YILDIZ TEKNİK ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Veritabanı Yönetimi Proje

Öğrenciler:

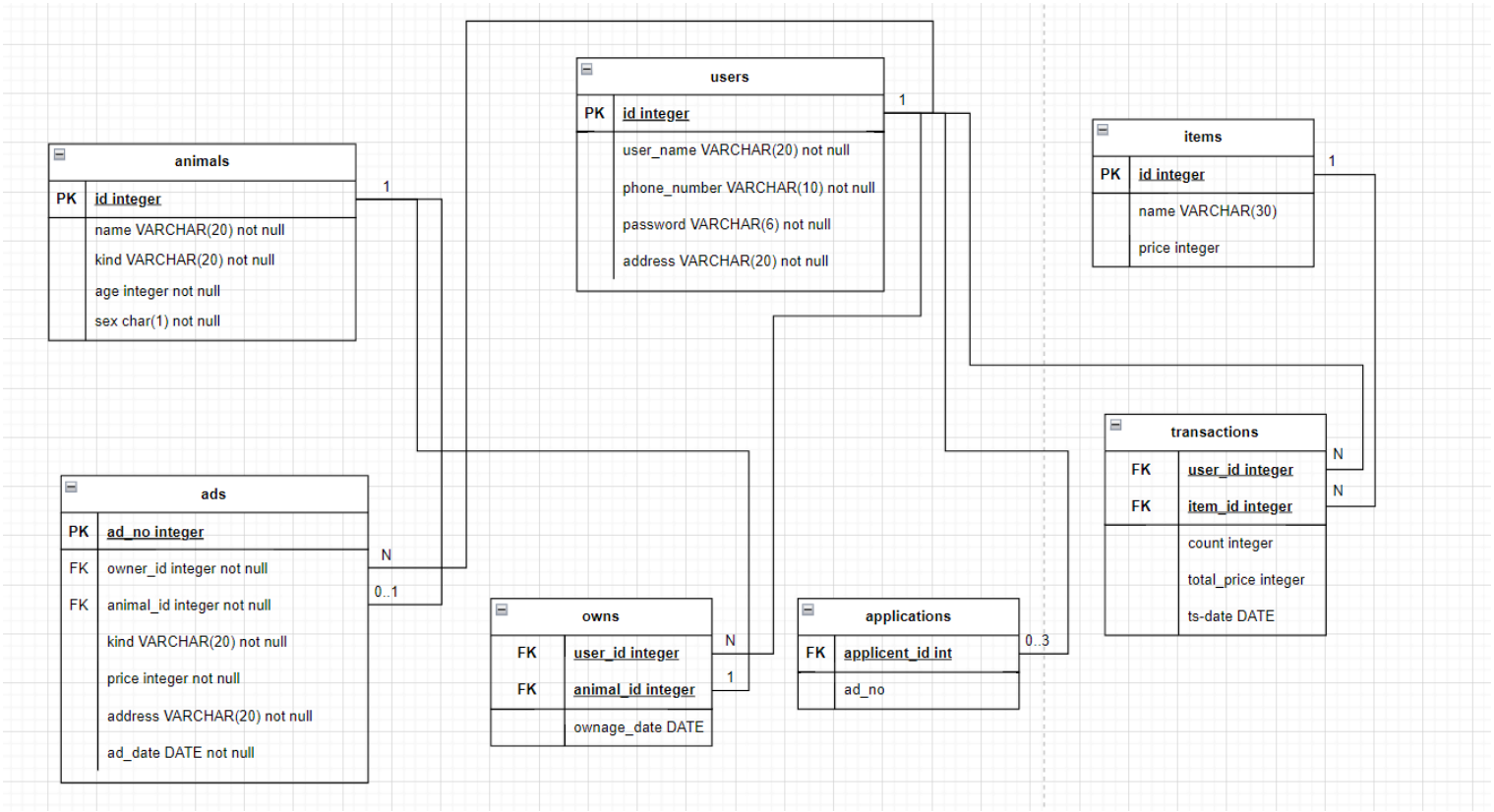
Yavuz ÇETİN	21011004
Ahmet Mahir DEMİRELLİ	21011063
Ahmed Asım SEVİMLİ	21011005
Osman Berkay SUKAS	20011502

E-Posta:

yavuz.cetin1@std.yildiz.edu.tr
mahir.demirelli@std.yildiz.edu.tr
asim.sevimli@std.yildiz.edu.tr
berkay.sukas@std.yildiz.edu.tr

Dersi Veren: Dr. Öğr. Üyesi Mustafa Utku KALAY

ER Diyagramı:



Tabloların Ekran Görüntüleri:

```
-- sequence
CREATE SEQUENCE user_id_seq START 1000;
CREATE SEQUENCE animal_id_seq START 1;
CREATE SEQUENCE ad_no_seq START 1;
-- Tables
CREATE TABLE animals(
    id int DEFAULT nextval('animal_id_seq'),
    name varchar(20) not null,
    kind varchar(20),
    age int not null,
    sex char(1) not null,
    CONSTRAINT PK_Animal PRIMARY KEY (id)
);
CREATE TABLE users(
    id int DEFAULT nextval('user_id_seq'),
    user_name varchar(20) not null,
    phone_number char(10) not null,
    password char(6) DEFAULT '123456',
    address varchar(20) not null,
    CONSTRAINT PK_Customer PRIMARY KEY (id)
);
```

```
CREATE TABLE owns(
    user_id int not null,
    animal_id int not null,
    ownage_date DATE not null,
    foreign key (user_id) references users(id) ON DELETE CASCADE,
    foreign key (animal_id) references animals(id) ON DELETE CASCADE
);
ALTER TABLE owns ADD CONSTRAINT unique_animal_user_id UNIQUE (user_id, animal_id);
CREATE TABLE ads(
    ad_no int DEFAULT nextval('ad_no_seq'),
    owner_id int not null,
    animal_id int not null,
    kind varchar(20) not null,
    price int not null,
    address varchar(20) not null,
    ad_date DATE not null,
    foreign key (animal_id) references animals(id) ON DELETE CASCADE,
    foreign key (owner_id) references users(id) ON DELETE CASCADE,
    CONSTRAINT price_ck CHECK  (price >=50)
);
```

```
CREATE TABLE applications(
    applicent_id int not null,
    ad_no int not null,
    foreign key (applicent_id) references users(id) ON DELETE CASCADE
);
ALTER TABLE applications ADD CONSTRAINT unique_ad_no_id UNIQUE (ad_no, applicent_id);
CREATE TABLE items(
    id int not null,
    name varchar(30) not null,
    price int not null,
    CONSTRAINT PK_Item PRIMARY KEY (id)
);
CREATE TABLE transactions(
    user_id int not null,
    item_id int not null,
    count int not null,
    total_price int not null,
    ts_date DATE not null,
    foreign key (user_id) references users(id) ON DELETE CASCADE,
    foreign key (item_id) references items(id) ON DELETE CASCADE
);
```

Yukarıdaki maddelerin sağlandığını gösteren kod blokları.

1- En az 4 tablo ve her tabloda en az 10 kayıt.

Query | Query History

```sql
SELECT
  (SELECT COUNT(*) FROM animals) AS animals_count,
  (SELECT COUNT(*) FROM users) AS users_count,
  (SELECT COUNT(*) FROM owns) AS owns_count,
  (SELECT COUNT(*) FROM ads) AS ads_count,
  (SELECT COUNT(*) FROM applications) AS applications_count,
  (SELECT COUNT(*) FROM items) AS items_count,
  (SELECT COUNT(*) FROM transactions) AS transactions_count;
```

Data Output | Messages | Notifications

	animals_count bigint	users_count bigint	owns_count bigint	ads_count bigint	applications_count bigint	items_count bigint	transactions_count bigint
1	24	16	24	15	10	20	10

2- Tablolarda primary ve foreign key kısıtları

```sql
animals      CONSTRAINT PK_Animal PRIMARY KEY (id)

users        CONSTRAINT PK_Customer PRIMARY KEY (id)

             foreign key (user_id) references users(id) ON DELETE CASCADE,
owns         foreign key (animal_id) references animals(id) ON DELETE CASCADE

ads          CONSTRAINT PK_Ads PRIMARY KEY (ad_no),

applications foreign key (applicent_id) references users(id) ON DELETE CASCADE

items        CONSTRAINT PK_Item PRIMARY KEY (id)

             foreign key (user_id) references users(id) ON DELETE CASCADE,
transactions foreign key (item_id) references items(id) ON DELETE CASCADE
```

3- En az 1 tabloda silme kısıtı ve sayı kısıtı

```sql
CREATE TABLE ads(
    ad_no int DEFAULT nextval('ad_no_seq'),
    owner_id int not null,
    animal_id int not null,
    kind varchar(20) not null,
    price int not null,
    address varchar(20) not null,
    ad_date DATE not null,
    foreign key (animal_id) references animals(id) ON DELETE CASCADE,
    foreign key (owner_id) references users(id) ON DELETE CASCADE,
    CONSTRAINT price_ck CHECK  (price >=50)
);
```

4- Arayüzden en az birer tane Insert, Update ve Delete işlemi

a. Insert : Kullanıcı ilan verirken ads tablosuna kullanıcıdan alınan bilgiler Insert edilmektedir.

CreateAd.java

```
                    statement_2.setInt(1, animal_id);
                    try (ResultSet result_1 = statement_2.executeQuery()) {
                        while (result_1.next()) {
                            kind = result_1.getString(1);
                        }
                    }
                }

                query = "INSERT INTO ads(owner_id,animal_id,kind,price,address,ad_date) VALUES (?, ?, ?, ?, ?, ?);";
                statement = conn.prepareStatement(query);
                statement.setInt(1, userId);
                statement.setInt(2, animal_id);
                statement.setString(3, kind);
                statement.setInt(4, price);
                statement.setString(5, address);
                statement.setDate(6, date);
                statement.execute();
                JOptionPane.showMessageDialog(null, "Your ad has been listed.");
                inputAnimalID.setText("");
                inputPrice.setText("");
                updateList();
```

b. Update : Kullanıcı bilgilerini değiştirmek istediğinde Update kullanılarak bilgileri güncellenmektedir.

UserPage.java

```
        p.clearParameters();
        // p.setString(1, toBeChanged);
        p.setString(1, newString);
        p.setInt(2, idOfUser);
        p.execute();

    }

    public void changePhone(String newString, int idOfUser) throws SQLException {
        String query = "UPDATE users SET phone_number = ? WHERE id = ?;";

        PreparedStatement p = conn.prepareStatement(query);
        p.clearParameters();
        // p.setString(1, toBeChanged);
        p.setString(1, newString);
        p.setInt(2, idOfUser);
        p.execute();

    }

    public void changeAddress(String newString, int idOfUser) throws SQLException {
        String query = "UPDATE users SET address = ? WHERE id = ?;";

        PreparedStatement p = conn.prepareStatement(query);
        p.clearParameters();
        // p.setString(1, toBeChanged);
        p.setString(1, newString);
        p.setInt(2, idOfUser);
        p.execute();

    }

    public void changePassword(String newString, int idOfUser) throws SQLException {
        String query = "UPDATE users SET password = ? WHERE id = ?;";

        PreparedStatement p = conn.prepareStatement(query);
        p.clearParameters();
        // p.setString(1, toBeChanged);
        p.setString(1, newString);
        p.setInt(2, idOfUser);
        p.execute();

    }
}
```

c. Delete : Kullanıcı başvurduğu bir ilandan başvurusunu silebilir.

MyApps.java

```
139                     try {
140                         statement = conn.prepareStatement(query);
141                         statement.setInt(1, adNo);
142                         statement.setInt(2, id);
143                         ResultSet result = statement.executeQuery();
144                         while (result.next()) {
145                             check = result.getInt(1);
146                         }
147                     } catch (SQLException e1) {
148                         // TODO Auto-generated catch block
149                         e1.printStackTrace();
150                     }
151                     if (check != -1) {
152                         query = "DELETE FROM applications WHERE ad_no = ? AND applicent_id = ?;";
153                         try {
154                             statement = conn.prepareStatement(query);
155                             statement.setInt(1, adNo);
156                             statement.setInt(2, id);
157                             statement.execute();
158                             JOptionPane.showMessageDialog(null, "Your application has been deleted");
159                             updateList();
160                             inputAdNo.setText("");
```

5- Arayüzden girilecek değere göre ekrana sonuç listeleme : ShopModule den gelen kind bilgisine göre başvurulabilir ilanları gösteriyor.

AdsPage.java

```
131             }
132         });
133         btnApply.setFont(new Font("Tahoma", Font.PLAIN, 16));
134         btnApply.setBounds(75, 200, 100, 30);
135         contentPane.add(btnApply);
136
137         printAdsTable = new JTable();
138         printAdsTable.setFont(new Font("Tahoma", Font.PLAIN, 15));
139         printAdsTable.setBounds(225, 75, 765, 300);
140         contentPane.add(printAdsTable);
141         model = new DefaultTableModel();
142         model.addColumn("Ad No");
143         model.addColumn("Owner ID");
144         model.addColumn("Kind");
145         model.addColumn("Price");
146         model.addColumn("Address");
147         model.addColumn("Ad Date");
148         model.addColumn("Name");
149         model.addColumn("Age");
150         model.addColumn("Sex");
151         String query = "CREATE OR REPLACE VIEW list_Ad_View AS "
152                 + "SELECT ads.ad_no, ads.owner_id, ads.kind,ads.price,ads.address,ads.ad_date, animals.name, animals.age, animals.sex"
153                 + " FROM ads, animals " + "WHERE ads.kind = '" + kind
154                 + "' AND ads.animal_id = animals.id AND owner_id !=" + id + " AND ads.ad_no NOT IN (SELECT ad_no FROM applications WHERE applicent_id = " + id
155                 + ") ORDER BY ad_no;";
```

6- Arayüzden çağırılan sorgulardan en az biri view : Ads Page de uygun ilanları listelemek için view kullandık.

AdsPage.java

```
143         model.addColumn("Owner ID");
144         model.addColumn("Kind");
145         model.addColumn("Price");
146         model.addColumn("Address");
147         model.addColumn("Ad Date");
148         model.addColumn("Name");
149         model.addColumn("Age");
150         model.addColumn("Sex");
151         String query = "CREATE OR REPLACE VIEW list_Ad_View AS "
152                 + "SELECT ads.ad_no, ads.owner_id, ads.kind,ads.price,ads.address,ads.ad_date, animals.name, animals.age, animals.sex"
153                 + " FROM ads, animals " + "WHERE ads.kind = '" + kind
154                 + "' AND ads.animal_id = animals.id AND owner_id !=" + id + " AND ads.ad_no NOT IN (SELECT ad_no FROM applications WHERE applicent_id = " + id
155                 + ") ORDER BY ad_no;";
156         try {
157             PreparedStatement statement = conn.prepareStatement(query);
158             statement.execute();
159             query = "SELECT * FROM list_Ad_View";
160             statement = conn.prepareStatement(query);
161             ResultSet result = statement.executeQuery();
162             while (result.next()) {
163                 Object[] row = new Object[model.getColumnCount()];
164                 for (int i = 0; i < model.getColumnCount(); i++) {
165                     row[i] = result.getObject(i + 1);
166                 }
167                 model.addRow(row);
168             }
169             printAdsTable.setModel(model);
```

7- En az 1 adet sequence oluşturulmalı ve arayüzden yapılacak Insert sırasında ilgili sütündaki değer otomatik sağlanmalı :

```
-- sequence
CREATE SEQUENCE user_id_seq START 1000;
CREATE SEQUENCE animal_id_seq START 1;
CREATE SEQUENCE ad_no_seq START 1;

CREATE TABLE users(
    id int DEFAULT nextval('user_id_seq'),

CREATE TABLE animals(
    id int DEFAULT nextval('animal_id_seq'),
CREATE TABLE ads(
    ad no int DEFAULT nextval('ad no seq'),
```

CreateAd.java

```
                }

                query = "INSERT INTO ads(owner_id,animal_id,kind,price,address,ad_date) VALUES (?, ?, ?, ?, ?, ?);";
                statement = conn.prepareStatement(query);
                statement.setInt(1, userId);
                statement.setInt(2, animal_id);
                statement.setString(3, kind);
                statement.setInt(4, price);
                statement.setString(5, address);
                statement.setDate(6, date);
                statement.execute();
```

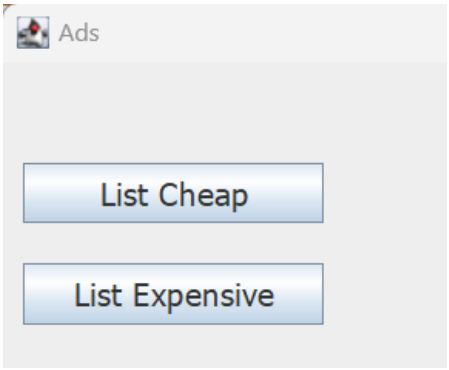
Ad_no için Sequence olduğundan Insert sırasında ad_no verilmiyor.

8- Arayüzden çağırılan sorgulardan en az biri Union : Advanced Search page de aynı anda 2 farklı hayvan türü ile ilan aranması union kullanarak sağlanmıştır.

AdvancedSearch.java

```
131                 DatabaseMetaData metaData = conn.getMetaData();
132                 ResultSet resultSet = metaData.getColumns(null, null, "ads", null);
133
134                 while (resultSet.next()) {
135                     String columnName = resultSet.getString("COLUMN_NAME");
136                     model.addColumn(columnName);
137                 }
138                 model.addColumn("Age");
139
140                 String query = "SELECT ad_no,owner_id,animal_id,d.kind,d.price,address,ad_date,a.age "
141                         + "FROM ads d,animals a WHERE d.kind = ? AND d.owner_id != ? "
142                         + "AND a.age >= ? AND a.age <= ?  AND a.id = d.animal_id UNION ";
143                 String query_2 = "SELECT ad_no,owner_id,animal_id,c.kind,c.price,address,ad_date,b.age "
144                         + "FROM ads c,animals b WHERE c.kind = ? AND c.owner_id != ? "
145                         + "AND b.age >= ? AND b.age <= ? AND b.id = c.animal_id  ORDER BY ad_no";
146                 String finalQuery = query + query_2;
147                 PreparedStatement p = conn.prepareStatement(finalQuery);
148                 p.setString(1, temp1);
149                 p.setInt(2,ids);
150                 p.setInt(3,Integer.parseInt(temp3) );
151                 p.setInt(4, Integer.parseInt(temp4) );
152
153                 p.setString(5, temp2);
154                 p.setInt(6,ids);
155                 p.setInt(7, Integer.parseInt(temp5) );
156                 p.setInt(8, Integer.parseInt(temp6) );
157
158                 try {
159                     ResultSet result = p.executeQuery();
```

9- Sorgulardan en az biri aggragete fonksiyon içermeli, having kullanılmalı : Ads page de bulunan list cheap ve listt expensive butonları ile having ve avg(price) kullanılarak ortalamnın altında ve ortalamanın üstünde olan ilanların listelenmesi sağlanmıştır.



AdvancedSearch.java | AdsPage.java

```
231        btnListExpensive.addActionListener(new ActionListener() {
232            public void actionPerformed(ActionEvent e) {
233                DefaultTableModel model_1 = new DefaultTableModel();
234                model = model_1;
235                model.addColumn("Ad No");
236                model.addColumn("Owner ID");
237                model.addColumn("Kind");
238                model.addColumn("Price");
239                model.addColumn("Address");
240                model.addColumn("Ad Date");
241                model.addColumn("Name");
242                model.addColumn("Age");
243                model.addColumn("Sex");
244                String subquery = "SELECT avg(price) FROM ads where kind = ? group by kind";
245
246                String query = "SELECT ads.ad_no, ads.owner_id, ads.kind, ads.price,ads.address,ads.ad_date,animals.name, animals.age, animals.sex "
247                        + "FROM ads,animals " + "WHERE ads.kind = ? AND ads.animal_id = animals.id AND owner_id != ? "
248                        + "GROUP BY ads.ad_no, ads.owner_id, ads.kind, ads.price,ads.address,ads.ad_date,animals.name , animals.age, animals.sex "
249                        + "HAVING ads.price>=?  order by ad_no;";
250
251                try {
252                    PreparedStatement substt = conn.prepareStatement(subquery);
253                    substt.setString(1, kind);
254                    ResultSet subResult = substt.executeQuery();
```

10- Arayüzden girilen değerleri parametre olarak alan ve ekrana sonuç döndüren en az 3 SQL fonksiyonu. Bu fonksiyonlardan en az biri record ve cursor tanımı kullanmalıdır.

a. Register function : Kullanıcın register ekranında girdiği bilgiler bu fonksiyona verilerek yeni bilgilerin users tablosuna insert edilmesi sağlanmıştır.

```
CREATE OR REPLACE FUNCTION register_user(
    p_username VARCHAR(255),
    p_address VARCHAR(255),
    p_phone VARCHAR(20),
    p_password VARCHAR(255)
)
RETURNS INTEGER AS $$
DECLARE
    new_user_id INTEGER;
BEGIN
    INSERT INTO users(user_name, address, phone_number, password)
    VALUES (p_username, p_address, p_phone, p_password)
    RETURNING id INTO new_user_id;
    RETURN new_user_id;
END;
$$ LANGUAGE plpgsql;
```

Main.java | RegisterPage.java

```
76          frame.getContentPane().add(passwordLabel);
77
78          JButton registerButton = new JButton("Register");
79          registerButton.addActionListener(new ActionListener() {
80              public void actionPerformed(ActionEvent e) {
81                  String query = "SELECT register_user(?,?,?,?);";
82                  PreparedStatement statement;
83                  try {
84                      if (usernameField.getText().length() >= 3 && addressField.getText().length() >= 3
85                              && phoneField.getText().length() >= 3 && passwordField.getText().length() >= 3) {
86                          statement = conn.prepareStatement(query);
87                          statement.setString(1, usernameField.getText());
88                          statement.setString(2, addressField.getText());
89                          statement.setString(3, phoneField.getText());
90                          statement.setString(4, passwordField.getText());
91                          ResultSet r = statement.executeQuery();
```

b. Login function : Kullanıcın girdiği login bilgileri record içine alınarak , login kontrolü için kullanılmıştır.

```
CREATE TYPE login_info AS(user_id INTEGER,password varchar(255));
CREATE OR REPLACE FUNCTION login_user(
    p_user_id INTEGER,
    p_password VARCHAR(255)
)
RETURNS BOOLEAN AS $$
DECLARE
    user_record login_info;
BEGIN
    SELECT id , password INTO user_record
    FROM users
    WHERE id = p_user_id;


    IF user_record IS NOT NULL AND user_record.password = p_password THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
$$ LANGUAGE plpgsql;
```

Main.java | LoginPage.java ×

```
82
83          JButton btnNewButton = new JButton("Login");
84          btnNewButton.setFocusable(false);
85          btnNewButton.addActionListener(new ActionListener() {
86              @SuppressWarnings("deprecation")
87              public void actionPerformed(ActionEvent e) {
88                  String query = "SELECT login_user(?,?);";
89                  try {
90                      PreparedStatement statement = conn.prepareStatement(query);
91                      statement.setInt(1, Integer.parseInt(userIdField.getText()));
92                      statement.setString(2, passwordField.getText());
93
94                      ResultSet r = statement.executeQuery();
```

c. Update price : Cursor kullanılarak istenen ilanın fiyatı verilen yüzdeye göre güncellenir

```
CREATE OR REPLACE FUNCTION update_price_by_percentage(p_ad_no INTEGER, p_percentage INTEGER)
RETURNS INTEGER AS $$
DECLARE
    v_current_price INTEGER;
    v_new_price INTEGER;
    c_ads CURSOR FOR SELECT price FROM ads WHERE ad_no = p_ad_no;
BEGIN
    FOR r_ad IN c_ads LOOP
        v_current_price := r_ad.price;
        v_new_price := v_current_price * p_percentage / 100;

        UPDATE ads SET price = v_new_price WHERE ad_no = p_ad_no;
    END LOOP;
    RETURN v_new_price;
END;
$$ LANGUAGE plpgsql;
```

Main.java | ChangeAnimalPrice.java ×

```
134                 }
135                 if (check != -1) {
136                     try {
137                         // Kullanıcıdan ad_no ve yüzdelik değerleri al
138                         int percentage = Integer.parseInt(temp2);
139
140                         query = "SELECT update_price_by_percentage(?, ?) as new_price";
141                         try (PreparedStatement statement = conn.prepareStatement(query)) {
142                             statement.setInt(1, adNo);
143                             statement.setInt(2, percentage);
144
145                             try (ResultSet resultSet = statement.executeQuery()) {
146                                 if (resultSet.next()) {
147                                     int newPrice = resultSet.getInt("new_price");
148                                     JOptionPane.showMessageDialog(null,
149                                             "Price updated successfully! New price: " + newPrice);
150                                     updateList();
151                                 }
```

11- 2 adet trigger tanımlı olmalı. Arayüzden girilen değerler ile tetiklenmeli.

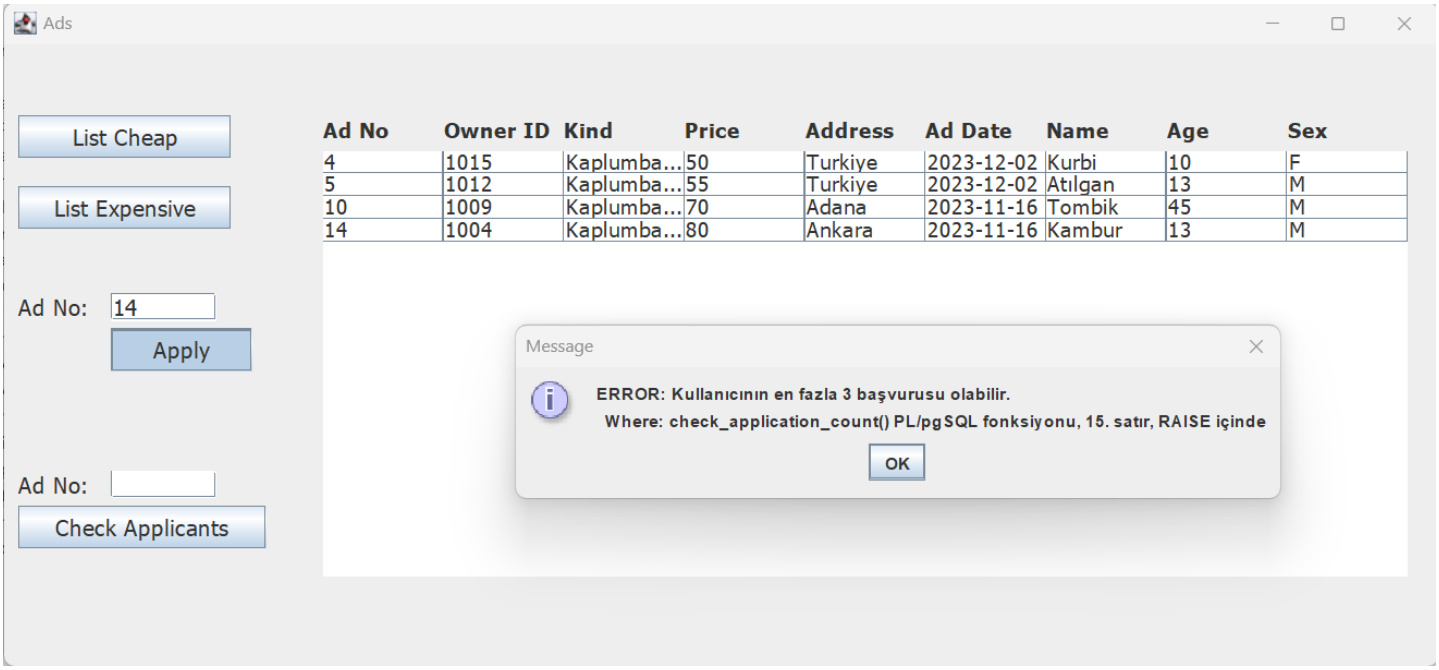
a. İlan sayısı trigger ı : Kullanıcı ilana başvurmaya çalışınca tetiklenen trigger sonucu trigger fonksiyonu kullanıcının başvuru sayısını kontrol eder ve 3 ten azsa başvuruya izin verir.

```
CREATE TRIGGER before_application_insert
BEFORE INSERT ON applications
FOR EACH ROW
EXECUTE FUNCTION check_application_count();
```

```
CREATE OR REPLACE FUNCTION check_application_count()
RETURNS TRIGGER AS $$
DECLARE
    applicent_id_val INT;
    ad_no_val INT;
    application_count INT;
BEGIN
    applicent_id_val := NEW.applicent_id;

    SELECT COUNT(*)
    INTO application_count
    FROM applications
    WHERE applicent_id = applicent_id_val;

    IF application_count >= 3 THEN
        RAISE EXCEPTION 'Kullanıcının en fazla 3 başvurusu olabilir.';
        RETURN NULL;
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```



b. Maks fiyat kontrolü : Kullanıcı ilan verirken 10.000 TL üzerinde ilan verememesi trigger kullanılarak sağlanmıştır.

```
CREATE TRIGGER check_price_trigger
BEFORE INSERT OR UPDATE ON ads
FOR EACH ROW
EXECUTE FUNCTION check_price();
```

```
CREATE OR REPLACE FUNCTION check_price()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.price > 10000 THEN
        RAISE EXCEPTION 'Trigger! Cant put ad price higher than 10.000!';
        RETURN NULL;
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```