie) ASSERT (movie.title) IS UNIQUE

Примеры запросов к графовой базе данных

```
MATCH (tom:Person)
WHERE tom.name = "Tom Hanks"
RETURN tom
```

neo4j\$ MATCH (tom:Person) WHERE tom.name = "Tom Hanks" RETURN tom

*(1) Person(1)

Graph

Table

Text

Code

Tom Hanks

Displaying 1 nodes, 0 relationships.

Примеры запросов к графовой базе данных

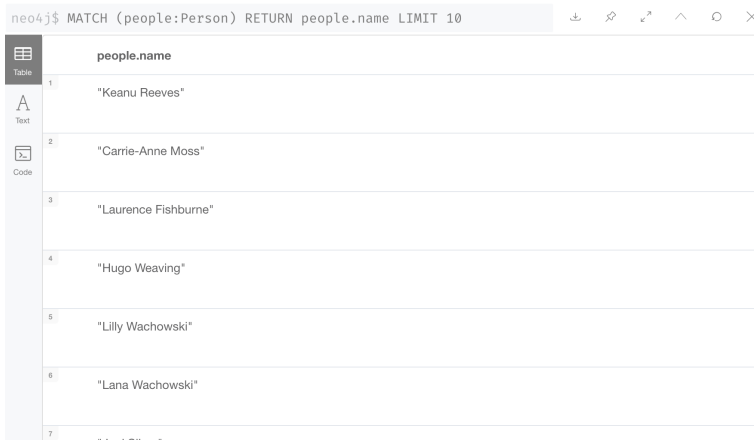
```
MATCH (cloudAtlas:Movie {title: "Cloud Atlas"})  
RETURN cloudAtlas
```



Примеры запросов к графовой базе данных

MATCH (people:Person)

RETURN people.name **LIMIT** 10



The screenshot shows the Neo4j web interface. At the top, a query editor contains the Cypher query: `neo4j$ MATCH (people:Person) RETURN people.name LIMIT 10`. Below the editor, there are icons for switching between Table, Text, and Code views. The 'Table' view is selected, displaying a table with the column header 'people.name'. The table contains 10 rows of results, with the first six rows visible. The results are: 'Keanu Reeves', 'Carrie-Anne Moss', 'Laurence Fishburne', 'Hugo Weaving', 'Lilly Wachowski', and 'Lana Wachowski'. The interface also includes standard window controls (download, copy, paste, undo, redo, close) at the top right of the table area.

	people.name
1	"Keanu Reeves"
2	"Carrie-Anne Moss"
3	"Laurence Fishburne"
4	"Hugo Weaving"
5	"Lilly Wachowski"
6	"Lana Wachowski"
7	

Started streaming 10 records after 1 ms and completed after 7 ms.

Примеры запросов к графовой базе данных

```
MATCH (nineties:Movie)
WHERE nineties.released > 1990
      AND nineties.released < 2000
RETURN nineties.title
```

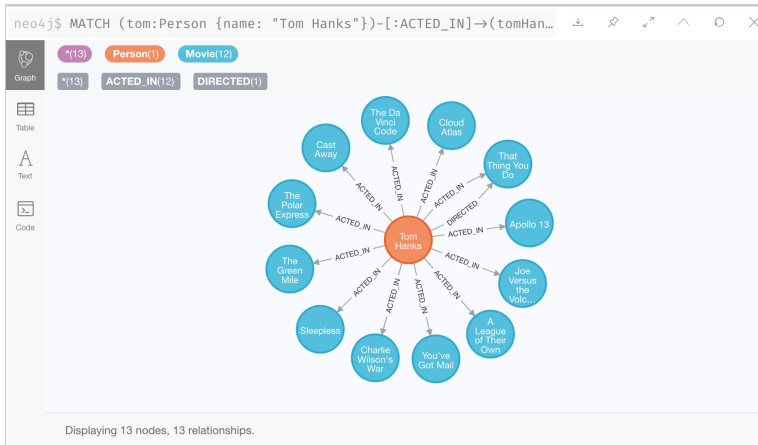
neo4j\$ MATCH (nineties:Movie) WHERE nineties.released > 1990 AND n...

	nineties.title
1	"The Matrix"
2	"The Devil's Advocate"
3	"A Few Good Men"
4	"As Good as It Gets"
5	"What Dreams May Come"
6	"Snow Falling on Cedars"

Примеры запросов к графовой базе данных

```
MATCH (tom:Person {name: "Tom Hanks"})-[:ACTED_IN]->(mv)
```

```
RETURN tom, mv
```



Примеры запросов к графовой базе данных

```
MATCH (cloudAtlas:Movie {title: "Cloud Atlas"})  
      <-[:DIRECTED]-(directors)  
RETURN directors.name
```

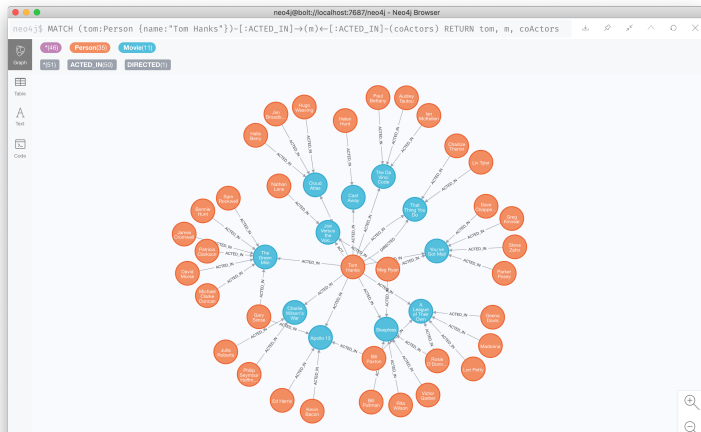
```
neo4j$ MATCH (cloudAtlas:Movie {title: "Cloud Atlas"})<-[:DIRECTED]-(directors) RETURN directors.name
```

	directors.name
1	"Tom Tykwer"
2	"Lana Wachowski"
3	"Lilly Wachowski"

Started streaming 3 records after 1 ms and completed after 4 ms.

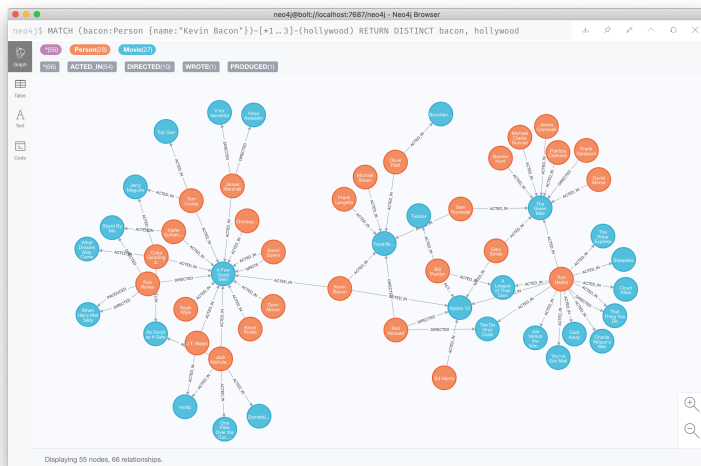
Примеры запросов к графовой базе данных

```
MATCH (tom:Person {name:"Tom Hanks"})  
      -[:ACTED_IN]->(m)<-[:ACTED_IN]-(coActors)  
RETURN tom, m, coActors
```



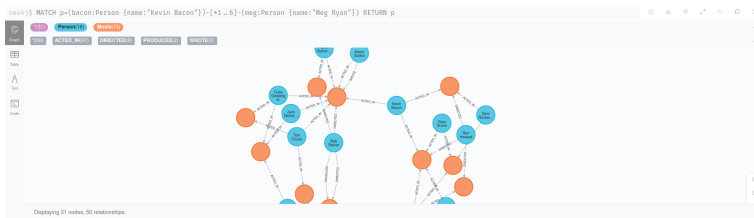
Примеры запросов к графовой базе данных

```
MATCH (bacon:Person {name:"Kevin Bacon"})-[*1..3]-(hollywood)  
RETURN DISTINCT bacon, hollywood
```



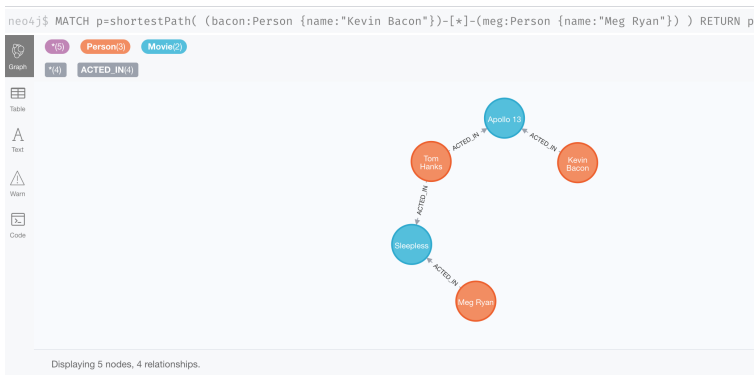
Примеры запросов к графовой базе данных

```
MATCH p=(bacon:Person {name:"Kevin Bacon"})-[*1..6]-(meg:Person {name:"Meg Ryan"})  
RETURN p
```



Примеры запросов к графовой базе данных

```
MATCH p=shortestPath(  
  (bacon:Person {name:"Kevin Bacon"})-[*]-  
    (meg:Person {name:"Meg Ryan"})  
)  
RETURN p
```



Добавление вершин и рёбер

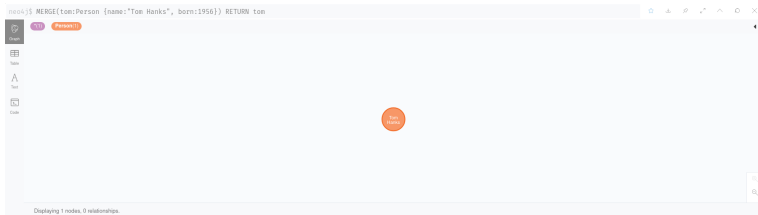
```
CREATE (tom:Person {name:"Tom Hanks", born:1956})  
RETURN tom
```

\$:schema

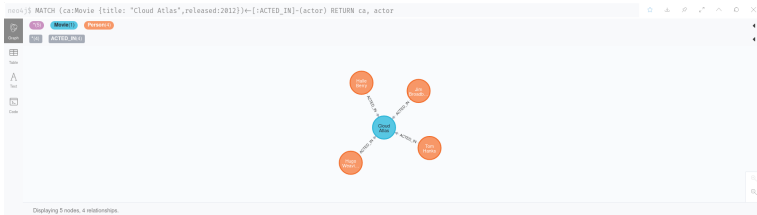
Index Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
__org_neo4j_schema_index_label_scan_store_converted_to_token_index	LOOKUP	NONUNIQUE	NODE	[]	[]	ONLINE
constraint_3044d997	BTREE	UNIQUE	NODE	["Movie"]	["title"]	ONLINE
constraint_e20b1a8b	BTREE	UNIQUE	NODE	["Person"]	["name"]	ONLINE
index_1d2d0ebd	BTREE	NONUNIQUE	NODE	["Person"]	["born"]	ONLINE
index_38c92700	BTREE	NONUNIQUE	NODE	["Movie"]	["released"]	ONLINE
Constraints						
ON (movie:Movie) ASSERT (movie.title) IS UNIQUE						
ON (person:Person) ASSERT (person.name) IS UNIQUE						

Добавление вершин и рёбер

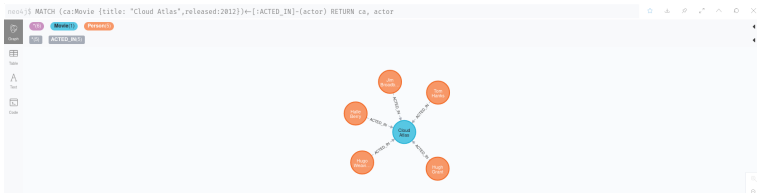
```
MERGE (tom:Person {name:"Tom Hanks", born:1956})  
RETURN tom
```



Добавление вершин и рёбер

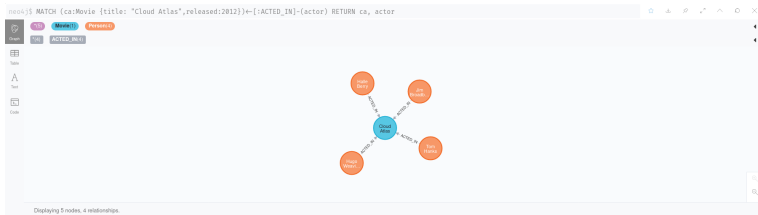


```
MATCH (ca:Movie {title: "Cloud Atlas",released:2012})
CREATE (hugh:Person {name: "Hugh Grant", born:1960}),
(hugh)-[:ACTED_IN {roles:
"Giles Horrox,Hotel Heavy,Lloyd Hooks"}]->(ca)
```



Удаление вершин и рёбер

```
MATCH (hugh:Person {name:"Hugh Grant"})-[r:ACTED_IN]->
      (ca:Movie {title: "Cloud Atlas"})
DELETE r, hugh
```



Что ещё есть в Neo4j

- Индексы
- Ограничения целостности
- Функции
- Пользовательские процедуры и функции
- Пользователи
- Роли
- Привелегии

Интеграция со средствами разработки

- Официальные

- ▶ Java

- ★ Neo4j Java Driver

- ★ Neo4j-OGM

- ★ Embedded

- ▶ .NET

- ▶ JavaScript

- ▶ Python

- ▶ Go

- ▶ HTTP API

- От сообщества

- ▶ Ruby

- ▶ PHP

- ▶ Erlang/Elixir

- ▶ Perl

- ▶ C/C++

- ▶ Clojure

- ▶ R