YILDIZ TEKNİK ÜNİVERSİTESİ

ELEKTRİK ELEKTRONİK FAKÜLTESİ

Veritabanı Yönetimi

(BLM3041)

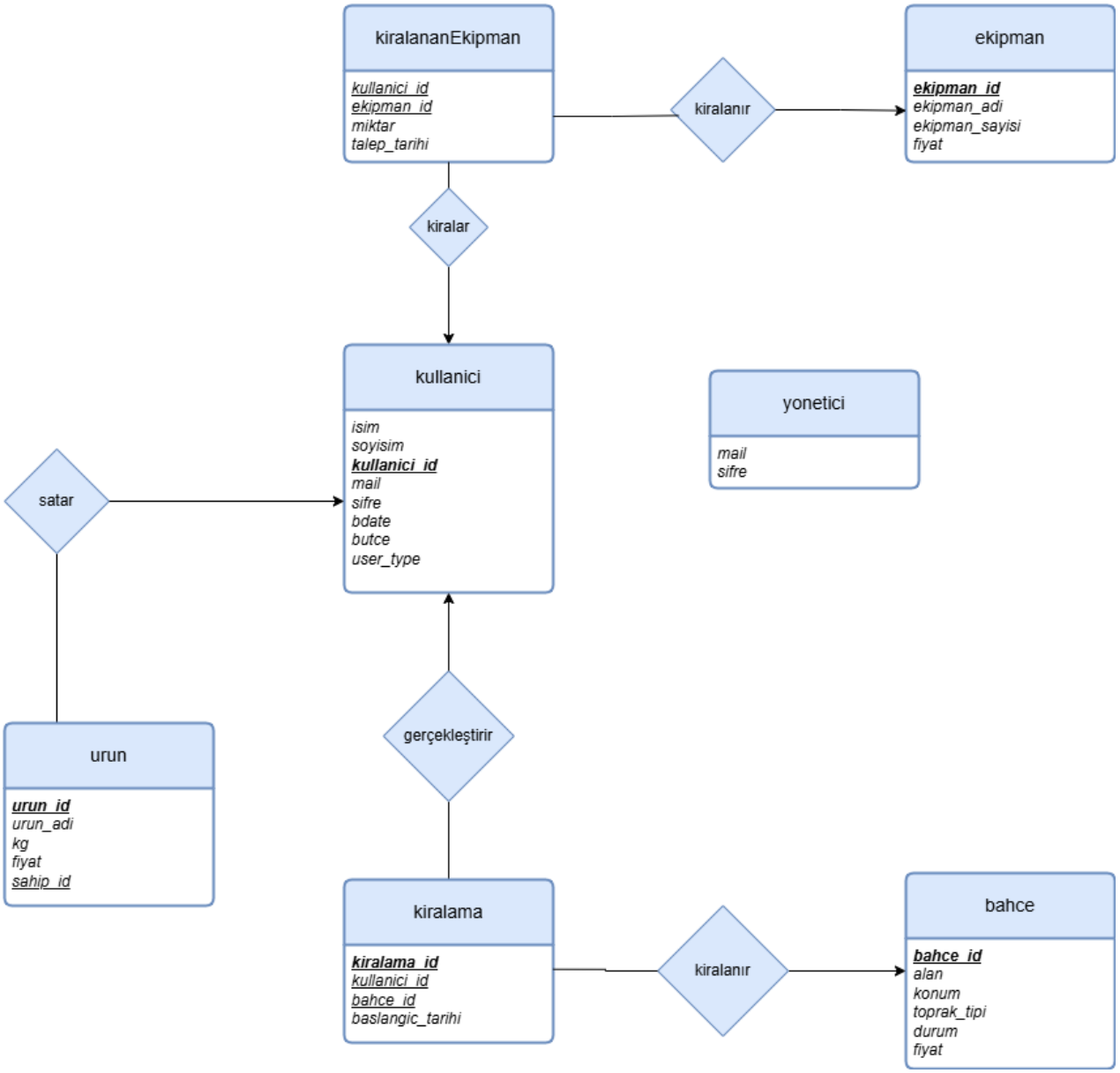
PROJE RAPORU

Grup 20

21011037 – İclal Ertürk
Ç19052100 – Şeyma Başaran
19011053 – Seray Çelik

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

1. ER Diyagramı



ER diyagramında primary key olan özellikler bold ve altı çizili şekilde, foreign key olan özellikler ise altı çizili şekilde gösterilmiştir. Ayrıca bağlantılarda ok şeklinde gösterilmiş olan uçlar many-to-one bağlantıların ‘one’ kısmını, ok olmayan uçlar ise ‘many’ kısmını simgelemektedir.

2. Tabloların Ekran Görüntüleri

a. ‘bahceler’ Tablosu

	bahce_id [PK] integer	alan numeric (5,2)	konum character varying (20)	toprak_tipi character varying (20)	durum character varying (20)	fiyat numeric (10,2)
1	1	10.00	1/1	Humuslu	Bos	800.00
2	2	10.00	1/2	Tınlı	Bos	700.00
3	3	10.00	1/3	Kireçli	Bos	600.00
4	4	10.00	1/4	Killi	Bos	600.00
5	5	10.00	2/1	Torf	Bos	600.00
6	6	10.00	2/2	Humuslu	Bos	800.00
7	7	10.00	2/3	Tınlı	Bos	700.00
8	8	10.00	2/4	Kireçli	Bos	600.00
9	9	10.00	3/1	Killi	Bos	600.00
10	10	10.00	3/2	Torf	Bos	600.00
11	11	10.00	3/3	Humuslu	Bos	800.00
12	12	10.00	3/4	Humuslu	Bos	800.00
13	13	10.00	4/1	Tınlı	Bos	700.00
14	14	10.00	4/2	Kireçli	Bos	600.00
15	15	10.00	4/3	Killi	Bos	600.00
16	16	10.00	4/4	Torf	Bos	600.00

b. ‘ekipman’ Tablosu

	ekipman_id [PK] integer	ekipman_adi character varying (20)	ekipman_sayisi integer	fiyat numeric (10,2)
1	10000	Kürek	12	40.00
2	10001	Çapa	7	50.00
3	10002	Budama Makası	8	60.00
4	10003	Balta	3	60.00
5	10004	Tırmık	5	40.00
6	10005	El Arabası	3	80.00
7	10006	Çim Makası	6	70.00
8	10007	Eldiven	15	30.00
9	10008	Bahçe Hortumu	16	55.00
10	10009	Kazma	4	50.00

c. ‘kiralamalar’ Tablosu (arayüzden kiralama gerçekleştirildiğinde kayıt oluşur)

	kiralama_id [PK] integer	kullanici_id integer	bahce_id integer	baslangic_tarihi date
1	1016	9	14	2025-01-04
2	1018	23	7	2025-01-04
3	1019	23	8	2025-01-04
4	1025	2	1	2025-01-12

d. ‘kiralananEkipmanlar’ Tablosu

	kullanici_id integer	ekipman_id integer	miktar integer	talep_tarihi date
1	3	10007	2	2025-01-01
2	5	10006	3	2025-01-03
3	8	10008	5	2024-12-03
4	4	10007	1	2024-12-03
5	4	10003	2	2025-01-01
6	6	10008	4	2025-01-01
7	2	10003	2	2025-01-01
8	1	10005	1	2025-01-01
9	1	10009	3	2025-01-01
10	3	10005	5	2025-01-01

e. ‘kullanicilar’ Tablosu

	isim character varying (20)	soyisim character varying (20)	kullanici_id [PK] integer	mail character varying (70)	sifre character varying (20)	bdate date	butce numeric (10,2)	user_type character varying (50)
1	Ford	Prefect	1	fordprefect@hitchhiker.com	12345678	1970-01-01	0.00	Kullanici
2	Arthur	Dent	2	arthurdent@earth.com	12345678	1977-06-24	0.00	Kullanici
3	Zaphod	Beeblebrox	3	zaphodbeeblebrox@galaxy.com	12345678	1965-11-29	0.00	Kullanici
4	Tricia	McMillan	4	trillian@earth.com	12345678	1978-02-10	0.00	Kullanici
5	Seray	Çelik	5	seraycelik@gardenhub.com	12345678	2000-01-01	0.00	Kullanici
6	İclal	Ertürk	6	iclalerturk@gardenhub.com	12345678	2000-01-01	0.00	Kullanici
7	Şeyma	Başaran	7	seymabasaran@gardenhub.com	12345678	2000-01-01	0.00	Kullanici
8	Luke	Skywalker	8	lukeskywalker@galaxy.com	12345678	1960-05-04	0.00	Kullanici
9	Leia	Organa	9	leiaorgana@galaxy.com	12345678	1960-05-04	0.00	Kullanici
10	Paul	Atreides	10	paulatreides@arrakis.com	12345678	1980-07-08	0.00	Kullanici

f. ‘urunler’ Tablosu

	urun_id [PK] integer	urun_adi character varying (20)	kg integer	fiyat integer	sahip_id integer
1	1034	Havuç	2	40	2
2	1035	Maydaoz	3	50	2
3	1036	Marul	5	60	2
4	1037	Nane	6	60	2
5	1038	Patates	4	40	2
6	1039	Biber	3	80	2
7	1040	Fasulye	8	70	9
8	1041	Salatalık	10	30	9
9	1042	Domates	5	55	9
10	1043	Üzüm	4	50	9

g. ‘yonetici’ Tablosu

	mail character varying (70)	sifre character varying (20)
1	admin@gardenhub.com	admin

3. Projede İstenilenlere Karşılık Gelen Kod Blokları

1. *Oluşturacağınız veritabanı en az 4 tablo içermelidir. Her tabloda en az 10 kayıt bulunmalıdır:* Raporun bir önceki başlığında tabloların ekran görüntülerinde görülebilir.
2. *Tablolarınızda primary key ve foreign key kısıtlarını kullanmalısınız:* Primary key ve foreign key içeren tablolardan biri aşağıda görülebilir. (GardenHub.sql satır 96)

```
96  CREATE TABLE Kiralamalar (
97      kiralama_id int PRIMARY KEY,
98      kullanici_id INT,
99      bahce_id INT REFERENCES Bahceler(bahce_id),
100     baslangic_tarihi DATE NOT NULL,
101     constraint kullanicilar_kullanici_id_fkey foreign key (kullanici_id)
102     references kullanicilar(kullanici_id) on delete cascade
103 );
```

3. *En az bir tabloda silme kısıtı ve sayı kısıtı olmalıdır.*:

- Sayı kısıtı (GardenHub.sql satır 14)

```
5   CREATE TABLE kullanicilar (
6       isim VARCHAR(20) NOT NULL,
7       soyisim VARCHAR(20) NOT NULL,
8       kullanici_id INT NOT NULL PRIMARY KEY,
9       mail VARCHAR(70) NOT NULL,
10      sifre VARCHAR(20) NOT NULL,
11      bdate DATE,
12      butce numeric(10,2) DEFAULT 0,
13      user_type VARCHAR(50) CHECK (user_type IN ('Kiraci', 'Kullanici')),
14      CONSTRAINT yas_sinir CHECK (AGE(CURRENT_DATE, bdate) >= INTERVAL '18 years')
15  );
```

- Sayı kısıtı (GardenHub.sql satır 89)

```
84  CREATE TABLE Bahceler (
85      bahce_id int NOT NULL PRIMARY KEY,
86      alan NUMERIC(5,2) NOT NULL,
87      konum varchar(20),
88      toprak_tipi varchar(20),
89      durum VARCHAR(20) NOT NULL CHECK (Durum IN ('Bos', 'Kiralanmis')),
90      fiyat NUMERIC(10,2) NOT NULL CHECK (Fiyat > 0)
91  );
```

- Sayı kısıtı (GardenHub.sql satır 62)

```
58  CREATE TABLE Ekipman (
59      ekipman_id int not null primary key,
60      ekipman_adi varchar(20) not null,
61      ekipman_sayisi int not null,
62      fiyat NUMERIC(10,2) NOT NULL CHECK (fiyat > 0)
63  );
```

- Silme kısıtı (GardenHub.sql satır 102, kullanıcı silindiğinde gerçekleştirdiği kiralamalar da silinir)

```
96  CREATE TABLE Kiralamalar (
97      kiralama_id int PRIMARY KEY,
98      kullanici_id INT,
99      bahce_id INT REFERENCES Bahceler(bahce_id),
100     baslangic_tarihi DATE NOT NULL,
101     constraint kullanicilar_kullanici_id_fkey foreign key (kullanici_id)
102     references kullanicilar(kullanici_id) on delete cascade
103 );
```

- Silme kısıtı (GardenHub.sql satır 173, kullanıcı silindiğinde kiraladığı ekipmanlar da silinir)

```
168 CREATE TABLE kiralananEkipmanlar (
169     kullanici_id INT REFERENCES Kullanicilar(kullanici_id),
170     ekipman_id INT REFERENCES Ekipman(ekipman_id),
171     miktar INT default 0,
172     talep_tarihi DATE DEFAULT CURRENT_DATE,
173     constraint kullanicilar_kullanici_id_fkey foreign key (kullanici_id)
174     references kullanicilar(kullanici_id) on delete cascade
175 );
```

- Silme kısıtı (GardenHub.sql satır 32, kullanıcı silindiğinde sattığı ürünler de silinir)

```
26  create table urunler(
27      urun_id int not null primary key,
28      urun_adi varchar(20) not null,
29      kg int,
30      fiyat int NOT NULL CHECK (Fiyat > 0),
31      sahip_id int,
32      constraint kullanicilar_kullanici_id_fkey foreign key (sahip_id)
33      references kullanicilar(kullanici_id) on delete cascade
34  );
```

4. *Arayüzden en az birer tane insert, update ve delete işlemi gerçekleştirilebilmelidir.*

1. DELETE işlemi (kullaniciSil.py satır 208, maile göre kullanıcı silme)

```
205                 cursor.execute("SELECT kullanici_mail_var_mi(%s);", (mail,))
206                 varmi = cursor.fetchone()[0]
207                 if varmi:
208                     query = "DELETE FROM kullanicilar WHERE mail = %s"
209                     cursor.execute(query, (mail,))
210                     conn.commit()
211                     conn.close()
```

2. INSERT işlemi aynı zamanda kullanici_mail_var_mi fonksiyonu kullanılmıştır. (kullaniciEkle.py satır 283, yöneticinin sisteme yeni bir kullanıcı eklemesi esnasında)

```
276             cursor.execute("SELECT kullanici_mail_var_mi(%s);", (mail,))
277             conn.commit()
278             varmi = cursor.fetchone()[0]
279             if varmi:
280                 QMessageBox.information(self, "Bilgi", "Bu Maile Sahip Kullanıcı Var.")
281                 conn.close()
282             else:
283                 query = "INSERT INTO kullanicilar VALUES( %s, %s, nextval('kullanici_id_seq'),%s, %s, %s)"
284                 cursor.execute(query, (isim, soyisim, mail, sifre, bdate))
```

3. UPDATE işlemi (ekipman.py satır 42, ekipman kiralandığında var olan ekipmanların sayısının bir azalması)

```
42              query = "UPDATE ekipman SET ekipman_sayisi = ekipman_sayisi - 1 WHERE ekipman_id = %s"
43              self.cursor.execute(query, (equipment_id,))
```

UPDATE işlemi (kullanicilar.py satır 55, bakiye yüklemek istenmesi)

```
45      def bakiye_guncelle(self, bakiye):
46          hostname = 'localhost'
47          username = 'postgres'
48          database = 'GardenHub'
49          password = '1234'
50          port_id = '5432'
51          try:
52              conn = psycopg2.connect(host=hostname, user=username, password=password, dbname=database, port=port_id)
53              cursor = conn.cursor()
54
55              cursor.execute("UPDATE kullanicilar SET butce = %s WHERE kullanici_id = %s", (bakiye, self.kullanici_id))
56              conn.commit()
57              conn.close()
58          except Exception as e:
59              print("Error: ", e)
```

5. *Arayüzden girilecek bir değere göre ekrana sonuçların listelendiği bir sorgu yazmalısınız.*: (bahçeler.py satır 516, bahce_class.py satır 45) Kullanıcı, kiralamak istediği toprak türüne göre bahçeleri listeleyebilir.

```
508         def bahceTuruSec(self):
509                 tur, ok = QInputDialog.getText(
510                         self,
511                         "Bahçe Türüne Göre Sınıflandırma",
512                         "Aramak istediğiniz bahçe türünü girin:",
513 
514                 )
515                 if ok:
516                         gelenBahceler = bahce_class.Bahce().get_bahce(tur)
517                         if gelenBahceler:
518                                 yazi = "Toprak Türüne Göre Bahçeler:\n"
519                                 for bahce in gelenBahceler:
520                                         yazi += f"Bahce Numarasi: {bahce[0]}\n"
521                                 QMessageBox.information(self, "Sonuç", yazi)
522                         else:
523                                 QMessageBox.information(self, "Sonuç Yok", "Belirtilen türde bahçe bulunamadı.")
524                 else:
525                         QMessageBox.warning(self, "İptal Edildi", "İşlemi iptal edildi.")
```

```
44     def get_bahce(self, toprak):
45         query = "SELECT bahce_id FROM bahceler WHERE toprak_tipi = %s ORDER BY bahce_id DESC"
46         self.cursor.execute(query, (toprak,))
47         rows = self.cursor.fetchall()
48         return rows
```

Örnek 2: (bahçeler.py satır 186, kiralama.py satır 52) Yönetici, istediği kullanıcının kiralama bilgisini görüntüleyebilir.

```
179     def showMailDialog(self):
180         mail, ok = QtWidgets.QInputDialog.getText(self, 'Mail Girin', 'Kullanıcı mailini girin:')
181         if ok and mail:
182             self.showKiralamaDetails2(mail)
183 
184     def showKiralamaDetails2(self, mail):
185         sonuc=0
186         rows = kiralama.Kiralama().showKiralamaDetails(mail,sonuc)
187         if sonuc==1:
188             QtWidgets.QMessageBox.warning(self, "Uyarı", "Geçersiz mail adresi.")
189         if rows:
190                     msg = QtWidgets.QMessageBox(self)
191                     msg.setWindowTitle("Kiralama Bilgileri")
192                     msg.setText(f"Mail: {mail}")
193                     table ="""
194                     <table border='1' style='background-color: white; border-collapse: collapse; width: 100%;'>
195                         <tr>
196                             <th style='background-color: rgb(240, 240, 240);'>Bahçe Numarası</th>
197                             <th style='background-color: rgb(240, 240, 240);'>Kiralama Tarihi</th>
198                         </tr>
199                     """
200                     for row in rows:
201 
202                         table += f"<tr style='background-color: white;'><td>{row[0]}</td><td>{row[1].strftime('%Y-%m-%d')}</td></t
203                     table += "</table>"
204                     msg.setInformativeText(table)
205                     msg.setStyleSheet("""
206                         QMessageBox {
207                             background-color: rgb(255, 255, 255);  /* Açık yeşil arka plan */
208                             color: black;  /* Yazı rengi */
```

```python
    def showKiralamaDetails(self, mail, sonuc):
        self.cursor = self.conn.cursor()
        query = "SELECT kullanici_id FROM kullanici_mail WHERE mail = %s"
        self.cursor.execute(query, (mail,))
        user_id = self.cursor.fetchone()
        query = "SELECT bahce_id, baslangic_tarihi FROM Kiralamalar WHERE kullanici_id = %s"
        if user_id:
            self.cursor.execute(query, (user_id,))
            rows = self.cursor.fetchall()

            return rows

        else:
            sonuc=1
            return None

```

6. *Arayüzden çağrılan sorgulardan en az biri "view" olarak tanımlanmış olmalıdır.* (GardenHub.sql satır 187, kiralama.py satır 29)

```sql
CREATE OR REPLACE VIEW kullanici_mail AS
SELECT mail, kullanici_id
FROM kullanicilar;
```

```python
    def get_kiralama_from_db(self):
            self.cursor = self.conn.cursor()
            self.cursor.execute("SELECT bahce_id,kullanici_id, baslangic_tarihi FROM Kiralamalar")
            self.conn.commit()
            user_data = self.cursor.fetchall()
            query = "SELECT mail FROM kullanici_mail WHERE kullanici_id = %s" #kullanici_mail viewi kullanıldı
```

Örnek 2: (GardenHub.sql satır 191, urunler.py satır 25, pazar.py satır 268)

```sql
CREATE OR REPLACE VIEW get_urun AS
SELECT urun_id, urun_adi, kg, fiyat, sahip_id from urunler;
```

```python
    def get_urun_from_db(self,order_by_price=False):
        try:
            query = "SELECT * from get_urun"
            if order_by_price:
                query += " ORDER BY fiyat ASC"
            self.cursor.execute(query)
            user_data = self.cursor.fetchall()
            return [Urunler(row[0], row[1], row[2], row[3], row[4]) for row in user_data]
        except Exception as e:
            print("Error: ", e)
            return []
```

```
266     def load_data(self, order_by_price=False):
267         self.ui.tableWidget.setRowCount(0)
268         gelenUrunler = self.urun.get_urun_from_db(order_by_price)
269         self.ui.tableWidget.setRowCount(len(gelenUrunler))
270         for row_idx, uruns in enumerate(gelenUrunler):
271             self.ui.tableWidget.setItem(row_idx, 0, QtWidgets.QTableWidgetItem(uruns.urun_adi))
272             self.ui.tableWidget.setItem(row_idx, 1, QtWidgets.QTableWidgetItem(str(uruns.kg)))
273             self.ui.tableWidget.setItem(row_idx, 2, QtWidgets.QTableWidgetItem(f"{uruns.fiyat:.2f} TL"))
274             self.ui.tableWidget.setItem(row_idx, 3, QtWidgets.QTableWidgetItem(str(uruns.sahip_id)))
275             button = QtWidgets.QPushButton("Satın Al")
276             button.setStyleSheet("""
277             QPushButton {
278                 background-color: rgb(131, 65, 0);
279             }
280             QPushButton:hover {
281                 background-color: rgb(170, 70, 0);
282             }
283             QPushButton:pressed {
284                 background-color: rgb(145, 70, 0);
285             }
286         """)
287             self.ui.tableWidget.setCellWidget(row_idx, 4, button)
288             button.clicked.connect(lambda _, r=row_idx: self.satinal_buton_tiklandi(r, gelenUrunler, order_by_price))
```

7. *En az bir adet "sequence" oluşturmalı ve arayüzden yapılacak insert sırasında ilgili sütundaki değerlerin otomatik olarak atanmasını sağlamalısınız.*: Kullanıcı tablosunun 'kullanici_id' sütunu sequence olarak tanımlanmıştır. Yönetici sisteme kullanıcı eklerken 'kullanici_id' kısmı sequence'ın bir sonraki değerini alır. (GardenHub.sql 1. satır, kullaniciEkle.py 283. satır)

```
1  create sequence kullanici_id_seq
2  minvalue 1
3  increment by 1
```

```
282             else:
283                 query = "INSERT INTO kullanicilar VALUES( %s, %s, nextval('kullanici_id_seq'),%s, %s, %s)"
284                 cursor.execute(query, (isim, soyisim, mail, sifre, bdate))
285                 conn.commit()
286                 conn.close()
287                 QMessageBox.information(self, "Bilgi", "Kullanıcı Ekleme İşlemi Başarılı.")
```

Urunler tablosunun 'urun_id' sütunu sequence olarak tanımlanmıştır. Kullanıcı sisteme urun eklerken urun_ekle fonksiyonunun içerisinde 'urun_id' kısmı sequence'ın bir sonraki değerini alır. (GardenHub.sql 22. satır, GardenHub.sql 46. satır, urunEkle.py 250. Satır)

```
22  create sequence urun_id_seq
23  minvalue 1000
24  increment by 1
25
```

```sql
create or replace function urun_ekle(urun_adi2 varchar(20),urun_kilosu2 int,
fiyat2 int, sahip_id2 int)
returns void as $$
declare
    mevcut urunler.urun_adi%type;
begin
    select urun_id into mevcut from urunler where urun_adi=urun_adi2 and sahip_id2
    = sahip_id;
    if mevcut is not null then
        update urunler set kg = kg+ urun_kilosu2
        where urun_adi=urun_adi2 and sahip_id2 = sahip_id;
    else
        INSERT INTO urunler VALUES(nextval('urun_id_seq'),urun_adi2,
        urun_kilosu2, fiyat2,sahip_id2);
    end if;
end;
$$ language 'plpgsql'
```

```python
    def add_product(self):
        urun_adi=self.ui.ad_line.text()
        urunun_kilosu=self.ui.soyad_line.text()
        fiyat=self.ui.mail_line.text()
        try:
            fiyat_int = int(fiyat)
            if fiyat_int <= 0:
                self.show_message("Başarısız", "Fiyat 0'dan büyük olmalıdır.", QMessageBox.Warning)
        except ValueError:
            self.show_message("Başarısız", "Geçersiz fiyat değeri. Lütfen bir sayı girin.", QMessageBox.Warning)
        else:
            query="select urun_ekle(%s,%s,%s,%s)"
            try:
                self.cursor.execute(query, (urun_adi,urunun_kilosu,fiyat,self.kullanici.kullanici_id))
                self.show_message("Başarılı", "Ürün başarıyla eklendi.", QMessageBox.Information)
            except psycopg2.IntegrityError as e:
                self.conn.rollback()
                self.show_message("Başarısız", "Ürün eklenemedi. Lütfen tekrar deneyin.", QMessageBox.Warning)
            except Exception as e:
                self.conn.rollback()
                self.show_message("Başarısız", "Ürün eklenemedi. Lütfen tekrar deneyin.", QMessageBox.Warning)
            self.conn.commit()
```

Ekipman tablosunun ‘ekipman_id’ sütunu sequence olarak tanımlanmıştır. Yönetici sisteme ekipman eklerken ekipman_ekle fonksiyonunun içerisinde ‘ekipman_id’ kısmı sequence’ın bir sonraki değerini alır. (GardenHub.sql 52. satır, GardenHub.sql 74. Satır, ekipmanEkle.py 242. Satır)

```sql
create sequence ekipman_id_seq
minvalue 10000
increment by 1
```

```
DROP FUNCTION ekipman_ekle(character varying,integer,numeric)
create or replace function ekipman_ekle(ekipman_adi2 ekipman.ekipman_adi%type,ekipman_sayisi2 int,
fiyat2 ekipman.fiyat%type )
returns void as $$
declare
    mevcut ekipman.ekipman_adi%type;
begin
    select ekipman_id into mevcut from ekipman where ekipman_adi=ekipman_adi2;
    if mevcut is not null then
        update ekipman set ekipman_sayisi = ekipman_sayisi+ ekipman_sayisi2
        where ekipman_adi=ekipman_adi2;
    else
        INSERT INTO ekipman VALUES(nextval('ekipman_id_seq'),ekipman_adi2,
        ekipman_sayisi2, fiyat2);
    end if;
end;
$$ language 'plpgsql'
```

```
        def add_equipment(self):
            try:
                ekipman_adi=self.ui.ad_line.text()
                ekipman_sayisi=self.ui.soyad_line.text()
                fiyat=self.ui.mail_line.text()
                query="select ekipman_ekle(%s,%s,%s)"
                self.cursor.execute(query, (ekipman_adi,ekipman_sayisi,fiyat))
                self.conn.commit()
                QMessageBox.information(self, "Bilgi", "Ekipman Eklendi.")
            except Exception as e:
                print("Error: ", e)
                self.conn.rollback()
                QMessageBox.information(self, "Bilgi", "Ekipman Eklenemedi.")
```

8. *Arayüzden çağrılan sorgulardan en az birinde union veya intersect veya except kullanmış olmalısınız.*: ‘Kiraci’ olmayan (henüz bahçe kiralamamış olan) kullanıcıları görüntülemek için kullanılan sorgu (kullanicilariGoruntule.py 197. satır)

```
    def get_kiraci_olmayan_kullanicilar(self):
        hostname = 'localhost'
        username = 'postgres'
        database = 'GardenHub'
        password = '1234'
        port_id = '5432'
        try:
            conn = psycopg2.connect(host=hostname, user=username, password=password, dbname=database, port=port_id
            )
            cursor = conn.cursor()
            cursor.execute("SELECT isim,soyisim,mail From kullanicilar EXCEPT SELECT isim,soyisim,mail From kullanicilar WHERE user_type='Kiraci'")
            users = cursor.fetchall()
            conn.close()
            self.ui.tableWidget.setRowCount(len(users))
            for row_idx, user in enumerate(users):
                self.ui.tableWidget.setItem(row_idx, 0, QtWidgets.QTableWidgetItem(user[0]))
                self.ui.tableWidget.setItem(row_idx, 1, QtWidgets.QTableWidgetItem(user[1]))
                self.ui.tableWidget.setItem(row_idx, 2, QtWidgets.QTableWidgetItem(user[2]))
        except Exception as e:
            print
```

Örnek 2: ‘Kiraci’ olan kullanıcıları görüntülemek için kullanılan sorgu (kullanicilariGoruntule.py 220. satır)

```
210     def get_kiraci_olan_kullanicilar(self):
211         hostname = 'localhost'
212         username = 'postgres'
213         database = 'GardenHub'
214         password = '1234'
215         port_id = '5432'
216         try:
217             conn = psycopg2.connect(host=hostname, user=username, password=password, dbname=database, port=port_id
218             )
219             cursor = conn.cursor()
220             cursor.execute("SELECT isim,soyisim,mail From kullanicilar EXCEPT SELECT isim,soyisim,mail From kullanicilar WHERE
                user_type='Kullanici'")
221             users = cursor.fetchall()
222             conn.close()
223             self.ui.tableWidget.setRowCount(len(users))
224             for row_idx, user in enumerate(users):
225                 self.ui.tableWidget.setItem(row_idx, 0, QtWidgets.QTableWidgetItem(user[0]))
226                 self.ui.tableWidget.setItem(row_idx, 1, QtWidgets.QTableWidgetItem(user[1]))
227                 self.ui.tableWidget.setItem(row_idx, 2, QtWidgets.QTableWidgetItem(user[2]))
228         except Exception as e:
229             print("Error: ", e)
230
```

9. *Sorgularınızın en az biri aggregate fonksiyonlar içermeli, having ifadesi kullanılmalıdır.*: Pazardaki satıcılarda 10 kilogramdan fazla stoğu olan satıcıların sattığı ürün türünün sayısını ve ürünlerin toplam kilogram değerini (toplam_kg), toplam_kg’ye göre azalan olarak listeleyen sorgu. (pazar.py 296. Satır)

```
293         def load_aggregated_data(self):
294             try:
295                 query = """
296                     SELECT sahip_id, COUNT(*) AS sattigi_urun_sayisi, SUM(kg) AS toplam_kg
297                     FROM urunler
298                     GROUP BY sahip_id
299                     HAVING SUM(kg) > 10
300                     ORDER BY toplam_kg DESC
301                 """
```

10. *Arayüzden girilen değerleri parametre olarak alıp ekrana sonuç döndüren 3 farklı SQL fonksiyonu tanımlamış olmalısınız. Bu fonksiyonların en az birinde “record” ve “cursor” tanımı-kullanımı olmalıdır.* :

    1. Kullanıcı ekleme ve silme sırasında girilen mail adresinin var olup olmadığını arayüze mesaj olarak döndüren fonksiyon (GardenHub.sql satır 173, kullaniciSil.py satır 205)

```sql
CREATE OR REPLACE FUNCTION kullanici_mail_var_mi(email TEXT)
RETURNS BOOLEAN AS $$
DECLARE
    mail_var BOOLEAN;
BEGIN
    -- E-posta adresinin var olup olmadığını kontrol et
    SELECT EXISTS (SELECT 1 FROM kullanicilar WHERE mail = email) INTO mail_var;

    RETURN mail_var;
END;
$$ LANGUAGE plpgsql;
```

```python
                cursor.execute("SELECT kullanici_mail_var_mi(%s);", (mail,))
                varmi = cursor.fetchone()[0]
                if varmi:
                    query = "DELETE FROM kullanicilar WHERE mail = %s"
                    cursor.execute(query, (mail,))
                    conn.commit()
                    conn.close()
                    QMessageBox.information(self, "Bilgi", "Kullanıcı Silme İşlemi Başarılı.
                else:
                    QMessageBox.information(self, "Bilgi", "Bu Maile Sahip Kullanıcı Yok.")
```

2. pazar.py 228. satırdan başlayan fonksiyonda, urun_stogu_hesapla fonksiyonu çağrılmıştır. Bu fonksiyon, farklı kullanıcıların aynı ürünü satması durumunda ürün adıyla beraber ürünün toplam miktarını ve toplam fiyatını urun_stogu tipinde döndürür. Arayüzden ürün adını alır ve buna göre veritabanında ilgili sonucu döndürür.

```python
    def cursorVeRecordlaUrununTumStogunuGor(self):
        urunAdi, ok = QInputDialog.getText(
            self,
            "Ürünün Stoğuna Bak",
            "Aradığınız Ürünü Girin:",

        )

        if ok:  # Kullanıcı 'Tamam'a tıkladıysa
            try:
                query = """
                    select urun_stogu_hesapla(%s)
                """
```

```sql
CREATE TYPE urun_stogu AS (urun_adi varchar(20), kg int, fiyat int);

CREATE OR REPLACE FUNCTION urun_stogu_hesapla(urun_adi_1 varchar(20))
RETURNS urun_stogu AS $$
DECLARE
    stok urun_stogu;
    toplam_fiyat int;
    toplam_kg int;
    cur1 CURSOR FOR SELECT * FROM urunler WHERE urun_adi = urun_adi_1;
BEGIN
    toplam_fiyat :=0;
    toplam_kg :=0;
    FOR satir IN cur1 LOOP
        toplam_fiyat := toplam_fiyat + (satir.fiyat * satir.kg);
        toplam_kg := toplam_kg + satir.kg;
    END LOOP;
    SELECT urun_adi_1, toplam_kg, toplam_fiyat INTO stok;
    RETURN stok;
END;
$$ language 'plpgsql'
```

3. Kullanıcının ürün eklemesi sırasında ekleme işleminin başarılı olup olmadığını mesaj olarak gösteren fonksiyon (GardenHub.sql satır 34, urunEkle.py satır 242)

```sql
create or replace function urun_ekle(urun_adi2 varchar(20),urun_kilosu2 int,
fiyat2 int, sahip_id2 int)
returns void as $$
declare
    mevcut urunler.urun_adi%type;
begin
    select urun_id into mevcut from urunler where urun_adi=urun_adi2 and sahip_id2
    = sahip_id;
    if mevcut is not null then
        update urunler set kg = kg+ urun_kilosu2
        where urun_adi=urun_adi2 and sahip_id2 = sahip_id;
    else
        INSERT INTO urunler VALUES(nextval('urun_id_seq'),urun_adi2,
        urun_kilosu2, fiyat2,sahip_id2);
    end if;
end;
$$ language 'plpgsql'
```

```python
            query="select urun_ekle(%s,%s,%s,%s)"
            try:
                self.cursor.execute(query, (urun_adi,urunun_kilosu,fiyat,self.kullanici.kullanici_id))
                self.show_message("Başarılı", "Ürün başarıyla eklendi.", QMessageBox.Information)
            except psycopg2.IntegrityError as e:
                self.show_message("Başarısız", "Ürün eklenemedi. Lütfen tekrar deneyin.", QMessageBox.Warning)
            except Exception as e:
                self.show_message("Başarısız", "Ürün eklenemedi. Lütfen tekrar deneyin.", QMessageBox.Warning)
            self.conn.commit()
```

4.Yöneticinin ekipman eklemesi işleminin başarılı olup olmadığını mesaj olarak gösteren fonksiyon (GardenHub.sql satır 62, ekipmanEkle.py satır 242)

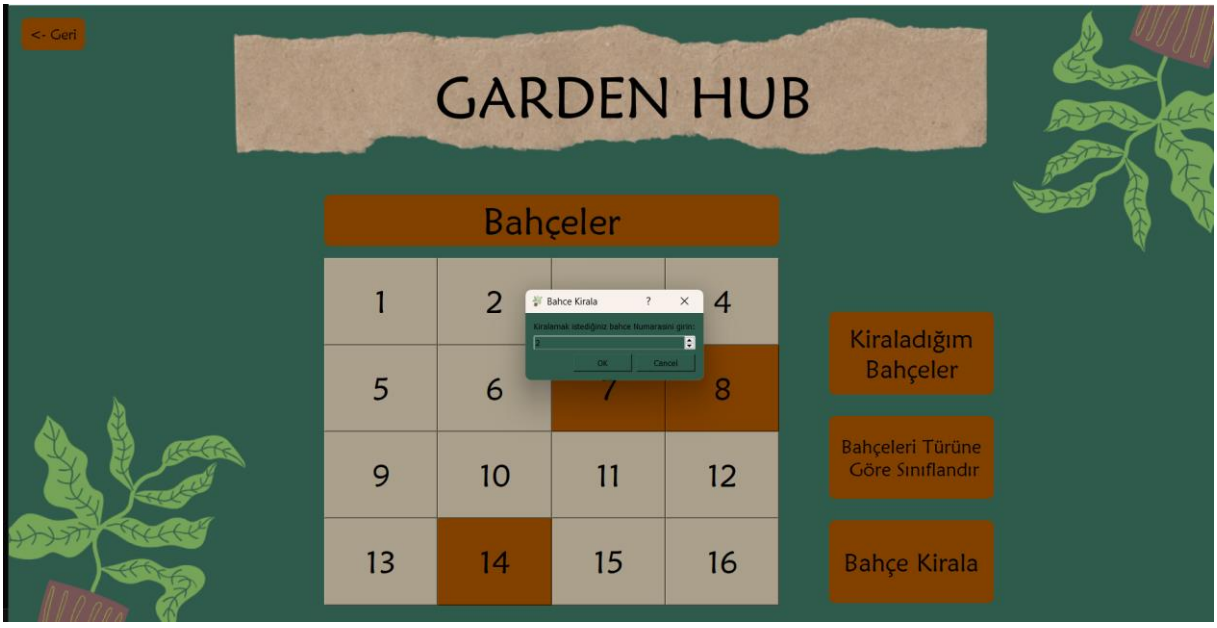
```
62  DROP FUNCTION ekipman_ekle(character varying,integer,numeric)
63  create or replace function ekipman_ekle(ekipman_adi2 ekipman.ekipman_adi%type,ekipman_sayisi2 int,
64  fiyat2 ekipman.fiyat%type )
65  returns void as $$
66  declare
67      mevcut ekipman.ekipman_adi%type;
68  begin
69      select ekipman_id into mevcut from ekipman where ekipman_adi=ekipman_adi2;
70      if mevcut is not null then
71          update ekipman set ekipman_sayisi = ekipman_sayisi+ ekipman_sayisi2
72          where ekipman_adi=ekipman_adi2;
73      else
74          INSERT INTO ekipman VALUES(nextval('ekipman_id_seq'),ekipman_adi2,
75          ekipman_sayisi2, fiyat2);
76      end if;
77  end;
78  $$ language 'plpgsql'
79
```

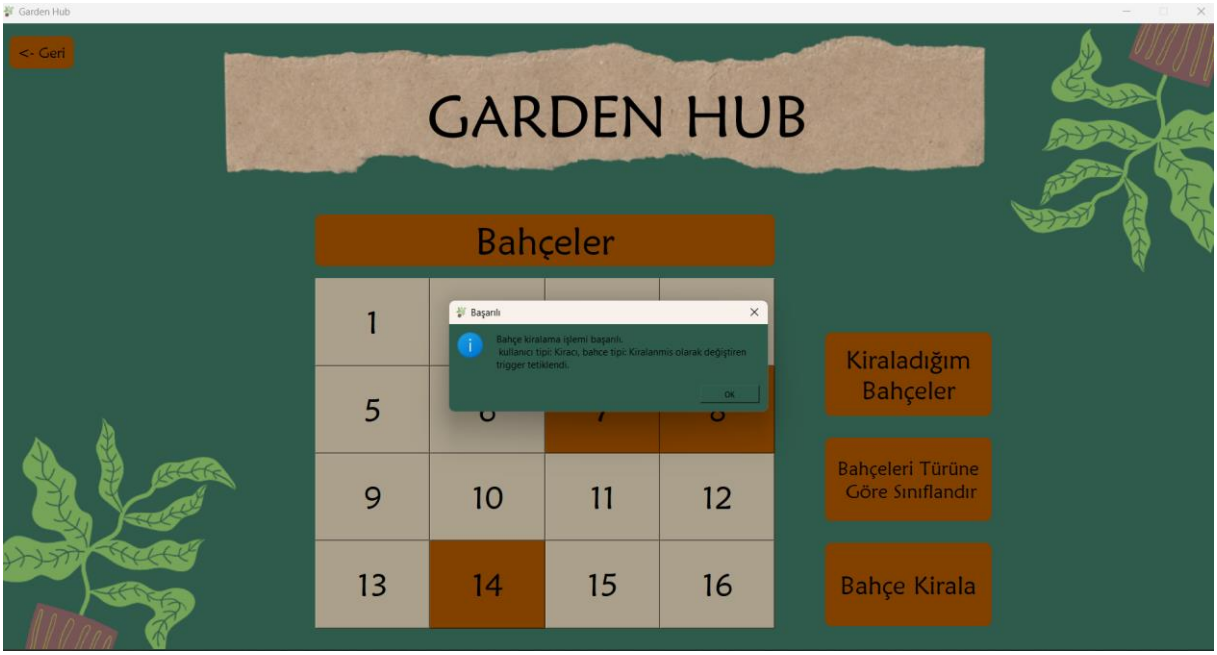
```
237          def add_equipment(self):
238              try:
239                  ekipman_adi=self.ui.ad_line.text()
240                  ekipman_sayisi=self.ui.soyad_line.text()
241                  fiyat=self.ui.mail_line.text()
242                  query="select ekipman_ekle(%s,%s,%s)"
243                  self.cursor.execute(query, (ekipman_adi,ekipman_sayisi,fiyat))
244                  self.conn.commit()
245                  QMessageBox.information(self, "Bilgi", "Ekipman Eklendi.")
246              except Exception as e:
247                  print("Error: ", e)
248                  self.conn.rollback()
249                  QMessageBox.information(self, "Bilgi", "Ekipman Eklenemedi.")
```

11. *2 adet trigger tanımlamalı ve arayüzden girilecek değerlerle tetiklemelisiniz. Trigger'ın çalıştığına dair arayüze bilgilendirme mesajı döndürülmelidir.*

1.Bir bahçe kiralandığı zaman kiralayan kişinin 'user_type' ını 'Kiracı' olarak değiştiren ve kiralanan bahçenin durumunu 'Kiralanmış' yapan trigger. (GardenHub.sql satır 109)

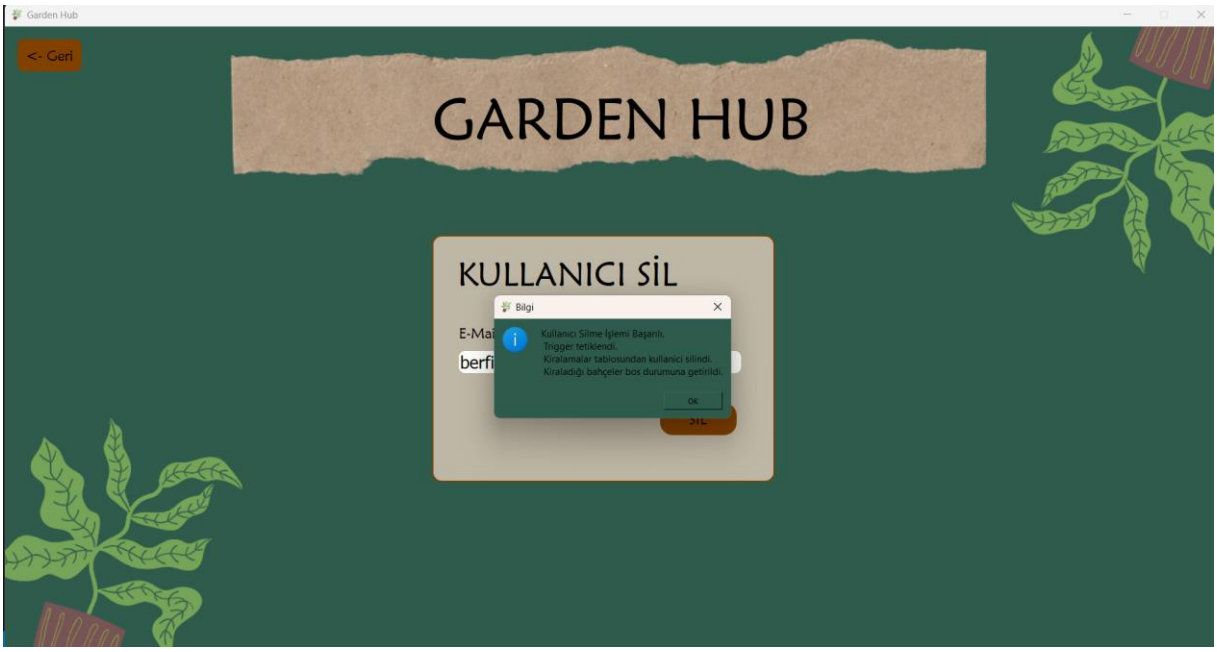
```
109  CREATE TRIGGER trigger_update_bahce_durum
110  AFTER INSERT ON Kiralamalar
111  FOR EACH ROW
112  EXECUTE FUNCTION update_bahce_durum_on_kiralama();
113
114  CREATE OR REPLACE FUNCTION update_bahce_durum_on_kiralama()
115  RETURNS TRIGGER AS $$
116  BEGIN
117      -- Kiralama eklenen bahçenin durumunu "Kiralanmış" olarak güncelle
118      UPDATE bahceler
119      SET durum = 'Kiralanmis'
120      WHERE bahce_id = NEW.bahce_id;
121      --kullanicinin türünü kiracı yap
122      UPDATE kullanicilar
123      set user_type = 'Kiraci'
124      where kullanici_id=NEW.kullanici_id;
125
126      RETURN NEW;
127  END;
128  $$ LANGUAGE plpgsql;
```





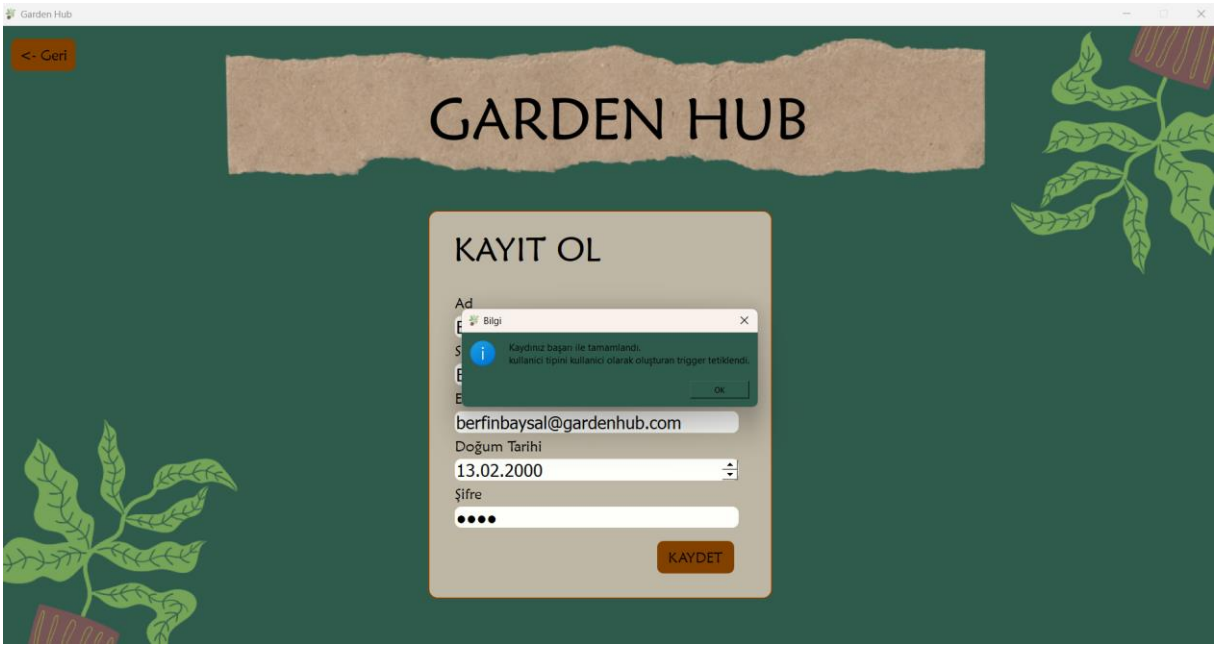
2.Bir kullanıcı silindiği zaman silme kısıtı nedeniyle kiralamalardan da kayıtı silinir. Kiralamalardan kayıt silindiği zaman silinen kayıttaki bahçenin durumu kiralanmış durumundan boş durumuna geçiren trigger.(GardenHub.sql satır 132)

```
132  CREATE TRIGGER trigger_sifirla_bahce_durum
133  AFTER DELETE ON Kiralamalar
134  FOR EACH ROW
135  EXECUTE FUNCTION sifirla_bahce_durum_on_bahceler();
136
137  CREATE OR REPLACE FUNCTION sifirla_bahce_durum_on_bahceler()
138  RETURNS TRIGGER AS $$
139  BEGIN
140      -- silinen bahçenin durumunu "Bos" olarak güncelle
141      UPDATE bahceler
142      SET durum = 'Bos'
143      WHERE bahce_id = OLD.bahce_id;
144      RETURN OLD;
145  END;
146  $$ LANGUAGE plpgsql;
```



3.Sisteme bir kişi kayıt olduğu zaman kişinin ‘user_type’ ını ‘Kullanıcı’ olarak belirleyen trigger. (GardenHub.sql satır 147)

```
147  CREATE TRIGGER kullanici_kaydi
148  after insert ON Kullanicilar
149  FOR EACH ROW
150  EXECUTE FUNCTION kullanici_kaydi_fonk();
151
152  CREATE OR REPLACE FUNCTION kullanici_kaydi_fonk()
153  RETURNS TRIGGER AS $$
154  BEGIN
155      -- eklenen kullanici Kullanici tipinde eklensin
156      UPDATE Kullanicilar
157      SET user_type = 'Kullanici'
158      WHERE kullanici_id = NEW.kullanici_id;
159      RETURN NEW;
160  END;
161  $$ LANGUAGE plpgsql;
```


Garden Hub
<- Geri
GARDEN HUB
KAYIT OL
Ad
Bilgi
Kaydınız başarı ile tamamlandı.
kullanici tipini kullanici olarak oluşturan trigger tetiklendi.
OK
berfinbaysal@gardenhub.com
Doğum Tarihi
13.02.2000
Şifre
KAYDET