

NoSql MongoDB

NoSQL Kavramı ve MongoDB Temelleri

NoSQL Nedir?

- NoSQL Veritabanı, sabit bir şema gerektirmeyen, ilişkisel olmayan bir Veri Yönetim Sistemi dir. Birleştirmeleri önler ve ölçeklendirilmesi kolaydır.
- NoSQL veri tabanları sabit bir şemaya dayanmaz , bunun yerine esnek ve dinamik veri yapılarına izin verir. Bu esneklik, NoSQL veri tabanlarını sosyal medya gönderileri, günlük dosyaları ve sensör verileri gibi büyük hacimli yapılandırılmış veya yarı yapılandırılmış verileri işlemek için çok uygun hale getirir.

NoSQL Nedir? (Yanlış Bilinenler)

- NoSQL, “SQL yok” anlamına gelmez
- Açılımı: Not Only SQL (Sadece SQL değil)
- İlişkisel veritabanlarına alternatif değil, farklı bir yaklaşımdır
- NoSQL veritabanları ilişkisel model kurallarına uymaz
 - Hiçbir zaman düz sabit sütunlu kayıtlar içeren tablolar sağlamaz
 - Bağımsız kümeler veya BLOB'larla çalışma
 - Nesne-ilişkisel haritalama ve veri normalleştirme gerektirmez
 - Sorgu dilleri, sorgu planlayıcıları, referans bütünlüğü bağlantıları, ACID gibi karmaşık özellikler yok
-

NoSQL Tarihçe

- 1998- Carlo Strozzi, hafif, açık kaynaklı ilişkisel veritabanı için NoSQL terimini kullandı
- 2000- Graf tabanlı veritabanı Neo4j başlatıldı
- 2004- Google BigTable kullanıma sunuldu
- 2005- CouchDB başlatıldı
- 2007- Araştırma makalesi Amazon Dinamo piyasaya sürüldü
- 2008- Facebook'un kaynakları açık Cassandra projesi
- 2009- NoSQL terimi yeniden kullanılmaya başlandı

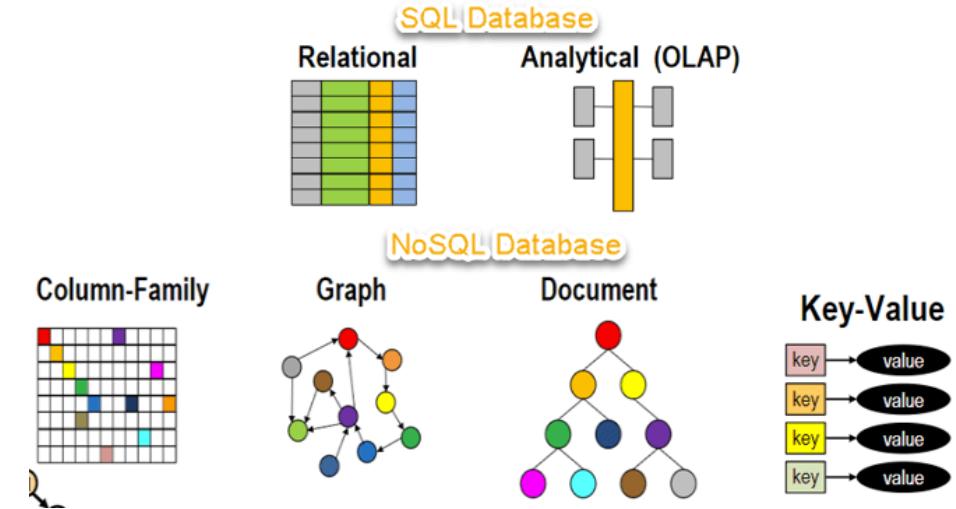
NoSql Neden Ortaya Çıktı?

- Yoğun kullanıcı trafiği
- Yüksek işlem hacmi
- Artan maliyet ve performans ihtiyacı
- RDBMS'lerin bu senaryolarda zorlanması
- Organik veri yapısıyla veri saklama
- Yatay ölçekleme ihtiyacı



Yaygın Yanlış Algı

- NoSQL = Yeni bir veritabanı ✗
- NoSQL = Sadece büyük veri ✗
- Gerçekte:
 - NoSQL bir veritabanı değil
 - Bir sistem, yaklaşım ve yöntemdir
 - Bu yaklaşımı kullanan farklı sistemler geliştirilmiştir



Büyük Veri Durumu

- Hız faktörü → NoSQL = Büyük veri algısı
 - Oysa: Günümüzde RDBMS'ler de milyarlarca kaydı yönetebiliyor
- NoSQL'in asıl gücü:
 - Şemasız (schema-less) yapı
 - Yatay ölçekleme
 - Normalize olmayan veriye hızlı erişim
- .

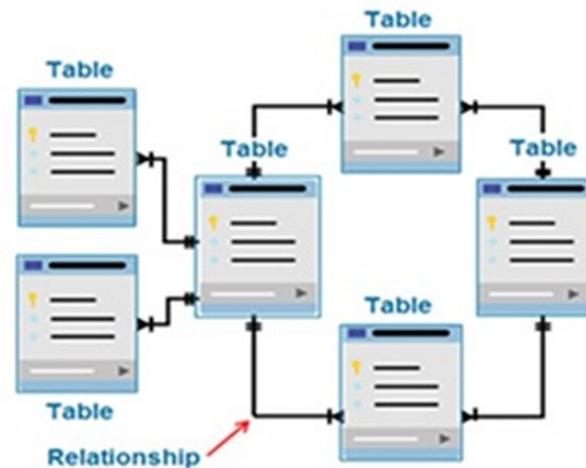
NoSql Nasıl Bir Yaklaşımdır

- Esnek şema yapısı
- Farklı veri saklama yaklaşımları (key-value, graph, document, column)
- Yatay ölçekleme
- Yüksek performans
- Normalize olmayan veri
- İlişkiler yok
- .



RDBMS Yapısı

- Önceden tanımlanmış:
 - Tablolar
 - Kolonlar
 - Veri tipleri
- Tablolar arası katı ilişkiler
- Veri bütünlüğü ön plandadır



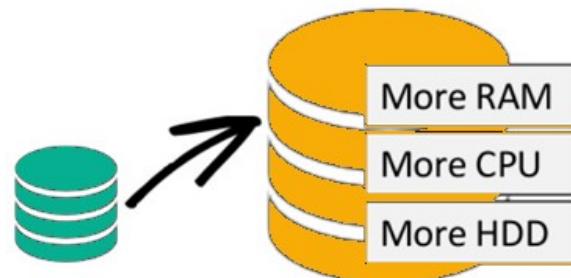
RDBMS

Relational Databases

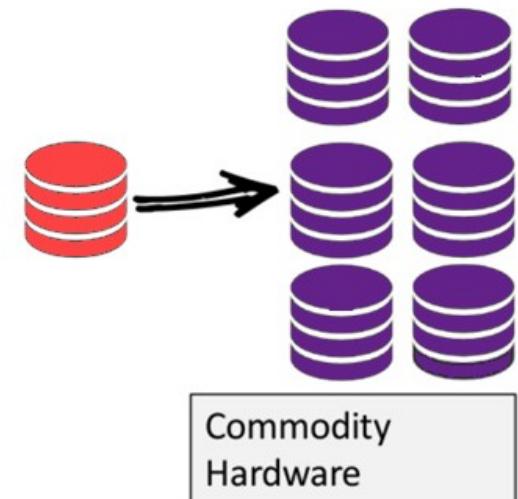
NoSql Yapısı

- Key / Value mantığı
- Veriler arasında fiziksel ilişki yok
- Bu sayede:
 - Daha hızlı erişim
 - Daha esnek yapı
- Yatay ölçekleme
- Esnek şema yapısı
- İlişki yok
- Veri bütünlüğü zamanla oluşur

Scale-Up (*vertical scaling*):



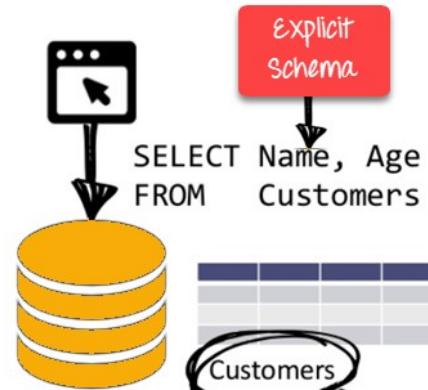
Scale-Out (*horizontal scaling*):



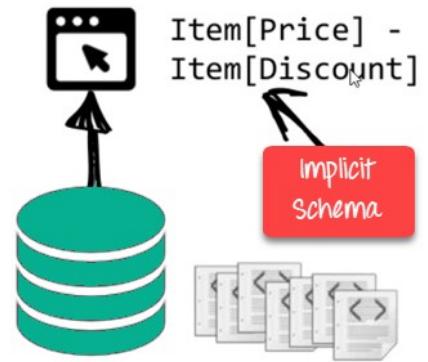
Organik Veri Kavramı

- NoSQL = Organik veri
- Veri yapısı:
- Süreç içinde değişebilir
- Önceden kesin sınırlarla belirlenmez
- Bu yönyle dinamik sistemler için uygundur

RDBMS:



NoSQL DB:



Şema Zorunluluğu Karşılaştırması

- RDBMS Şema değişirse:
 - Hata riski
 - Migrasyon gereksinimi
- NoSQL Şema Değişirse:
 - Şema zorunluluğu yok
- Yeni veri yapıları:
 - NoSql Eski verilerden bağımsız eklenebilir
 - RDBMS tasarım yeniden düzenlenir
- .

Veri Saklama Yapısı

- RDBMS
 - Tablolar
 - Satır / Sütun yapısı
- NoSQL
 - Document (Döküman)
 - JSON tabanlı veri saklama
- .

Esneklik Avantajı

- RDBMS
 - Tüm satırlar aynı kolonları uymalı
- NoSQL
 - Her doküman farklı alanlara sahip olabilir
 - Kolon ekleme/çıkarma serbesttir
 - Tüm verileri etkilemez
- .

Performans ve Maliyet

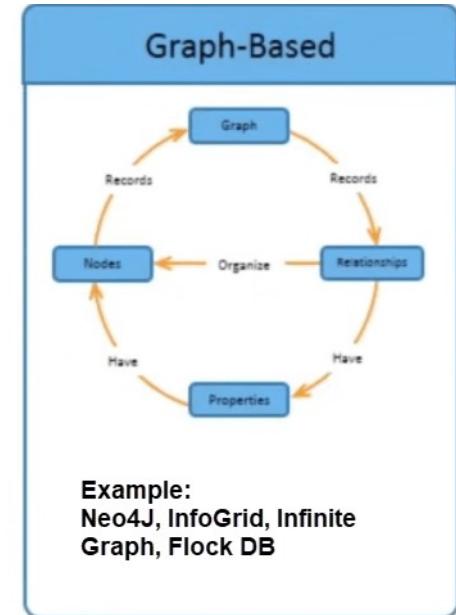
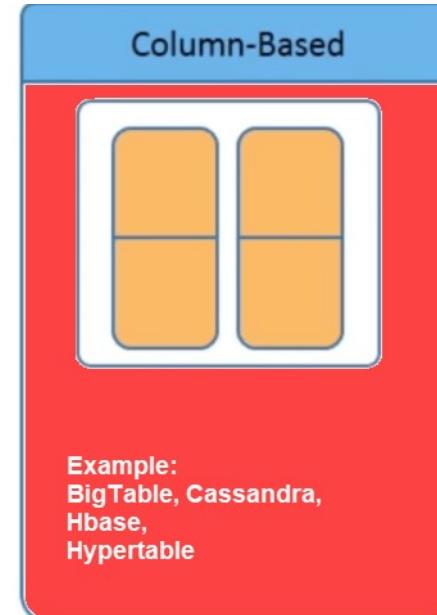
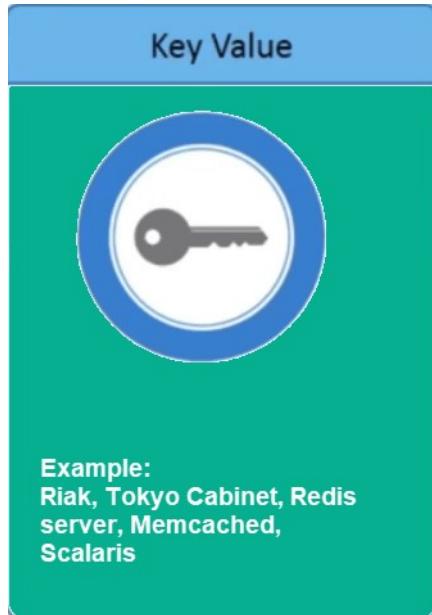
- NoSQL Sistemleri:
 - Yüksek erişilebilirlik
 - Daha az maliyet
 - Daha yüksek performans
 - Dağıtık mimariye uygundur

Ölçeklenebilirlik

- RDBMS
 - Dikey ölçekte
 - Daha güçlü donanım
- NoSQL
 - Yatay ölçekte
 - Daha fazla sunucu
 - Dağıtık sistem avantajı

NoSQL Sistem Yaklaşımıları

NoSQL sistemleri verileri dört kategoriden biri şeklinde saklar



NoSQL Sistem Yaklaşımıları

Sütun Tabanlı Sistemler

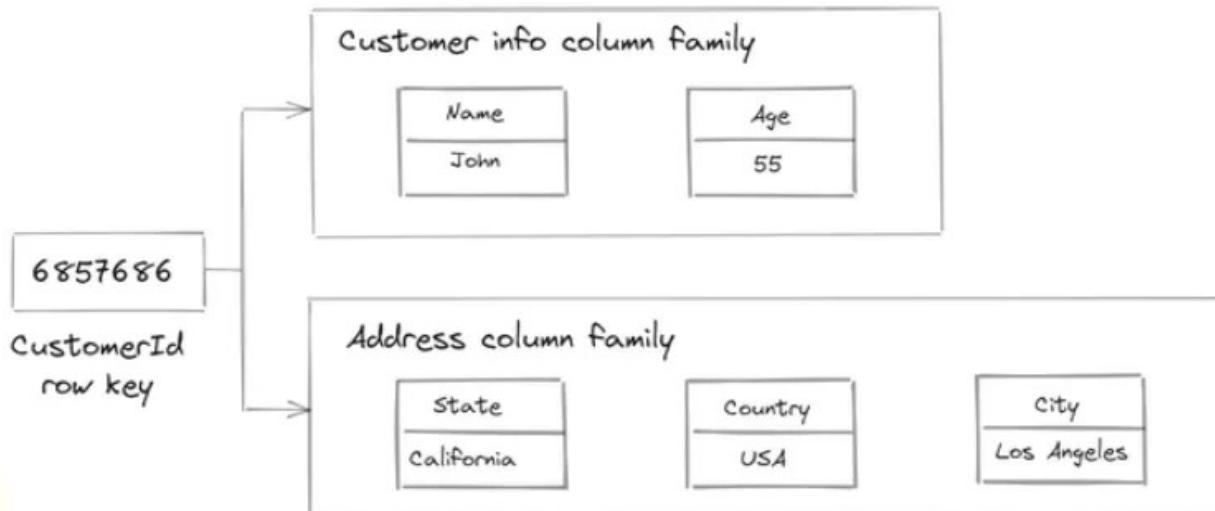
ColumnFamily			
Row Key	Column Name		
	Key	Key	Key
	Value	Value	Value
Column Name			
	Key	Key	Key
	Value	Value	Value

- Veri tabanı sütunlar üzerinde çalışır
- Google Bigtable
- Her sütun ayrı ayrı ele alınır
- Veriler tek bir sutunda kolayca bulunabildiğinden SUM, COUNT, AVG, MIN vb. gibi toplama sorgularında yüksek performans sağlarlar.
- Veri ambarlarını yönetmek için yaygın kullanılır
- HBase, Cassandra, Hypertable, BigTable örnek sistemlerdir

Sütun Tabanlı Yaklaşım

CustomerId	Name	Age	City	State	Country
6857686	John	55	Los Angeles	California	USA

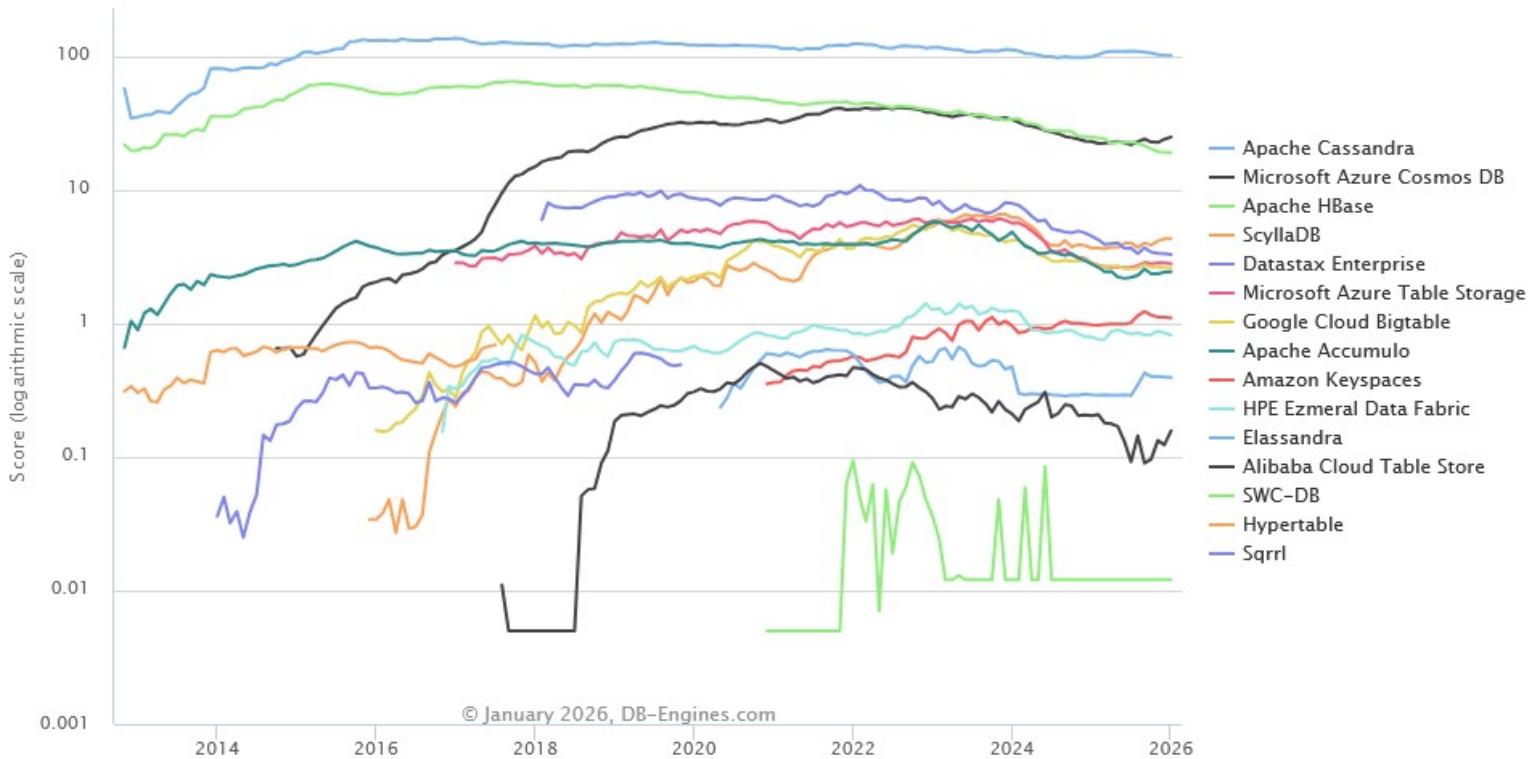
RELATIONAL DATABASE



WIDE-COLUMN
DATABASE

Sütun Tabanlı NoSql Veri Tabanları Eğilimleri

DB-Engines Ranking of Wide Column Stores



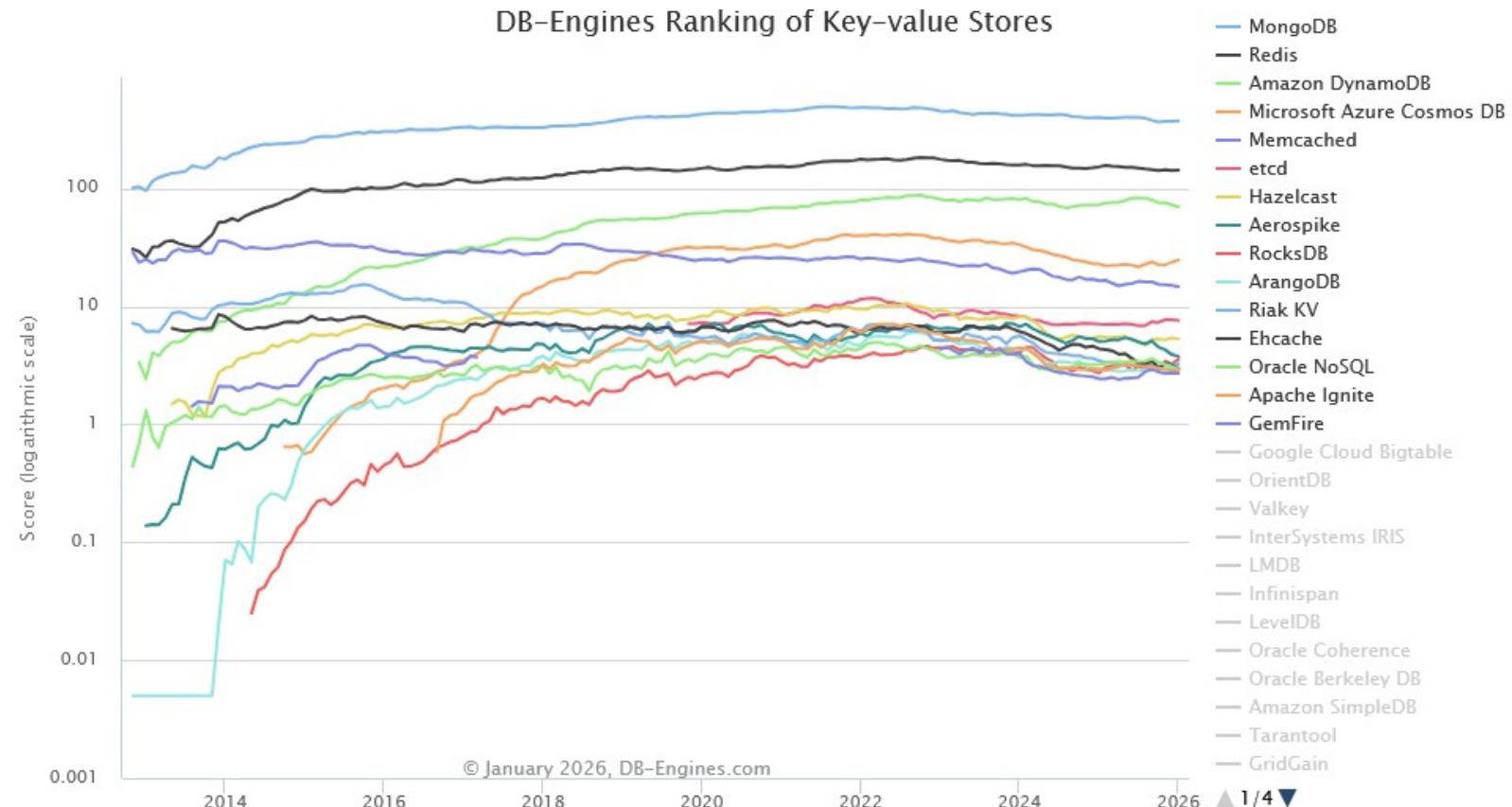
NoSQL Sistem Yaklaşımıları

Key-Value Tabanlı Sistemler

Key	Value
Name	Joe Bloggs
Age	42
Occupation	Stunt Double
Height	175cm
Weight	77kg

- Veri tabanı sütunlar üzerinde çalışır
- Google Bigtable
- Her sütun ayrı ayrı ele alınır
- Veriler tek bir sutunda kolayca bulunabildiğinden SUM, COUNT, AVG, MIN vb. gibi toplama sorgularında yüksek performans sağlarlar.
- Veri ambarlarını yönetmek için yaygın kullanılır
- HBase, Cassandra, Hypertable, BigTable örnek sistemlerdir

Key-Value Veri Tabanları Eğilimleri



NoSQL Sistem Yaklaşımıları

Doküman(Belge) Tabanlı Sistemler

- Veriler anahtar-değer çifti ile saklanır
- Değer kısmı belge olarak depolanır
- Belge JSON veya XML formatta olur
- CMS sistemleri, blog platformları, gerçek zamanlı analizler ve e-ticaret uygulamalarında kullanılır
- Amazon DynamoDB, CouchDB, MongoDB, Riak, Lotus Notes örnek verilebilir

Col1	Col2	Col3	Col4
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data

Document 1

```
{  
  "prop1": data,  
  "prop2": data,  
  "prop3": data,  
  "prop4": data  
}
```

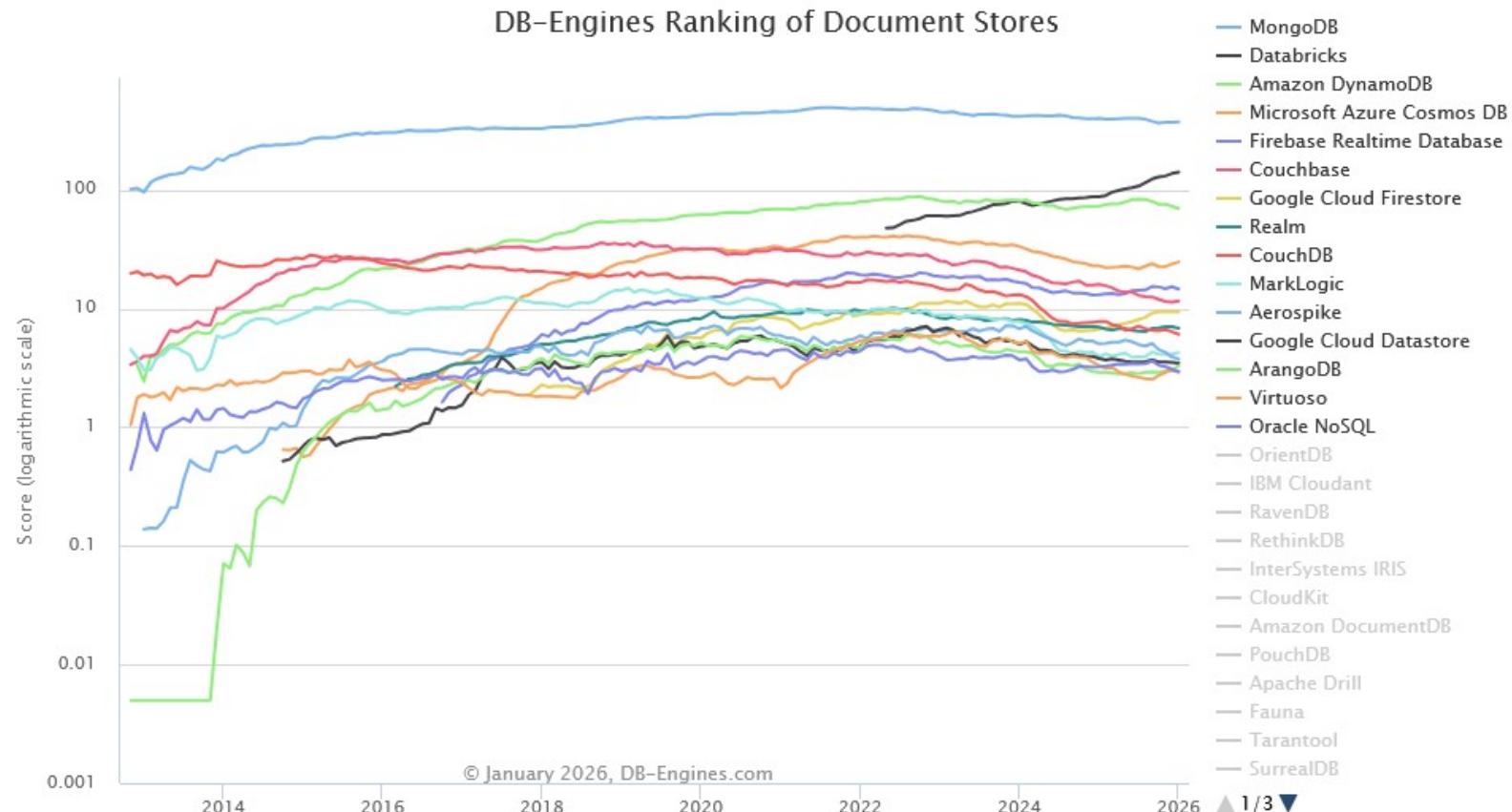
Document 2

```
{  
  "prop1": data,  
  "prop2": data,  
  "prop3": data,  
  "prop4": data  
}
```

Document 3

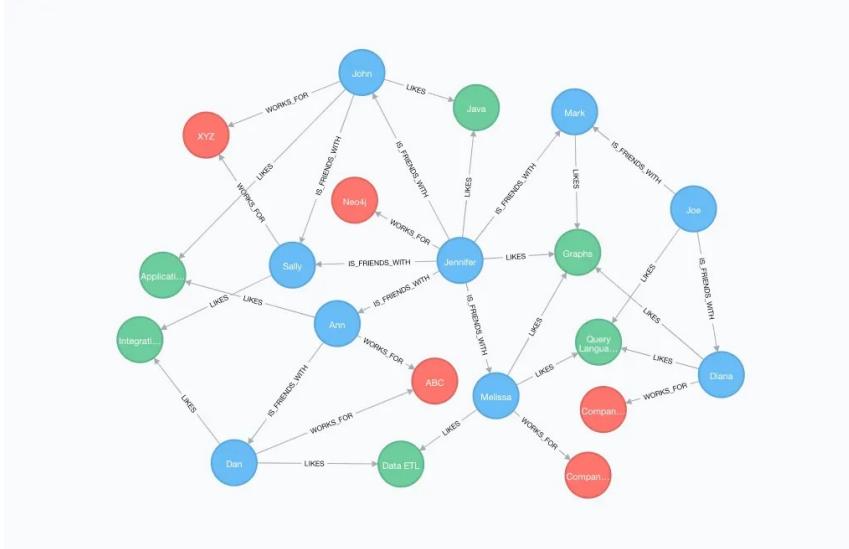
```
{  
  "prop1": data,  
  "prop2": data,  
  "prop3": data,  
  "prop4": data  
}
```

Doküman Tabanlı Sistem Eğilimleri

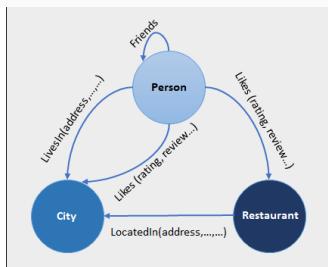


NoSQL Sistem Yaklaşımıları

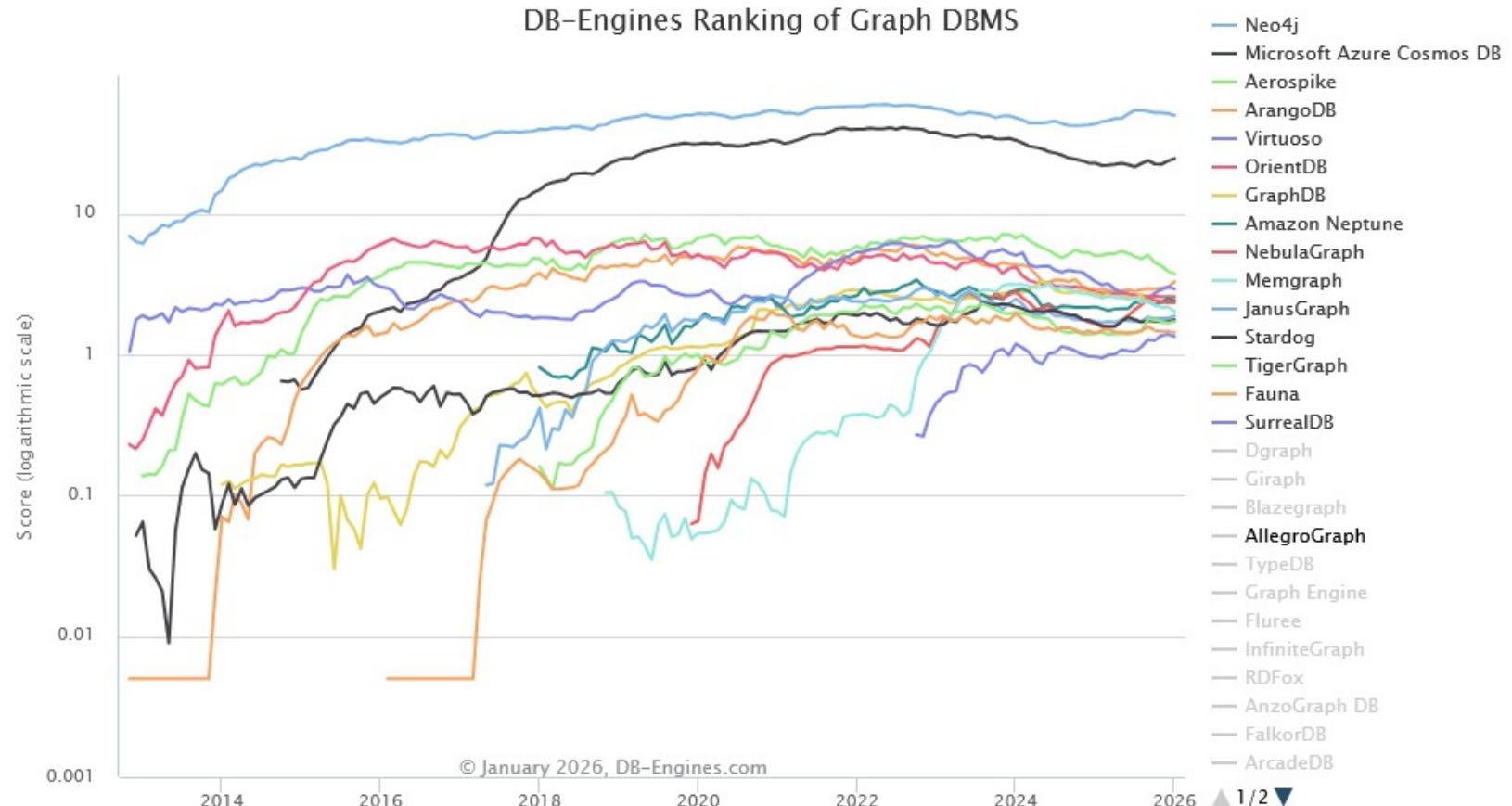
Graf Tabanlı Sistemler



- Veriler düğüm yapısında tutulur
- Çok hızlı ilişki sorguları yazılabilir
- Düğümler özellik taşıyabilir
- Klasik join yerine doğrudan ilişkiler gezilir bu yüzden çok hızlıdır
- Derin ilişki sorgularında oldukça hızlıdır
- Sosyal ağlar, lojistik ve mekansal veriler için uygun yapıdadır
- Neo4J, OrientDB, FlockDB, Amazon Neptune, ArangoDB, JanusGraph örnek verilebilir

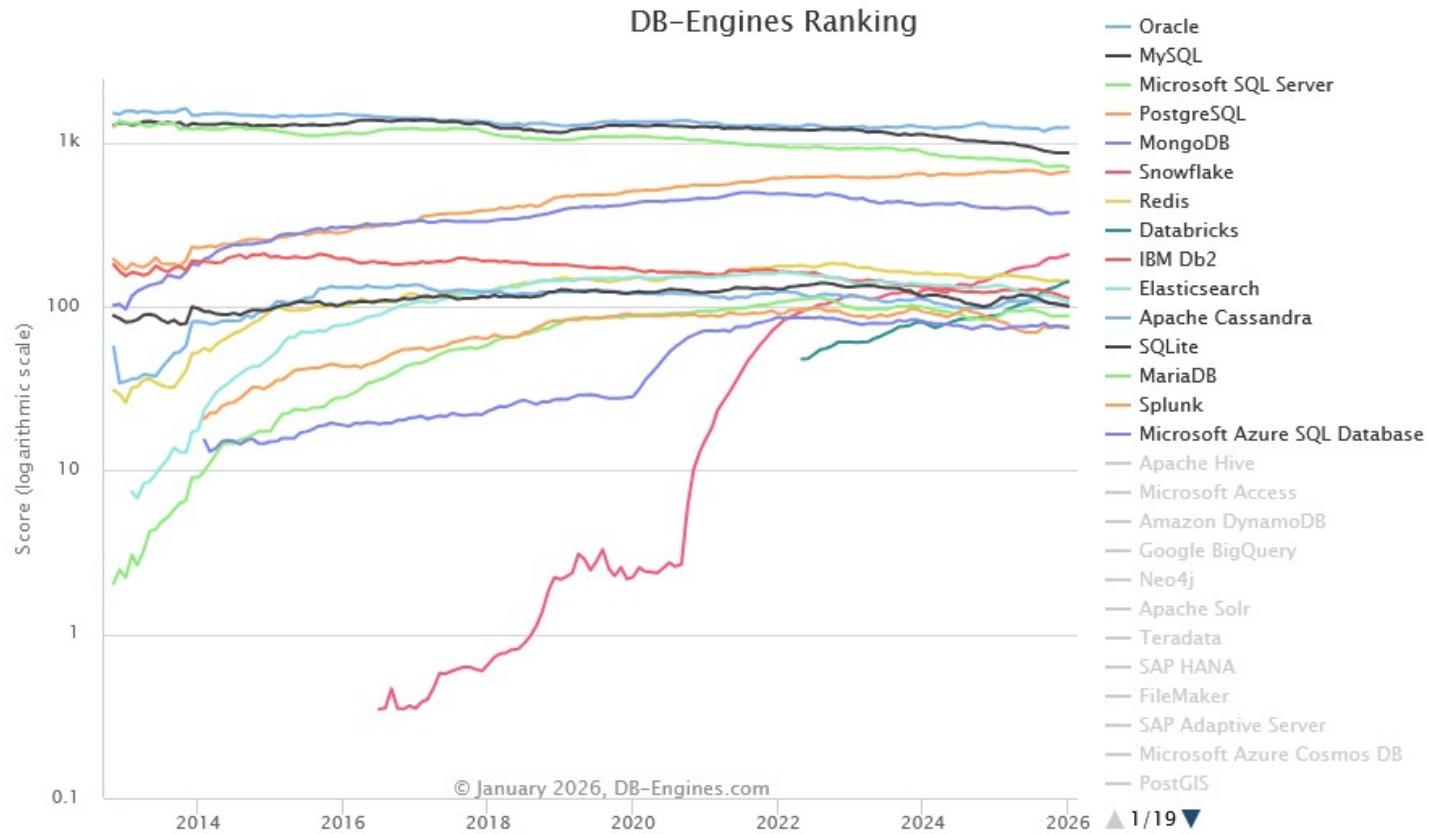


Graf Tabanlı Veri Tabanı Kullanımı



Veri Tabanları Genel Kullanım Trendleri

January



Özet

- NoSQL = Esneklik, Hız, Ölçeklenebilirlik
- Büyük veriden ziyade:
 - Organik ve değişken veri için uygundur

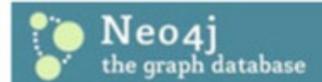
MongoDB Nedir?

- Doküman tabanlı NoSQL veri tabanı
- Şubat 2009'da ortaya çıkmıştır
- Collection içinde veriler saklanır
- Herhangi bir alana göre veya aralığa göre sorgu yazılabilir
- C++ dili ile yazılmıştır, Windows, Linux, Unix, BSD, OSX sistemlerinde çalışmaktadır
- Veriler BSON doküman yapısında saklanır
- Büyük ölçekli uygulamalar için tasarlanmıştır
- Geo-Spatial Queries (Harita tabanlı uygulamalarda konuma göre sorgulama)

MongoDB Nedir?

- MongoDB ile ilişkisel veri tabanı karşılaştırma
- Collection ---> Tablo
- Document ---> Satır
- Sabit tablo şeması yoktur
- Karmaşık veri yapıları
- Join ve Foreign Key kullanılmaz
- Yatay ölçeklenebilir

NoSQL Sistemler



Key / Value

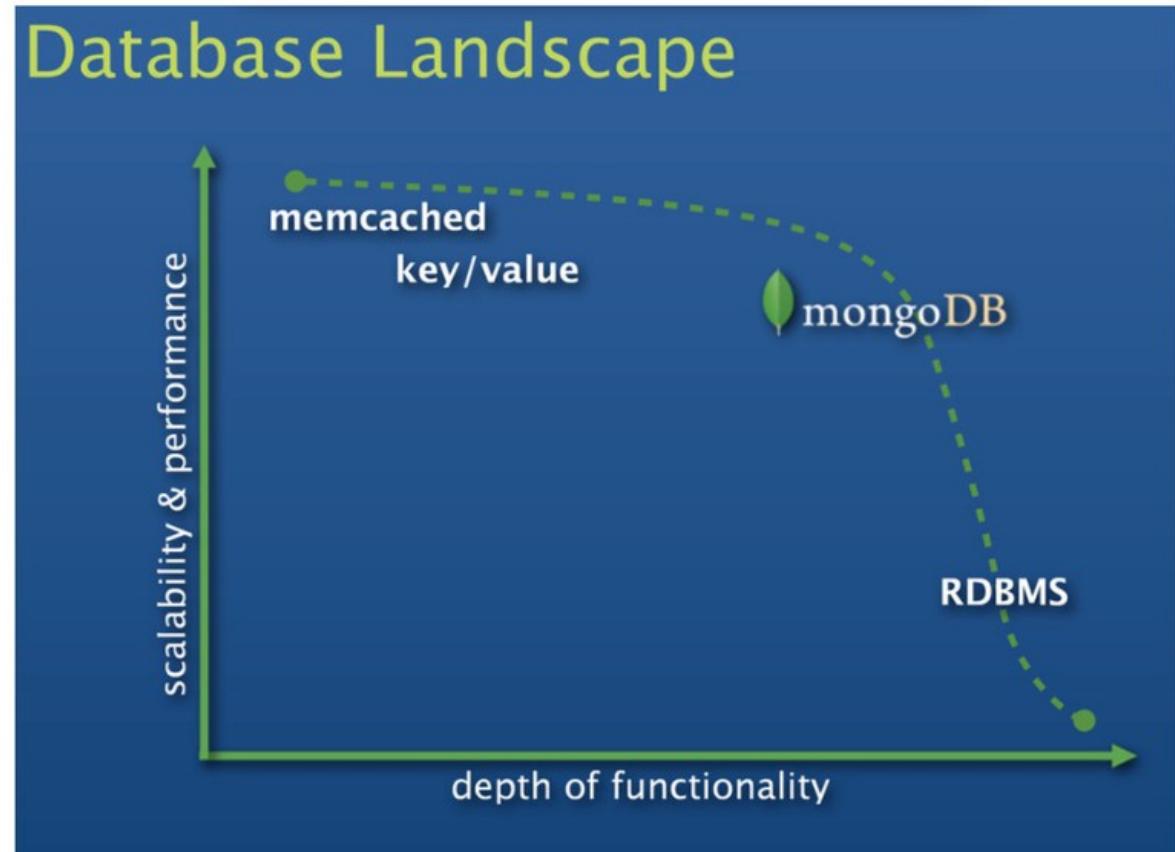
Column

Graph

Document

Veri Tabanı Dünyası

- Veri boyutu
arttıkça, veri
yapısı
karmaşıklaştıkça
RDBMS
sistemlerinde
ölçeklenibilirlik ve
performans azalır



MongoDB Hangi Dillerde Çalışır?

- C, C#(.Net), C++, ERLANG, HASKELL, JAVA
- JavaScript, Ruby, Perl, Python, Node.js

RDBMS ve MongoDB Terminolojisi

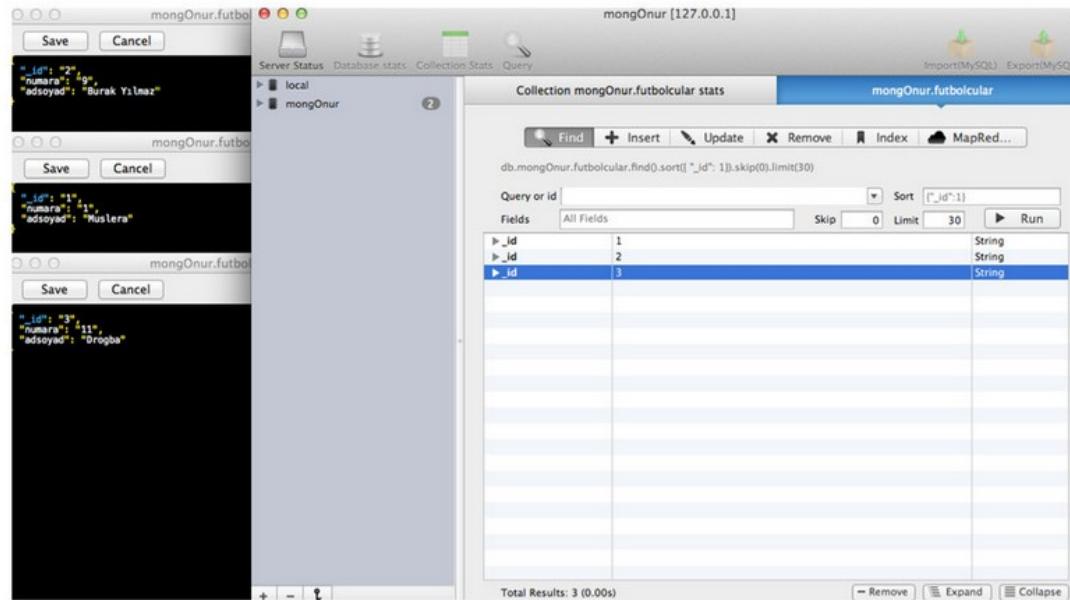
RDBMS	MongoDB
Table	Collection
Row(s)	JSON Document
Index	Index
Join	Embedding & Linking
Partition	Shard
Partition Key	Shard Key

MongoDB Kurulum

- İndirme adresi: <https://www.mongodb.com/try/download/community>
- İlgili adreste yukarıdaki ekran görüntüsünde olduğu gibi “Community Server” sekmesinde bilgisayarınıza uygun versiyonu belirttikten sonra “Download” butonuna tıklamanız yeterlidir.
- İndirdiğiniz programı çalıştırıp, kurulumu gerçekleştirdikten sonra “C:\Program Files\MongoDB\Server” dizisine göz atarsak MongoDB'nin kaynak dosyalarına erişebilmekteyiz.
- Dizin içerisinde versiyon isminde klasör altında “/bin” klasörü bizim MongoDB'ye erişmemizi sağlamaktadır.

MongoDB için GUI

- MongoHUB
- MongoVUE
- UMongo



Komut Satırı Arayüzü

- “Komut İstemi” penceresini açıp “C:\Program Files\MongoDB\Server\xxxversiyonumarası.0\bin” dizinine geçiniz.
- mongo –version
- mongo komutu MongoDB Komut Satırı Arayüzüünü aktifleştirecektir. Bu komuttan sonra yazılan komutlar MongoDB’de yorumlanır ve çalıştırılır
- show dbs: Sunucudaki tüm veri tabanlarını getirir

Komut Satırı Arayüzü

- use[veri tabanı adı]:belirtilen veri tabanını kullanılabılır vaziyette seçer
- Show collections: üzerinde çalışılan veri tabanındaki collectionları getirir

Verileri Dizi Şeklinde Görünümü

Bir kişiye ait verileri tanımlayan basit bir json belgesi

```
{  
  "_id" : 1 ,  
  "isim" : {  
    "ilk" : "Ada" ,  
    "son" : "Lovelace"  
  },  
  "Başlık" : "İlk Programcı" ,  
  "ilgi alanları" : [ "matematik" , "programlama" ]  
}
```