

FIRAT ÜNİVERSİTESİ
Mühendislik Fakültesi
Bilgisayar Mühendisliği Bölümü

BMÜ329 VERİ TABANI SİSTEMLERİ
DÖNEM PROJESİ RAPORU

Endemik Bitki Yetiştirme
Bölge Yönetim Sistemi

Grup No	15
Takım Üyesi 1	240260174 – Taha Buğra AK
Takım Üyesi 2	240260178 – Yusuf ÇİNAR
Takım Üyesi 3	240260168 – Mustafa ÇEKCEOGLU
Ders Sorumlusu	Ertan Bütün
Teslim Tarihi	06.01.2026

İÇİNDEKİLER

1. GİRİŞ VE PROJE TANIMI	3
1.1 Projenin Amacı ve Kapsamı	3
1.2 Hedef Kullanıcılar	3
2. PROJE GEREKSİNİMLERİ	4
2.1 Fonksiyonel Gereksinimler	4
2.2 Fonksiyonel Olmayan Gereksinimler	5
2.3 İş Kuralları, Kısıtlamalar	5
3. VERİ TABANI TASARIMI – E-R MODELİ	7
3.1 E-R Diyagramı	7
4. İLİŞKİSEL ŞEMALAR	8
4.1 Tablolar ve Alanlar	8
5. NORMALİZASYON	9
5.1 Normalizasyon Süreci	9
5.2 Son Normalize Edilmiş Şema	9
6. SQL SERVER VERİ TABANI ŞEMASI	10
6.1 Tablo Oluşturma Komutları	10
6.3 Veri Tabanı Diyagramı	12
7. ÖRNEK VERİLER	13
7.1 Veri Ekleme Komutları (INSERT)	13
8. SQL KOMUTLARI VE SCRIPT DOSYALARI	14
8.1 Gereksinim Bazlı SQL Komutları	14
9. SAKLI YORDAM VE TETİKLEYİCİ	15
9.1 Saklı Yordam (Stored Procedure)	15
9.1.1 Amaç ve İş Operasyonu	15
9.1.2 Saklı Yordam Kodu	15
9.1.3 Test Senaryoları ve Sonuçları	15
9.2 Tetikleyici (Trigger)	17
9.2.1 Amaç ve Tetikleme Koşulu	17
9.2.2 Tetikleyici Kodu	17
9.2.3 Test Senaryoları ve Sonuçları	17
10. TRANSACTION YÖNETİMİ	19
10.1 Transaction Senaryosu Adı	19
10.2 Transaction Kodu	19
10.3 Başarılı Senaryo Testi	20
10.4 Hata Senaryosu ve ROLLBACK Testi	21
11. TAKIM ÇALIŞMASI VE GÖREV DAĞILIMI	21
11.1 Takım Üyeleri ve Roller	21
11.2 Gerçekleştirilen İşler ve Sorumluluk Matrisi	22
12. SONUÇ VE DEĞERLENDİRME	24

1. GİRİŞ VE PROJE TANIMI

1.1 Projenin Amacı ve Kapsamı

Bu projenin temel amacı, Türkiye'deki endemik bitki türlerinin farklı coğrafi bölgelerdeki yetiştirilme potansiyelini bilimsel ve karşılaştırmalı bir yaklaşımla inceleyebilecek bir veri tabanı ve web tabanlı arayüz geliştirmektir. Sistem, bitki türleri, bölgeler, habitat/iklim koşulları ve bu koşullar altında gerçekleştirilen yetiştirme denemelerine ait gözlem verilerini merkezî bir yapıda toplamayı hedeflemektedir. Böylece araştırmacılar, belirli bir endemik türün farklı bölgelerdeki performansını, hangi çevresel koşullarda daha başarılı sonuç verdiğini ve zaman içindeki gelişimini tek bir sistem üzerinden analiz edebilecektir.

Projenin kapsamı; bitki, bölge, habitat, deneme ve ölçüm verilerinin tutulduğu ilişkisel veritabanının tasarlanmasını, bu veritabanının SQL Server üzerinde normalizasyon kurallarına uygun olarak gerçekleştirilmesini ve temel sorgulama/filtreleme işlevlerini destekleyen bir kullanıcı arayüzünün tasarlanmasını içermektedir. İlk sürümde otomatik sensör entegrasyonu, mobil uygulama, makine öğrenmesi tabanlı tahmin modelleri gibi gelişmiş özellikler kapsam dışında bırakılmıştır. Proje özellikle lisansüstü öğrenciler, akademisyenler ve ilgili kamu kurumlarında çalışan uzmanların endemik bitkilerle ilgili veri yönetimi ve analiz ihtiyaçlarını karşılamaya odaklanmaktadır

1.2 Hedef Kullanıcılar

- Araştırmacılar / akademisyenler
- Lisansüstü öğrenciler (yüksek lisans ve doktora)
- Tarım, orman, belediye gibi kurumlarda çalışan uzmanlar
- Sistem yöneticisi (veri tabanı ve kullanıcı yönetiminden sorumlu teknik kullanıcı)

2. PROJE GEREKSİNİMLERİ

2.1 Fonksiyonel Gereksinimler

ID	Gereksinim Adı	Açıklama
FR-01	Kullanıcı Girişi ve Yetkilendirme	Araştırmacı, öğrenci, kurum uzmanı ve admin kullanıcıları sisteme kullanıcı adı ve şifre ile giriş yapabilmeli, her rol yalnızca yetkili olduğu işlemlere erişebilmelidir.
FR-02	Bitki Kaydı Yönetimi	Endemik bitki türleri için bilimsel ad, yerel ad, familya, endemik durumu ve koruma durumu gibi bilgilerle yeni kayıt eklenebilmeli, mevcut kayıtlar güncellenebilmelidir.
FR-03	Bölge ve Habitat Bilgisi Yönetimi	Çalışma bölgeleri ve bu bölgelerdeki habitat/iklim özellikleri (rakım, sıcaklık aralığı, yağış, toprak tipi vb.) sisteme kaydedilebilmeli ve güncellenebilmelidir.
FR-04	Deneme / Gözlem Kaydı Oluşturma	Belirli bir bitki ve habitat için tarih aralığı, sorumlu araştırmacı, deneme durumu, başarı oranı ve açıklama bilgilerini içeren deneme kayıtları oluşturulabilmelidir.
FR-05	Ölçüm Verisi Kaydı	Bir deneme süresi boyunca farklı tarihlerde yapılan ölçümler (bitki boyu, canlılık durumu, verim, hastalık durumu vb.) sisteme kaydedilebilmelidir.
FR-06	Gelişmiş Filtreleme ve Sorgulama	Bitki, bölge, rakım, sıcaklık, toprak tipi, deneme durumu ve başarı oranı gibi ölçütlere göre kayıtlar filtrelenebilir ve listelenebilmelidir.
FR-07	Raporlama ve Özet İstatistikler	Tür bazında başarı oranı, bölge bazında deneme sayısı ve zaman içindeki deneme dağılımı gibi özet raporlar üretilebilmelidir.
FR-08	Harita Üzerinde Gösterim	Denemelerin yapıldığı bölgeler, koordinat bilgileri kullanılarak harita üzerinde görüntülenebilmeli ve seçilen bitki türüne göre filtrelenebilmelidir.
FR-09	Veri Dışa Aktarma	Yetkili kullanıcılar seçtikleri verileri CSV/Excel formatında dışa aktararak harici analizlerde kullanabilmelidir.

2.2 Fonksiyonel Olmayan Gereksinimler

ID	Gereksinim Adı	Açıklama
NFR-01	Güvenlik	Kullanıcı oturumları güvenli bağlantı üzerinden sağlanmalı, parolalar hash'lenmiş olarak saklanmalı ve rol tabanlı erişim kontrolü uygulanmalıdır.
NFR-02	Veri Bütünlüğü	Tüm veri ilişkileri foreign key kısıtlarıyla korunmalı, zorunlu alanlar NOT NULL tanımlanmalı ve uygun CHECK kısıtlarıyla hatalı veri girişi engellenmelidir.
NFR-03	Yedekleme	Veritabanı en az günlük yedekleme politikasını desteklemeli, kritik hatalarda belirli bir zaman noktasına geri dönebilmelidir
NFR-04	Performans	Normal yük altında temel sorguların çoğu 3 saniye içinde sonuçlanmalı, sık kullanılan alanlar için uygun indeksler tanımlanmalıdır.
NFR-05	Kullanılabilirlik	Web arayüzü basit ve anlaşılır olmalı, temel işlemlere en fazla birkaç adımda erişilebilmelidir
NFR-06	Ölçeklenebilirlik	Artan bitki, bölge ve deneme sayısına rağmen sistem yatay/dikey ölçekleme seçenekleriyle çalışmaya devam edebilmelidir.
NFR-07	Denetlenebilirlik	Kritik işlemler (giriş denemeleri, deneme ekleme/güncelleme, silme) loglanmalı ve sistem yöneticisi tarafından izlenebilir olmalıdır.

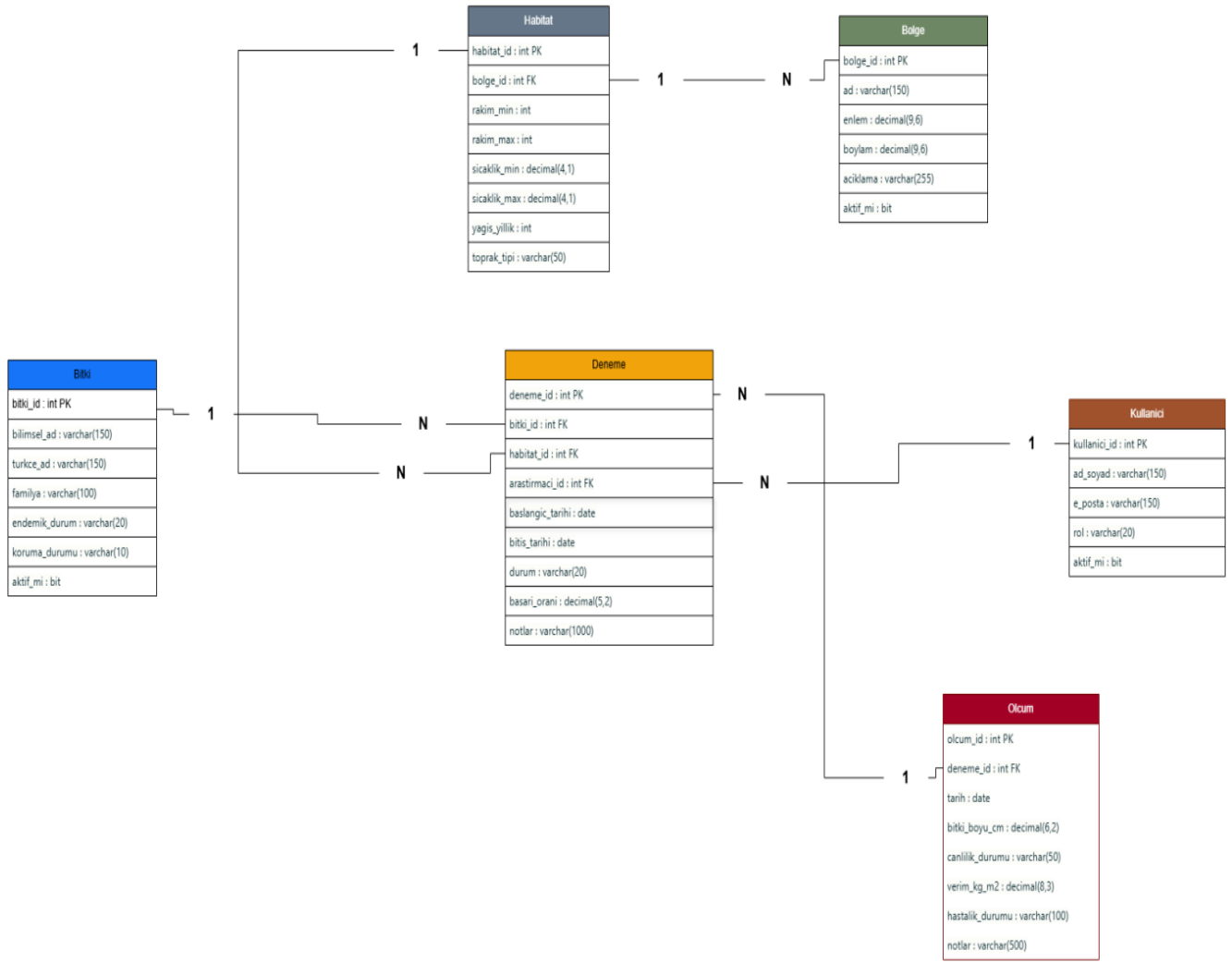
2.2 İş Kuralları, Kısıtlamalar

- Aynı bitki, aynı habitat ve aynı başlangıç tarihi için birden fazla aktif deneme kaydı oluşturulamaz; mükerrer kayıt reddedilmelidir.
- Her deneme mutlaka bir bitki ve bir habitat ile ilişkilendirilmelidir; bu alanlar NULL olamaz.
- Deneme durumu alanı yalnızca şu değerlerden birini alabilir: PLANLANDI, DEVAM, BASARILI, BASARISIZ, IPTAL.

- Başarı oranı yüzdesi 0 ile 100 aralığında olmalıdır; bu aralığın dışındaki değerler kabul edilmemelidir.
- Bir denemede başlangıç tarihi, bitiş tarihinden büyük olamaz; tarih koşulunu sağlamayan kayıtlar kaydedilemez.
- Ölçüm kaydı eklenirken en az tarih alanı doldurulmalı, ölçüm bir deneme ile ilişkilendirilmeden kaydedilemez.
- Araştırmacılar yalnızca kendi oluşturdukları denemeleri güncelleyebilir; diğer kullanıcıların denemelerini sadece görüntüleyebilir.
- Bir bitki “pasif” durumuna alındığında bu bitkiye ait yeni deneme açılmaz; yalnızca mevcut denemeler görüntülenebilir.
- Bir bölge veya habitat “pasif” durumdaysa, bu bölge/habitat için yeni deneme oluşturulamaz; eski denemeler raporlanmaya devam eder.
- Ölçüm kaydı bulunmayan bir denemenin durumu BASARILI veya BASARISIZ olarak güncellenemez; önce en az bir ölçüm kaydı girilmelidir.

3. VERİ TABANI TASARIMI - VARLIK-İLİŞKİ (E-R) MODELİ

3.1 E-R Diyagramı



4. İLİŞKİSEL ŞEMALAR

4.1 E-R'dan İlişkisel Şemalarına Dönüşüm

BITKI(bitki_id, bilimsel_ad, turkce_ad, familya, endemik_durum, koruma_durumu, aktif_mi)

BOLGE(bolge_id, ad, enlem, boylam, aciklama, aktif_mi)

HABITAT(habitat_id, bolge_id, rakim_min, rakim_max, sicaklik_min, sicaklik_max, yagis_yillik, toprak_tipi)

KULLANICI(kullanici_id, ad_soyad, e_posta, rol, aktif_mi)

DENEME(deneme_id, bitki_id, habitat_id, arastirmaci_id, baslangic_tarihi, bitis_tarihi, durum, basari_orani, notlar)

OLCUM(olcum_id, deneme_id, tarih, bitki_boyu_cm, canlilik_durumu, verim_kg_m2, hastalik_durumu, notlar)

İLİŞKİ TÜRLERİ:

BITKI — DENEME 1 → N

BOLGE — HABITAT 1 → N

HABITAT — DENEME 1 → N

KULLANICI — DENEME 1 → N

DENEME — OLCUM 1 → N

5. NORMALİZASYON

5.1 Normalizasyon Süreci

- Tüm tablolar önce 1NF'e getirilmiş, her alan atomik hale getirilmiş ve tekrar eden grup içeren sütun bırakılmamıştır.
- Ardından 2NF için her tabloda bileşik anahtar gerektiren durumlar ayrılmış, kısmi bağımlılık oluşturan nitelikler yeni tablolara taşınmıştır.
- Son olarak 3NF sağlanarak anahtar olmayan nitelikler arasındaki geçişli bağımlılıklar giderilmiş, her nitelik yalnızca kendi tablosunun birincil anahtarına tam bağımlı olacak şekilde düzenlenmiştir

5.2 Son Normalize Edilmiş Şema

Son normalize edilmiş şema, BITKI, BOLGE, HABITAT, KULLANICI, DENEME ve OLCUM tablolarından oluşan, tümü en az 3NF koşullarını sağlayan ilişkisel yapıdır

6. SQL SERVER VERİ TABANI ŞEMASI

6.1 Tablo Oluşturma Komutları

```
-- DB
CREATE DATABASE EndemikBitkiDB;
GO
USE EndemikBitkiDB;
GO

-- Bitki
CREATE TABLE dbo.Bitki (
    bitki_id          INT IDENTITY(1,1) PRIMARY KEY,
    bilimsel_ad       VARCHAR(150) NOT NULL,
    turkce_ad        VARCHAR(150) NULL,
    familya          VARCHAR(100) NULL,
    endemik_durum     VARCHAR(20) NOT NULL
        CHECK (endemik_durum IN ('EVET', 'HAYIR')),
    koruma_durumu     VARCHAR(10) NULL,
    aktif_mi         BIT NOT NULL DEFAULT 1
);
GO

-- Bölge
CREATE TABLE dbo.Bolge (
    bolge_id          INT IDENTITY(1,1) PRIMARY KEY,
    ad                VARCHAR(150) NOT NULL,
    enlem             DECIMAL(9,6) NOT NULL,
    boylam            DECIMAL(9,6) NOT NULL,
    aciklama          VARCHAR(255) NULL,
    aktif_mi         BIT NOT NULL DEFAULT 1
);
GO

-- Habitat
CREATE TABLE dbo.Habitat (
    habitat_id        INT IDENTITY(1,1) PRIMARY KEY,
    bolge_id          INT NOT NULL,
    rakim_min         INT NULL,
    rakim_max         INT NULL,
    sicaklik_min      DECIMAL(4,1) NULL,
    sicaklik_max      DECIMAL(4,1) NULL,
    yagis_yillik      INT NULL,
    toprak_tipi       VARCHAR(50) NOT NULL,
    CONSTRAINT fk_Habitat_Bolge
        FOREIGN KEY (bolge_id) REFERENCES dbo.Bolge(bolge_id)
        ON UPDATE CASCADE,
    CONSTRAINT chk_Habitat_Rakim
        CHECK (rakim_min IS NULL OR rakim_max IS NULL OR rakim_min <= rakim_max),
    CONSTRAINT chk_Habitat_Sicaklik
        CHECK (sicaklik_min IS NULL OR sicaklik_max IS NULL OR sicaklik_min <=
sicaklik_max)
);
GO

-- Kullanici
CREATE TABLE dbo.Kullanici (
    kullanici_id      INT IDENTITY(1,1) PRIMARY KEY,
    ad_soyad          VARCHAR(150) NOT NULL,
    e_posta           VARCHAR(150) NOT NULL UNIQUE,
    rol               VARCHAR(20) NOT NULL
);
```

```

        CHECK (rol IN ('ARASTIRMACI', 'OGRENCI', 'KURUM', 'ADMIN')),
    aktif_mi          BIT NOT NULL DEFAULT 1
);
GO

-- Deneme
CREATE TABLE dbo.Deneme (
    deneme_id          INT IDENTITY(1,1) PRIMARY KEY,
    bitki_id           INT NOT NULL,
    habitat_id         INT NOT NULL,
    arastirmaci_id     INT NOT NULL,
    baslangic_tarihi   DATE NOT NULL,
    bitis_tarihi       DATE NULL,
    durum              VARCHAR(20) NOT NULL
        CHECK (durum IN ('PLANLANDI', 'DEVAM', 'BASARILI', 'BASARISIZ', 'IPTAL')),
    basari_orani        DECIMAL(5,2) NULL
        CHECK (basari_orani IS NULL OR (basari_orani >= 0 AND basari_orani <=
100)),
    notlar              VARCHAR(1000) NULL,
    CONSTRAINT fk_Deneme_Bitki
        FOREIGN KEY (bitki_id) REFERENCES dbo.Bitki(bitki_id)
        ON UPDATE CASCADE,
    CONSTRAINT fk_Deneme_Habitat
        FOREIGN KEY (habitat_id) REFERENCES dbo.Habitat(habitat_id)
        ON UPDATE CASCADE,
    CONSTRAINT fk_Deneme_Kullanici
        FOREIGN KEY (arastirmaci_id) REFERENCES dbo.Kullanici(kullanici_id)
        ON UPDATE CASCADE,
    CONSTRAINT chk_Deneme_Tarih
        CHECK (bitis_tarihi IS NULL OR baslangic_tarihi <= bitis_tarihi),
    CONSTRAINT uq_Deneme_BitkiHabitatBaslangic
        UNIQUE (bitki_id, habitat_id, baslangic_tarihi)
);
GO

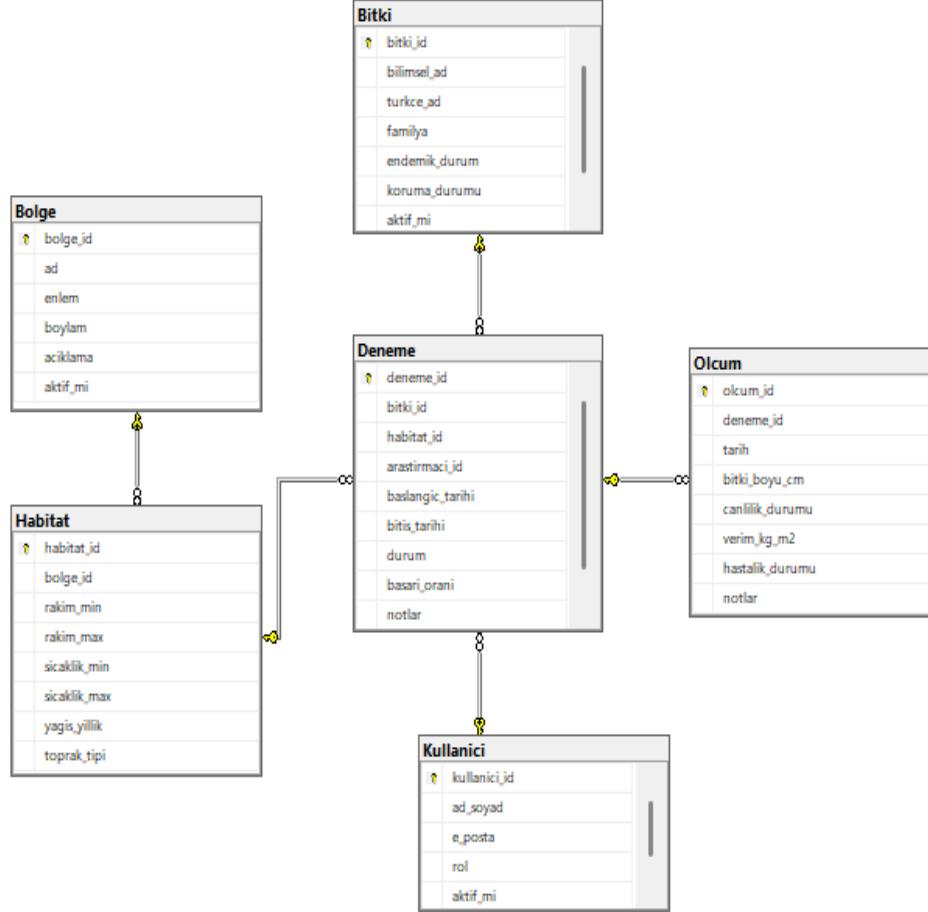
-- Olcum
CREATE TABLE dbo.Olcum (
    olcum_id           INT IDENTITY(1,1) PRIMARY KEY,
    deneme_id          INT NOT NULL,
    tarih              DATE NOT NULL,
    bitki_boyu_cm       DECIMAL(6,2) NULL,
    canlilik_durumu     VARCHAR(50) NULL,
    verim_kg_m2         DECIMAL(8,3) NULL,
    hastalik_durumu     VARCHAR(100) NULL,
    notlar              VARCHAR(500) NULL,
    CONSTRAINT fk_Olcum_Deneme
        FOREIGN KEY (deneme_id) REFERENCES dbo.Deneme(deneme_id)
        ON UPDATE CASCADE ON DELETE CASCADE
);
GO

-- Indexler
CREATE INDEX IX_Deneme_Bitki          ON dbo.Deneme(bitki_id);
CREATE INDEX IX_Deneme_Habitat        ON dbo.Deneme(habitat_id);
CREATE INDEX IX_Deneme_Durum          ON dbo.Deneme(durum);
CREATE INDEX IX_Olcum_DenemeTarih     ON dbo.Olcum(deneme_id, tarih);
GO

```

6.3 Veri Tabanı Diyagramı

VERİ TABANI DİYAGRAMI EKRAN GÖRÜNTÜSÜ



7. ÖRNEK VERİLER

7.1 Veri Ekleme Komutları (INSERT)

```
USE EndemikBitkiDB;
GO

-- BITKI
INSERT INTO Bitki (bilimsel_ad, turkce_ad, familya, endemik_durum, koruma_durumu)
VALUES
('Centaurea tchihatcheffii', 'Yanardoner Peygamber Cicegi', 'Asteraceae', 'EVET',
'EN'),
('Astragalus gummifer', 'Keven', 'Fabaceae', 'EVET', 'VU'),
('Verbascum davisianum', 'Sigirkuyruğu', 'Scrophulariaceae', 'EVET', 'NT'),
('Pinus nigra subsp. pallasiana', 'Karacam', 'Pinaceae', 'HAYIR', 'LC');

-- BOLGE
INSERT INTO Bolge (ad, enlem, boylam, aciklama)
VALUES
('Acigol - Nevsehir', 38.550000, 34.500000, 'Ic Anadolu, Acigol cevresi'),
('Erciyes Kuzey Yamac', 38.600000, 35.450000, 'Yuksek rakimli daglik alan'),
('Antalya Beydaglari', 36.550000, 30.200000, 'Akdeniz iklimi daglik alan');

-- HABITAT
INSERT INTO Habitat (bolge_id, rakim_min, rakim_max, sicaklik_min, sicaklik_max,
yagis_yillik, toprak_tipi)
VALUES
(1, 1100, 1300, -5.0, 30.0, 350, 'Kirecli'),
(1, 900, 1100, -3.0, 32.0, 320, 'Tinli'),
(2, 1500, 1800, -10.0, 25.0, 500, 'Tasli'),
(3, 800, 1000, 0.0, 35.0, 700, 'Kumlu-tinli');

-- KULLANICI
INSERT INTO Kullanici (ad_soyad, e_posta, rol)
VALUES
('Ayse Yilmaz', 'ayse.yilmaz@gmail.com', 'ARASTIRMACI'),
('Mehmet Demir', 'mehmet.demir@gmail.com', 'ARASTIRMACI'),
('Zeynep Koc', 'zeynep.koc@gmail.com', 'OGRENCI'),
('Tarim Il Mudurlugu Uzmani', 'uzman@tarim.gov.tr', 'KURUM'),
('Sistem Yöneticisi', 'admin@endemik.local', 'ADMIN');
```

8. SQL KOMUTLARI VE SCRIPT DOSYALARI

8.1 Gereksinim Bazlı SQL Komutları ve Script Dosyaları

- Belirli bir bitki türü için yapılan tüm denemeler listelenebilecek.

SQL Komutu:

```
SELECT
    d.deneme_id,
    b.bilimsel_ad,
    d.baslangic_tarihi,
    d.bitis_tarihi,
    d.durum,
    d.basari_orani
FROM Deneme d
JOIN Bitki b ON d.bitki_id = b.bitki_id
WHERE b.bilimsel_ad = 'Astragalus gummifer';
```

- Devam eden denemeler ve sorumlu araştırmacılar listelenebilecek.

SQL Komutu:

```
SELECT
    d.deneme_id,
    b.bilimsel_ad,
    k.ad_soyad,
    d.baslangic_tarihi,
    d.durum
FROM Deneme d
JOIN Bitki b ON d.bitki_id = b.bitki_id
JOIN Kullanici k ON d.arastirmaci_id = k.kullanici_id
WHERE d.durum = 'DEVAM';
```

- Belirli bir tarihten sonra yapılan ölçümler listelenebilecek.

SQL Komutu:

```
SELECT
    o.olcum_id,
    o.deneme_id,
    o.tarih,
    o.bitki_boyu_cm,
    o.verim_kg_m2
FROM Olcum o
WHERE o.tarih >= '2025-01-01';
```

Script Dosyası:

<https://drive.google.com/file/d/1YEG8jYyoA4yMK3mjSNS8JB9DoeZMHkEC/view?usp=sharing>

9. SAKLI YORDAM VE TETİKLEYİCİ

9.1 Saklı Yordam Adı (Stored Procedure)

9.1.1 Amaç ve İş Operasyonu

Bu saklı yordam, yeni bir deneme kaydı ile bu denemeye ait ilk ölçüm kaydını tek bir işlemde eklemek için kullanılır.

İşlem sırasında tarih ve başarı oranı gibi temel kurallar kontrol edilir; hata olursa tüm işlem geri alınır

9.1.2 Saklı Yordam Kodu

```
CREATE PROCEDURE sp_DenemeVeIlkOlcumEkle1
    @bitki_id          INT,
    @habitat_id        INT,
    @arastirmaci_id    INT,
    @baslangic_tarihi  DATE,
    @bitis_tarihi      DATE = NULL,
    @durum              VARCHAR(20),
    @basari_orani       DECIMAL(5,2) = NULL,
    @deneme_not         VARCHAR(1000) = NULL,
    @olcum_tarihi       DATE,
    @bitki_boyu_cm      DECIMAL(6,2) = NULL,
    @canlilik_durumu    VARCHAR(50) = NULL,
    @verim_kg_m2        DECIMAL(8,3) = NULL,
    @hastalik_durumu    VARCHAR(100) = NULL,
    @olcum_not          VARCHAR(500) = NULL
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRAN;

    IF @bitis_tarihi IS NOT NULL AND @baslangic_tarihi > @bitis_tarihi
    BEGIN
        RAISERROR('Baslangic tarihi bitis tarihinden büyük olamaz.', 16, 1);
        ROLLBACK TRAN;
        RETURN;
    END;

    INSERT INTO Deneme (
        bitki_id, habitat_id, arastirmaci_id,
        baslangic_tarihi, bitis_tarihi, durum,
        basari_orani, notlar
    )
    VALUES (
        @bitki_id, @habitat_id, @arastirmaci_id,
        @baslangic_tarihi, @bitis_tarihi, @durum,
        @basari_orani, @deneme_not
    );

    DECLARE @deneme_id INT = SCOPE_IDENTITY();

    INSERT INTO Olcum (
        deneme_id, tarih, bitki_boyu_cm,
        canlilik_durumu, verim_kg_m2,
        hastalik_durumu, notlar
    )
    VALUES (
```



```

        @deneme_id, @olcum_tarihi, @bitki_boyu_cm,
        @canlilik_durumu, @verim_kg_m2,
        @hastalik_durumu, @olcum_not
    );

    COMMIT TRAN;
END;
GO

```

9.1.3 Test Senaryoları ve Sonuçları

```

EXEC sp_DenemeVeIlkOlcumEkle
    @bitki_id = 18,
    @habitat_id = 12,
    @arastirmaci_id = 9,
    @baslangic_tarihi = '2025-03-01',
    @bitis_tarihi = '2025-06-01',
    @durum = 'DEVAM',
    @basari_orani = NULL,
    @deneme_not = 'Test denemesi',
    @olcum_tarihi = '2025-03-05',
    @bitki_boyu_cm = 12.5,
    @canlilik_durumu = 'iyi',
    @verim_kg_m2 = NULL,
    @hastalik_durumu = NULL,
    @olcum_not = 'İlk ölçüm';

```

BAŞARILI DENEME

-- başlangıç tarihi bitişten büyük

```

EXEC sp_DenemeVeIlkOlcumEkle
    @bitki_id = 18,
    @habitat_id = 12,
    @arastirmaci_id = 9,
    @baslangic_tarihi = '2025-07-10',
    @bitis_tarihi = '2025-07-01', -- HATALI KISIM
    @durum = 'PLANLANDI',
    @basari_orani = NULL,
    @deneme_not = 'Tarih hatası testi',
    @olcum_tarihi = '2025-07-10',
    @bitki_boyu_cm = 10.0,
    @canlilik_durumu = 'orta',
    @verim_kg_m2 = NULL,
    @hastalik_durumu = NULL,
    @olcum_not = 'Bu kayıt eklenmemeli';

```

HATA:

İleti 50000, Düzey 16, Durum 1, Yordam sp_DenemeVeIlkOlcumEkle, Satır 24 [Toplu İş Başlangıç Satır 0]
 Baslangic tarihi bitis tarihinden büyük olamaz.

Tamamlanma süresi:2025-12-27T19:36:10.8082538+03:00

9.2 Tetikleyici (Trigger)

9.2.1 Amaç ve Tetikleme Koşulu

Bu tetikleyici, DENEME tablosunda durum alanı BASARILI veya BASARISIZ olarak güncellendiğinde çalışır.

Amaç, sonuçlanan bir denemenin en az bir tane ölçüm kaydına (OLCUM) sahip olmasını zorunlu kılmaktır.

9.2.2 Tetikleyici Kodu

```
CREATE TRIGGER trg_Deneme_Durum_Kontrol
ON dbo.Deneme
AFTER UPDATE
AS
BEGIN
    SET NOCOUNT ON;

    -- Sadece durum değişen satırlar
    IF NOT EXISTS (
        SELECT 1
        FROM inserted i
        JOIN deleted d ON i.deneme_id = d.deneme_id
        WHERE i.durum <> d.durum
    )
        RETURN;

    -- BASARILI / BASARISIZ yapılan denemeler için ölçüm kontrolü
    IF EXISTS (
        SELECT 1
        FROM inserted i
        WHERE i.durum IN ('BASARILI', 'BASARISIZ')
        AND NOT EXISTS (
            SELECT 1
            FROM Olcum o
            WHERE o.deneme_id = i.deneme_id
        )
    )
    BEGIN
        RAISERROR('Deneme BASARILI/BASARISIZ yapılamaz: Ilgili Olcum kaydi yok.',
16, 1);
        ROLLBACK TRANSACTION;
        RETURN;
    END;
END;
GO
```

9.2.3 Test Senaryoları ve Sonuçları

```
-- ölçüm kaydı olan bir deneme id'si
UPDATE Deneme
SET durum = 'BASARILI'
WHERE deneme_id = 2;

SELECT * from Deneme where deneme_id=2
```

deneme_id	durum
2	BASARILI

```
-- ölçüm kaydı olmayan bir deneme id'si
```

```
UPDATE Deneme
SET durum = 'BASARILI'
WHERE deneme_id = 11;
```

HATA:

```
İleti 50000, Düzey 16, Durum 1, Yordam trg_Deneme_Durum_Kontrol, Satır 29 [Toplu
İş Başlangıç Satırı 43]
Deneme BASARILI/BASARISIZ yapılamaz: İlgili Olcum kaydi yok.
İleti 3609, Düzey 16, Durum 1, Satır 44
The transaction ended in the trigger. The batch has been aborted.
```

10. TRANSACTION YÖNETİMİ

10.1 Transaction Senaryosu Adı

Deneme ve ilk ölçüm kaydının tek transaction ile eklenmesi.

10.2 Transaction Kodu

```
CREATE PROCEDURE sp_DenemeVeIlkOlcumEkle
    @bitki_id          INT,
    @habitat_id        INT,
    @arastirmaci_id    INT,
    @baslangic_tarihi  DATE,
    @bitis_tarihi       DATE = NULL,
    @durum              VARCHAR(20),
    @basari_orani       DECIMAL(5,2) = NULL,
    @deneme_not         VARCHAR(1000) = NULL,
    @olcum_tarihi       DATE,
    @bitki_boyu_cm      DECIMAL(6,2) = NULL,
    @canlilik_durumu    VARCHAR(50) = NULL,
    @verim_kg_m2        DECIMAL(8,3) = NULL,
    @hastalik_durumu    VARCHAR(100) = NULL,
    @olcum_not          VARCHAR(500) = NULL
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRAN;

    IF @bitis_tarihi IS NOT NULL AND @baslangic_tarihi > @bitis_tarihi
    BEGIN
        RAISERROR('Baslangic tarihi bitis tarihinden büyük olamaz.', 16, 1);
        ROLLBACK TRAN;
        RETURN;
    END;

    INSERT INTO Deneme (
        bitki_id, habitat_id, arastirmaci_id,
        baslangic_tarihi, bitis_tarihi, durum,
        basari_orani, notlar
    )
    VALUES (
        @bitki_id, @habitat_id, @arastirmaci_id,
        @baslangic_tarihi, @bitis_tarihi, @durum,
        @basari_orani, @deneme_not
    );

    DECLARE @deneme_id INT = SCOPE_IDENTITY();

    INSERT INTO Olcum (
        deneme_id, tarih, bitki_boyu_cm,
        canlilik_durumu, verim_kg_m2,
        hastalik_durumu, notlar
    )
    VALUES (
        @deneme_id, @olcum_tarihi, @bitki_boyu_cm,
        @canlilik_durumu, @verim_kg_m2,
        @hastalik_durumu, @olcum_not
    );
```

```

);

COMMIT TRAN;

END;

GO

```

10.3 Başarılı Senaryo Testi

```

EXEC sp_DenemeVeIlkOlcumEkle
    @bitki_id = 18,
    @habitat_id = 12,
    @arastirmaci_id = 9,
    @baslangic_tarihi = '2025-03-01',
    @bitis_tarihi = '2025-06-01',
    @durum = 'DEVAM',
    @basari_orani = NULL,
    @deneme_not = 'Transaction basarili senaryo',
    @olcum_tarihi = '2025-03-05',
    @bitki_boyu_cm = 12.5,
    @canlilik_durumu = 'iyi',
    @verim_kg_m2 = NULL,
    @hastalik_durumu = NULL,
    @olcum_not = 'Ilk olcum';

```

DENEME TABLOSU

deneme_id	bitki_id	baslangic_tarihi	bitis_tarihi	durum
2	18	2025-09-01	2026-03-01	BASARILI
7	18	2025-03-01	2025-06-01	BASARILI
11	18	2025-11-01	2026-01-01	PLANLANDI

OLCUM TABLOSU

olcum_id	deneme_id	tarih	bitki_boyu_cm	verim_kg_m2	hastalik_durumu
1	2	2025-09-15	4.50	NULL	NULL
6	7	2025-03-05	12.50	NULL	NULL

10.4 Hata Senaryosu ve ROLLBACK Testi

```
EXEC sp_DenemeVeIlkOlcumEkle
    @bitki_id = 18,
    @habitat_id = 12,
    @arastirmaci_id = 9,
    @baslangic_tarihi = '2025-07-10',
    @bitis_tarihi = '2025-07-01',    -- Hatalı tarih
    @durum = 'PLANLANDI',
    @basari_orani = NULL,
    @deneme_not = 'Transaction hata senaryosu',
    @olcum_tarihi = '2025-07-10',
    @bitki_boyu_cm = 10.0,
    @canlilik_durumu = 'orta',
    @verim_kg_m2 = NULL,
    @hastalik_durumu = NULL,
    @olcum_not = 'Bu kayıt eklenmemeli';
```

İleti 50000, Düzey 16, Durum 1, Yordam sp_DenemeVeIlkOlcumEkle, Satır 24 [Toplu İş Başlangıç Satırı 0]
Baslangic tarihi bitis tarihinden büyük olamaz.

Tamamlanma süresi:2025-12-27T20:15:34.5370928+03:00

11. TAKIM ÇALIŞMASI VE GÖREV DAĞILIMI

11.1 Takım Üyeleri ve Roller

Takım Üyesi	Öğrenci No	Temel Rol
Yusuf ÇINAR	240260178	Proje Lideri / Veri Tabanı Tasarımcısı
Taha Buğra AK	240260174	SQL Geliştirici / Kalite Kontrol
Mustafa ÇEKCEOĞLU	240260168	Dokümantasyon / Test Sorumlusu

11.2 Gerçekleştirilen İşler ve Sorumluluk Matrisi

No	Yapılan İş / Özellik	Talimat Veren	Gerçekleştiren	Kontrol Eden	Durum
1	Proje gereksinimlerinin belirlenmesi	Yusuf ÇINAR	Taha Buğra AK	Mustafa ÇEKCEOĞLU	✓
2	E-R diyagramının oluşturulması	Yusuf ÇINAR	Mustafa ÇEKCEOĞLU	Taha Buğra AK	✓
3	İlişkisel şemalara dönüştürme	Taha Buğra AK	Yusuf ÇINAR	Mustafa ÇEKCEOĞLU	✓
4	Normalizasyon (3NF/BCNF)	Taha Buğra AK	Mustafa ÇEKCEOĞLU	Yusuf ÇINAR	✓
5	SQL Server'da şema oluşturma	Mustafa ÇEKCEOĞLU	Yusuf ÇINAR	Taha Buğra AK	✓
6	Birincil/Yabancı anahtar tanımları	Mustafa ÇEKCEOĞLU	Taha Buğra AK	Yusuf ÇINAR	✓
7	Nitelik kısıtlamalarının eklenmesi	Yusuf ÇINAR	Mustafa ÇEKCEOĞLU	Taha Buğra AK	✓
8	Örnek verilerin eklenmesi	Taha Buğra AK	Yusuf ÇINAR	Mustafa ÇEKCEOĞLU	✓
9	SQL sorgu script dosyalarının hazırlanması	Mustafa ÇEKCEOĞLU	Taha Buğra AK	Yusuf ÇINAR	✓
10	Saklı yordamın (Stored Procedure) yazılması	Yusuf ÇINAR	Mustafa ÇEKCEOĞLU	Taha Buğra AK	✓
11	Tetikleyicinin (Trigger) yazılması	Taha Buğra AK	Yusuf ÇINAR	Mustafa ÇEKCEOĞLU	✓

No	Yapılan İş / Özellik	Talimat Veren	Gerçekleştiren	Kontrol Eden	Durum
12	Transaction yönetimi kodunun yazılması	Mustafa ÇEKCEOĞLU	Taha Buğra AK	Yusuf ÇİNAR	✓
13	COMMIT/ROLLBACK test senaryoları	Yusuf ÇİNAR	Taha Buğra AK	Mustafa ÇEKCEOĞLU	✓
14	Rapor hazırlama ve dokümantasyon	Taha Buğra AK	Mustafa ÇEKCEOĞLU	Yusuf ÇİNAR	✓
15	Son kontrol ve teslim	Mustafa ÇEKCEOĞLU	Yusuf ÇİNAR	Taha Buğra AK	✓

12. SONUÇ VE DEĞERLENDİRME

Bu projede ilişkisel veritabanı tasarımının tüm adımlarını uçtan uca deneyimleme fırsatı elde edildi. Bitki, habitat ve deneme süreçlerini modelleme sırasında gereksinim analizi, E-R diyagramı, normalizasyon (3NF'e kadar) ve SQL Server üzerinde şema kurma konularında pratik bilgi kazanıldı.

Süreçte en çok zorlayıcı olan kısımlar; tablolar arası ilişkileri doğru kurmak, uygun birincil/yabancı anahtarları seçmek ve saklı yordam ile tetikleyici mantığını hatasız kurgulamaktı. Deneme-ölçüm ilişkisi, transaction yönetimi ve hata durumlarında ROLLBACK kullanımı üzerinde çalışılırken birden fazla kez FK ve tarih kısıtı hatalarıyla karşılaşıldı; bu hatalar incelenip düzeltilerek veri bütünlüğünün önemi daha iyi kavrandı.

Sonuç olarak, projede hem teorik kavramlar (ilişkisel model, normalizasyon, kısıtlar) hem de uygulama becerileri (SQL DDL/DML komutları, stored procedure, trigger, index kullanımı) pekiştirildi. Elde edilen deneyim, daha büyük ve karmaşık veritabanı projelerinde gereksinim toplama, tasarım yapma ve hataları sistematik biçimde giderme konusunda önemli bir temel oluşturdu.