

B Ö L Ü M

5

SQL'e Giriş: Veri Tipleri ve Veri Tanımlama



5.1 SQL Nedir?

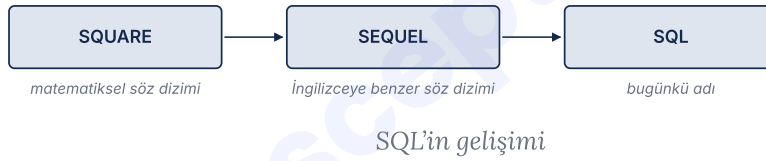
TANIM

SQL (Structured Query Language – Yapısal Sorgulama Dili), veri tabanları üzerinde karmaşık sorgular yapmak için tasarlanmış bir dildir.

SQL ile yalnızca veri tabanı üzerinde işlem yapılabilir: veri tabanına kayıt eklenebilir, var olan kayıtlar değiştirilebilir, silinebilir ve bu kayıtlardan listeler oluşturulabilir.

Tarihçe

Veri tabanı yaklaşımı ile birlikte bir veri sorgulama diline ihtiyaç duyulmuştur. İlk olarak matematiksel bir söz dizimine sahip **SQUARE** adlı dil geliştirilmiştir. Geniş kullanıcı kitleleri tarafından kolay kullanılabilmesi için bu dilden vazgeçilerek İngilizceye benzer söz dizimine sahip **SEQUEL** (Structured English Query Language) oluşturulmuş, bu dil daha sonra İngilizce söylenişine paralel olarak **SQL** adını almıştır.



SQL dilini kullanan veri tabanı yönetim sistemleri: Microsoft SQL Server (MSSQL), MySQL, PostgreSQL, Oracle, Firebird.

5.2 SQL Komut Grupları: DDL, DML, DCL

SQL komutları kullanım amaçlarına göre üç gruba ayrılır. Bu ayrım önemlidir: CREATE, ALTER, DROP gibi komutlar veri tabanının **mimarisiyle** ilgiliyken; SELECT, UPDATE, DELETE gibi komutlar tablonun **içindeki veriyle** ilgilidir.

DDL

DATA DEFINITION LANGUAGE · VERİ TANIMLAMA DİLİ

Veri tabanının şemasını, yapısını ve tablolarını oluşturmak, değiştirmek veya silmek için kullanılır. Doğrudan veri tabanı mimarisiyle ilgilenir.

CREATE	Veri tabanı, tablo, view veya indeks gibi yeni bir nesne oluşturur.
ALTER	Mevcut bir tablonun yapısını değiştirir (sütun ekleme, sütun silme vb.).
DROP	Veri tabanını veya tabloyu şemasıyla birlikte tamamen siler.
TRUNCATE	Tablonun yapısını bozmadan içindeki tüm verileri tek hamlede boşaltır.
RENAME	Mevcut bir tablonun veya nesnenin adını değiştirir.

DML

DATA MANIPULATION LANGUAGE · VERİ İŞLEME DİLİ

Tabloların içindeki verileri sorgulamak, eklemek, güncellemek ve silmek için kullanılır. Tablo yapısına dokunmaz, yalnızca satırlarla çalışır.

SELECT	Tablodaki verileri listeler, sorgular.
INSERT	Tabloya yeni bir satır ekler.
UPDATE	Tablodaki mevcut veriyi günceller.
DELETE	Tablodan belirli bir şarta uyan satırları siler.

DCL

DATA CONTROL LANGUAGE · VERİ KONTROL DİLİ

Veri tabanının güvenliğini, kullanıcı yetkilerini ve erişim izinlerini yönetmek için kullanılır.

GRANT	Bir kullanıcıya belirli nesneler üzerinde işlem yapma yetkisi verir.
REVOKE	Kullanıcıya daha önce verilmiş yetkileri geri alır.
DENY	Kullanıcının bir işlemi yapmasını açıkça yasaklar.

HATIRLATMA

VIEW oluşturma, ALTER TABLE, TRUNCATE ve DROP birer **DDL** işlemidir. SELECT, INSERT, UPDATE, DELETE ve INNER JOIN içeren sorgular birer **DML** işlemidir. Kullanıcı oluşturma, değiştirme ve silme komutları (CREATE USER, ALTER USER, DROP USER) veri işleme komutu değildir; yetkilendirme (DCL) tarafında ele alınır.

5.3 Kurulum: SQL Server Express ve SSMS

Önce veri tabanı motoru (**SQL Server**), ardından bu motoru yönetmek için kullanılan arayüz (**SQL Server Management Studio – SSMS**) kurulur. SQL Server yalnızca arka planda çalışan bir servistir; tabloları görmek ve sorgu yazmak için SSMS gereklidir.

1 Hazırlık ve indirme.

Microsoft SQL Server indirme sayfasında **Express** seçeneği bulunur ve “Download now” ile indirilir. İnen SQL2022-SSEI-Expr.exe dosyası yönetici olarak çalıştırılır.

2 SQL Server Express kurulumu.

Açılan pencerede Basic, Custom ve Download Media seçenekleri vardır. **Basic** en hızlı yoldur ve varsayılan ayarları kullanır. Lisans koşulları “Accept” ile kabul edilir, yükleme konumu değiştirilmeden “Install” denir.

3 Bağlantı bilgileri.

Kurulum bitince bağlantı bilgileri ekranı gelir. **Instance Name** genellikle SQLEXPRESS olur; mevcut Windows kullanıcısı yönetici (SQL Administrator) olarak eklenir. Bu ekran kapatılmadan **Install SSMS** düğmesine tıklanır.

4 SSMS kurulumu.

Açılan web sayfasından SSMS indirilir, SSMS-Setup-ENU.exe çalıştırılır ve “Install” denir. Kurulum sonunda bilgisayarın yeniden başlatılması istenebilir.

5 İlk bağlantı.

SSMS açılır. “Connect to Server” penceresinde **Server type:** Database Engine, **Server name:** .\SQLEXPRESS (veya bilgisayar adı), **Authentication:** Windows Authentication seçilir ve “Connect” denir. Solda **Object Explorer** paneli görünüyorsa kurulum başarıyla tamamlanmıştır.

NOT

Kurulum sırasında hata alınırsa nedeni genellikle eksik Windows güncellemeleri veya bekleyen bir sistem yeniden başlatmasıdır.

5.4 Veri Tipleri

Tablo oluşturulurken her sütun için bir veri tipi seçilir. Doğru tip hem gereksiz bellek kullanımını önler hem de yanlış veri girilmesini engeller. Aşağıdaki tablolar **SQL Server** veri tiplerini göstermektedir.



Veri tipi seçmek, içine konacak şeye uygun boyda kap seçmek gibidir

Tam Sayı Tipleri

Tip	Boyut	Değer aralığı	Not
bigint	8 bayt	-9.223.372.036.854.775.808 ... 9.223.372.036.854.775.807	Çok büyük tam sayılar
int	4 bayt	-2.147.483.648 ... 2.147.483.647	En yaygın kullanılan tam sayı tipi
smallint	2 bayt	-32.768 ... 32.767	Daha ekonomik seçenek
tinyint	1 bayt	0 ... 255	Küçük pozitif sayılar: durum kodu, sıra numarası
bit	1 bit	0 veya 1	Evet/Hayır, Açık/Kapalı gibi ikili durumlar

Ondalıklı ve Parasal Tipler

Tip	Açıklama
decimal / numeric	Toplam ve ondalık basamak sayısı tanımlanabilir. Örneğin decimal(18,0) : toplam 18 basamak, 0 ondalık basamak. Boyutu tanıma göre değişir.
float	Çok büyük veya çok küçük kesirli sayılar için yaklaşık değerli tiptir. Bilimsel hesaplamalarda tercih edilir.
real	float'ın küçük sürümüdür. 4 bayt yer kaplar, yaklaşık 7 basamak hassasiyet sağlar.
money	8 bayt. Parasal değerler için; ondalık kısım 4 basamaktır.
smallmoney	4 bayt. Yaklaşık -214,748,3648 ile 214,748,3647 arasındaki parasal değerler; ondalık kısım 4 basamaktır.

Tarih ve Zaman Tipleri

Tip	Açıklama
date	Yalnızca tarih. 3 bayt. YYYY-AA-GG biçiminde işlenir (örnek: 2024-01-15).
time	Yalnızca zaman (saat, dakika, saniye, salise).
datetime	Tarih ve saat birlikte. 8 bayt. 1753-01-01 ile 9999-12-31 arasını destekler.
datetime2	Geliştirilmiş tarih-zaman tipi. 0001-01-01 ile 9999-12-31 arası; 7 basamaklı salise hassasiyeti.
smalldatetime	4 bayt. Dakika düzeyinde hassasiyet.
datetimeoffset	Zaman dilimi farkını da içeren tarih-zaman tipi.

Karakter ve Metin Tipleri

Tip	Uzunluk	Unicode	Açıklama
char(n)	Sabit	Hayır	Her zaman n karakterlik yer kaplar; daha az karakter girilse bile. En çok 8000 karakter.
varchar(n)	Değişken	Hayır	Girilen verinin boyutu kadar yer kaplar. En çok 8000 karakter.
nchar(n)	Sabit	Evet	Unicode karakterleri sabit uzunlukta saklar. En çok 4000 karakter.
nvarchar(n)	Değişken	Evet	Unicode destekli, değişken uzunluklu en kullanışlı metin tipi. En çok 4000 karakter.

UNICODE

Unicode, dünyadaki tüm dillerin karakterlerini tek bir standartta temsil eden evrensel karakter kodlama sistemidir. Türkçe karakterlerin (ı, ğ, ş, İ) farklı sistemler arasında bozulmadan taşınması için metin alanlarında `nvarchar` tercih edilir.

Diğer Tipler

Tip	Açıklama
<code>binary(n)</code> , <code>varbinary(n)</code>	Resim, ses, şifreli veri gibi ikilik veriler. <code>varbinary(MAX)</code> 2 GB'a kadar içerik saklar.
<code>uniqueidentifier</code>	Benzersiz tanımlayıcı (GUID). 16 bayt; dağıtık sistemlerde birincil anahtar olarak kullanılır.
<code>xml</code>	XML biçiminde yapılandırılmış veriler.
<code>text</code> , <code>ntext</code> , <code>image</code>	Eski tiplerdir. Yeni projelerde yerlerine <code>nvarchar(MAX)</code> ve <code>varbinary(MAX)</code> kullanılır.

EK BİLGİ: MYSQL VERİ TÜRLERİ

Derste MySQL veri türlerine de değinilmiştir. Adlar büyük ölçüde aynıdır; başlıca farklar şunlardır:

MySQL tipi	Özelliği
TINYINT	İşaretli -128 ... 127, işaretli (UNSIGNED) 0 ... 255. SQL Server'da tinyint yalnızca 0 ... 255'tir.
SMALLINT / MEDIUMINT / INT / BIGINT	İşaretli ve işaretli tanımlanabilir. INT: -2.147.483.648 ... 2.147.483.647 (işaretsiz 0 ... 4.294.967.295).
FLOAT / DOUBLE / DECIMAL	Kesirli sayılar. DECIMAL'de toplam en çok 65, ondalık kısımda en çok 30 basamak.
DATE / DATETIME / TIMESTAMP	DATE: 1000-01-01 ... 9999-12-31. TIMESTAMP: 1 Ocak 1970 ... 19 Ocak 2038.
CHAR(n) / VARCHAR(n) / TEXT / MEDIUMTEXT	TEXT en çok 65.535, MEDIUMTEXT en çok 16.777.215 karakter.
BOOL	0 veya 1 değerini alır.

5.5 Veri Tabanı ve Tablo Oluşturma: CREATE

Veri tabanı `CREATE DATABASE`, tablo ise `CREATE TABLE` komutuyla oluşturulur. Arayüz kullanmadan, yalnızca kod yazarak **pansiyon** adında bir veri tabanı ve içinde nöbet listesi tablosu şöyle oluşturulur:

```
CREATE DATABASE pansiyon
GO
USE pansiyon
GO
CREATE TABLE nobetlistesi
(
    id      INT PRIMARY KEY IDENTITY NOT NULL,
    tarih   VARCHAR(50),
    nob1    VARCHAR(50),
    nob2    VARCHAR(50),
    nob3    VARCHAR(50),
    nob4    VARCHAR(50)
)
```

CREATE DATABASE pansiyon adında bir veri tabanı oluşturur.

GO Kendinden önceki komutların çalıştırılıp bitirilmesini sağlar; komut gruplarını birbirinden ayırır.

USE Sonraki komutların hangi veri tabanında çalışacağını belirtir. Yazılmazsa tablo yanlış veri tabanında oluşabilir.

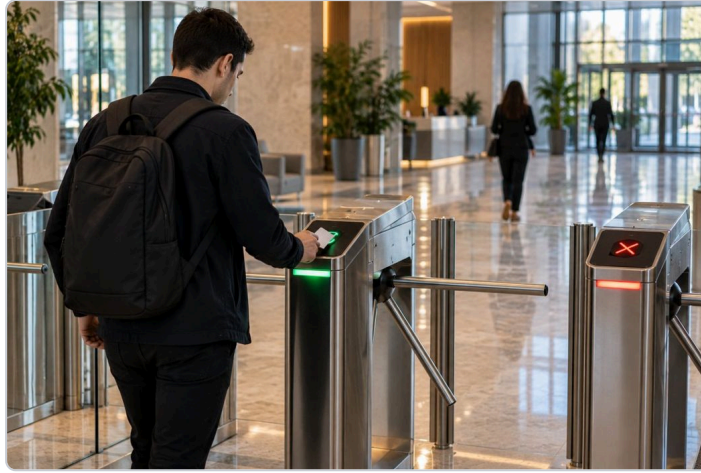
IDENTITY Sütunun değerini her yeni kayıta kendiliğinden artırır. `IDENTITY(1,1)` : 1'den başla, 1'er artır.

nobetlistesi

id	PK	tarih	nob1	nob2	nob3	nob4
10		06/03/2024	Alt kat	bahçe	2. kat	salon
11						

5.6 Kısıtlamalar (Constraints)

Kısıtlamalar, tabloya girilebilecek verilere kural koyarak veri bütünlüğünü korur. Kurala uymayan veri girildiğinde veri tabanı yönetim sistemi işlemi reddeder.



Kısıtlama bir turnike gibidir: kurala uyan geçer, uymayan geçemez

Kısıtlama	Görevi	Örnek
PRIMARY KEY	Satırı benzersiz tanımlar. NULL olamaz, tekrar edemez. Her tabloda bir tane bulunur (birden çok sütundan oluşabilir).	OgrId INT PRIMARY KEY
FOREIGN KEY	Sütunu başka bir tablonun birincil anahtarına bağlar; o tabloda bulunmayan bir değer girilmesini engeller.	FOREIGN KEY (BolumId) REFERENCES Bolum(BolumId)
UNIQUE	Sütundaki değerlerin tekrar etmesini engeller (benzersiz anahtar, UK). Bir tabloda birden çok olabilir.	OgrNo VARCHAR(20) UNIQUE
NOT NULL	Sütunun boş bırakılmasını engeller.	Ad VARCHAR(50) NOT NULL
CHECK	Değerin belirli bir koşulu sağlamasını zorunlu kılar.	CHECK (Sinif BETWEEN 1 AND 4)
DEFAULT	Değer girilmediğinde kullanılacak varsayılan değeri belirler.	Kredi INT DEFAULT 3

NULL

NULL, “değer yok / bilinmiyor” demektir; 0 ya da boş metin ile aynı şey değildir. Birincil anahtar hiçbir zaman NULL olamaz.

Örnek: Fakülte Veri Tabanı

Fakülte veri tabanı dört tablodan oluşur: Bolum, Ogrenci, Dersler ve Dersi_Alanlar. Bolum tablosu diğerlerinden önce oluşturulmalıdır, çünkü Ogrenci ve Dersler tabloları ona yabancı anahtarla bağlıdır. Bu veri tabanı 6. ve 7. bölümlerdeki sorgularda kullanılacaktır.

```

CREATE DATABASE FakulteDB;
GO
USE FakulteDB;
GO

-- 1. Bolum tablosu (önce oluşturulur; diğer tablolar buna bağlıdır)
CREATE TABLE Bolum (
    BolumId    INT IDENTITY(1,1) PRIMARY KEY,
    BolumKod   VARCHAR(10) NOT NULL UNIQUE,
    BolumAdi   VARCHAR(100) NOT NULL
);

-- 2. Ogrenci tablosu
CREATE TABLE Ogrenci (
    OgrId      INT IDENTITY(1,1) PRIMARY KEY,
    OgrNo      VARCHAR(20) NOT NULL UNIQUE,
    Ad         VARCHAR(50) NOT NULL,
    Soyad      VARCHAR(50) NOT NULL,
    Ili        VARCHAR(50),
    BolumId    INT NOT NULL,
    Sinif      INT NOT NULL,
    FOREIGN KEY (BolumId) REFERENCES Bolum(BolumId)
);

-- 3. Dersler tablosu
CREATE TABLE Dersler (
    DersId      INT IDENTITY(1,1) PRIMARY KEY,
    DersKodu    VARCHAR(10) NOT NULL UNIQUE,
    DersAdi     VARCHAR(100) NOT NULL,
    Kredi       INT,
    Teori       INT,
    Uyg         INT,
    BolumId     INT NOT NULL,
    FOREIGN KEY (BolumId) REFERENCES Bolum(BolumId)
);

-- 4. Dersi_Alanlar tablosu (N-M ilişkiyi kuran köprü tablo)
CREATE TABLE Dersi_Alanlar (
    DersId      INT NOT NULL,
    OgrId       INT NOT NULL,
    PRIMARY KEY (DersId, OgrId),
    FOREIGN KEY (DersId) REFERENCES Dersler(DersId),
    FOREIGN KEY (OgrId) REFERENCES Ogrenci(OgrId)
);

```

Tablo	PK	FK	UK
Ogrenci	OgrId	BolumId	OgrNo
Bolum	BolumId	—	BolumKod
Dersler	DersId	BolumId	DersKodu
Dersi_Alanlar	DersId + OgrId	DersId, OgrId	—

HATIRLATMA

PK: Primary Key / birincil anahtar. FK: Foreign Key / yabancı (ikincil) anahtar. UK: Unique Key / benzersiz anahtar. İki veya daha fazla sütundan oluşan birincil anahtara bileşik (kompozit) anahtar denir.

5.7 Tablo Yapısını Değiştirme: ALTER TABLE

ALTER TABLE, veri tabanında hâlihazırda var olan, içi veriyle dolu bir tablonun mimarisini (sütunlarını, veri tiplerini veya kısıtlamalarını) tabloyu silip yeniden oluşturmak zorunda kalmadan değiştirir. Canlı sistemdeki veriyi kaybetmeden tabloyu güncellemeyi sağladığı için en sık başvurulan komutlardan biridir.



ALTER TABLE: içinde yaşam süren binaya, yıkmadan yeni kat eklemek gibi

1. Tabloya Yeni Sütun Eklemek (ADD)

```
ALTER TABLE Ogrenci ADD burs_miktari INT;
```

-- Birden fazla sütun eklemek (SQL Server)

```
ALTER TABLE kullanıcı ADD e_posta VARCHAR(50), yas INT, yer VARCHAR(80);
```

Oğrenci tablosuna tam sayı tipinde burs_miktari adında yeni bir sütun eklenir. Eski kayıtların bu yeni alanı kendiliğinden **NULL** (boş) olarak başlar.

2. Tablodan Sütun Silmek (DROP COLUMN)

```
ALTER TABLE Urunsatis DROP COLUMN tarih;
```

Urunsatis tablosundaki tarih sütunu, içindeki tüm verilerle birlikte geri dönülemez şekilde silinir.

3. Sütunun Veri Tipini veya Boyutunu Değiştirmek

```
-- SQL Server:
ALTER TABLE Musteri ALTER COLUMN adi VARCHAR(50);

-- MySQL / Oracle:
ALTER TABLE Musteri MODIFY adi VARCHAR(50);
```

Musteri tablosundaki adi sütununun karakter sınırı 50 yapılır. Komutun yazımı veri tabanı yönetim sistemine göre değişir.

4. Kısıtlama veya Anahtar Ekleme (ADD CONSTRAINT)

```
ALTER TABLE Urunler
ADD CONSTRAINT FK_Kategori_Urun
FOREIGN KEY (ktgr_no) REFERENCES Kategori(ktgr_no);
```

Tablo oluşturulurken unutulmuş ya da sonradan ihtiyaç duyulan birincil anahtar, yabancı anahtar veya benzersizlik kuralı tabloya eklenir. Burada Urunler tablosundaki ktgr_no alanı, Kategori tablosunun ktgr_no alanına yabancı anahtar olarak bağlanmıştır.

ALTER TABLE İLE UPDATE FARKI

ALTER TABLE tablonun fiziksel sütun yapısını değiştirir (sütun ekler, siler). **UPDATE** ise tablonun yapısına dokunmadan, mevcut sütunların içindeki satır verilerini değiştirir. Bu iki komut sıkça karıştırılır.

5.8 DELETE, TRUNCATE ve DROP

Üç komut da “silme” işi yapar, ancak sildikleri şey farklıdır: **DELETE** tablodan seçilen satırları, **TRUNCATE** tablonun bütün içeriğini, **DROP** ise tablonun kendisini siler.



Solda DELETE: tek dosya çıkarılır. Ortada TRUNCATE: çekmeceler boşalır, dolap kalır. Sağda DROP: dolap da yoktur.

```
DELETE FROM Musteri WHERE musID = 3; -- yalnızca 3 numaralı müşteri
TRUNCATE TABLE Musteri;             -- tablo boş bir şablon kalır
DROP TABLE Urun;                     -- tablo tamamen silinir
```

Özellik	DELETE	TRUNCATE	DROP
Komut türü	DML (veri işleme)	DDL (veri tanımlama)	DDL (veri tanımlama)
Tablo şeması kalır mı?	Evet	Evet	Hayır, tamamen silinir
Şart (WHERE) alır mı?	Evet, satır seçilebilir	Hayır, tümü silinir	Hayır, her şey silinir
Hız	Yavaş (satır satır log tutar)	Çok hızlı	Çok hızlı
Sayacı (IDENTITY) sıfırlar mı?	Hayır	Evet (1'den başlar)	Tablo kalmadığı için geçerli değil
Geri alınabilir mi?	Evet (ROLLBACK ile)	Genellikle hayır	Hayır
Sonrasında tablo	Durur; kalan satırlarla	Durur; boş bir şablon olarak	Yoktur; sorgulanırsa "tablo bulunamadı" hatası alınır

Log (Günlük) Kavramı

Log, sistemde gerçekleşen her türlü işlemin veya hatanın tarih ve saat damgasıyla, kronolojik olarak kaydedildiği dosyadır. Bir uçağın kara kutusuna ya da bir binanın ziyaretçi defterine benzer. Veri tabanı açısından üç önemli görevi vardır:



Log, uçağın kara kutusu ya da ziyaretçi defteri gibi olup biten her şeyi kaydeder

**İşlem günlüğü
(transaction log)**

INSERT, UPDATE veya DELETE komutu önce işlem günlüğüne kaydedilir. İşlem sırasında elektrik kesilir ya da sunucu çökerse, veri tabanı yeniden açıldığında bu günlüğe bakar; yarım kalmış, onaylanmamış (COMMIT edilmemiş) işlemi kendiliğinden geri alır (ROLLBACK).

DELETE–TRUNCATE hız farkı

DELETE, bir milyon satır silerken günlüğe her satırı tek tek yazar; bu yüzden yavaş ama güvenlidir. TRUNCATE satır satır kayıt tutmaz, tablonun boşaltıldığını not eder ve geçer; bu yüzden çok hızlıdır.

**Denetim günlüğü
(audit log)**

“Maaş verisini kim değiştirdi?”, “Müşteri verilerini kim dışarı çıkardı?” gibi soruların cevabı denetim günlüklerinde aranır.

5.9 Veri Tabanını Taşıma ve Yedekleme: .sql, .mdf, .bak

**SQL script dosyası
(.sql)**

Tamamen metin tabanlıdır; içinde veri tabanını sıfırdan kuracak SQL komutları (CREATE TABLE, INSERT INTO, ALTER...) yer alır. Not Defteri ile bile açılıp okunabilir. Başka bir bilgisayarda çalıştırıldığında (Execute) komutlar yukarıdan aşağıya sırayla işler ve veri tabanını yeniden oluşturur. Boyutu çok küçüktür ve sürüm farklılıklarından en az etkilenen yöntemdir; ancak milyonlarca satırlık veri tabanlarında yavaştır.

MDF dosyası (.mdf)

SQL Server'ın **ana veri dosyasıdır** (Master Data File). Tablolar, indeksler, view'lar ve asıl satır verileri bu dosyada fiziksel olarak, ikili (binary) biçimde saklanır. SQL Server çalışırken dosya kilitlidir; taşımak için önce veri tabanı **Detach** edilir, .mdf ve yanındaki .ldf (log) dosyası alınır, hedef bilgisayarda **Attach** edilir. Sürüm bağımlılığı katıdır: üst sürümde çalışan bir .mdf alt sürüme bağlanamaz.

Yedek dosyası (.bak)

Veri tabanının belirli bir andaki durumunun sıkıştırılarak tek paket hâline getirilmiş **yedeğidir**. SSMS'te Tasks → Back Up ile üretilir; canlı sistemi durdurmaya gerek yoktur. Başka bir sunucuda **Restore** (geri yükleme) ile açılır. Veri kaybını önlemenin yoludur; burada da üst sürümde alınan yedek alt sürüme geri yüklenemez.



Yedekleme: veri tabanının bir kopyası ayrı bir ortamda saklanır

Özellik	.sql (script)	.mdf (ana veri dosyası)	.bak (yedek)
Dosya tipi	Düz metin (okunabilir)	İkili, canlı veri	Sıkıştırılmış yedek paketi
Sürüm esnekliği	Çok yüksek	Düşük (sürüme bağımlı)	Düşük (sürüme bağımlı)
Canlı sistemde taşıma	Doğrudan taşınır	Taşınamaz; önce Detach gerekir	Sistem çalışırken alınır ve taşınır
Büyük verilerde	Yavaş ve hantal	Çok hızlı (Attach ile anında)	En uygun ve güvenli yöntem
Kullanım amacı	Kod paylaşımı, şema taşıma	Etkin ve canlı çalışma	Güvenlik, arşiv ve yedekleme