Veritabanı Yönetimi ve Modellemesi

HAFTA 12

Dr. Fatmana Şentürk

Haftalık Ders Akışı

1. Veritabanı Kavramlarına Giriş
2. Veri Tabanı Türleri, İlişkisel Veri Tabanı Tasarımı
3. ER Diyagramları ve Normalizasyon
4. SQL Server Arayüzü, Veri Tabanı Nesneleri
5. T-SQL ve SQL Sorguları
6. İndeks ve View
7. Geçici Tablolar, Kontrol Yapıları
8. Ara Sınav
9. Stored Procedure
10. Fonksiyonlar
11. Tetikleyiciler(Trigers)
12. Yedekleme
13. **Kullanıcı Türleri ve Kullanıcı Yönetimi**
14. **No-SQL Veri Tabanları**

Kullanıcı Türleri

- Database Admin
- Database Designer
- Users

Database Admin

- Veritabanına erişim izinleri
- Veritabanı kullanıcılarının koordinasyonu
- Yazılım ve donanım kaynaklarının kullanımı
- Job, kullanıcı tanımı, otomatik yedekleme..vb gibi işlemler
- Güvenlik ihlalleri

Database Designer

- Saklanacak olan verilerin modellemesi
- Verilerin türlerinin belirlenmesi
- Kısıtların oluşturulması
- Şemaların tespiti

Users

- Raporları sorgulamak
- Verileri güncellemek ve oluşturmak
- Kullanıcı kategorisi :
- Sıradan son kullanıcılar zaman zaman veritabanına erişir, her seferinde farklı bilgilere ihtiyaç duyabilirler(üst düzey yöneticiler)
- Yerel veya parametrik kullanıcılar: Başlıca iş işlevlerini(ekleme, güncelleme, silme) gibi işlemler. Sürekli sorgulama (Stok takibi)
- Örnek:
- Banka gişe görevlisi hesap bakiyelerini kontrol eder ve para çekme ve yatırma işlemlerini yapar.
- Rezervasyon acenteleri veya havayolları, oteller ve araç kiralama şirketleri için müşteriler belirli bir istek için uygunluğunu kontrol eder ve rezervasyon yapar.

T-Sql

- Dışarıdan erişim için bir kullanıcı oluşturuldu
- CREATE LOGIN userName WITH PASSWORD=‘password’
- Sql içerisinde işlem yapmak için bir USER oluşturuldu
- CREATE USER userName FOR LOGIN userName
- İzinlerin verilmesi

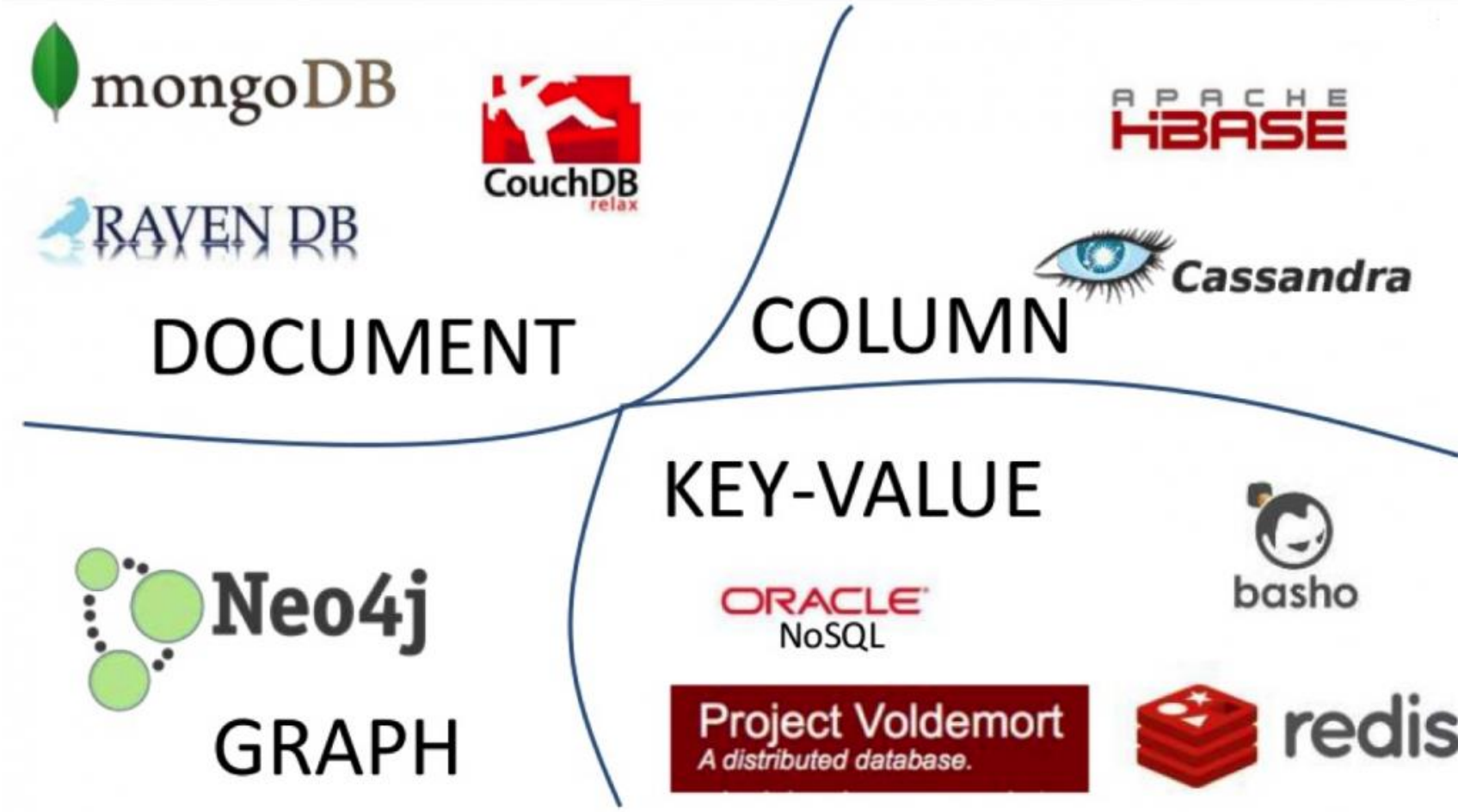
Kulanım İzinleri

- GRANT: Bir iznin verilmesini sağlar
  - GRANT CREATE TABLE TO Username
  - GRANT INSERT,UPDATE,DELETE TO Username
  - GRANT SELECT ON tblName TO Username
- WITH GRANT: Kullanıcı kendisinde bulunan yetkileri başka kullanıcılara verebilir.
  - GRANT SELECT,INSERT ON tblName TO Username WITH GRANT OPTION
- DENY: Kullanıcının yetkilerinin geri alınmasını sağlar.
  - DENY INSERT, SELECT ON tblName TO Username
- REVOKE: GRANT ile değiştirdiğimiz hakları eski haline döndürmek için kullanılır. Bir nesneyi oluşturan kullanıcının REVOKE ile nesne üzerindeki yetkilendirme ve kullanma hakkı yok edilemez.
  - REVOKE ALL ON REGION TO Username

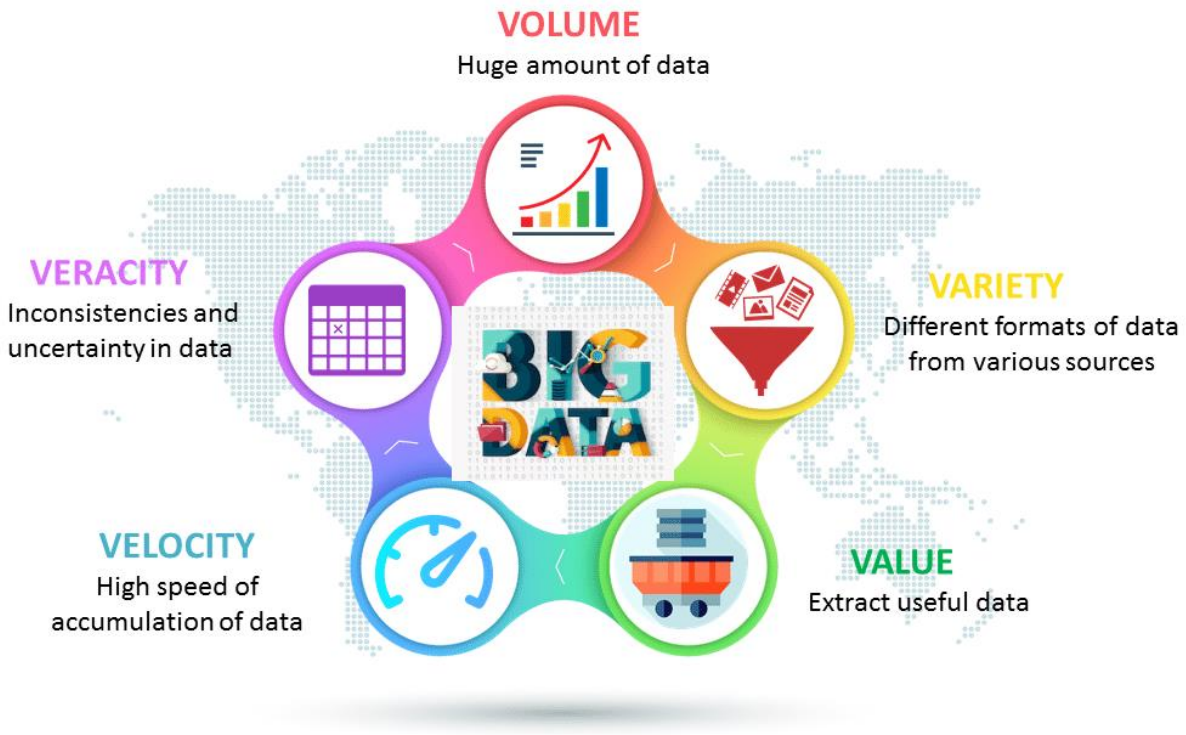
Arayüz aracılığı ile Kullanıcı tanımlama işlemleri

NoSql

- Not Only SQL

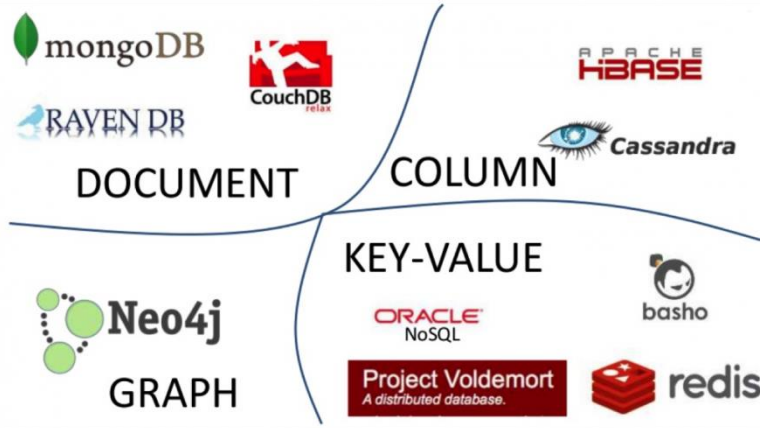


Büyük veri ve V kavramı

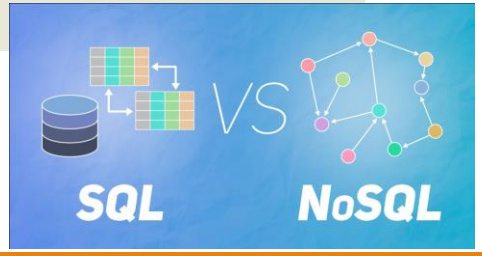


NoSQL

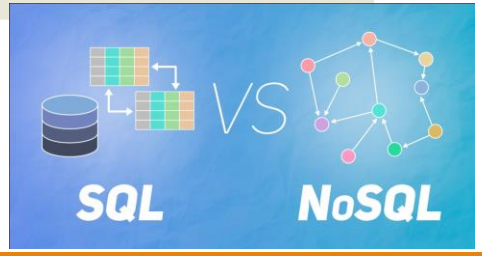
- Belirli veri modelleri için özel olarak tasarlanmış(İlişkisel veri modelini kullanmazlar)
- Modern uygulamalar oluşturmaya yönelik
- Esnek
- Ölçeklenebilir
- Hata toleransına sahip
- Yüksek miktarda veri işleyebilen
- Hız Dağıtık veritabanı mimarisini destekler
- Önceliği işlem tutarlılığı yerine performansa verir



	İlişkisel veritabanları	NoSQL veritabanları
En uygun iş yükleri	İşlemsel ve güçlü tutarlılığa sahip çevrimiçi işlem gerçekleştirme (OLTP) uygulamaları için tasarlanan ilişkisel veritabanları, çevrimiçi analitik işlem (OLAP) için uygundur.	NoSQL anahtar-değer, belge, grafik ve bellek içi veritabanları, düşük gecikme süreli uygulamaları içeren çeşitli veri erişimi desenlerine yönelik olarak OLTP için tasarlanmıştır. NoSQL arama veritabanları, yarı yapılandırılmış veriler üzerinde analitik için tasarlanmıştır.
Veri modeli	İlişkisel model, verileri satır ve sütunlardan oluşan tablolar halinde normalleştirir. Tablolar, satırlar, sütunlar, dizinler, tablolar arasındaki ilişkiler ve diğer veritabanı öğeleri bir şema tarafından kesin bir şekilde tanımlanır. Veritabanı, tablolar arasındaki ilişkilerde başvurusal bütünlük uygular.	NoSQL veritabanları belge, grafik, anahtar-değer, bellek içi ve arama dahil olmak üzere çeşitli veri modelleri sağlar.
ACID özellikleri	İlişkisel veritabanları bölünmezlik, tutarlılık, yalıtım ve dayanıklılık (ACID) özelliklerini sağlar:	NoSQL veritabanları yatay olarak ölçeklendirilebilen daha esnek bir veri modeli sağlamak için genellikle ilişkisel veritabanlarının bazı ACID özelliklerini esneterek bunlardan ödün verirler. Bu, tek bir bulut sunucusunun ulaşamayacağı derecede yatay ölçeklendirme gerektiren yüksek performanslı, düşük gecikme süreli kullanım örnekleri için NoSQL veritabanlarının mükemmel bir seçim olmasını sağlar.
	Bölünmezlik, bir işlemin ya tamamen yürütülmesini ya da hiç yürütülmemesini gerektirir.	
	Tutarlılık, bir işlem gönderildiğinde verilerin veritabanı şemasına uygun olmasını gerektirir.	
	Yalıtım, eş zamanlı işlemlerin birbirinden bağımsız olarak yürütülmesini gerektirir.	
	Dayanıklılık, beklenmeyen bir sistem hatasından veya güç kesintisinden son bilinen duruma kurtarma becerisi gerektirir.	



	İlişkisel veritabanları	NoSQL veritabanları
Performans	Performans genellikle disk alt sistemine bağlıdır. En üst düzey performans için genellikle sorguların, dizinlerin ve tablo yapısının optimize edilmesi gerekir.	Performans genel olarak temel donanımın küme boyutu, ağ gecikme süresi ve çağrı yapan uygulama gibi etmenlerin birleşimine bağlıdır.
Ölçek	İlişkisel veritabanları genellikle donanımın işlem kapasitesini artırarak ölçeği artırır veya salt okunur iş yüklerine yönelik replikalar ekleyerek ölçeği genişletir.	Anahtar-değer erişim desenleri aktarım hızını artırmak için neredeyse sınırsız ölçekte tutarlı performans sağlayan dağıtılmış mimariyi kullanarak ölçeği genişletebildiğinden, NoSQL veritabanları genellikle bölümlendirilebilen veritabanlarıdır.
API'ler	Veri depolama ve alma istekleri, yapılandırılmış sorgu diline (SQL) uygun sorgular kullanılarak iletilir. Bu sorgular ilişkisel veritabanı tarafından ayrıştırılır ve yürütülür.	Nesne tabanlı API'ler, uygulama geliştiricilerinin bellek içi veri yapılarını kolayca depolamasına ve almasına imkan tanır. Bölüm anahtarları, uygulamaların anahtar-değer çiftlerini, sütun kümelerini veya seri hale getirilmiş uygulama nesneleri ve öznitelikleri içeren yarı yapılandırılmış belgeleri bulmasına imkan tanır.



	NoSQL veya ilişkisel olmayan	SQL veya ilişkisel
ŞUNLAR İÇİN EN İYİSİ:	Büyük, ilişkisiz, belirsiz veya hızla değişen verileri işleme.	İlişkisel olan ve önceden tanımlanabilen mantıksal ve ayrık gereksinimleri bulunan verileri işleme.
	Şemadan bağımsız veriler veya uygulama tarafından yönetilen şema.	Şema, uygulama ve veritabanı arasında korunmalı ve eşitlenmiş durumda tutulmalıdır.
	Performans ve kullanılabilirliğin güçlü tutarlılıktan daha fazla öneme sahip olduğu uygulamalar.	İlişkisel yapılar için oluşturulmuş eski sistemler.
	Dünyanın dört bir yanındaki kullanıcılara hizmet sunan sürekli çalışan uygulamalar.	Karmaşık sorgu veya çok satırlı işlem gerektiren uygulamalar.
SENARYOLAR:	Mobil uygulamalar.	Muhasebe, finans ve bankacılık sistemleri.
	Gerçek zamanlı analiz.	
	İçerik yönetimi.	Envanter yönetimi sistemleri.
	Kişiselleştirme.	
	IoT uygulamaları.	İşlem yönetimi sistemleri.
	Veritabanı geçişi.	
ÖLÇEK:	Sunucular arasında parçalama gerçekleştirerek verileri yatay olarak ölçeklendirir.	Sunucu yükünü artırarak verileri dikey olarak depolar.
VERİ MODELİ:	Veritabanı türleri: anahtar-değer, belge, sütunlu ve graf veritabanları.	Veritabanı türü: ilişkiler olarak gruplandırılmış, satırlardan oluşan tablolar.
	Veritabanı türüne bağlı olarak veri depolar.	Yapılandırılmış Sorgu Dili'ni (SQL) kullanır.
		Verileri tablolarda satır olarak depolar. İlgili veriler ayrı depolanır ve karmaşık sorgular için birleştirilir.

NoSQL


NoSQL
Gaming
Social
IoT
Web
Mobile
Enterprise
$
Key/value store
Document database
Column family store
SQL
Web
Mobile
Enterprise
Data mart
Relational table storage
Relationships use joins

NoSQL Avantajları

- İlişkisel veritabanlarına göre yüksek erişilebilirlik imkanı sunarlar.
- Okuma ve yazma performansları olarak göreceli olarak ilişkisel veritabanı sistemlerine göre daha performanslı olabilirler.
- Yatay olarak genişletilebilirler. Binlerce sunucu birarada küme olarak çalışabilir ve çok büyük veri üzerinde işlem yapabilirler.
- Esnek yapılarından dolayı programlama ve bakım anlamında kolaylık sağlarlar.
- Farklı özelliklere sahip birçok implementasyon arasından seçim yapma şansınız vardır.
- Birçok açık kaynak kodlu projelere ve bulut bilişim teknolojilerine uygun olduğu için maliyet olarak ilişkisel veritabanı yönetim sistemlerine göre daha avantajlıdır.

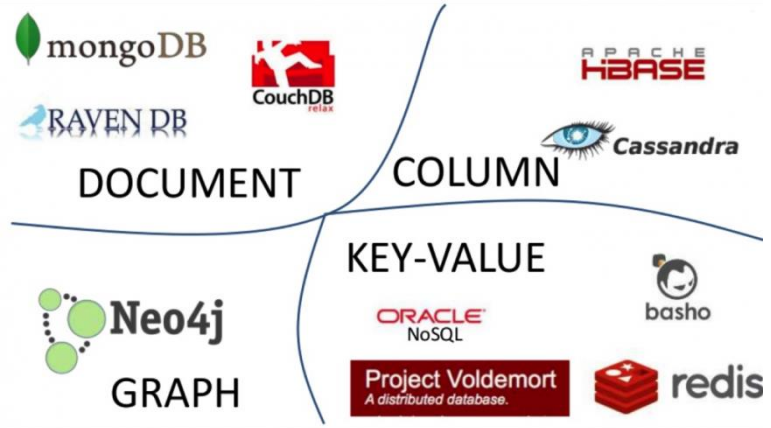
NoSQL Dezavantajları

- İlişkisel veritabanı yönetim sistemlerini kullanan uygulamaların NoSQL sistemlere taşınması başlangıçta zor olacaktır. Veri başarılı bir şekilde taşınsa bile bağlantıyı (join) kullanan kodlarda düzenlemelerin yapılması gerekecektir.
- İlişkisel veritabanı yönetim sistemlerindeki sorgu tabanlı veri erişimi yerine NoSQL sistemlerdeki anahtar tabanlı veri erişimi sağlamak gerekmektedir. Buna göre bir yapılandırmaya gidilmesi zaman alabilmektedir.
- İlişkisel veritabanı yönetim sistemlerindeki işlem hareketleri (transaction) kavramı, NoSQL veritabanı sistemlerinde bulunmadığı için veri kaybı söz konusu olabilmektedir.
- NoSQL veritabanı sistemleri veri güvenliği konusunda ilişkisel veritabanı yönetim sistemleri kadar gelişmiş özelliklere henüz sahip değiller. BazıNoSQL projelerin dökümantasyon ve profesyonel destek konusunda eksikleri vardır.

NoSQL’de tanımlı veritabanları

- Doküman veritabanı
- Sütun veritabanı
- Anahtar-değer (key-value) veritabanı
- Graf veritabanı

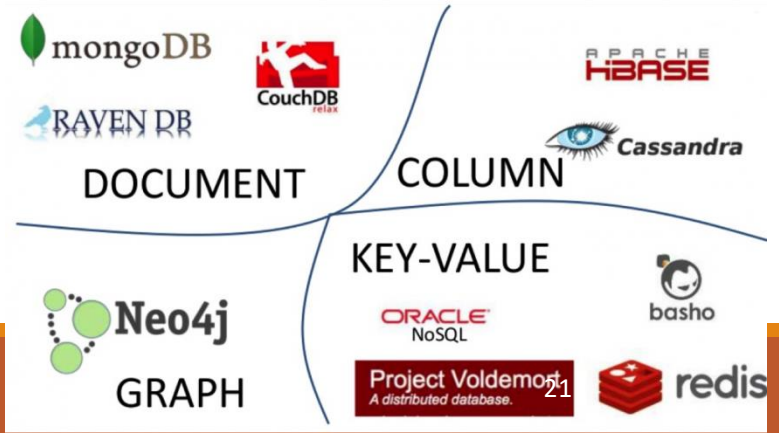
Doküman Veritabanı



- En önemli özelliği “esnek”
- Bir anahtara karşılık gelen veriler “doküman” adı verilen nesnelerde saklanırlar
- Genellikle JSON formatındadır
- Dokümanlar çok sayıda alan içerebilir ve her dokümanın yapısı birbirinden farklı olabilir
- İlişkisel veritabanlarında çok biçimli (polymorphic) veriler çok sayıda tabloya dağıtılır. Ancak Doküman tabanlı veritabanları esnek yapısı ile bu ihtiyacı ortadan kaldırmaktadır.
- Doküman tabanlı veritabanları, içerik yönetim sistemleri, elektronik ticaret uygulamaları ve günlük (blog) siteleri gibi esnek veri yapısına ihtiyaç duyan uygulamalar için uygundur.
- Örnek:
  - Cvlerin tutulduğu bir veri tabanı tasarımında doküman veri tabanında saklanabilir.

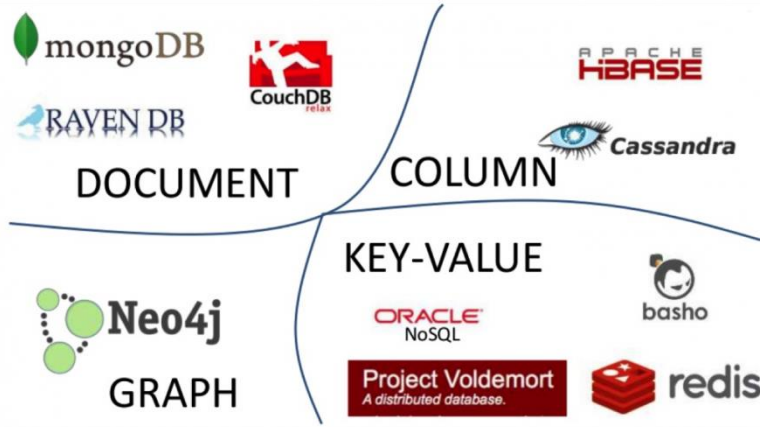
Sütun(Kolon) Veritabanı

- Yüksek okuma yazma performansı ve yüksek erişilebilirlik (high availibity) için tasarlanmıştır.
- Birden çok sunucu üzerinde dağıtık olarak çalışabilir
- Büyük verileri saklama kapasitesi
- Yazma işleminde kesinti yaşanmaz fakat dağıtık yapısından dolayı kısa süreli veri tutarsızlığı (inconsistency) yaşanabilir. Bu özelliği tolere edemeyen uygulamalar için uygun değildir.
- Sütun tabanlı veritabanları, içerik yönetim sistemleri, günlük (blog) uygulamaları, uygulama kayıtlarının (log) saklanması gibi uygulamalarda yaygın olarak kullanılmaktadır.
- Örnek:
  - Facebook üzerindeki özel mesajlaşma



Anahtar- Değer Veritabanı

- Küçük ve çok sayıda okuma yazma işleminin yapıldığı uygulamalar için uygundur.
- Bir anahtara karşılık gelen veri genellikle boolean, integer gibi basit verilerdir.
- Önbellek (caching) yazılımları, alışveriş sepeti uygulamaları ve görüntü dosyalarının saklanması gibi uygulamalar için uygundur.
- Örnek:
  - Twitter verileri
  - Snapchat'in Snapchat Stories özelliğinin



Graf(Çizge) Veritabanı

- Graf tabanlı veritabanlarında veriler düğümler (node), ilişkiler (edge) ve özellikler (properties) şeklinde tutulurlar.
- Diğer veritabanı türlerinden farklı olarak veriler arasındaki ilişkiler de saklanabilir.
- Graf tabanlı veritabanlarının kullanım alanı daha kısıtlıdır.
- Graf tabanlı veritabanları, sosyal ağ uygulamaları, kimlik ve erişim yönetimi uygulamaları ve tavsiye uygulamaları için uygundur.
- Örnek:
  - Sosyal medya üzerindeki arkadaşlık ilişkileri, önerileri