Ağ (Network) Temelleri

TCP/IP nedir? Özellikleri nelerdir?

TCP/IP (İletim Kontrol Protokolü/İnternet Protokolü), internetin ve çoğu ağın temel iletişim protokolleri kümesidir. Bilgisayarların ve diğer cihazların bir ağ üzerinden birbirleriyle nasıl veri alışverişi yapacağını tanımlayan bir dizi kuraldır.

TCP/IP'nin Temel Özellikleri:

- **Katmanlı Mimari:** TCP/IP, her biri belirli bir işlevi yerine getiren farklı katmanlardan oluşan bir model kullanır. Bu katmanlar genellikle şu şekilde sıralanır:
    - **Uygulama Katmanı (Application Layer):** Kullanıcıların doğrudan etkileşimde bulunduğu protokolleri içerir (örneğin, HTTP web tarama için, FTP dosya transferi için, SMTP e-posta için).
    - **Taşıma Katmanı (Transport Layer):** Verinin güvenilir ve sıralı bir şekilde iletilmesini sağlar. En yaygın protokoller TCP (güvenilir, bağlantı odaklı) ve UDP (hızlı, bağlantısız) protokolleridir.
    - **İnternet Katmanı (Internet Layer):** Veri paketlerinin (datagramların) ağlar arasında yönlendirilmesinden sorumludur. IP (İnternet Protokolü) bu katmanın temel protokolüdür ve adresleme ile yönlendirme işlevlerini yerine getirir.
    - **Ağ Arayüzü Katmanı (Network Interface Layer) / Bağlantı Katmanı (Link Layer):** Verilerin fiziksel ağ ortamı üzerinden (örneğin Ethernet, Wi-Fi) nasıl iletileceğini tanımlar.
- **Paket Anahtarlama (Packet Switching):** Veriler, gönderilmeden önce paket adı verilen daha küçük birimlere bölünür. Her paket, hedefine bağımsız olarak yönlendirilebilir ve varış noktasında yeniden birleştirilir. Bu, ağ kaynaklarının daha verimli kullanılmasını sağlar.
- **Adresleme (Addressing):** Ağdaki her cihaza benzersiz bir IP adresi atanır. Bu adresler, veri paketlerinin doğru hedefe yönlendirilmesini sağlar.
- **Yönlendirme (Routing):** İnternet katmanı, veri paketlerinin kaynak cihazdan hedef cihaza ulaşması için en uygun yolu belirler. Bu işlem yönlendiriciler (router) tarafından gerçekleştirilir.
- **Güvenilirlik (TCP ile):** TCP protokolü, veri iletiminde kayıp veya bozuk paketlerin yeniden gönderilmesini sağlayarak güvenilir bir iletişim sunar. Ayrıca verilerin doğru sırada teslim edilmesini garanti eder.

- **Bağlantısız İletişim (UDP ile):** UDP protokolü, hızın güvenilirlikten daha önemli olduğu durumlar için (örneğin, canlı yayın, online oyunlar) bağlantısız ve daha az ek yüke sahip bir iletişim sunar.
- **Standartlaştırma ve Açıklık:** TCP/IP protokolleri açık standartlardır, yani herhangi bir üretici tarafından uygulanabilirler. Bu, farklı üreticilerin cihazlarının ve yazılımlarının birbiriyle uyumlu çalışmasını sağlar.
- **Esneklik ve Ölçeklenebilirlik:** TCP/IP, çok küçük yerel ağlardan devasa küresel internete kadar her türlü ağ boyutuna uyum sağlayabilir.
- **Platform Bağımsızlığı:** TCP/IP, farklı işletim sistemleri ve donanım platformlarında çalışabilir.

Özetle TCP/IP, günümüzdeki internetin ve birçok özel ağın çalışmasını sağlayan, verilerin nasıl paketleneceğini, adresleneceğini, iletileceğini, yönlendirileceğini ve alınacağını tanımlayan temel kurallar bütünüdür.

OSI Katmanları nelerdir? Her katmanın görevi nedir?

OSI (Open Systems Interconnection - Açık Sistemler Bağlantısı) modeli, bir ağdaki iki bilgisayar arasında iletişimin nasıl gerçekleşeceğini tanımlayan kavramsal bir çerçevedir. ISO (International Organization for Standardization - Uluslararası Standardizasyon Örgütü) tarafından geliştirilmiştir. OSI modeli, karmaşık ağ iletişim sürecini daha küçük ve yönetilebilir yedi katmana ayırır. Her katman belirli bir görevi yerine getirir ve bir üst katmana hizmet sunarken bir alt katmandan hizmet alır.

İşte OSI katmanları ve her birinin temel görevleri:

1. **Fiziksel Katman (Physical Layer - Katman 1):**
    - **Görevi:** Verinin bitler halinde fiziksel bir ortam (kablolar, fiber optik, radyo dalgaları vb.) üzerinden nasıl iletileceğini tanımlar. Elektriksel ve mekanik arayüzleri, voltaj seviyelerini, veri iletim hızlarını, kablo tiplerini ve konektörleri belirler.
    - **Örnekler:** Ethernet kabloları (Cat5, Cat6), fiber optik kablolar, Wi-Fi sinyalleri, Bluetooth, hub'lar, tekrarlayıcılar (repeater).
2. **Veri Bağlantı Katmanı (Data Link Layer - Katman 2):**
    - **Görevi:** Fiziksel katman üzerinden hatasız veri transferini sağlamakla sorumludur. Veriyi çerçevelere (frame) böler, fiziksel adreslemeyi (MAC adresleri) yönetir, erişim kontrolünü (hangi cihazın ne zaman veri göndereceğini) sağlar ve hata tespiti ve bazen düzeltmesi yapar.
    - **İki Alt Katmanı Vardır:**

        - **LLC (Logical Link Control - Mantıksal Bağlantı Kontrolü):** Ağ katmanı protokollerini tanımlar ve çerçeve senkronizasyonunu, akış kontrolünü ve hata kontrolünü sağlar.
        - **MAC (Media Access Control - Ortam Erişim Kontrolü):** Paylaşılan bir iletim ortamına erişimi yönetir ve fiziksel adreslemeyi (MAC adresi) gerçekleştirir.
    - **Örnekler:** Ethernet, Wi-Fi (802.11), PPP (Point-to-Point Protocol), switch'ler, köprüler (bridge).
3. **Ağ Katmanı (Network Layer - Katman 3):**
    - **Görevi:** Farklı ağlar arasında veri paketlerinin (packet) yönlendirilmesinden (routing) sorumludur. Mantıksal adreslemeyi (IP adresleri gibi) yönetir, en iyi yol seçimini yapar ve paketleri hedefe ulaştırır. Ağlar arası bağlantıyı kurar, sürdürür ve sonlandırır.
    - **Örnekler:** IP (Internet Protocol), ICMP (Internet Control Message Protocol), IGMP (Internet Group Management Protocol), yönlendiriciler (router).
4. **Taşıma Katmanı (Transport Layer - Katman 4):**
    - **Görevi:** Uçtan uca (end-to-end) güvenilir veya güvenilirsiz veri iletimini sağlar. Veriyi segmentlere (segment) veya datagramlara böler. Akış kontrolü (flow control), hata kontrolü (error control) ve tıkanıklık kontrolü (congestion control) gibi mekanizmalar sunar. Bağlantı odaklı (TCP) veya bağlantısız (UDP) hizmetler sunabilir. Port numaralarını kullanarak farklı uygulamalar arasındaki iletişimi ayırt eder.
    - **Örnekler:** TCP (Transmission Control Protocol), UDP (User Datagram Protocol).
5. **Oturum Katmanı (Session Layer - Katman 5):**
    - **Görevi:** Uygulamalar arasındaki iletişim oturumlarını kurar, yönetir ve sonlandırır. Diyalog kontrolü (kimin ne zaman konuşacağını belirleme), senkronizasyon (uzun veri transferlerinde kesinti olursa kalınan yerden devam etmeyi sağlayan kontrol noktaları ekleme) ve oturum yönetimi (oturumun açılması, sürdürülmesi, kesilmesi ve sonlandırılması) gibi işlevleri yerine getirir.
    - **Örnekler:** NetBIOS, RPC (Remote Procedure Call), PPTP (Point-to-Point Tunneling Protocol).
6. **Sunum Katmanı (Presentation Layer - Katman 6):**

- **Görevi:** Verinin uygulamalar tarafından anlaşılabilir bir formatta sunulmasını sağlar. Veri formatlama, şifreleme/şifre çözme (encryption/decryption) ve sıkıştırma/açma (compression/decompression) gibi işlemleri gerçekleştirir. Uygulama katmanına gelen verinin sözdizimini (syntax) düzenler.
- **Örnekler:** ASCII, EBCDIC gibi karakter kodlamaları; JPEG, GIF, TIFF gibi resim formatları; SSL/TLS (şifreleme için).

7. **Uygulama Katmanı (Application Layer - Katman 7):**
    - **Görevi:** Kullanıcının doğrudan etkileşimde bulunduğu veya ağ servislerine erişim sağlayan uygulamalara arayüz sunar. Ağ kaynaklarına erişimi, dosya transferini, e-posta gönderimini, web'de gezinmeyi ve diğer ağ tabanlı hizmetleri destekleyen protokolleri içerir.
    - **Örnekler:** HTTP (Hypertext Transfer Protocol), HTTPS (HTTP Secure), FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), DNS (Domain Name System), Telnet.

OSI modeli, ağ sorunlarını giderme, yeni protokoller tasarlama ve farklı ağ teknolojilerinin birbiriyle nasıl etkileşime girdiğini anlama konusunda önemli bir referans noktasıdır. TCP/IP modeli daha yaygın olarak kullanılsa da, OSI modeli ağ iletişimi kavramlarını anlamak için hala değerli bir araçtır.

IP Adresi nedir? Türleri nelerdir?

Bir IP adresi (İnternet Protokolü Adresi), bir bilgisayar ağına bağlı cihazları (bilgisayarlar, sunucular, akıllı telefonlar, yönlendiriciler vb.) benzersiz bir şekilde tanımlayan sayısal bir etikettir. Tıpkı bir evin posta adresi gibi, IP adresi de verilerin ağ üzerinde doğru hedefe yönlendirilmesini sağlar.

IP Adresinin Temel Görevleri:

- **Kimlik Belirleme:** Ağa bağlı her cihaz için benzersiz bir tanımlayıcı görevi görür.
- **Adresleme:** Veri paketlerinin kaynak ve hedef cihazlar arasında doğru bir şekilde iletilmesini sağlar.

IP Adresi Türleri:

IP adresleri çeşitli kriterlere göre sınıflandırılabilir:

1. Kullanılan Protokol Sürümüne Göre:

- **IPv4 (Internet Protocol version 4):**

    - En yaygın kullanılan IP adresi türüdür.
    - 32 bitlik bir adres alanına sahiptir (örneğin, 192.168.1.1).
    - Bu, yaklaşık 4.3 milyar (232) benzersiz adresin oluşturulabileceği anlamına gelir.
    - Adresler genellikle nokta ile ayrılmış dört ondalık sayı (her biri 0-255 arasında) şeklinde gösterilir.
    - Artan internet kullanımıyla birlikte IPv4 adresleri tükenme noktasına gelmiştir.
- **IPv6 (Internet Protocol version 6):**
    - IPv4 adreslerinin tükenmesi sorununu çözmek ve internetin gelecekteki büyümesini desteklemek için geliştirilmiştir.
    - 128 bitlik bir adres alanına sahiptir (örneğin, 2001:0db8:85a3:0000:0000:8a2e:0370:7334).
    - Bu, çok daha büyük bir adres havuzu (yaklaşık 3.4×1038 veya 2128 benzersiz adres) sunar.
    - Adresler, iki nokta üst üste ile ayrılmış sekiz grup onaltılık (hexadecimal) sayı ile gösterilir. Sıfırların olduğu bölümler kısaltılabilir.
    - Gelişmiş güvenlik (IPsec entegrasyonu), otomatik yapılandırma ve daha verimli yönlendirme gibi özellikler sunar.

2. Erişilebilirlik ve Kapsama Alanına Göre:

- **Genel (Public) IP Adresi:**
    - İnternet üzerinde benzersizdir ve internet servis sağlayıcınız (İSS) tarafından atanır.
    - İnternete doğrudan bağlı olan cihazların (örneğin, ev veya iş yeri yönlendiricinizin dış arayüzü) dünya çapında erişilebilir olmasını sağlar.
    - Web sitelerine, e-posta sunucularına ve diğer internet servislerine erişmek için kullanılır.
- **Özel (Private) IP Adresi:**
    - Yerel bir ağ (LAN) içinde kullanılmak üzere ayrılmış adreslerdir (örneğin, evinizdeki veya ofisinizdeki cihazlar).

- Bu adresler internet üzerinde doğrudan yönlendirilemez ve benzersiz olmak zorunda değildir (farklı özel ağlarda aynı özel IP adresleri kullanılabilir).
- NAT (Network Address Translation - Ağ Adresi Çevirisi) adı verilen bir teknoloji, özel IP adreslerine sahip cihazların tek bir genel IP adresi üzerinden internete erişmesine olanak tanır.
- Yaygın özel IP adresi aralıkları şunlardır:
  - 10.0.0.0 ile 10.255.255.255
  - 172.16.0.0 ile 172.31.255.255
  - 192.168.0.0 ile 192.168.255.255

3. Atanma Yöntemine Göre:

- **Dinamik IP Adresi:**
  - Cihaz ağa her bağlandığında (veya belirli aralıklarla) DHCP (Dynamic Host Configuration Protocol - Dinamik Ana Bilgisayar Yapılandırma Protokolü) sunucusu tarafından otomatik olarak atanır.
  - Ev kullanıcıları ve çoğu işletme için yaygın bir yöntemdir çünkü yönetimi kolaydır ve mevcut adreslerin verimli kullanılmasını sağlar.
  - Atanan IP adresi zamanla değişebilir.
- **Statik IP Adresi:**
  - Bir cihaza kalıcı olarak atanan ve değişmeyen bir IP adresidir.
  - Genellikle sunucular (web sunucuları, e-posta sunucuları gibi), VPN erişimi veya belirli ağ cihazları için gereklidir çünkü bu cihazlara her zaman aynı adresten erişilmesi gerekir.
  - Genellikle İSS'ler tarafından ek bir ücret karşılığında sağlanır veya ağ yöneticisi tarafından manuel olarak yapılandırılır.

4. Kullanım Amacına Göre (Özel Durumlar):

- **Paylaşımlı (Shared) IP Adresi:** Birden fazla web sitesinin veya servisin aynı genel IP adresini paylaştığı bir durumdur. Genellikle web hosting hizmetlerinde maliyeti düşürmek için kullanılır.
- **Atanmış (Dedicated) IP Adresi:** Tek bir web sitesi veya servis için özel olarak ayrılmış genel bir IP adresidir. Genellikle SSL sertifikaları, belirli güvenlik gereksinimleri veya e-posta gönderim itibarını yönetmek için tercih edilir.

Özetle, IP adresleri ağ iletişiminin temel taşlarından biridir ve farklı türleri, ağların ve cihazların çeşitli ihtiyaçlarını karşılamak üzere tasarlanmıştır.

Router (Yönlendirici) nedir? Ne işe yarar?

Bir **Router (Yönlendirici)**, bilgisayar ağları arasında veri paketlerini ileten bir ağ donanım cihazıdır. Temel işlevi, gelen veri paketlerini inceleyerek bu paketlerin hedeflerine ulaşması için en uygun yolu belirlemek ve paketleri o yola yönlendirmektir.

Router'ların Temel İşlevleri ve Ne İşe Yaradıkları:

1. **Ağları Birbirine Bağlamak:**
    - Router'lar, farklı ağları (örneğin, evinizdeki yerel ağınızı (LAN) internete (WAN - Geniş Alan Ağı) veya bir şirketin farklı departman ağlarını birbirine) bağlar.
    - Bu sayede farklı ağlardaki cihazlar birbirleriyle iletişim kurabilir.
2. **Yönlendirme (Routing):**
    - Bu, router'ın en temel ve en önemli görevidir.
    - Gelen veri paketlerinin başlık bilgisindeki hedef IP adresini okur.
    - Kendi yönlendirme tablosuna (routing table) bakarak, bu hedef IP adresine ulaşmak için en iyi yolu (bir sonraki atlama noktasını veya çıkış arayüzünü) belirler.
    - Paketi belirlenen yola doğru iletir. Yönlendirme tabloları statik olarak (manuel) veya dinamik yönlendirme protokolleri (RIP, OSPF, BGP gibi) aracılığıyla otomatik olarak güncellenebilir.
3. **Veri Paketlerini Filtreleme ve İletme:**
    - Router'lar, belirli kriterlere göre (kaynak/hedef IP adresi, port numarası vb.) veri paketlerini filtreleyebilir. Bu, güvenlik duvarı (firewall) işlevselliğinin bir parçası olabilir.
    - Yalnızca hedefine uygun paketlerin iletilmesini sağlar, gereksiz ağ trafiğini engeller.
4. **Ağ Adresi Çevirisi (NAT - Network Address Translation):**
    - Ev ve küçük ofis router'larında çok yaygın bir özelliktir.
    - Yerel ağdaki birden fazla cihazın (özel IP adresleri kullanan) tek bir genel IP adresi üzerinden internete çıkmasını sağlar. Bu, IPv4 adreslerinin daha

verimli kullanılmasına yardımcı olur ve yerel ağdaki cihazlara ek bir güvenlik katmanı sunar.

5. **DHCP Sunucusu (Dynamic Host Configuration Protocol):**
    - Birçok ev ve küçük ofis router'ı, yerel ağdaki cihazlara otomatik olarak IP adresi, alt ağ maskesi, varsayılan ağ geçidi ve DNS sunucu bilgileri atayan bir DHCP sunucusu görevi görür. Bu, ağ yapılandırmasını basitleştirir.
6. **Güvenlik Duvarı (Firewall):**
    - Çoğu modern router, temel güvenlik duvarı özelliklerine sahiptir.
    - Yetkisiz erişim girişimlerini ve zararlı trafiği engelleyerek yerel ağı dış tehditlerden korumaya yardımcı olur.
7. **Kablosuz Erişim Noktası (Wireless Access Point - WAP):**
    - Günümüzde birçok router, özellikle ev ve küçük ofis kullanımı için tasarlananlar, Wi-Fi özelliği sunarak kablosuz cihazların (dizüstü bilgisayarlar, akıllı telefonlar, tabletler) ağa bağlanmasını sağlar.
8. **QoS (Quality of Service) Yönetimi:**
    - Bazı gelişmiş router'lar, ağ trafiğini önceliklendirme yeteneği sunar. Örneğin, video konferans veya online oyun gibi gecikmeye duyarlı uygulamaların trafiğine, dosya indirme gibi daha az hassas trafik türlerine göre öncelik verebilir.

Özetle Router:

- Farklı ağlar arasında bir köprü görevi görür.
- Veri paketlerinin doğru hedefe en verimli şekilde ulaşmasını sağlar.
- Yerel ağdaki cihazların internete erişimini yönetir ve kolaylaştırır.
- Ağ güvenliğine katkıda bulunur.

Router'lar, internetin ve modern bilgisayar ağlarının temel yapı taşlarından biridir ve verinin küresel ölçekte akışını mümkün kılar.

- Switch nedir? Router ile farkı nedir?

Switch (Ağ Anahtarı) Nedir?

Switch (Ağ Anahtarı), bir yerel alan ağı (LAN - Local Area Network) içerisinde birden fazla bilgisayarı, yazıcıyı, sunucuyu veya diğer ağ cihazlarını birbirine bağlayan bir ağ donanım cihazıdır. Temel amacı, veri paketlerini (Ethernet çerçeveleri olarak da bilinir) alıp yalnızca hedeflenen cihaza ileterek ağ verimliliğini artırmaktır.

Switch'in Temel İşlevleri:

1. **Cihazları Birbirine Bağlamak:** Bir LAN üzerindeki çeşitli cihazların birbirleriyle iletişim kurabilmesi için fiziksel bir bağlantı noktası sağlar. Genellikle üzerinde çok sayıda Ethernet portu bulunur.
2. **MAC Adreslerini Öğrenme ve Kullanma:** Switch'ler, kendilerine bağlı cihazların MAC (Media Access Control) adreslerini öğrenir ve bu bilgiyi bir MAC adres tablosunda saklar. MAC adresi, her ağ arayüz kartının (NIC) donanımsal olarak benzersiz bir tanımlayıcısıdır.
3. **Hedefe Yönelik Veri İletimi (Frame Switching):** Bir cihazdan veri paketi aldığında, paketin başlığındaki hedef MAC adresini okur. MAC adres tablosunu kullanarak bu MAC adresinin hangi porta bağlı olduğunu belirler ve paketi yalnızca o belirli porta yönlendirir. Bu, eski teknoloji olan hub'ların aksine, gereksiz ağ trafiğini önler ve performansı artırır çünkü veri sadece ilgili cihaza gönderilir, tüm ağa yayılmaz.
4. **Çarpışma Alanlarını Azaltma (Collision Domain Isolation):** Her bir switch portu ayrı bir çarpışma alanıdır. Bu, aynı anda birden fazla cihazın veri göndermeye çalışması durumunda oluşabilecek veri çarpışmalarını önemli ölçüde azaltır veya ortadan kaldırır (full-duplex modunda). Hub'larda ise tüm cihazlar aynı çarpışma alanını paylaşırdı.
5. **Yayın Alanları Oluşturma (Broadcast Domain):** Genellikle bir switch, tek bir yayın alanı (broadcast domain) oluşturur. Yani, bir cihazdan gönderilen bir yayın (broadcast) paketi, switch'e bağlı tüm diğer cihazlara iletilir. VLAN (Virtual LAN) özelliği olan yönetilebilir switch'ler ile birden fazla yayın alanı oluşturulabilir.

Router ile Switch Arasındaki Farklar Nedir?

Router ve switch, ağ oluşturmada temel bileşenler olsalar da farklı işlevlere sahiptirler ve OSI modelinin farklı katmanlarında çalışırlar:

Özellik	Switch (Ağ Anahtarı)	Router (Yönlendirici)
OSI Katmanı	Genellikle Katman 2 (Veri Bağlantı Katmanı). (Bazı gelişmiş Layer 3 switch'ler de bulunur.)	Katman 3 (Ağ Katmanı).
Temel İşlevi	Bir yerel alan ağı (LAN) içindeki cihazları birbirine bağlamak ve veri trafiğini yönetmek.	Farklı ağları (örneğin, LAN'ı WAN'a veya farklı LAN'ları birbirine) bağlamak ve ağlar arasında veri paketlerini yönlendirmek.
Adresleme Türü	MAC (Media Access Control) adreslerini kullanır.	IP (Internet Protocol) adreslerini kullanır.
Çalışma Alanı	Genellikle tek bir ağ içinde (LAN).	Birden fazla ağ arasında (LAN'lar arası, LAN-WAN).
Yönlendirme	Genellikle yönlendirme yapmaz (Layer 2 switch'ler). Layer 3 switch'ler sınırlı yönlendirme yapabilir.	Temel görevi yönlendirmedir; en iyi yolu bulur.
Veri Birimi	Çerçeveler (Frames).	Paketler (Packets).
Bağlantı Noktası	Genellikle çok sayıda Ethernet portuna sahiptir.	Genellikle daha az sayıda LAN portuna ve en az bir WAN portuna sahiptir.
NAT & DHCP	Genellikle bu özellikleri sunmaz.	Ev/küçük ofis router'ları genellikle NAT ve DHCP sunucusu özelliklerine sahiptir.
Çarpışma Alanı	Her port ayrı bir çarpışma alanıdır.	Her portu ayrı bir çarpışma alanına ve ayrı bir yayın alanına (broadcast domain) bağlar.
Yayın Alanı	Varsayılan olarak tek bir yayın alanı oluşturur (VLAN yoksa).	Her arayüzü (portu) farklı bir yayın alanına aittir. Yayınları ağlar arasında iletmez (varsayılan olarak).

Kullanım Amacı	Bir evdeki veya ofisteki bilgisayarları, yazıcıları vb. birbirine bağlamak.	Ev ağını internete bağlamak, farklı ofis ağlarını birbirine bağlamak.
Karar Verme	"Bu MAC adresi hangi porta bağlı?"	"Bu IP adresi hangi ağda ve oraya en iyi nasıl ulaşılır?"

E-Tablolar'a aktar

Özetle:

- **Switch**, aynı yerel ağdaki cihazları birbirine bağlayarak aralarında verimli bir iletişim sağlar (cihazdan cihaza).
- **Router**, farklı ağları birbirine bağlar ve veri paketlerinin bu ağlar arasında doğru hedefe yönlendirilmesini sağlar (ağdan ağa).

Bir ev ağında tipik bir senaryo şöyledir: Modem internet bağlantısını alır, bu bağlantı bir router'a gider. Router, evdeki cihazlar için bir yerel ağ oluşturur (genellikle Wi-Fi ve kablolu portlar aracılığıyla) ve bu cihazların internete erişmesini sağlar. Eğer daha fazla kablolu bağlantı noktasına ihtiyaç duyulursa, router'ın LAN portlarından birine bir switch bağlanarak port sayısı artırılabilir. Bu durumda switch, router'ın oluşturduğu yerel ağ içindeki cihazların birbirleriyle iletişimini yönetir.

Server (Sunucu) nedir?

Sunucu (Server), bilgisayar ağlarında diğer bilgisayarlara, cihazlara veya kullanıcılara (istemcilere) çeşitli hizmetler sunan özel bir bilgisayar veya yazılımdır. Temel olarak, istemcilerden gelen istekleri karşılamak ve onlara bilgi, veri veya kaynak sağlamakla görevlidir.

Normal bir kişisel bilgisayardan farklı olarak sunucular genellikle çok daha güçlü donanıma (işlemci, RAM, depolama) sahiptir ve kesintisiz (7/24) çalışacak şekilde tasarlanmıştır. Yüksek performans, güvenilirlik ve sürekli erişilebilirlik sunucuların temel özelliklerindendir.

Sunucular farklı amaçlara hizmet edebilir ve bu amaçlarına göre çeşitlere ayrılırlar:

- **Web Sunucuları:** Web sitelerinin dosyalarını depolar ve kullanıcıların web tarayıcılarından gelen istekler üzerine bu sayfaları sunar. İnternet sitelerinin görüntülenmesini sağlarlar.
- **Dosya Sunucuları:** Kullanıcıların ağ üzerinden dosya depolamasına, paylaşmasına ve erişmesine olanak tanır. Merkezi bir dosya deposu görevi görürler.

- **Veritabanı Sunucuları:** Veritabanlarını barındırır ve yönetir. İstemcilerden gelen veri sorgularını işler ve sonuçları döndürür.
- **E-posta Sunucuları:** E-posta iletişimini yönetir. Gelen ve giden e-postaları alır, depolar ve ilgili alıcılara iletir.
- **Uygulama Sunucuları:** Belirli uygulamaları çalıştırır ve bu uygulamalara ağ üzerinden istemcilerin erişmesini sağlar.
- **Oyun Sunucuları:** Çok oyunculu çevrimiçi oyunların çalışması için gerekli ortamı sağlar ve oyuncular arasındaki etkileşimi yönetir.
- **Yazdırma Sunucuları:** Ağ üzerindeki yazıcıları yönetir ve istemcilerin yazdırma isteklerini sıraya koyarak işlemesini sağlar.

Sunucular fiziksel makineler olabileceği gibi, sanallaştırma teknolojisi ile tek bir fiziksel sunucu üzerinde çalışan birden fazla sanal sunucu şeklinde de olabilirler. Ayrıca günümüzde bulut bilişim ile birlikte internet üzerinden erişilebilen ve yönetilen bulut sunucuları da yaygın olarak kullanılmaktadır.

Özetle sunucular, modern bilgi teknolojileri altyapısının temel taşlarıdır ve internetin, ağ iletişiminin ve birçok dijital hizmetin çalışması için kritik öneme sahiptirler. İstemci-sunucu modeli, bilgisayar ağlarındaki iletişimin ve kaynak paylaşımının temelini oluşturur.

Client (İstemci) nedir?

İstemci (Client), bilgisayar ağlarında sunucu adı verilen diğer bilgisayarlardan veya sistemlerden hizmet talep eden ve bu hizmetleri kullanan bilgisayar, cihaz veya yazılımdır. İstemciler genellikle kullanıcıların doğrudan etkileşimde bulunduğu arayüzü sağlarlar.

İstemci-sunucu modelinde, istemci hizmeti talep eden taraftır, sunucu ise bu talebi karşılayarak hizmeti sunan taraftır. Örneğin, internette bir web sitesini ziyaret etmek istediğinizde, web tarayıcınız (istemci) ilgili web sitesinin barındığı sunucuya bir istek gönderir ve sunucu da web sayfasının içeriğini (HTML, resimler vb.) istemciye gönderir. İstemci olan web tarayıcınız bu içeriği işleyerek size web sayfasını görüntüler.

İstemciler farklı şekillerde karşımıza çıkabilir:

- **Cihazlar:** Kişisel bilgisayarlar, akıllı telefonlar, tabletler gibi internete veya bir ağa bağlı olan cihazlar birer istemci olabilir.
- **Yazılımlar:** Web tarayıcıları (Chrome, Firefox, Safari vb.), e-posta programları (Outlook, Thunderbird vb.), dosya transfer programları (FTP istemcileri), anlık

mesajlaşma uygulamaları, oyunlar ve diğer birçok uygulama birer istemci yazılımıdır.

İstemcilerin temel görevi, kullanıcının ihtiyaçları doğrultusunda sunucularla iletişim kurarak bilgi almak, veri göndermek veya belirli işlemleri gerçekleştirmek için isteklerde bulunmaktır. Sunucudan gelen yanıtları alırlar, işlerler ve genellikle kullanıcıya anlamlı bir formatta sunarlar.

"İstemci" terimi aynı zamanda, iş yükünün ne kadarının istemci tarafında işlendiğine bağlı olarak "ince istemci" (thin client) ve "kalın istemci" (fat client) gibi farklı türdeki istemcileri tanımlamak için de kullanılabilir. İnce istemciler genellikle sunucuya daha bağımlıdır ve iş yükünün büyük kısmını sunucuya bırakırken, kalın istemciler kendi işlem güçlerini daha fazla kullanır.

Kısacası istemci, ağ üzerindeki hizmet sağlayıcı olan sunucularla etkileşim kurarak kullanıcıların dijital kaynaklara ve servislere erişmesini sağlayan uç noktadır.

Port nedir? Hangi portlar ne için kullanılır?

Bilgisayar ağlarında **port (bağlantı noktası)**, bir ağdaki belirli bir bilgisayar veya cihaz üzerinde çalışan belirli bir uygulama veya hizmeti tanımlamak için kullanılan yazılımsal bir numaradır. Fiziksel bir bağlantı noktası (USB portu, Ethernet portu gibi) değildir; ağ iletişimi sırasında verinin hangi uygulamaya gideceğini belirlemeye yarayan mantıksal bir kapıdır.

İnternet üzerinden veya yerel bir ağda iletişim kurulurken, veriler IP adresleri aracılığıyla doğru cihaza ulaşır. Ancak o cihaz üzerinde çalışan birden fazla uygulama veya hizmet olabilir (aynı anda hem web sayfası görüntülüyor, hem e-posta alıyor, hem de dosya indiriyor olabilirsiniz). Portlar, gelen verinin hangi uygulamaya iletilmesi gerektiğini belirleyerek bu trafiğin düzenlenmesini sağlar. Her uygulamanın veya hizmetin genellikle belirli bir port numarası atanmıştır.

Port numaraları 0 ile 65535 arasında değişir ve genellikle üç kategoriye ayrılır:

- **İyi Bilinen Portlar (Well-Known Ports):** 0 ile 1023 arasındaki portlardır. Belirli ve yaygın olarak kullanılan hizmetler için ayrılmıştır ve IANA (Internet Assigned Numbers Authority) tarafından belirlenir.
- **Kayıtlı Portlar (Registered Ports):** 1024 ile 49151 arasındaki portlardır. Belirli uygulamalar veya hizmetler için tescil edilebilir, ancak İyi Bilinen Portlar kadar evrensel olarak atanmamışlardır.
- **Dinamik/Özel Portlar (Dynamic/Private Ports):** 49152 ile 65535 arasındaki portlardır. Genellikle istemci uygulamalar tarafından geçici iletişimler için dinamik olarak atanır.

Yaygın Olarak Kullanılan Bazı Portlar ve Amaçları:

Aşağıda sıkça karşılaşılan bazı portlar ve genellikle hangi hizmetler için kullanıldıkları listelenmiştir:

- **Port 20 & 21 (TCP): FTP (File Transfer Protocol)** - Dosya transferi için kullanılır. 20 veri transferi, 21 ise kontrol bağlantısı içindir.
- **Port 22 (TCP): SSH (Secure Shell)** - Güvenli uzaktan erişim, komut çalıştırma ve dosya transferi (SFTP, SCP) için kullanılır. Şifreli iletişim sağlar.
- **Port 23 (TCP): Telnet** - Şifresiz metin tabanlı uzaktan erişim için kullanılır. Güvenlik açıkları nedeniyle kullanımı azdır.
- **Port 25 (TCP): SMTP (Simple Mail Transfer Protocol)** - E-posta göndermek için e-posta sunucuları arasında veya e-posta istemcilerinden sunucuya e-posta iletimi için kullanılır.
- **Port 53 (TCP/UDP): DNS (Domain Name System)** - Alan adlarını (örneğin https://www.google.com/search?q=google.com) IP adreslerine çözümlemek için kullanılır.
- **Port 80 (TCP): HTTP (Hypertext Transfer Protocol)** - Şifresiz web sayfalarına erişim için kullanılır. Bir web tarayıcısı adres çubuğuna bir web adresi yazdığınızda genellikle bu port üzerinden iletişim kurulur.
- **Port 110 (TCP): POP3 (Post Office Protocol version 3)** - E-posta istemcilerinin sunucudan e-postaları indirmesi için kullanılır. E-postalar genellikle sunucudan silinir.
- **Port 143 (TCP): IMAP (Internet Message Access Protocol)** - E-posta istemcilerinin e-postalara sunucu üzerinde erişmesi ve yönetmesi için kullanılır. E-postalar sunucuda kalır.
- **Port 443 (TCP): HTTPS (Hypertext Transfer Protocol Secure)** - Güvenli (şifreli) web sayfalarına erişim için kullanılır. SSL/TLS protokolleri ile HTTP iletişimini şifreler. Çevrimiçi alışveriş ve bankacılık gibi güvenliğin önemli olduğu durumlarda kullanılır.
- **Port 3389 (TCP): RDP (Remote Desktop Protocol)** - Windows işletim sistemlerinde bir bilgisayara uzaktan bağlanarak masaüstünü kontrol etmek için kullanılır.

Bu liste yaygın portlardan sadece birkaçıdır. Binlerce farklı port, farklı uygulamalar ve hizmetler tarafından kullanılmaktadır. Güvenlik açısından, gereksiz portların açık bırakılmaması önemlidir, çünkü kötü niyetli kişiler bu açık portları sistemlere sızmak veya saldırı düzenlemek için kullanabilirler.

Socket (Soket) nedir? Hangi amaçla kullanılır?

Ağ iletişiminde **Socket (Soket)**, farklı cihazlardaki veya aynı cihaz üzerindeki farklı süreçler (çalışan programlar) arasında veri alışverişi yapmak için kullanılan bir iletişim bitiş noktasıdır. Bir ağ bağlantısının her iki ucunda bulunan birer yazılımsal yapıdır.

Daha teknik bir ifadeyle soket, bir IP adresi ve bir port numarasının birleşimidir. IP adresi verinin hangi bilgisayara gideceğini belirtirken, port numarası o bilgisayar üzerindeki hangi spesifik uygulamaya veya sürece iletileceğini gösterir. Bu IP adresi ve port numarası ikilisi, ağdaki benzersiz bir iletişim noktasını tanımlar.

Hangi Amaçla Kullanılır?

Soketlerin temel amacı, ağ üzerinden süreçler arası iletişimi (Inter-process Communication - IPC) sağlamaktır. Soket programlama, geliştiricilerin ağ üzerinden veri gönderip alabilen uygulamalar oluşturmasına olanak tanır. Başlıca kullanım alanları şunlardır:

- **İstemci-Sunucu Uygulamaları:** İnternet üzerindeki çoğu uygulamanın temelini oluşturur. Web tarayıcıları (istemci) web sunucularıyla soketler aracılığıyla iletişim kurar. E-posta programları, dosya transferi (FTP) programları ve veritabanı uygulamaları da istemci-sunucu modelinde soketleri kullanır.
- **Gerçek Zamanlı Uygulamalar:** Çevrimiçi oyunlar, anlık mesajlaşma programları ve video konferans uygulamaları gibi gerçek zamanlı veri akışı gerektiren uygulamalar, düşük gecikmeli iletişim için soketleri yoğun şekilde kullanır. Özellikle WebSocket gibi teknolojiler, istemci ve sunucu arasında kalıcı bir bağlantı üzerinden sürekli veri akışını sağlamak için soketleri temel alır.
- **Dağıtık Sistemler:** Birden fazla bilgisayarın birlikte çalışarak tek bir görev üzerinde çalıştığı dağıtık sistemlerde, bileşenler arasındaki iletişimi sağlamak için soketler kullanılır.
- **Özel Ağ Protokolleri:** HTTP, FTP, SMTP gibi standart ağ protokollerinin yanı sıra, ihtiyaca özel iletişim kurallarını belirleyen uygulamalar da kendi veri alışverişleri için soketleri kullanır.

Özetle soketler, ağdaki cihazlar ve uygulamalar arasında kapılar görevi görerek verinin doğru adrese ve doğru uygulamaya ulaşmasını sağlar. Ağ tabanlı uygulamaların geliştirilmesi ve çalışması için temel bir mekanizmadır.

Request (İstek) nedir?

Bilgisayar ağları ve özellikle istemci-sunucu iletişiminde **Request (İstek)**, bir istemcinin (örneğin bir web tarayıcısı veya mobil uygulama) bir sunucuya belirli bir eylemi gerçekleştirmesi veya belirli bir kaynağı (bir web sayfası, bir dosya, veri vb.) sağlaması için gönderdiği bir mesajdır.

İstek, istemci ve sunucu arasındaki iletişim döngüsünün ilk adımıdır. İstemci bir kaynağa ihtiyaç duyduğunda veya sunucuda bir işlem yapmak istediğinde bir istek oluşturur ve bu isteği sunucuya gönderir. Sunucu bu isteği alır, işler ve bir yanıt (Response) ile istemciye geri döner.

Bir HTTP isteği genellikle şu temel bileşenleri içerir:

- **İstek Metodu (Request Method):** İstemcinin sunucudan ne tür bir eylem beklediğini belirtir. En yaygın HTTP metotları şunlardır:
    - GET: Belirtilen kaynaktan veri almak için kullanılır. (Örn: Bir web sayfasını görüntülemek)
    - POST: Sunucuya işlenmek üzere veri göndermek için kullanılır. (Örn: Bir form göndermek, bir kullanıcı kaydı oluşturmak)
    - PUT: Belirtilen kaynağı sunucuya gönderilen veri ile tamamen değiştirmek için kullanılır.
    - DELETE: Belirtilen kaynağı sunucudan silmek için kullanılır.
    - PATCH: Belirtilen kaynak üzerinde kısmi değişiklikler yapmak için kullanılır.
    - HEAD: GET metodu gibidir ancak sunucu sadece yanıt başlıklarını döner, yanıt gövdesini döndürmez. Kaynağın varlığını veya başlık bilgilerini kontrol etmek için kullanılır.
- **URL (Uniform Resource Locator):** İstemcinin erişmek istediği kaynağın veya yapmak istediği işlemin adresini belirtir.
- **Başlıklar (Headers):** İstek hakkında ek bilgiler içerir. İstemci, sunucu, gönderilen verinin tipi, yetkilendirme bilgileri, çerezler gibi çeşitli meta veriler bu kısımda yer alır.
- **Gövde (Body):** Özellikle POST, PUT ve PATCH gibi metotlarda, sunucuya gönderilen asıl veri içeriğini taşır. (Örn: Bir formdaki kullanıcı adı ve şifre bilgileri)

İstekler, web tarayıcılığından mobil uygulamalara, API iletişiminden farklı ağ servislerine kadar birçok alanda kullanılır. İstemcinin sunucu ile etkileşim kurabilmesi ve sunucunun sunduğu hizmetlerden yararlanabilmesi için istek mekanizması temel bir rol oynar.

Response (Yanıt) nedir?

Bilgisayar ağlarında ve özellikle istemci-sunucu modelindeki iletişimde **Response (Yanıt)**, bir sunucunun, kendisine daha önce bir istemci tarafından gönderilmiş olan bir **istek (Request)** mesajına karşılık olarak gönderdiği mesajdır. Yanıt, sunucunun isteği işledikten sonraki sonucunu veya talep edilen kaynağı içerir.

İstemci bir sunucuya (örneğin bir web sunucusuna) bir istek gönderdiğinde, sunucu bu isteği alır, yorumlar ve uygun bir yanıt oluşturur. Bu yanıt daha sonra istemciye geri gönderilir. Bu **istek-yanıt döngüsü** ağ iletişiminin temelini oluşturur.

Bir HTTP yanıtı genellikle şu ana bileşenlerden oluşur:

- **Durum Satırı (Status Line):** Yanıtın ilk satırıdır ve HTTP versiyonunu, bir durum kodunu (Status Code) ve bu durum kodunun kısa bir metinsel açıklamasını içerir.
    - **HTTP Versiyonu:** Kullanılan HTTP protokolünün versiyonunu belirtir (örneğin, HTTP/1.1).
    - **Durum Kodu (Status Code):** Üç basamaklı sayısal bir koddur ve isteğin sonucunu belirtir. Bu kodlar, isteğin başarılı olup olmadığını, bir hata oluşup oluşmadığını veya başka bir eylemin gerekip gerekmediğini bildirir. Yaygın durum kodlarına örnekler:
        - 200 OK: İstek başarılı oldu ve talep edilen kaynak veya işlem sonucu yanıt gövdesinde döndürüldü.
        - 404 Not Found: Talep edilen kaynak sunucuda bulunamadı.
        - 500 Internal Server Error: Sunucuda isteği yerine getirirken beklenmeyen bir hata oluştu.
        - 301 Moved Permanently: Talep edilen kaynak kalıcı olarak başka bir adrese taşındı.
    - **Sebep İfadesi (Reason Phrase):** Durum kodunu açıklayan kısa, insan tarafından okunabilir metindir (örneğin, "OK", "Not Found").
- **Başlıklar (Headers):** Yanıt hakkında ek bilgiler içeren satırlardır. Sunucu, yanıtın içeriği, boyutu, tipi, önbelleğe alma talimatları, sunucu bilgisi gibi çeşitli meta verileri bu kısımda belirtir.
- **Gövde (Body):** Yanıtın ana içeriğidir. Eğer istek bir kaynak (örneğin bir web sayfası veya resim) talep ediyorsa, bu kaynak yanıtın gövdesinde yer alır. Eğer istek bir işlem yapılmasını talep ediyorsa, işlemin sonucu veya ilgili veriler de gövdede bulunabilir. Bazı yanıt türlerinde (örneğin HEAD isteklerine verilen yanıtlarda veya bazı hata yanıtlarında) gövde boş olabilir.

Yanıt, istemcinin gönderdiği isteğe karşılık sunucunun verdiği geri bildirimdir ve istemcinin sonraki adımlarını belirlemesinde kritik rol oynar (örneğin, web sayfasını görüntülemek, hata mesajını göstermek veya başka bir sayfaya yönlenmek gibi).

FTP (File Transfer Protocol) nedir? Ne işe yarar?

FTP (File Transfer Protocol - Dosya Transfer Protokolü), bir bilgisayar ağı üzerindeki istemci ve sunucu arasında dosya transferi yapmak için kullanılan standart bir ağ protokolüdür. İnternetin ilk geliştirilen protokollerinden biridir ve günümüzde hala belirli amaçlarla kullanılmaktadır.

Ne İşe Yarar?

FTP'nin temel amacı, farklı bilgisayarlar arasında kolay ve etkili bir şekilde dosya alışverişini sağlamaktır. Ne işe yaradığı daha detaylı açıklanırsa:

- **Dosya Yükleme (Upload):** Yerel bilgisayarınızdaki dosyaları bir sunucuya (örneğin bir web sitesi barındırma sunucusuna) göndermek için kullanılır. Web sitesi sahipleri genellikle web sitelerinin dosyalarını sunucuya yüklemek için FTP kullanır.
- **Dosya İndirme (Download):** Bir sunucuda bulunan dosyalara erişmek ve bu dosyaları kendi bilgisayarınıza indirmek için kullanılır. Yazılım indirme siteleri veya genel kullanıma açık dosya arşivleri genellikle FTP üzerinden erişilebilir.
- **Dosya Yönetimi:** FTP bağlantısı kurulduktan sonra, sunucu üzerindeki dosya ve dizinler üzerinde temel işlemler (dosya silme, yeniden adlandırma, klasör oluşturma vb.) yapılabilir.

FTP, istemci-sunucu modeli üzerinde çalışır. Bir FTP istemci yazılımı (FileZilla, WinSCP gibi) aracılığıyla bir FTP sunucusuna bağlanılır. FTP'nin dikkat çekici bir özelliği, kontrol bağlantısı ve veri bağlantısı olmak üzere genellikle iki ayrı bağlantı kullanmasıdır. Port 21 genellikle kontrol komutları için kullanılırken, veri transferi için farklı portlar (aktif veya pasif moda göre değişir) kullanılır.

Güvenlik açısından, standart FTP verileri şifrelemeden aktardığı için hassas bilgiler için tek başına kullanılması önerilmez. Bu nedenle, daha güvenli alternatifler olan FTPS (FTP Secure) ve SFTP (SSH File Transfer Protocol) günümüzde daha yaygın olarak tercih edilmektedir. Ancak basit dosya transfer ihtiyaçları veya güvenlik endişesinin düşük olduğu durumlar için FTP hala pratik bir çözüm olabilir.

Alan Adı Sistemleri ve İletişim

DNS (Domain Name System) nedir?

DNS (Domain Name System - Alan Adı Sistemi), internetin temel yapı taşlarından biridir ve internete bağlı bilgisayarların, servislerin veya diğer kaynakların isimlerini (alan adlarını) IP adreslerine çevirmek için kullanılan hiyerarşik ve dağıtılmış bir isimlendirme sistemidir. Basitçe ifade etmek gerekirse, DNS internetin "telefon rehberi" gibidir.

Ne İşe Yarar?

İnsanlar web sitelerine veya diğer internet kaynaklarına genellikle alan adları aracılığıyla erişirler (örneğin, www.google.com, www.youtube.com). Ancak bilgisayarlar ve ağ cihazları birbirleriyle iletişim kurmak için bu isimleri değil, sayısal IP adreslerini kullanırlar (örneğin, 172.217.169.36). IP adreslerini akılda tutmak insanlar için zor ve pratik değildir. İşte tam burada DNS devreye girer.

DNS'in temel işlevi şunlardır:

- **Alan Adı Çözümleme (Name Resolution):** Kullanıcının girdiği alan adını (örneğin www.google.com) karşılık gelen IP adresine çevirir. Bu sayede web tarayıcısı veya başka bir istemci uygulama, doğru sunucuya bağlanmak için gerekli olan IP adresini öğrenir.
- **İnternette Gezinmeyi Kolaylaştırma:** Kullanıcıların karmaşık IP adresleri yerine akılda kalıcı alan adları kullanarak internet kaynaklarına erişmesini sağlar. Bu, internet kullanımını çok daha pratik hale getirir.
- **E-posta Yönlendirme:** E-posta gönderirken, DNS e-posta adresindeki alan adını kullanarak ilgili e-posta sunucusunun IP adresini bulmaya yardımcı olur.
- **Yük Dengeleme ve Yönlendirme:** Büyük web siteleri veya servisler için DNS, trafiği farklı sunuculara yönlendirerek yük dengelemesi yapabilir ve coğrafi konuma göre kullanıcıları kendilerine en yakın sunucuya yönlendirebilir.

DNS sistemi, dünya genelinde dağıtılmış sunuculardan (DNS sunucuları) oluşur. Bir kullanıcı bir alan adına erişmek istediğinde, bilgisayarı bir DNS sorgusu başlatır. Bu sorgu sırasıyla yerel DNS sunucularına, kök sunuculara, TLD (Üst Düzey Alan Adı) sunucularına ve yetkili ad sunucularına iletilerek ilgili alan adının IP adresi bulunur ve kullanıcıya döndürülür. Bu süreç genellikle milisaniyeler içinde gerçekleşir.

Özetle, DNS, internet üzerindeki isim tabanlı adresleme ile sayısal IP adresleme arasında köprü kurarak internetin düzenli ve kullanılabilir olmasını sağlayan hayati bir servistir.

Domain (Alan Adı) nedir?

Domain (Alan Adı), internet üzerindeki bir web sitesini, sunucuyu veya diğer internet kaynaklarını tanımlamak için kullanılan, insanlar tarafından okunabilir ve akılda

tutulabilir adreslerdir. Sayısal IP adreslerinin aksine (örneğin 172.217.169.36), alan adları kelimelerden oluşur (örneğin google.com, wikipedia.org).

Ne İşe Yarar?

Alan adlarının temel işlevi, internet kaynaklarına erişimi kolaylaştırmaktır. İnternetin çalışması için bilgisayarlar IP adreslerini kullanırken, insanların her web sitesinin IP adresini ezberlemesi pratik değildir. Alan adları bu sorunu çözmek için geliştirilmiştir.

Alan adlarının başlıca işlevleri şunlardır:

- **Erişim Kolaylığı:** Kullanıcıların karmaşık IP adresleri yerine hatırlanması kolay isimler kullanarak web sitelerine ve diğer internet servislerine ulaşmasını sağlarlar.
- **Markalaşma ve Kimlik:** Bir web sitesi veya çevrimiçi varlık için benzersiz bir kimlik oluştururlar. Markaların çevrimiçi dünyadaki temsilidirler. İyi seçilmiş bir alan adı, markanın tanınırlığını ve akılda kalıcılığını artırır.
- **DNS ile İlişki:** Alan adları, Alan Adı Sistemi (DNS) ile birlikte çalışır. Kullanıcı bir alan adını yazdığında, DNS bu alan adını ilgili IP adresine çevirir ve böylece kullanıcının doğru sunucuya yönlendirilmesi sağlanır.
- **E-posta Adresleri:** E-posta adreslerinde "@" işaretinden sonra gelen kısım, e-posta sunucusunun bulunduğu alanı belirtir ve bu da bir alan adıdır (örneğin, kullaniciadi@firmaadi.com).

Alan adları hiyerarşik bir yapıya sahiptir. En sağda yer alan kısım **Üst Düzey Alan Adı (TLD - Top-Level Domain)** olarak adlandırılır (örneğin .com, .org, .net, .edu, .gov veya ülkelere özgü .tr, .uk, .de gibi uzantılar). TLD'nin solundaki kısım genellikle **İkinci Düzey Alan Adı (SLD - Second-Level Domain)** olarak adlandırılır ve genellikle markanın veya kuruluşun adını içerir (örneğin, google google.com'da). Daha solda alt alan adları (subdomain) bulunabilir (örneğin, mail.google.com adresindeki mail).

Alan adları tescil kuruluşları aracılığıyla belirli bir süre için kiralanır ve süresi dolduğunda yenilenmesi gerekir, aksi takdirde başkaları tarafından tescil edilebilir. Her alan adı internet üzerinde benzersizdir.

• DHCP (Dynamic Host Configuration Protocol) nedir?

DHCP (Dynamic Host Configuration Protocol - Dinamik Ana Bilgisayar Yapılandırma Protokolü), bilgisayar ağlarında cihazlara (istemcilere) otomatik olarak IP adresleri ve diğer ağ yapılandırma bilgilerini atamak için kullanılan bir ağ yönetim protokolüdür.

Ne İşe Yarar?

DHCP'nin temel amacı, ağ yöneticilerinin iş yükünü azaltmak ve ağdaki cihazların yapılandırılmasını kolaylaştırmaktır. DHCP olmasaydı, ağa bağlanan her cihaza (bilgisayar, akıllı telefon, yazıcı vb.) manuel olarak bir IP adresi, alt ağ maskesi, varsayılan ağ geçidi ve DNS sunucusu gibi bilgilerin girilmesi gerekirdi. Özellikle büyük ağlarda veya sık sık yeni cihazların bağlanıp ayrıldığı durumlarda bu işlem oldukça zaman alıcı ve hataya açık olurdu.

DHCP, bu süreci otomatikleştirerek aşağıdaki faydaları sağlar:

- **Otomatik IP Ataması:** Ağdaki cihazlar ağa bağlandıklarında DHCP sunucusundan otomatik olarak bir IP adresi ve gerekli diğer ağ bilgilerini alırlar.
- **IP Adresi Yönetiminin Kolaylaşması:** Ağ yöneticileri, IP adreslerini merkezi bir havuzdan yönetir ve dağıtır. Bu, IP adresi çakışmalarını önler ve adres yönetimini daha verimli hale getirir.
- **Ağ Yapılandırma Hatalarının Azalması:** Manuel yapılandırma hataları ortadan kalkar.
- **Cihaz Hareketliliği:** Dizüstü bilgisayarlar veya mobil cihazlar gibi farklı ağlara sık sık bağlanan cihazlar, her bağlandıklarında otomatik olarak yeni ağdan IP adresi alabilirler.
- **Zaman ve Maliyet Tasarrufu:** Ağ yöneticilerinin her cihazla tek tek uğraşmasına gerek kalmadığı için zamandan ve işgücünden tasarruf sağlanır.

DHCP, genellikle bir yönlendirici (router), sunucu veya özel bir DHCP sunucu cihazı üzerinde çalışır. Bir cihaz ağa bağlandığında bir DHCP isteği gönderir ve DHCP sunucusu bu isteği alarak havuzundaki uygun bir IP adresini ve diğer yapılandırma bilgilerini cihaza kiralar. Bu kiralama süresi sonunda cihaz genellikle IP adresini yeniler veya yeni bir adres talep eder.

Özetle, DHCP, ağdaki cihazların hızlı, kolay ve hatasız bir şekilde ağa bağlanabilmesi ve iletişim kurabilmesi için gerekli olan ağ yapılandırma bilgilerini otomatik olarak sağlayan kritik bir protokoldür.

WWW (World Wide Web) nedir?

WWW (World Wide Web - Dünya Çapında Ağ), internet üzerinde çalışan ve birbirine bağlı hipermetin (hypertext) belgelerinden oluşan devasa bir bilgi sistemidir. Genellikle internetin kendisiyle karıştırılsa da, WWW internetin üzerinde çalışan bir servistir.

Ne İşe Yarar?

WWW'nin temel amacı, bilgiye kolayca erişimi ve bilginin paylaşımını sağlamaktır. Başlıca işlevleri şunlardır:

- **Bilgiye Erişim:** Kullanıcıların web tarayıcıları aracılığıyla internet üzerindeki web sitelerine ve sayfalara erişmesini sağlar. Bu web sayfaları metin, resim, video, ses ve diğer multimedya içeriklerini barındırabilir.
- **Hipermetin Bağlantıları (Hyperlinks):** Web sayfaları arasındaki bağlantılar (linkler) sayesinde kullanıcılar bir sayfadan başka bir sayfaya kolayca geçiş yapabilirler. Bu bağlantılı yapı, bilgiyi keşfetmeyi ve gezinmeyi mümkün kılar.
- **Web Siteleri:** WWW, web siteleri adı verilen, birbiriyle ilişkili web sayfalarının bir araya geldiği dijital alanları oluşturur. Her web sitesi genellikle benzersiz bir alan adı (domain name) ile tanımlanır.
- **Evrensel Bilgi Paylaşımı:** Dil, coğrafya veya platformdan bağımsız olarak herkesin bilgiye erişmesine ve kendi bilgilerini yayınlamasına olanak tanır.

WWW, 1989 yılında Tim Berners-Lee tarafından CERN'de geliştirilmiştir. HTTP (Hypertext Transfer Protocol), HTML (Hypertext Markup Language) ve URL (Uniform Resource Locator) gibi temel teknolojilere dayanır. Kullanıcılar bir web tarayıcısına bir URL girdiğinde, tarayıcı ilgili web sunucusuna bir HTTP isteği gönderir, sunucu da web sayfasını (HTML ve diğer dosyalar) HTTP yanıtı olarak geri gönderir ve tarayıcı bu sayfayı kullanıcıya görüntüler.

Kısacası WWW, internetin küresel bilgi ağı olarak kullanılmasını sağlayan, web siteleri ve hipermetin bağlantıları aracılığıyla bilgiye erişimi ve gezinmeyi kolaylaştıran bir sistemdir. İnternetin yaygınlaşmasında ve modern bilgi çağının şekillenmesinde çok büyük bir rol oynamıştır.

- A Kaydı (Address Record) DNS'te ne anlama gelir?

DNS (Domain Name System - Alan Adı Sistemi) içerisinde **A Kaydı (Address Record)**, bir alan adını veya alt alan adını doğrudan bir IPv4 adresiyle eşleyen en temel ve en sık kullanılan kayıt türüdür. Buradaki "A" harfi "Address" (Adres) kelimesini temsil eder.

Ne Anlama Gelir ve Ne İşe Yarar?

A kaydının temel anlamı ve işlevi, bir internet adresinin (alan adının) hangi sayısal adrese (IPv4 adresi) karşılık geldiğini DNS sistemine bildirmektir. Kullanıcılar bir web sitesine erişmek için tarayıcılarına bir alan adı yazdıklarında (örneğin, www.ornek.com), DNS sistemi bu alan adının A kaydını sorgular. DNS sunucusu, www.ornek.com ile eşleşen A kaydında belirtilen IPv4 adresini (örneğin, 203.0.113.15) kullanıcıya döndürür. Ardından kullanıcının cihazı, bu IP adresini kullanarak web sitesinin barındığı sunucuya bağlanır.

Özetle A kaydı:

- Bir alan adını (veya alt alan adını) bir IPv4 adresiyle ilişkilendirir.
- Kullanıcıların alan adları aracılığıyla web siteleri, sunucular ve diğer internet kaynaklarına erişmesini sağlar.
- DNS çözümleme sürecinin temelini oluşturur.

Bir alan adının çalışabilmesi ve internet üzerinden erişilebilmesi için genellikle en az bir adet A kaydına sahip olması gerekir. Örneğin, ornek.com alan adının web sitesine yönlenmesi için bu alan adına ait bir A kaydı, web sitesinin barındığı sunucunun IPv4 adresini göstermelidir. Benzer şekilde, blog.ornek.com gibi bir alt alan adının farklı bir sunucuya yönlenmesi istenirse, bu alt alan adı için ayrı bir A kaydı oluşturulur.

IPv6 adresleri için ise A kaydının muadili **AAAA kaydı** kullanılır.

Veri Tabanı Temelleri

- NoSQL veri tabanı nedir? Özellikleri nelerdir?

NoSQL Veritabanı, geleneksel ilişkisel (SQL) veritabanlarından farklı olarak, verileri tablo ve ilişkisel model dışında farklı şekillerde depolayan ve yöneten veritabanı türleridir. "Not only SQL" (Sadece SQL değil) veya "Non-relational" (İlişkisel olmayan) olarak da adlandırılırlar. Büyük hacimli, hızla değişen, yapısal veya yarı yapısal verilere daha esnek ve ölçeklenebilir çözümler sunmak amacıyla ortaya çıkmışlardır.

Özellikleri Nelerdir?

NoSQL veritabanlarının bir veya daha fazlasına sahip olabileceği çeşitli temel özellikleri vardır:

- **Esnek Şema (Schema-less veya Flexible Schema):** İlişkisel veritabanlarının aksine, önceden tanımlanmış katı bir şema gerektirmezler. Aynı veritabanındaki farklı belgeler veya öğeler farklı alanlara sahip olabilir. Bu durum, uygulama geliştirme süreçlerinde hızlı değişikliklere ve veri yapısındaki evrime kolayca uyum sağlamayı mümkün kılar.
- **Yatay Ölçeklenebilirlik (Horizontal Scalability):** Trafik veya veri hacmi arttığında, mevcut sunucuların kapasitesini artırmak (dikey ölçeklendirme) yerine, sisteme ek sunucular ekleyerek (yatay ölçeklendirme) kolayca genişleyebilirler. Bu, büyük veri kümelerini ve yüksek trafiği yönetmek için önemlidir.

- **Çeşitli Veri Modelleri (Various Data Models):** NoSQL tek bir veri modeline bağlı değildir. Farklı ihtiyaçlara yönelik çeşitli veri modelleri sunar:
    - **Anahtar-Değer (Key-Value):** Veriyi basit anahtar-değer çiftleri halinde depolar. (Örnek: Redis, DynamoDB)
    - **Belge Tabanlı (Document-Oriented):** Veriyi genellikle JSON veya XML gibi belge formatlarında depolar. Her belge kendi içinde yapılandırılmış olabilir. (Örnek: MongoDB, Couchbase)
    - **Sütun Ailesi Tabanlı (Column-Family):** Veriyi satırlar yerine sütun aileleri halinde depolar. Geniş sütunlara sahip ve seyrek veriler için optimize edilmiştir. (Örnek: Cassandra, HBase)
    - **Graf Tabanlı (Graph-Based):** Veriyi düğümler (entity) ve kenarlar (ilişkiler) olarak depolar. İlişkisel verilerin karmaşık ağlarını modellemek için idealdir. (Örnek: Neo4j, ArangoDB)
- **Dağıtılmış Hesaplama ve Veri Dağıtımı:** Verileri birden fazla sunucuya dağıtarak yüksek erişilebilirlik ve hata toleransı sağlarlar. Veri, ağdaki farklı düğümlere çoğaltılabilir.
- **Nihai Tutarlılık (Eventual Consistency):** Çoğu NoSQL veritabanı, dağıtılmış yapıları nedeniyle anlık tutarlılık yerine nihai tutarlılığı tercih eder. Bu, bir veride yapılan değişikliğin tüm kopyalarına hemen yansımayabileceği, ancak zamanla sistemin tutarlı hale geleceği anlamına gelir. Bu durum genellikle performans ve ölçeklenebilirlik lehine bir dengedir.
- **Yüksek Performans ve Erişilebilirlik:** Özel veri modelleri ve yatay ölçeklenebilirlik sayesinde belirli iş yüklerinde ilişkisel veritabanlarından daha yüksek performans ve sürekli erişilebilirlik sunabilirler.

NoSQL veritabanları, büyük veri analizi, gerçek zamanlı web uygulamaları, içerik yönetim sistemleri, mobil uygulamalar ve IoT (Nesnelerin İnterneti) gibi alanlarda sıklıkla tercih edilmektedir. İlişkisel veritabanlarının ACID (Atomicity, Consistency, Isolation, Durability) garantilerini tam olarak sağlamasalar da, modern uygulamaların değişen ve büyük ölçekli veri ihtiyaçlarına daha uygun çözümler sunarlar.

SQL Server Veri Tipleri nelerdir?

SQL Server, bir veritabanında depolanacak farklı türdeki bilgileri tanımlamak için çeşitli veri tipleri sunar. Bu veri tipleri, sütunlarda ne tür verilerin saklanabileceğini belirler ve veritabanının verimli ve doğru çalışmasını sağlar. İşte SQL Server'daki bazı yaygın veri tipleri ve örnekleri:

1. Tam Sayısal Veri Tipleri (Exact Numerics)

Tam sayıları depolamak için kullanılır. Farklı boyutlarda ve değer aralıklarında seçenekler sunarlar.

- **TINYINT:** 0 ile 255 arasındaki tam sayıları saklar (1 byte).

SQL

```
CREATE TABLE Urunler (
  UrunID INT PRIMARY KEY,
  StokAdedi TINYINT
);

INSERT INTO Urunler (UrunID, StokAdedi) VALUES (1, 50);
```

- **SMALLINT:** -32,768 ile 32,767 arasındaki tam sayıları saklar (2 bytes).

SQL

```
CREATE TABLE Siparisler (
  SiparisID INT PRIMARY KEY,
  MusteriSayisi SMALLINT
);

INSERT INTO Siparisler (SiparisID, MusteriSayisi) VALUES (101, 150);
```

- **INT:** -2,147,483,648 ile 2,147,483,647 arasındaki tam sayıları saklar (4 bytes). Genellikle en yaygın tam sayı tipidir.

SQL

```
CREATE TABLE Kullanicilar (
  KullaniciID INT PRIMARY KEY,
  Yas INT
);

INSERT INTO Kullanicilar (KullaniciID, Yas) VALUES (1001, 30);
```

- **BIGINT:** Çok büyük veya çok küçük tam sayıları saklar (-9,223,372,036,854,775,808 ile 9,223,372,036,854,775,807 arası) (8 bytes).

SQL

```
CREATE TABLE BuyukSayilar (
  ID INT PRIMARY KEY,
  Nufus BIGINT
);

INSERT INTO BuyukSayilar (ID, Nufus) VALUES (1, 80000000);
```

- **DECIMAL / NUMERIC:** Sabit ondalık sayıları saklar. Duyarlılık (precision) ve ölçek (scale) belirtilebilir. Para birimleri gibi hassas değerler için idealdir.

SQL

```
CREATE TABLE Finans (
  IslemID INT PRIMARY KEY,
  Tutar DECIMAL(10, 2) -- Toplam 10 basamak, virgülden sonra 2 basamak
);

INSERT INTO Finans (IslemID, Tutar) VALUES (201, 1234.56);
```

- **MONEY / SMALLMONEY:** Para birimi değerlerini saklar. MONEY daha geniş bir aralığa sahiptir.

SQL

```
CREATE TABLE Satislar (
  SatisID INT PRIMARY KEY,
  Fiyat MONEY
);

INSERT INTO Satislar (SatisID, Fiyat) VALUES (301, 199.99);
```

- **BIT:** Boolean (doğru/yanlış) değerleri saklar (0, 1 veya NULL).

SQL

```
CREATE TABLE Ayarlar (
  AyarID INT PRIMARY KEY,
```

```
  Aktif BIT
);

INSERT INTO Ayarlar (AyarID, Aktif) VALUES (1, 1); -- 1 doğruyu temsil eder
```

2. Yaklaşık Sayısal Veri Tipleri (Approximate Numerics)

Ondalık sayıları yaklaşık değerleriyle saklar. Bilimsel hesaplamalar veya çok büyük/küçük kayan noktalı sayılar için kullanılır, ancak hassasiyet garantisi vermez.

- **REAL:** Tek duyarlıklı kayan noktalı sayılar (4 bytes).
- **FLOAT:** Çift duyarlıklı kayan noktalı sayılar (8 bytes). Duyarlılık parantez içinde belirtilebilir (float(n)), bu da depolama boyutunu etkileyebilir.

SQL

```
CREATE TABLE Olcumler (
  OlcumID INT PRIMARY KEY,
  Deger FLOAT
);

INSERT INTO Olcumler (OlcumID, Deger) VALUES (401, 3.14159);
```

3. Tarih ve Saat Veri Tipleri (Date and Time)

Tarih ve saat bilgilerini saklamak için kullanılır.

- **DATE:** Yalnızca tarih bilgisini saklar (YYYY-MM-DD) (3 bytes).

SQL

```
CREATE TABLE Etkinlikler (
  EtkinlikID INT PRIMARY KEY,
  EtkinlikTarihi DATE
);

INSERT INTO Etkinlikler (EtkinlikID, EtkinlikTarihi) VALUES (501, '2025-12-31');
```

- **TIME:** Yalnızca saat bilgisini saklar (HH:MM:SS[.kesir]) (3-5 bytes).

SQL

```
CREATE TABLE Toplantilar (
  ToplantiID INT PRIMARY KEY,
  ToplantiSaati TIME(7) -- Saniyenin kesir kısmı için duyarlılık
);

INSERT INTO Toplantilar (ToplantiID, ToplantiSaati) VALUES (601, '14:30:00.1234567');
```

- **DATETIME:** Tarih ve saat bilgisini saklar (1753-01-01 ile 9999-12-31 arası) (8 bytes). Daha düşük hassasiyete sahiptir.

SQL

```
CREATE TABLE Kayitlar (
  KayitID INT PRIMARY KEY,
  KayitZamani DATETIME
);

INSERT INTO Kayitlar (KayitID, KayitZamani) VALUES (701, '2025-05-08 19:00:00');
```

- **SMALLDATETIME:** Daha küçük bir tarih ve saat aralığını saklar (1900-01-01 ile 2079-06-06 arası) (4 bytes). Hassasiyeti dakikadır.

SQL

```
CREATE TABLE Loglar (
  LogID INT PRIMARY KEY,
  LogZamani SMALLDATETIME
);

INSERT INTO Loglar (LogID, LogZamani) VALUES (801, '2025-05-08 19:05');
```

- **DATETIME2:** Daha geniş bir tarih ve saat aralığını ve daha yüksek hassasiyeti saklar (0001-01-01 00:00:00.0000000 ile 9999-12-31 23:59:59.9999999 arası) (6-8 bytes).

SQL

```
CREATE TABLE Islemler (
```

```
  IslemLogID INT PRIMARY KEY,
  IslemZamani DATETIME2(7)
);
```

```
INSERT INTO Islemler (IslemLogID, IslemZamani) VALUES (901, '2025-05-08 19:06:19.1234567');
```

- **DATETIMEOFFSET:** Tarih, saat ve saat dilimi ofset bilgisini saklar (8-10 bytes).

SQL

```
CREATE TABLE GlobalEtkinlikler (
  EtkinlikID INT PRIMARY KEY,
  EtkinlikZamani DATETIMEOFFSET(7)
);
```

```
INSERT INTO GlobalEtkinlikler (EtkinlikID, EtkinlikZamani) VALUES (1001, '2025-05-08 19:06:19.1234567 +03:00');
```

4. Karakter Dizisi Veri Tipleri (Character Strings)

ASCII karakterlerini içeren metin verilerini saklar.

- **CHAR(n):** Belirtilen uzunlukta (n) sabit uzunluklu karakter dizisi saklar (1 ile 8000 arası). Belirtilen uzunluktan kısa veri girilirse boşluklarla doldurulur.

SQL

```
CREATE TABLE Kodu (
  UlkeKodu CHAR(2) PRIMARY KEY
);
```

```
INSERT INTO Kodu (UlkeKodu) VALUES ('TR');
```

- **VARCHAR(n):** Belirtilen maksimum uzunlukta (n) değişken uzunluklu karakter dizisi saklar (1 ile 8000 arası). Depolama boyutu girilen verinin uzunluğuna göre değişir.

SQL

```
CREATE TABLE KullaniciBilgileri (
```

```
    KullaniciID INT PRIMARY KEY,
    Ad VARCHAR(50)
);
```

```
INSERT INTO KullaniciBilgileri (KullaniciID, Ad) VALUES (1, 'Ahmet');
```

- **VARCHAR(MAX):** 2^31-1 byte'a kadar değişken uzunluklu karakter dizisi saklar. Büyük metin blokları için kullanılır (örn: makale içerikleri).

SQL

```
CREATE TABLE Makaleler (
    MakaleID INT PRIMARY KEY,
    Icerik VARCHAR(MAX)
);
```

```
INSERT INTO Makaleler (MakaleID, Icerik) VALUES (1, 'Bu çok uzun bir makale içeriği...');
```

- **TEXT:** VARCHAR(MAX)'in eski karşılığıdır ve ileriki sürümlerde kaldırılması planlanmaktadır. Yeni geliştirmelerde kullanılması önerilmez.

5. Unicode Karakter Dizisi Veri Tipleri (Unicode Character Strings)

Farklı dillerdeki karakterleri (Unicode) içeren metin verilerini saklar. Her karakter genellikle 2 byte yer kaplar.

- **NCHAR(n):** Belirtilen uzunlukta (n) sabit uzunluklu Unicode karakter dizisi saklar (1 ile 4000 arası).

SQL

```
CREATE TABLE Diller (
    DilKodu NCHAR(2) PRIMARY KEY
);
```

```
INSERT INTO Diller (DilKodu) VALUES (N'TR'); -- Unicode sabitleri için 'N' ön eki kullanılır
```

- **NVARCHAR(n):** Belirtilen maksimum uzunlukta (n) değişken uzunluklu Unicode karakter dizisi saklar (1 ile 4000 arası).

SQL

```
CREATE TABLE Ceviriler (
  CeviriID INT PRIMARY KEY,
  Metin NVARCHAR(100)
);

INSERT INTO Ceviriler (CeviriID, Metin) VALUES (1, N'Merhaba Dünya');
```

- **NVARCHAR(MAX):** 2^31-1 byte'a kadar değişken uzunluklu Unicode karakter dizisi saklar. Büyük Unicode metin blokları için kullanılır.

SQL

```
CREATE TABLE Kitaplar (
  KitapID INT PRIMARY KEY,
  Ozet NVARCHAR(MAX)
);

INSERT INTO Kitaplar (KitapID, Ozet) VALUES (1, N'Bu kitabın uzun özeti...');
```

- **NTEXT:** NVARCHAR(MAX)'in eski karşılığıdır ve ileriki sürümlerde kaldırılması planlanmaktadır. Yeni geliştirmelerde kullanılması önerilmez.

6. İkili Veri Tipleri (Binary Strings)

İkili veriyi (bayt dizileri) saklamak için kullanılır. Resimler, belgeler veya diğer dosya içerikleri için kullanılabilir.

- **BINARY(n):** Belirtilen uzunlukta (n) sabit uzunluklu ikili veri saklar (1 ile 8000 arası).
- **VARBINARY(n):** Belirtilen maksimum uzunlukta (n) değişken uzunluklu ikili veri saklar (1 ile 8000 arası).

SQL

```
CREATE TABLE Dosyalar (
  DosyaID INT PRIMARY KEY,
```

```
  DosyaIcerigi VARBINARY(MAX) -- Büyük ikili veriler için MAX kullanılır
);

-- Dosya içeriği genellikle uygulama tarafından ikili veri olarak sağlanır.
```

- **IMAGE:** VARBINARY(MAX)'in eski karşılığıdır ve ileriki sürümlerde kaldırılması planlanmaktadır. Yeni geliştirmelerde kullanılması önerilmez.

7. Diğer Veri Tipleri

Yukarıdaki kategorilere girmeyen özel amaçlı veri tipleri.

- **UNIQUEIDENTIFIER:** Küresel olarak benzersiz tanımlayıcı (GUID) saklar (16 bytes).

SQL

```
CREATE TABLE KayitDefteri (
  LogID UNIQUEIDENTIFIER PRIMARY KEY DEFAULT NEWID(), -- Otomatik GUID oluşturur
  IslemAciklama VARCHAR(100)
);

INSERT INTO KayitDefteri (IslemAciklama) VALUES ('Kullanıcı girişi');
```

- **XML:** XML verilerini saklar.

SQL

```
CREATE TABLE UrunDetay (
  UrunID INT PRIMARY KEY,
  Detay XML
);

INSERT INTO UrunDetay (UrunID, Detay) VALUES (1,
'<urun><renk>Kırmızı</renk><beden>M</beden></urun>');
```

- **GEOMETRY / GEOGRAPHY:** Mekansal (konumsal) veri saklar. GEOMETRY düzlem koordinat sistemlerini, GEOGRAPHY ise yuvarlak dünya koordinat sistemlerini (enlem/boylam) temsil eder.

SQL

```
CREATE TABLE Lokasyonlar (
  LokasyonID INT PRIMARY KEY,
  Konum GEOGRAPHY
);


-- Örnek ekleme (enlem ve boylam ile bir nokta oluşturma)
-- INSERT INTO Lokasyonlar (LokasyonID, Konum) VALUES (1, geography::Point(41.0082, 28.9784, 4326)); -- Örnek WKT formatı ve SRID
```

- **ROWVERSION:** Bir tablodaki satırların sürümünü belirten otomatik olarak oluşturulan ikili sayıdır. Genellikle eşzamanlılık kontrolü için kullanılır. Eski adı TIMESTAMP'tir, ancak tarih/saat ile ilgisi yoktur.

SQL

```
CREATE TABLE Envanter (
  UrunID INT PRIMARY KEY,
  UrunAdi VARCHAR(50),
  SonGuncelleme ROWVERSION -- Otomatik olarak güncellenir
);
```

CRUD İşlemleri nedir? (Create, Read, Update, Delete)

CRUD İşlemleri, yazılım geliştirme ve veritabanı yönetiminde kalıcı depolama (persistent storage) üzerindeki dört temel işlemi temsil eden bir kısaltmadır. İngilizce **C**reate, **R**ead, **U**pdate ve **D**elete kelimelerinin baş harflerinden oluşur. Bu dört işlem, bir veri kümesi (örneğin bir veritabanı tablosundaki kayıtlar) üzerinde yapılabilecek en temel manipülasyonları kapsar.

İşte CRUD işlemlerinin her biri ve ne anlama geldikleri:

1. **Create (Oluşturma):**
    - Bu işlem, veritabanına veya kalıcı depolama alanına yeni veri eklemek için kullanılır.
    - İlişkisel veritabanlarında (SQL) genellikle INSERT komutu ile gerçekleştirilir.

- Örnek: Bir kullanıcı kayıt formunda yeni bir kullanıcı oluşturma işlemi.

SQL

```
-- 'Kullanicilar' tablosuna yeni bir kayıt ekleme
INSERT INTO Kullanicilar (Ad, Soyad, Eposta)
VALUES ('Ahmet', 'Yılmaz', 'ahmet.yilmaz@example.com');
```

2. **Read (Okuma / Listeleme):**
    - Bu işlem, veritabanında bulunan mevcut veriyi almak veya görüntülemek için kullanılır.
    - Belirli kriterlere göre veri sorgulama, filtreleme veya tüm veriyi listeleme bu kategoriye girer.
    - İlişkisel veritabanlarında genellikle SELECT komutu ile gerçekleştirilir.
    - Örnek: Bir web sitesinde ürün listesini görüntüleme veya belirli bir kullanıcının bilgilerini çekme.

SQL

```
-- 'Urunler' tablosundaki tüm kayıtları listeleme
SELECT * FROM Urunler;


-- 'Kullanicilar' tablosundan belirli bir kullanıcının bilgilerini çekme
SELECT * FROM Kullanicilar WHERE KullaniciID = 101;
```

3. **Update (Güncelleme):**
    - Bu işlem, veritabanında bulunan mevcut veriyi değiştirmek veya güncellemek için kullanılır.
    - Genellikle belirli bir kaydın veya kayıtların belirli alanlarındaki veriyi değiştirmeyi içerir.
    - İlişkisel veritabanlarında genellikle UPDATE komutu ile gerçekleştirilir.
    - Örnek: Bir kullanıcının e-posta adresini veya telefon numarasını güncelleme.

SQL

```
-- 'Kullanicilar' tablosunda KullaniciID'si 101 olan kaydın Eposta alanını güncelleme
UPDATE Kullanicilar
```

SET Eposta = 'yeni.eposta@example.com'

WHERE KullaniciID = 101;

4. **Delete (Silme):**
    - Bu işlem, veritabanında bulunan mevcut veriyi silmek veya kaldırmak için kullanılır.
    - Belirli bir kaydı veya belirli kriterlere uyan birden fazla kaydı silmeyi içerebilir.
    - İlişkisel veritabanlarında genellikle DELETE komutu ile gerçekleştirilir.
    - Örnek: Bir kullanıcının hesabını silme veya eski sipariş kayıtlarını kaldırma.

SQL

-- 'Urunler' tablosundan UrunID'si 505 olan kaydı silme

DELETE FROM Urunler WHERE UrunID = 505;

-- 'Siparisler' tablosundan tarihi 1 yıldan eski olan tüm kayıtları silme

-- (Gerçek senaryoda daha dikkatli koşullar kullanılır)

-- DELETE FROM Siparisler WHERE SiparisTarihi < DATEADD(year, -1, GETDATE());

CRUD işlemleri, bir uygulamanın veya sistemin verilerle etkileşimde bulunabilmesi için en temel yapı taşlarını oluşturur. Çoğu yazılım uygulaması, kullanıcıların veri oluşturmasına, okumasına, güncellemesine ve silmesine olanak tanıyan arayüzler ve işlevler sunar ve bu işlevlerin altında yatan temel prensip CRUD'dir. Web servislerinde (REST API'ler gibi) de GET, POST, PUT, DELETE gibi HTTP metotları genellikle CRUD işlemlerine karşılık gelir.

Foreign Key nedir? Nasıl kullanılır?

Veritabanı yönetim sistemlerinde **Foreign Key (Yabancı Anahtar)**, bir tablodaki (çoğunlukla "alt tablo" veya "çocuk tablo" olarak adlandırılır) bir sütun veya sütunlar kümesinin, başka bir tablodaki (genellikle "ana tablo" veya "ebeveyn tablo" olarak adlandırılır) birincil anahtarına (Primary Key) veya benzersiz anahtarına (Unique Key) referans veren bir kısıtlamadır.

Ne İşe Yarar?

Foreign Key'lerin temel amacı, ilişkisel veritabanlarında tablolar arasında bağlantı kurmak ve **referans bütünlüğünü (referential integrity)** sağlamaktır. Referans

bütünlüğü, ilişkili tablolardaki verilerin tutarlı ve geçerli olmasını garanti eder. Foreign Key kullanımı sayesinde:

1. **Tablolar Arası İlişki Kurulur:** İki tablo arasında mantıksal bir bağlantı oluşturulur. Örneğin, bir "Siparisler" tablosundaki "MusteriID" sütunu, "Musteriler" tablosundaki "MusteriID" (Primary Key) sütununa Foreign Key olarak bağlanabilir. Bu, hangi siparişin hangi müşteriye ait olduğunu gösteren bir ilişki kurar.
2. **Veri Tutarlılığı Sağlanır:** Alt tablodaki Foreign Key sütununa girilen değerlerin ana tablonun referans gösterilen sütununda (Primary Key veya Unique Key) var olması zorunlu hale getirilebilir. Bu sayede, var olmayan bir müşteriye ait sipariş gibi tutarsız verilerin girilmesi engellenir.
3. **"Yetim" Kayıtlar Önlenir:** Ana tablodan bir kayıt silindiğinde veya güncellendiğinde, bu kayda bağlı alt tablodaki kayıtların (Foreign Key aracılığıyla ilişkili olanların) ne olacağı belirlenerek "yetim" (ilişkisi kopmuş) kayıtların oluşması önlenir.

Nasıl Kullanılır?

Foreign Key kısıtlamaları, genellikle tablo oluşturulurken (CREATE TABLE) veya mevcut bir tablo değiştirilirken (ALTER TABLE) tanımlanır. SQL sözdizimi aşağıdaki gibidir:

Tablo Oluşturulurken Foreign Key Tanımlama:

SQL

```
CREATE TABLE AnaTablo (
  AnaTabloID INT PRIMARY KEY,
  Ad VARCHAR(50)
);


CREATE TABLE AltTablo (
  AltTabloID INT PRIMARY KEY,
  Detay VARCHAR(100),
  AnaTabloID INT, -- Bu sütun Foreign Key olacak

  FOREIGN KEY (AnaTabloID) REFERENCES AnaTablo(AnaTabloID)
```

-- AltTabloID sütununu AnaTablo tablosunun AnaTabloID sütununa Foreign Key olarak bağlar

);

Mevcut Tabloya Foreign Key Ekleme:

SQL

-- Mevcut AltTablo'ya Foreign Key kısıtlaması ekleme

ALTER TABLE AltTablo

ADD FOREIGN KEY (AnaTabloID) REFERENCES AnaTablo(AnaTabloID);

CONSTRAINT ile Foreign Key Adı Belirleme (Daha Okunabilir ve Yönetilebilir):

SQL

CREATE TABLE AltTablo (

AltTabloID INT PRIMARY KEY,

Detay VARCHAR(100),

AnaTabloID INT,

CONSTRAINT FK_AltTablo_AnaTablo -- Foreign Key'e bir isim verme

FOREIGN KEY (AnaTabloID) REFERENCES AnaTablo(AnaTabloID)

);

-- veya ALTER TABLE ile

ALTER TABLE AltTablo

ADD CONSTRAINT FK_AltTablo_AnaTablo -- Foreign Key'e bir isim verme

FOREIGN KEY (AnaTabloID) REFERENCES AnaTablo(AnaTabloID);

ON UPDATE ve ON DELETE Eylemleri:

Foreign Key tanımlarken, ana tablodaki referans alınan kayıt güncellendiğinde (ON UPDATE) veya silindiğinde (ON DELETE) alt tablodaki ilişkili kayıtlar için ne yapılacağını belirtebilirsiniz. Yaygın eylemler şunlardır:

- **NO ACTION (Varsayılan):** Ana tablodaki ilgili kayıt silinmeye veya güncellenmeye çalışıldığında, alt tabloda bu kayda referans veren kayıtlar varsa işleme izin verilmez (hata döner).

- **CASCADE:** Ana tablodaki ilgili kayıt silindiğinde veya güncellendiğinde, alt tablodaki bu kayda referans veren kayıtlar da otomatik olarak silinir veya güncellenir.

SQL

FOREIGN KEY (AnaTabloID) REFERENCES AnaTablo(AnaTabloID)

ON DELETE CASCADE

ON UPDATE CASCADE;

- **SET NULL:** Ana tablodaki ilgili kayıt silindiğinde veya güncellendiğinde, alt tablodaki bu kayda referans veren Foreign Key sütunlarının değeri NULL olarak ayarlanır. (Foreign Key sütununun NULL değerlere izin vermesi gerekir).

SQL

FOREIGN KEY (AnaTabloID) REFERENCES AnaTablo(AnaTabloID)

ON DELETE SET NULL;

- **SET DEFAULT:** Ana tablodaki ilgili kayıt silindiğinde veya güncellendiğinde, alt tablodaki bu kayda referans veren Foreign Key sütunlarının değeri DEFAULT olarak belirlenmiş değere ayarlanır. (Foreign Key sütununda DEFAULT kısıtlaması olmalıdır).

SQL

FOREIGN KEY (AnaTabloID) REFERENCES AnaTablo(AnaTabloID)

ON DELETE SET DEFAULT;

Örnek Kullanım Senaryosu:

Bir "Musteriler" ve bir "Siparisler" tablonuz olduğunu varsayalım. Her siparişin bir müşteriye ait olması gerektiğini garantilemek istiyorsunuz.

SQL

-- Ana Tablo: Musteriler

CREATE TABLE Musteriler (

  MusteriID INT PRIMARY KEY,

  MusteriAdi VARCHAR(50)

);

-- Alt Tablo: Siparisler

```
CREATE TABLE Siparisler (
    SiparisID INT PRIMARY KEY,
    SiparisTarihi DATE,
    MusteriID INT, -- Hangi müşteriye ait olduğunu belirtir

    -- MusteriID sütununu Musteriler tablosunun MusteriID sütununa bağlayan Foreign Key
    FOREIGN KEY (MusteriID) REFERENCES Musteriler(MusteriID)
    ON DELETE CASCADE -- Eğer bir müşteri silinirse, o müşteriye ait tüm siparişler de silinsin
);

-- Veri Ekleme (Geçerli Senaryo)
INSERT INTO Musteriler (MusteriID, MusteriAdi) VALUES (1, 'Ali');
INSERT INTO Siparisler (SiparisID, SiparisTarihi, MusteriID) VALUES (101, '2024-01-15', 1); -- Geçerli MusteriID

-- Veri Ekleme (Geçersiz Senaryo - Hata Verecektir)
-- INSERT INTO Siparisler (SiparisID, SiparisTarihi, MusteriID) VALUES (102, '2024-01-16', 99); -- Musteriler tablosunda 99 ID'li müşteri yok

-- Veri Silme (ON DELETE CASCADE Örneği)
-- DELETE FROM Musteriler WHERE MusteriID = 1; -- Bu komut çalıştırıldığında, Siparisler tablosundaki MusteriID'si 1 olan tüm kayıtlar da otomatik olarak silinir.
```

Foreign Key'ler, veritabanı tasarımının önemli bir parçasıdır ve ilişkisel veritabanlarının gücünü artıran, veri bütünlüğünü sağlayan temel mekanizmalardır. Doğru kullanıldığında, veritabanınızdaki verilerin güvenilirliğini ve tutarlılığını önemli ölçüde artırır.

Join işlemleri nedir? Çeşitleri nelerdir? (Inner Join, Left Join, vb.)

Veritabanlarında **JOIN işlemleri**, aralarında ilgili bir sütun (genellikle Foreign Key ve Primary Key ilişkisi olan sütunlar) bulunan iki veya daha fazla tablodaki satırları

birleştirerek tek bir sonuç kümesi elde etmek için kullanılır. Normalizasyon prensipleri gereği veriler genellikle farklı tablolara dağıtılır. JOIN işlemleri, bu dağıtılmış verileri anlamlı bir şekilde bir araya getirmeyi sağlar.

JOIN Çeşitleri Nelerdir?

SQL'de başlıca beş tür JOIN işlemi bulunur:

1. **INNER JOIN (İç Birleştirme):**
    - En yaygın kullanılan JOIN türüdür.
    - İki tablodaki birleştirme koşulunu (ON şartı) sağlayan **eşleşen** satırları getirir.
    - Eşleşmeyen satırlar sonuç kümesine dahil edilmez.
    - Anahtar kelime olarak sadece JOIN kullanıldığında da varsayılan olarak INNER JOIN kabul edilir.
    - **Örnek Senaryo:** "Musteriler" ve "Siparisler" tablolarını kullanarak, siparişi olan müşterilerin listesini ve sipariş bilgilerini getirme.
    - **Tablo Yapıları:** Musteriler tablosu: | MusteriID | MusteriAdi | |-----------|-----------| | 1 | Ali | | 2 | Ayşe | | 3 | Can |

Siparisler tablosu: | SiparisID | MusteriID | SiparisTutari | |-----------|-----------|---------------| | 101 | 1 | 150.00 | | 102 | 2 | 220.50 | | 103 | 1 | 85.75 | | 104 | 4 | 300.00 | -- Bu siparişin Musteriler tablosunda karşılığı yok

    - **SQL Sorgusu:**

SQL

SELECT Musteriler.MusteriAdi, Siparisler.SiparisID, Siparisler.SiparisTutari

FROM Musteriler

INNER JOIN Siparisler ON Musteriler.MusteriID = Siparisler.MusteriID;

    - **Sonuç:** | MusteriAdi | SiparisID | SiparisTutari | |------------|-----------|---------------| | Ali | 101 | 150.00 | | Ayşe | 102 | 220.50 | | Ali | 103 | 85.75 | -- Can (Musteriler tablosunda siparişi yok) ve SiparisID 104 (Siparisler tablosunda müşterisi yok) sonuçta yer almaz.

2. **LEFT JOIN (Sol Dış Birleştirme - LEFT OUTER JOIN):**
    - Sol tablodaki tüm satırları getirir.
    - Sağ tablodan ise yalnızca birleştirme koşulunu sağlayan eşleşen satırları getirir.

- Sol tablodaki bir satırın sağ tabloda karşılığı yoksa, sağ tablodaki sütunlar için NULL değerleri döner.
- OUTER anahtar kelimesi genellikle isteğe bağlıdır ve LEFT JOIN kullanımı yeterlidir.
- **Örnek Senaryo:** Siparişi olsun veya olmasın tüm müşterilerin listesini ve eğer varsa sipariş bilgilerini getirme.
- **SQL Sorgusu:**

SQL

SELECT Musteriler.MusteriAdi, Siparisler.SiparisID, Siparisler.SiparisTutari

FROM Musteriler

LEFT JOIN Siparisler ON Musteriler.MusteriID = Siparisler.MusteriID;

- **Sonuç:** | MusteriAdi | SiparisID | SiparisTutari | |------------|-----------|------------| | Ali | 101 | 150.00 | | Ayşe | 102 | 220.50 | | Ali | 103 | 85.75 | | Can | NULL | NULL | -- Can'ın siparişi olmadığı için sağ tablo sütunları NULL gelir

3. **RIGHT JOIN (Sağ Dış Birleştirme - RIGHT OUTER JOIN):**
   - Sağ tablodaki tüm satırları getirir.
   - Sol tablodan ise yalnızca birleştirme koşulunu sağlayan eşleşen satırları getirir.
   - Sağ tablodaki bir satırın sol tabloda karşılığı yoksa, sol tablodaki sütunlar için NULL değerleri döner.
   - OUTER anahtar kelimesi genellikle isteğe bağlıdır ve RIGHT JOIN kullanımı yeterlidir.
   - **Örnek Senaryo:** Müşterisi olsun veya olmasın tüm siparişlerin listesini ve eğer varsa müşteri bilgilerini getirme.
   - **SQL Sorgusu:**

SQL

SELECT Musteriler.MusteriAdi, Siparisler.SiparisID, Siparisler.SiparisTutari

FROM Musteriler

RIGHT JOIN Siparisler ON Musteriler.MusteriID = Siparisler.MusteriID;

- **Sonuç:** | MusteriAdi | SiparisID | SiparisTutari | |------------|-----------|------------| | Ali | 101 | 150.00 | | Ayşe | 102 | 220.50 | | Ali | 103 | 85.75 | | NULL |

104 | 300.00 | -- Sipariş 104'ün müşterisi olmadığı için sol tablo sütunları NULL gelir

4. **FULL OUTER JOIN (Tam Dış Birleştirme):**
    - Her iki tablodaki tüm satırları getirir.
    - Birleştirme koşulunu sağlayan eşleşen satırlar normal şekilde birleştirilir.
    - Sol tablodaki bir satırın sağ tabloda karşılığı yoksa, sağ tablo sütunları NULL döner.
    - Sağ tablodaki bir satırın sol tabloda karşılığı yoksa, sol tablo sütunları NULL döner.
    - Eşleşen ve eşleşmeyen tüm satırları sonuç kümesine dahil eder.
    - Anahtar kelime olarak FULL OUTER JOIN veya sadece FULL JOIN kullanılır.
    - **Örnek Senaryo:** Tüm müşterilerin (siparişi olan veya olmayan) ve tüm siparişlerin (müşterisi olan veya olmayan) listesini getirme.
    - **SQL Sorgusu:**

SQL

SELECT Musteriler.MusteriAdi, Siparisler.SiparisID, Siparisler.SiparisTutari

FROM Musteriler

FULL OUTER JOIN Siparisler ON Musteriler.MusteriID = Siparisler.MusteriID;

    - **Sonuç:** | MusteriAdi | SiparisID | SiparisTutari | |------------|-----------|------------| | Ali | 101 | 150.00 | | Ayşe | 102 | 220.50 | | Ali | 103 | 85.75 | | Can | NULL | NULL | | NULL | 104 | 300.00 |

5. **CROSS JOIN (Çapraz Birleştirme):**
    - İki tablonun tüm satırlarının **Kartezyen Çarpımını** döndürür.
    - Birleştirme koşulu (ON şartı) kullanılmaz.
    - Sonuç kümesindeki toplam satır sayısı, birinci tablodaki satır sayısı ile ikinci tablodaki satır sayısının çarpımına eşittir.
    - Genellikle dikkatli kullanılmalıdır, çünkü büyük tablolar için çok büyük sonuç kümeleri üretebilir.
    - **Örnek Senaryo:** Her müşteriyi her siparişle eşleştirme.
    - **SQL Sorgusu:**

SQL

```
SELECT Musteriler.MusteriAdi, Siparisler.SiparisID

FROM Musteriler

CROSS JOIN Siparisler;
```

- **Sonuç:** (Musteriler tablosunda 3, Siparisler tablosunda 4 kayıt olduğu varsayıldığında, 3 * 4 = 12 satır döner) | MusteriAdi | SiparisID | |------------|----------| | Ali | 101 | | Ali | 102 | | Ali | 103 | | Ali | 104 | | Ayşe | 101 | | Ayşe | 102 | | Ayşe | 103 | | Ayşe | 104 | | Can | 101 | | Can | 102 | | Can | 103 | | Can | 104 |

Bu farklı JOIN türleri, veritabanındaki ilişkili verileri sorgularken ve birleştirirken farklı ihtiyaçlara yönelik esneklik sağlarlar. Hangi JOIN türünün kullanılacağı, istenen sonuç kümesinin ne tür satırları içermesi gerektiğine bağlıdır.