**14.HAFTA**

**NOT ONLY SQL:** Kastedilen şey not relational, ilişkisel olmayan veri tabanları.

Örnekler: MongoDB CouchDB OrientDB …

**ÖZELLİKLER**

Ölçeklenebilir: yatay ve dikey ölçeklendirme

Replication models**: 1) Master-slave** (Kopyanın ana kopyasına ihtiyaç var. Kopyalar ilk etapta master da var olmalı ardından iki taraftada. Bütün okumalar ana kopyada ya da slave de olabilir. **2) Master-master**

Dosyaların parçalanması : Aynı anda birden fazla kullanıcı trf erişilebilir olmalı.

Veri Erişiminde yüksek performans: Hashing veya aralıklı bölmeleme ile…

Şemaya ihtiyaç duymaz: Tek tabloda tutulur.

Sıralı dillerden daha az güçlü:

Versioning:

**KATEGORİLER**

1. Document-based NOSQL systems
2. NOSQL key-value stores
3. Column-based or wide column NOSQL systems
4. Graph-based NOSQL systems
5. Hybrid NOSQL systems
6. Object databases
7. XML databases

**CAP TEORİSİ**

Eğer dağıtık bir yapı söz konusu ise Tutarlılık, Erişilebilirlik ve Bölüm Toleransı aynı anda garanti edilemez, sağlanamaz. 3 özellikten 2 si tercih edilmelidir. Erişebilirlik ve Tolerans ön plandadır. Tutarlılık anında değil nihayetinde ortaya çıkacağı için…

**KEY-VALUES**

**JSON**

Nest yapısı ne zaman yapılabilir?

* Bire bir veya bire çoksa

**VERİ MODELLEME ve EVRAK TABANLI VERİ TABANI**

Collection fiziksel veri tabanında table manasına geliyor. Dökümanlarsa row manasına gelir.

Document Data Base verileri Document yani evrak olarak tutar. Döküman bir nesnenin nitliklerini taşır.

Bir dökümanın fieldleri vardır. Bu fieldlar salt değer, değerler dizini veya döküment içerisinde döküment olabilir.

Field veri tipi dökümandan dökümana değişebilir.

Benzer evrakları bir collectionda toplamakta fayda var.

Bire çok ilişki:

1. Embed
2. Reference

Çoka çok ilişki:

Duplication çoksa reference tercih edilir.

**PATTERNS**

**Subset pattern :** problem, döküman büyükse . Veri tabanı diskte yer alır. Erişme istedğim zaman veri hacmi büyükse sıkıntılar oluşur. 2 ye ayrılır sık ve azs erişilen veri olarak. Sık erişilen verileri working set belirleyerek kullanırız.

**Attribute pattern** : bir yapıda benzer fieldlar var ise (3 ayrı tabloda da price ve name fieldi var mesela)

**Schema Versioning pattern :** Sistemi kapatmadan bir deployment yapmamız gerekiyorsa. İşlem devam ederken başka bir işlem devam ettirir.

**Extended Reference Pattern**: JOIN den kaçınabilmek için kullanılır. Kolay erişim için. Erişilmesi sık istenen fieldlar.

**Bucket Pattern:** Kova yapısı. Erişim için belirli sütunları seçiyoruz. Büyük bir havuzdan bir kova alarak çalışıyoruz.

**Tree Patterns:** Parent veya child a gitmek istiyorsun.

Patterns ları incelerken, seçerken bakmamız gerekenler:

1. Duplication
2. Data Staleness(Verinin güncellenmeden kalma süresi, sık erişilmeyen tarihi geçmiş veri)
3. Data Referntial Integrity(Collectionlar arasındadır.)

**Handling Duplication**

Embed ile dökümana daha hızlı erişilebilir.

Tekrar eden verilerdir.