**BÖLÜM 1**

* Generative AI nedir?

Generative AI yani kelime anlamıyla üretken yapay zeka, vermiş olduğumuz veri ve girdilerden türeyen yeni içerik/sonuç oluşturabilen bir yapay zeka türüdür. Deep Learning kullanılarak büyük veri setleri ile eğitilip özelleştirilebilir.

* Hangi alanlarda kullanılır?

Generative AI içerik üretimi, tasarım ve sanat dalları, yazılım geliştirme, sözleşme analizi ve otomasyonu gibi birçok farklı alanda kullanılabilir. Günümüzde son derece popüler olan sosyal medya platformlarında içerik/post üretme amaçlı, grafik tasarım ve mimari taslaklar modellenebilir, kod yazım, otomatik tamamlama, hata bulma ve dokümantasyon gibi işlemlerde zamandan tasarruf sağlayabilir. Sağlık, oyun, eğitim gibi farklı sektörlerde de kullanım alanları oluşmaya başlamıştır.

* GenAI ile grafik veritabanları birleştirildiğinde ne tür uygulamalar gerçekleştirilebilir?

Neo4j gibi grafik veritabanları ilişkisel veritabanlarının haritalanması diyebileceğimiz işlemlerde son derece etkilidir. GenAI graphQL kullanan bu yapılarla birleştiğinde ise, karmaşık işlemleri anlamlandıran öneri sistemleri, sosyal ağ analizleri, sözleşme analizleri, soru-cevap motorları gibi farklı ürünler geliştirilebilir. Hukuki metinlerde yapılan atıflar arasındaki ilişkileri ve bağlantıları tespit edip bunları özetleyen asistan oluşturulabilir. Hali hazırda fraud sistemlerde kullanılan suç analizi gibi durumlarda da ilişkili kişi ve olayları keşfeden sistemler geliştirilebilir.

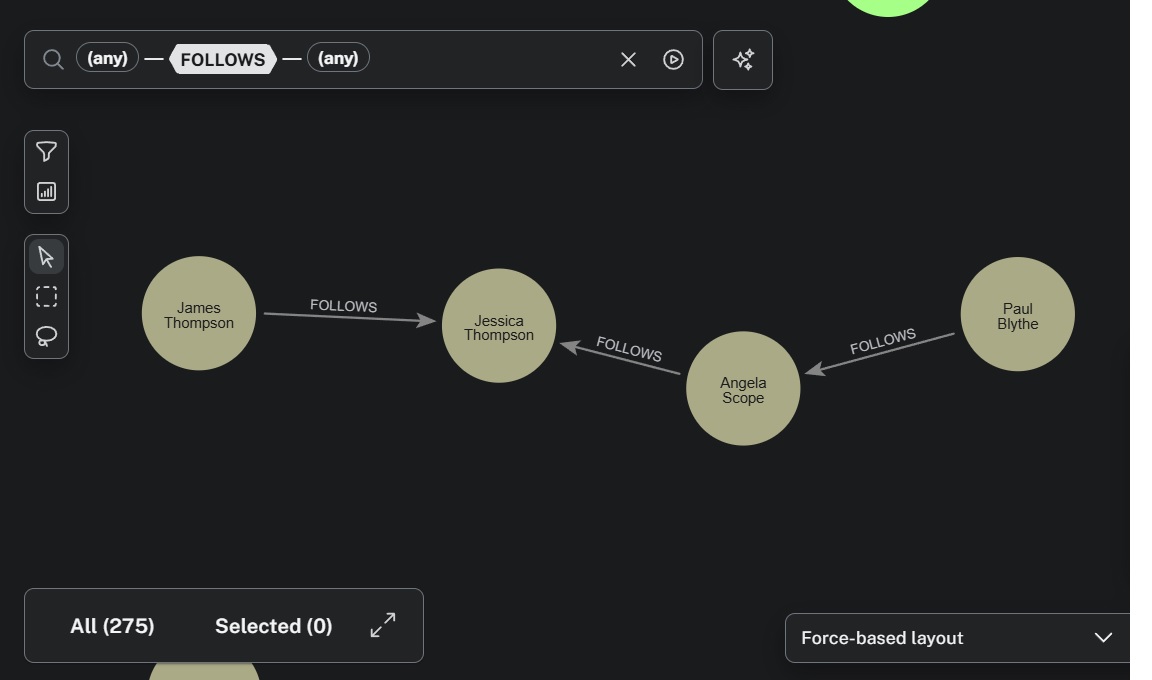
* Riskleri veya etik tartışmaları var mıdır?

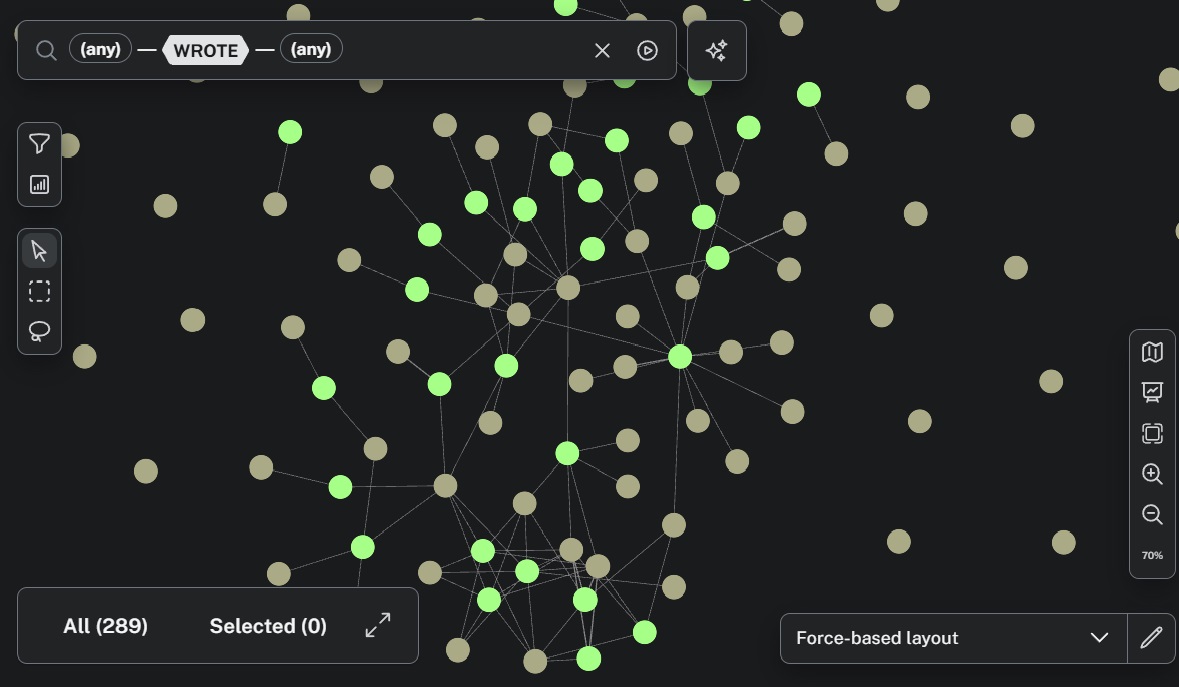
Generative AI’ın bazı ciddi riskleri ve etik tartışmaları olduğunu söyleyebiliriz. Çünkü herhangi bir kontrol mekanizmasına bağlı olmadan yanıltıcı bilgi, telif hakkı ihlalleri, nefret söylemleri, normal şartlarda erişilmesi mümkün olmayan yasa dışı bilgi aktarımı gibi olumsuz potansiyel içeren durumları mevcuttur. Ayrıca insan emeğinin yerini alması, özel alan ihlali gibi konularda da etik anlamda eleştiri almakta olduğunu söyleyebiliriz. Bu nedenle bu teknolojinin şeffaf, güvenli ve sorumlu bir şekilde kullanılması gerektiğini düşünüyorum.

**BÖLÜM 2**

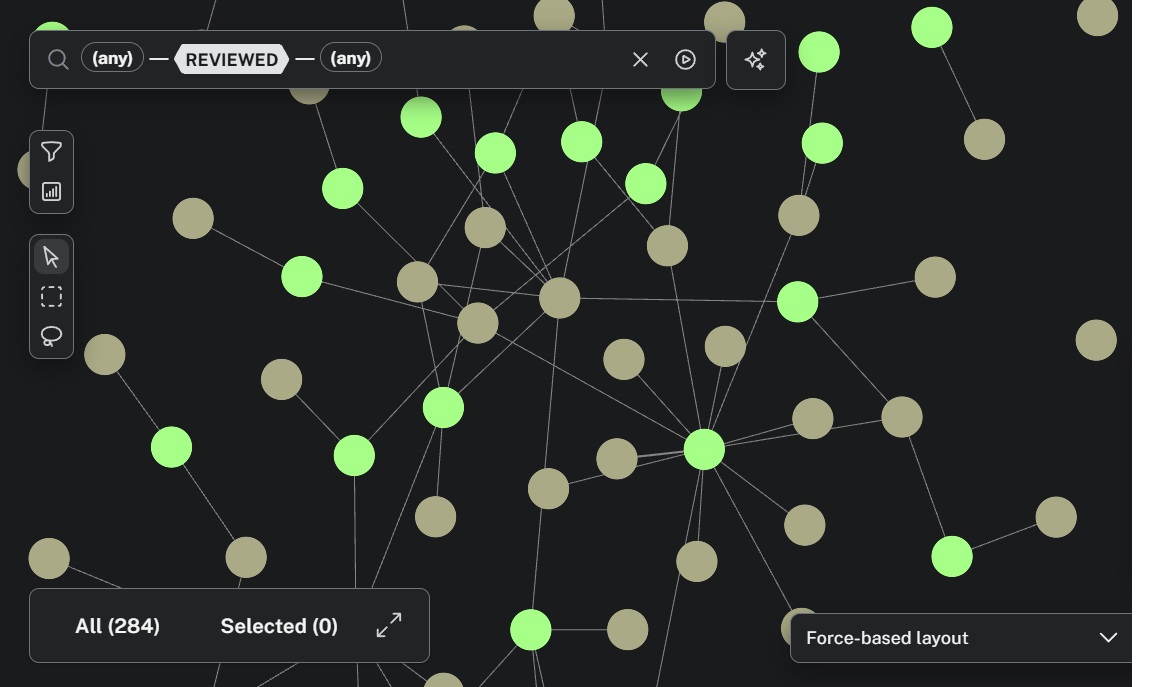
Neo4j Aura’da önce bize verilen movies datasetini importladım. Ardından explore tool’u ile aralarındaki ilişkileri keyword olarak kullanarak birkaç sorgu gerçekleştirdim. Örnek sorgular aşağıdaki resimler şeklinde oldu. Node’lar arasındaki ilişkinin her iki yöne doğru da olabildiğini, one-to-many, many-to-one, many-to-many ilişkiler kurulabildiğini de bu şekilde görmüş oldum.

Keywordlerle yaptığım Explorer kısmı sorgularında:

kişilerin birbirini takip ettiği [:FOLLOWS] relation,

kişilerin yazmış olduğu filmler [:WROTE] relation, 

ve kişilerin incelemiş olduğu filmler [:REVIEWED] relation için sorgu sonuçları:



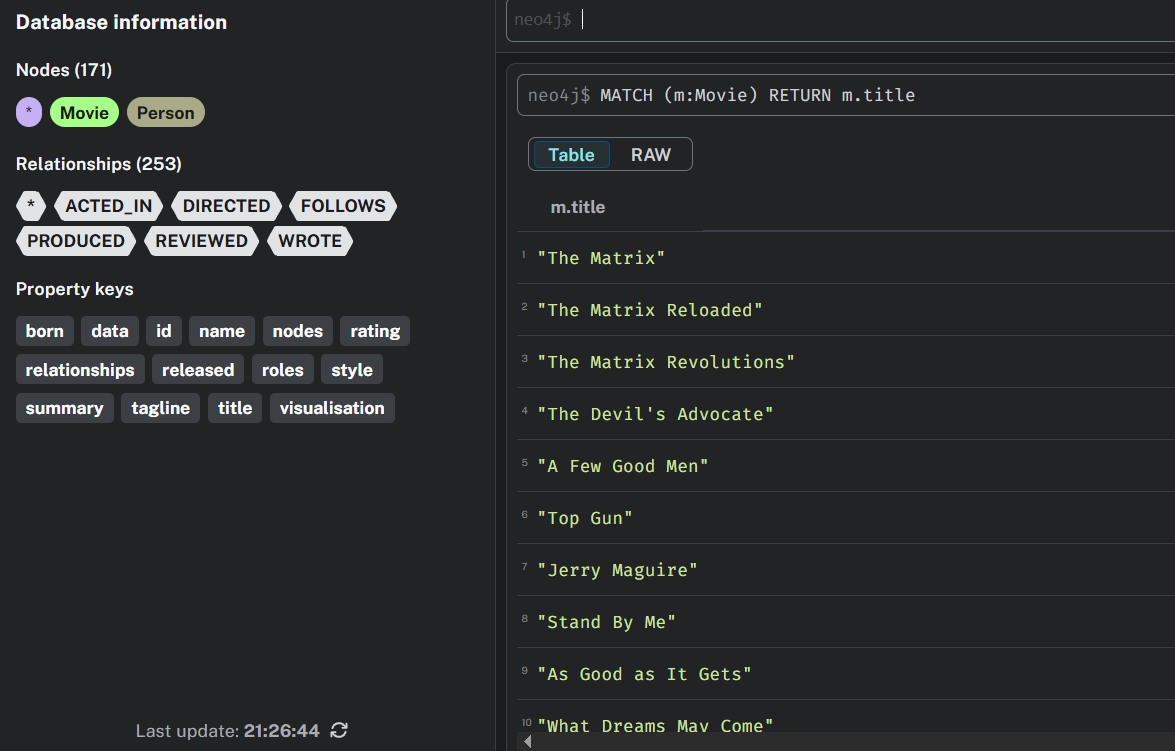
**BÖLÜM 3**

1. Veritabanındaki tüm film adlarını listeleyin.

**MATCH (m:Movie)**

**RETURN m.title**

*--Tüm nodelar içinden label’ı Movie olanları getirip title property’leri listeledim.*

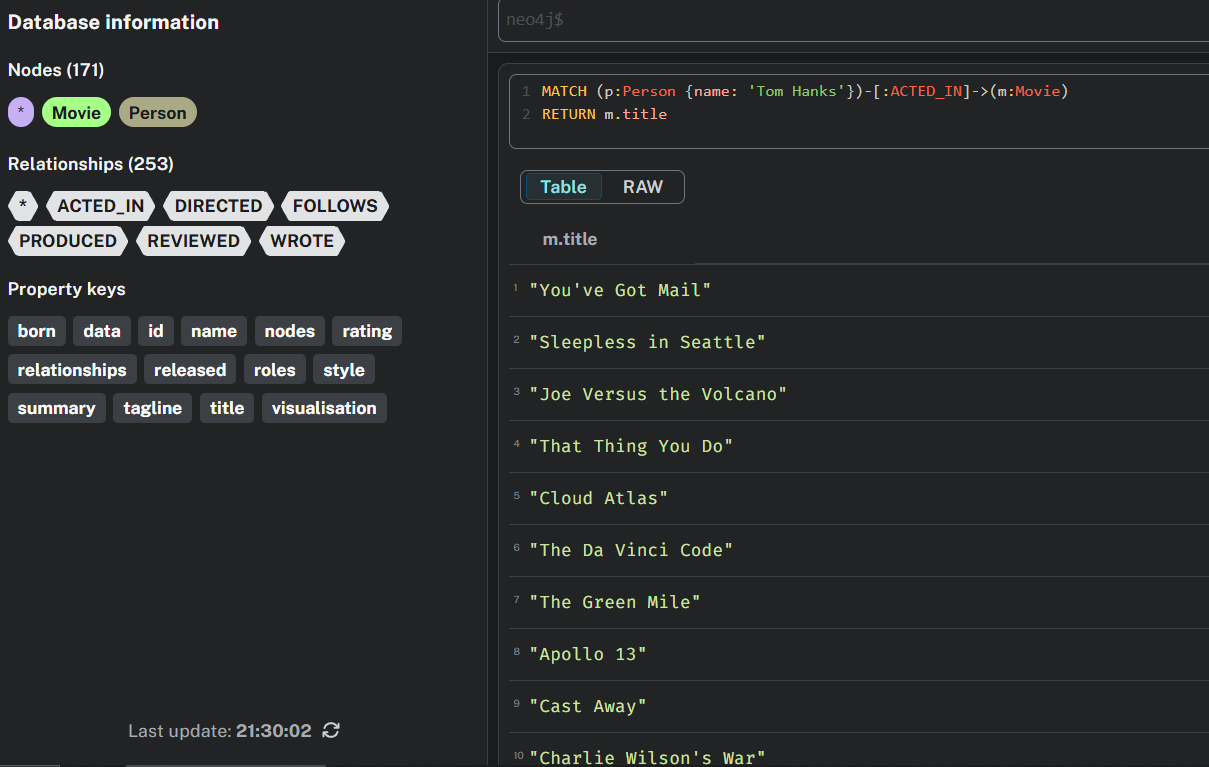
**

1. “Tom Hanks”in oynadığı filmleri bulun.

**MATCH (p:Person {name: 'Tom Hanks'})-[:ACTED\_IN]->(m:Movie)**

**RETURN m.title**

*--İsmi Tom Hanks olan kişinin oynamış olduğu filmleri ACTED\_IN relation’ı ile bulup title property’lerini listeledim.*

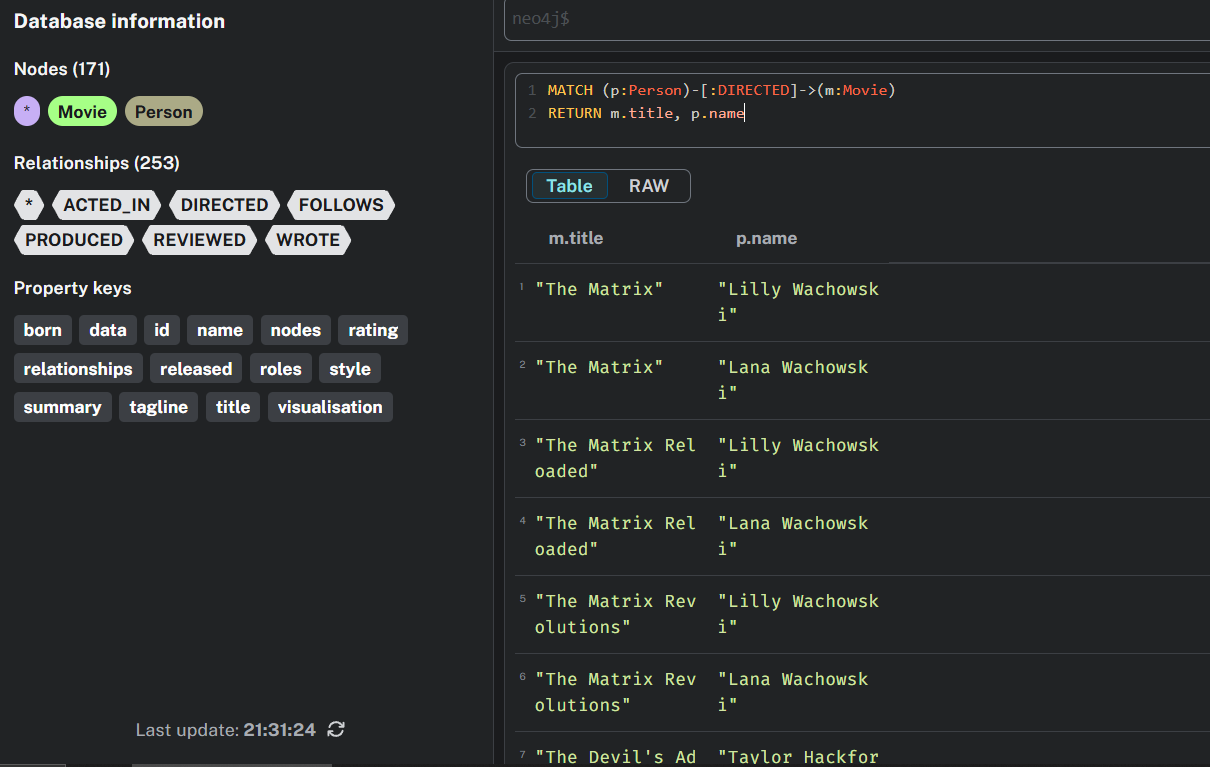
**

1. Her film için yönetmenlerini listeleyin.

**MATCH (p:Person)-[:DIRECTED]->(m:Movie)**

**RETURN m.title, p.name**

*--Person nodeları ile Movie nodeları arasındaki DIRECTED relation’ı kullanıp film isimlerini ve kişi isimlerini listeledim.*

****

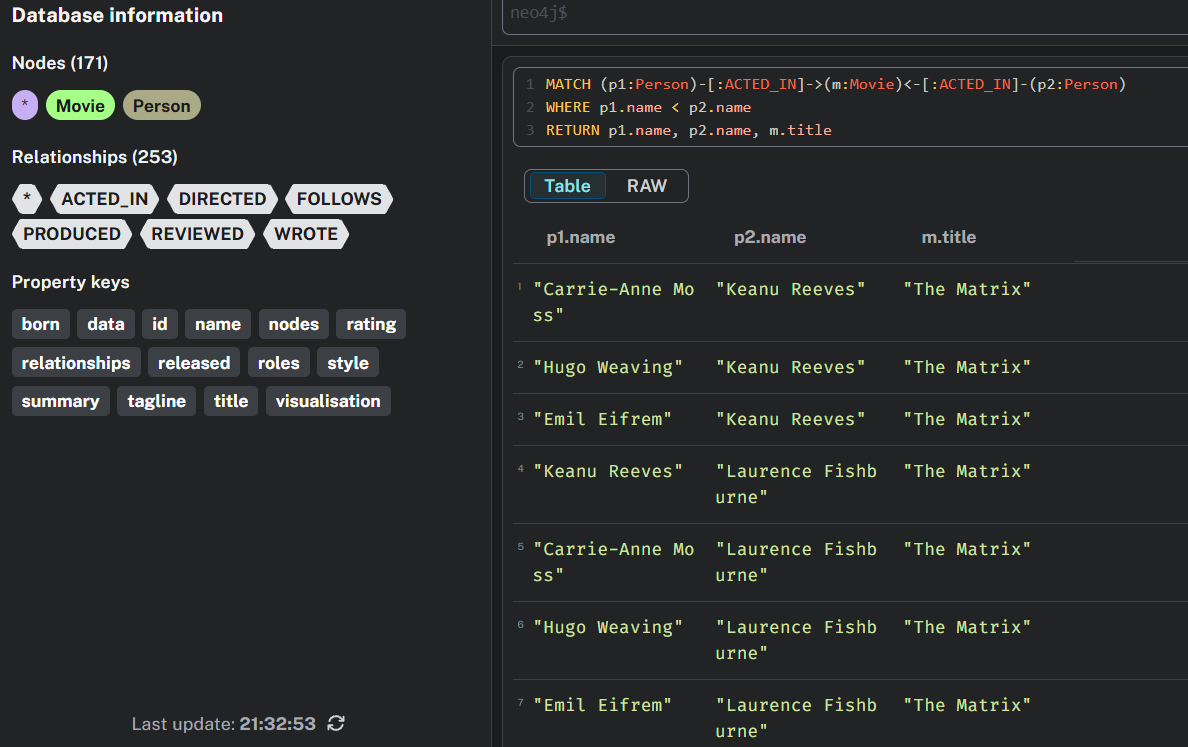
1. Aynı filmde oynamış iki farklı oyuncu çiftini listeleyin (tekrarsız).

**MATCH (p1:Person)-[:ACTED\_IN]->(m:Movie)<-[:ACTED\_IN]-(p2:Person)**

**WHERE p1.name < p2.name**

**RETURN p1.name, p2.name, m.title**

*--Aynı filme ACTED\_IN relation’ı ile bağlı kişileri tespit edip isimleri ve film adını listeledim. Fakat Ahmet, Mehmet ve Mehmet, Ahmet gibi tekrar olmaması için bir koşul ekledim.*

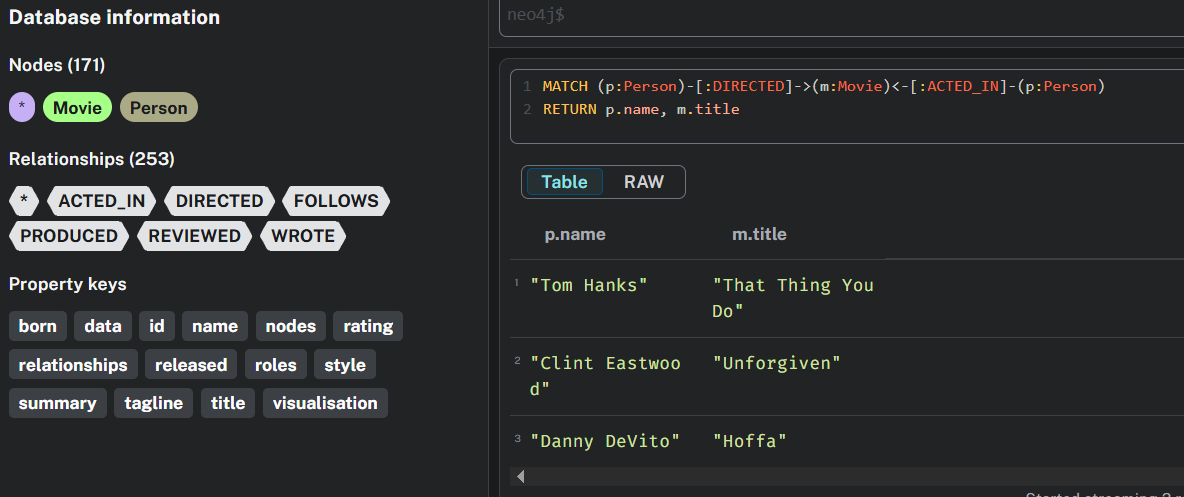
****

1. Hem yönetip hem oynadığı bir film olan kişileri ve filmleri bulun.

**MATCH (p:Person)-[:DIRECTED]->(m:Movie)<-[:ACTED\_IN]-(p:Person)**

**RETURN p.name, m.title**

*--Hem yönetip hem oynadığı film olan kişileri bulmak için DIRECTED ve ACTED\_IN relation’larının işaret ettiği kişileri listeledim.*

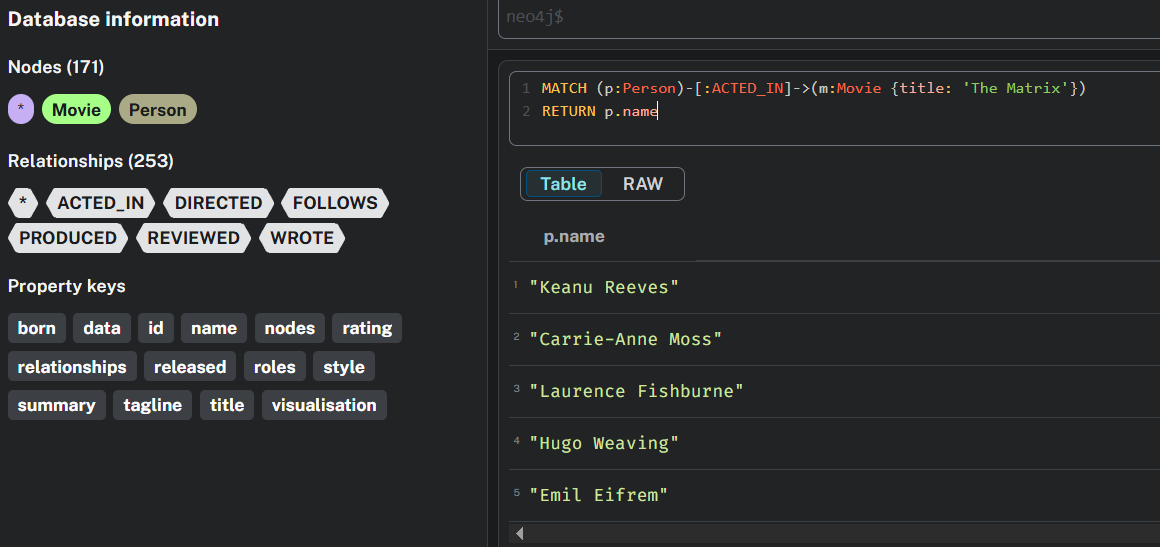


1. “The Matrix” filminde oynamış oyuncuları listeleyin.

**MATCH (p:Person)-[:ACTED\_IN]->(m:Movie {title: 'The Matrix'})**

**RETURN p.name**

*--İsmi “The Matrix” olan filmde oynayan kişileri ACTED\_IN relation’ıyla seçip listeledim.*

****

1. En çok filmde oynamış 5 kişiyi bulun.

**MATCH (p:Person)-[:ACTED\_IN]->(m:Movie)**

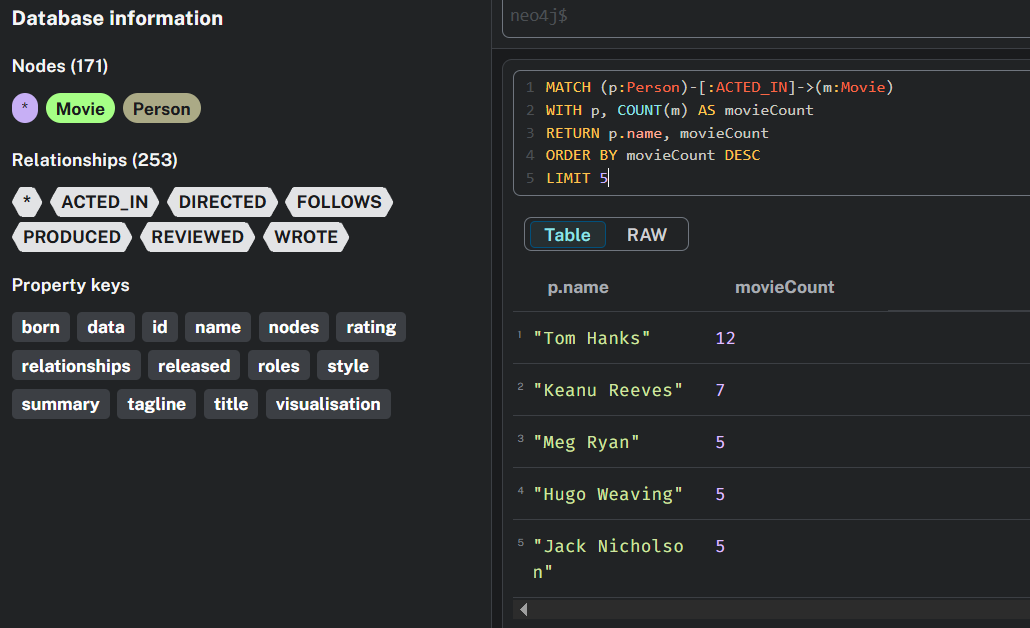
**WITH p, COUNT(m) AS movieCount**

**RETURN p.name, movieCount**

**ORDER BY movieCount DESC**

**LIMIT 5**

*--Öncelikle WITH keyword’ü ve ACTED\_IN relation’ıyla filmlerde oynayan kişilerin kaç filmde oynadıklarını COUNT fonksiyonu sayesinde movieCount olarak buldum, sonrasında bu sayıları büyükten küçüğe sıraladım ve 5 tane olarak limitledim.*

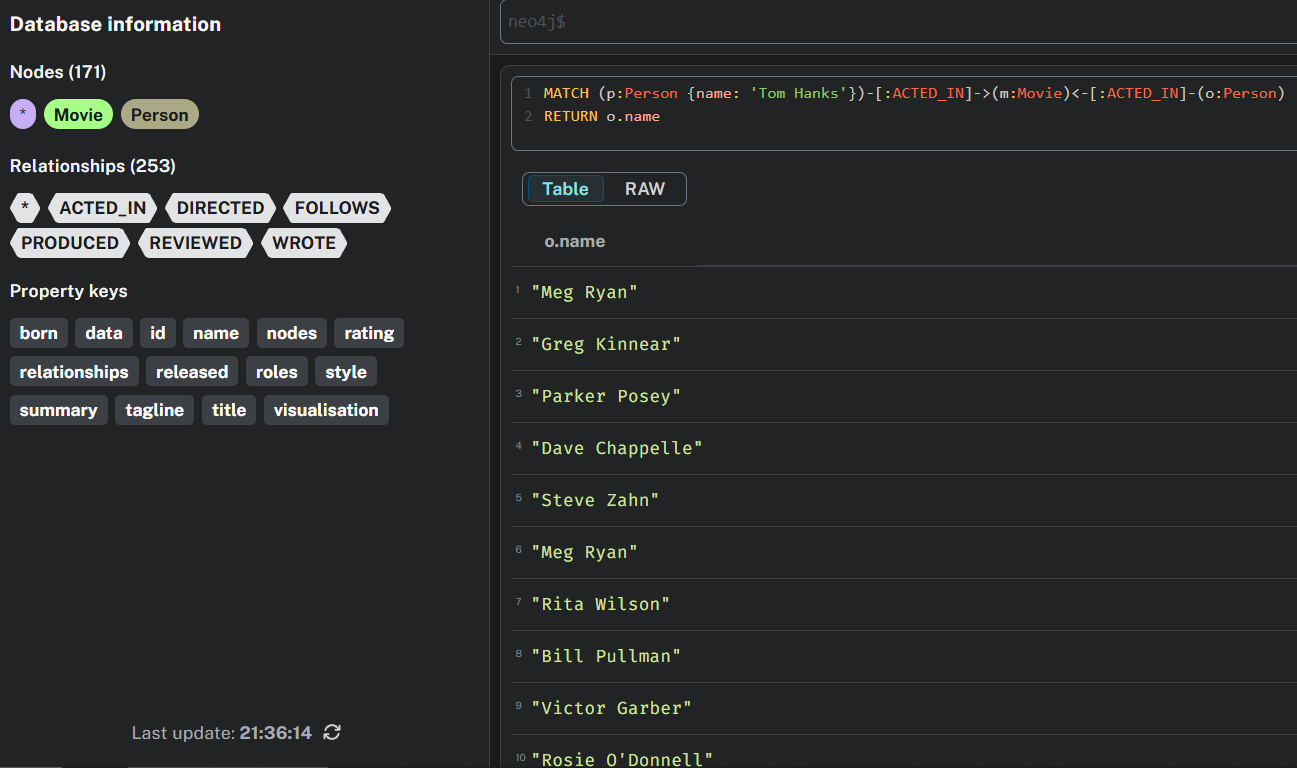
****

1. “Tom Hanks” ile aynı filmde oynamış diğer oyuncuları listeleyin.

**MATCH (p:Person {name: 'Tom Hanks'})-[:ACTED\_IN]->(m:Movie)<-[:ACTED\_IN]-(o:Person)**

**RETURN o.name**

*--Tom Hanks’in oynadığı filmlere ACTED\_IN relation’ıyla bağlı kişileri listeledim.*

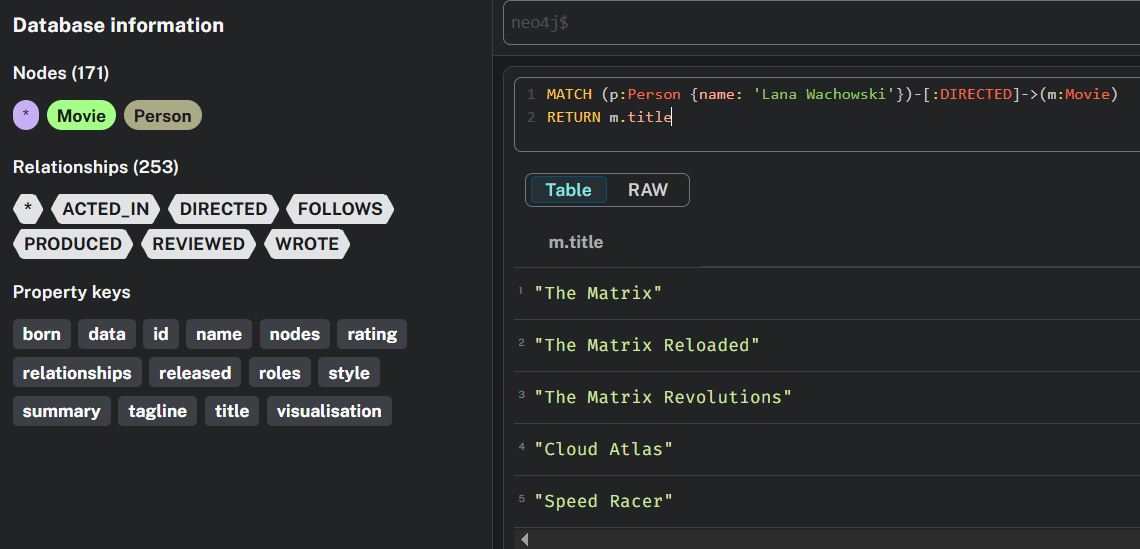
****

1. Yönetmeni “Lana Wachowski” olan filmleri bulun.

**MATCH (p:Person {name: 'Lana Wachowski'})-[:DIRECTED]->(m:Movie)**

**RETURN m.title**

*--İsmi Lana Wachowski olan kişinin yönettiği filmleri DIRECTED relation’ıyla listeledim.*

****

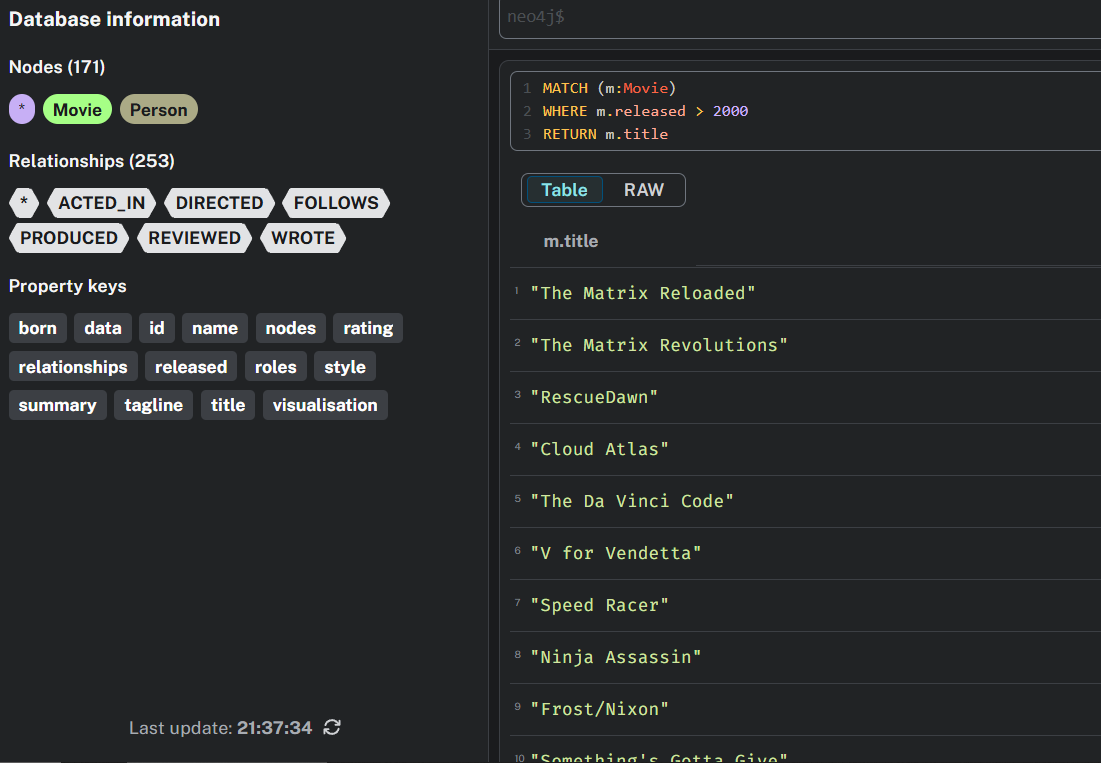
1. 2000 yılından sonra yayınlanmış filmleri listeleyin.

**MATCH (m:Movie)**

**WHERE m.released > 2000**

**RETURN m.title**

*--Tüm filmleri WHERE koşuluyla yayınlanma yılı 2000’den büyükleri alacak şekilde listeledim.*

**