

B Ö L Ü M

7

Uygulamalar

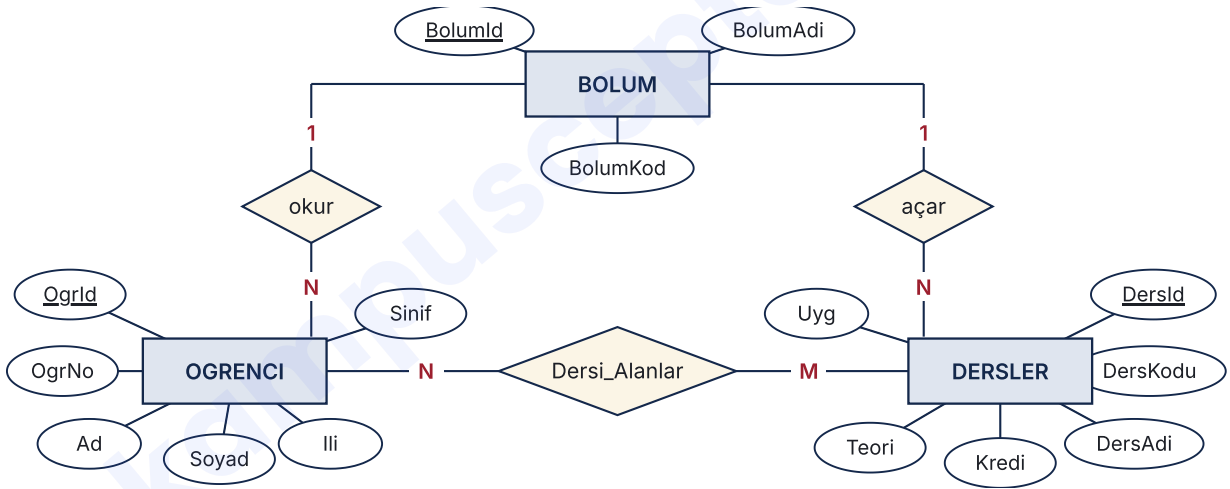


7.1 Uygulama 1: Fakülte Veri Tabanı

Fakülte veri tabanı dört tablodan oluşur: Bolum, Ogrenci, Dersler ve Dersi_Alanlar (CREATE kodları 5.6'da verilmiştir). Bu uygulamada önce tasarım incelenecek, sonra tablolar üzerinde sorular çözülecektir.

Tablolar Arasındaki İlişkiler

İlişki	Tür	Gerekçe
Bolum - Ogrenci	1-N	Bir bölümde birden fazla öğrenci olabilir; her öğrenci yalnızca bir bölüme bağlıdır (Ogrenci tablosunda BolumId bulunur).
Bolum - Dersler	1-N	Bir bölüm birçok ders açabilir; her ders yalnızca bir bölüme aittir (Dersler tablosunda BolumId bulunur).
Ogrenci - Dersler	N-M	Bir öğrenci birden fazla ders alabilir, bir ders birden fazla öğrenci tarafından alınabilir. İlişkiyi Dersi_Alanlar ara tablosu kurar; bu tablo iki tarafa da 1-N bağlıdır.



Fakülte veri tabanının E-R diyagramı. N-M ilişki, tabloya dönüştürülürken Dersi_Alanlar (DersId, OgrId) köprü tablosu olur.

Sorular

Aşağıdaki sorular bu tablolara göre çözülecektir:

Oğrenci								
OgrId	PK	OgrNo	Ad	Soyad	İli	BolumId	FK	Sınıf
1		4051	Ali	Ay	Van	1		1
2		3042	Veli	Demir	Muş	1		2
3		2033	Elif	Yıldız	Rize	2		3
4		2024	Can	Güneş	İzmir	5		3
5		4015	Ayşe	Çelik	Kilis	2		1
6		1026	Ali	Demir	Muş	1		4
7		2067	Elif	Can	Niğde	2		3
8		1088	Murat	Temiz	Ordu	1		4

Dersler								
DersId	PK	DersKodu	DersAdi	Kredi	Teori	Uyg	BolumId	FK
1		ABM105	Algoritma ve Programlama	5	3	2	1	
2		ABM106	Programlama Dilleri	5	3	2	1	
3		ABM313	İşletim Sistemleri	3	3	0	1	
4		ABM409	Yazılım Etiği	2	2	0	1	
5		INT205	Dinamik	3	3	0	4	
6		YMT211	Ayrık Yapılar	2	2	0	5	
7		YMT217	Programlama Dilleri	4	3	2	5	
8		YMT321	Ağ Sistemleri	3	3	0	5	

Bolum			
BolumId	PK	BolumKod	BolumAdi
1		ABM	Adli Bilişim Mühendisliği
2		EEM	Elektrik Elektronik Mühendisliği
3		ESM	Enerji Sistemleri Mühendisliği
4		INT	İnşaat Mühendisliği
5		YMT	Yazılım Mühendisliği

Dersi_Alanlar			
DersId	PK, FK	OgrId	PK, FK
1		1	
1		2	
3		6	
3		1	
4		8	
6		4	
7		4	

Soru 1

FAKÜLTE VT

`SELECT DersId, SUM(Kredi) / COUNT(Kredi) AS Kredi_Ort FROM Dersler GROUP BY DersId;` sorgusu ne yapar?

CEVAP

Dersleri DersId'ye göre gruplar ve her grup için kredi toplamını kredi sayısına böler. Her ders tabloda yalnızca bir satırda bulunduğu için her grup tek satırdan oluşur; sonuç $Kredi / 1$, yani her dersin **kendi kredi değeridir**: 5, 5, 3, 2, 3, 2, 4, 3.

Soru 2

FAKÜLTE VT

Dersler tablosunda BolumId'si 5 olan kayıtları silen SQL sorgusunu yazınız.

CEVAP

`DELETE FROM Dersler WHERE BolumId = 5;`

YMT bölümüne ait üç ders (YMT211, YMT217, YMT321) silinmek istenir. **Uygulamada dikkat:** Dersi_Alanlar tablosunda 6 ve 7 numaralı derslere ait kayıtlar bulunduğu için komut yabancı anahtar hatası verir. Önce `DELETE FROM Dersi_Alanlar WHERE DersId IN (6, 7, 8);` çalıştırılmalıdır.

Soru 3

FAKÜLTE VT

`UPDATE Dersler SET Kredi = 5 WHERE DersKodu LIKE 'ABM%';` komutu ne yapar?

CEVAP

Ders kodu "ABM" ile başlayan derslerin Kredi değerini 5 yapar. Dört kayıt etkilenir: ABM105 ve ABM106 zaten 5'tir (değer değişmez), ABM313 3'ten 5'e, ABM409 2'den 5'e çıkar. Aynı komut `LIKE 'YMT%'` ile yazılırsa YMT211, YMT217 ve YMT321 derslerinin kredisi 5 olur.

Soru 4

FAKÜLTE VT

`INSERT INTO Bolum (BolumId, BolumKod, BolumAdi) VALUES (6, 'SUB', 'Su Ürünleri');` ve `INSERT INTO Dersi_Alanlar (DersId, OgrId) VALUES (8, 8);` komutları ne yapar?

CEVAP

Birincisi Bolum tablosuna 6 numaralı, SUB kodlu “Su Ürünleri” bölümünü ekler. İkincisi 8 numaralı dersi (YMT321) 8 numaralı öğrencinin (Murat Temiz) aldığını kaydeder. BolumId sütunu IDENTITY olarak tanımlanmışsa değeri sistem verir; bu durumda alan listesinde BolumId yazılmaz.

Soru 5

FAKÜLTE VT

`INSERT INTO Dersler (DersId, DersKodu, DersAdi, Kredi, Teori, Uyg, BolumId) VALUES (9, 'BMD103', 'Bilişim', 5, 2, 3, 8);` komutu ne yapar?

CEVAP

Dersler tablosuna BMD103 kodlu, 5 kredilik (2 teori + 3 uygulama) “Bilişim” dersini eklemek ister. Ancak dersin bağlanacağı **8 numaralı bölüm Bolum tablosunda yoktur** (bölümler 1–5 arasındadır). BolumId yabancı anahtar olduğu için veri tabanı bu kaydı kabul etmez ve hata verir.

Soru 6

FAKÜLTE VT

`ALTER TABLE Bolum DROP COLUMN BolumAdi;` komutu ne yapar?

CEVAP

Bolum tablosunun **yapısını** değiştirir: BolumAdi sütununu ve içindeki tüm verileri kalıcı olarak siler. Komuttan sonra tablo yalnızca BolumId ve BolumKod sütunlarını içerir; satır sayısı değişmez.

7.2 Uygulama 2: “Gelecek Sepeti”

“Gelecek Sepeti” adında bir e-ticaret sitesinin üç tablosu vardır: Musteriler, Ürünler ve Siparişler. Amaç, bu verilerden işletme yönetimi için anlamlı sonuçlar çıkarmaktır.

```
CREATE TABLE Musteriler (  
    MusteriID INT PRIMARY KEY,  
    Ad VARCHAR(50),  
    Soyad VARCHAR(50),  
    Sehir VARCHAR(50)  
);  
CREATE TABLE Urunler (  
    UrunID INT PRIMARY KEY,  
    UrunAdi VARCHAR(100),  
    Kategori VARCHAR(50),  
    BirimFiyat DECIMAL(10,2)  
);  
CREATE TABLE Siparisler (  
    SiparisID INT PRIMARY KEY,  
    MusteriID INT,  
    UrunID INT,  
    Adet INT,  
    SiparisTarihi DATE,  
    FOREIGN KEY (MusteriID) REFERENCES Musteriler(MusteriID),  
    FOREIGN KEY (UrunID) REFERENCES Urunler(UrunID)  
);  
  
-- Veriler aynı sırayla eklenir, örneğin:  
INSERT INTO Musteriler VALUES (1, 'Ahmet', 'Yılmaz', 'İstanbul');  
INSERT INTO Urunler VALUES (101, 'Laptop', 'Teknoloji', 15000);  
INSERT INTO Siparisler VALUES (1, 1, 101, 1, '2023-10-01');
```

NEREDE ÇALIŞTIRILIR?

Program kurmadan denemek için **db-fiddle.com** kullanılabilir: sol üstten “MySQL 8.0” veya “PostgreSQL 15” seçilir, kodlar soldaki Schema SQL alanına yapıştırılır, sorgular sağ tarafa yazılır. Derste paylaşılan betikte müşteri tablosunun son sütunu *Bolum*, sipariş tablosu *Siparişler* adıyla geçer; burada *Sehir* ve *Siparisler* adları kullanılmıştır.

Tabloların içeriği aşağıdadır; dört soru da bu verilere göre çözülecektir.

Musteriler

MusteriID PK	Ad	Soyad	Sehir
1	Ahmet	Yilmaz	İstanbul
2	Ayşe	Demir	Ankara
3	Mehmet	Kaya	İzmir
4	Canan	Yıldız	Bursa

Urunler

UrunID PK	UrunAdi	Kategori	BirimFiyat
101	Laptop	Teknoloji	15000.00
102	Kitap: Sosyolojiye Giriş	Eğitim	150.00
103	Akıllı Telefon	Teknoloji	8000.00
104	Kahve Makinesi	Mutfak	2500.00

Siparisler

SiparisID PK	MusteriID FK	UrunID FK	Adet	SiparisTarihi
1	1	101	1	2023-10-01
2	2	102	3	2023-10-02
3	3	104	1	2023-10-03
4	1	103	1	2023-10-04
5	4	102	1	2023-10-05
6	2	101	1	2023-10-06

Soru 1

INNER JOIN

Hangi müşteri hangi ürünü aldı? Müşterinin adını, soyadını ve aldığı ürünün adını listeleyen sorguyu yazınız.

CEVAP

```
SELECT M.Ad, M.Soyad, U.UrunAdi
FROM Siparisler S
INNER JOIN Musteriler M ON S.MusteriID = M.MusteriID
INNER JOIN Urunler U ON S.UrunID = U.UrunID;
```

SONUÇ

Ad	Soyad	UrunAdi
Ahmet	Yılmaz	Laptop
Ayşe	Demir	Kitap: Sosyolojiye Giriş
Mehmet	Kaya	Kahve Makinesi
Ahmet	Yılmaz	Akıllı Telefon
Canan	Yıldız	Kitap: Sosyolojiye Giriş
Ayşe	Demir	Laptop

Siparisler tablosu, müşteri ile ürün arasındaki köprüdür; bu yüzden iki JOIN gerekir.

Soru 2

GROUP BY

Hangi kategoride kaç adet ürün satılmıştır? Kategorileri ve toplam satış adetlerini listeleyniz.

CEVAP

```
SELECT U.Kategori, SUM(S.Adet) AS ToplamAdet
FROM Siparisler S
INNER JOIN Urunler U ON S.UrunID = U.UrunID
GROUP BY U.Kategori;
```

SONUÇ

Kategori	ToplamAdet
Eğitim	4
Mutfak	1
Teknoloji	3

Eğitim: 3 + 1 = 4 kitap. Teknoloji: 2 laptop + 1 telefon = 3. Mutfak: 1 kahve makinesi.

Soru 3

HAVING

Toplam harcaması (Adet × BirimFiyat) 10.000 TL'den fazla olan müşterilerin adlarını ve toplam harcama tutarlarını listeleyiniz.

CEVAP

```
SELECT M.Ad, M.Soyad, SUM(S.Adet * U.BirimFiyat) AS ToplamHarcama
FROM Siparisler S
INNER JOIN Musteriler M ON S.MusteriID = M.MusteriID
INNER JOIN Urunler U ON S.UrunID = U.UrunID
GROUP BY M.MusteriID, M.Ad, M.Soyad
HAVING SUM(S.Adet * U.BirimFiyat) > 10000;
```

SONUÇ

Ad	Soyad	ToplamHarcama
Ahmet	Yılmaz	23000.00
Ayşe	Demir	15450.00

Önce JOIN ile tablolar bağlanır, sonra her müşterinin harcaması toplanır, en son HAVING ile gruplar filtrelenir. Ahmet: $15000 + 8000 = 23000$. Ayşe: $3 \times 150 + 15000 = 15450$. Mehmet (2500) ve Canan (150) elenir.

Soru 4

WHERE + JOIN

“Eğitim” kategorisindeki ürünleri satın alan müşterilerin listesini (ad ve soyad) getiren sorguyu yazınız.

CEVAP

```
SELECT DISTINCT M.Ad, M.Soyad
FROM Siparisler S
INNER JOIN Musteriler M ON S.MusteriID = M.MusteriID
INNER JOIN Urunler U ON S.UrunID = U.UrunID
WHERE U.Kategori = 'Eğitim';
```

SONUÇ

Ad	Soyad
Ayşe	Demir
Canan	Yıldız

DISTINCT, aynı müşteri bu kategoriden birden çok sipariş verdiğinde adının tekrar etmesini önler.

7.3 Uygulama 3: SATIŞ Veri Tabanının SQL'de Kurulması

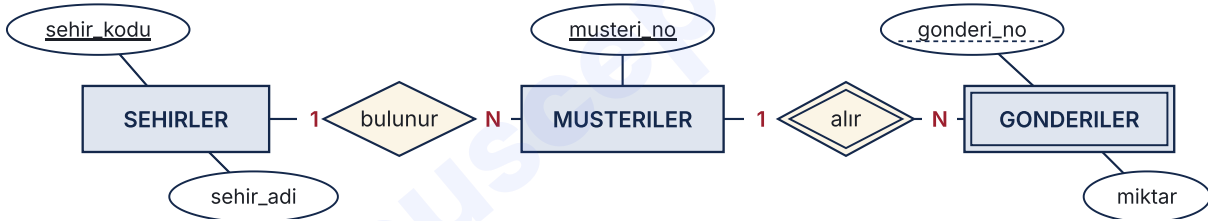
SATIŞ veri tabanındaki tek tablo aşağıdadır. Tablonun normalizasyonu 4.6'da adım adım yapılmış ve üç tabloya ulaşılmıştır. Bu uygulamada o tablolar SQL Server'da oluşturulacak, beş kayıt girilecek ve veri tabanı diyagramı çıkarılacaktır.

SATIŞ				
müşteri_no	şehir_kodu	şehir_adi	gönderi_no	miktar
1	34	İstanbul	1, 2, 3, 4, 6	300, 200, 400, 200, 100
2	6	Ankara	1, 2	300, 400
3	6	Ankara	2	200
4	34	İstanbul	2, 4, 5	200, 300, 400

Şehirler (şehir_kodu, şehir_adi)

Musteriler (musteri_no, şehir_kodu)

Gonderiler (musteri_no, gonderi_no, miktar)



Bir şehrin birden fazla müşterisi, bir müşterinin birden fazla gönderisi olabilir (iki 1-N ilişki)

1. Veri Tabanının ve Tabloların Oluşturulması

```

CREATE DATABASE SATIS;
GO
USE SATIS;
GO

-- Şehirler tablosu (ilk bu oluşturulur)
CREATE TABLE Şehirler (
    şehir_kodu INT PRIMARY KEY,
    şehir_adi VARCHAR(50) NOT NULL
);
  
```

```
-- Müşteriler tablosu (Şehirler tablosuna bağlı)
CREATE TABLE Musteriler (
    musteri_no INT PRIMARY KEY,
    sehir_kodu INT,
    FOREIGN KEY (sehir_kodu) REFERENCES Sehirler(sehir_kodu)
);

-- Gönderiler tablosu (Müşteriler tablosuna bağlı)
CREATE TABLE Gonderiler (
    musteri_no INT,
    gonderi_no INT,
    miktar INT,
    PRIMARY KEY (musteri_no, gonderi_no),
    FOREIGN KEY (musteri_no) REFERENCES Musteriler(musteri_no)
);
```

2. Kayıtların Girilmesi

Yabancı anahtar hatası almamak için önce ana tablolara, sonra bağlı tablolara veri girilir. Sıra: Şehirler → Musteriler → Gonderiler.

```
INSERT INTO Sehirler (sehir_kodu, sehir_adi)
VALUES (34, 'İstanbul'), (6, 'Ankara');

INSERT INTO Musteriler (musteri_no, sehir_kodu)
VALUES (1, 34), (2, 6), (3, 6), (4, 34);

-- 1 numaralı müşterinin beş gönderisi
INSERT INTO Gonderiler (musteri_no, gonderi_no, miktar)
VALUES (1, 1, 300), (1, 2, 200), (1, 3, 400), (1, 4, 200), (1, 6, 100);
```

3. SSMS'te Veri Tabanı Diyagramı (Database Diagram)

- 1 **Veri tabanını bul.**
Soldaki Object Explorer penceresinde SATIS veri tabanı genişletilir.
- 2 **Yeni diyagram aç.**
Database Diagrams klasörüne sağ tıklanır ve **New Database Diagram** seçilir.
- 3 **Tabloları ekle.**
“Add Table” penceresinde Şehirler, Musteriler ve Gonderiler tabloları seçilip **Add** düğmesine basılır.
- 4 **İlişkileri gör.**
FOREIGN KEY tanımları sayesinde tablolar arasındaki bağlantı çizgileri kendiliğinden çizilir. Anahtar simgesi birincil anahtar (1) tarafını, sonsuz (∞) simgesi çok (N) tarafını gösterir.