# ÖDEV 1:

## *Bölüm 1:*

Generative AI (Üretken Yapay Zeka) , eğitim aldığı verilerden öğrendiği kalıplar aracılığıyla yeni ve özgün içerikler üretebilen yapay zeka sistemleridir. GenAI kullanarak metin, görsel, ses, video veya kod gibi içerikleri yaratabiliriz. En başta görsel, ses veya video üreten modeller genellikle alana özgü verilerle eğitilmiş ve çıktıları da yine bu modalitede olmuştur. Örneğin LLM'ler metin verisiyle eğitilmiştir ve çıktısıda en başta sadece metindi. Ancak son yıllarda gelişen modeller birden fazla modaliteyle eğitilmiştir. Dolayısıyla girdi ve çıktı türü farklı olabiliyor. Bu tür modellere multi-modal (çoklu modalite) deniyor. Örneğin artık hepimizin bildiği ChatGPT'nin, multi-modal versiyonunu deneyimleyebiliyoruz. Yazı türündeki girdimizden görüntü oluşturabiliyoruz. Runway Gen-2, Sora (OpenAI) gibi modellerle metinden video üretebiliyoruz.

Generative AI birçok alanda kullanılmaktadır. Sağlık alanında hasta verilerinden klinik özetler, tanı destek sistemi ve medikal görsel oluşturulabilir. Eğitim alanında kişileştirilmiş öğrenme içerikleri konusu en dikkat çeken alanlarda oldu. Artık finanstan, psikolojiye, tasarımdan, bilime bir çok alanda kullanılan ve kullanılması gerekilen bir noktada. GenAI kullanarak kişiselleştirilmiş oyun içi karakterler oluşturabilir ve ardından NFT ile bu karakterlerin dijital genetik satışını yapabiliriz. GenAI ile zaten varolan(yapay zeka ile gelişimi oldukça hızlanan) birçok teknolojininde Web3, kuantum şifreleme gibi hayatımızda rolünü arttıracağını düşünmekteyim. Çünkü Web3 algısına yakın son ürünler ve beraberinde getirdiği güvenlik sorunları bu dönüşümü tetikleyecektir diye düşünüyorum.

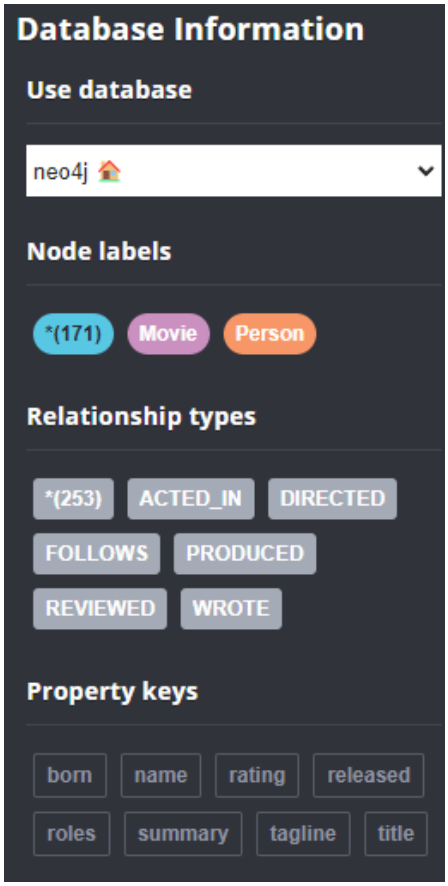
Grafik veritabanları, veriler arasındaki ilişkileri etkili bir şekilde modelleyerek, veritabanlarındaki bilgiye hızlı erişimi sağlar. RAG, bu bilgiye dayanarak, belirli bir sorguya veya ihtiyaca yönelik anlamlı metinleri ve içerikleri dinamik olarak üretebilir. Örneğin, bir e-ticaret platformunda ürünler ve kullanıcılar arasındaki ilişkileri içeren bir grafik veritabanı olduğunu düşünelim. Bu grafik veritabanı, ürünlerin kullanıcı yorumları, satın alma geçmişi, benzer ürünlerle etkileşimler gibi bilgileri içerir. RAG modeli, grafik veritabanındaki kullanıcı ve ürün arasındaki etkileşimleri analiz ederek, kullanıcıya en uygun ürünleri seçmek için bu verilerden anlamlı bilgi toplar ve metin tabanlı içerik oluşturur (örneğin, "Bu ürünleri sevebilirsiniz çünkü..."). Bu şekilde, grafik veritabanındaki ilişkiler, RAG'in kullanıcıya özgü ve alakalı içerik üretmesini sağlar, böylece kullanıcı deneyimi iyileştirilmiş olur. Ayrıca, platformda her bir ürün için farklı kullanıcı gruplarının yorumları ve etkileşim geçmişi RAG tarafından sorgulanarak, kullanıcıların ilgisini çekecek daha özelleştirilmiş içerikler önerilebilir. Hem halüsinasyon sorununa çözüm olarak kullanılmış oluyor hem de daha derin ilişkiler kurarak daha zeki bir sistem oluşturmamızıda sağlayabiliyor.

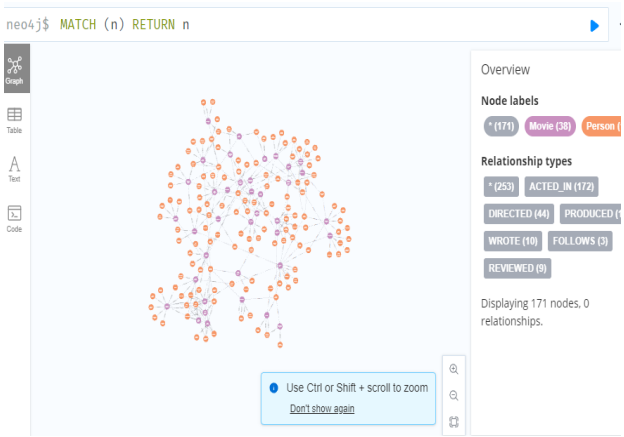
Riskleri ve etik davranışları GenAI'ın her adımında sorgulandı. Hatta yapay zeka süreci 1950 civarlarına kadar uzanmaktadır ve bu yüzden varlığı beynimizde yeni değildir. Ve henüz bu güncel ilerlemeler kaydedilmeden öncede sanat -özellike sinema- sektörüyle etik ve riskleri üzerinde sürekli durulan bir konu oldu. Yapay zeka konusunu işleyen neredeyse tüm dizi ve filmleri izlediğimi düşünüyorum. Ve neredeyse hepsinde kaotik bir sonla düşmanca bir yaklaşım var. Ben bir gün robotlar bizi yok edecek korkusu yaşamıyorum. Yok olma riskine odaklansam aklıma gelen ilk teknoloji nükleer olurdu. Önlem alınabilir risklerine gelirsek özellikle banka sektöründe çok olacağını düşünmekteyim fakat bu durumlar için banka sektöründe kuantum şifreleme üzerine gelişen özellikle Hollanda merkezli bir çok gelişmelerde beraberinde ilerliyor. Aynı zamanda hukuksal alanda da birçok riskli örnek bulunmaktadır akıllı araçların kazası, kişisel kimlik kullanılması gibi durumlarda da riskler bulunmaktadır. Yapay zeka teknolojisi öncelikle insan tarafından kullanılacağı bağlama göre etik olmama sorunları doğurabilir. Bu kullanımlar benim gözümde kişinin kendi kişisel tercileri değildir. Tercihlerinin sonucu başkasına zarar veriyorsa etik olmayacaktır.

*Bölüm 2:*

## Adım 1: Movies Veri Setini Yükleyin

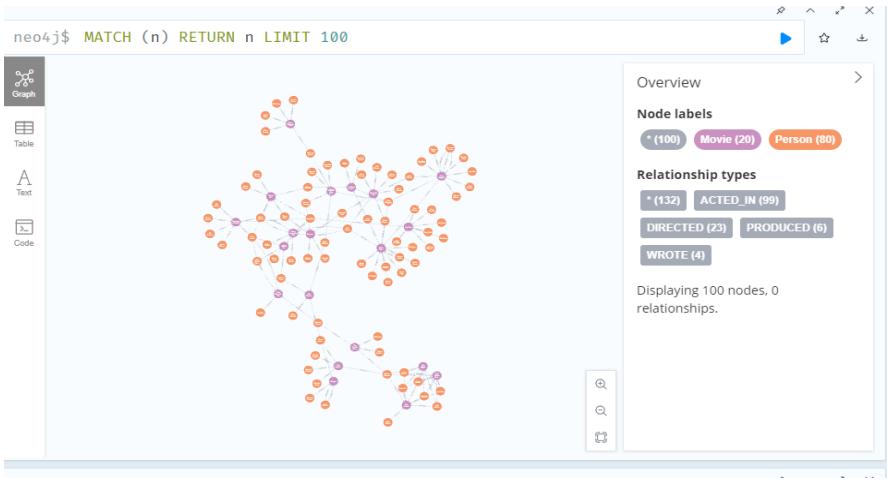
```
neo4j$
$ CREATE CONSTRAINT movie_title IF NOT EXISTS FOR (m:Movie) REQUIRE m.title IS…
neo4j$ MERGE (FrostNixon:Movie {title:'Frost/Nixon'}) ON CREATE SET FrostNix…
neo4j$ MERGE (Hoffa:Movie {title:'Hoffa'}) ON CREATE SET Hoffa.released=1992…
neo4j$ MERGE (Apollo13:Movie {title:'Apollo 13'}) ON CREATE SET Apollo13.rel…
neo4j$ MERGE (Twister:Movie {title:'Twister'}) ON CREATE SET Twister.release…
neo4j$ MERGE (CastAway:Movie {title:'Cast Away'}) ON CREATE SET CastAway.rel…
neo4j$ MERGE (OneFlewOvertheCuckoosNest:Movie {title:'One Flew Over the Cuck…
neo4j$ MERGE (SomethingsGottaGive:Movie {title:'Something\'s Gotta Give'}) O…
neo4j$ MERGE (BicentennialMan:Movie {title:'Bicentennial Man'}) ON CREATE SE…
neo4j$ MERGE (CharlieWilsonsWar:Movie {title:'Charlie Wilson\'s War'}) ON CR…
neo4j$ MERGE (ThePolarExpress:Movie {title:'The Polar Express'}) ON CREATE S…
neo4j$ MERGE (ALeagueofTheirOwn:Movie {title:'A League of Their Own'}) ON CR…
neo4j$ MATCH (CloudAtlas:Movie {title:'Cloud Atlas'}) MATCH (TheReplacements…
```



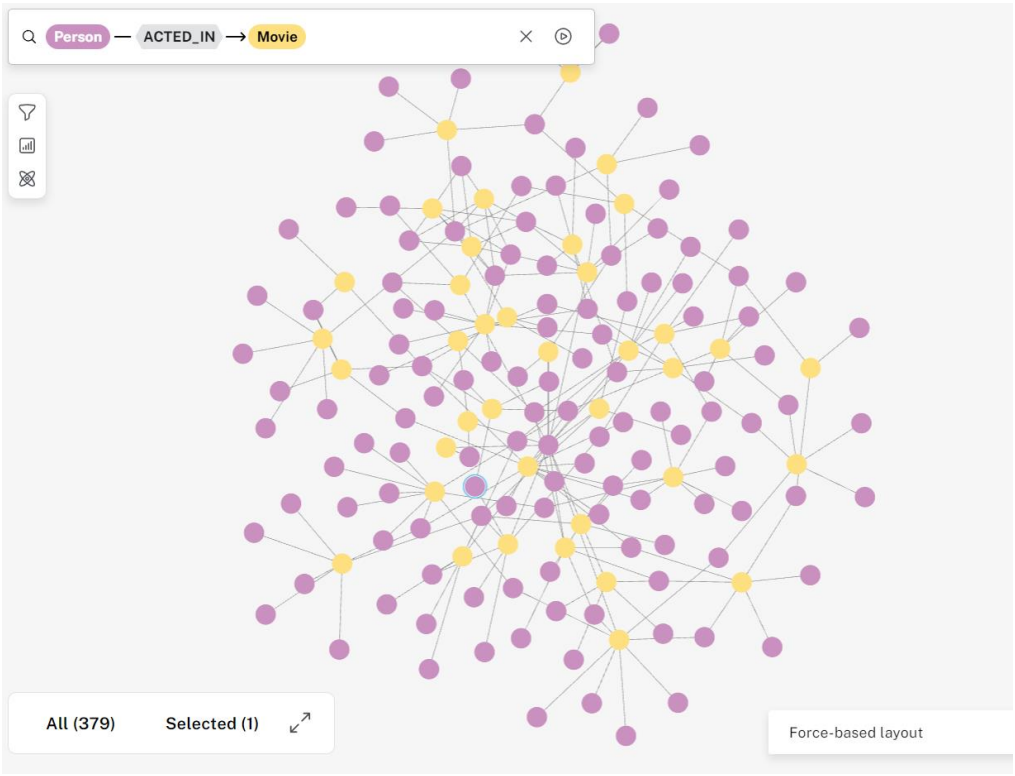




## Adım 2: Explore Sekmesinde Veriyi Keşfedin

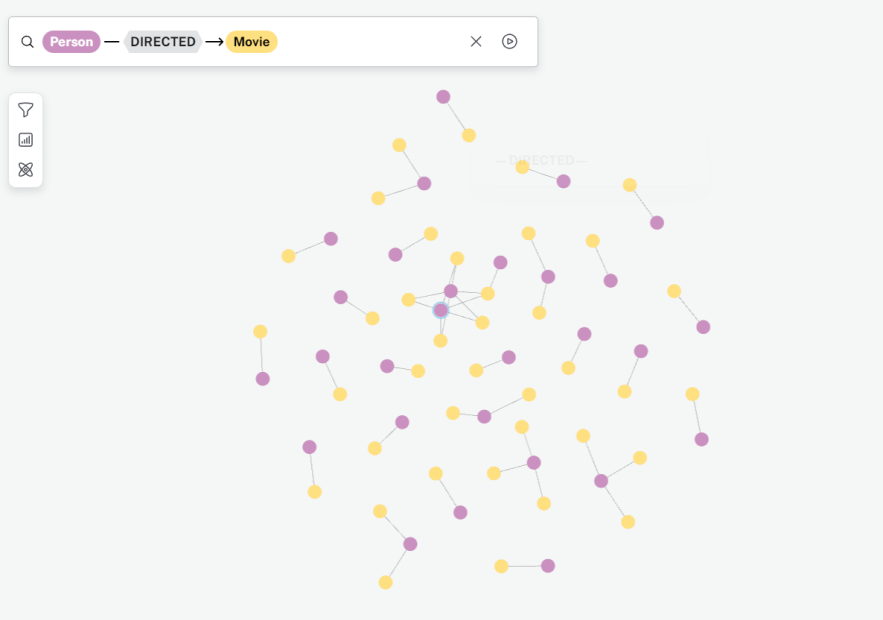
- MATCH (n) RETURN n LIMIT 100 -> Bu komut, veritabanında herhangi bir veri olup olmadığını görmek, veri yüklendi mi diye kontrol etmek için kullanılır. Özellikle ilk testlerde çok kullanışlıdır.



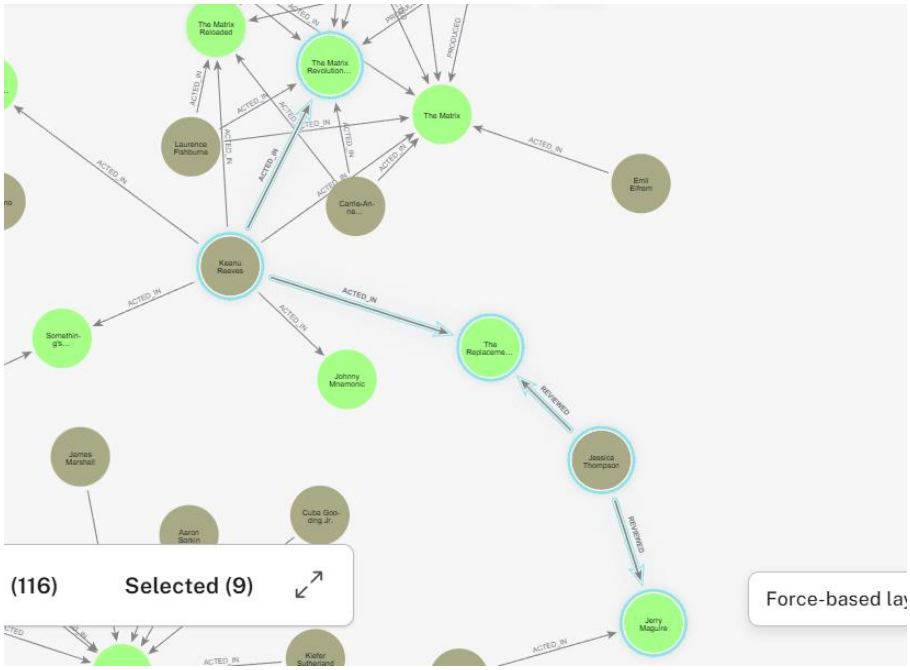

- Oyuncuları ve oynadıkları filmleri inceleyin:



- Filmleri yönetenleri bulun:




- Desktop' ta  Neo4j Bloom ile Shortest Path'e Ctrl kısa yoluyla 2 node seçip ulaştım . Ayrıca online Aura'da da pratik açısından çalıştım.

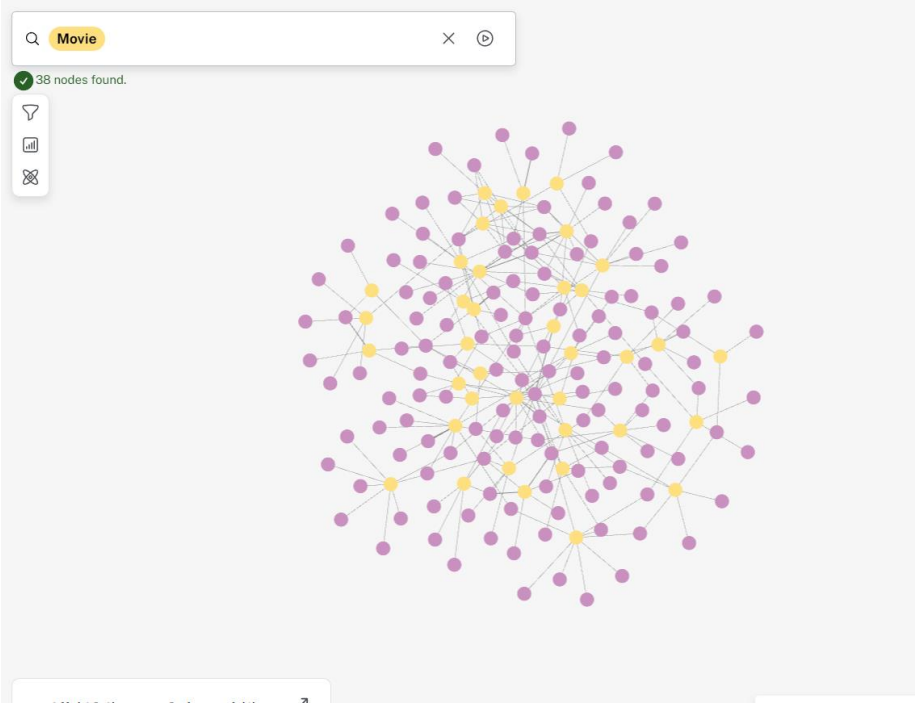



## Adım 3: Arama Alanıyla Özgün Sorgular Çalıştırın

- Person yazın ve Tab + Enter tuşlarıyla tüm kişileri getirip inceleyin:

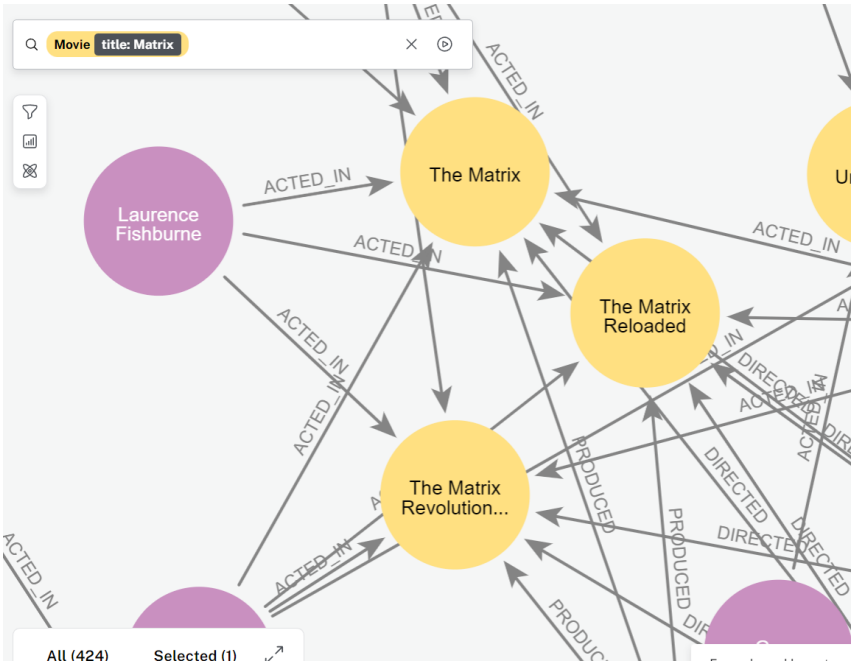

Burada person nodenun bağlı olduğu tüm ilişkileri görmüş olduk. Dolayısıyla tüm filmlerde gelmiş oldu.

- Movie yazın ve tüm filmleri görüntüleyin:

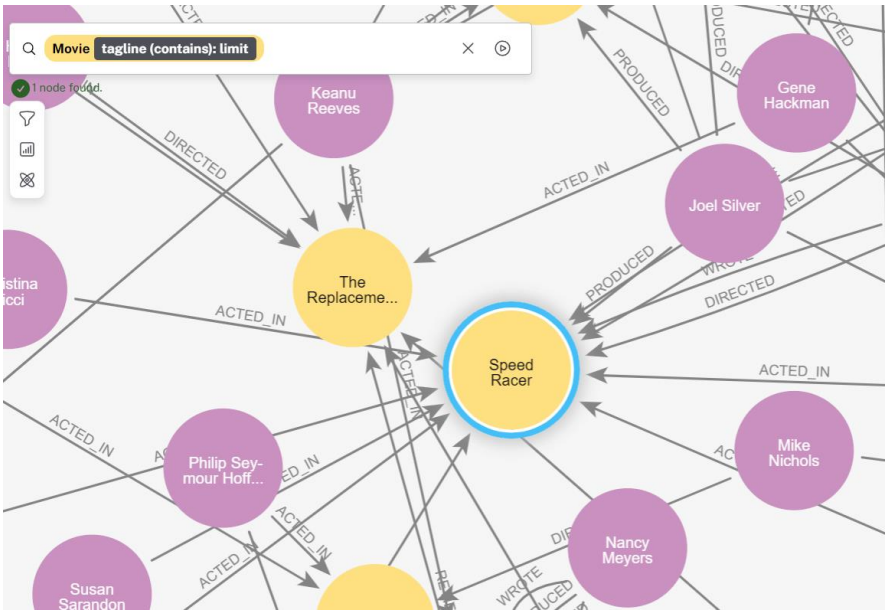



Burada movie nodelarını görüyoruz. Teknik olarak bu dataset için yine tüm personlarında gelmesi demek oldu.  Burada bize 23 node bulundu olarak çıkarıyor. Direkt Movie sayısını almış olduk. Person tab tab yaptığımızda ise 171 node bulundu yazdı çünkü orada persona ait tüm ilişkileri istediğimiz için ek olarak ilişki kurduğu node sayısınıda verdi.

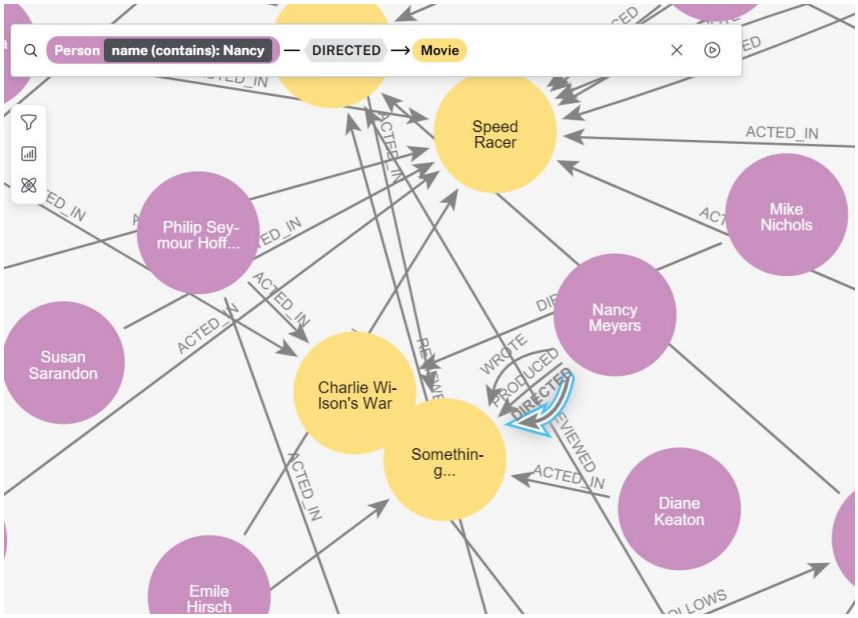
- Category, Director, Actor, Matrix gibi anahtar kelimelerle arama yaparak örnek alt grafikler oluşturun.



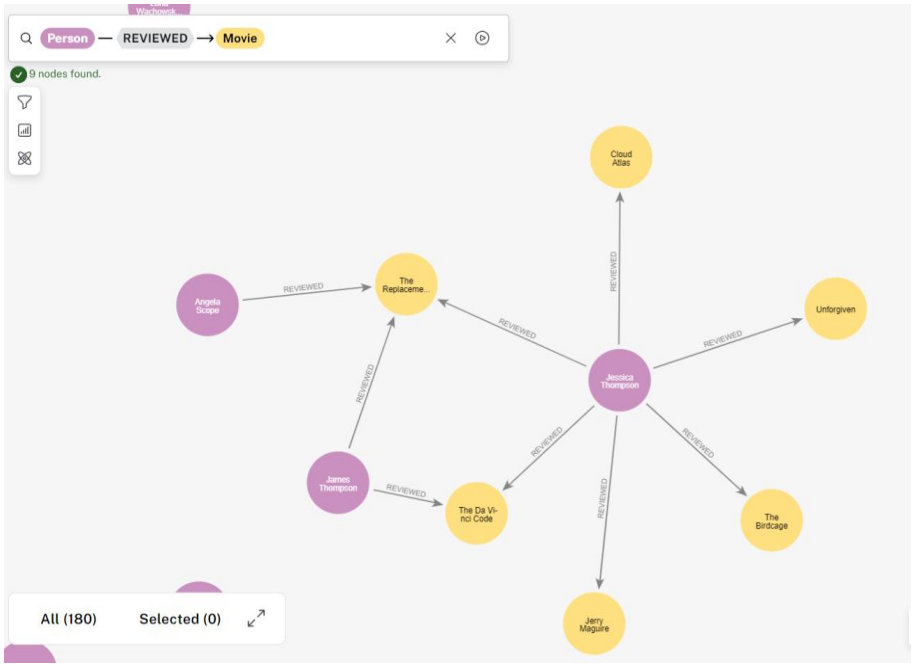

Burada Movie node'unda başlığı The Matrix olanları istedim. Bu tarz aramalarda bütünün içinden yakınlaştırma yaparak sonucu çıkardığını gördüm .

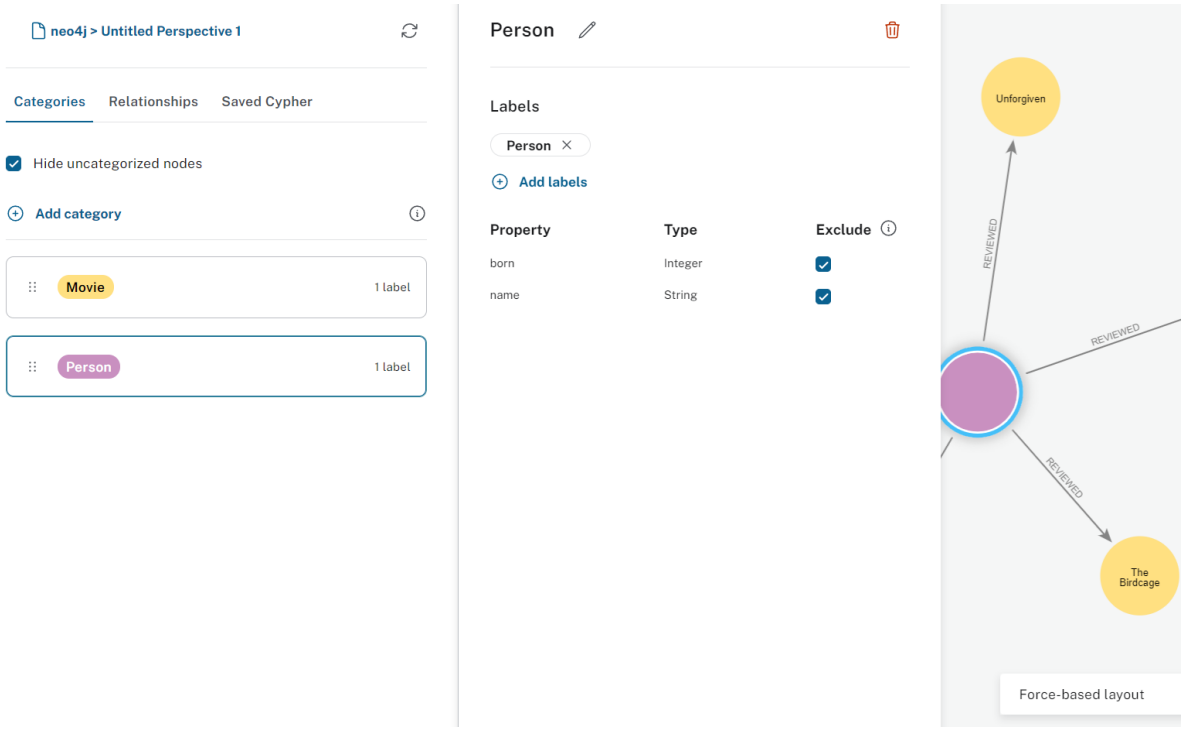



Bu arammada Movie node una ai özelliklerden biri olarak verilmiş tagline da 'limit' içerenleri şeçtim. Ayrıca sayısal değerlerde örneğin Movie yayınlama tarihinde de özelleştirilmiş aramalrı farkettim. Less than ,equal gibi.




Burada da Person adında ‘Nancy’ içeren node lardan yönettiği filmi almış oldum . Containste küçüş harf-büyük harf duyarlılığı var olduğunuda görmül oldum. (case-sensitive)






**Burada da gözükmesini istemediğimiz özellikleri gizleyebiliyoruz. Şekilde arkada mor olan person node unda görüldüğü üzere artık ismi yazmamakta.**

## *Bölüm 3: Cypher Sorguları ile Veriyi Keşfetme*

1. Veritabanındaki tüm film adlarını listeleyin.

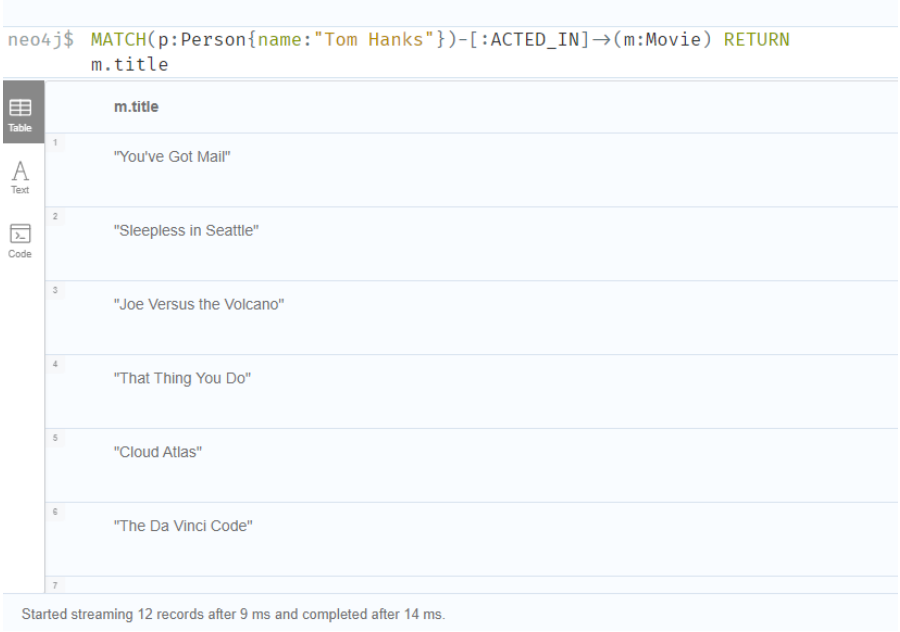
```
neo4j$ MATCH(m:Movie) RETURN m.title
```

| | m.title |
|---|---|
| 1 | "The Matrix" |
| 2 | "The Matrix Reloaded" |
| 3 | "The Matrix Revolutions" |
| 4 | "The Devil's Advocate" |
| 5 | "A Few Good Men" |
| 6 | "Top Gun" |
| 7 | |

Started streaming 38 records after 3 ms and completed after 4 ms.

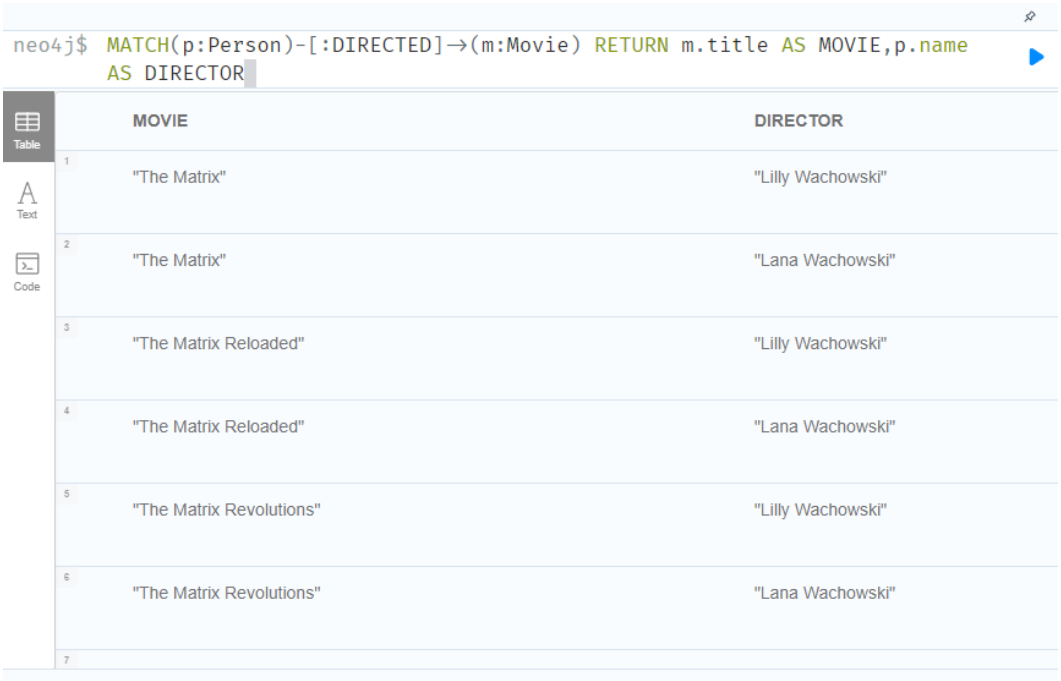
Bu sorguda veritabanındaki tüm Movie etiketindeki düğümlere git ve her birini m ile temsil et diyoruz. Daha sonrasında RETURN ile her m düğümünün başlığını döndürüyoruz.

2. "Tom Hanks"'in oynadığı filmleri bulun.




Bu sorguda node-relationship-node baglantısı kuruyoruz önce. Yani ismi Tom Hanks olan person düğümünün oynadığı filmler diyoruz. Ve bu filmlerinde adını döndürüyoruz daha sonra.

3. Her film için yönetmenlerini listeleyin.




Insanlardan filmi yönetenleri seç ve bu filmlerin adını ve ymnetmenlerin adını getir. AS ile de yönetmenlerin adı olan sütunu DIRECTOR ile göster film adlarının olduğu sütunun başlığını MOVIE ile göster

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

```
neo4j$ MATCH(p1:Person)-[:ACTED_IN]→(m:Movie)←[:ACTED_IN]-(p2:Person)
       WHERE p1.name < p2.name RETURN p1.name AS Actor1,p2.name AS
       Actor2,m.title AS Movie ORDER BY Movie
```

Table | Text | Code

| | Actor1 | Actor2 | Movie |
|---|---|---|---|
| 1 | "Jack Nicholson" | "Tom Cruise" | "A Few Good Men" |
| 2 | "Demi Moore" | "Tom Cruise" | "A Few Good Men" |
| 3 | "Kevin Bacon" | "Tom Cruise" | "A Few Good Men" |
| 4 | "Kiefer Sutherland" | "Tom Cruise" | "A Few Good Men" |
| 5 | "Noah Wyle" | "Tom Cruise" | "A Few Good Men" |
| 6 | "Cuba Gooding Jr." | "Tom Cruise" | "A Few Good Men" |
| 7 | | | |

Burada öncelikle p1 ve p1 2 person tanımladım ve aynı filmde rolma koşulu için acted ilişkisini tek bir filme yönlendirdim. Bu sorguda  A ve B aynı filmde oynadıysa tekrarsız olması için  WHERE p1.name< p2.name  ile filtreleme yaptım böylece A<B olduğu için p1=B p2=A durumunu yazmaz ve tekrarsız şekilde listemiş oldum. Order by ile sonuçları film adına göre alfabetitk sıraladım.

NOT:Eğer veri setimiz çok büyük olsaydı ve bazı fazladan eklenmiş bağlantılar olsaydı aynı zamanda birde DISTINCT ekleyebilirdim RETURN ederken . Fakat elimizdeki verisetinde gerek olmadı.

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

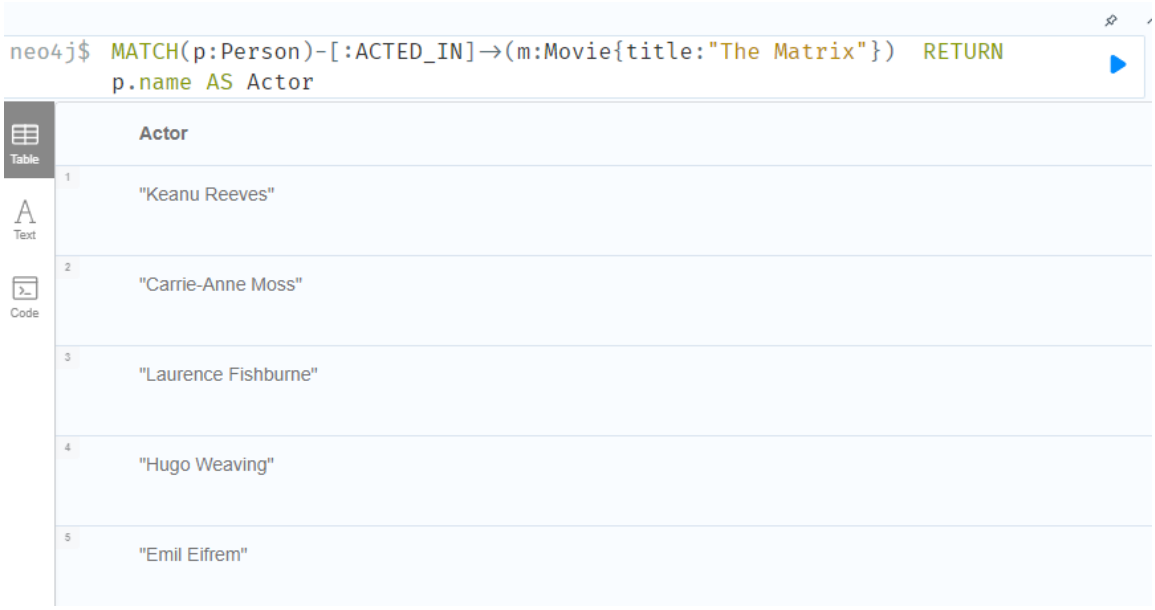
```
neo4j$ MATCH(p:Person)-[:DIRECTED]→(m:Movie)←[:ACTED_IN]-(p)  RETURN
       p.name AS Actor,m.title AS Movie
```

Table | Text | Code

| | Actor | Movie |
|---|---|---|
| 1 | "Tom Hanks" | "That Thing You Do" |
| 2 | "Clint Eastwood" | "Unforgiven" |
| 3 | "Danny DeVito" | "Hoffa" |

Burada tek bir person tanımladım p ile ve bu şekilde hem yönettiği hem oynadığı film aynı olması ilişkisini çekebildim . Ve daha sonrasında Return ile sonucu döndürdüm

6. "The Matrix" filminde oynamış oyuncuları listeleyin.




Bu sorguda filtrelemeyi MATCH içinde yaptım bu derstede öğrendiğimiz üzere daha verimli . Return ile sonucu döndürdüm.

7. En çok filmde oynamış 5 kişiyi bulun.

```
neo4j$ MATCH(p:Person)-[:ACTED_IN]→(m:Movie)  RETURN p.name AS Actor,
       COUNT(m) AS MovieCount ORDER BY MovieCount DESC LIMIT 5
```

| | Actor | MovieCount |
|---|---|---|
| 1 | "Tom Hanks" | 12 |
| 2 | "Keanu Reeves" | 7 |
| 3 | "Meg Ryan" | 5 |
| 4 | "Hugo Weaving" | 5 |
| 5 | "Jack Nicholson" | 5 |

Started streaming 5 records after 27 ms and completed after 32 ms.

MATCH ile filmde oynayan kişileri buldum . Return kişi adını ve  ile kaç filmde oynadığı sayısını aldım .  Daha sora FilmSayısı(Kaç filmde oynadığı AS (ile ifade ettiğim) sütununu azalan şekilde sıraladım (DESC-Descending) ve LIMIT ile ilk 5 satırı yazdırdım.

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

```
neo4j$
1 MATCH(p1:Person)-[:ACTED_IN]→(m:Movie)←[:ACTED_IN]-(p2:Person{name:'Tom
  Hanks'})
2 RETURN DISTINCT p1.name AS Actor
3 ORDER BY Actor
```

| | Actor |
|---|---|
| 1 | "Audrey Tautou" |
| 2 | "Bill Paxton" |
| 3 | "Bill Pullman" |
| 4 | "Bonnie Hunt" |
| 5 | "Charlize Theron" |
| 6 | "Dave Chappelle" |
| 7 | |

Burada da p1 ve p2=Tom Hanks olarak tanımladım ve RETURN de DISTINCT ekledim çünkü bazı kişilerle 2 farklı filmde oynadıysa tekrara düşüyor bunu engelledim. ORDER BY koymak zorunda değildim ama daha okunaklı olması adına uyguladım.

9. Yönetmeni "Lana Wachowski" olan filmleri bulun.

```
neo4j$
```

```
1 MATCH(p:Person{name:"Lana Wachowski"})-[:DIRECTED]→(m:Movie)
2 RETURN  m.title AS MOVIE
3 ORDER BY MOVIE
```

Table | Text | Code

| | MOVIE |
|---|---|
| 1 | "Cloud Atlas" |
| 2 | "Speed Racer" |
| 3 | "The Matrix" |
| 4 | "The Matrix Reloaded" |
| 5 | "The Matrix Revolutions" |

Burada da MATCH içinde person adını filtrelemiş oldum . Ve "Lana Wachowski" adında olan person düğümlerinden yönetmen olarak ilişki kurmuş bağlı olduğu film düğümlerini. Daha sonra Return ile döndürdüm. Order by yine okuma ve incelememde kolaylık adına kullandım.

10. 2000 yılından sonra yayınlanmış filmleri listeleyin.

```
1 MATCH(m:Movie)
2 WHERE m.released>2000
3 RETURN  m.title AS MOVIE,m.released AS Publication
4 ORDER BY m.released
```

Table | Text | Code

| | MOVIE | Publication |
|---|---|---|
| 1 | "The Matrix Reloaded" | 2003 |
| 2 | "The Matrix Revolutions" | 2003 |
| 3 | "Something's Gotta Give" | 2003 |
| 4 | "The Polar Express" | 2004 |
| 5 | "RescueDawn" | 2006 |
| 6 | "The Da Vinci Code" | 2006 |
| 7 | | |

Burada filtrelemeyi MATCH içinde yapamıyorum çünkü ancak direkt bir eşleşme olursa burada yaabilirim . Bu yüzden WHERE ile filtrleme yapıyorum . Daha sonra yine RETURN ile döndürüyorum . ORDER BY la da default ASC (Ascending) olduğu için artan sıralamayla çıktı almış oldum .