
# Veritabanı Yönetim Sistemleri

## OTOBÜS OTOMASYONU PROJESI

Muhammet Barış Yılmaz

muhammet.yilmaz38@ogr.sakarya.edu.tr

2C GRUBU / G201210057

# Tanıtım

Projenin isminden de anlaşılabileceği üzere otobüs-bilet alım satımı yapan bir firmanın kullanabileceği uygulamadır. Uygulama ofiste çalışan bir bilet satıcısının yapabileceği tüm işlemler düşünülerek tasarlanmıştır. Projede dış görünüme önem verilmemiş, ağırlıkla projede istenen bazda işler yapılmış, istenen yapıların hepsi tamamlanmıştır.

# İş Tanımı

Her müşterinin adı,soyadı,email'i,telefonu,doğum tarihi, cinsiyeti istenir.

Her çalışanın adı,soyadı,email'i,telefonu,ev adresi,hangi şubede çalıştığı, mesleği istenir.

Bir otobüsün bir adet markası olabilir.

Bir otobüs birden çok sefer düzenleyebilir veya hiç sefer düzenlemeyebilir.

Bir otobüsün bir veya birden çok koltuğu olabilir.

Bir müşteri bir adet bilet alabilir.

Bir bilet bir koltuğa aittir.

Bir müşterinin bir tane cinsiyeti olur.

Bir çalışan bir şubeye aittir.

Bir şehirde sıfır adet şube de olabilir, çok sayıda şube de olabilir.

## İlişkisel Şema

Otobusler(**otobusID: Integer**, plaka: Varchar(8), koltukSayisi: Integer, markaID: Integer)

Markalar(**markaID: Integer**, markaAdi: Text, markaModeli: Text)

Seferler(**seferID: Integer**, kalkisZamani: Date, inisZamani: Date, ucret: Integer, otobusID: Integer)

Koltuklar(**otobusKoltukID: Integer**, bosKoltuk: Integer, otobusID: Integer)

Biletler(**yolcuID: Integer**, musteriID: Integer, seferID: Integer, otobusKoltukID: Integer)

Musteriler(**musteriID: Integer**, ad: Text, soyad: Text, email: Varchar(8), telefon: Varchar(12), dogumTarihi: Date, cinsiyetID: Integer)

Calisanlar(**personelID: Integer**, ad: Text, soyad: Text, email: Varchar(8), telefon: Varchar(12), adres: Varchar(255), subeID: Integer, calisanlarTurID: Integer)

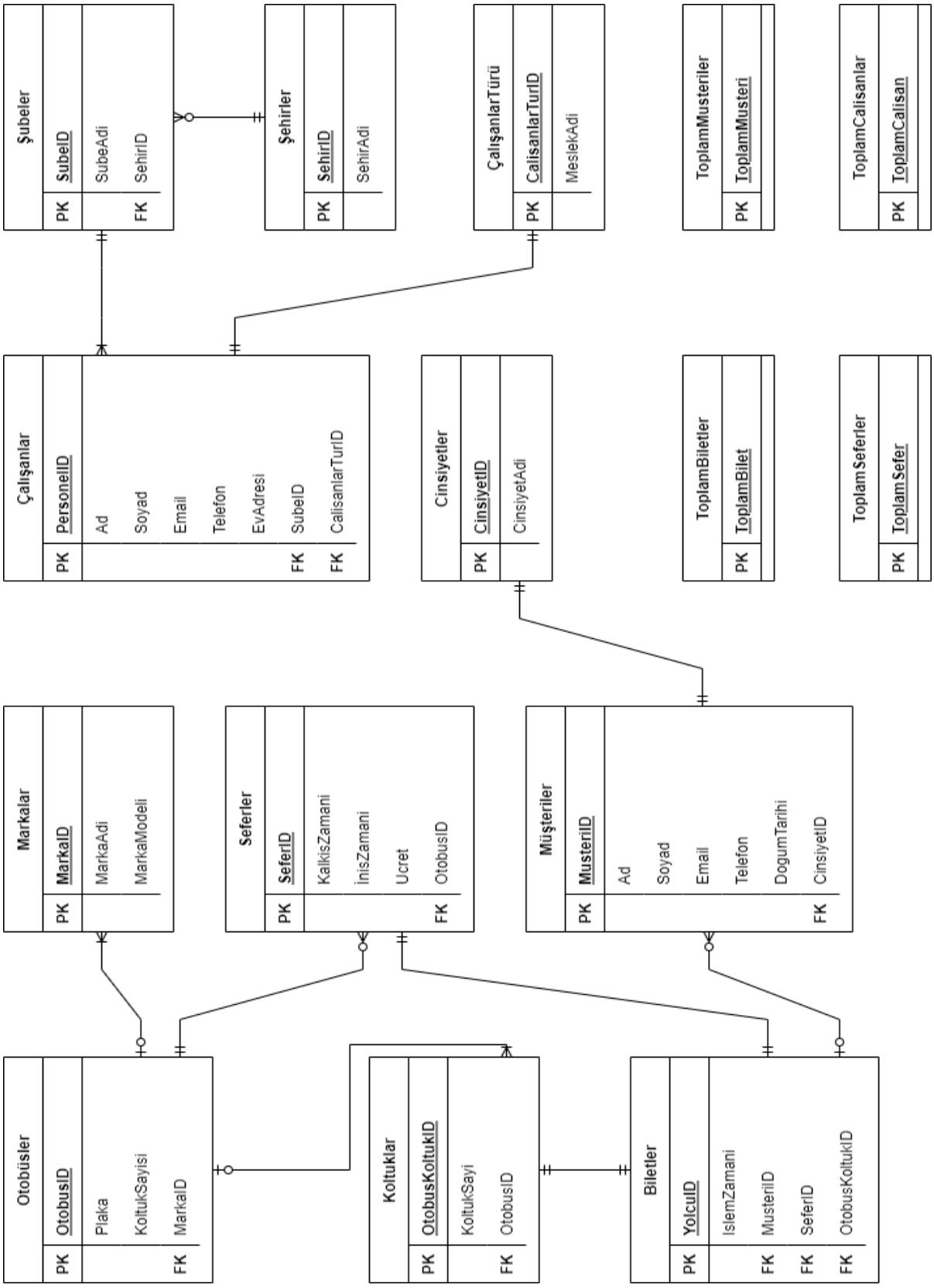
Cinsiyetler(**cinsiyetID: Integer**, cinsiyetAdi: Text)

CalisanlarTuru(**calisanlarTurID: Integer**, meslekAdi: Text)

Sehirler(**sehirID: Integer**, sehirAdi: Text)

Subeler(**subeID: Integer**, subeAdi: Text, sehirID: Integer)

# Varlık Bağıntı Modeli



# SQL İfadeleri

```
class Biletler:
    def biletEkle():
        Musteriler.MusteriGoruntule()
        musteri = input("Bilet almak isteyen MüşteriID'sini giriniz: ")
        Fonksiyonlar.seferGetir()
        sefer = input("SeferID seçiniz: ")
        Fonksiyonlar.koltukGetir()
        otobus = input("Uygun olan OtobusKoltukID seçiniz: ")

        sql = "INSERT INTO Biletler (musteriid,seferid,otobuskoltukid) VALUES(%s,%s,%s)"
        values = (musteri,sefer,otobus)
        cursor.execute(sql,values)
        print("Bilet başarıyla eklendi.")
    def biletGoruntule():
        query_Biletler_secelt = """
        SELECT yolcuid, musteriler.name,musteriler.surname,seferler.kalkiszamani,seferler.iniszamani FROM Biletler
        JOIN musteriler ON biletler.musteriid = musteriler.musteriid
        JOIN seferler ON biletler.seferid = seferler.seferid
        """
        cursor.execute(query_Biletler_secelt)
        biletler = cursor.fetchall()
        tablo = PrettyTable(['BiletID','Müşteri Adı','Müşteri Soyadı','Kalkış Zamanı','İniş Zamanı'])
        for bilet in biletler:
            tablo.add_row(bilet)
        print(tablo)
    def biletGuncelle():
        query_Biletler_select = """
        SELECT musteriler.musteriid,musteriler.name, musteriler.surname,biletler.yolcuid FROM Musteriler
        JOIN biletler ON musteriler.musteriid = biletler.musteriid
        """
        cursor.execute(query_Biletler_select)
        biletler = cursor.fetchall()
        tablo = PrettyTable(['MüşteriID','Müşteri Adı', 'Müşteri Soyadı', 'Müşteri YolcuID'])
        for bilet in biletler:
            tablo.add_row(bilet)
        print(tablo)

        guncelleSecim = int(input("Düzenlemek istediğiniz MusteriID giriniz: "))

        for bilet in biletler:
            if(bilet[0] == guncelleSecim):
                Musteriler.MusteriGoruntule()
                musteri = input("Bilet almak isteyen MüşteriID'sini giriniz: ")
                Fonksiyonlar.seferGetir()
                sefer = input("SeferID seçiniz: ")
                Fonksiyonlar.koltukGetir()
                otobus = input("Uygun olan OtobusKoltukID seçiniz: ")
                sql = "UPDATE Biletler SET musteriid=%s, seferid=%s, otobuskoltukid=%s WHERE yolcuid=%s"
                values = (guncelleSecim,sefer,otobus,bilet[3])
                cursor.execute(sql,values)
                count = cursor.rowcount
                print(f"\n{count} kayıt başarıyla güncellenmiştir.")
    def biletSil():
        Biletler.biletGoruntule()
        silSecim = input("Silmek istediğiniz biletin yolcu ID'sini giriniz: ")
        query_Bilet_delete = "DELETE FROM Biletler WHERE yolcuid={0}".format(silSecim)
        cursor.execute(query_Bilet_delete)

        print("Seçilen bilet başarıyla silinmiştir.")
    def seferEkle():
        kalkiszamani = input("Kalkış Zamanını Giriniz(GG/AA/YYYY Şeklinde): ")
        iniszamani = input("İniş Zamanını Giriniz(GG/AA/YYYY Şeklinde): ")
        ucret = input("Ücreti giriniz: ")
        Otobusler.OtobusGoruntule()
        otobus = input("Görevli Otobus ID'sini giriniz: ")

        sql = """
        INSERT INTO Seferler (kalkiszamani,iniszamani,ucret,otobusid) VALUES (%s,%s,%s,%s)
        """
        values = (kalkiszamani,iniszamani,ucret,otobus)
        cursor.execute(sql,values)
        print("Sefer başarıyla eklendi.")
```

```
class Calisanlar:
    def calisanGoruntule():
        query_Calisanlar_select = """
        SELECT calisanlar.personelid,calisanlar.name,calisanlar.surname,subeler.subeadi,calisanlarturu.meslekadi FROM calisanlar
        JOIN subeler ON calisanlar.subeid = subeler.subeid
        JOIN calisanlarturu ON calisanlar.calisanlarturid = calisanlarturu.calisanlarturid
        """
        cursor.execute(query_Calisanlar_select)
        calisanlar = cursor.fetchall()
        tablo = PrettyTable(['PersonelID','İsim','Soyisim','Şube İsmi','Meslek'])
        for calisan in calisanlar:
            tablo.add_row(calisan)
        print(tablo)
    def calisanEkle():
        calisanAd = input("Çalışan ismini giriniz: ")
        calisanSoyad = input("Çalışan soyadını giriniz: ")
        calisanEmail = input("Çalışan e-mailini giriniz (kamilkoc@koc.com gibi): ")
        calisanTelefon = input("Çalışan telefon numarasını giriniz (90123456789 gibi): ")
        calisanAdres = input("Çalışan adresini yazınız: ")
        Fonksiyonlar.subeGetir()
        calisanSube = input("Çalışan kişinin Şube ID'sini seçiniz: ")
        Fonksiyonlar.meslekGetir()
        calisanTur = input("Çalışanın meslek ID'sini seçiniz: ")

        sql = "INSERT INTO Calisanlar (name,surname,email,phoneNumber,adress,subeid,calisanlarturID) VALUES(%s,%s,%s,%s,%s,%s,%s)"
        values = (calisanAd,calisanSoyad,calisanEmail,calisanTelefon,calisanAdres,calisanSube,calisanTur)

        cursor.execute(sql,values)
        print(f"\n{calisanAd} {calisanSoyad} adlı çalışan başarıyla eklendi.")
    def calisanGuncelle():
        Calisanlar.calisanGoruntule()
        print("Düzenlemek istediğiniz çalışanın ID'sini giriniz.")
        guncelleSecim = int(input("Lütfen ID giriniz: "))

        query_Calisanlar_select = """
        SELECT * FROM Calisanlar
        """
        cursor.execute(query_Calisanlar_select)
        calisanlar = cursor.fetchall()

        for calisan in calisanlar:
            if(calisan[0] == guncelleSecim):
                calisanAd = input("Çalışan ismini giriniz: ")
                calisanSoyad = input("Çalışan soyadını giriniz: ")
                calisanEmail = input("Çalışan e-mailini giriniz (kamilkoc@koc.com gibi): ")
                calisanTelefon = input("Çalışan telefon numarasını giriniz (90123456789 gibi): ")
                calisanAdres = input("Çalışan adresini yazınız: ")
                Fonksiyonlar.subeGetir()
                calisanSube = input("Çalışan kişinin Şube ID'sini seçiniz: ")
                Fonksiyonlar.meslekGetir()
                calisanTur = input("Çalışanın meslek ID'sini seçiniz: ")

                sql = """
                UPDATE Calisanlar SET name=%s,surname=%s,email=%s,phonenumber=%s,adress=%s,subeid=%s,calisanlarturid=%s WHERE personelid=%s
                """
                cursor.execute(sql,(calisanAd,calisanSoyad,calisanEmail,calisanTelefon,calisanAdres,calisanSube,calisanTur,guncelleSecim))

                count = cursor.rowcount
                print(f"\n{count} kayıt başarıyla güncellenmiştir.")
    def calisanSil():
        Calisanlar.calisanGoruntule()

        print("Silmek istediğiniz çalışanın ID'sini giriniz.")
        silSecim = input("Lütfen ID giriniz: ")
        query_Calisanlar_delete = """
        DELETE FROM Calisanlar WHERE personelid=%s
        """
        cursor.execute(query_Calisanlar_delete,silSecim)

        count = cursor.rowcount
        print(f"\n{count} kayıt basarıyla silinmistir.")
class Musteriler:
    def MusteriEkle():
        musteriAd = input("Müşteri adını giriniz: ")
        musteriSoyad = input("Müşteri soyadını giriniz: ")
        musteriEmail = input("Müşteri e-mailini giriniz (kamilkoc@koc.com gibi): ")
        musteriTelefon = input("Müşteri telefon numarasını giriniz (90123456789 gibi): ")
        Fonksiyonlar.cinsiyetGetir()
        musteriCinsiyet = input("Müşterinin Cinsiyet ID'sini seçiniz: ")
        musteriDogumTarihi = input("Müşteri doğum tarihini giriniz (01-01-1990 gibi): ")

        sql = "INSERT INTO Musteriler (name,surname,email,phoneNumber,dateBirth,cinsiyetID) VALUES(%s,%s,%s,%s,%s,%s)"
        values = (musteriAd,musteriSoyad,musteriEmail,musteriTelefon,musteriDogumTarihi,musteriCinsiyet)

        cursor.execute(sql,values)
        print(f"\n{musteriAd} {musteriSoyad} adlı müşteri başarıyla eklendi.")
    def MusteriGoruntule():
        query_Musteri_select = """
        SELECT musteriler.musteriid,musteriler.name,musteriler.surname,cinsiyet.cinsiyetAdi FROM musteriler
        JOIN cinsiyet ON musteriler.cinsiyetid = cinsiyet.cinsiyetID
        """
        cursor.execute(query_Musteri_select)
        musteriler = cursor.fetchall()

        tablo = PrettyTable(['MüşteriID','İsim','Soyisim','Cinsiyet'])
        for musteri in musteriler:
            tablo.add_row(musteri)
        print(tablo)
```

```
def MusteriGuncelle():
    Musteriler.MusteriGoruntule()

    print("Düzenlemek istediğiniz müşterinin ID'sini giriniz.")
    guncelleSecim = int(input("Lütfen ID giriniz: "))

    query_Musteri_select = """
    SELECT musteriler.musteriid,musteriler.name,musteriler.surname,cinsiyet.cinsiyetAdi FROM musteriler JOIN cinsiyet ON musteriler.cinsiyetid = cinsiyet.cinsiyetID
    """
    cursor.execute(query_Musteri_select)
    musteriler = cursor.fetchall()

    for musteri in musteriler:
        if(musteri[0] == guncelleSecim):
            musteriAd = input("Müşteri adını giriniz: ")
            musteriSoyad = input("Müşteri soyadını giriniz: ")
            musteriEmail = input("Müşteri e-mailini giriniz (kamilkoc@koc.com gibi): ")
            musteriTelefon = input("Müşteri telefon numarasını giriniz (90123456789 gibi): ")
            Fonksiyonlar.cinsiyetGetir()
            musteriCinsiyet = input("Müşterinin Cinsiyet ID'sini seçin: ")
            musteriDogumTarihi = input("Müşteri doğum tarihini giriniz (01-01-1990 gibi): ")

            query_Musteri_update = """
            UPDATE Musteriler SET name=%s, surname=%s, email=%s, phoneNumber=%s, dateBirth=%s, cinsiyetID=%s WHERE musteriid=%s
            """
            cursor.execute(query_Musteri_update,(musteriAd,musteriSoyad,musteriEmail,musteriTelefon,musteriDogumTarihi,musteriCinsiyet,guncelleSecim))

            count = cursor.rowcount
            print(f"\n{count} kayıt başarıyla güncellenmiştir.")
def MusteriSil():
    Musteriler.MusteriGoruntule()

    print("Silmek istediğiniz müşterinin ID'sini giriniz.")
    silSecim = input("Lütfen ID giriniz: ")
    query_Musteri_delete = """
    DELETE FROM Musteriler WHERE musteriid={0}
    """.format(silSecim)
    cursor.execute(query_Musteri_delete,silSecim)

    count = cursor.rowcount
    print(f"\n{count} kayıt başarıyla silinmiştir.")
def MusteriSayisi():
    sql = """
    SELECT * FROM toplammusteriler
    """
    cursor.execute(sql)
    musteriSayisi = cursor.fetchone()
    print("\nTOPLAM MÜŞTERİ SAYISI: ", musteriSayisi[0])
```

```
def OtobusEkle():
    plaka = input("Otobüs plakasını giriniz (34GGH445 gibi): ")
    Fonksiyonlar.markaGetir()
    markaSecim = input("Otobüsün Marka ID'sini seçiniz: ")

    sql = "INSERT INTO Otobusler (plaka,markaid) VALUES(%s,%s)"
    values = (plaka,markaSecim)
    cursor.execute(sql,values)

    sql = "SELECT otobusid,plaka FROM Otobusler"
    cursor.execute(sql)
    otobusler = cursor.fetchall()
    tablo = PrettyTable(['OtobusID','Plaka'])
    for otobus in otobusler:
        tablo.add_row(otobus)
    print(tablo)
    otobusid = input("Yeni eklenen otobusun ID'sini giriniz: ")
    koltuk = input("Koltuk sayısını giriniz: ")
    sql = "INSERT INTO Koltuklar (boskoltuk,otobusid) VALUES (%s,%s)"
    values = (koltuk,otobusid)
    cursor.execute(sql,values)


    print(f"\n{plaka} plakalı otobüs başarıyla eklendi.")
def OtobusGoruntule():
    query_Otubus_select = """
    SELECT otobusler.otobusid, otobusler.plaka, markalar.markaAdi, markalar.markamodeli FROM Otobusler
    JOIN markalar ON otobusler.markaid = markalar.markaid
    """
    cursor.execute(query_Otubus_select)
    otobusler = cursor.fetchall()

    tablo = PrettyTable(['OtobüsID','Plaka','Marka Adı','Modeli'])
    for otobus in otobusler:
        tablo.add_row(otobus)
    print(tablo)
def OtobusGuncelle():
    Otobusler.OtobusGoruntule()

    guncelleSecim = int(input("Düzenlemek istediğiniz OtobusID giriniz: "))

    query_Otobus_select = """
    SELECT * FROM Otobusler
    """
    cursor.execute(query_Otobus_select)
    otobusler = cursor.fetchall()

    for otobus in otobusler:
        if(otobus[0] == guncelleSecim):
            plaka = input("Otobüs plakasını giriniz (34GGH445 gibi): ")
            Fonksiyonlar.markaGetir()
            markaid = input("Marka ID seçiniz: ")
            query_Otobus_update = """
            UPDATE Otobusler SET plaka=%s, markaid=%s WHERE otobusid=%s
            """
            cursor.execute(query_Otobus_update,(plaka,markaid,guncelleSecim))
            count = cursor.rowcount
            print(f"\n{count} kayıt başarıyla güncellenmiştir.")
def OtobusSil():
    Otobusler.OtobusGoruntule()
    silSecim = input("Silmek istediğiniz otobüsün ID'sini giriniz: ")
    query_Otobus_delete = """
    SET session_replication_role = 'replica';
    DELETE FROM Otobusler WHERE otobusid=%s;
    DELETE FROM Seferler WHERE otobusid=%s;
    DELETE FROM Koltuklar WHERE otobusid=%s;
    SET session_replication_role = 'origin';
    """
    cursor.execute(query_Otobus_delete,(silSecim,silSecim,silSecim))

    print("Seçilen otobüs başarıyla silinmiştir.")
```

## Saklı Yordam

```
CREATE FUNCTION public.calisangetir() RETURNS TABLE(id integer, ad text, soyad text, meslek text)
    LANGUAGE plpgsql
    AS $$
Begin
    Return Query
    Select
    calisanlar.personelid,
    calisanlar.name,
    calisanlar.surname,
    calisanlarturu.meslekadi
 From
    calisanlar
 JOIN calisanlarturu ON calisanlar.calisanlarturid = calisanlarturu.calisanlarturid;
End;
$$;
```

```
CREATE FUNCTION public.cinsiyetgetir() RETURNS TABLE(id integer, ad text)
    LANGUAGE plpgsql
    AS $$
Begin
    Return Query
    Select
    cinsiyetid,
    cinsiyetadi
 From
    cinsiyet;
End;
$$;


ALTER FUNCTION public.cinsiyetgetir() OWNER TO postgres;
```

```
CREATE FUNCTION public.markagetir() RETURNS TABLE(id integer, ad text, model text)
    LANGUAGE plpgsql
    AS $$
Begin
    Return Query
    Select
    markaid,
    markaadi,
    markamodeli
 From
    markalar;
End;
$$;


ALTER FUNCTION public.markagetir() OWNER TO postgres;
```

```
CREATE FUNCTION public.meslekgetir() RETURNS TABLE(id integer, meslek text)
    LANGUAGE plpgsql
    AS $$
Begin
    Return Query
    Select
    calisanlarturid,
    meslekadi
 From
    calisanlarturu;
End;
$$;


ALTER FUNCTION public.meslekgetir() OWNER TO postgres;
```

```
CREATE FUNCTION public.sefergetir() RETURNS TABLE(id integer, kalkis date, inis date, tutar integer, otobus integer)
    LANGUAGE plpgsql
    AS $$
begin
    Return Query
    Select
    seferid,
    kalkiszamani,
    iniszamani,
    ucret,
    otobusid
 from
    seferler;
End;
$$;


ALTER FUNCTION public.sefergetir() OWNER TO postgres;
```

```
CREATE FUNCTION public.subegetir() RETURNS TABLE(id integer, subead text)
    LANGUAGE plpgsql
    AS $$
Begin
    Return Query
    Select
    subeid,
    subeadi
 From
    subeler;
End;
$$;


ALTER FUNCTION public.subegetir() OWNER TO postgres;
```

## Trigger

```
CREATE FUNCTION public.biletarttir() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
begin
update toplambiletler set toplambilet=toplambilet+1;
return new;
end;
$$;


ALTER FUNCTION public.biletarttir() OWNER TO postgres;
```

```
CREATE FUNCTION public.biletazalt() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
begin
update toplambiletler set toplambilet = toplambilet -1;
return new;
end;
$$;


ALTER FUNCTION public.biletazalt() OWNER TO postgres;
```

```
CREATE FUNCTION public.biletkoltuk() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
BEGIN
UPDATE koltuklar SET boskoltuk = boskoltuk - 1;
RETURN new;
END;
$$;


ALTER FUNCTION public.biletkoltuk() OWNER TO postgres;
```

```
CREATE FUNCTION public.calisanarttir() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
begin
update toplamcalisanlar set toplamcalisan = toplamcalisan + 1;
return new;
end;
$$;


ALTER FUNCTION public.calisanarttir() OWNER TO postgres;
```

```
CREATE FUNCTION public.calisanazalt() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
begin
update toplamcalisanlar set toplamcalisan = toplamcalisan -1;
return new;
end;
$$;


ALTER FUNCTION public.calisanazalt() OWNER TO postgres;
```

```
CREATE FUNCTION public.musteriarttir() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
begin
update toplammusteriler set toplammusteri = toplammusteri + 1;
return new;
end;
$$;


ALTER FUNCTION public.musteriarttir() OWNER TO postgres;
```

```
CREATE FUNCTION public.musteriazalt() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
begin
update toplammusteriler set toplammusteri = toplammusteri -1;
return new;
end;
$$;


ALTER FUNCTION public.musteriazalt() OWNER TO postgres;
```

```
CREATE FUNCTION public.seferarttir() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
begin
update toplamseferler set toplamsefer = toplamsefer + 1;
return new;
end;
$$;


ALTER FUNCTION public.seferarttir() OWNER TO postgres;
```

```
CREATE FUNCTION public.seferazalt() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
begin
update toplamseferler set toplamsefer = toplamsefer -1;
return new;
end;
$$;


ALTER FUNCTION public.seferazalt() OWNER TO postgres;
```

## Uygulamaya Ait Ekran Görüntüleri

```
------------Bilet Menüsü------------
[1]Biletleri Göster
[2]Bilet Ekle
[3]Bilet Güncelle
[4]Bilet Sil
[5]Sefer Ekle
[6]Sefer Sayısını Gör
[7]Bilet Sayısını Gör
[8]Anamenüye Dön
Lütfen seçenek giriniz: 1
+---------+------------+----------------+---------------+-------------+
| BiletID | Müşteri Adı | Müşteri Soyadı | Kalkış Zamanı | İniş Zamanı |
+---------+------------+----------------+---------------+-------------+
|    65   |    Barış    |     Yılmaz     |   2021-12-16  |  2021-12-17 |
|    66   |    Barış    |     Yılmaz     |   2021-12-16  |  2021-12-17 |
|    70   |    Deniz    |     Güler      |   2021-12-16  |  2021-12-17 |
+---------+------------+----------------+---------------+-------------+
```

```
##########################################
Otobüs Firması Admin Paneline Hoşgeldiniz.

[1]Müşteri Menüsü
[2]Bilet Menüsü
[3]Otobüs Menüsü
[4]Şube Menüsü
[5]Çıkış

Lütfen seçenek giriniz:
```

```
------------Müşteri Menüsü------------
[1]Müşterileri Göster
[2]Müşteri Ekle
[3]Müşteri Güncelle
[4]Müşteri Sil
[5]Müşteri Sayısını Gör
[6]Anamenüye Dön
Lütfen seçenek giriniz:
```

```
------------Müşteri Menüsü------------
[1]Müşterileri Göster
[2]Müşteri Ekle
[3]Müşteri Güncelle
[4]Müşteri Sil
[5]Müşteri Sayısını Gör
[6]Anamenüye Dön
Lütfen seçenek giriniz: 1
+------------+--------+---------+---------------+
| MüşteriID |  İsim  | Soyisim |    Cinsiyet   |
+------------+--------+---------+---------------+
|     1      | Barış  |  Yılmaz |     Erkek     |
|     2      | Cemil  |  Öztürk |     Erkek     |
|     4      | İlayda |  Bekdaş |     Kadın     |
|     5      | Merve  |  Aydın  |     Kadın     |
|     8      | Erhan  |  Korkut |     Erkek     |
|     3      | Deniz  |  Güler  |     Kadın     |
|     9      | Özlem  |  Pelit  | Belirtilmemiş |
|     12     |  Akif  |   Kaya  | Belirtilmemiş |
+------------+--------+---------+---------------+
```

```
------------Otobüs Menüsü------------
[1]Otobüsleri Göster
[2]Otobüs Ekle
[3]Otobüs Güncelle
[4]Otobüs Sil
[5]Anamenüye Dön
Lütfen seçenek giriniz: 1
+-----------+----------+---------------+---------+
| OtobüsID |   Plaka   |   Marka Adı   |  Modeli |
+-----------+----------+---------------+---------+
|    12     | 34PAS990 | Mercedes-Benz | Travego |
+-----------+----------+---------------+---------+
```

```
------------Şube Menüsü------------
[1]Çalışanları Göster
[2]Çalışan Ekle
[3]Çalışan Güncelle
[4]Çalışan Sil
[5]Çalışan Sayısını Gör
[6]Anamenüye Dön
Lütfen seçenek giriniz: 1
+-------------+--------+---------+-----------------+----------+
| PersonelID |  İsim  | Soyisim |    Şube İsmi    |  Meslek  |
+-------------+--------+---------+-----------------+----------+
|     1      | Mahmut |   Hüda  | Ümraniye Merkez |  Şoför   |
|     8      |  Ece   |  Sezgin |  Dudullu Otogar | Sekreter |
|     7      | Samet  |   Elçi  |  Dudullu Otogar |  Muavin  |
|            |        |         |                 |          |
|     6      | Ahmet  |   Kara  |  Dudullu Otogar |  Şoför   |
|     2      | Hamza  |   Usta  |   Eryaman YHT   |  Muavin  |
|     3      | Kamil  |  Yiğit  |  Akdeniz Merkez | Sekreter |
|     5      | Mehtap |  Algül  | Umuttepe Kampüs | Yönetici |
+-------------+--------+---------+-----------------+----------+
```

# Uygulamanın Kaynak Kodları

Github dosyası: https://github.com/barisylmz53/otobusApp/

Youtube video adresi: https://youtu.be/vAXSktAUme4

**DİPNOT:** SQL ifadeleri github klasörünün içerisinde mevcuttur. Çok fazla satır olduğundan dolayı rapora yüklenmemiştir.