VERİ TABANI YÖNETİM SİSTEMLERİ DERSİ DÖNEM PROJESİ RAPORU

Öğrenci Bilgileri:

Ad Soyad: Abdulhamid Geylani BAĞCI

Öğrenci No: 242104032

Dönem: 2025-2026 Güz/Bahar

Proje Konusu: BGC KARGO - Kargo Takip ve Yönetim Sistemi

1. PROJENİN AMACI VE KAPSAMI

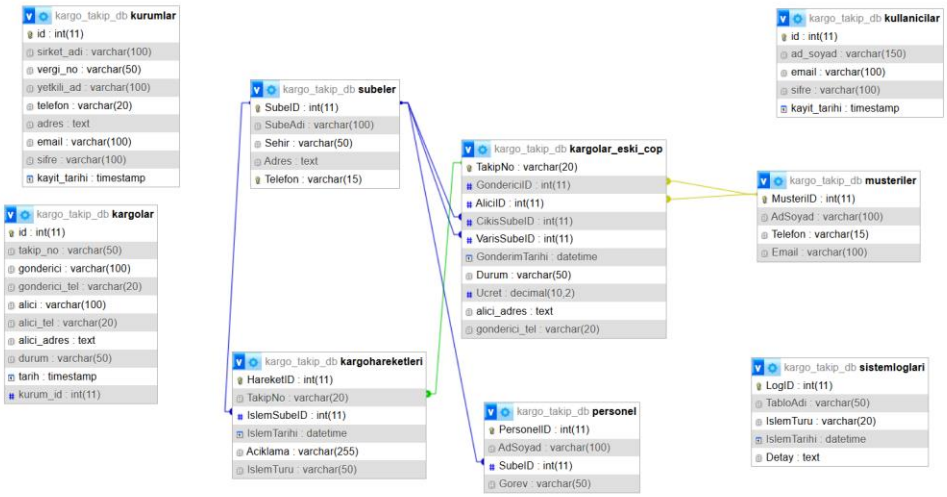
Bu proje, **BGC KARGO** isimli lojistik firmasının operasyonel süreçlerini (kargo kabul, transfer, teslimat ve takip) dijital ortamda yönetmek amacıyla geliştirilen ilişkisel bir veri tabanı sistemidir. Projenin temel amacı; müşterilerin kargolarını web arayüzü üzerinden anlık takip edebilmesini sağlamak, yöneticilerin şube performanslarını raporlayabilmesi ve veri bütünlüğünün **ileri SQL teknikleri** (Trigger, Transaction, Stored Procedure) ile korunmasıdır.

Sistem, **PHP** programlama dili ile geliştirilen web tabanlı bir arayüze sahiptir ve arka planda **MySQL** veri tabanı yönetim sistemini kullanır.

2. VERİ TABANI TASARIMI VE İLİŞKİSEL YAPI (Bölüm A)

2.1. Tablo Yapısı ve İlişkiler

Proje gereksinimlerinde belirtilen asgari şartları sağlamak üzere sistemde **6 adet temel tablo** oluşturulmuştur. Tabloların tasarımında veri tekrarını önlemek için ilişkisel modelleme (Relational Model) kullanılmıştır.



Tablo Listesi ve Görevleri:

1. **`kargolar`:** Sistemin merkez tablosudur. Takip no, durum, fiyat ve ilişki ID'lerini tutar.
2. **`kargohareketleri`:** Kargonun tarihçesini (Loglarını) tutar (1:N İlişki).
3. **`musteriler`:** Gönderici ve alıcı bilgilerini saklar.
4. **`subeler`:** Şube iletişim ve lokasyon bilgilerini tutar.
5. **`personel`:** Sisteme giriş yapacak yetkili kullanıcıları barındırır.
6. **`sistemloglari`:** Veri tabanındaki kritik değişiklikleri (Trigger ile) kaydeden log tablosudur.

2.2. Tablo Oluşturma Scriptleri (DDL)

SQL

```
USE kargo_takip_db;

-- 1. Şubeler Tablosu
CREATE TABLE IF NOT EXISTS subeler (
  sube_id INT AUTO_INCREMENT PRIMARY KEY,
    sube_adi VARCHAR(100) NOT NULL,
    il VARCHAR(50) DEFAULT 'Şanlıurfa',
    telefon VARCHAR(20)
) ENGINE=InnoDB;
```

```
-- 2. Müşteriler Tablosu
CREATE TABLE IF NOT EXISTS musteriler (
    musteri_id INT AUTO_INCREMENT PRIMARY KEY,
    ad_soyad VARCHAR(100) NOT NULL,
    telefon VARCHAR(20),
    adres TEXT,
    tc_kimlik VARCHAR(11) UNIQUE
) ENGINE=InnoDB;

-- 3. Personel Tablosu (Giriş İşlemleri İçin)
CREATE TABLE IF NOT EXISTS personel (
    personel_id INT AUTO_INCREMENT PRIMARY KEY,
    sube_id INT,
    ad_soyad VARCHAR(100) NOT NULL,
    kullanici_adi VARCHAR(50) UNIQUE,
    sifre VARCHAR(255),
    gorev VARCHAR(50) DEFAULT 'Kurye',
    FOREIGN KEY (sube_id) REFERENCES subeler(sube_id)
) ENGINE=InnoDB;

-- 4. Kargolar Tablosu (Ana Tablo)
CREATE TABLE IF NOT EXISTS kargolar (
    id INT AUTO_INCREMENT PRIMARY KEY,
    takip_no VARCHAR(20) NOT NULL UNIQUE,
    gonderici_id INT,
    alici_id INT,
    cikis_sube_id INT,
    durum VARCHAR(50) DEFAULT 'Hazırlanıyor',
    fiyat DECIMAL(10,2),
    olusturma_tarihi TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (gonderici_id) REFERENCES musteriler(musteri_id),
    FOREIGN KEY (alici_id) REFERENCES musteriler(musteri_id),
    FOREIGN KEY (cikis_sube_id) REFERENCES subeler(sube_id)
) ENGINE=InnoDB;

-- 5. Kargo Hareketleri Tablosu
CREATE TABLE IF NOT EXISTS kargohareketleri (
    hareket_id INT AUTO_INCREMENT PRIMARY KEY,
    kargo_id INT,
    durum_aciklama VARCHAR(255),
    islem_tarihi TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (kargo_id) REFERENCES kargolar(id) ON DELETE CASCADE
```

```
) ENGINE=InnoDB;

-- 6. Sistem Logları Tablosu
CREATE TABLE IF NOT EXISTS sistemloglari (
    log_id INT AUTO_INCREMENT PRIMARY KEY,
    islem_tipi VARCHAR(50),
    mesaj TEXT,
    tarih TIMESTAMP DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB;
```

2.3. Normalizasyon ve Kısıtlamalar

Veri tabanı **3. Normal Form (3NF)** kurallarına uygun tasarlanmıştır:

- **Tekrarsızlık:** Müşteri ve şube bilgileri ana tabloda tekrar edilmemiş, ayrı tablolara (lookup tables) taşınarak ID ile referans verilmiştir.
- **Constraints (Kısıtlamalar):**
  - `PRIMARY KEY`: Tüm tablolarda benzersiz kayıt kimliği.
  - `FOREIGN KEY`: Tablolar arası veri bütünlüğü.
  - `UNIQUE`: Takip numarası ve TC Kimlik gibi alanların eşsizliği.
  - `DEFAULT`: Durum girilmezse otomatik 'Hazırlanıyor' atanması.

3. SQL UYGULAMALARI (Bölüm B)

3.1. CRUD İşlemleri

Temel veri manipülasyonu işlemleri aşağıdaki gibidir:

SQL
```
-- EKLEME (INSERT)
INSERT INTO kargolar (takip_no, cikis_sube_id, durum, fiyat) VALUES
('BGC999TR', 1, 'Hazırlanıyor', 120.50);

-- GÜNCELLEME (UPDATE)
UPDATE kargolar SET durum = 'Yolda' WHERE takip_no = 'BGC999TR';

-- SİLME (DELETE)
DELETE FROM kargolar WHERE takip_no = 'BGC999TR';
```

3.2. Karmaşık Sorgular (10 Adet)

Raporlama ve analiz için hazırlanan ileri seviye sorgular:

1. Kargo Detaylarını İsimlerle Listeleme (INNER JOIN):

SQL
```
SELECT k.takip_no, k.durum, k.fiyat, s.sube_adi, m1.ad_soyad AS
gonderen, m2.ad_soyad AS alici
FROM kargolar k
JOIN subeler s ON k.cikis_sube_id = s.sube_id
JOIN musteriler m1 ON k.gonderici_id = m1.musteri_id
JOIN musteriler m2 ON k.alici_id = m2.musteri_id;
```

2. Şubelerin Kargo Sayılarını Sıralama (GROUP BY & ORDER BY):

SQL
```
SELECT s.sube_adi, COUNT(k.id) AS toplam_kargo
FROM subeler s
LEFT JOIN kargolar k ON s.sube_id = k.cikis_sube_id
GROUP BY s.sube_adi
ORDER BY toplam_kargo DESC;
```

3. Ortalamanın Üzerinde Fiyata Sahip Kargolar (SUBQUERY):

SQL
```
SELECT takip_no, fiyat FROM kargolar WHERE fiyat > (SELECT
AVG(fiyat) FROM kargolar);
```

4. Yoğun Şubeleri Tespit Etme (HAVING - En az 5 kargo):

SQL
```
SELECT s.sube_adi, COUNT(k.id) AS kargo_sayisi
FROM subeler s
JOIN kargolar k ON s.sube_id = k.cikis_sube_id
GROUP BY s.sube_adi
HAVING kargo_sayisi >= 5;
```

5. Kargo Hareket Geçmişi (LEFT JOIN):

```
SQL
SELECT k.takip_no, h.durum_aciklama, h.islem_tarihi
FROM kargolar k
LEFT JOIN kargohareketleri h ON k.id = h.kargo_id
WHERE k.takip_no = 'BGC123TR'
ORDER BY h.islem_tarihi DESC;
```

6. Son 7 Günde Oluşturulan Kargolar (DATE Functions):

```
SQL
SELECT * FROM kargolar WHERE olusturma_tarihi >= DATE_SUB(NOW(),
INTERVAL 7 DAY);
```

7. Hiç Kargo Göndermemiş Müşteriler (NOT IN):

```
SQL
SELECT ad_soyad FROM musteriler WHERE musteri_id NOT IN (SELECT
DISTINCT gonderici_id FROM kargolar);
```

8. Şube Bazlı Toplam Ciro (SUM):

```
SQL
SELECT s.sube_adi, SUM(k.fiyat) AS toplam_ciro
FROM subeler s
JOIN kargolar k ON s.sube_id = k.cikis_sube_id
GROUP BY s.sube_adi;
```

9. En Pahalı ve En Ucuz Kargo (MAX / MIN):

```
SQL
SELECT MAX(fiyat) AS en_yuksek_ucret, MIN(fiyat) AS en_dusuk_ucret
FROM kargolar;
```

10. Teslim Edilmeyen Kargolar (Complex WHERE):

```
SQL
SELECT k.takip_no, k.durum, s.il
FROM kargolar k
JOIN subeler s ON k.cikis_sube_id = s.sube_id
```

```
WHERE k.durum != 'Teslim Edildi' AND k.durum != 'İptal';
```

3.3. Stored Procedures (Saklı Yordamlar - 3 Adet)

İş süreçlerini otomatize etmek için oluşturulan prosedürler:

1. Yeni Kargo Girişi (SP_YeniKargo):

```
SQL
CREATE PROCEDURE SP_YeniKargo(IN p_takip VARCHAR(20), IN p_gonderici
INT, IN p_sube INT)
BEGIN
    INSERT INTO kargolar (takip_no, gonderici_id, cikis_sube_id,
durum)
    VALUES (p_takip, p_gonderici, p_sube, 'Sipariş Alındı');
END;
```

2. Durum Güncelleme ve Geçmişe Ekleme (SP_DurumGuncelle):

```
SQL
CREATE PROCEDURE SP_DurumGuncelle(IN p_takip VARCHAR(20), IN
p_yeni_durum VARCHAR(50))
BEGIN
    DECLARE v_id INT;
    SELECT id INTO v_id FROM kargolar WHERE takip_no = p_takip;
    UPDATE kargolar SET durum = p_yeni_durum WHERE id = v_id;
    INSERT INTO kargohareketleri (kargo_id, durum_aciklama)
    VALUES (v_id, CONCAT('Durum güncellendi: ', p_yeni_durum));
END;
```

3. Müşteri Kargo Sayısı (SP_MusteriKargoSayisi):

```
SQL
CREATE PROCEDURE SP_MusteriKargoSayisi(IN p_musteri_id INT, OUT
p_sayi INT)
BEGIN
    SELECT COUNT(*) INTO p_sayi FROM kargolar WHERE gonderici_id =
p_musteri_id;
END;
```

3.4. Views (Görünümler - 3 Adet)

Kullanıcı dostu sanal tablolar:

1. Müşteri Odaklı Liste (V_KargoDetay):

SQL
```
CREATE VIEW V_KargoDetay AS
SELECT k.takip_no, k.durum, k.fiyat, g.ad_soyad AS gonderen
FROM kargolar k
JOIN musteriler g ON k.gonderici_id = g.musteri_id;
```

2. Şube Performans Raporu (V_SubePerformans):

SQL
```
CREATE VIEW V_SubePerformans AS
SELECT s.sube_adi, COUNT(k.id) AS islem_hacmi, SUM(k.fiyat) AS ciro
FROM subeler s
JOIN kargolar k ON s.sube_id = k.cikis_sube_id
GROUP BY s.sube_adi;
```

3. Aktif Kargolar (V_AktifKargolar):

SQL
```
CREATE VIEW V_AktifKargolar AS
SELECT takip_no, durum, olusturma_tarihi FROM kargolar
WHERE durum NOT IN ('Teslim Edildi', 'İptal');
```

3.5. Transaction (İşlemler - 3 Adet)

Veri bütünlüğü senaryoları:

1. Güvenli Silme: Hata olursa işlemi geri alır.

SQL
```
START TRANSACTION;
    DELETE FROM kargohareketleri WHERE kargo_id = 10;
    DELETE FROM kargolar WHERE id = 10;
COMMIT;
```

2. Kargo İptali: Durumu günceller ve log atar.

```
SQL
START TRANSACTION;
    UPDATE kargolar SET durum = 'İptal' WHERE takip_no = 'BGC123TR';
    INSERT INTO sistemloglari (islem_tipi, mesaj) VALUES ('IPTAL',
'BGC123TR iptal edildi.');
COMMIT;
```

3. Toplu Fiyat Güncellemesi:

```
SQL
START TRANSACTION;
    UPDATE kargolar SET fiyat = fiyat * 1.10 WHERE durum =
'Hazırlanıyor';
    -- Hata kontrolü yapılırsa ROLLBACK kullanılabilir
COMMIT;
```

4. VERİ BÜTÜNLÜĞÜ VE LOGLAMA (Bölüm C)

Trigger (Tetikleyici) Mekanizması

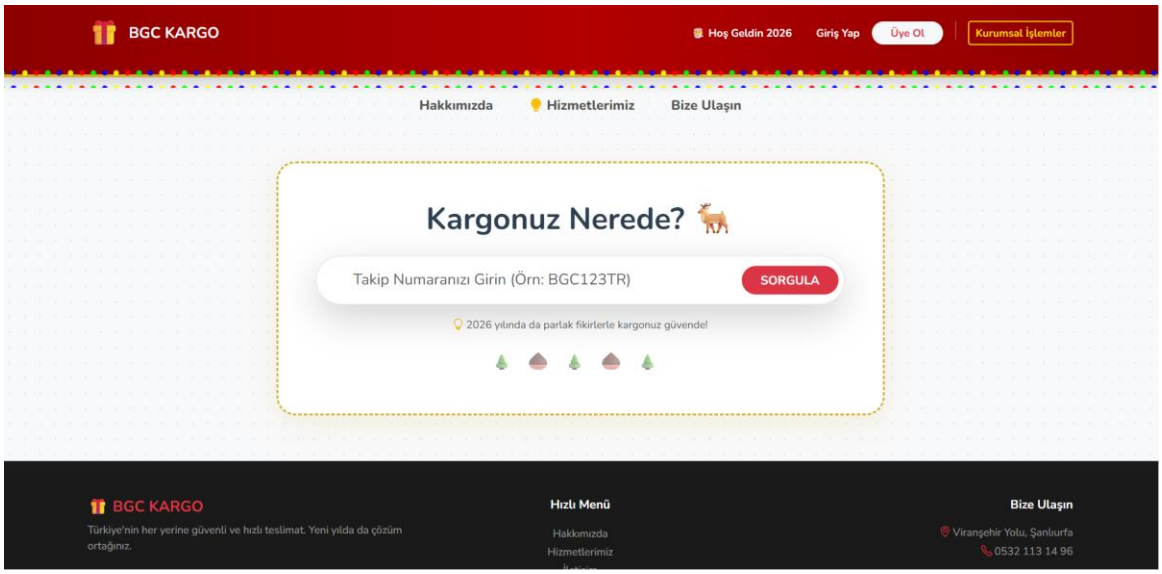
Sistemdeki veri değişikliklerini izlemek için `TRG_DurumDegisimi` tetikleyicisi yazılmıştır. Bir kargonun durumu değiştiğinde (Örn: Hazırlanıyor -> Yolda), sistem bunu otomatik olarak algılar ve `sistemloglari` tablosuna kayıt atar.

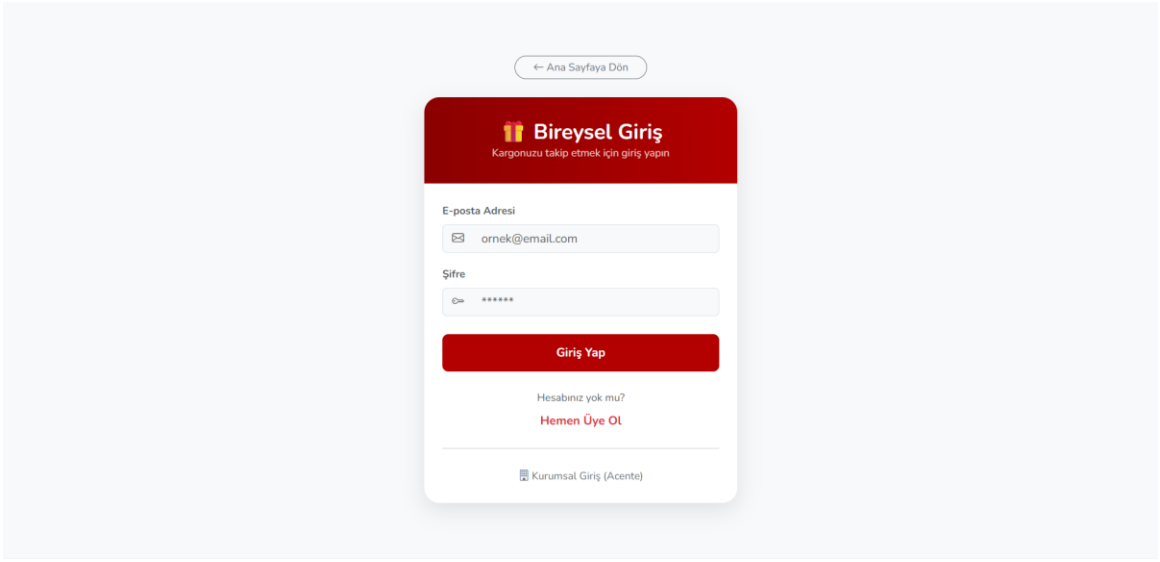
```
SQL
CREATE TRIGGER TRG_DurumDegisimi
AFTER UPDATE ON kargolar
FOR EACH ROW
BEGIN
    INSERT INTO sistemloglari (islem_tipi, mesaj, tarih)
    VALUES ('GUNCELLEME', CONCAT(OLD.takip_no, ' durumu değişti: ',
OLD.durum, ' -> ', NEW.durum), NOW());
END;
```

5. ARAYÜZ VE SUNUM (Bölüm D)

Proje, son kullanıcıların kargo takibi yapabilmesi için modern bir Web Arayüzü ile desteklenmiştir. Tasarımda "BGC Kargo 2026" konsepti ve kurumsal renkler kullanılmıştır.

Arayüz Özellikleri:

- **Sorgulama Ekranı:** Kullanıcılar takip numarasını girerek `kargolar` ve `kargohareketleri` tablolarından `JOIN` ile çekilen verileri görüntüleyebilir.
- 


← Ana Sayfaya Dön
Bireysel Giriş
Kargonuzu takip etmek için giriş yapın
E-posta Adresi
ornek@email.com
Şifre
******
Giriş Yap
Hesabınız yok mu?
Hemen Üye Ol
Kurumsal Giriş (Acente)


Kurumsal İş Ortağımız mısınız?
Lütfen yapmak istediğiniz işlemi seçiniz.
Kurumsal Giriş
Mevcut acente veya kurum hesabınızla sisteme erişin.
Kurumsal Kayıt
BGC Kargo ailesine katılmak için kurumsal başvurunuzu yapın.
Ana Sayfaya Dön