# Modul 3 - Day 1

## Veritabanları (Databases)

Veri tabanları birbirleriyle ilişkili bilgilerin depolandığı alanlardır. Bilgi artışıyla birlikte bilgisayarda bilgi depolama ve bilgiye erişim konularında yeni yöntemlere ihtiyaç duyulmuştur. Veri tabanları; büyük miktardaki bilgileri depolamada geleneksel yöntem olan “dosya-işlem sistemine” alternatif olarak geliştirilmiştir.

### İlişkisel Veritabanları (Relational Databases)

**MySQL:** Çoklu iş parçacıklı (multi-threaded), çok kullanıcılı (multi-user), hızlı ve sağlam (robust) bir veri tabanı yönetim sistemidir.

**PostgreSQL**, veritabanları için ilişkisel modeli kullanan ve SQL standart sorgu dilini destekleyen bir veritabanı yönetim sistemidir. PostgreSQL aynı zamanda iyi performans veren, güvenli ve geniş özellikleri olan bir Veri Tabanı Yönetim Sistemi'dir.  PostgreSQL ücretsiz ve açık kodludur.

**MariaDB** ilişkisel veritabanı sistemi olan MySQL'in kaynak kodundan türemiş, GNU Genel Kamu Lisansı altında dağıtılarak ücretsiz olarak kullanılabilen, geliştirilmesi ve bakımı topluluk tarafından sürdürülen veritabanıdır. MySQL, önde gelen açık kaynaklı yazılım sistemi olarak ticari bir şirket olan Oracle tarafından satın alındıktan sonra MySQL'in ilk geliştiricileri tarafından Monty AB çatısı altında yine açık kaynak olarak MariaDB adıyla yola devam edeceği duyurulmuş ve oldukça ilgi görmüştür.

### Key, Primary Key, Unique Key ve Foreign Key Tanımlamaları (Constrains)

***Kaynak***: *https://ceaksan.com/tr/primary-unique-foreign-key*, https://medium.com/gokhanyavas/constraints-kullanımı-26bb89dbcd2b

🔥 **Key (anahtar)** veri tabanı içeriğinde, bir veya daha fazla alanın bir satırı nitelemesi amacıyla tanımlanması olarak özetlenebilir. Bir anahtar farklı amaçlar doğrultusunda kullanılabilecek **Unique Key(Tekil Anahtar), Primary Key(Birincil Anahtar)** ve **Foreign Key (Yabancı Anahtar)** gibi özel bir tip ile tanımlanabilir.

1. Primary Key Constraint
2. Unique Constraint
3. Foreign Key Constraint
4. Default Constraint
5. Check Constraint

## Primary Key Constraint

Birincil anahtar kısıtlayıcısıdır. Her tabloda bir adet bulunabilir. Girilen her değerin farklı olması anlamına gelmektedir. Yani eşsiz kayıtlar tutmakta kullanılır. Bu alanlar NULL değere sahip olamazlar. Genelde otomatik artan sayılar için kullanılırlar. Otomatik arttırma **Identity** komutuyla gerçekleştirilir. **Identity** komutundan sonra işlemin kaçtan başlayacağı ve kaçar kaçar artacağı belirtilir. Identity(1,1) 1'den başlayacağını ve 1'er 1'er artacağını gösterir.

## Unique Constraint

Tablodaki bir sütünün benzersiz olmasını istediğimiz durumlarda kullanırız. Örnek vermek gerekirse, T.C Kimlik numaraları, Banka Hesap Numaraları gibi vs. Primary key den farkı ise Unique key bir tabloda birden fazla olmasıdır, primary key ise tabloda sadece 1 adet olabilir. Unique olarak tanımlanmış bir alan NULL olabilir. Değeri NULL'dan farklı olacak olursa kesinlikle daha önce girilen değerlerden farklı olmak zorundadır.

## Foreign Key Constraint

Yabancıl Anahtar anlamına gelen bu kısıtlayıcı tabloları ilişkilendirme de kullanılır. Yabancıl anahtar ile kısıtlanan tablodaki sütun diğer tablodaki sütun ile kısıtlanmış olur. Kod tarafında eşleştirme işlemi aşağıdaki biçimde yapılır;

## Default Constraint

Varsayılan kısıtlayıcı demektir. Tablodaki herhangi bir alan için girilmesi gereken değerin atanmasıdır. INSERT komutu için geçerlidir. Örnek olarak bir kayıt eklendiğinde, kaydın eklenme zamanını default olarak belirtebiliriz.

## Check Constraint

Kontrol Kısıtlayıcısı anlamına gelmektedir. Yani istediğimiz biçime göre verilerin girilmesini sağlar. Örneğin T.C Kimlik NO alanına 11 karakterin girilmesini sağlayabiliriz.

## Query

Türkçe karşılığı “sorgu” olan Query, basitçe, bilgi için yapılan bir istektir.

## SQL

*(İngilizce "Structured Query Language", Türkçe: Yapılandırılmış Sorgu Dili)* verileri yönetmek ve tasarlamak için kullanılan bir dildir. SQL, kendisi bir programlama dili olmamasına rağmen birçok kişi tarafından programlama dili olarak bilinir. SQL herhangi bir veri tabanı ortamında kullanılan bir alt dildir. SQL ile yalnızca veri tabanı üzerinde işlem yapılabilir; veritabanlarında bulunan sistemlere bilgi ekleme, bilgi değiştirme, bilgi çıkarma ve bilgi sorgulama için kullanılmaktadır. Özellikle de ilişkisel veritabanı sistemleri üzerinde yoğun olarak kullanılmaktadır. SQL'e özgü cümleler kullanarak veri tabanına kayıt eklenebilir, olan kayıtlar değiştirilebilir, silinebilir ve bu kayıtlardan listeler oluşturulabilir.

## SQL Index

SQL indekslemenin amacı işlenen verinin daha az veri okunarak sorgu sonucunun daha kısa zamanda getirilmesini sağlamaktır. Indeksleme kullanarak tablonun tamamını okumaktansa oluşturacağımız indeks key i aracılığı ile okumak istediğimiz kayıda ulaşabilmemiz daha hızlı bir şekilde mümkün olacaktır. Bu sayede tamamlanması saatler süren sorgunun uygun indeksler kullanılarak saniyeler içinde getirilmesini sağlayabiliriz. *Kaynak: https://medium.com/fullstackturkiye/sql-indexing-nedir-nasıl-çalışır-588be1f43029*

## Yatay ve Dikey Ölçeklenebilirlik:

***Kaynak***: *http://www.ilterismutlu.com/yatay-vs-dikey-olceklenebilirlik-horizontally-vs-vertically-scalable-scalability/*

**Dikey Ölçeklenebilirlik Nedir ?**

Sistemin/Veritabanının Dikeyde ölçeklenebilir olması (dikey ölçeklenebilirlik, vertically scalable, scale up); bir tane çok güçlü aynı zamanda pahalı bir makine/donanım kullanılmasıdır. Dikey Ölçeklenebilir sistemlerde donanım kısıtları mevcuttur. Örneğin mevcut sisteminizin CPU frekansını 5 ghz yapamazsınız veya 1 tb ram yapamazsınız.

Yatay ölçeklenebilirlikte yüzlerce, binlerce makinelik server (sunucu) ağı kurulabilir. Yatay ölçeklenebilirliğin yönetimi dikey ölçeklenebilirliğe göre daha zordur.Sonuçta onlarca veya yüzlerce makinayı yönetmek, tabi ki tek bir makinayı yönetmekten daha zordur.

**Yatay Ölçeklenebilirlik Nedir ?**

Sistemin/Veritabanının Yatayda ölçeklenebilir olması (horizontally scalable, scale out); ucuz ve çok sayıda makinenin aynı anda kullanılması anlamına gelir. Yatay ölçeklenebilirlik sayesinde yedeklilik de performans artışı da sağlanabilir.

# ACID

***Kaynak:*** *https://medium.com/cloud-and-servers/acid-nedir-53f729f2bbb2*

ACID, ilişkisel veritabanlarındaki Transaction için tanımlanmış özellik setidir.

Transaction için olarak verilen örnek bir banka hesabından başka bir banka hesabına paranın transfer edilmesi olarak anlatılabilir. Burada 2 hesap gönderici ve alıcının hesabı üzerinde mantıksal bir operasyon gerçekleştiriliyor. Bu işleme ***Transaction*** deniyor.

Bu transaction başarılı bir şekilde gerçekleşebilmesi için ACID ilkelerine uyması gerekmektedir.

- Atomicity
- Consistency

- Isolation
- Durability

**Atomicity:** Transaction işlemini bir bütün olarak görür. İşlem sırasında birden fazla veritabanı/tablodaki verinin güncellenmesi gerçekleşiyor ise tüm bunların hepsi birden başarılı olacaktır veya başarısız olacaktır

- Veritabanları erişilemez olabilir.
- Network problemi olabilir.
- Herhangi bir hata oluşabilir.

⚠ Bu durumda işlem geçersiz sayılacaktır.

**Consistency(Tutarlılık) :** Transaction işlemi sonucunda veritabanındaki verinin geçerli durumunun, bir sonraki geçerli duruma geçmesidir. Özetle Transaction tam anlamı ile gerçekleşinceye kadar (constraints, cascades, triggers) işlemden etkilenen verilerin değerlerinin bir önceki geçerli değeri vermesidir.

**Isolation:** Aynı anda aynı veri üzerinde birden fazla Transaction değiştirme gereksinimi olabilir. Transaction'ların birbirlerinin işlemlerinden etkilenmemesi için işlemler Seri olarak yapılması gerekir. Transaction sırasında ilgili ve etkilenecek veri setleri kilitlenir. Taki işlem başarılı ve başarısız olarak sonuç dönünceye kadar.

**Durability(Dayanıklılık):** Transaction sırasında fiziksel veya işlemsel bir hata olması durumunda sistemin kendisini bir önceki geçerli veri durumuna döndürebilme kabiliyetidir.

## Veritabanı Normalleştirmesi

***Kaynak:*** *https://www.lifeacode.com/sql-dersleri/rdbms-nedir.html*

Veritabanı normalleştirmesi, bir veritabanında verileri verimli bir şekilde düzenleme işlemidir. Bu normalleşme sürecinin iki nedeni var

- Yedekli verileri ortadan kaldırmak, örneğin aynı verileri birden fazla tabloda saklamak.
- Veri bağımlılıklarının sağlanması mantıklı.

Her iki sebep de, bir veri tabanının tükettiği alanı azalttığı ve verilerin mantıksal olarak depolanmasını sağladığı için değerli hedeflerdir.

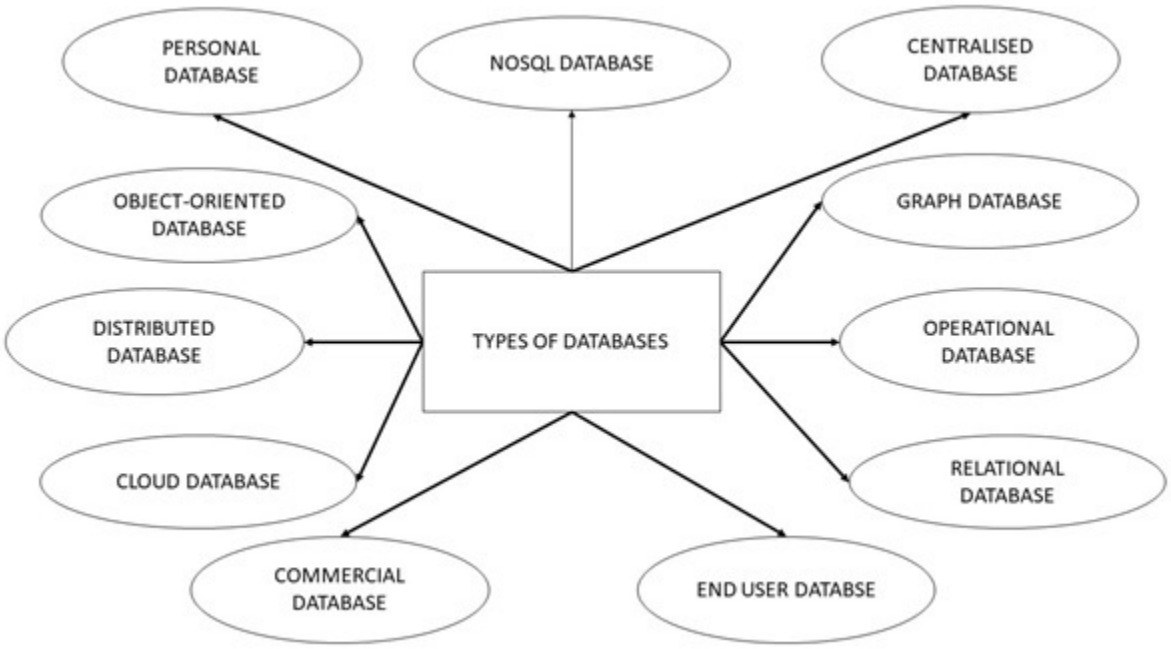
> 💡 **Normalizasyon,** iyi bir veritabanı yapısı oluşturmanızda size yardımcı olan bir dizi kılavuzdan oluşur.

Normalizasyon kuralları normal formlara ayrılır; Bir formu, bir veritabanı yapısının biçimi veya biçimi olarak düşünün. Normal formların amacı, veritabanı yapısını organize etmektir, böylece ilk normal formun kurallarına, sonra ikinci normal forma ve son olarak üçüncü normal forma uygundur.

Daha fazla almak ve dördüncü normal forma, beşinci normal forma ve benzerlerine gitmek sizin seçiminizdir, ama genel olarak, üçüncü normal form fazlasıyla yeterlidir.

- İlk Normal Form (1NF)
- İkinci Normal Form (2NF)
- Üçüncü Normal Form (3NF)

## Database Türleri



# Database Caching

***Kaynak:***
*https://www.beyaz.net/tr/guvenlik/makaleler/onbellege_alma_caching_nedir.html*

Önbellek, geçici bir veri alt kümesini depolayan yüksek hızlı veri depolama katmanıdır. Önbelleğe alma, daha önce alınan veya hesaplanan verinin verimli bir şekilde yeniden kullanılmasını sağlar. Önbellekleme yöntemi ile ilgili verilerin sonraki süreçte talep edildiğinde, verilere birincil depolama konumundan erişildiği için daha yüksek bir performans elde edilir.

Bir önbellekteki veriler genellikle RAM gibi donanımlarlarda saklanır ve veriye erişmek için bir yazılım üzerinden bağlantı kurulması gerekebilir. Önbelleğe alma işleminin amacı altta bulunan yavaş depolama katmanına erişme gereksinimini mimumuma indirerek veri erişim performansını arttırmaktır.



**Önbelleğe Alma Sisteminin Faydaları:** Uygulama performansı arttırılır.Veritabanı maliyeti düşürülür.Arka uçtaki yük azaltılır.Tahmin edilebilir performans sağlanır.Veritabanı bağlantı noktaları ortadan kaldırılır.Okuma verimini arttırır.

### Veritabanı (Database) Önbelleğe Alma

Web uygulamasında kullanılan veritabanının hız ve verimlilik performansı, web uygulamasının performansı için büyük bir etkendir. Veritabanının önbelleğe alınması, uygulama performansını etkileyen arka uç veritabanlarından veri alışveriş sonucu doğacak gecikmelerin azaltılmasını sağlar.