**NEO4J**

**Nedir?**

Bir Graph Database örneği olan **Neo4j** Java ve Scala dosyası yazılmış olan [Neo4J Technology](https://tr.wikipedia.org/w/index.php?title=Neo4J_Technology&action=edit&redlink=1) isimli firma tarafından açık kaynak olarak dağıtılan, yüksek performanslı ve ölçeklenebilen ve geliştirilen [graph](https://tr.wikipedia.org/w/index.php?title=Graph&action=edit&redlink=1) veritabanıdır. NASA, eBay, cisco gibi kuruluşlar tarafından kullanılmaktadır.

A black background with a black square

Description automatically generated with medium confidence

Geliştiricilerin yerel grafik depolama ve işleme özelliğine sahip ACID uyumlu bir işlem veri tabanı olarak tanımladığı Neo4j, DB-Engines sıralamasına göre en popüler graph tabanlı ve 22.en popüler genel veri tabanıdır.

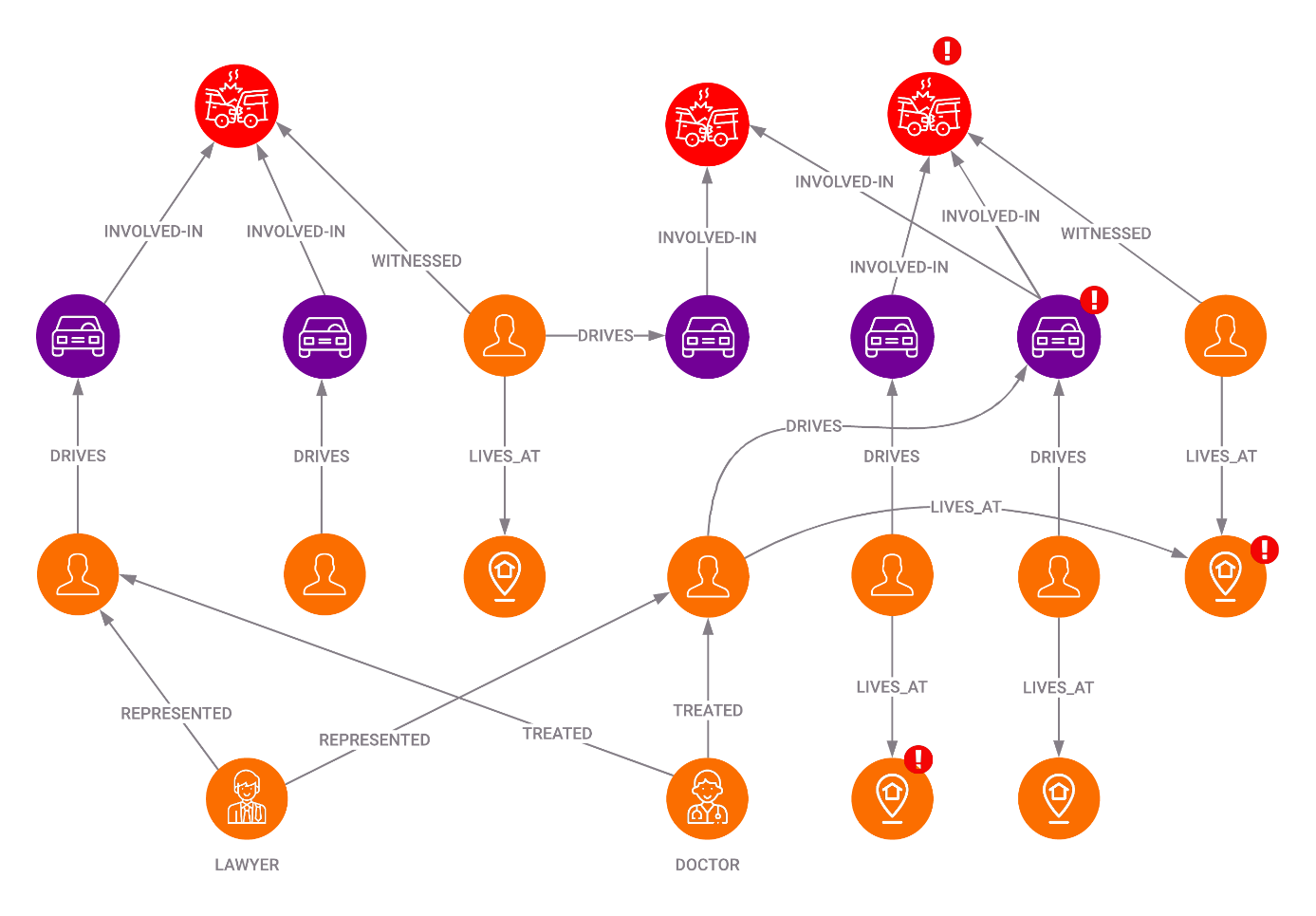
İlişkisel veritabanlarından farklı olarak veri arasındaki ilişkileri vurgulamak için graph teorisine kullanan, node (düğüm) ve ilişki (ilişkiler) olmak üzere iki temel unsurdan oluşan bir NoSql veritabanıdır. Daha çok graf yapısına göre çalıştığından dolağı ağ, harita, sosyal ağ gibi bir birine bağlı düğümlerin saklandığı ve işlendiği alanlarda kullanılmaktadır. Oracle NoSQL Database, OrientDB, HypherGraphDB, GraphBase, InfiniteGraph ve AllegroGraph gibi diğer popüler grafik veritabanları da vardır. Aşağıdaki görseldeki gbi düşünebiliriz.

A diagram of a person

Description automatically generated

Neo4j'de her şey düğüm, kenar veya nitelik biçiminde tutulur. Her bir düğüm ve kenar istenildiği kadar niteliğe sahip olabilir. Düğümler ve kenarlar etiketlenebilir ve etiketlenmesi arama daraltılması adına kullanışlı olmaktadır.

**Neden Graph Database?**

Yeterli geliştirme süresi ve işlem gücüyle, RDBMS ideal olarak uygun olmadığı birçok şeyi yapabilir. Grafik veritabanlarının aksine, geleneksel ilişkisel veritabanları veri kümeleri arasındaki ilişkileri yerel olarak depolamaz. Bunun yerine, RDBMS yalnızca verilerin kendisini depolar. Daha sonra yalnızca çalışma zamanında ilişkileri hesaplayabilir. Aynı bilgi basit bir grafik sorgusundan milisaniyeler içinde döndürülebildiğinde bu zaman alıcıdır ve pahalı bir işlemdir. Pratik olarak konuşursak, RDBMS birçok kullanım durumu için uygun değildir, oysa Neo4j ve diğer grafik veritabanları mükemmeldir. 

Giderek daha fazla birbirine bağlanan bir dünyada yaşıyoruz. Sonuç olarak, verilerimiz de daha fazla birbirine bağlanıyor. Her gün küresel olarak üretilen veri hacmi göz önüne alındığında, verinin içindeki ilişkilerin değeri, verinin kendisinden hızla daha değerli hale geliyor. Grafik veritabanlarının benzersiz değeri , işletmeler ve diğer kuruluşlar üzerinde önemli etkileri olan analitik içgörüler olarak, yerel olarak ve ölçekte yeni, birbiriyle bağlantılı bilgileri ortaya çıkarma yeteneğinden gelir.

**Neo4j'in (Grafik Veritabanı) Başlıca Avantajları**

1. **İlk hamle yapan**: Neo4j, piyasada diğer tüm çözümlerden daha yaygın olarak benimsenmiştir. Aslında, Neo4j Kurucusu  Emil [Efreim](https://www.linkedin.com/in/emileifrem/) "grafik veritabanı" [terimini ortaya atan kişidir .](https://neo4j.com/news/birth-graph-databases-neo4j-built-product-category/)
2. **Topluluk**: Neo4j, grafik veritabanıyla ilgili sorular için gelişen bir kullanıcı topluluğuna, aktif forumlara, derin dokümantasyona ve kaynaklara sahiptir.
3. **Performans**: Neo4j , büyük performans kazanımları için [endekssiz bitişikliğe](https://www.graphable.ai/blog/neo4j-performance/) olanak tanıyan çok az sayıdaki gerçek yerel grafik veritabanlarından biridir .
4. **Kullanılabilirlik**: Büyük gerçek zamanlı uygulamalardan analitik odaklı grafik ve grafik veri makine öğrenimi uygulamalarına kadar Neo4j, HA gereksinimlerini karşılamak için çıtayı belirler.
5. **ACID Uyumlu**: Neo4j'in hem okuma hem de yazma performansı, kurumsal düzeyde kanıtlanabilir bir şekilde ölçeklenmiştir. Ayrıca, diğer tekliflerin çoğunda hala eksik olan gerçek ACID uyumluluğuyla bütünlüğünü korumuştur.
6. **Kolay erişim**: Neo4j ile etkileşim kurmak kolaydır, ister Cypher sorgu diliyle Neo4j Tarayıcısı UI'si üzerinden (Gremlin gibi alternatiflere göre çok daha kolaydır) ister Java API'si üzerinden.
7. **Yapılandırılmamış / yarı yapılandırılmış / metinsel veriler**: Günümüzde mevcut olan kesinlikle büyük ve artan miktardaki yapılandırılmamış metinsel verilerden değer elde etmek hiç bu kadar kolay olmamıştı. Grafiklerin anlamı yerleştirme ve kavramları birbirine bağlama konusundaki benzersiz kapasitesi göz önüne alındığında, Neo4j içgörüyü yönlendirmek için ideal bir çözümdür. Ayrıca, Neo4j grafiklerle doğal dil işleme (NLP) alanında liderdir.
8. **Grafik veri bilimi**: Neo4j, NLP kullanım örnekleri de dahil olmak üzere grafiklerle veri biliminde ticari liderdir. Google'ın [veri biliminin geleceğinin](https://www.zdnet.com/article/google-ponders-the-shortcomings-of-machine-learning/) ağ teorisi / grafik veritabanları etrafında inşa edildiğini iddia ettiği göz önüne alındığında, bu Neo4j için önemli bir avantajdır.

**Hangi Amaçlarla Kullanılır?**

Neo4j çoğu kullanım durumu için kullanılabilirken, benzersiz değeri veriler içindeki bağlantıya odaklanan kullanım durumları için belirginleşir. En yaygın kullanım durumlarından bazıları şunlardır:

****

**Nasıl Kullanılır?**

**1- Kurulum:**

Neo4j, hem masaüstü uygulaması hem de bulut tabanlı olarak kullanılabilir. Masaüstü versiyonu <https://spring.io/guides/gs/accessing-data-neo4j> üzerinden indirilebilir.

* Bulut tabanlı hizmet olarak Neo4j Aura kullanarak başlamak mümkündür.

**2-Veri Modelleme:**

* Düğümler (Nodes): Gerçek dünya varlıklarını temsil eder (örneğin, bir kişi veya ürün).
* Kenarlar (Edges): Düğümler arasındaki ilişkileri temsil eder (örneğin, "ARKADAŞ", "SATIN ALDI").
* Özellikler (Properties): Hem düğümler hem de kenarlar üzerinde tutulan ek bilgilerdir (örneğin, bir kişinin adı veya yaşadığı şehir).

**3-Cypher Sorgu Dili:**

* Neo4j'nin sorgu dili olan Cypher'i kullanarak veri ekleyebilir, güncelleyebilir ve sorgulayabilirsiniz.

**Düğüm ekleme:**

CREATE (n:Person {name: 'John', age: 30})

Bu sorgu, adı John ve yaşı 30 olan bir Person düğümü oluşturur.

**İlişki ekleme:**

MATCH (a:Person {name: 'John'}), (b:Person {name: 'Doe'})

CREATE (a)-[:FRIENDS\_WITH]->(b)

Bu sorgu, John ve Doe kişileri arasında "FRIENDS\_WITH" ilişkisi oluşturur.

**Veri sorgulama:**

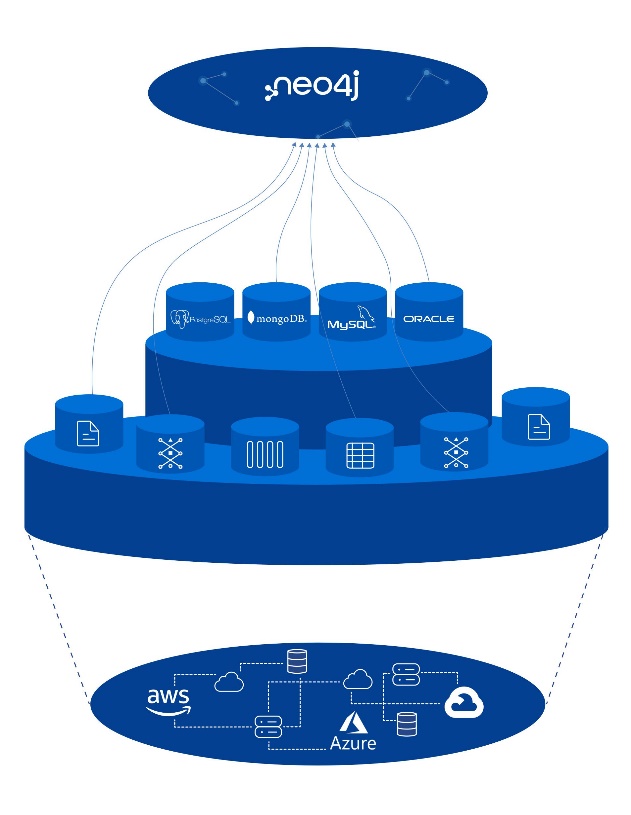
MATCH (p:Person)-[:FRIENDS\_WITH]->(f)

WHERE p.name = 'John'

RETURN f

Bu sorgu, John’un arkadaşlarınıdöndürür.

Neo4j DB dökümantasyonuna buradan ulaşabilirsiniz: <https://neo4j.com/>



Kaynakça:

<https://spring.io/guides/gs/accessing-data-neo4j>

<https://www.graphable.ai/software/what-is-neo4j-graph-database/>

<https://neo4j.com/docs/getting-started/get-started-with-neo4j/graph-database/>