# \*\*Veritabanı ve SQL ile İlgili Temel Kavramlar\*\*

## 1. \*\*Veritabanı Nedir?\*\*

Veritabanı, bilgileri organize bir şekilde depolamak, yönetmek ve paylaşmak için tasarlanmış bir sistemdir. Veriler, tablolar (tables) şeklinde saklanır ve ilişkisel yapılar aracılığıyla yönetilir. Örneğin, bir e-ticaret sitesinde müşteriler, ürünler ve siparişler gibi veriler bir veritabanında yönetilebilir.

---

## 2. \*\*SQL Kavramı Nedir?\*\*

\*\*SQL (Structured Query Language)\*\*, veritabanlarıyla iletişim kurmak için kullanılan standart bir programlama dilidir. Verileri sorgulama, ekleme, silme, güncelleme ve veritabanı yapısını yönetme işlemleri SQL komutları ile yapılır. Örneğin: `SELECT`, `INSERT`, `UPDATE`, `DELETE`.

---

## 3. \*\*Veritabanı Yönetim Sistemi (DBMS) Nedir?\*\*

DBMS (Database Management System), veritabanlarının oluşturulması, yönetilmesi ve korunması için gereken araçları içeren yazılımdır. SQL Server, Oracle, MySQL gibi popüler DBMS örnekleridir.

---

## 4. \*\*Veritabanı Sunucusu Nedir?\*\*

Veritabanı sunucusu, veritabanı işlemlerini gerçekleştiren ve verilere erişimi sağlayan bir sunucudur. Genellikle veritabanı motoru (örneğin SQL Server) olarak adlandırılır ve istemcilerden gelen talepleri işler.

---

## 5. \*\*Client-Server Mimarisi Nasıl Çalışır?\*\*

- \*\*İstemci (Client):\*\* Kullanıcıların verilere erişim için kullandığı uygulamalar (örneğin web tarayıcı, mobil uygulama).

- \*\*Sunucu (Server):\*\* İstemcilerden gelen veritabanı isteklerini işleyen ve verileri depolayan sistem.

İstemci-sunucu modelinde veritabanı işlemlerinin tüm yükü sunucuda çalışır.

---

## 6. \*\*İlişkisel Veritabanı Nedir?\*\*

İlişkisel veritabanı, verileri tablolar halinde depolayarak ilişkileri belirten veritabanıdır. Tablolar arasında anahtar (primary key) ve yabancı anahtar (foreign key) ilişkileri kullanılır. Örneğin, "müşteriler" ve "siparişler" tablaları müşteri ID ile ilişkilendirilebilir.

---

## 7. \*\*Veritabanı Yönetim Sistemi Yazılımları Hangileridir?\*\*

Popüler DBMS örnekleri:

- Microsoft SQL Server

- Oracle Database

- MySQL

- PostgreSQL

- IBM Db2

---

# \*\*Sanallaştırma ve VMware ile İlgili Sorular\*\*

## 8. \*\*Sanallaştırma Ne Demektir?\*\*

Sanallaştırma, fiziksel kaynakları (disk, bellek, CPU) yazılımsal olarak parçalara bölerek birden fazla sanal makine çalıştırmayı ermögülren teknolojidir. Bu, kaynak verimliliğini artırır ve maliyetleri düşürür.

---

## 9. \*\*VMWare Ne Demektir? Nasıl Kurulur?\*\*

\*\*VMware\*\*, sanallaştırma çözümü oluşturan bir şirketin adıdır. VMware vSphere veya Workstation gibi araçlarıyla Windows/Linux tabanlı sanal makineler oluşturabilirsiniz. Kurulum adımları genellikle:

1. VMware software'ünü indirip kurulum dosyasını çalıştırın.

2. Yeni bir sanal makine oluşturun.

3. İşletim sistemi yükleyin.

---

## 10. \*\*VMWare Üzerine Sanal Bir İşletim Sistemi Nasıl Kurulur?\*\*

1. VMware'da yeni bir sanal makine oluşturun.

2. ISO dosyasını seçerek işletim sistemi (Windows, Linux) yükleyin.

3. Kurulum sürecini tamamlayın ve sanal makinenizi başlatın.

---

## 11. \*\*SQL Server Nasıl Kurulur?\*\*

SQL Server kurulumu genellikle:

1. Microsoft websiteinden SQL Server yükleyiciğini indirin.

2. Lisans koşullarını kabul edip kurulum adımlarını takip edin.

3. SQL Server hizmetlerini etkinleştirin.

---

## 12. \*\*SQL Server Sürümleri Nelerdir?\*\*

- \*\*Express:\*\* Ücretsiz, sınırlı kaynak kullanımı.

- \*\*Standard:\*\* Ortak kullanım için.

- \*\*Enterprise:\*\* Büyük ölçekli uygulamalar için.

- \*\*Developer:\*\* Geliştirme amaçlı (test için kullanılabilir).

---

## 13. \*\*SQL Server Management Studio (SSMS) Nedir? Nasıl Kurulur?\*\*

\*\*SSMS\*\*, SQL Server'ı yönetmek için kullanılan bir araçtır. Kurulumu:

1. Microsoft websiteinden SSMS'i indirin.

2. Yükleyiciyi çalıştırıp kurulum adımlarını tamamlayın.

---

## 14. \*\*SQL Server Hizmetleri Hangileridir?\*\*

- \*\*SQL Server (MSSQLSERVER):\*\* Veritabanı motoru.

- \*\*SQL Server Agent:\*\* Zamanlanmış görevler için.

- \*\*SQL Server Browser:\*\* Port numaralarını yönetir.

- \*\*SQL Server Reporting Services:\*\* Raporlama için.

---

## 15. \*\*Collation Kavramı Ne Demektir?\*\*

Collation, veritabanındaki metinsel verilerin karşılaştırılmasında kullanılan karakter kümesidir. Örneğin, `Latin1\_General\_CI\_AS` collation’ı büyük/küçük harf duyarlılığı yoktur.

---

# \*\*SQL Server Management Studio ve Komutlar\*\*

## 16. \*\*Management Studio Kullanımı\*\*

SSMS ile veritabanlarınıza bağlanabilir, sorgu çalıştırabilir, tablo oluşturabilir ve verileri yönetebilirsiniz. Bağlanma adımları:

1. SSMS’i açın.

2. \*\*Yeni Bağlantı\*\* > \*\*Microsoft SQL Server\*\* sekmesini seçin.

3. Sunucu adını, kimlik bilgilerini girip \*\*Bağlan\*\* butonuna tıklayın.

---

## 17. \*\*SQL Server Authentication Türleri\*\*

- \*\*Windows Authentication:\*\* Windows kullanıcı hesabını kullanarak oturum açma.

- \*\*SQL Server Authentication:\*\* SQL Server’da oluşturulan kullanıcı hesabını kullanarak oturum açma.

---

## 18. \*\*SQL Server Üzerinde Tablo Oluşturma\*\*

```sql

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name NVARCHAR(50),

Email NVARCHAR(100)

);

```

---

## 19. \*\*Management Studio Üzerinden Tabloya Kayıt Girme\*\*

1. Tabloya sağ tıklayıp \*\*Yeni Kayıt Ekle\*\* seçeneğini bulun.

2. Alanlara değerleri girip \*\*Tamam\*\* butonuna tıklayın.

---

## 20. \*\*SQL Server’ın Geri Planında Sadece TSQL Komutları Çalıştırıyor Olması\*\*

Evet, SSMS gibi araçlar arka planda T-SQL komutlarını çalıştırır. Örneğin, bir sorgu yazdığınızda, bu komut SQL Server’a gönderilir ve sonuç döndürülür.

---

# \*\*SQL Komutları ve Veri Manipülasyonu\*\*

## 21. \*\*SQL Komutları Temel Anlamda Nelerdir?\*\*

- \*\*DML (Data Manipulation Language):\*\* `SELECT`, `INSERT`, `UPDATE`, `DELETE`.

- \*\*DDL (Data Definition Language):\*\* `CREATE`, `ALTER`, `DROP`.

- \*\*DCL (Data Control Language):\*\* `GRANT`, `REVOKE`.

---

## 22. \*\*Data Manipülasyon ve Veritabanı Manipülasyon Komutları Ne Demektir?\*\*

- \*\*Data Manipülasyon:\*\* Verileri değiştirmek (INSERT, UPDATE, DELETE).

- \*\*Veritabanı Manipülasyon:\*\* Veritabanı yapısını değiştirmek (CREATE TABLE, ALTER TABLE).

---

## 23. \*\*TRUNCATE ile DELETE Arasında Ne Fark Var?\*\*

- \*\*DELETE:\*\* Satırları tek tek siler, geri alınamaz (ROLLBACK ile iptal edilebilir).

- \*\*TRUNCATE:\*\* Tablodaki tüm verileri anında siler, daha hızlıdır ancak geri alınamaz.

---

## 24. \*\*SELECT Komutu Ne İşe Yarar ve Nasıl Kullanılır?\*\*

```sql

SELECT CustomerID, Name FROM Customers WHERE Email = 'example@example.com';

```

---

## 25. \*\*INSERT Komutu Ne İşe Yarar ve Nasıl Kullanılır?\*\*

```sql

INSERT INTO Customers (CustomerID, Name, Email) VALUES (1, 'Ahmet', 'ahmet@example.com');

```

---

## 26. \*\*UPDATE Komutu Ne İşe Yarar ve Nasıl Kullanılır?\*\*

```sql

UPDATE Customers SET Email = 'newemail@example.com' WHERE CustomerID = 1;

```

---

## 27. \*\*DELETE Komutu Ne İşe Yarar ve Nasıl Kullanılır?\*\*

```sql

DELETE FROM Customers WHERE CustomerID = 1;

```

---

## 28. \*\*WHERE Şartı ile Sorgu Nasıl Filtrelenir?\*\*

```sql

SELECT \* FROM Products WHERE Price > 100;

```

---

## 29. \*\*=, >, <, >=, <=, LIKE, BETWEEN Komutları Nasıl Kullanılır?\*\*

- `=` Eşittir

- `>` Büyüktür

- `<` Küçüktür

- `LIKE` Benzerlik kontrolü (`%` wildcards ile)

- `BETWEEN` Aralık filtreleme

---

## 30. \*\*WHERE Şartında AND ve OR Kullanımı?\*\*

```sql

SELECT \* FROM Employees WHERE Department = 'IT' AND Salary > 5000;

```

---

## 31. \*\*DISTINCT Komutu Ne İşe Yarar? Nasıl Kullanılır?\*\*

```sql

SELECT DISTINCT City FROM Customers;

```

---

## 32. \*\*TOP Komutu Ne İşe Yarar? Nasıl Kullanılır?\*\*

```sql

SELECT TOP 5 \* FROM Orders ORDER BY OrderDate DESC;

```

---

# \*\*Gelişmiş SQL İşlemleri\*\*

## 33. \*\*SQL Server’a Bir Database Nasıl Restore Edilir?\*\*

1. SSMS’de \*\*Yedekleme ve Kurtarma\*\* aracıyla restore işlemi yapılır.

2. `.bak` uzantılı yedek dosyasını seçip kurtarın.

---

## 34. \*\*Aggregate Function Ne Demektir? Hangi Çeşitleri Vardır?\*\*

Aggregate fonksiyonlar, gruplandırılmış verileri özette gösterir. Örneğin:

- `SUM()`: Toplam hesaplar.

- `AVG()`: Ortalama hesaplar.

- `COUNT()`: Sayı sayar.

---

## 35. \*\*Group By Ne Amaçla ve Nasıl Kullanılır?\*\*

```sql

SELECT Category, AVG(Price) AS AveragePrice FROM Products GROUP BY Category;

```

---

## 36. \*\*MIN, MAX, AVG, COUNT, COUNT DISTINCT Komutları Hangi Amaçla ve Nasıl Kullanılır?\*\*

- `MIN()`: En küçük değeri döndürür.

- `MAX()`: En büyük değeri döndürür.

- `AVG()`: Ortalama hesaplar.

- `COUNT()`: Satır sayısını sayar.

- `COUNT(DISTINCT)`: Tekrarlayan değerleri saymaz.

---

## 37. \*\*Excel’de Özet Tablo Nasıl Kullanılır?\*\*

Excel’de \*\*Power Query\*\* veya \*\*Özet Tablo\*\* araçlarıyla verileri filtreleyip özete çekebilirsiniz.

---

## 38. \*\*Excel’den Bir Veritabanına Nasıl Bağlanılır ve Nasıl Sorgu Çekilir?\*\*

1. Excel’de \*\*Veri\*\* > \*\*External Data\*\* > \*\*From SQL Server\*\* seçeneğini kullanın.

2. Sunucu adresini ve kimlik bilgilerini girin.

3. Sorgu çalıştırıp verileri çekebilirsiniz.

---

## 39. \*\*HAVING Komutu Hangi Amaçla ve Nasıl Kullanılır?\*\*

```sql

SELECT Category, AVG(Price) FROM Products GROUP BY Category HAVING AVG(Price) > 100;

```

---

## 40. \*\*Group By Kullanılan Sorguda Order By Nasıl Kullanılır?\*\*

```sql

SELECT Category, AVG(Price) FROM Products GROUP BY Category ORDER BY AVG(Price) DESC;

```

---

## 41. \*\*Sözel Olarak İfade Ettiğimiz Raporların Sorguları Aggregate Function ve Group By Kullanarak Nasıl Yazılır?\*\*

Örnek: "Her kategorideki ürün ortalaması 500’den yüksek olanları listeleyin."

```sql

SELECT Category, AVG(Price) AS AvgPrice

FROM Products

GROUP BY Category

HAVING AVG(Price) > 500;

```

---

# \*\*SQL Server Veri Tipleri ve İlişkisel Veritabanı Kavramları\*\*

---

## \*\*1. SQL Server’daki Veri Tipi Kavramı Nedir?\*\*

Veri tipi, bir sütun veya değişkende depolanan verilerin türünü ve boyutunu tanımlayan bir özelliklidir. Örneğin, `INT` tam sayı değerleri için, `VARCHAR` metin için kullanılır. Veri tipleri, veri bütünlüğünü sağlama, bellek kullanımını optimize etme ve veritabanı performansını iyileştirme amaçlıdır.

---

## \*\*2. Sayısal Veri Tipleri Nelerdir?\*\*

- \*\*INT:\*\* 4 baytlık tam sayı (-2,147,483,648 ile 2,147,483,647 arası).

- \*\*SMALLINT:\*\* 2 baytlık tam sayı (-32,768 ile 32,767 arası).

- \*\*BIGINT:\*\* 8 baytlık tam sayı (-9,223,372,036,854,775,808 ile 9,223,372,036,854,775,807 arası).

- \*\*TINYINT:\*\* 1 baytlık tam sayı (0 ile 255 arası).

- \*\*FLOAT:\*\* Ondalıklı sayılarda yaklaşık değerler için (4 veya 8 bayt).

- \*\*DECIMAL:\*\* Ondalıklı sayılarda kesin değerler için (user-defined precisión ve scale).

- \*\*MONEY:\*\* Para birimleri için (-922,337,203,685,477.5808 ile 922,337,203,685,477.5807 arası).

---

## \*\*3. Integer, SmallInt, BigInt, TinyInt Arasındaki Farklar\*\*

| \*\*Veri Tipi\*\* | \*\*Boyut\*\* | \*\*Değer Aralığı\*\* |

|---------------|-----------|---------------------------------------|

| TINYINT | 1 bayt | 0 – 255 |

| SMALLINT | 2 bayt | -32,768 – 32,767 |

| INT | 4 bayt | -2,147,483,648 – 2,147,483,647 |

| BIGINT | 8 bayt | -9,223,372,036,854,775,808 – 9,223,372,036,854,775,807 |

Kullanım Alanı: Daha küçük aralıklarda veri saklamak için TINYINT veya SMALLINT tercih edilmelidir (bellek tasarrufu).

---

## \*\*4. Float, Decimal, Money Veri Tipleri Arasındaki Farklar\*\*

- \*\*FLOAT:\*\* Yaklaşık değerler için (örneğin bilimsel hesaplamalar).

- \*\*DECIMAL:\*\* Kesin değerler için (para hesaplamaları).

- \*\*MONEY:\*\* Para birimleri için optimize edilmiş veri tipi.

Örnek:

```sql

DECIMAL(10,2) -- 10 hane, 2 ondalık hane (99999999.99)

```

---

## \*\*5. Varchar, Char, Nchar, Nvarchar Arasındaki Farklar\*\*

| \*\*Veri Tipi\*\* | \*\*Unicode\*\* | \*\*Değişken Uzunluk\*\* | \*\*Max Boyut\*\* |

|-----------------|-------------|----------------------|---------------------|

| CHAR | Hayır | Hayır | 255 karakter |

| VARCHAR | Hayır | Evet | 8.000 karakter |

| NCHAR | Evet | Hayır | 4.000 karakter |

| NVARCHAR | Evet | Evet | 4.000 karakter |

---

## \*\*6. Hangi Veri Tipi Hafızada Ne Kadar Yer Tutar?\*\*

- \*\*INT:\*\* 4 bayt

- \*\*VARCHAR(n):\*\* n bayt (n ≤ 8.000)

- \*\*DECIMAL(p,s):\*\* 5–17 bayt (p: toplam hane, s: ondalık hane)

- \*\*DATE:\*\* 3 bayt

- \*\*DATETIME:\*\* 8 bayt

---

## \*\*7. Veri Tipi Seçimi İçin Kriterler\*\*

1. \*\*Veri Türü:\*\* Sayı, metin, tarih gibi.

2. \*\*Derece:\*\* Tam sayı mı, ondalıklı mı?

3. \*\*Uzunluk:\*\* Metin uzunluğu belirsiz mi?

4. \*\*Performans:\*\* Daha az bayt tüketen veri tipleri daha hızlıdır.

5. \*\*Bütünlük:\*\* Para hesaplamaları için DECIMAL tercih edilmelidir.

---

## \*\*8. Date, Time ve Datetime Arasındaki Farklar\*\*

| \*\*Veri Tipi\*\* | \*\*Depolama\*\* | \*\*Precision\*\* |

|-----------------|----------------------------|---------------------|

| DATE | Yılıl/tarih bilgisi | 1 gün |

| TIME | Saat/dakika/saniye | 100 nanosaniye |

| DATETIME | Tarih + saat | 3 milisaniye |

---

# \*\*İlişkisel Veritabanı ve RDBMS\*\*

---

## \*\*9. İlişkisel Veritabanı (RDBMS) Nedir?\*\*

İlişkisel veritabanı, verileri tablolar halinde depolayarak ilişkileri anahtar (key) yapısına dayanan bir veritabanıdır. Örneğin, "müşteriler" ve "siparişler" tablaları müşteri ID ile ilişkilendirilebilir.

---

## \*\*10. RDBMS Sistemine Sahip Bir Yapı Nasıl Çalışır? (Fatura Örneği)\*\*

- \*\*Müşteriler Tablosu:\*\* MüşteriID (PK), Ad, Soyad, Email.

- \*\*Siparişler Tablosu:\*\* SiparişID (PK), MüşteriID (FK), ÜrünAdı, Tutar.

- \*\*İlişki:\*\* Siparişler.MüşteriID → Müşteriler.MüşteriID.

Bu ilişkiyle, bir faturalar için müşteri bilgileri tek bir yerde saklanır ve siparişlerle ilişkilendirilir.

---

## \*\*11. E-Ticaret Sistemi Temel olarak Nasıl Çalışır?\*\*

- \*\*Ürün Tablosu:\*\* ÜrünID (PK), ÜrünAdı, Fiyat.

- \*\*Müşteriler Tablosu:\*\* MüşteriID (PK), Ad, Email.

- \*\*Siparişler Tablosu:\*\* SiparişID (PK), MüşteriID (FK), ÜrünID (FK), Adet.

Müşteriler sipariş verdiğinde, sipariş detayları bu tablolara kaydedilir.

---

## \*\*12. RDMS’te Veritabanı Mimarisi Nasıl Oluşturulur?\*\*

1. \*\*Analiz:\*\* Sistem gereksinimleri belirlenir.

2. \*\*Normalleştirme:\*\* Veri tekrarlarını önlemek için normalizasyon yapılır.

3. \*\*Tablo Tasarımı:\*\* Anahtarlar ve ilişkiler tanımlanır.

4. \*\*Index Oluşturma:\*\* Sorgu hızını artırmak için indeksler eklenir.

---

# \*\*JOIN İşlemleri ve ALIAS\*\*

---

## \*\*13. Birden Fazla Tablo Nasıl Birleştirilir?\*\*

`JOIN` komutuyla tablolar birleştirilir. Örneğin:

```sql

SELECT Customers.Name, Orders.OrderDate

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

```

---

## \*\*14. JOIN Kavramı Nedir?\*\*

`JOIN`, iki veya daha fazla tabloyu belirli bir koşula dayanarak birleştiren işlemdir.

---

## \*\*15. INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN Arasındaki Farklar\*\*

| \*\*JOIN Türü\*\* | \*\*Sonuç\*\* |

|-----------------|------------------------------------|

| INNER JOIN | Her iki tabloda da eşleşen satırlar |

| LEFT JOIN | Sol tablonun tüm satırları + Sağdaki eşleşenler |

| RIGHT JOIN | Sağ tablonun tüm satırları + Soldaki eşleşenler |

| FULL JOIN | Hem sol hem de sağ tablonun tüm satırları |

---

## \*\*16. Master-Detail Tabloları Nasıl Bağlanır?\*\*

Örnek:

```sql

SELECT Orders.OrderID, OrderDetails.ProductName, Quantity

FROM Orders

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID;

```

---

## \*\*17. ALIAS Nedir?\*\*

`ALIAS`, tablo veya sütun için geçici bir ad tanımlamaya yarar. Örneğin:

```sql

SELECT c.Name AS CustomerName FROM Customers AS c;

```

---

## \*\*18. JOIN Komutunda ALIAS Kullanımı\*\*

```sql

SELECT c.Name, o.OrderDate

FROM Customers AS c

JOIN Orders AS o ON c.CustomerID = o.CustomerID;

```

---

## \*\*19. JOIN Komutunda GROUP BY Kullanımı\*\*

```sql

SELECT c.City, COUNT(o.OrderID) AS TotalOrders

FROM Customers AS c

JOIN Orders AS o ON c.CustomerID = o.CustomerID

GROUP BY c.City;

```

---

# \*\*SUBQUERY ve String Fonksiyonları\*\*

---

## \*\*20. SUBQUERY Kavramı Nedir?\*\*

`SUBQUERY`, başka bir sorgu içinde çalıştırılan alt sorgudur. Örneğin:

```sql

SELECT \* FROM Products WHERE Price > (SELECT AVG(Price) FROM Products);

```

---

## \*\*21. SUBQUERY Kullanımı\*\*

```sql

-- Ürünlerin ortalamadan yüksek fiyatlılarını getirir

SELECT ProductName, Price

FROM Products

WHERE Price > (SELECT AVG(Price) FROM Products);

```

---

## \*\*22. JOIN ve SUBQUERY Arasındaki Avantaj/Dezavantajlar\*\*

- \*\*JOIN:\*\* Verileri birleştirmek için daha hızlıdır.

- \*\*SUBQUERY:\*\* Karmaşık koşullar için daha esnektir ancak performansı JOIN’den düşükdür.

---

## \*\*23. Nerede JOIN, Nerede SUBQUERY Kullanılır?\*\*

- \*\*JOIN:\*\* İki tabloyu doğrudan birleştirmek için.

- \*\*SUBQUERY:\*\* Alt sorguların sonucu üzerinde işlem yapmak için.

---

## \*\*24. String Fonksiyonları\*\*

| \*\*Fonksiyon\*\* | \*\*Amaç\*\* |

|---------------|-----------------------------------|

| ASCII | Karakterin ASCII kodunu döndürür |

| CHAR | ASCII kodundan karakter döndürür |

| CONCAT | Metinleri birleştirir |

| LEN | Metnin uzunluğunu verir |

| LOWER | Metni küçük harfe çevirir |

| UPPER | Metni büyük harfe çevirir |

| REPLACE | Metinde karakter değiştirir |

| TRIM | Metnin baş ve sondaki boşlukları siler |

---