Veritabanı Yönetim Sistemleri
1906003022015

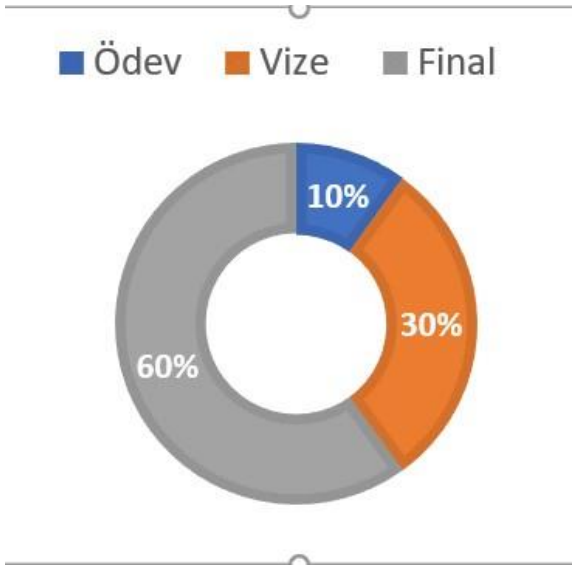
Dr. Öğr. Üy. Önder EYECİOĞLU
Bilgisayar Mühendisliği

Giriş

Ders Günü ve Saati:

Pazartesi: 09:15-12:45

- Devam zorunluluğu %70
- Uygulamalar MS SQL ve MongoDB üzerinde gerçekleştirilecektir.



HAFTA	KONULAR
Hafta 1	VT ve VTYS'ne giriş
Hafta 2	ER Veri Modeli
Hafta 3	İlişkisel Modeller, İlişkisel model tasarımı
Hafta 4	İlişkisel Cebir ve Hesaplamalar
Hafta 5	İlişkisel Sorgular, SQL giriş
Hafta 6	SQL ile veri tabanı programlama
Hafta 7	SQL-Kısıtlar:Veri-tipi,birincil-anahtar,ikinci-anahtar,
Hafta 8	Vize
Hafta 9	İlişkisel Veri Tabanı Tasarımı ve Normalizasyon
Hafta 10	yarı-yapısal veri modelleri, XML
Hafta 11	JSON
Hafta 12	İlişkisel olmayan DB, NoSQL
Hafta 13	NoSQL
Hafta 14	DBMS -Eşzamanlılık (Concurrency) Kontrolü

İÇERİK

nosql

NoSQL

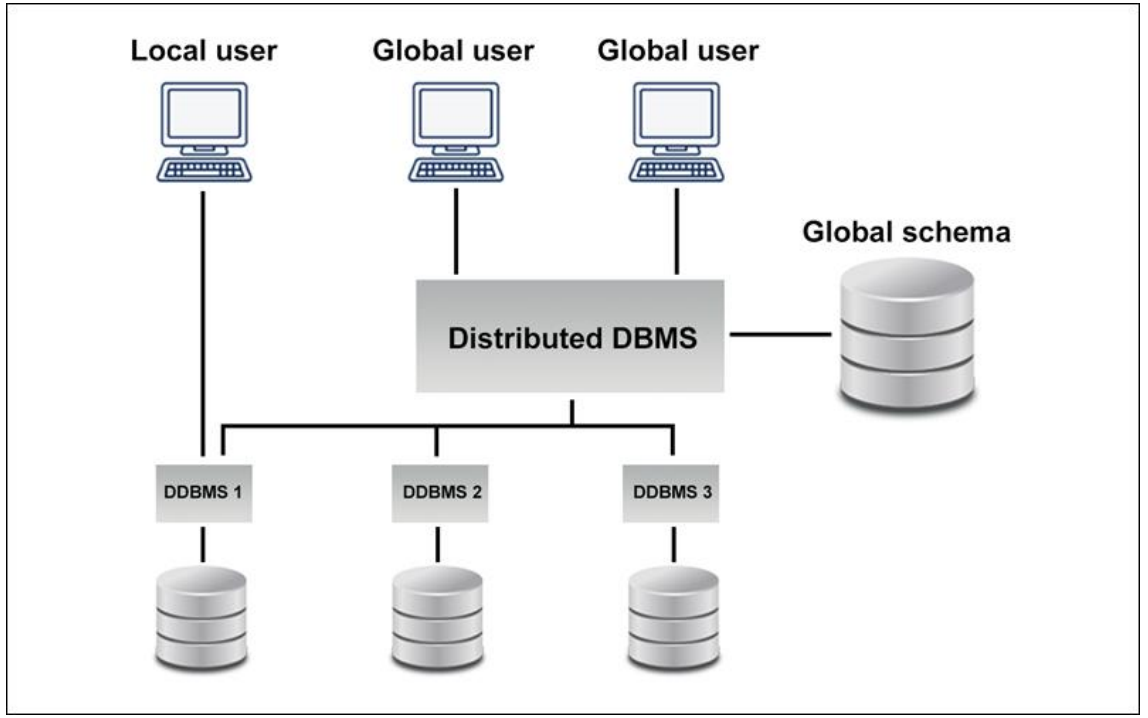
Bir veritabanı işlemi, atomik, tutarlı, yalıtılmış ve dayanıklı olmalıdır. Aşağıda bu dört noktayı tartıştık.

- **Atomik** : İşlem, tüm veri değişiklikleriyle tamamlanması gereken veya hiçbiri gerçekleştirilmeyen mantıksal bir iş birimidir.
- **Tutarlı** : İşlemin sonunda tüm veriler tutarlı bir durumda bırakılmalıdır.
- **İzole** : Bir işlem tarafından gerçekleştirilen veri değişiklikleri, başka bir işlemden bağımsız olmalıdır. Bu olmadıkça, bir işlemin sonucu hatalı olabilir.
- **Dayanıklı** : İşlem tamamlandığında, işlemin yaptığı değişikliklerin etkilerinin sistemde kalıcı olması gerekir.

Genellikle bir işlemin bu dört özelliği ACID olarak kısaltılır .

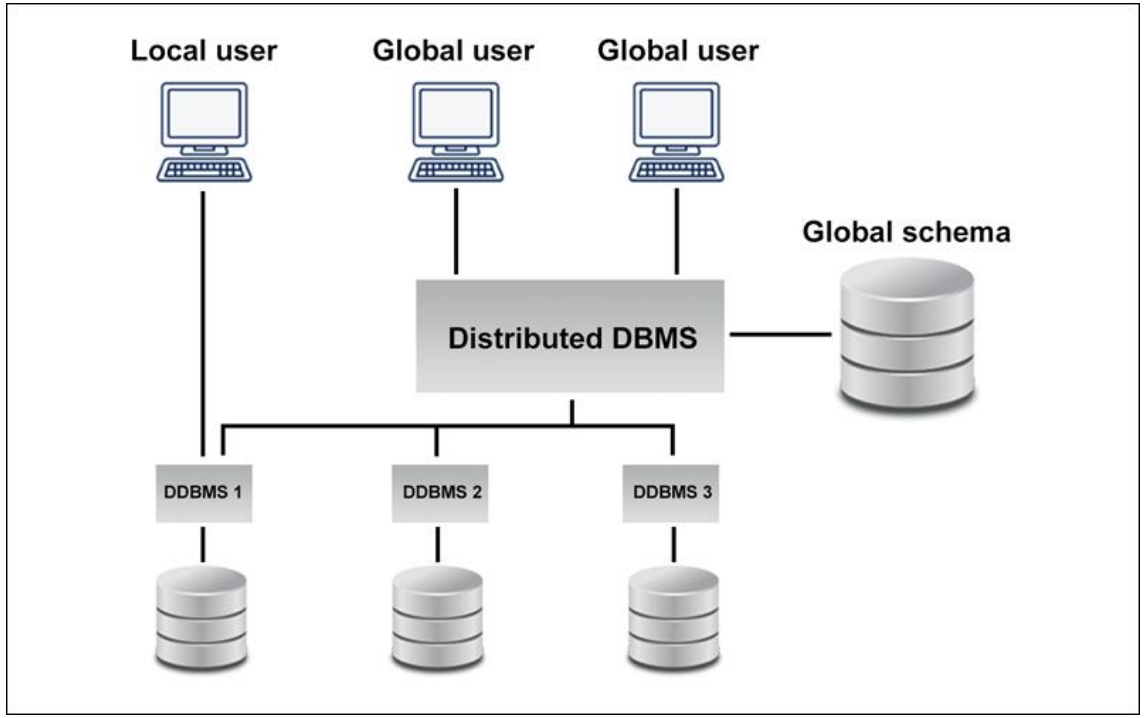
Dağıtık Sistemler

Dağıtılmış bir sistem, bir bilgisayar ağı (yerel ağ veya geniş alan ağı) aracılığıyla iletişim kuran birden çok bilgisayar ve yazılım bileşeninden oluşur. Dağıtılmış bir sistem, ana bilgisayarlar, iş istasyonları, kişisel bilgisayarlar vb. gibi herhangi bir sayıda olası yapılandırmadan oluşabilir. Bilgisayarlar birbirleriyle etkileşime girer ve ortak bir hedefe ulaşmak için sistemin kaynaklarını paylaşır.



Dağıtık Sistemler

- Genel olarak, dağıtılmış veritabanları aşağıdaki özellikleri içerir:
- Konumdan bağımsız
- Dağıtılmış sorgu işleme
- Dağıtılmış işlem yönetimi
- Donanımdan bağımsız
- İşletim sisteminden bağımsız
- ağdan bağımsız
- İşlem şeffaflığı
- DBMS'den bağımsız



Dağıtılmış Bilgi İşlemin Avantajları

Güvenilirlik (hata toleransı):

- Dağıtılmış bilgi işlem sisteminin önemli avantajı güvenilirliktir. Sistemdeki bazı makineler çökerse, geri kalan bilgisayarlar etkilenmez ve çalışma durmaz.

Ölçeklenebilirlik:

- Dağıtılmış hesaplamada sistem, gerektiğinde daha fazla makine eklenerek kolayca genişletilebilir.

Kaynakların Paylaşımı :

- Paylaşılan veriler, bankacılık, rezervasyon sistemi gibi birçok uygulama için esastır. Veriler veya kaynaklar dağıtılmış sistemde paylaşıldığı için diğer kaynaklar da paylaşılabilir (örneğin pahalı yazıcılar).

Dağıtılmış Bilgi İşlemin Avantajları

Esneklik :

Sistem çok esnek olduğu için yeni servisleri kurmak, uygulamak ve hata ayıklamak çok kolaydır.

Hız :

Dağıtılmış bir bilgi işlem sistemi daha fazla bilgi işlem gücüne sahip olabilir ve hızı onu diğer sistemlerden farklı kılar.

Açık sistem :

Açık sistem olduğundan, her hizmete yerel veya uzak her müşteriye eşit olarak erişilebilir.

Performans :

Sistemdeki işlemcilerin toplanması, merkezi bir bilgisayardan daha yüksek performans (ve daha iyi fiyat/performans oranı) sağlayabilir.

Dağıtık Sistemler

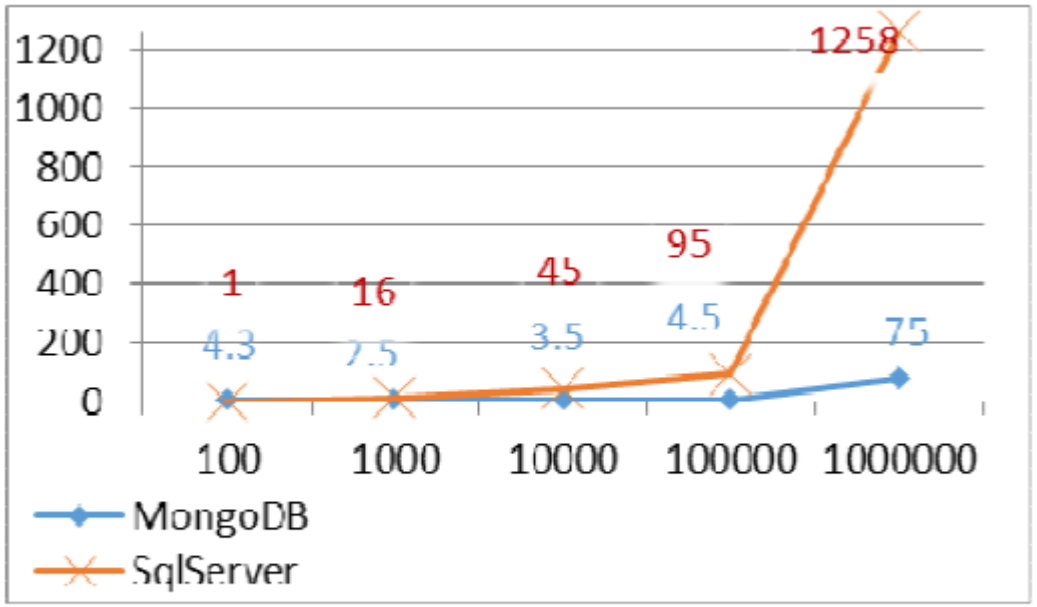
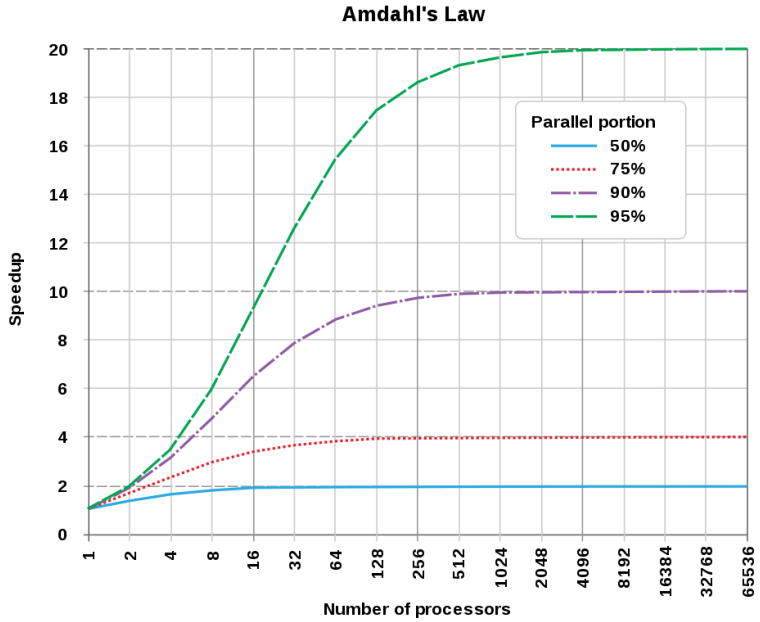


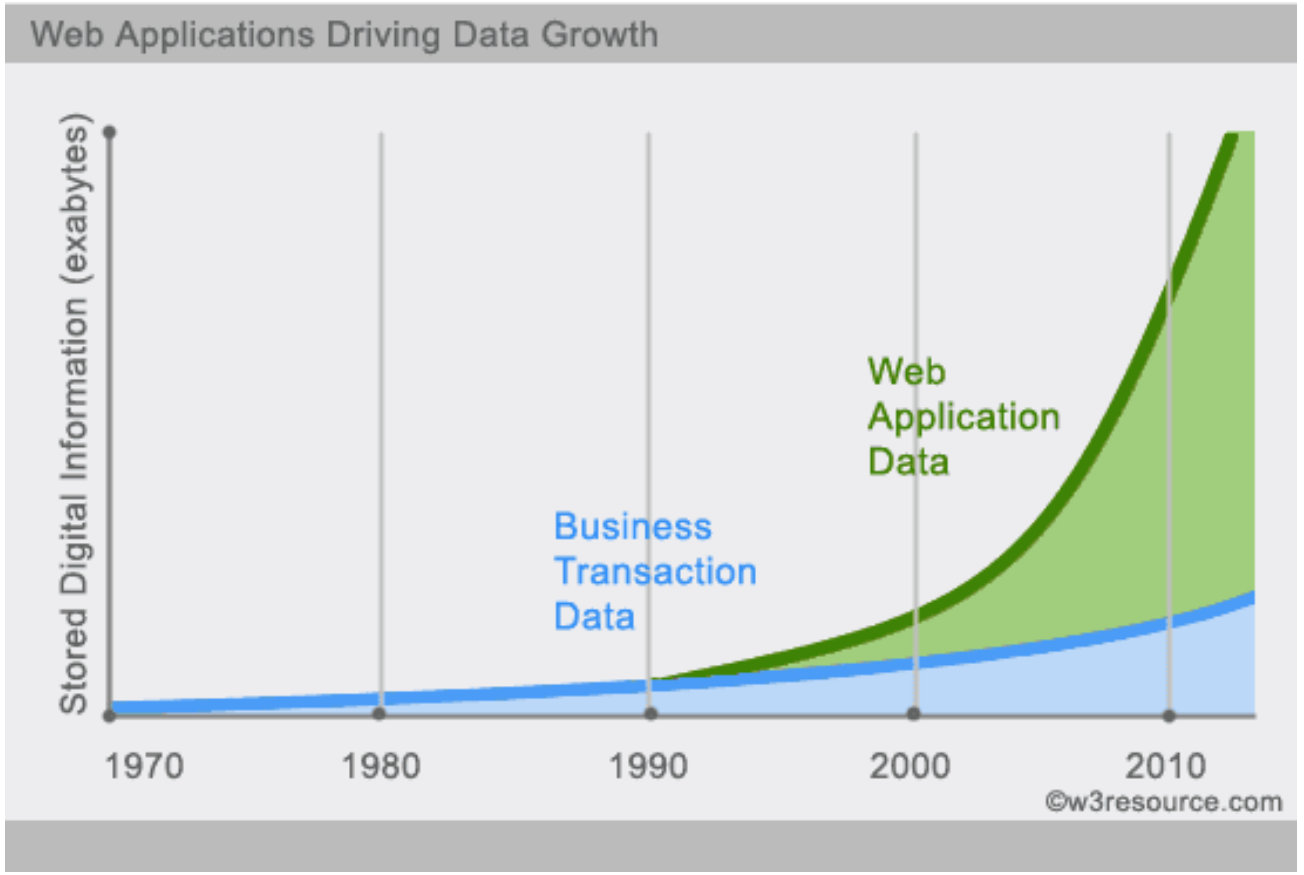
Fig. 5. Comparing the performance of deleting the records at different scales (times are in milliseconds)



$$\text{Overall Speedup} = \frac{\text{Old execution time}}{\text{New execution time}}$$

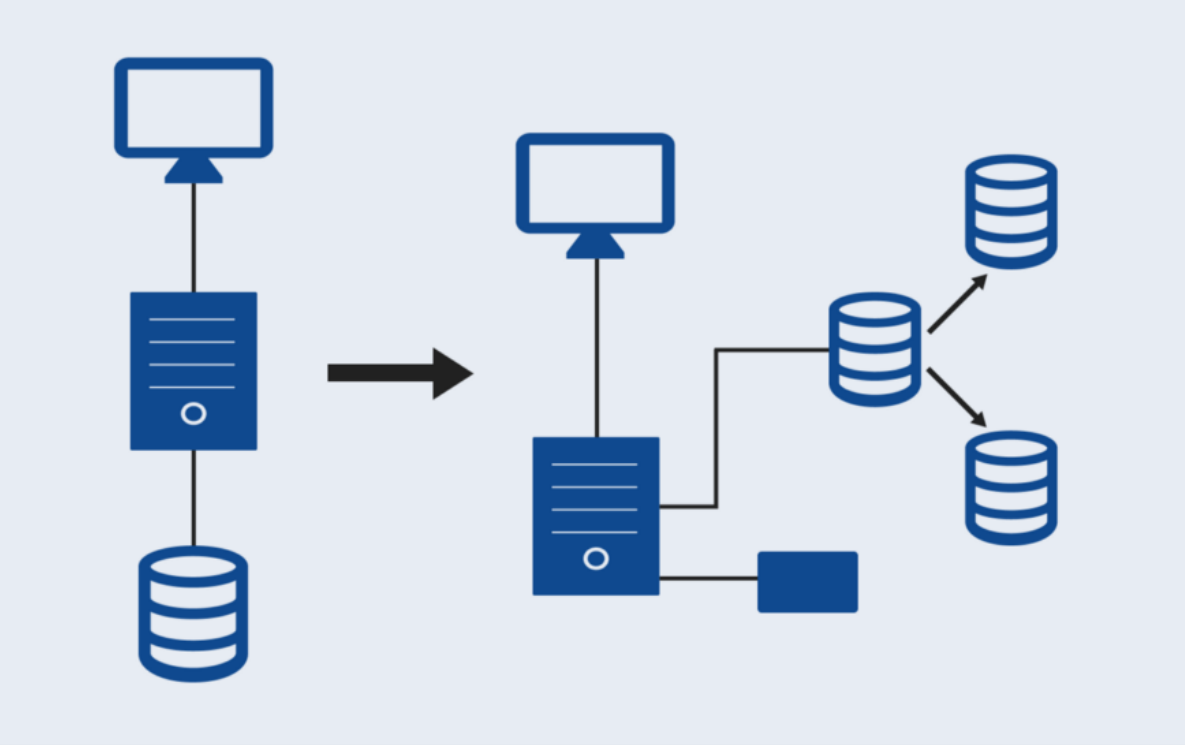
$$= \frac{1}{\left(1 - \text{Fraction}_{\text{enhanced}} \right) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

NoSQL


Web Applications Driving Data Growth
Stored Digital Information (exabytes)
Web Application Data
Business Transaction Data
1970
1980
1990
2000
2010
©w3resource.com

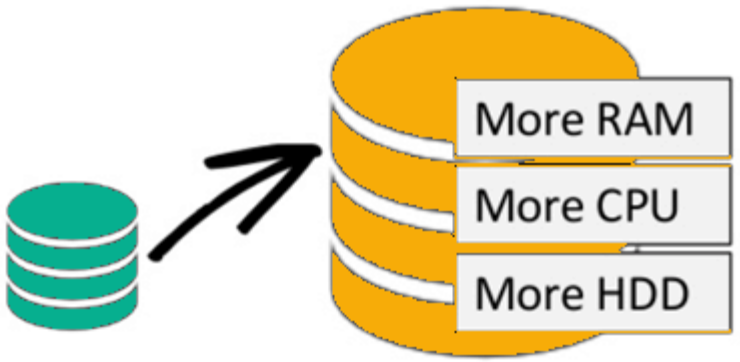
Ölçeklendirme

- Elektronikte (donanım, iletişim ve yazılım dahil), ölçeklenebilirlik, bir sistemin iş ihtiyaçlarınızı karşılayacak şekilde genişleme yeteneğidir. Örneğin, bir web uygulamasını ölçeklendirmek, uygulamanızı daha fazla kişinin kullanmasına izin vermekle ilgilidir. Uygulamayı çok fazla değiştirmeden veya fazladan donanım ekleyerek mevcut donanımı yükselterek bir sistemi ölçeklendirirsiniz.

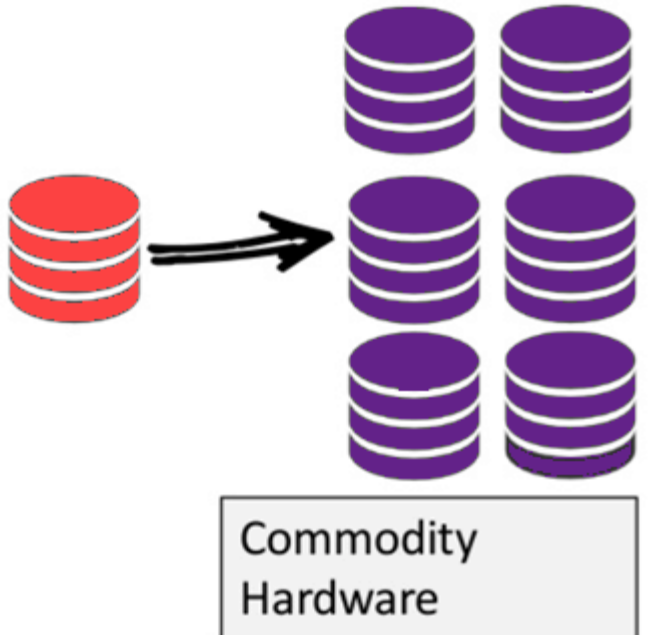


Dağıtık Sistemler

Scale-Up (*vertical* scaling):



Scale-Out (*horizontal* scaling):

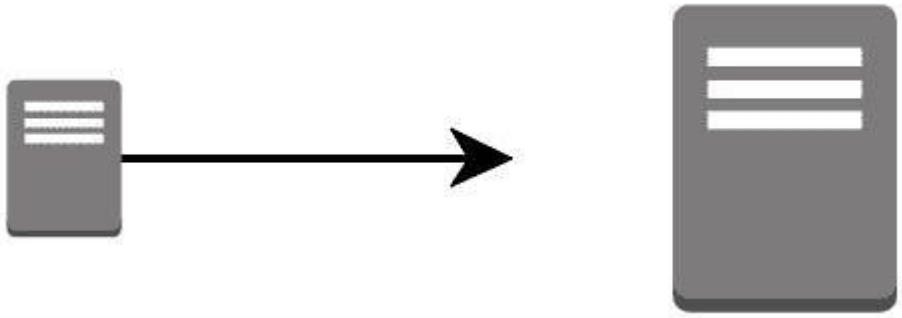


Ölçeklendirme

Dikey ölçeklendirme Dikey

Dikey ölçeklendirme , tek bir sunucunun veya kümenin işlem gücünü artırmayı ifade eder. Hem ilişkisel hem de ilişkisel olmayan veritabanları ölçeklenebilir, ancak nihayetinde maksimum işlem gücü ve çıktı açısından bir sınır olacaktır. Ek olarak, maliyetler doğrusal olarak ölçeklenmediğinden, yüksek performanslı donanıma kadar ölçeklendirme ile artan maliyetler vardır.

.



Ölçeklendirme

Yatay ölçeklendirme Yatay

- olarak ölçeklemek (veya ölçeği genişletmek), dağıtılmış bir yazılım uygulamasına yeni bir bilgisayar eklemek gibi bir sisteme daha fazla düğüm eklemek anlamına gelir. NoSQL sisteminde, bir sisteme daha fazla düğüm eklemek ve yükü bu düğümler arasında dağıtmak anlamına gelen "ölçeklendirme" avantajından yararlandığı için veri deposu çok daha hızlı olabilir.

NoSQL

Günümüzde ağırlıklı olarak ilişkisel veritabanı yönetim sistemleri kullanılmaktadır. Büyük veri için ise bu sistemlere alternatif olarak NoSQL veritabanları önerilmiştir. NoSQL veritabanları, açılımı “Not Only SQL” olan, ilişkisel olmayan, farklı tipteki verilerin hızlı organize edilmesini sağlayan dağıtık sistemlerdir.

NoSQL, geleneksel ilişkisel veritabanı yönetim sistemlerinden bazı önemli yönlerden farklı olan, ilişkisel olmayan bir veritabanı yönetim sistemidir. Çok büyük ölçekte veri depolama ihtiyacı olan dağıtılmış veri depoları için tasarlanmıştır (örneğin, kullanıcıları için her gün terabit veri toplayan Google veya Facebook). Bu tür veri depolama, sabit şema gerektirmeyebilir, birleştirme işlemlerinden kaçınabilir ve tipik olarak yatay olarak ölçeklenebilir.

NoSQL

Büyük veri kavramının yaygınlaşması ve veri miktarındaki artışın hızlanması ile NoSQL veritabanlarının kullanımı artmaktadır. NoSQL veritabanları genellikle önceden tanımlı şema içermezler ve bu sayede kayıtlar birbirinden farklı alanlara sahip olabilirler.

İlişkisel veritabanı yönetim sistemleri milyonlarca aktif kullanıcının verilerinin yoğunluğa göre uygulama sunucularına bölünüp saklanması konusunda uygun değildir. NoSQL veritabanları büyük boyutlu işlemlerde, büyük veri setlerine düşük gecikme ile erişimde avantajlıdırlar. Performans sağlamak amacı ile **ACID** prensibinin tümünü sağlamazlar fakat karşılığında milyonlarca kullanıcıya hizmet verebilirler. Buna örnek olarak Facebook ile kullanılan Cassandra örnek verilebilir

NoSQL'in kısa tarihi

NoSQL terimi 1998 yılında Carlo Strozzi tarafından icat edildi. Bu terimi SQL arayüzü olmayan Açık Kaynak, Hafif, Veri Tabanı olarak adlandırmak için kullandı.

2009'un başlarında, last.fm açık kaynaklı dağıtılmış veritabanları üzerine bir etkinlik düzenlemek istediğinde, Rackspace çalışanı Eric Evans, terimi ilişkisel olmayan, dağıtılmış ve atomiklik, tutarlılık ile uyumlu olmayan veritabanlarına atıfta bulunmak için yeniden kullandı. , izolasyon, dayanıklılık - geleneksel ilişkisel veritabanı sistemlerinin dört belirgin özelliği.

Aynı yıl ABD'nin Atlanta kentinde NoSQL'de düzenlenen "no:sql(east)" konferansı çok konuşuldu ve tartışıldı.

Ardından, NoSQL'in tartışılması ve uygulanması bir ivme kazandı ve NoSQL benzeri görülmemiş bir büyüme gördü.

NoSQL veritabanı özellikleri

Her NoSQL veritabanının kendine has özellikleri vardır. Yüksek düzeyde, birçok NoSQL veritabanı aşağıdaki özelliklere sahiptir:

- Esnek şemalar
- yatay ölçekleme
- Veri modeli sayesinde hızlı sorgular
- Geliştiriciler için kullanım kolaylığı

NoSQL veritabanı özellikleri

```
{
  _id: <ObjectId1>,
  username: "123xyz",
  contact: {
             phone: "123-456-7890",
             email: "xyz@example.com"
           },
  access: {
            level: 5,
            group: "dev"
          }
}
```

Embedded sub-document

Embedded sub-document

user document

```
{
  _id: <ObjectId1>,
  username: "123xyz"
}
```

contact document

```
{
  _id: <ObjectId2>,
  user_id: <ObjectId1>,
  phone: "123-456-7890",
  email: "xyz@example.com"
}
```

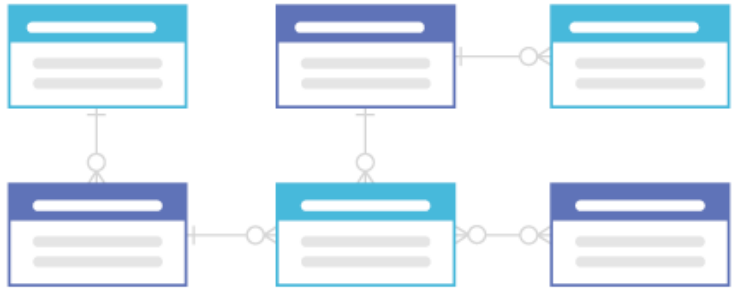
access document

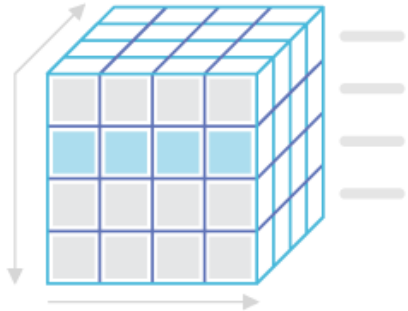
```
{
  _id: <ObjectId3>,
  user_id: <ObjectId1>,
  level: 5,
  group: "dev"
}
```

NoSQL

SQL

Relational Database Management Systems (RDBMS)

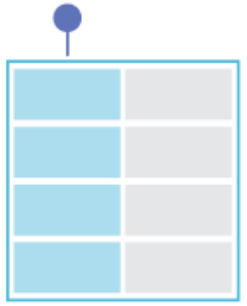




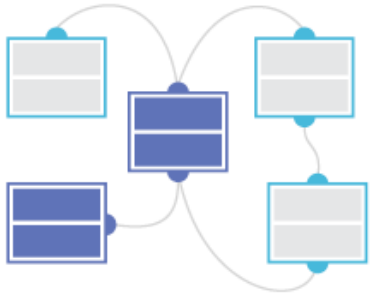
Online Analytical Processing (OLAP) Cube

NoSQL

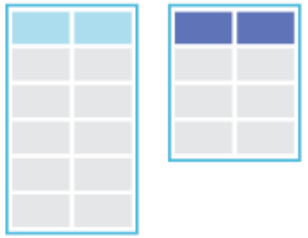
Key-Value



Graph



Document

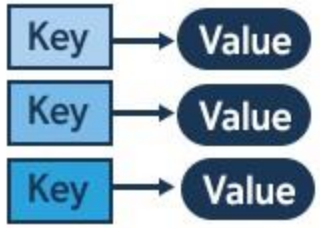


Column store

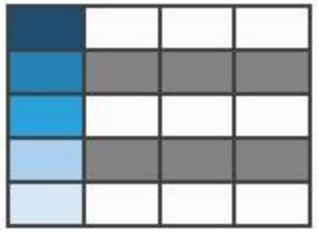
NoSQL veritabanlarının türleri

NoSQL

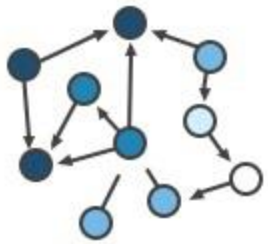
Key-Value



Column-Family



Graph



Document

NoSQL veritabanlarının türleri

NoSQL veritabanları 4 ana kategoride incelenebilir

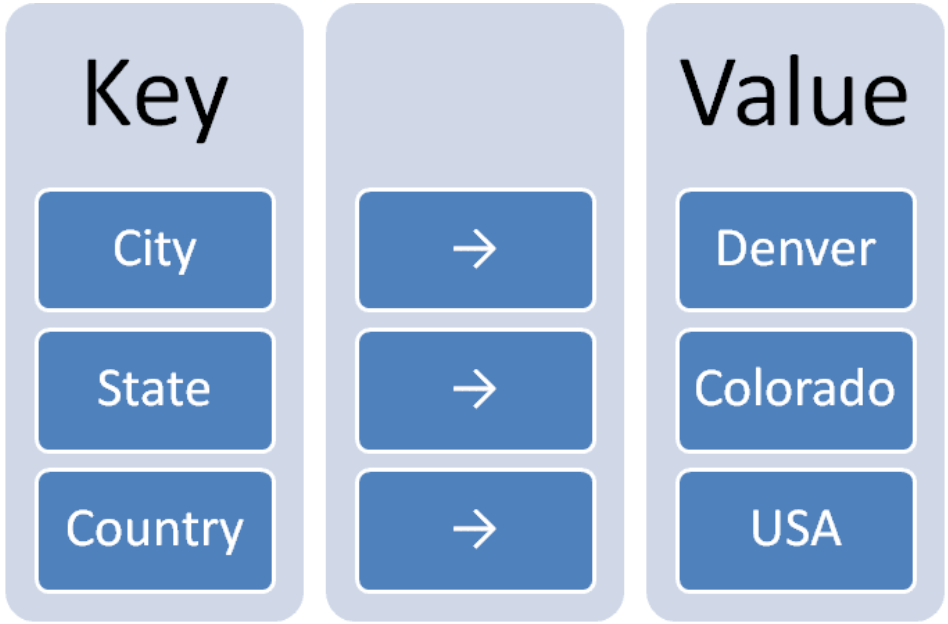
A. Anahtar-Değer Deposu (Key-Value Store)

En basit NoSQL veritabanı türüdür. Bir anahtar ve anahtarın belirlediği veriden oluşur. Verinin içeriğinin herhangi bir önemi yoktur. Veri bir BLOB olarak kabul edilir ve anahtar ile ilişkilendirilerek veritabanında saklanır. İstendiğinde anahtar ile veri elde edilebilir veya silinebilir. Anahtar-değer depolarına örnek olarak LevelDB ve Oracle NoSQL Database verilebilir.

Geleneksel ilişkisel veritabanları, anahtar-değer depolarının öne çıktığı yüksek hacimli okuma/yazma işlemlerini işlemek için tasarlanmamıştır. Kolayca ölçeklenebilir olduğundan, anahtar/değer çifti herhangi bir saniyede binlerce kullanıcıyı işleyebilir.

NoSQL veritabanlarının türleri

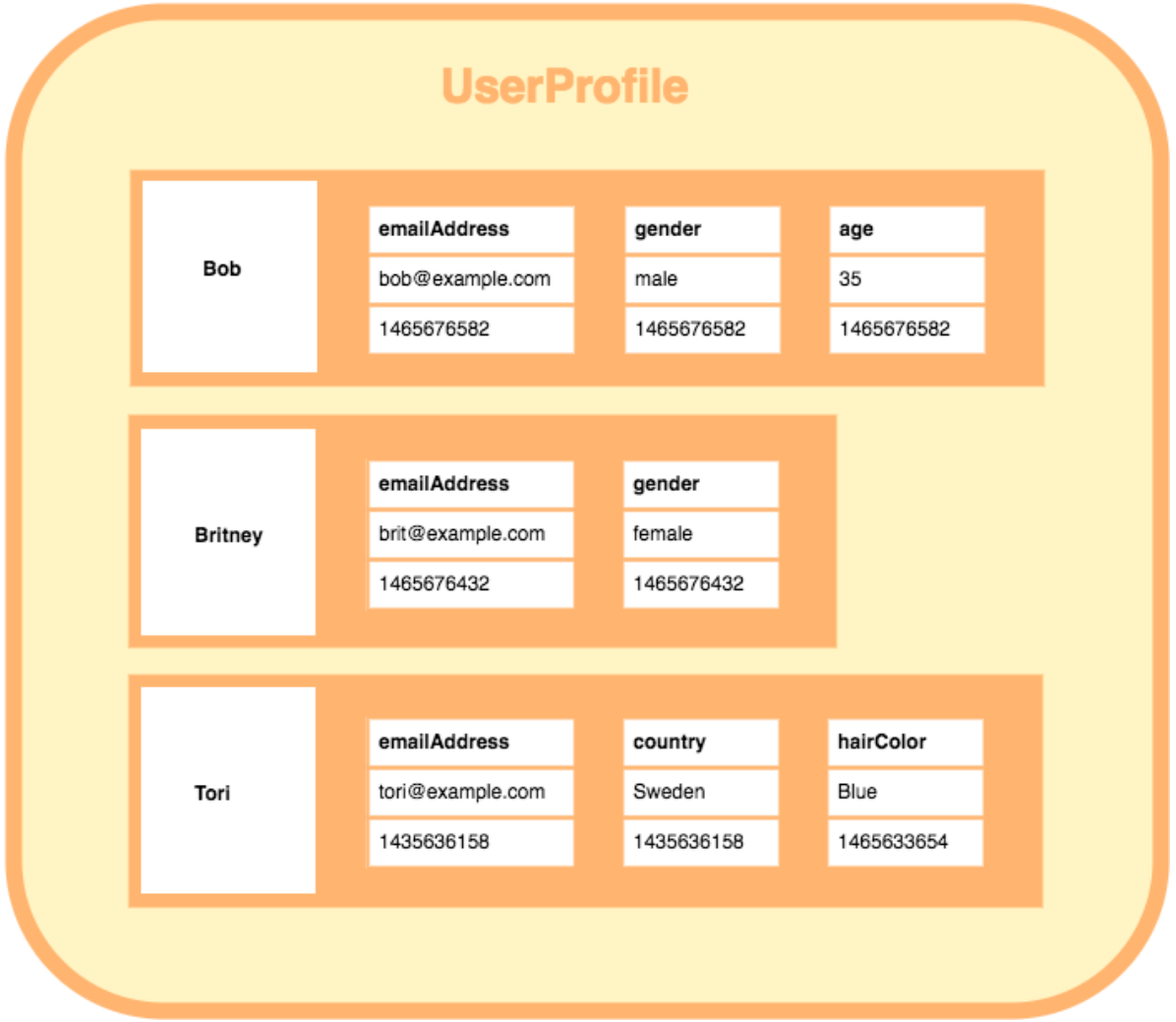
A. Anahtar-Değer Deposu (Key-Value Store)



NoSQL veritabanlarının türleri

B. Sütun Deposu (Column Store)

Sütun bazlı veritabanları temelde her anahtarın birden fazla anahtar – değer ikilisine sahip olduğu iki boyutlu dizilerdir. Her satır bir anahtara karşılık birden fazla sütun içerir ve sütunlar satır içerisinde sütun anahtarları ile sıralanırlar. Cassandra [9] ve BigTable [10] sütun depolarına örnek olarak verilebilir.



NoSQL veritabanlarının türleri

C. Belge Veritabanı (Document Database)

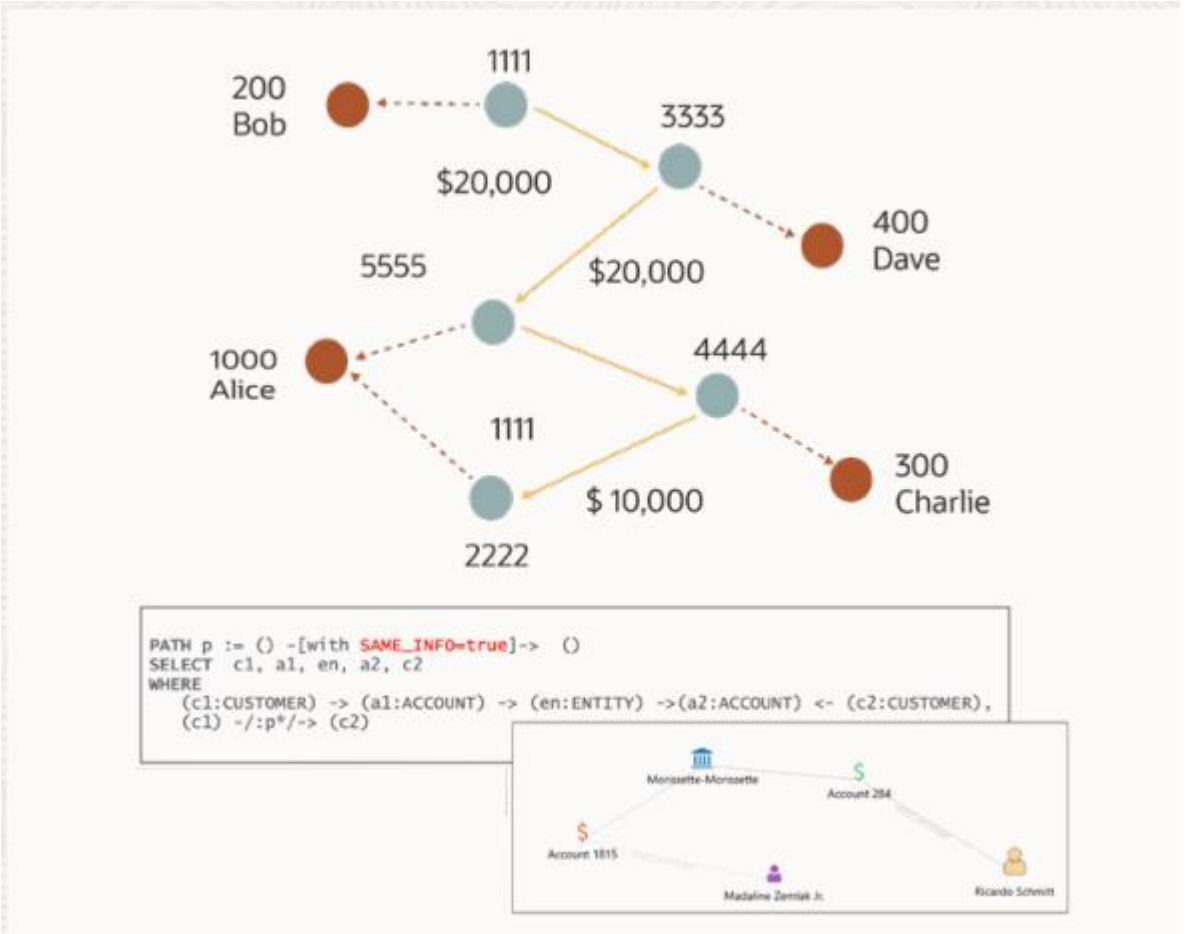
Belge Veritabanı XML, JSON gibi hiyerarşik ve tanımlı belgeleri saklar. Anahtar-değer depolarında olduğu gibi anahtara karşılık bir değer bulunur. Fakat buradaki değer bir belgedir ve anahtar-değer deposundan farklı olarak içeriği bilinebilir ve sorgulanabilir. En çok bilinen belge veritabanları MongoDB ve CouchDB [11]'dir.

```
{
  "_id": "5cf0029caff5056591b0ce7d",
  "firstname": "Jane",
  "lastname": "Wu",
  "address": {
    "street": "1 Circle Rd",
    "city": "Los Angeles",
    "state": "CA",
    "zip": "90404"
  }
  "hobbies": ["surfing", "coding"]
}
```

NoSQL veritabanlarının türleri

D. Çizge Veritabanı (Graph Database)

Çizge veritabanlarında varlıklar ve bu varlıkların arasındaki ilişkiler, düğümler ve bu düğümler arasındaki kenarlar biçiminde saklanır. Özellikle karmaşık hiyerarşik yapıların taranması ve bu yapılardan bilgi elde edilmesi için kullanılması uygundur. Neo4J ve OrientDB çizge veritabanlarına örnek olarak verilebilir.



RDBMS ve NoSQL veritabanları arasındaki fark

İlişkisel veritabanı yönetim sistemleri (RDBMS) ve NoSQL veritabanları arasında çeşitli farklılıklar olsa da, temel farklılıklardan biri verilerin veritabanında modellenme şeklidir. Bu bölümde, bir ilişkisel veritabanında ve bir NoSQL veritabanında aynı verileri modellemenin bir örneğini inceleyeceğiz. Ardından, ilişkisel veritabanları ile NoSQL veritabanları arasındaki diğer bazı önemli farklılıkları vurgulayacağız.

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
}
```

RDBMS ve NoSQL veritabanları arasındaki fark

RDBMS

- Yapılandırılmış ve organize veriler
- Yapılandırılmış sorgu dili (SQL)
- Veriler ve ilişkileri ayrı tablolarda depolanır.
- Veri Manipülasyon Dili, Veri Tanımlama Dili
- Sıkı Tutarlılık

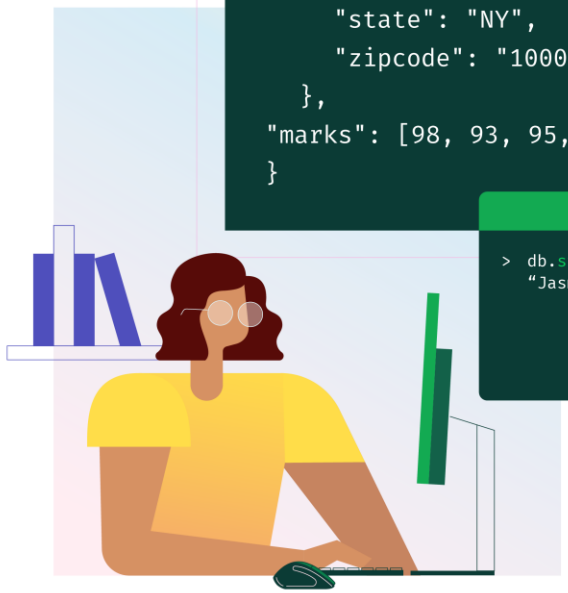
NoSQL

- Sadece SQL değil
- Bildirimsel sorgu dili yok
- Önceden tanımlanmış şema yok
- ACID özelliği yerine nihai tutarlılık
- Yapılandırılmamış ve öngörülemeyen veriler
- CAP Teoremi
- Yüksek performansa, yüksek kullanılabilirlik ve ölçeklenebilirliğe öncelik verir.
- BASE İşlemi

RDBMS ve NoSQL veritabanları arasındaki fark

MongoDB

```
{
"_id": 1,
"student_name": "Jasmin Scott",
  "school": {
     "school_id": 226,
     "name": "Tech Secondary",
     "address": "100 Broadway St",
     "city": "New York",
     "state": "NY",
     "zipcode": "10001"
  },
"marks": [98, 93, 95, 88, 100],
}
```



```
mongo
> db.students.find({“student_name”: “Jasmin Scott”})
```

SQL

students

id	name	school_id
1	Jasmin Scott	226
...	...	...

marks

id	student_id	mark
10	1	98
...	...	...

school_details

id	name	address	city	state	zipcode
226	Tech Secondary	100 Broadway St	New York	NY	10001
...	...	...	...	...	...

Results

name	mark	school_name	city
Jasmin Scott	98	Tech Secondary	New York
...	...	...	...

```sql
SELECT s.name, m.mark, d.name as "school name",
d.city
FROM students s
INNER JOIN marks m ON s.id = m.student_id
INNER JOIN school_details d ON s.school_id = d.id
WHERE s.name = "Jasmin Scott";
```

RDBMS ve NoSQL veritabanları arasındaki fark

RDBMS ve NoSQL: Veri Modelleme Örneği

Bir kullanıcı ve hobileri hakkında bilgi depolamanın bir örneğini ele alalım. Bir kullanıcının adını, soyadını, cep telefonu numarasını, şehrini ve hobilerini kaydetmemiz gerekiyor.

İlişkisel bir veritabanında, muhtemelen iki tablo oluştururuz: biri Kullanıcılar ve diğeri Hobiler için.

Kullanıcılar

İD	ilk adı	Soyadı	hücre	Kent
1	Leslie	evet	8125552344	Pawnee

Hobiler

İD	Kullanıcı kimliği	hobi
10	1	karalama defteri
11	1	waffle yemek
12	1	Çalışma

RDBMS ve NoSQL veritabanları arasındaki fark

RDBMS ve NoSQL: Veri Modelleme Örneği

Bir kullanıcı ve hobileri hakkında tüm bilgileri almak için, Kullanıcılar tablosundaki ve Hobiler tablosundaki bilgilerin birleştirilmesi gerekir.

Bir NoSQL veritabanı için tasarladığımız veri modeli, seçtiğimiz NoSQL veritabanının türüne bağlı olacaktır. Bir kullanıcı ve hobileri hakkında aynı bilgilerin MongoDB gibi bir belge veritabanında nasıl saklanacağını düşünelim.

```
{
   "_id": 1,
   "first_name": "Leslie",
   "last_name": "Yepp",
   "cell": "8125552344",
   "city": "Pawnee",
   "hobbies": ["scrapbooking", "eating waffles", "working"]
}
```

NoSQL

Soyadi	Adi	Yas
AYDIN	Onur Ekin	16
AYDIN	Efe Çınar	9
AYDIN	Gürkan	42

```
{
    "_id": ObjectId("124efa8d2b7d284dad101e4bc8"),
    "Last Name": " AYDIN ",
    "First Name": "Onur Ekin",
    "Age": 16,
    "Adres": "Bulgurlu Mh. Karlıdere Cd.",
    "Sehir": "İstanbul"
},
```

NoSQL

	JSON	BSON
kodlama	UTF-8 Dize	İkili
Veri Desteği	Dize, Boole, Sayı, Dizi	String, Boolean, Number (Integer, Float, Long, Decimal128...), Array, Date, Raw Binary
okunabilirlik	İnsan ve Makine	Sadece Makine

RDBMS ve NoSQL veritabanları arasındaki fark

RDBMS	MongoDB	Neo4j
Table	Collection	Graph
Row	Document	Node
Column	Field	Property
Join	Embedded Document	Traversal

JSON (JavaScript Object Notation)

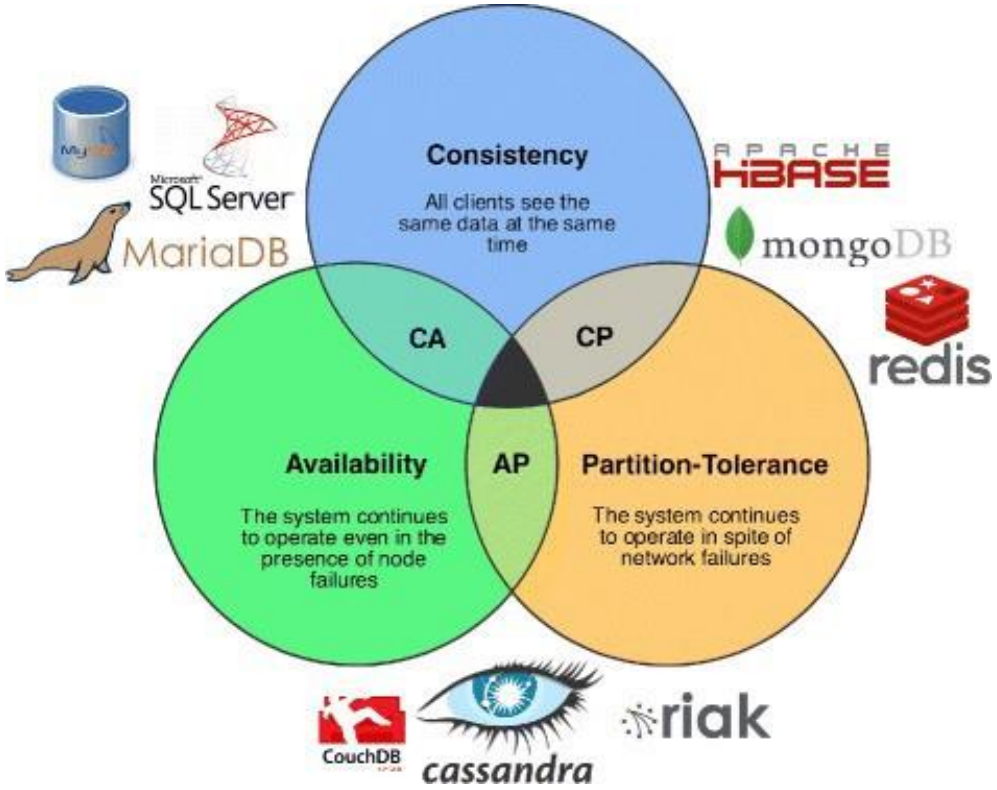
- Genel Bakış
- JSON – Sözdizimi
  - Veri Türleri
  – Nesneler
  – Şeması
  - XML ile karşılaştırma

CAP Teoremi

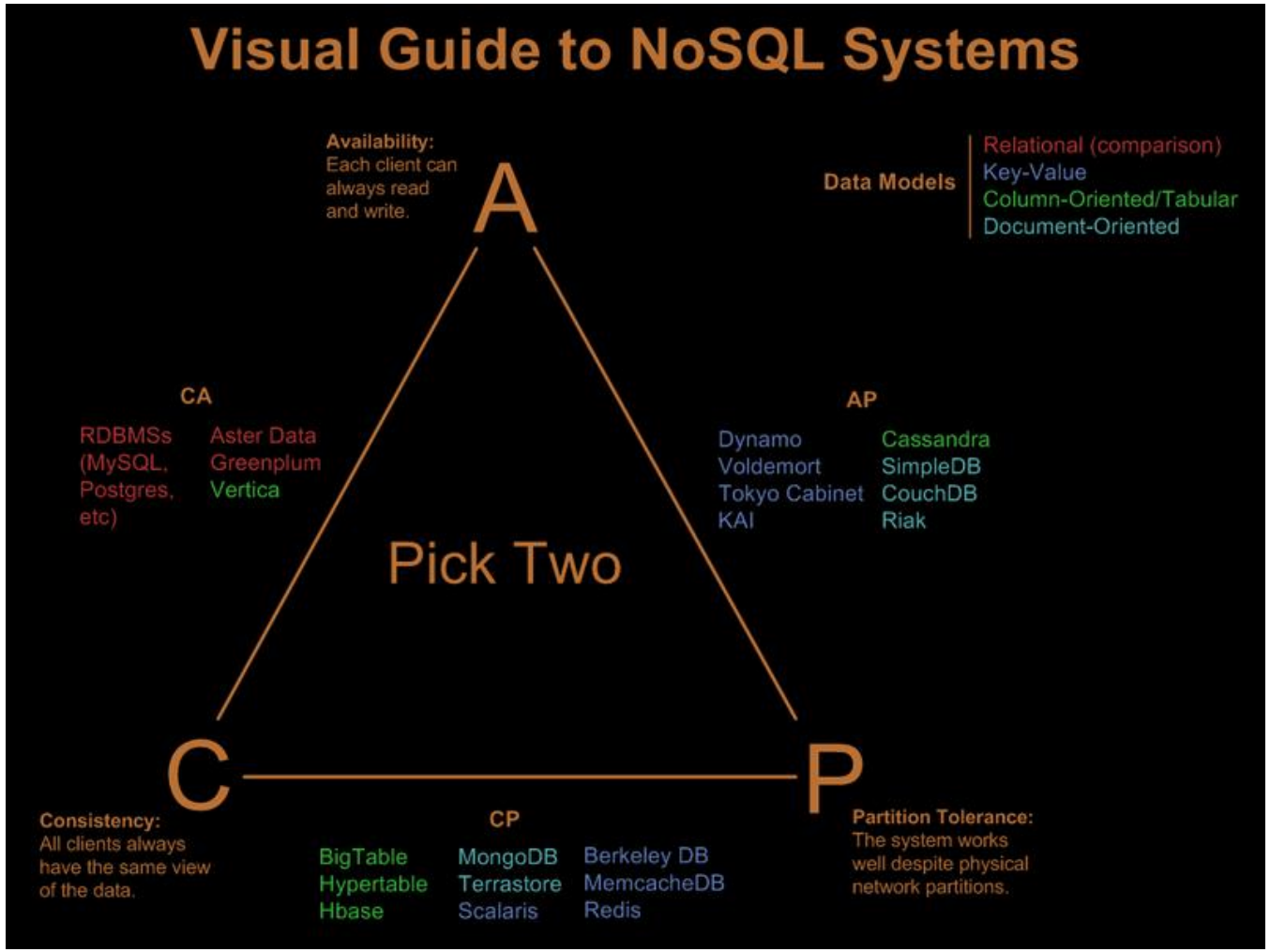
NoSQL veritabanları CAP teoremindeki maddelere uyum sağlamalıdır. CAP teoremi (Brewer teoremi) dağıtık bilgisayar sistemlerinde aşağıda verilmiş 3 ana maddenin tümünün sağlanmasının mümkün olmadığını savunur [7].

- Consistency (Veri Bütünlüğü)
- Availability (Erişilebilirlik)
- Partition tolerance (Bölüm Töleransı)

Çoğu NoSQL veritabanı bu özelliklerin ikisini sağlamayı hedefler.



CAP Teoremi


Visual Guide to NoSQL Systems
Availability:
Each client can always read and write.
A
Data Models
Relational (comparison)
Key-Value
Column-Oriented/Tabular
Document-Oriented
CA
RDBMSs (MySQL, Postgres, etc)
Aster Data
Greenplum
Vertica
AP
Dynamo
Voldemort
Tokyo Cabinet
KAI
Cassandra
SimpleDB
CouchDB
Riak
Pick Two
C
P
Consistency:
All clients always have the same view of the data.
CP
BigTable
Hypertable
Hbase
MongoDB
Terrastore
Scalaris
Berkeley DB
MemcacheDB
Redis
Partition Tolerance:
The system works well despite physical network partitions.

İÇERİK