PUC Minas

Banco de Dados Relacional e Não Relacional

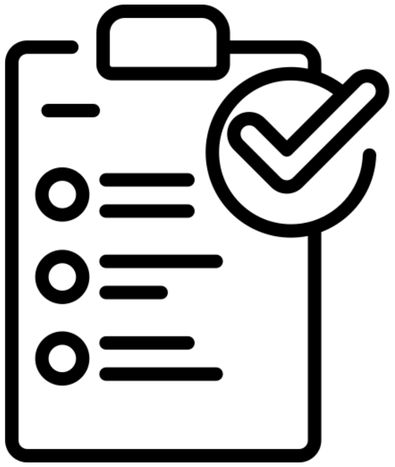
Aula - Banco de Dados Orientado a Grafo Neo4j

Prof. Anderson Theobaldo

PUC Minas

Nesta Aula

- Definição de banco de dados NoSQL tipo grafo
- Características
- Casos de Uso
- Bancos mais populares
- Banco de dados Neo4j
- Demonstração



PUC Minas

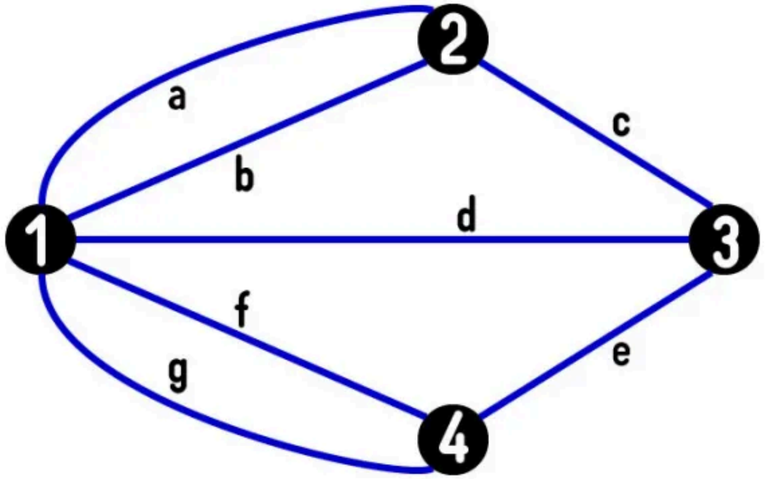
Definição

- Os grafos são poderosas estruturas de dados que modelam relações e conexões entre entidades.
- São frequentemente utilizados para modelar relações em uma variedade de domínios, incluindo redes sociais, sistemas de transporte, circuitos elétricos, entre outros.

PUC Minas

Definição

- Grafo é uma estrutura composta por um conjunto de vértices ou nós, e um conjunto de arestas que conectam esses vértices.
- Os vértices representam entidades distintas, enquanto as arestas representam as relações ou conexões entre essas entidades.
- Dependendo das características das arestas, os grafos podem ser direcionados (com arestas que têm uma direção) ou não direcionados (com arestas sem uma direção específica).



PUC Minas

Características

- Banco de dados NoSQL do tipo grafo (graph database) é baseado na teoria de grafo sendo útil em contextos que sejam ricos em relacionamentos.
- O foco é modelar e armazenar dados sobre relacionamentos.
- Fornece uma fonte rica para algoritmos e aplicativos que precisam manipular dados com essa estrutura.
- Situações em que os relacionamentos se tornam tão importantes quanto os próprios dados são os campos em que esses tipos de bancos se destacam.

PUC Minas

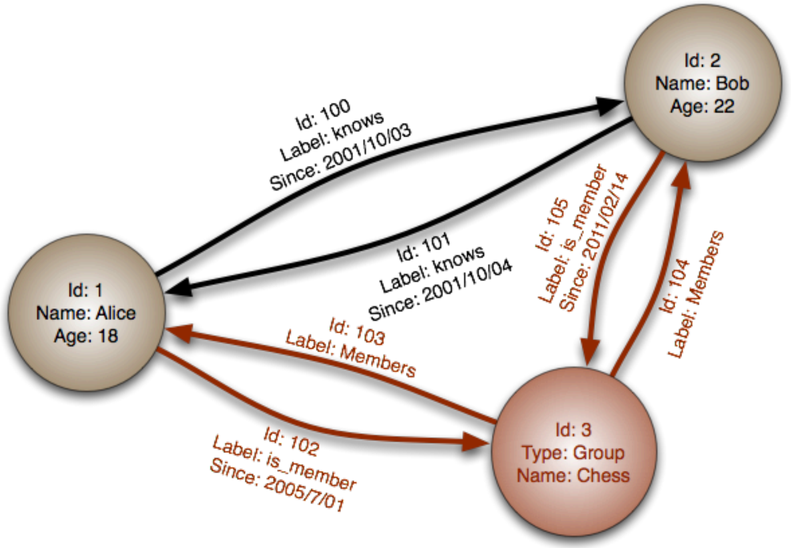
Características

- Não requerem estruturas de dados predefinidas.
- Não suportam a linguagem SQL.
- Otimizados para fornecer velocidade de processamento para dados intensivos em relacionamento.
- Não apresentam escalabilidade tão aprimorada quanto os outros tipos de bancos de dados NoSQL.

PUC Minas

Principais Componentes

- Os componentes primários dos bancos de dados de grafo são nós, arestas e propriedades.
- Um nó corresponde à ideia de uma instância de entidade relacional. O nó é uma instância de algo sobre o qual queremos manter os dados.
- As propriedades são como atributos no qual desejamos armazenar sobre o nó. Todos os nós podem ter propriedades, mas nem todos os nós precisam ter as mesmas propriedades.
- Uma aresta é um relacionamento entre nós e podem estar em uma direção ou ser bidirecionais.



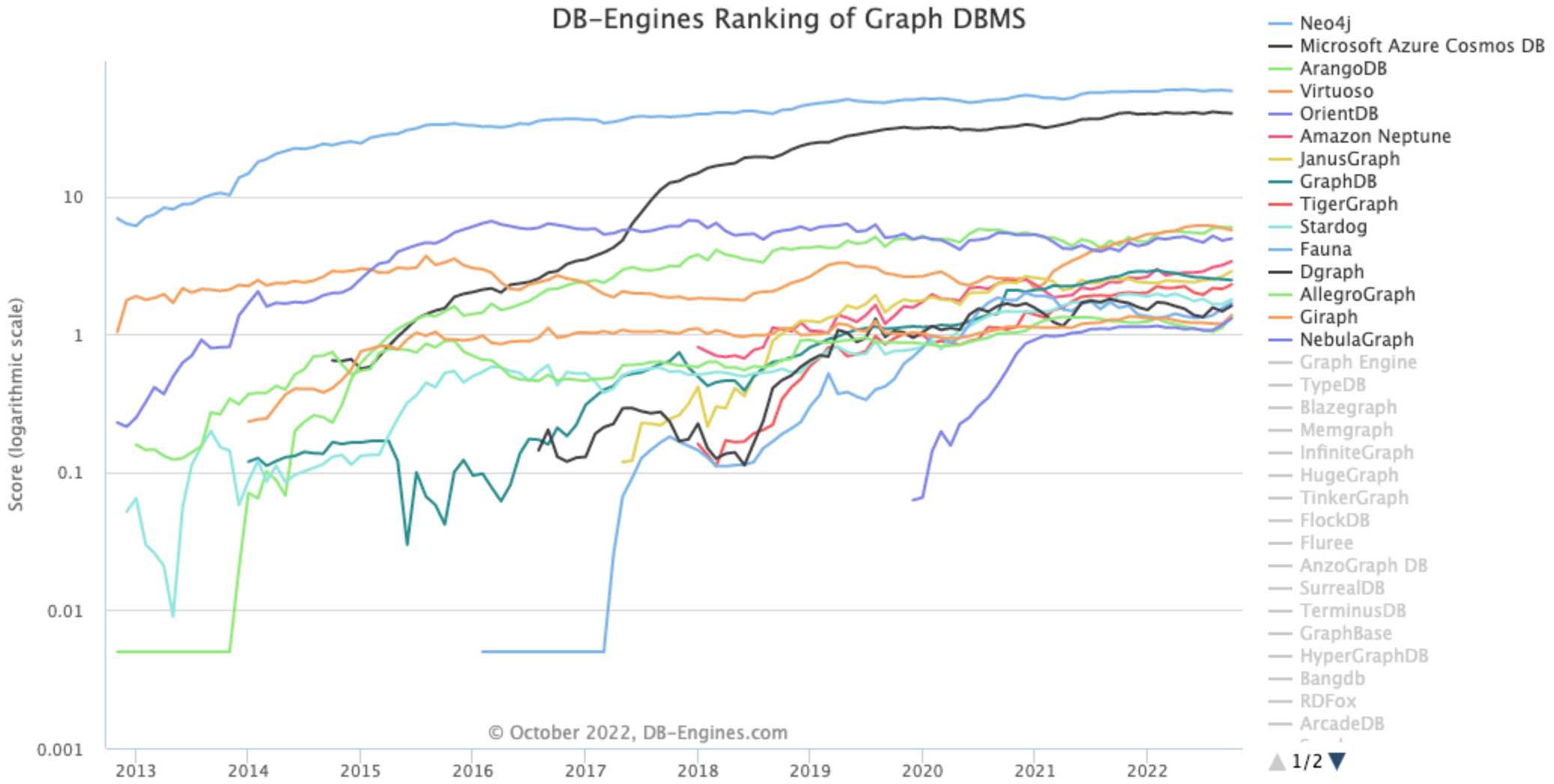
PUC Minas

Casos de Uso

- O interesse pelas bases de dados grafo teve origem na área das redes sociais. Porém, vão além disso, abrangendo questões que dependem do rastreamento de relacionamentos complexos entre objetos:
- Casos de uso comuns:
  - Gerenciamento de recomendações
  - Logística e roteamento
  - Detecção de fraudes
  - Gerenciamento de identidade e acesso

PUC Minas

Bancos Mais Utilizados



PUC Minas

PUC Minas

Neo4J

- Criado em 2007, mantido pela Neo Technologies
- Software livre
- Usa como modelo grafos de propriedades rotulados
- Tem modelo de armazenamento e processamento nativos para grafos
- Usa uma linguagem de consulta chamada Cypher

PUC Minas

Anatomia

Nós

- É algo sobre o que queremos manter os dados.

Label

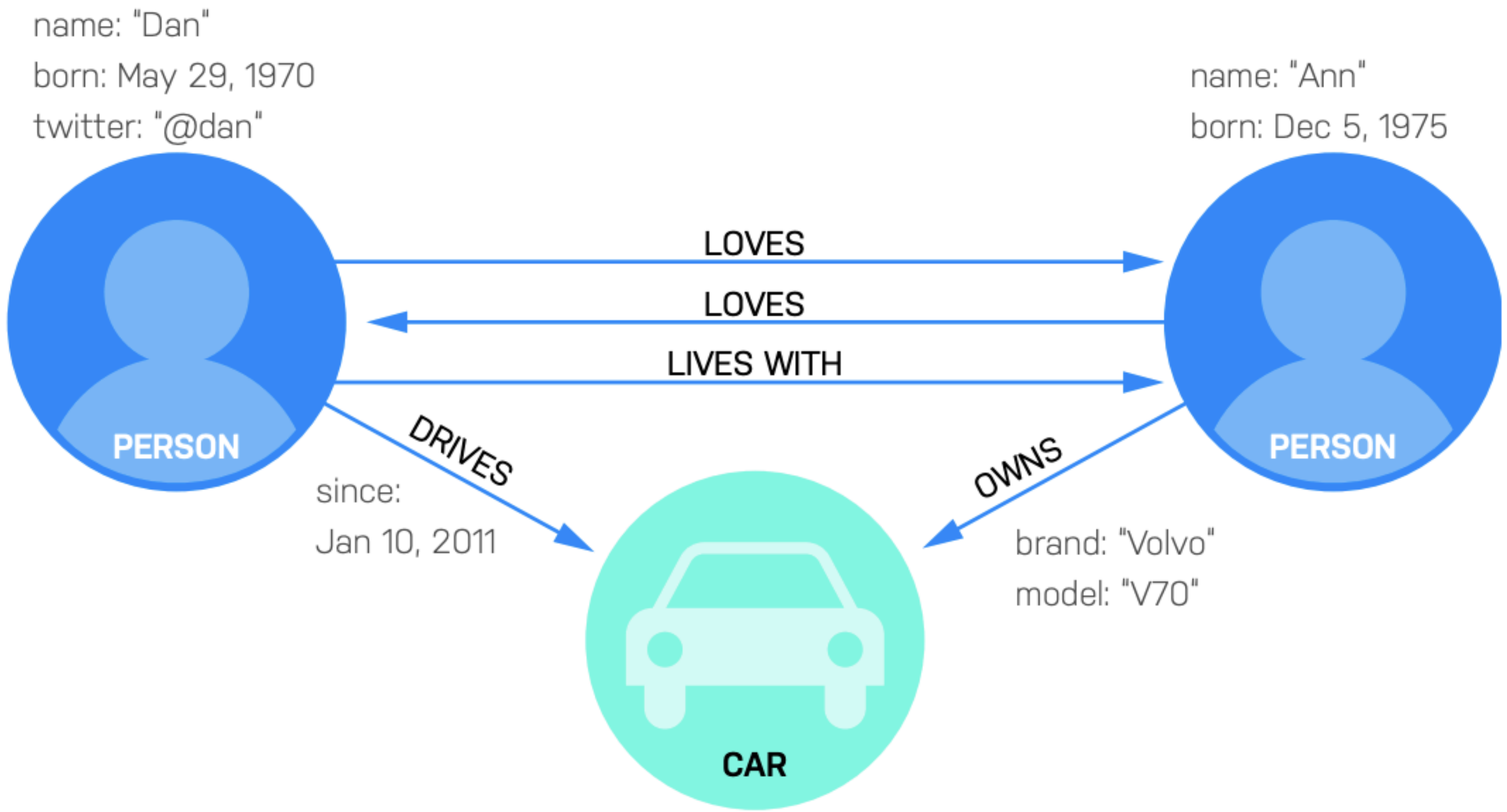
- Agrupam nós do mesmo tipo

Relacionamento

- Relacionam nós e podem estar em uma direção ou ser bidirecionais

Propriedades

- São representados como pares de chave/valor que podem ser associados ao nó ou relacionamento



A simple property graph data model.

PUC Minas

Linguagem Cypher

- É uma linguagem de consulta de grafos declarativa que permite consultas, atualizações e administração expressivas e eficientes do grafo.
- Consultas de banco de dados altamente complicadas podem ser facilmente expressas, permitindo que se concentre em seu domínio.
- É usada para encontrar padrões nos relacionamentos de um grafo.
- Tem comandos inspirados em outras linguagens de consulta conhecidas, como a SQL e a SPARQL
- Sua especificação é aberta.

PUC Minas

Instalação

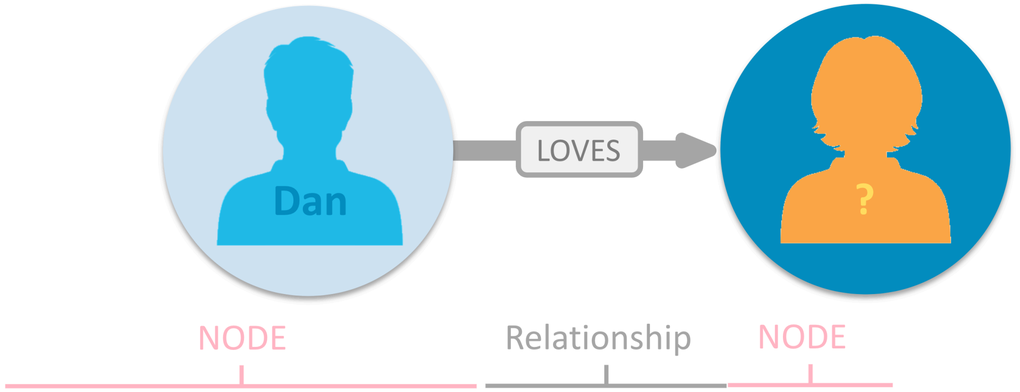
- Instale o Neo4j a partir:
  - https://neo4j.com/download-center/#community
- Neo4j as a service
  - https://neo4j.com/cloud/platform/aura-graph-database/?ref=neo4j-home-hero

Obs: a partir da versão 4.0 do Neo4j, a criação de um novo banco de dados usando o comando CREATE DATABASE só está disponível nas edições Enterprise e Developer. Se estiver usando a edição Community, essa funcionalidade não estará disponível e precisará criar apenas um novo grafo no banco de dados padrão.

PUC Minas

Sintaxe Básica do Cypher

- Nós são denotados por um par de parênteses.
- Uma variável de nó pode ser usada em outros lugares dentro do mesmo comando em que foi definida.
- O rótulo (Label) funciona como a definição de um tipo ou papel para o nó.
  - Possibilita agrupar os nós.
  - Bastante usado como filtro nas consultas.
- Um nó pode ter mais de um rótulo (Label).
- Com exceção dos parênteses, todos os demais elementos de definição de nó são opcionais.




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

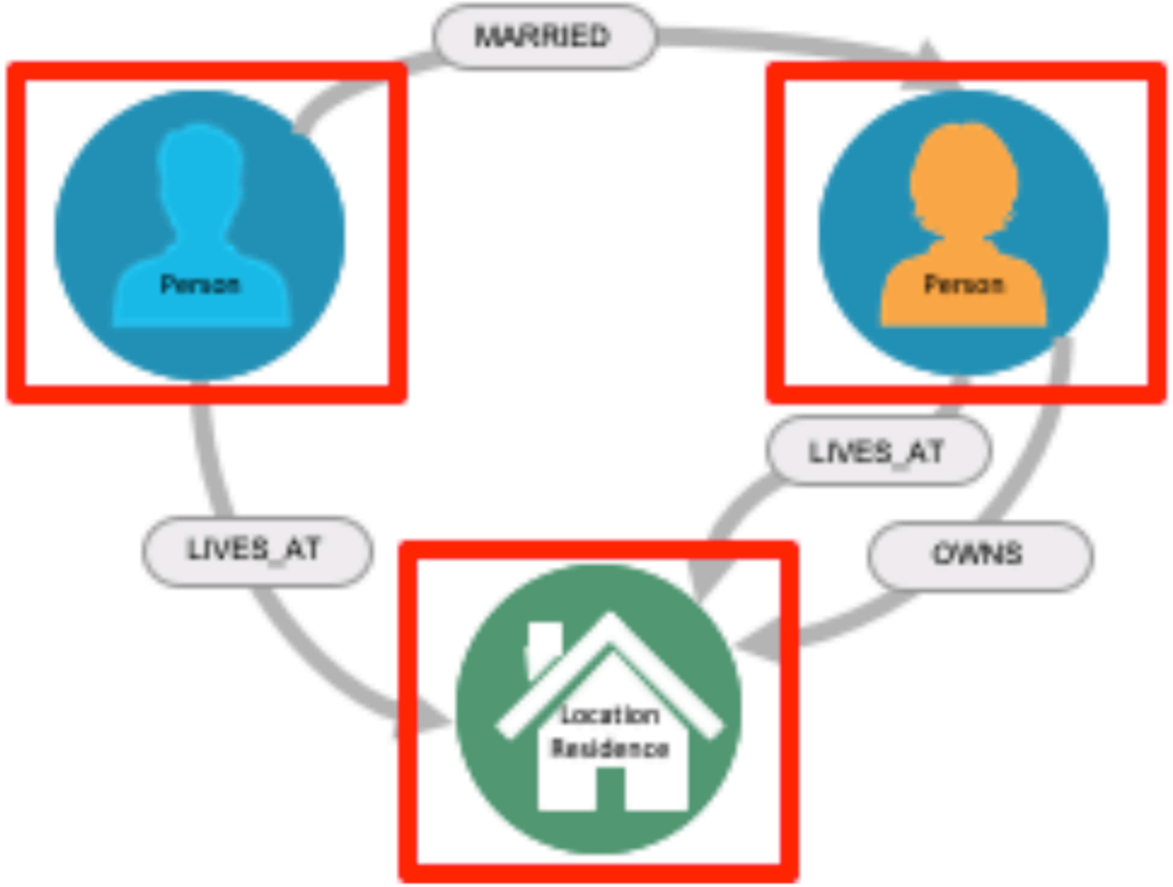
LABEL PROPERTY VARIABLE

PUC Minas

Sintaxe Básica do Cypher

Sintaxe Node

```
()
(p)
```

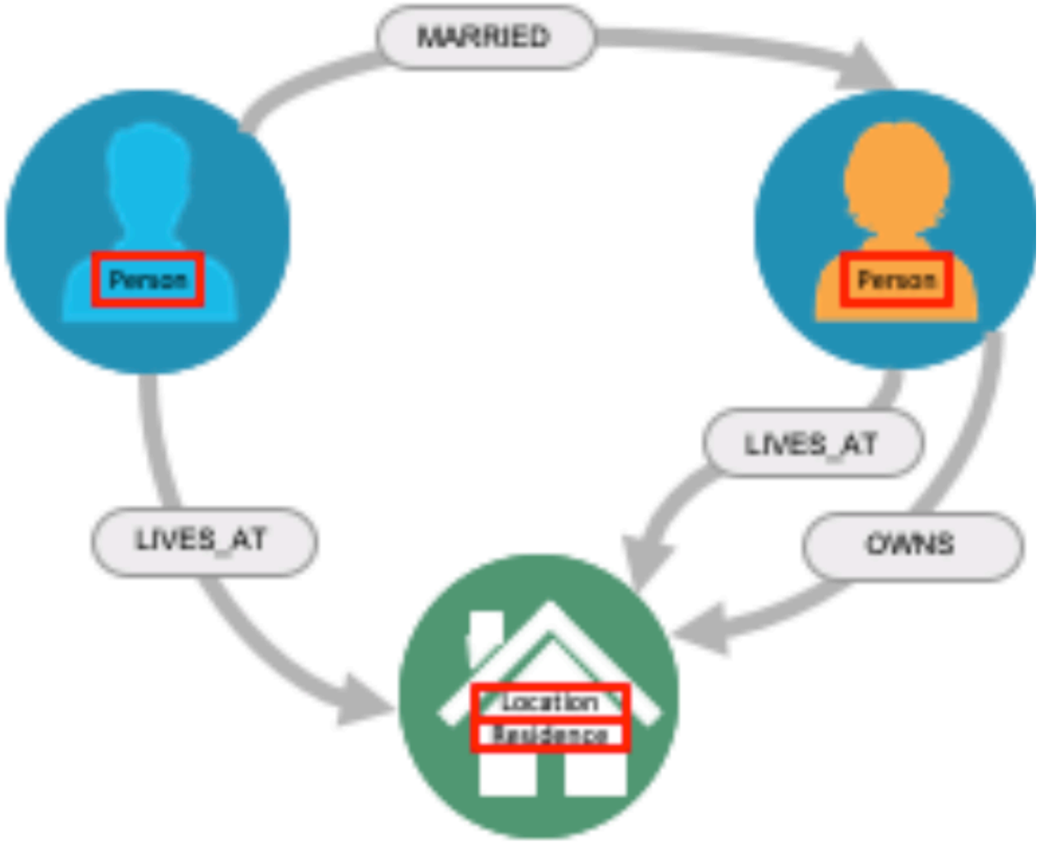


PUC Minas

Sintaxe Básica do Cypher

Sintaxe Label(Rótulo)

```
(:Person)

(p:Person)

(:Location)

(l:Location)

(x:Residence)

(x:Location:Residence)
```



PUC Minas

Sintaxe Básica do Cypher

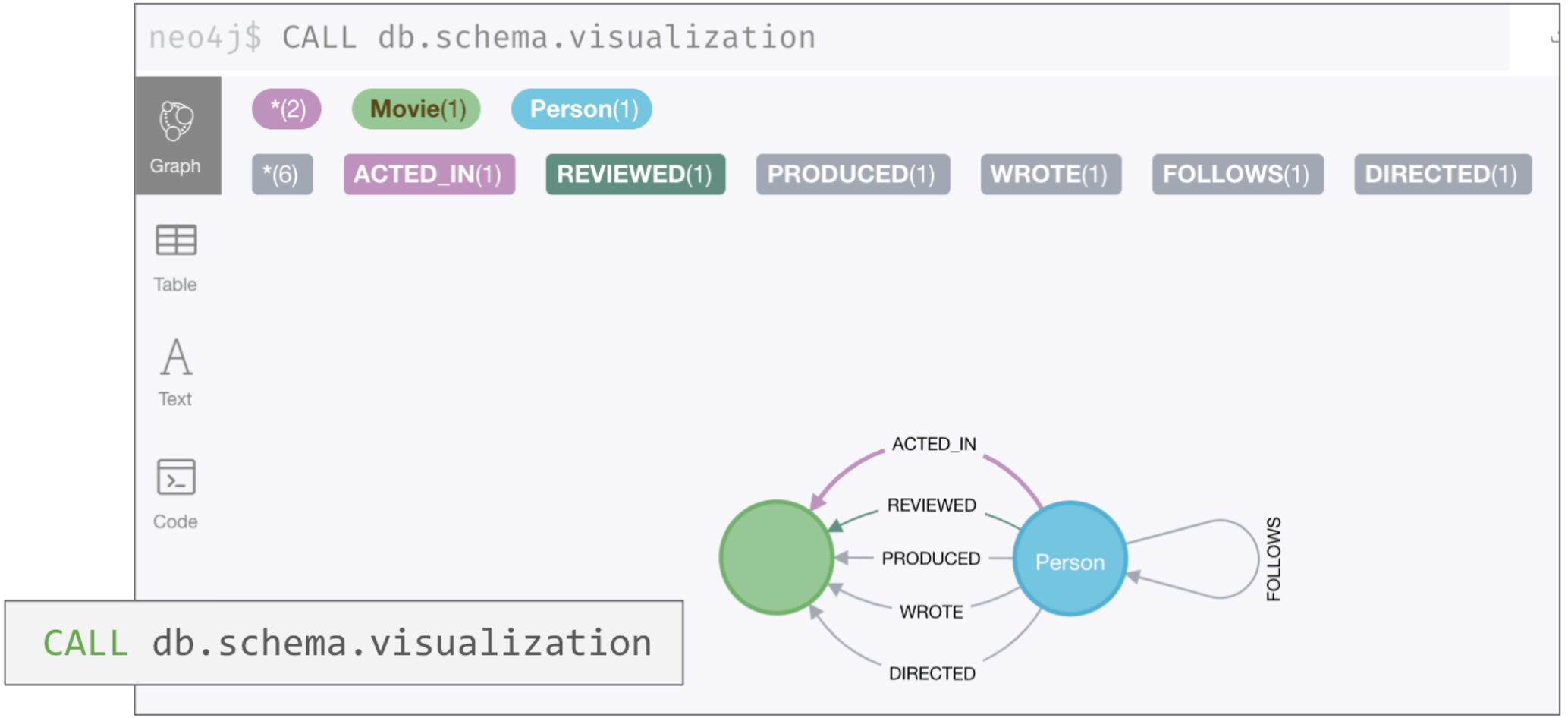
Comentários:

```
// anonymous node not be referenced later in the query
()

(p) // variable p, a reference to a node used later

(:Person)      // anonymous node of type Person
(p:Person)     // p, a reference to a node of type Person
     // p, a reference to a node of types Actor and Director
(p:Actor:Director)
```

PUC Minas

Sintaxe Básica do Cypher

Visualizando o modelo de dados:




```
CALL db.schema.visualization
```

PUC Minas

Sintaxe Básica do Cypher

Match e Return: utilizado para consulta

Sintaxe:

```
MATCH (variável)
RETURN variável
```

```
MATCH (variável: Rótulo)
RETURN variável
```

Recuperando todos os nodes:

```
MATCH (n)
RETURN n
```

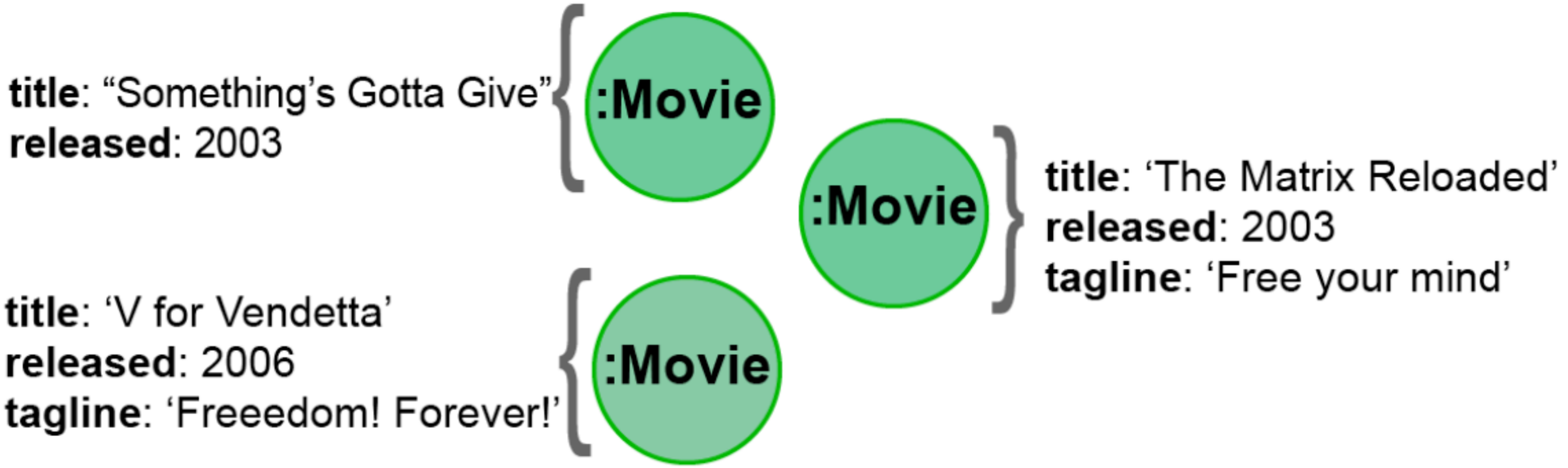
Recuperando todos os nodes Person:

```
MATCH (p:Person)
RETURN p
```

PUC Minas

Sintaxe Básica do Cypher

Sintaxe de propriedade:



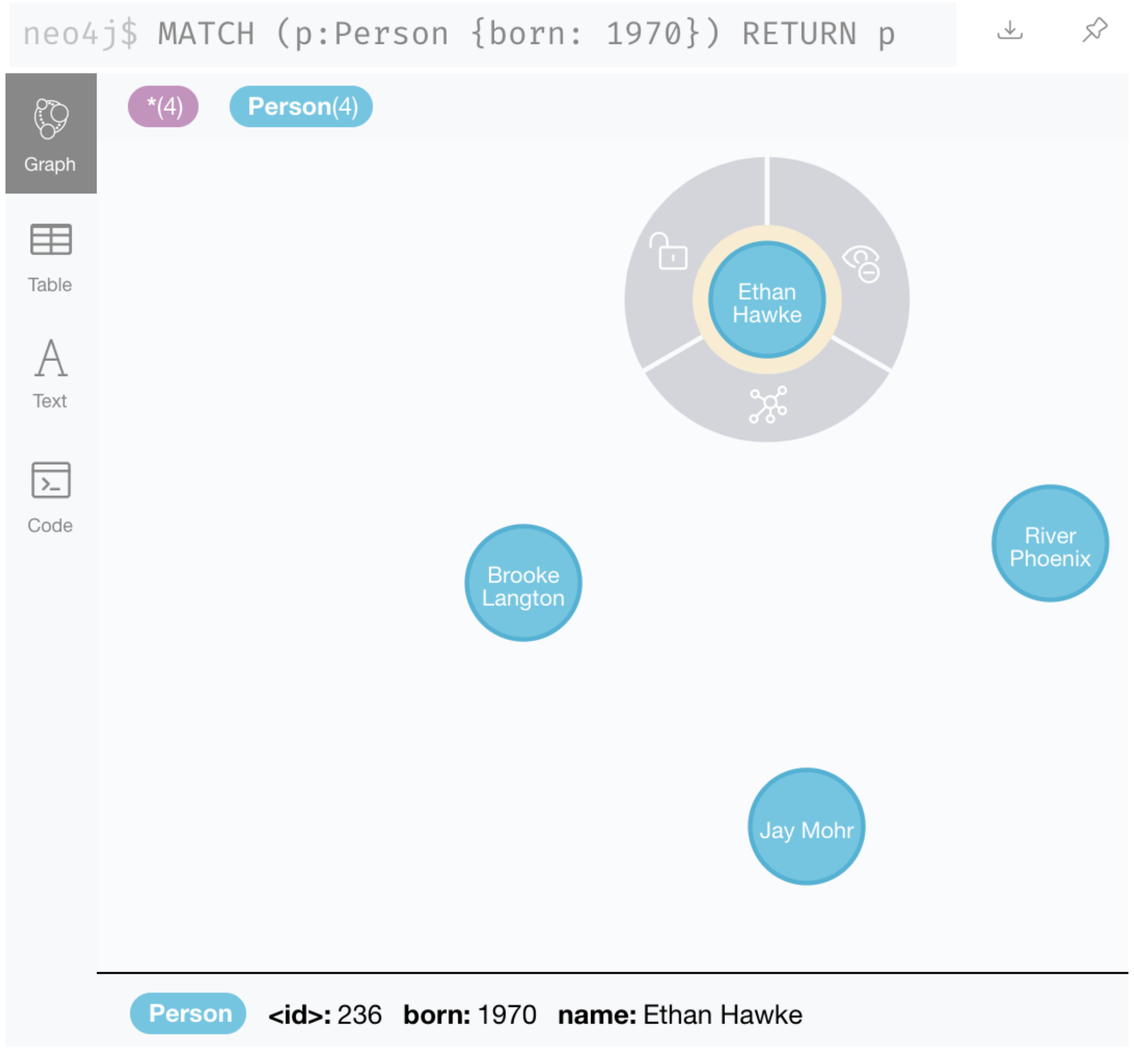
PUC Minas

Sintaxe Básica do Cypher

Filtrando através de propriedade:

```
MATCH (n:Person {born: 1970})
RETURN n
```

Propriedade



Obs: para utilizar várias propriedades no filtro, basta separá-las por vírgula.

PUC Minas

Sintaxe Básica do Cypher

Retornando propriedades específicas

```
MATCH (n:Person {born: 1970})
RETURN n.name, p.born
```

Propriedades retornadas

Table | Text | Code

	p.name	p.born
1	"Lana Wachowski"	1965
2	"Tom Tykwer"	1965
3	"John C. Reilly"	1965

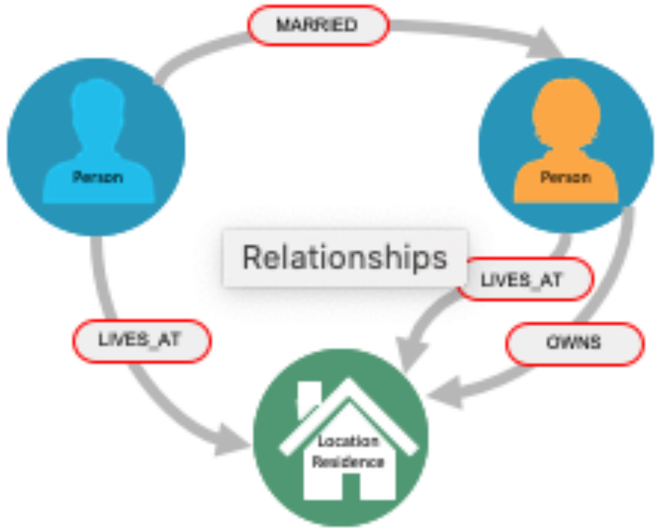
Started streaming 3 records after 1 ms and completed after 3 ms.

PUC Minas

Sintaxe Básica do Cypher

Relacionamentos

- Conexão direcionada entre dois nodes
- Relacionamento tem um tipo
- Relacionamentos podem ter propriedades



PUC Minas

Sintaxe Básica do Cypher

Sintaxe Relacionamento:

```
( )          // retorna um node
( ) - - ()   // retorna nodes que tem algum tipo de relacionamento
( ) - -> ()  // retorna nodes do lado esquerdo que tem algum tipo de relacionamento com nodes do lado direito
( ) <- - ()  // retorna nodes do lado direito que tem algum tipo de relacionamento com nodes do lado esquerdo
```

PUC Minas

Sintaxe Básica do Cypher

Usando relacionamento na consulta:

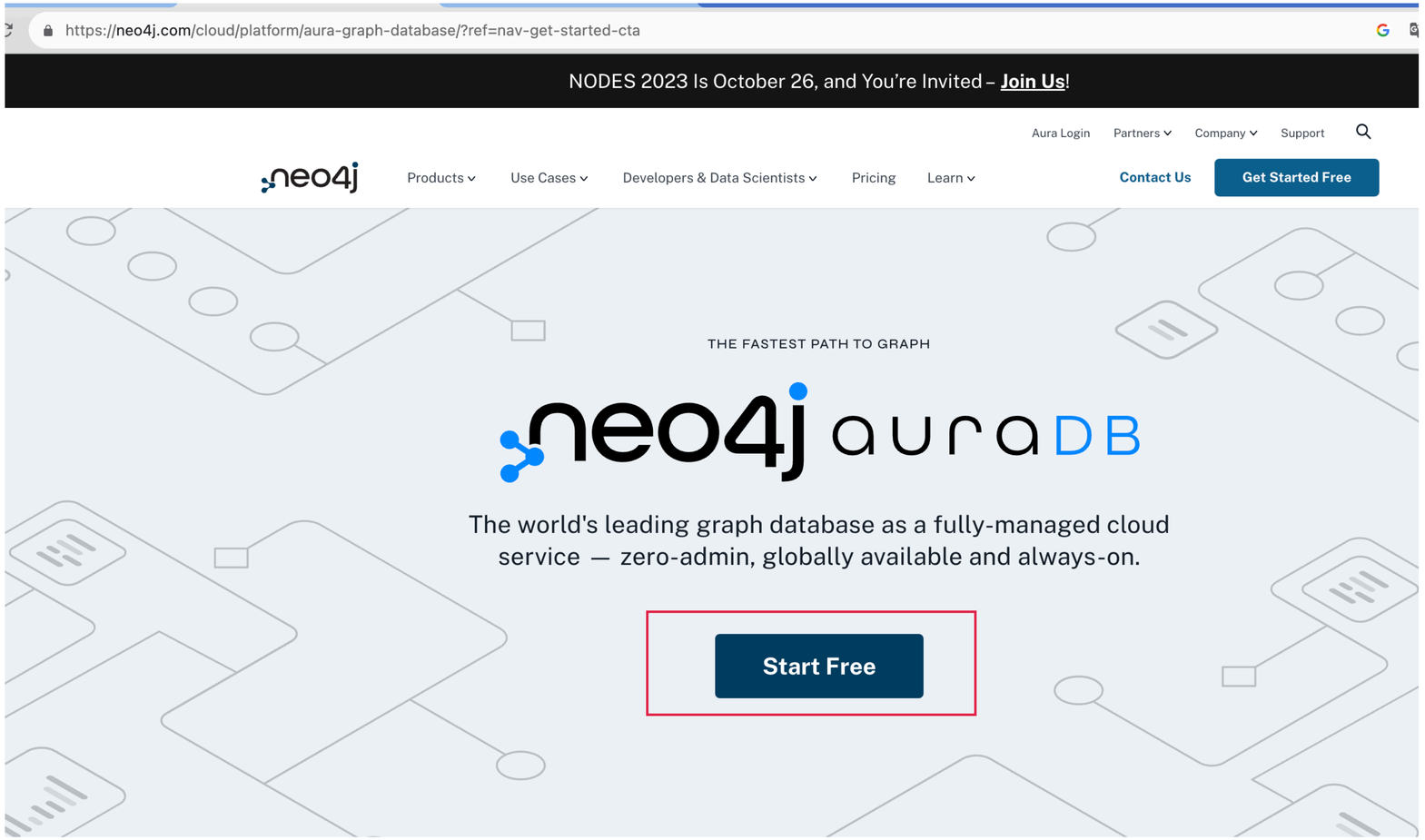
```
MATCH (n:Person) - [rel:ACTED_IN] -> (m:Movie {title: 'The Matrix'})
RETURN p, rel, m
```

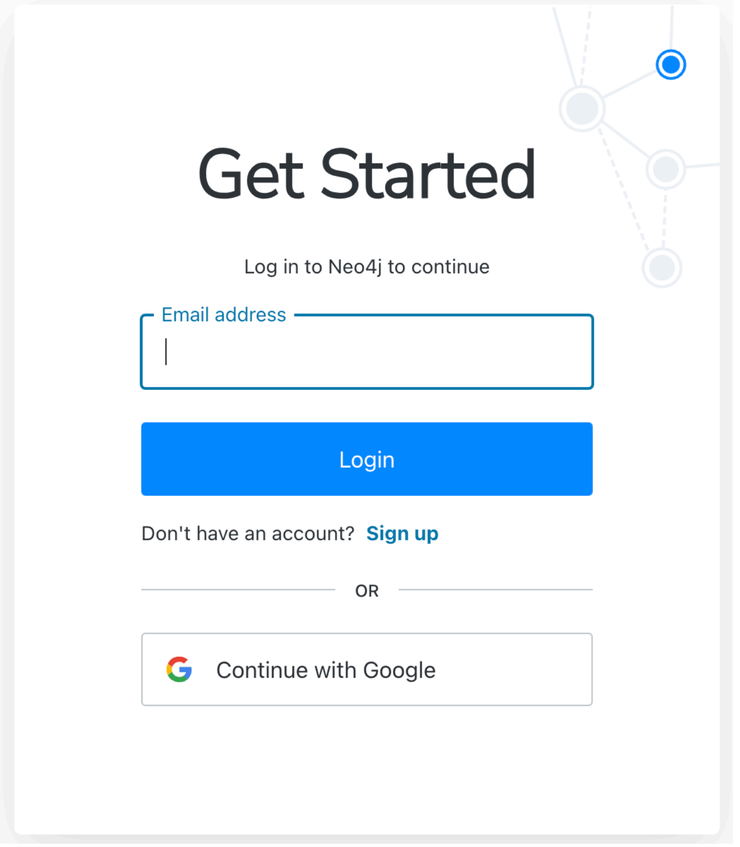
Recupera todas as pessoas (Person) que atuaram no filme 'The Matrix', retornando os nodes e relacionamentos.

PUC Minas

Demonstração

- Criar uma nova conta para Neo4j AuraDB através do link:
  https://neo4j.com/cloud/platform/aura-graph-database/?ref=nav-get-started-cta

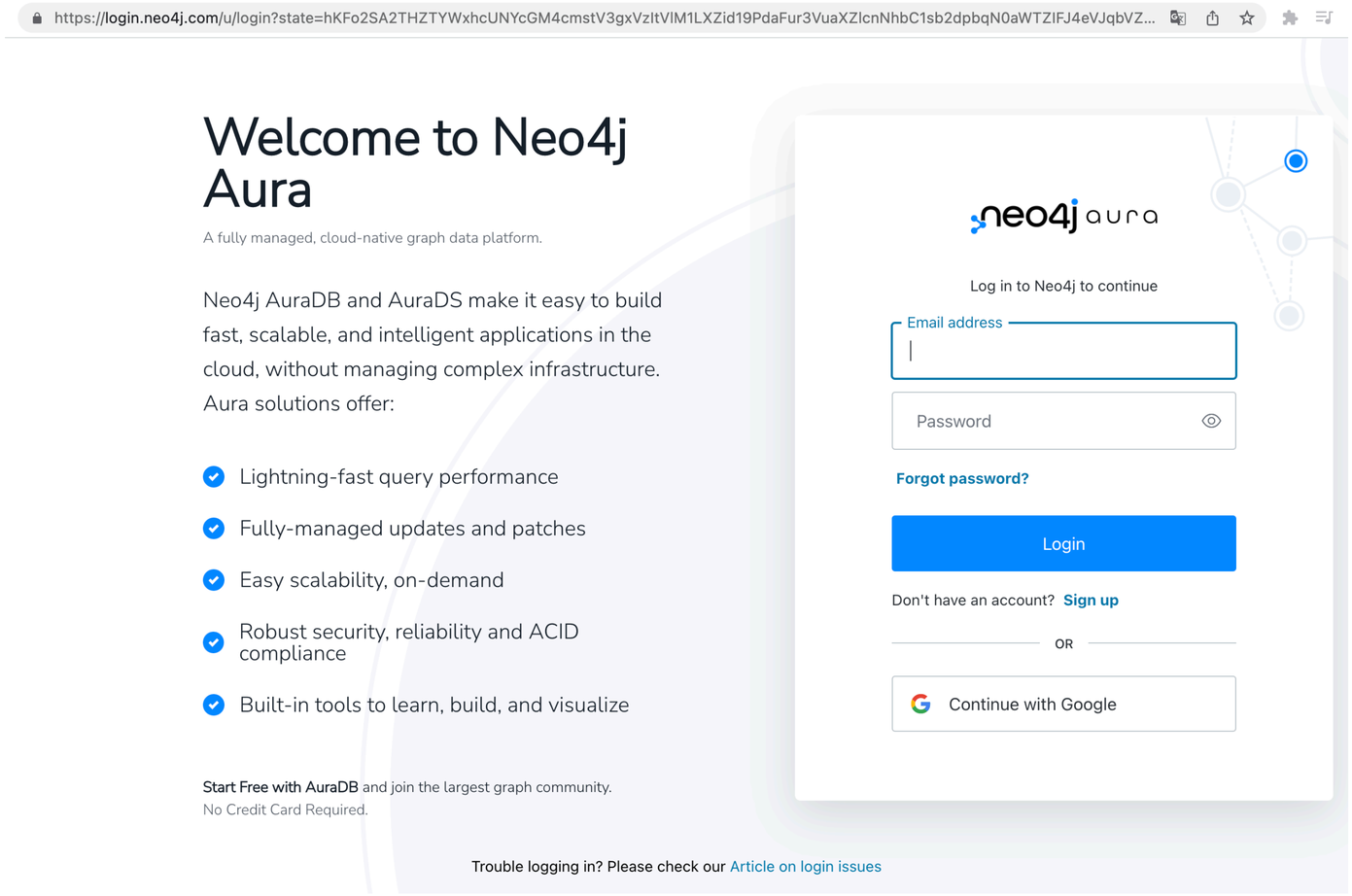




PUC Minas

Demonstração

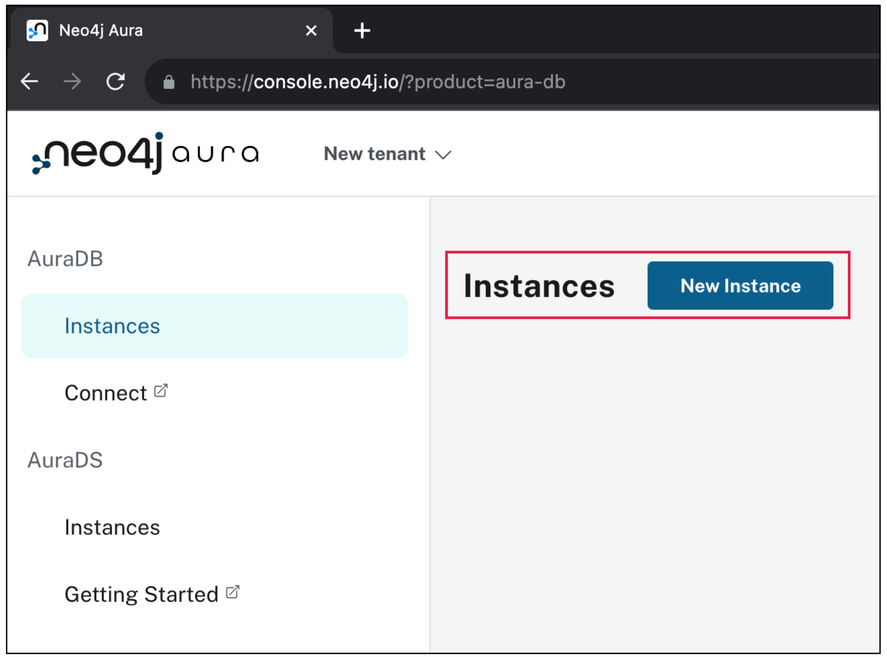
Criar uma nova conta para Neo4j AuraDB através do link: https://login.neo4j.com/u/login?state=hKFo2SA2THZTYWxhcUNYcGM4cmstV3gxVzItVlM1LXZid19PdaFur3VuaXZlcnNhbC1sb2dpbqN0aWTZIFJ4eVJqbVZ1UjB2NWtCcGU2bDM4WGVRLVZmOVl5em1Qo2NpZNkgV1NMczYwNDdrT2pwVVNXODNnRFooSnlZaElrNXpZVG8

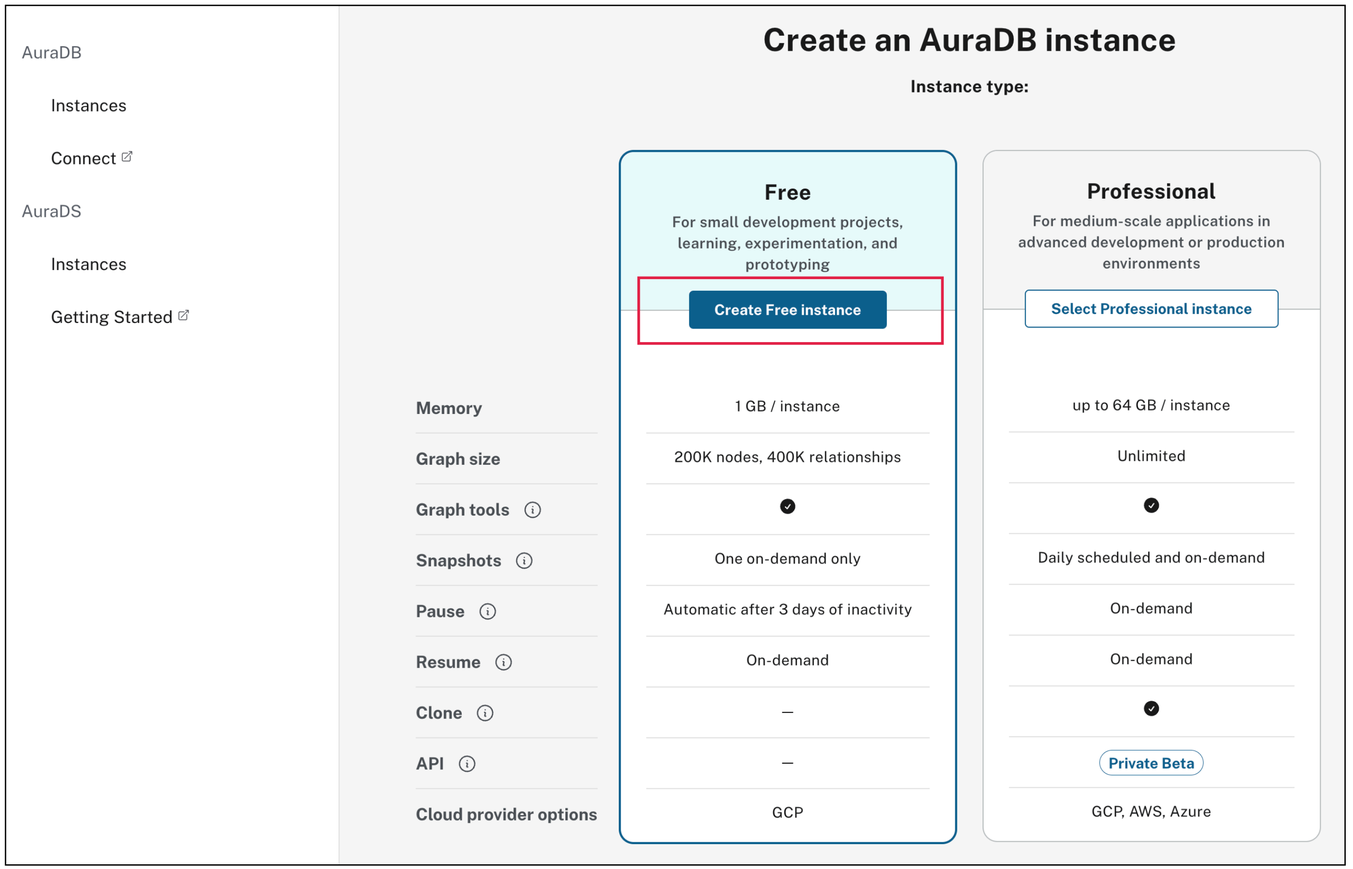


PUC Minas

Demonstração

- Ao conectar no Neo4j, criar uma na instância selecionando no botão ***New Instance***.
- Como nosso propósito será realizar somente algumas demonstrações, selecione o tipo de instância ***Free***.

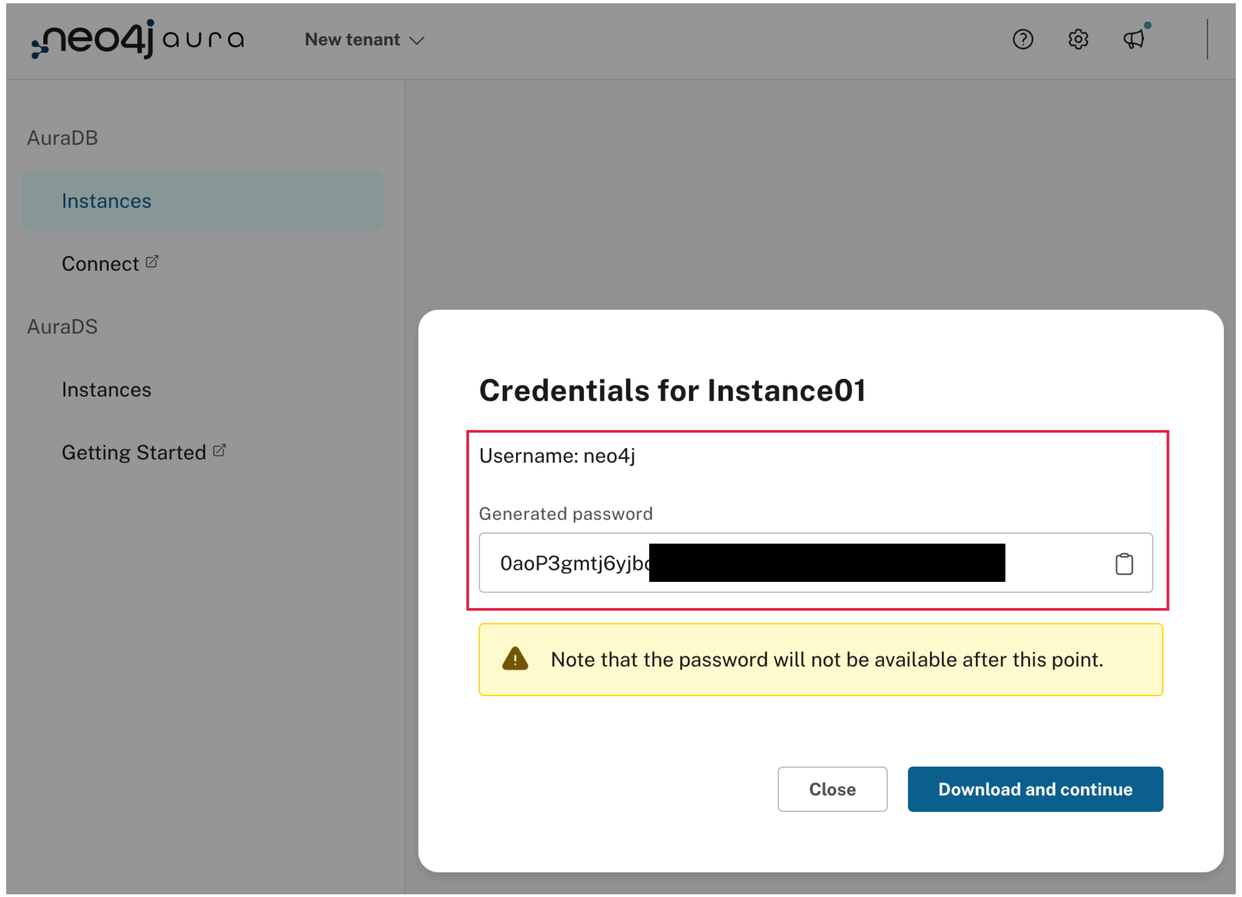




PUC Minas

Demonstração

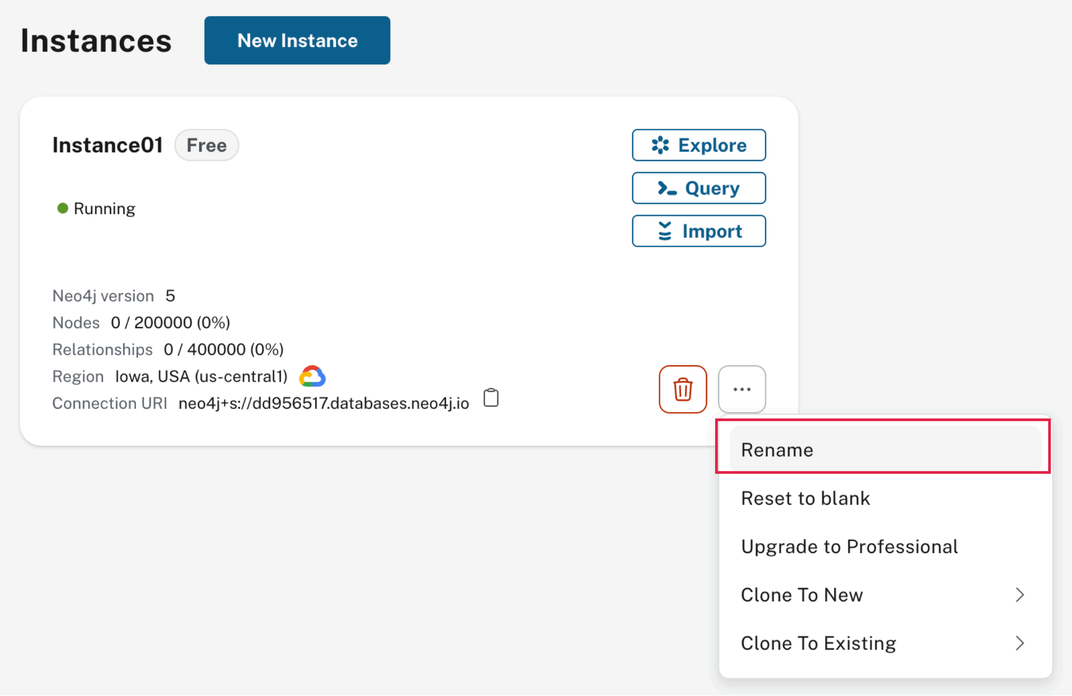
- Copie ou faça o download da senha gerada, conforme é apresentado na tela abaixo.
- Obs.: não será possível obter a senha novamente a partir deste ponto.



PUC Minas

Demonstração

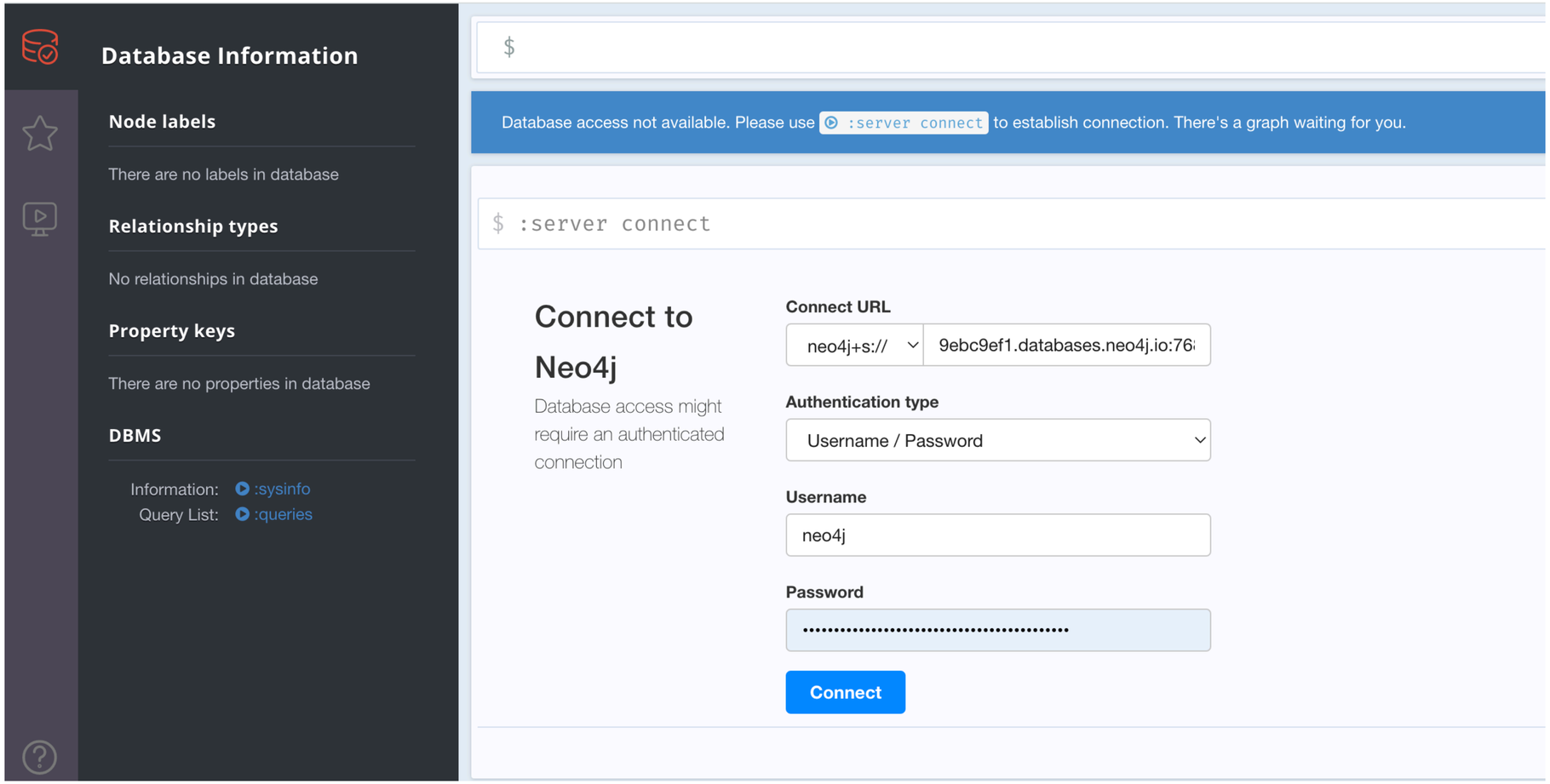
- Para renomear a nova instância selecione a opção conforme demonstrado na figura.
- Selecionar o botão *Query* para iniciar o módulo de execução de instruções.



PUC Minas

Demonstração

- Utilizar as credenciais criada na etapa anterior, conectar a nova instância.



PUC Minas

Demonstração

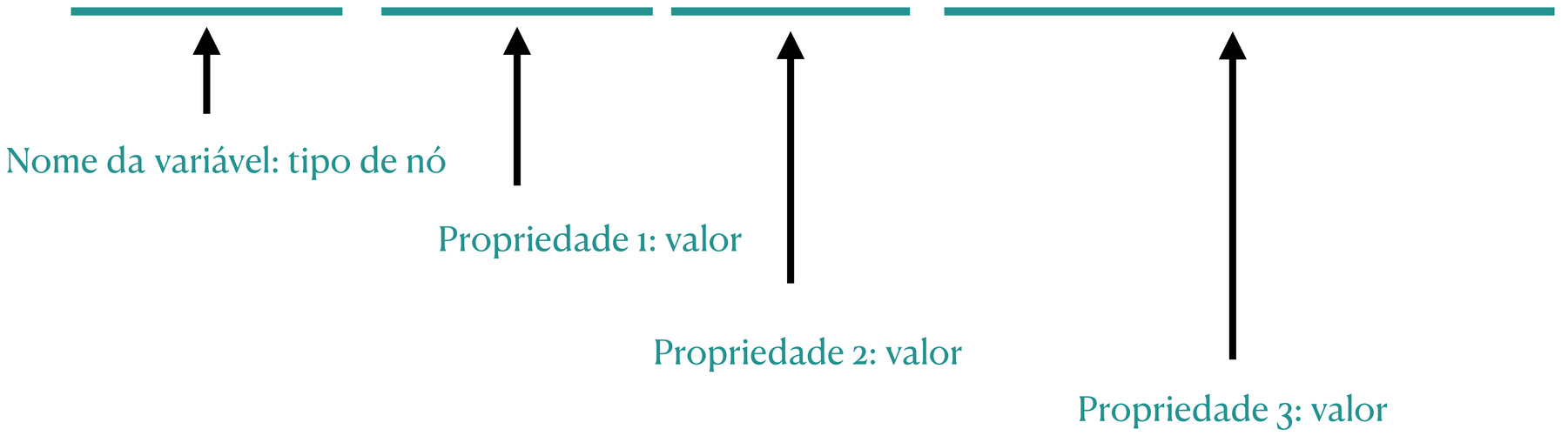
- Só há dois tipos de dados no Neo4J: o nó (semelhante ao documento no MongoDB) e o relacionamento (aresta).
- Um nós pode ter um tipo e possuir diversos atributos.

PUC Minas

Demonstração - Criando nó

- Vamos criar 2 nós:
  - Um para o tipo Person(Pessoa) com os atributos name(nome) e born(nascimento)
  - E outro para o tipo Movie(Filme) com os atributos título(title), lançamento(released) e slogan(tagline).
- Execute o seguinte comando na interface:

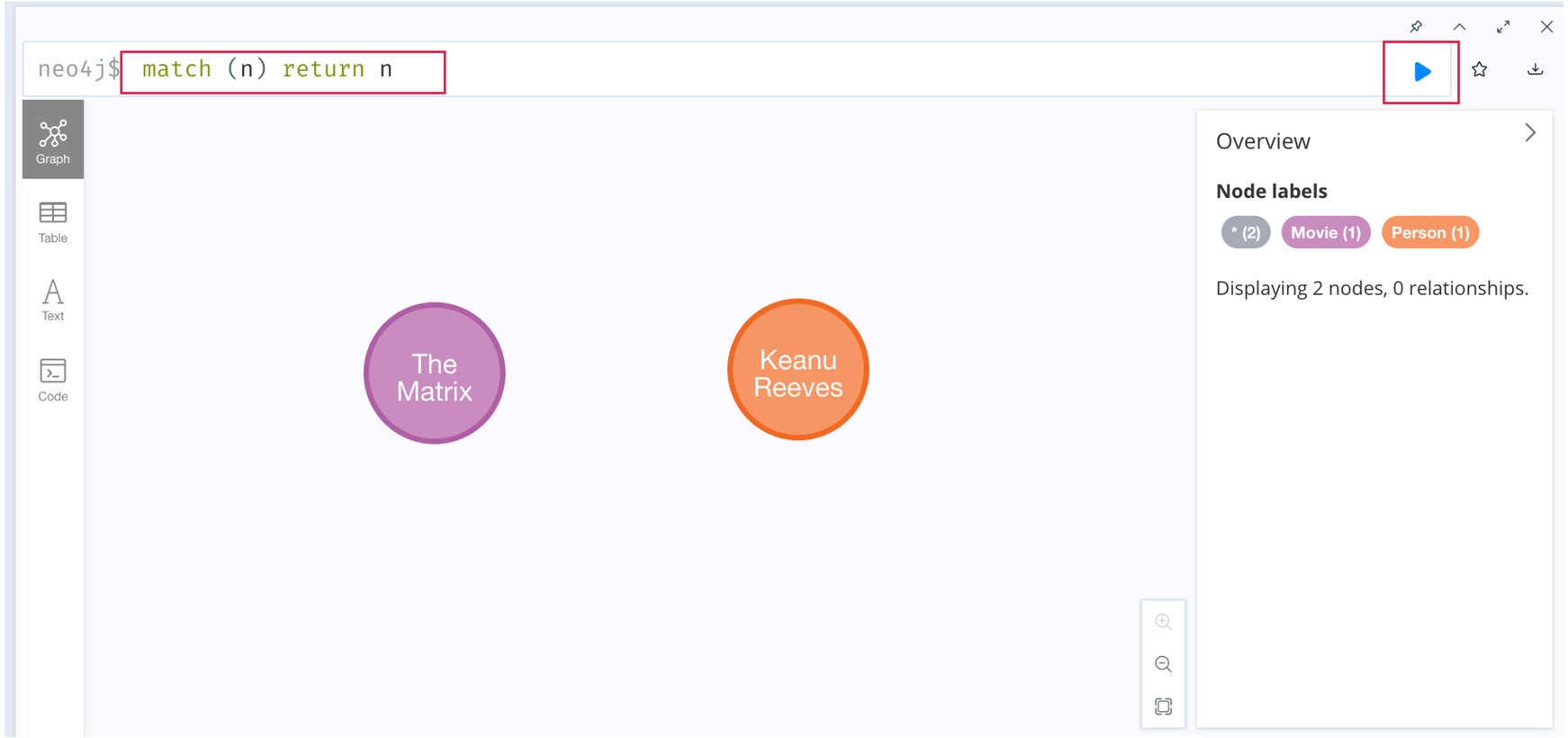
```
CREATE (:Person {name:'Keanu Reeves', born:1964})
CREATE ( TheMatrix:Movie { title: 'The Matrix', released: 1999, tagline: 'Welcome to the Real World' } )
```



PUC Minas

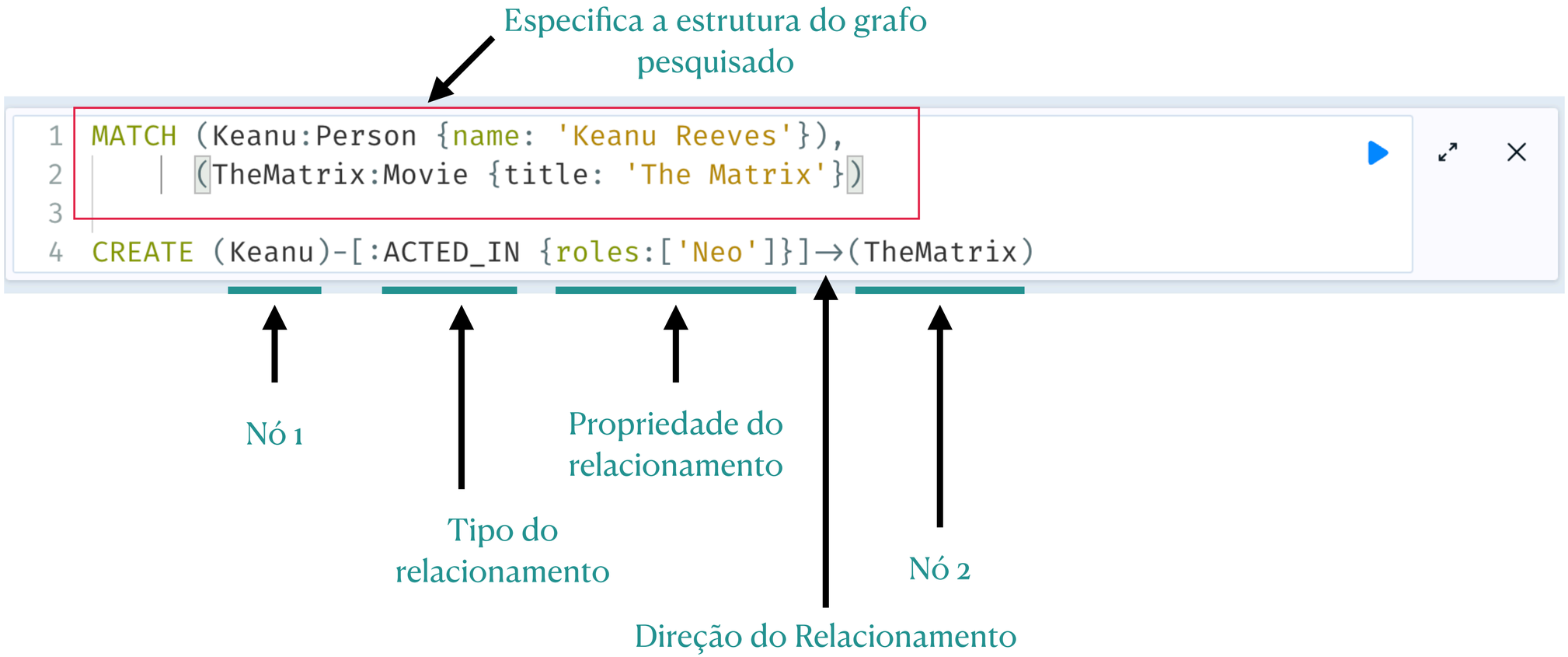
Demonstração - Consultando os nós

- Para recuperar os nós criados, basta executar o seguinte comando na interface:



Demonstração - Criando um relacionamento

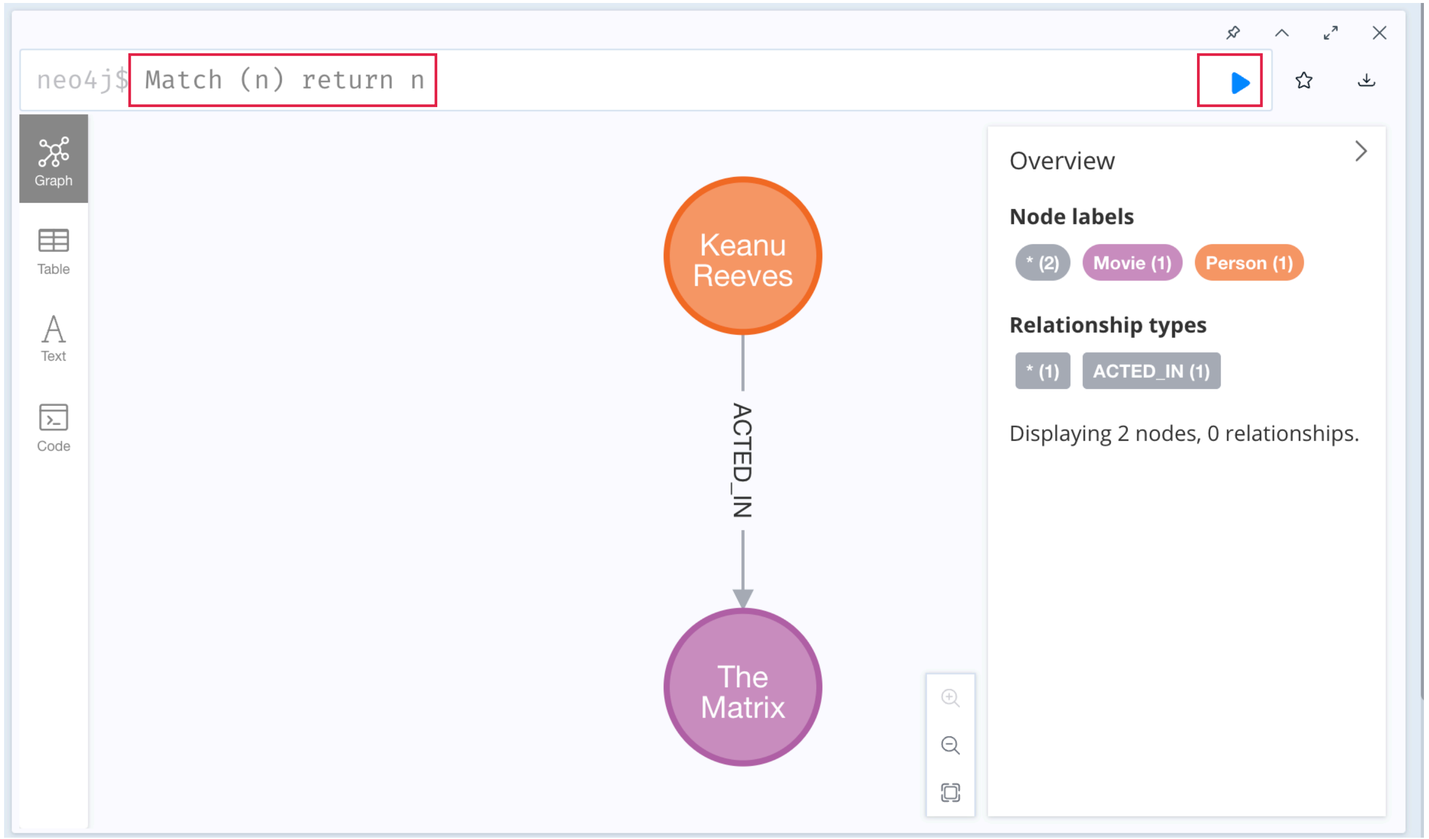
- Para criar um relacionamento entre os dois nós (Movie e Person), execute o seguinte comando:



PUC Minas

Demonstração - Consultando os nós relacionados

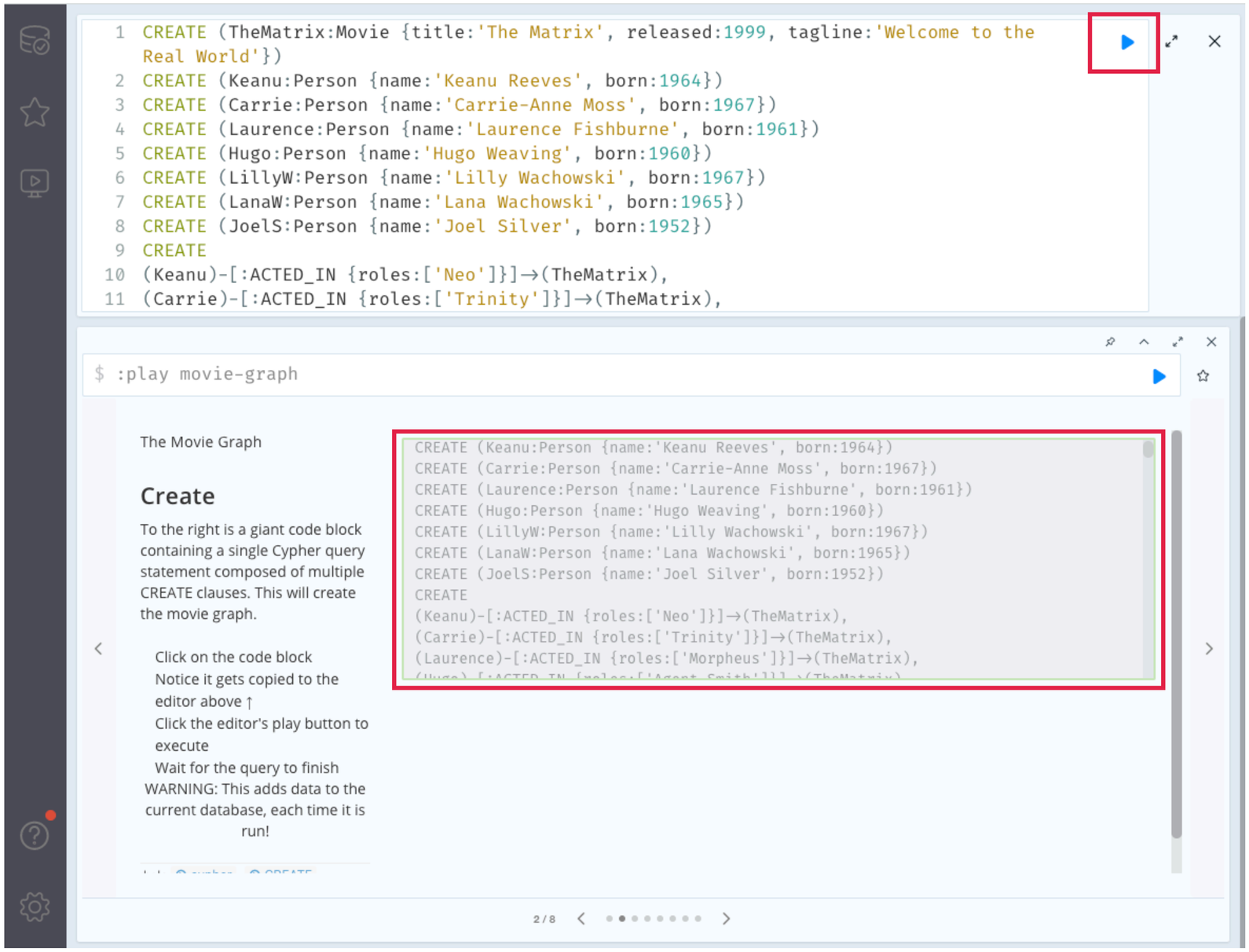
- Para consultar os nós relacionados, execute o seguinte comando na interface:



PUC Minas

Demonstração

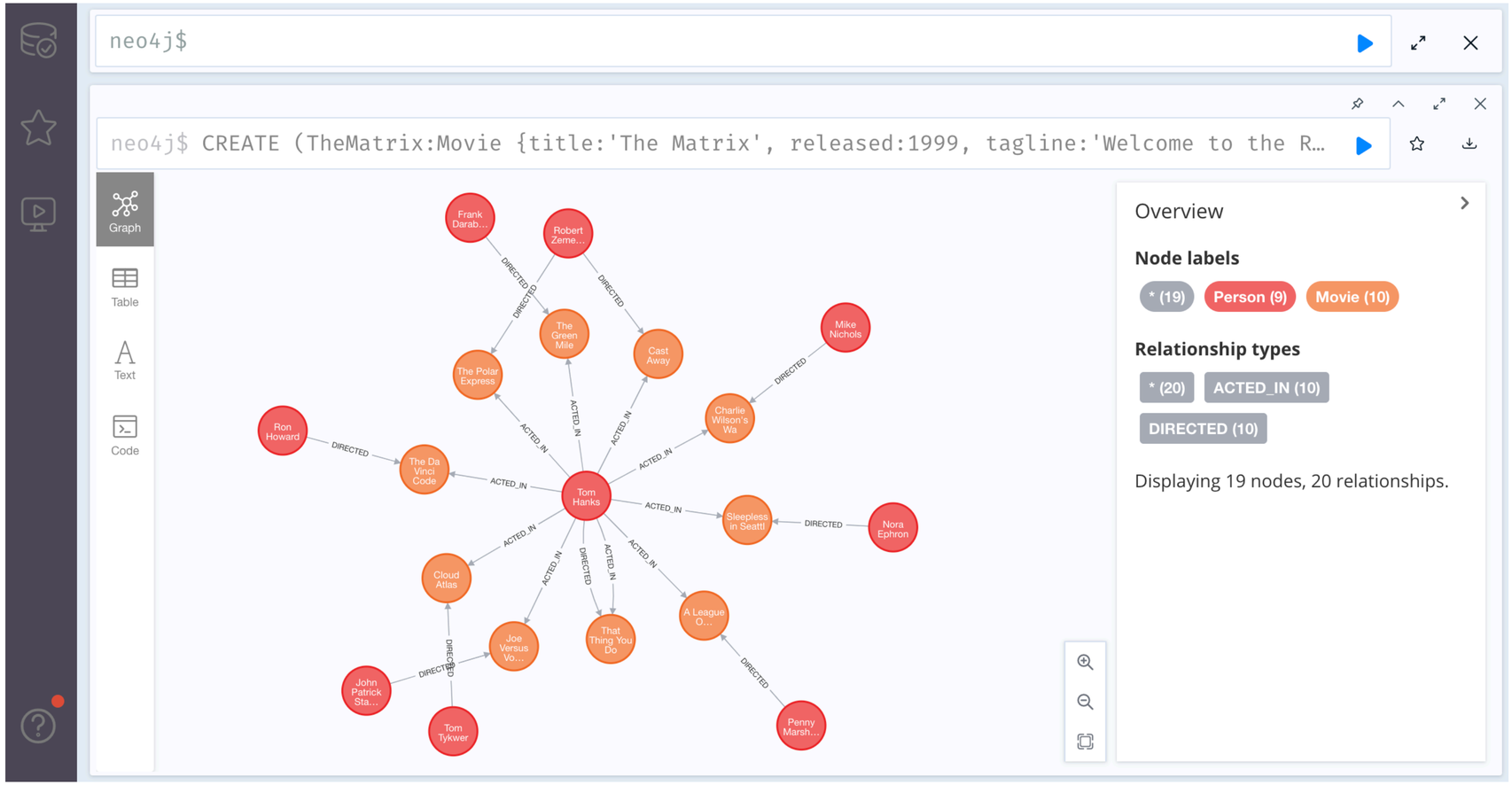
- Nesta etapa será possível criar os dados que irão popular a base de dados de filmes (movies).
- Será criado os nós atores (Person) e filmes (Movie) e diversos relacionamentos (ACTED_IN, DIRECTED, WROTE, entre outros) e propriedades (name, born e outros).



PUC Minas

Demonstração

- Base de Filmes (Movies) criada.



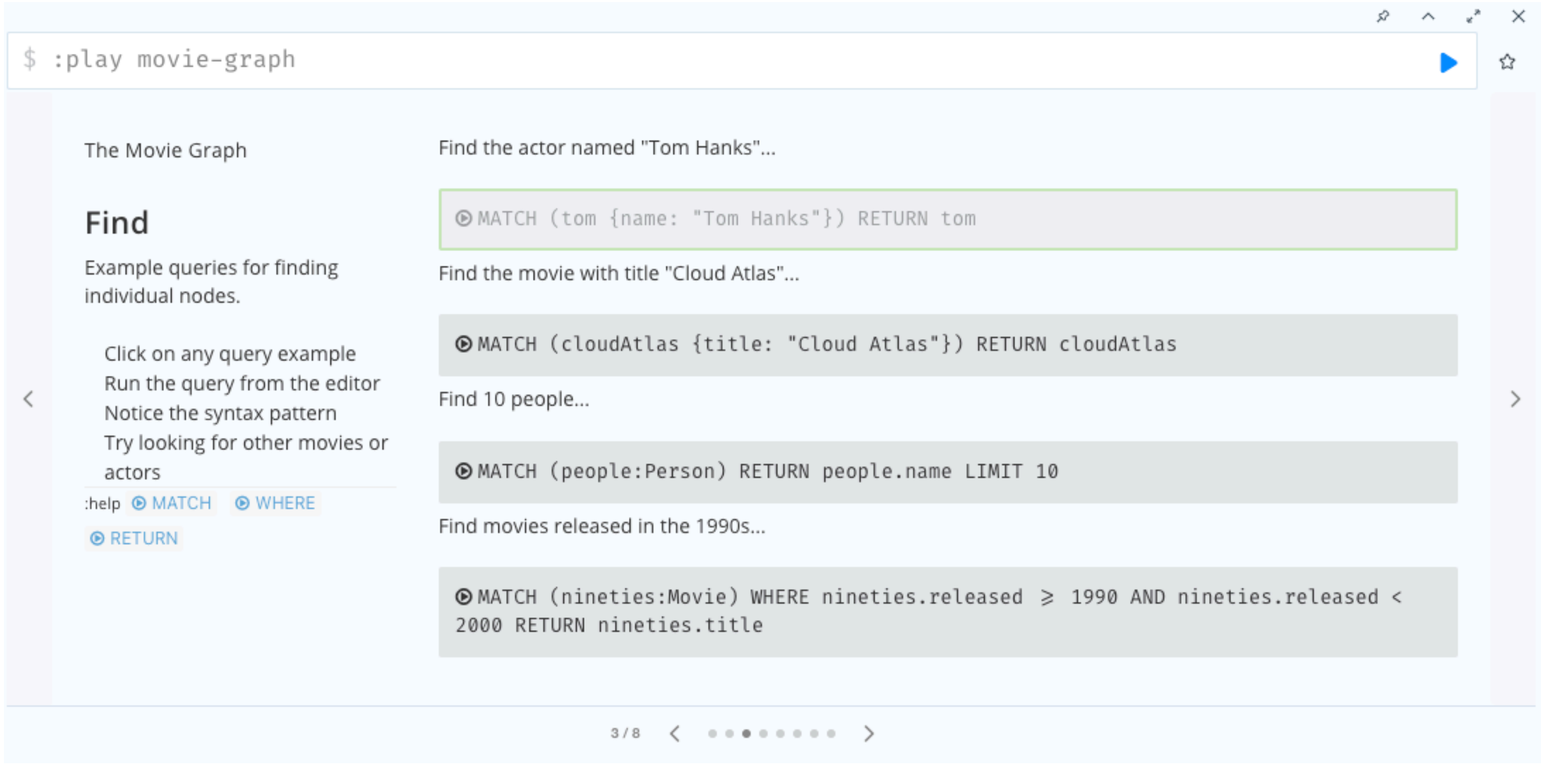
PUC Minas

Demonstração

- Consultas para recuperar nós individuais com ou sem critério de filtro.

- As consultas podem ser escritas de forma alternativa:

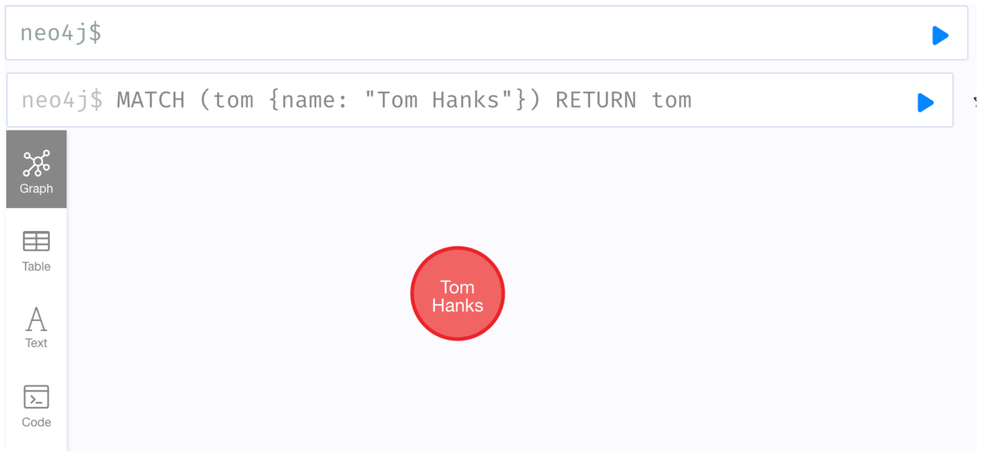
```
MATCH (m:Movie)
WHERE m.title = "Cloud Atlas"
RETURN m

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

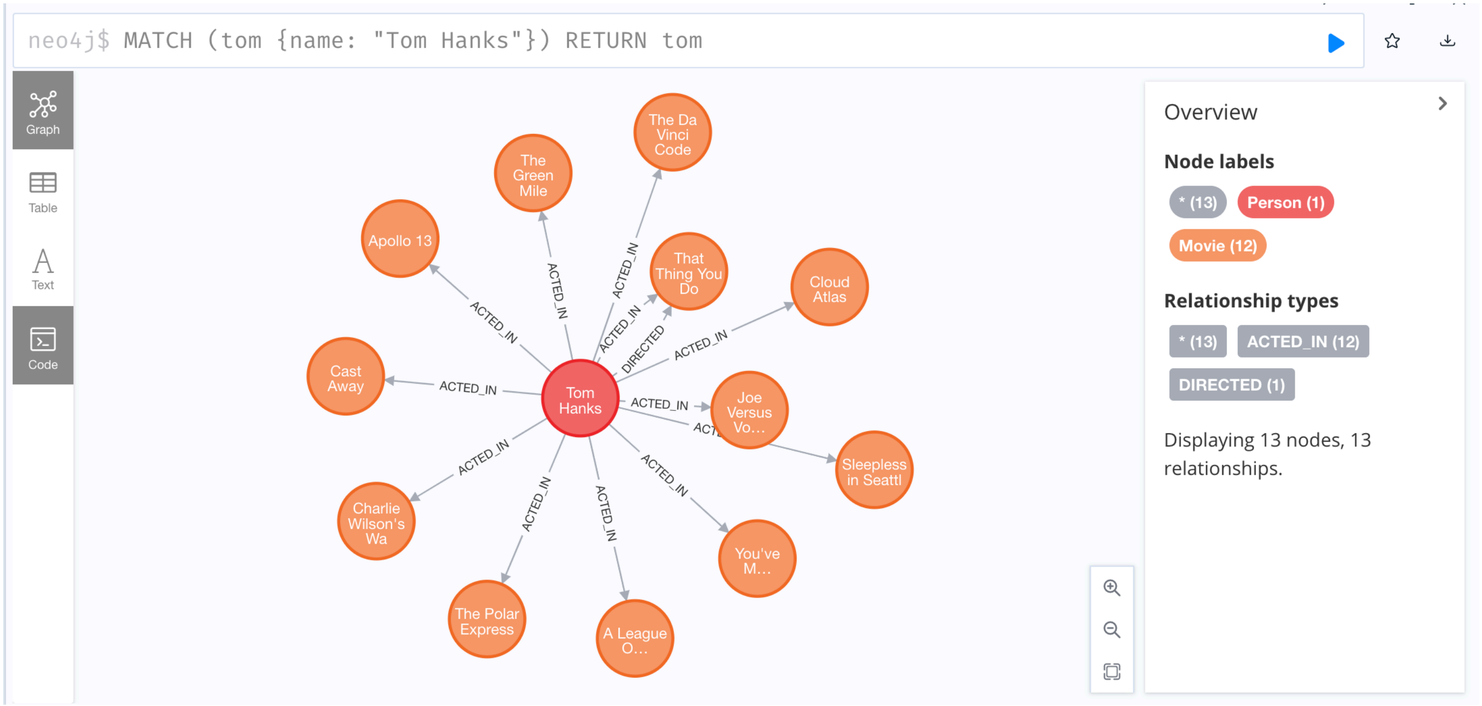


PUC Minas

Demonstração

- Um duplo clique no nó Tom Hanks irá recuperar todos os nós com relacionamento de primeiro nível.

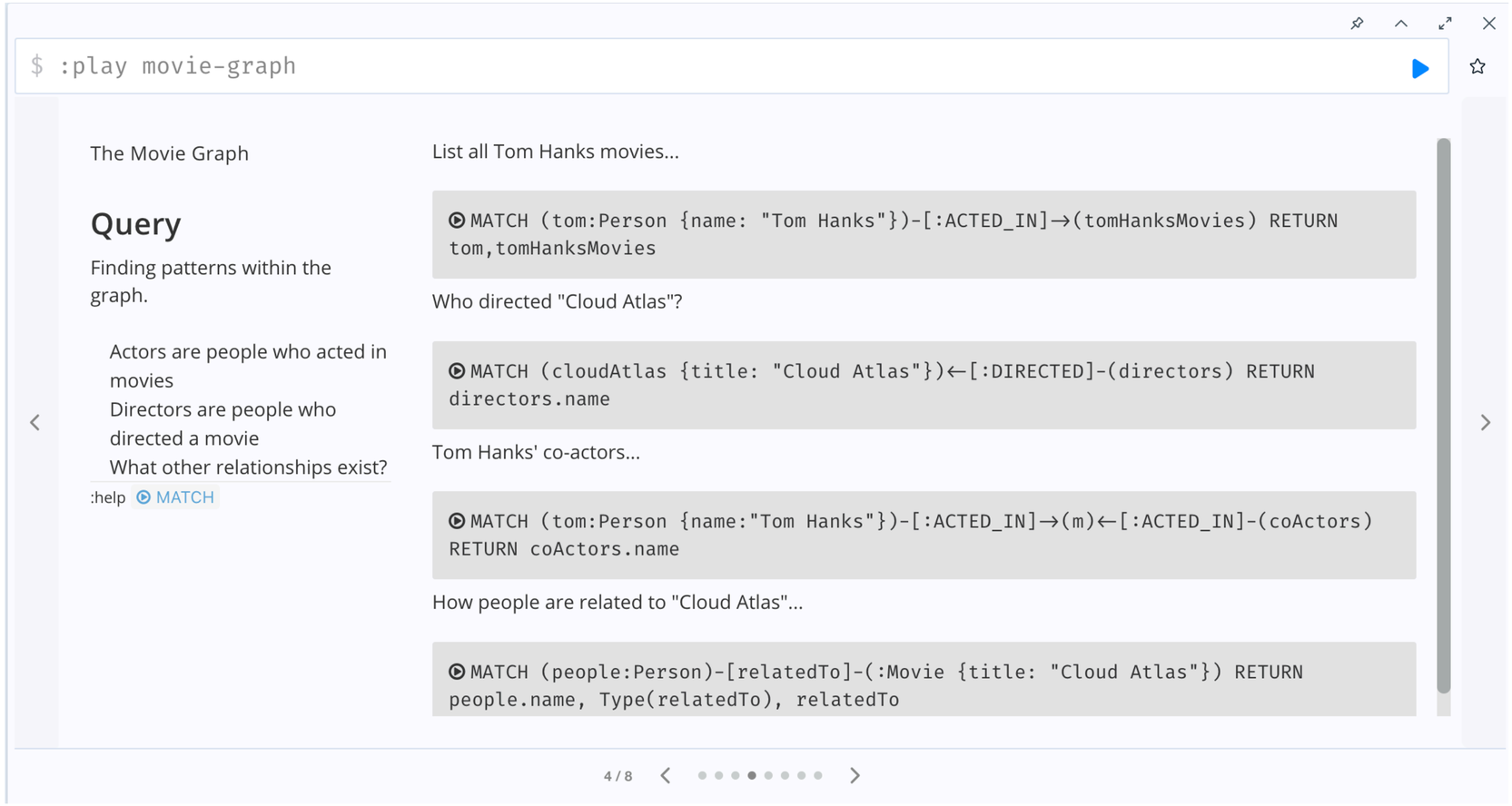




PUC Minas

Demonstração

- Consultas para recuperar nós com padrões estabelecidos.

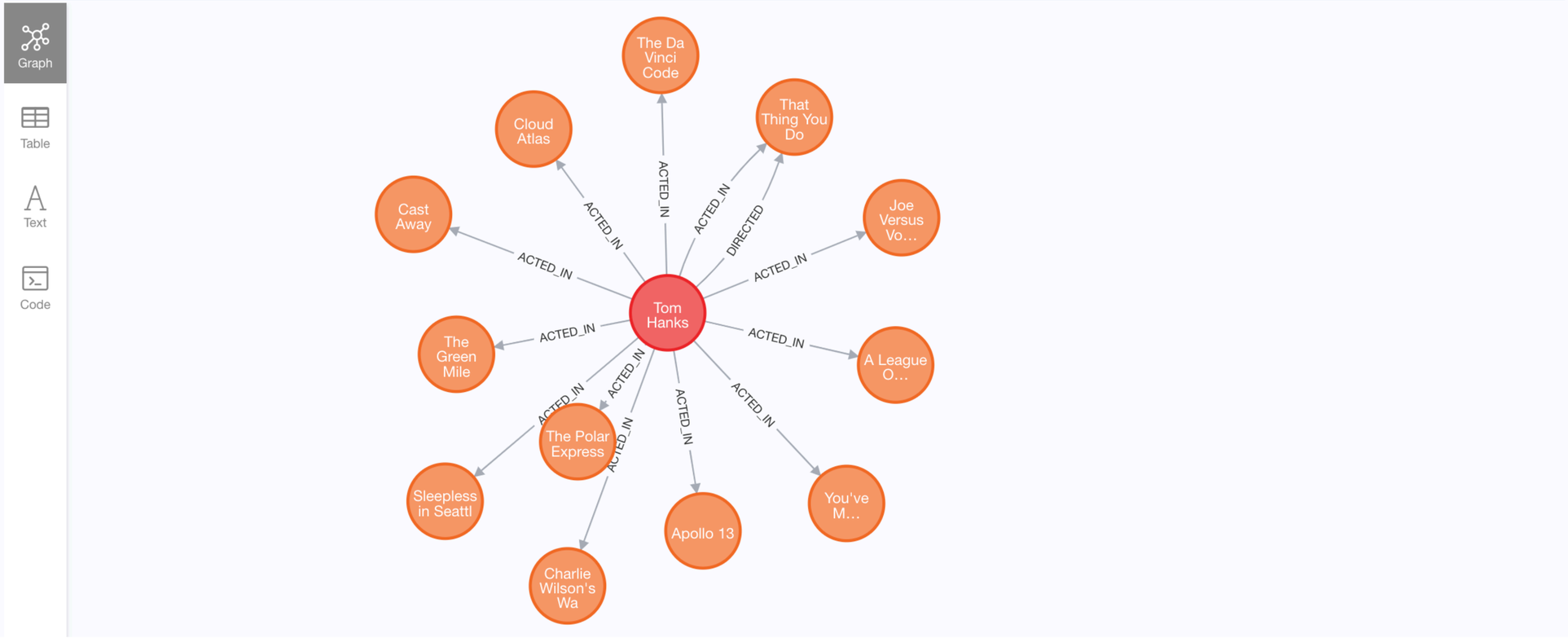


PUC Minas

Demonstração

- Lista todos os filmes que Tom Hanks atuou.

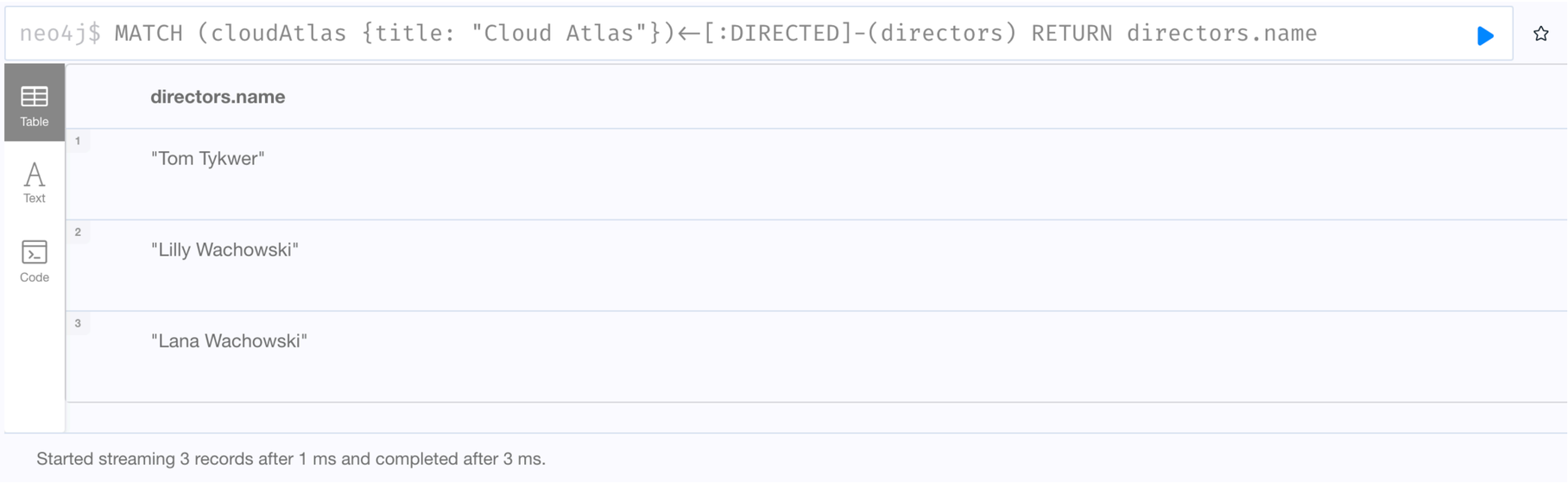
```
neo4j$ MATCH (tom:Person {name: "Tom Hanks"})-[:ACTED_IN]→(tomHanksMovies) RETURN tom,tomHanksMovies
```



PUC Minas

Demonstração

- Lista todos os diretores do filme Cloud Atlas



PUC Minas

Demonstração

- Lista todos os co-atores que atuaram nos filmes com Tom Hanks

```
neo4j$ MATCH (tom:Person {name:"Tom Hanks"})-[:ACTED_IN]→(m)←[:ACTED_IN]-(coActors) RETURN coActors.na…
```

Table | Text | Code

	coActors.name
1	"Ed Harris"
2	"Gary Sinise"
3	"Kevin Bacon"
4	"Bill Paxton"
5	"Parker Posey"
6	"Greg Kinnear"
7	

Started streaming 39 records after 1 ms and completed after 4 ms.

PUC Minas

Demonstração

- Lista todos as pessoas que trabalharam no filme Cloud Atlas

```
neo4j$ MATCH (people:Person)-[relatedTo]-(:Movie {title: "Cloud Atlas"}) RETURN people.name, Type(relatedTo), related…
```

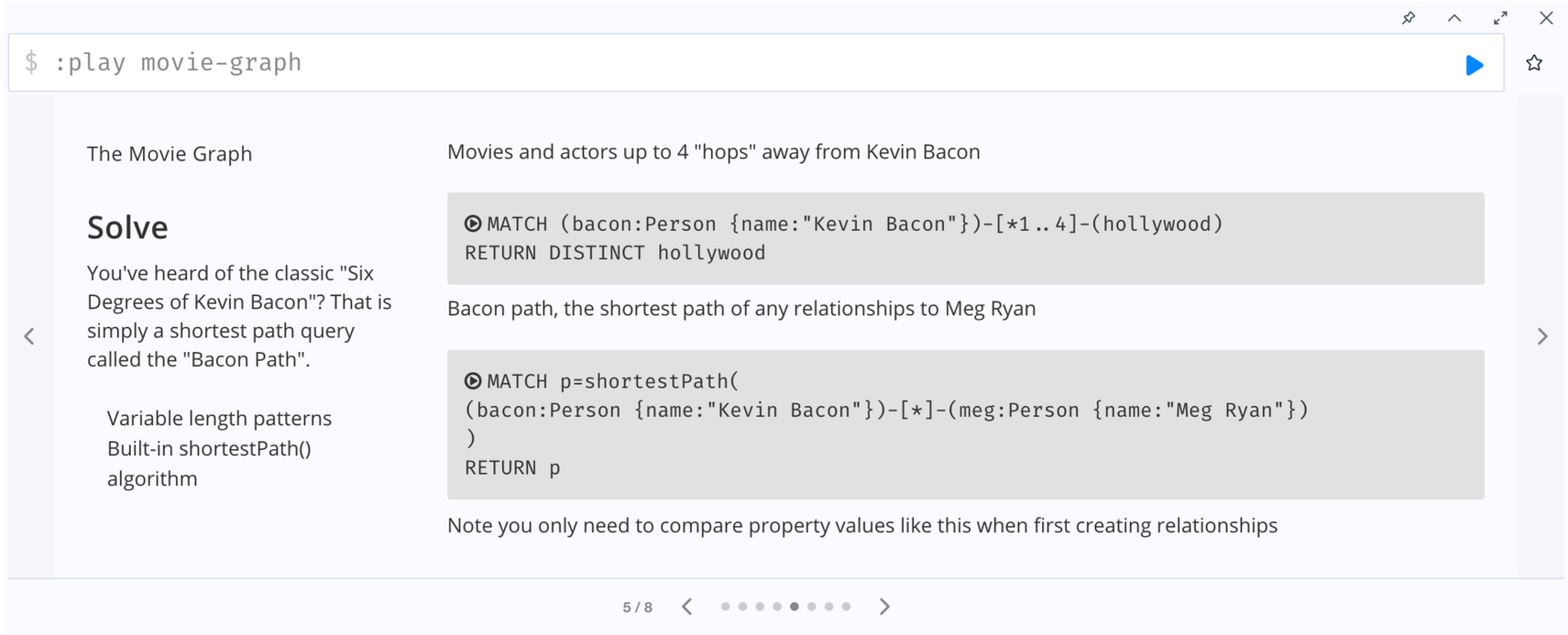
Table | Text | Code

"people.name"	"Type(relatedTo)"	"relatedTo"
"Tom Hanks"	"ACTED_IN"	{"roles":["Zachry","Dr. Henry Goose","Isaac Sachs","Dermot Hoggins"]}
"Jim Broadbent"	"ACTED_IN"	{"roles":["Vyvyan Ayrs","Captain Molyneux","Timothy Cavendish"]}
"David Mitchell"	"WROTE"	{}
"Tom Tykwer"	"DIRECTED"	{}
"Lana Wachowski"	"DIRECTED"	{}
"Stefan Arndt"	"PRODUCED"	{}
"Jessica Thompson"	"REVIEWED"	{"summary":"An amazing journey","rating":95}
"Halle Berry"	"ACTED_IN"	{"roles":["Luisa Rey","Jocasta Ayrs","Ovid","Meronym"]}
"Hugo Weaving"	"ACTED_IN"	{"roles":["Bill Smoke","Haskell Moore","Tadeusz Kesselring","Nurse Noakes","Boardman Mephi","Old Georgie"]}
"Lilly Wachowski"	"DIRECTED"	{}

PUC Minas

Demonstração

- Consultas baseadas em níveis de relacionamento e menor caminho

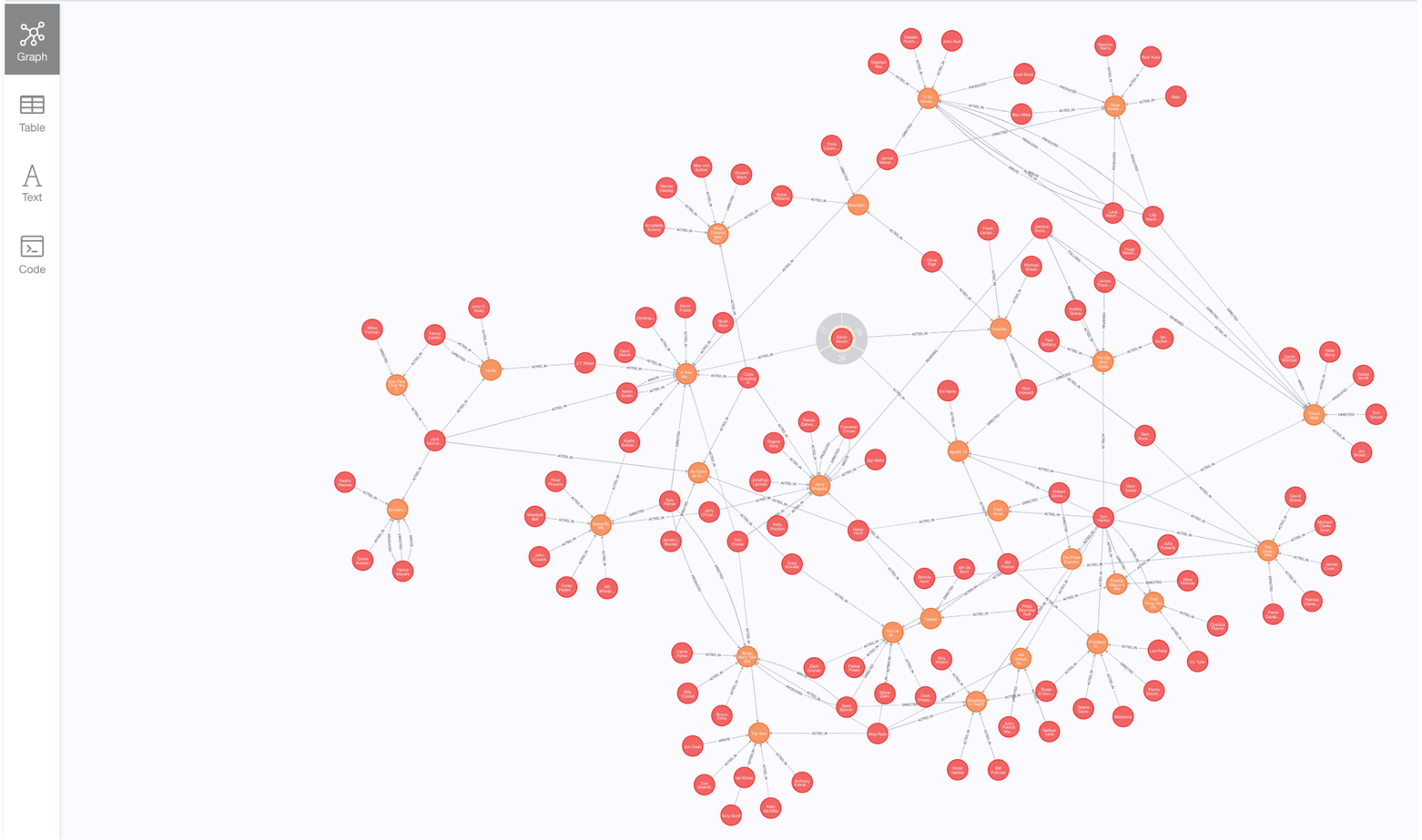


PUC Minas

Demonstração

- Recupera todos os filmes e atores com até 4 níveis de relacionamento do ator Kevin Bacon

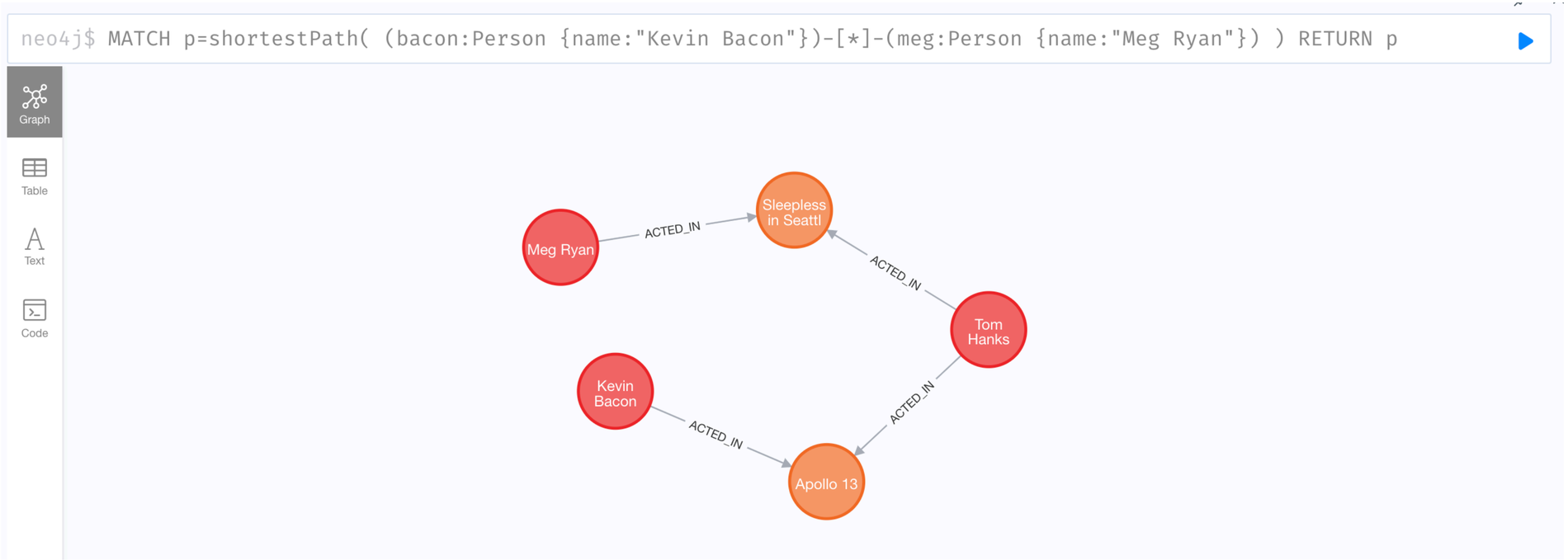
```
1  MATCH (bacon:Person {name:"Kevin Bacon"})-[*1..4]-(hollywood)
2  RETURN DISTINCT hollywood
```



PUC Minas

Demonstração

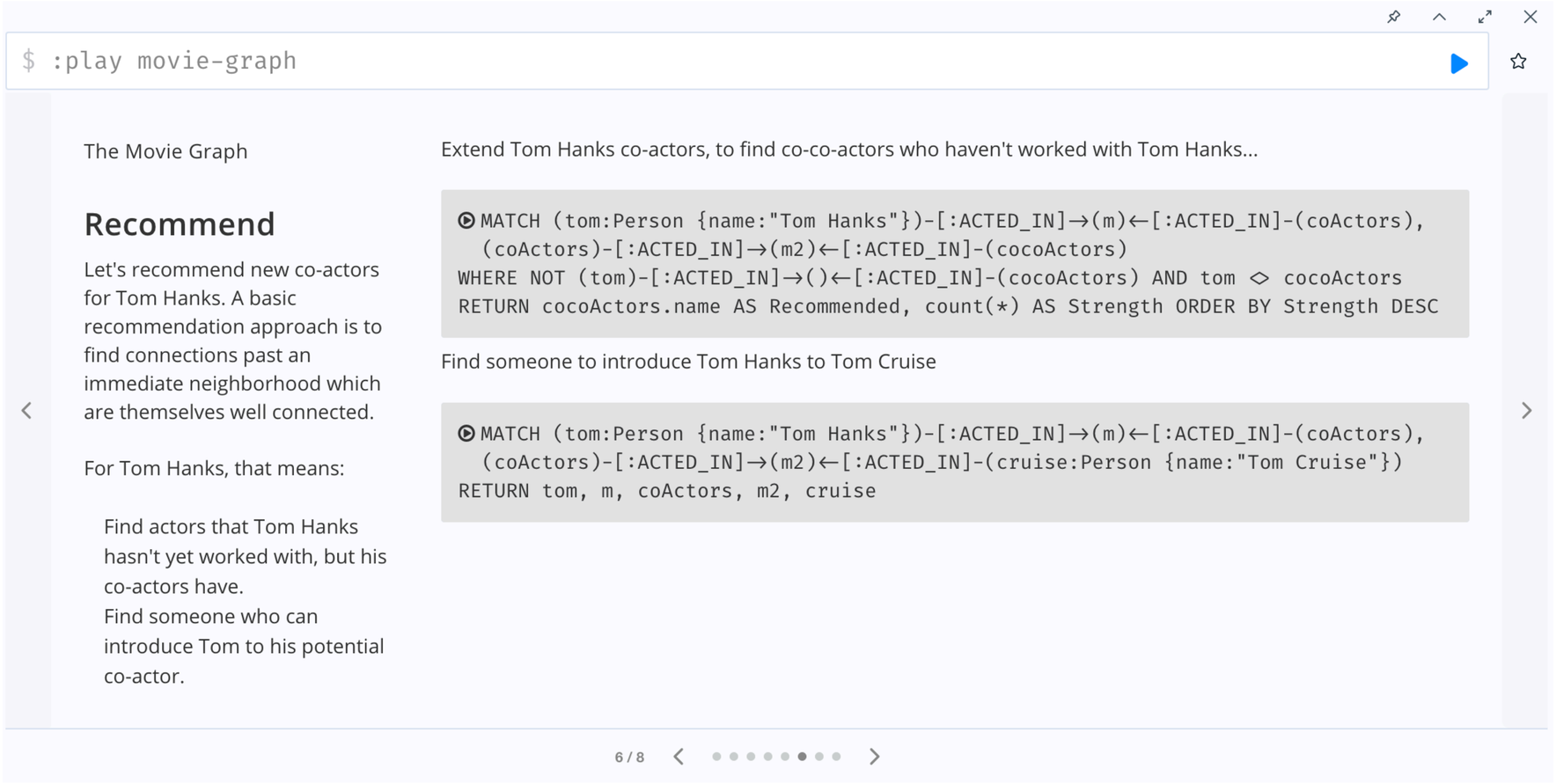
- Recupera o caminho mais curto entre os nós Kevin Bacon e Meg Ryan.



PUC Minas

Demonstração

- Abordagem de recomendação básica para encontrar conexões além de uma vizinhança imediata que estejam bem conectadas.



PUC Minas

Demonstração

- Lista os co-atores de Tom Hanks, para encontrar co-co-atores que não trabalharam com Tom Hanks.

```
1 MATCH (tom:Person {name:"Tom Hanks"})-[:ACTED_IN]→(m)←[:ACTED_IN]-(coActors),
2   (coActors)-[:ACTED_IN]→(m2)←[:ACTED_IN]-(cocoActors)
3 WHERE NOT (tom)-[:ACTED_IN]→()←[:ACTED_IN]-(cocoActors) AND tom <> cocoActors
4 RETURN cocoActors.name AS Recommended, count(*) AS Strength ORDER BY Strength DESC
```

Table | Text | Code

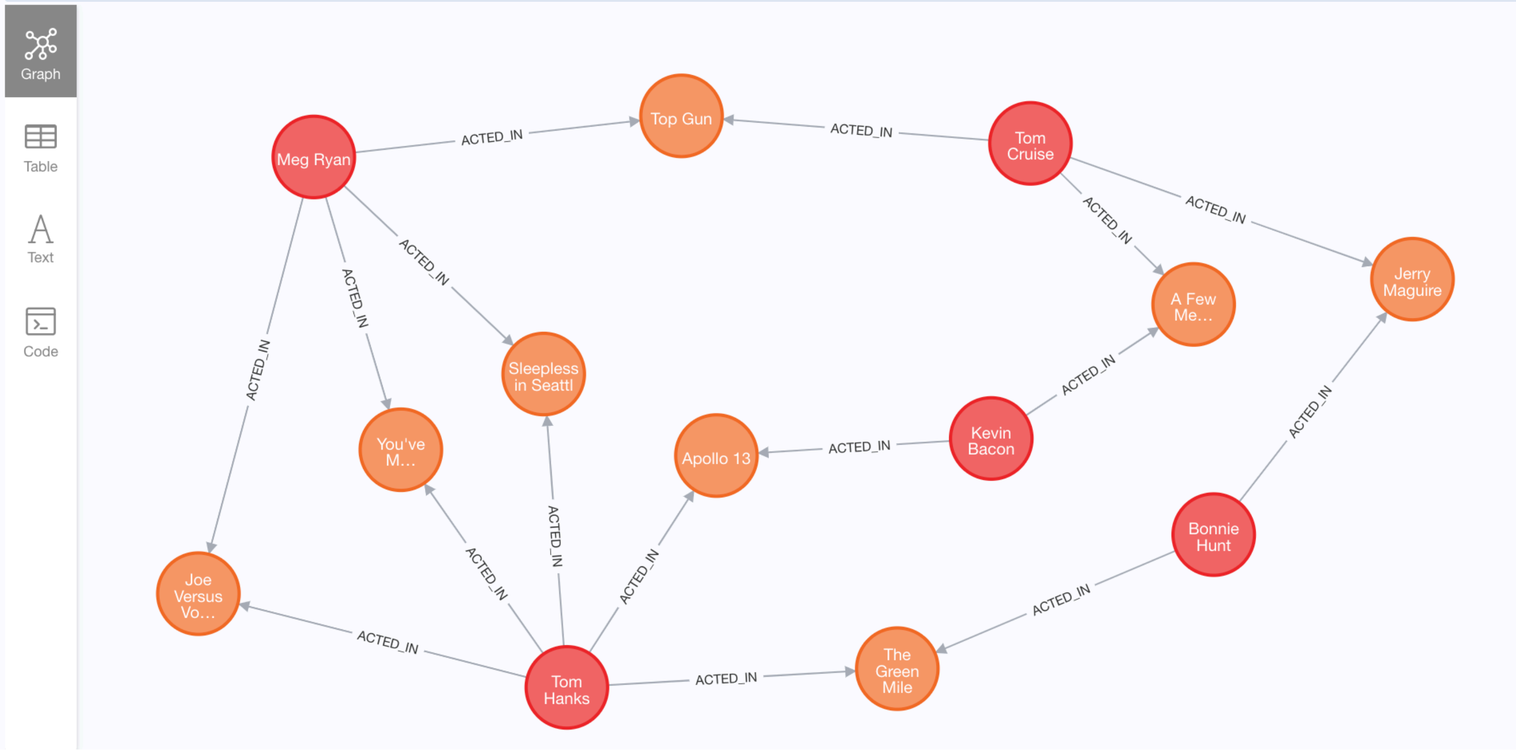
Recommended	Strength
"Tom Cruise"	5
"Zach Grenier"	5
"Cuba Gooding Jr."	4
"Keanu Reeves"	4
"Jack Nicholson"	3
"Val Kilmer"	3
"Tom Skerritt"	3
"Kelly McGillis"	3
"Anthony Edwards"	3
"Carrie Fisher"	3
"Billy Crystal"	3
"Bruno Kirby"	3
"Laurence Fishburne"	3
"Carrie-Anne Moss"	3
"Michael Sheen"	2
"Frank Langella"	2
"Oliver Platt"	2
"James Marshall"	1
"Kevin Pollak"	1
"J.T. Walsh"	1
"Aaron Sorkin"	1
"Christopher Guest"	1
"Noah Wyle"	1

PUC Minas

Demonstração

- Encontrar alguém que possa apresentar Tom Hanks à Tom Cruise

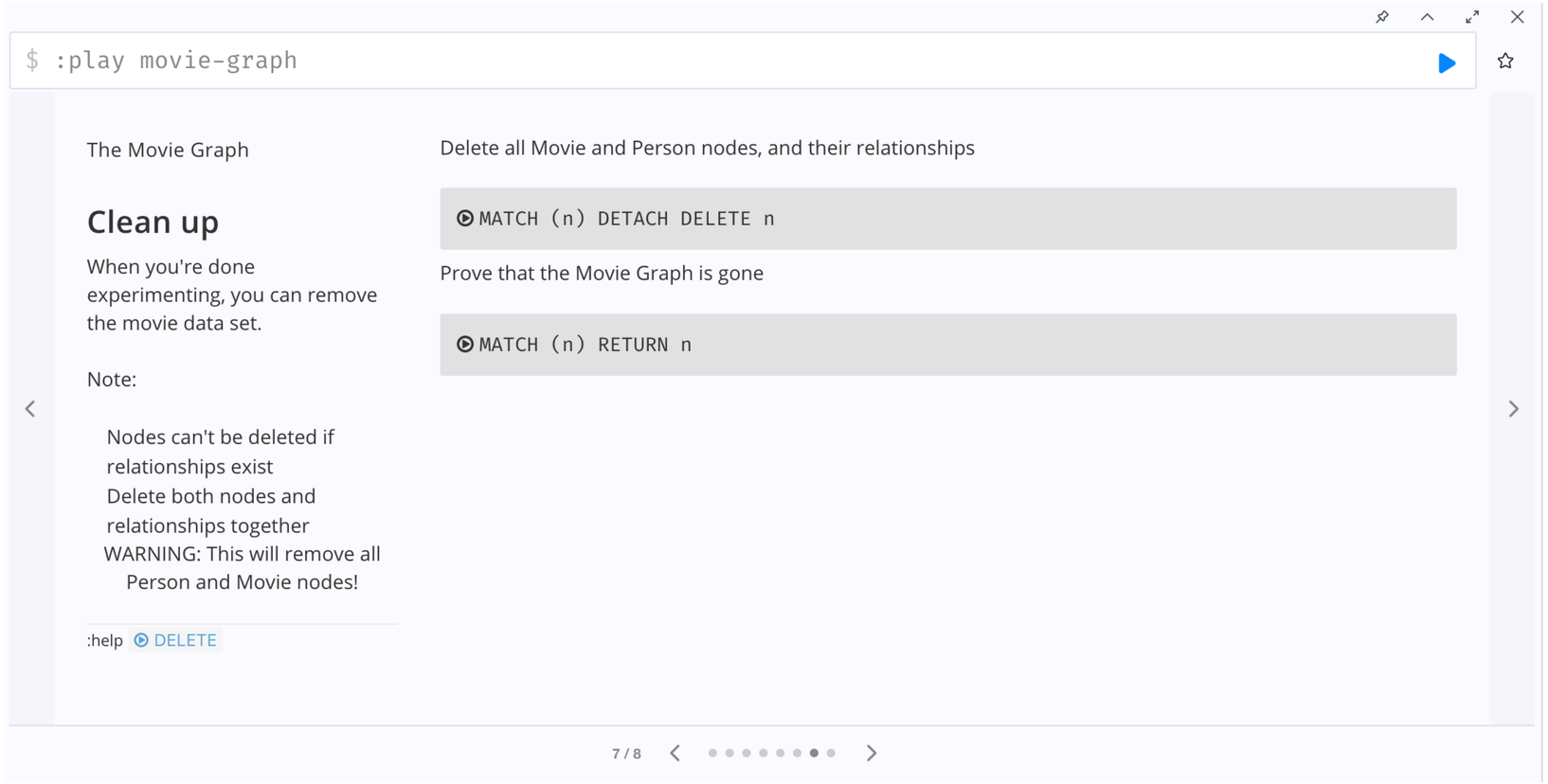
```
1 MATCH (tom:Person {name:"Tom Hanks"})-[:ACTED_IN]→(m)←[:ACTED_IN]-(coActors),
2 | (coActors)-[:ACTED_IN]→(m2)←[:ACTED_IN]-(cruise:Person {name:"Tom Cruise"})
3 RETURN tom, m, coActors, m2, cruise
```



PUC Minas

Demonstração

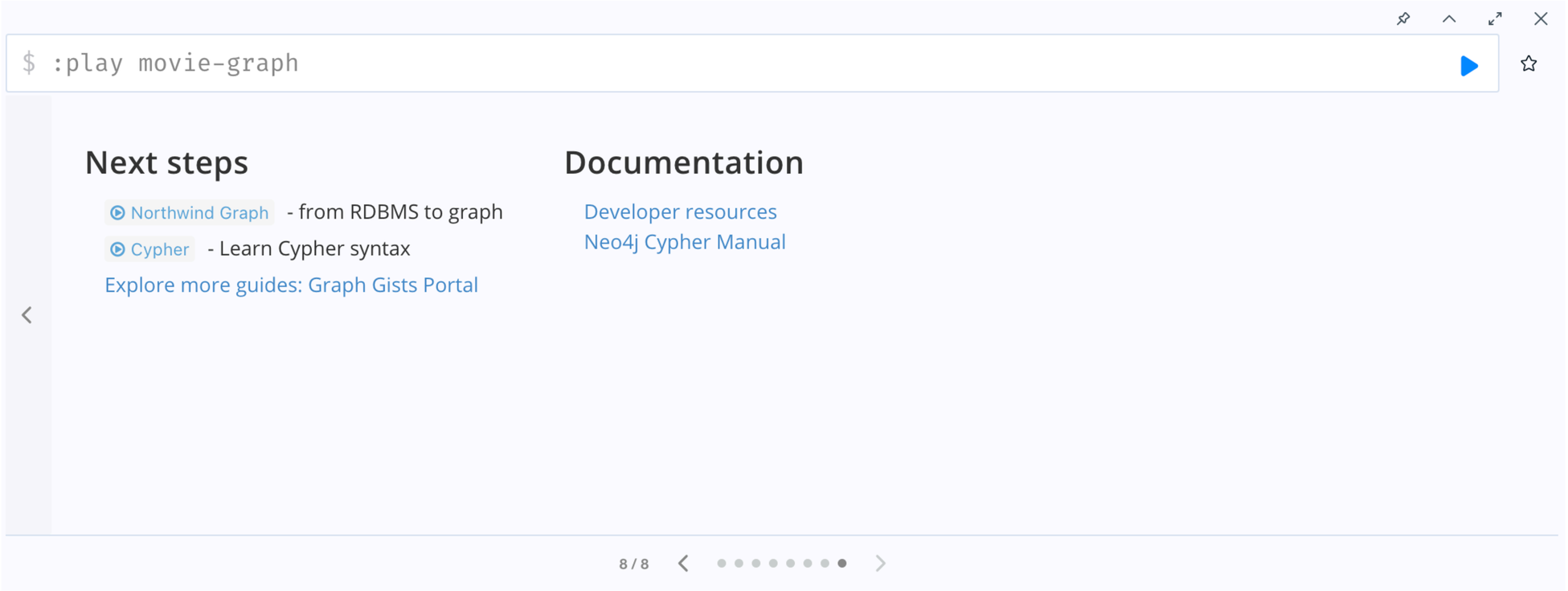
- Remover todos os nós e seus relacionamentos



PUC Minas

Demonstração

- Próximos passos



PUC Minas

Referências

- Documentação do Neo4J
  - Https://neo4j.com/developer/
- Documentação
  - https://neo4j.com/developer/cypher/
- Manual
  - http://neo4j.com/docs/developer-manual/current/#cypher-query-lang
- Treinamento Oficial
  - https://neo4j.com/graphacademy/