VERİ TABANI YÖNETİM SİSTEMİ
PROJE ÖDEVİ

Dr.Öğr.Üyesi İSMAİL ÖZTEL

Öğrenci Adı : Manar AL SAYED ALI
Öğrenci NO : G221210558
Öğrenci Eposta Adresi : manar.ali@ogr.sakarya.edu.tr

PROBLEM TANITIMI :

Film dünyasındaki zenginliğin içinde kaybolmak zaman zaman zor olabilir. İşte bu noktada uygulamamız devreye giriyor. Film Veritabanı ve Öneri Sistemi, kullanıcılarına istedikleri filmi bulma ve yeni içerikleri keşfetme konusunda rehberlik ediyor. Güçlü arama ve filtreleme seçenekleri ile kullanıcılar, kişisel tercihlerine uygun filmleri anında bulabilirler.

SENARYO :

Uygulamamızın amacı, kullanıcıların istedikleri filmleri kolayca bulmalarını sağlamak. Üretim şirketlerinin ürettiği filmlerin bilgileri kaydedilecek. Filmin türü, yönetmeni ve oyuncuların isimleri de dahil olmak üzere birçok bilgi kaydedilecek. Kullanıcılar istedikleri filmi değerlendirebilecek. Her kullanıcıya ait farklı 2 liste oluşturulacak: izleme listesi ve favorite, filmlerin ve Kullanıcıların id’lerini tutulacaklar. Kullanıcılar istedikleri filmi ekleyip silebilecekler.

İŞ KURALLARI :

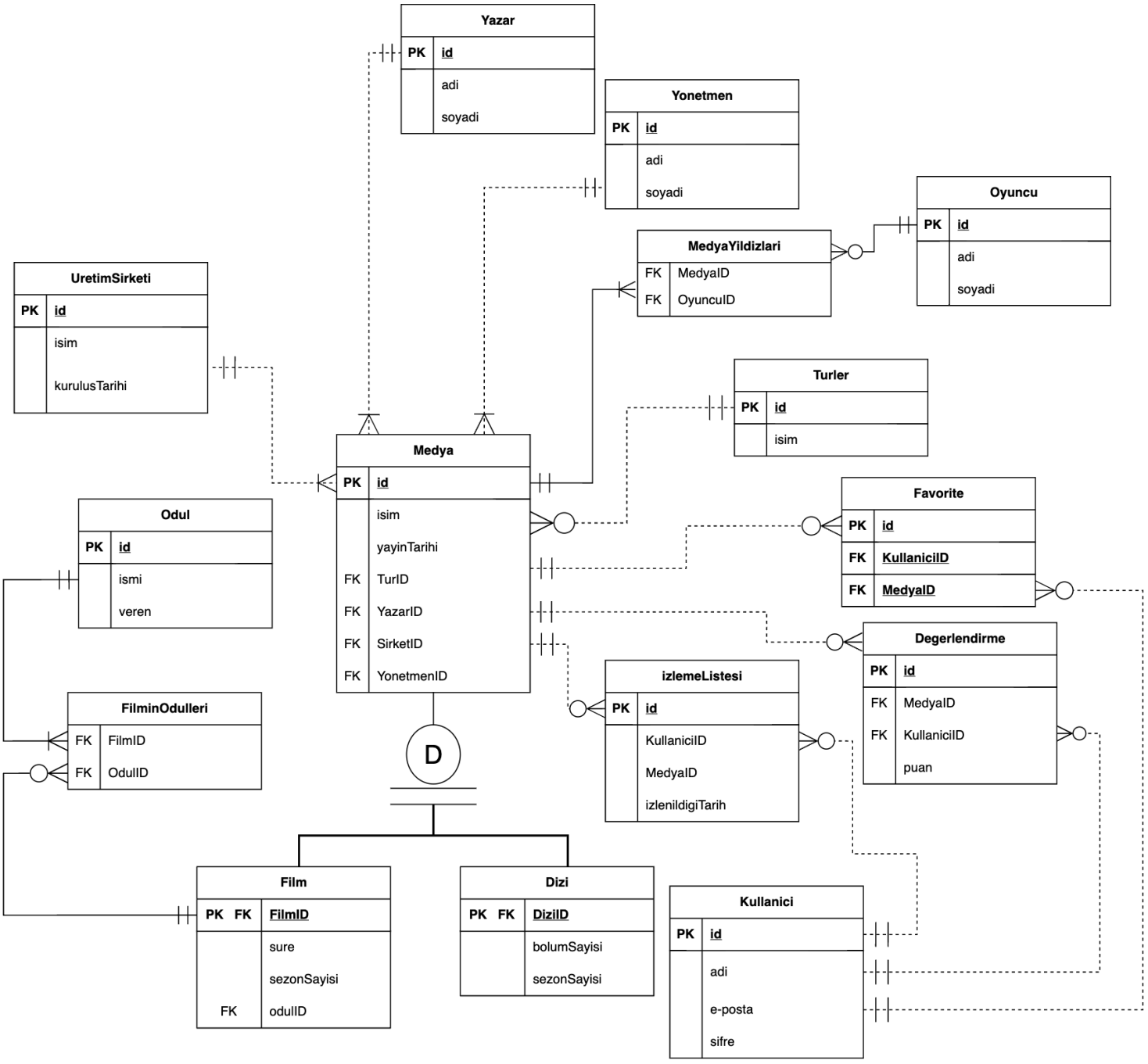
- Her kullanıcı, benzersiz bir e-posta adresi ile kayıt olmalıdır.
- üretim şirketin id kodu, ismi ve kuruluş tarihi bulunmaktadır.
- Her bir üretim şirketin kendi ürettiği medyaya sahip, Her bir üretim şirketi birden fazla medyaya sahip olabilir, en az bir medyaya sahip olması gerekiyor. Bir medya ise yalnızca bir üretim şirkete ait olabilir.
- Yönetmenin id kodu, adi ve soyadı bulunmaktadır.
- Oyuncunun id kodu, adi ve soyadı bulunmaktadır.
- Medya dizi ya da film olabilir, ikisi aynı anda olamaz başka bir şey de olamaz
- Her medyanın kodu, ismi, turu, yazarı, yönetmeni, yıldızları ve üretim şirketi bulunacaktır.
- Film ve dizi medyadan kalıtım özelliğiyle bilgileri alır.
- Filmin süresi ve sezon sayısı da bilgilere eklenir, ayrıca film ödül alabilir, her film birden fazla ödül alabilir hiç bir ödül de almayabilir.
- Dizinin bölüm ve sezon sayısı da bulunacak.
- Kullanıcıların id kodu, ismi, e-postası, şifresi, beğendiği, izlediği ya da izlemek istediği filmlerin isimleri bulunacak.
- Medya yönetmen tarafından yönetilir. Bir yönetmen en az bir olmak üzere birden fazla medya yönetebilir. Bir medya ise sadece bir yönetmene ait olabilir.

- Medya yazar tarafından yazılır. Bir yazar en az bir olmak üzere birden fazla medya yazabilir. Bir medya ise sadece bir yazara ait olabilir.
- Medyaya ait birden fazla oyuncu olabilir, en az bir oyuncu. Oyuncu birden fazla medyada oynamış olabilir, en az bir medyada oynamış olması gerekiyor.
- Kullanıcı istediği medyayı ya da medyaları değerlendirebilir, isterse hiç bir medyayı değerlendirmeyebilir. Medya birden fazla kullanıcı tarafından değerlendirilebilir, hiç bir kullanıcı tarafından da değerlendirilmeyebilir.

İlişkisel şema :

1. UretimSirketi (id: int, isim: varchar, kurulusTarihi: date)
2. Kullanici (id: int, adi: varchar, e-posta: varchar, sifre: varchar)
3. Medya (medyaid: int, isim: varchar, yayinTarihi: date, TurID: int, YazarID: int, SirketID: int, YonetmenID: int)
4. Film (FilmID: int, sure: int, sezonSayisi: int, odulID: int)
5. Dizi (DiziID: int, bolumSayisi: int, sezonSayisi: int)
6. Yonetmen (id: int, adi: varchar, soyadı: varchar)
7. Yazar (id: int, adi: varchar, soyadı: varchar)
8. Oyuncu (id: int, adi: varchar, soyadı: varchar)
9. MedyaYildizlari (MedyaID: int, OyuncuID:int)
10. Turler (id: int, isim: varchar)
11. Favorite (id: int, MedyaID: int, KullanicID: int)
12. İzleme listesi (id: int, MedyaID: int, KullanicID: int, izlenildigiTarih: date)
13. Degerlendirme (id: int, MedyaID: int, KullanicID: int, puan: int)
14. Ödül (id: int, isim: varchar, veren: varchar)
15. FilminOdulleri (FilmID: int, OdulID: int)

Varlık Bağıntı modeli :



Veritabanını, içerisindeki verilerle birlikte oluşturmayı sağlayan SQL ifadeleri :

```
CREATE TABLE "Medya"."Medya" (
        "id" SERIAL,
        "isim" CHARACTER VARYING(80) NOT NULL,
        "yayinTarihi" DATE,
        "MedyaTipi" CHARACTER(1) NOT NULL,
        CONSTRAINT "MedyaPK" PRIMARY KEY ("id")
);

CREATE TABLE "Medya"."Film" (
        "id" INT,
        "suresi" int,
        "sezonSayisi" int not null DEFAULT 1,
        CONSTRAINT "FilmPK" PRIMARY KEY ("id")
);

CREATE TABLE "Medya"."Dizi" (
        "id" INT,
        "bolumSayisi" int,
        "sezonSayisi" int not null DEFAULT 1,
        CONSTRAINT "DiziPK" PRIMARY KEY ("id")
);

ALTER TABLE "Medya"."Film"
        ADD CONSTRAINT "MedyaFilm" FOREIGN KEY ("id")
        REFERENCES "Medya"."Medya" ("id")
        ON DELETE CASCADE
        ON UPDATE CASCADE;

ALTER TABLE "Medya"."Dizi"
        ADD CONSTRAINT "MedyaDizi" FOREIGN KEY ("id")
        REFERENCES "Medya"."Medya" ("id")
        ON DELETE CASCADE
        ON UPDATE CASCADE;

CREATE TABLE "Kullanici" (
   "id" SERIAL PRIMARY KEY NOT NULL,
   "adi" VARCHAR(40) NOT NULL,
   "e-posta" VARCHAR(40) NOT NULL,
   "sifre" VARCHAR(15) NOT NULL
) ;

CREATE TABLE "UretimSirketi" (
        "id" SERIAL PRIMARY KEY NOT NULL,
        "isim" VARCHAR(40) NOT NULL,
        "KurulusTarihi" DATE
) ;
```

```
CREATE TABLE "Yonetmen" (
        "id" SERIAL PRIMARY KEY NOT NULL,
        "isim" VARCHAR(40) NOT NULL,
        "soyadi" VARCHAR(40)
) ;

CREATE TABLE "Yazar" (
        "id" SERIAL PRIMARY KEY NOT NULL,
        "isim" VARCHAR(40) NOT NULL,
        "soyadi" VARCHAR(40)
) ;

CREATE TABLE "Oyuncu" (
        "id" SERIAL PRIMARY KEY NOT NULL,
        "isim" VARCHAR(40) NOT NULL,
        "soyadi" VARCHAR(40)
) ;

CREATE TABLE "Turler" (
        "id" SERIAL PRIMARY KEY NOT NULL,
        "isim" VARCHAR(15) NOT NULL
) ;

CREATE TABLE "Odul" (
        "id" SERIAL PRIMARY KEY NOT NULL,
        "isim" VARCHAR(40) NOT NULL,
        "veren" VARCHAR(40)
) ;

ALTER TABLE "Medya"."Medya"
   add "TurID" INT NOT NULL,
   add "YazarID" INT NOT NULL,
   add "SirketID" INT,
   add "YonetmenID" INT NOT NULL,
   add CONSTRAINT "FK_Medya_Tur" FOREIGN KEY("TurID") REFERENCES "Turler"("id"),
   add CONSTRAINT "FK_Medya_Yazar" FOREIGN KEY("YazarID") REFERENCES "Yazar"("id"),
   add CONSTRAINT "FK_Medya_Sirket" FOREIGN KEY("SirketID") REFERENCES "UretimSirketi"("id"),
   add CONSTRAINT "FK_Medya_Yonetmen" FOREIGN KEY("YonetmenID") REFERENCES "Yonetmen"("id")

CREATE TABLE "izlemeListesi" (
   "id" SERIAL PRIMARY KEY NOT NULL,
   "izlenildigiTarihi" DATE,
   "KullaniciID" INT NOT NULL,
   "MedyaID" INT NOT NULL,

   CONSTRAINT "FK_Kullanici" FOREIGN KEY("KullaniciID") REFERENCES "Kullanici"("id"),
   CONSTRAINT "FK_Medya" FOREIGN KEY("MedyaID") REFERENCES "Medya"."Medya"("id")
);

CREATE TABLE "Favorite" (
   "id" SERIAL PRIMARY KEY NOT NULL,
   "KullaniciID" INT NOT NULL,
   "MedyaID" INT NOT NULL,

   CONSTRAINT "FK_Kullanici" FOREIGN KEY("KullaniciID") REFERENCES "Kullanici"("id"),
   CONSTRAINT "FK_Medya" FOREIGN KEY("MedyaID") REFERENCES "Medya"."Medya"("id")
);
```

```
CREATE TABLE "Degerlendirme" (
    "id" SERIAL PRIMARY KEY NOT NULL,
    "puan" INT,
    "KullaniciID" INT NOT NULL,
    "MedyaID" INT NOT NULL,

    CONSTRAINT "FK_Kullanici" FOREIGN KEY("KullaniciID") REFERENCES "Kullanici"("id"),
    CONSTRAINT "FK_Medya" FOREIGN KEY("MedyaID") REFERENCES "Medya"."Medya"("id")
);

CREATE TABLE "MedyaYildizlari" (
    "OyuncuID" INT NOT NULL,
    "MedyaID" INT NOT NULL,

    CONSTRAINT "FK_Oyuncu" FOREIGN KEY("OyuncuID") REFERENCES "Oyuncu"("id"),
    CONSTRAINT "FK_Medya" FOREIGN KEY("MedyaID") REFERENCES "Medya"."Medya"("id")
);

CREATE TABLE "FilminOduller" (
    "FilmID" INT NOT NULL,
    "OdulID" INT NOT NULL,

    CONSTRAINT "FK_Film" FOREIGN KEY("FilmID") REFERENCES "Medya"."Film"("id"),
    CONSTRAINT "FK_Odul" FOREIGN KEY("OdulID") REFERENCES "Odul"("id")
);

INSERT INTO "Medya"."Medya" ("isim", "yayinTarihi", "MedyaTipi", "TurID", "YazarID", "SirketID",
"YonetmenID")
VALUES ('Örnek Medya', '2023-01-01', 'F', 1, 1, 1, 1);
```

Yazdığım Fonksiyonlar :

1.Kullanıcı ekleme:

```
CREATE OR REPLACE FUNCTION public.add_user(user_name character varying, user_email character
varying, user_password character varying)
 RETURNS void
 LANGUAGE plpgsql
AS $function$
BEGIN
    INSERT INTO "Kullanici" ("adi", "e-posta", "sifre")
    VALUES (user_name, user_email, user_password);
END;
$function$
```

2.Medya Silme Fonksiyonu:

```
CREATE OR REPLACE FUNCTION public.delete_media_with_ratings(media_id integer)
 RETURNS void
 LANGUAGE plpgsql
AS $function$
BEGIN
    -- "MedyaYildizlari" tablosundan ilişkili kayıtları sil
    DELETE FROM "MedyaYildizlari" WHERE "MedyaID" = media_id;

    -- "Medya" tablosundan kaydı sil
    DELETE FROM "Medya"."Medya" WHERE "id" = media_id;
END;
$function$
```

3.Kullanıcı arama:

```
CREATE OR REPLACE FUNCTION public.search_user(user_name character varying)
 RETURNS TABLE(id integer, adi character varying, "e-posta" character varying, sifre character varying)
 LANGUAGE plpgsql
AS $function$
BEGIN
    RETURN QUERY
    SELECT "id", "adi", "e-posta", "sifre"
    FROM "Kullanici"
    WHERE "adi" ILIKE '%' || user_name || '%';
END;
$function$
```

4.Medya Güncelleme Fonksiyonu:

```
CREATE OR REPLACE FUNCTION public.update_media(media_id integer, media_name character varying, media_type character, media_genre integer, media_author integer, media_sirket integer, media_director integer)
 RETURNS void
 LANGUAGE plpgsql
AS $function$
BEGIN
    UPDATE "Medya"."Medya"
    SET
        "isim" = media_name,
        "MedyaTipi" = media_type,
        "TurID" = media_genre,
        "YazarID" = media_author,
        "SirketID" = media_sirket,
        "YonetmenID"= media_director

    WHERE
        "id" = media_id;
END;
$function$
```

Triggers (Tetikleyici):

1.Check password:

```
CREATE OR REPLACE FUNCTION public.check_password_strength()
 RETURNS trigger
 LANGUAGE plpgsql
AS $function$
BEGIN
    IF NEW."sifre" IS NOT NULL THEN
        IF LENGTH(NEW."sifre") < 8 OR NOT (
            (NEW."sifre" ~ '\d') AND   -- En az bir rakam içermeli
            (NEW."sifre" ~ '[A-Z]') AND  -- En az bir büyük harf içermeli
            (NEW."sifre" ~ '[a-z]')     -- En az bir küçük harf içermeli
        ) THEN
            RAISE EXCEPTION 'Şifre güvenlik kriterlerini karşılamıyor.';
        END IF;
    END IF;
    RETURN NEW;
END;
$function$
```

2.Benzersiz Yönetmen adı:

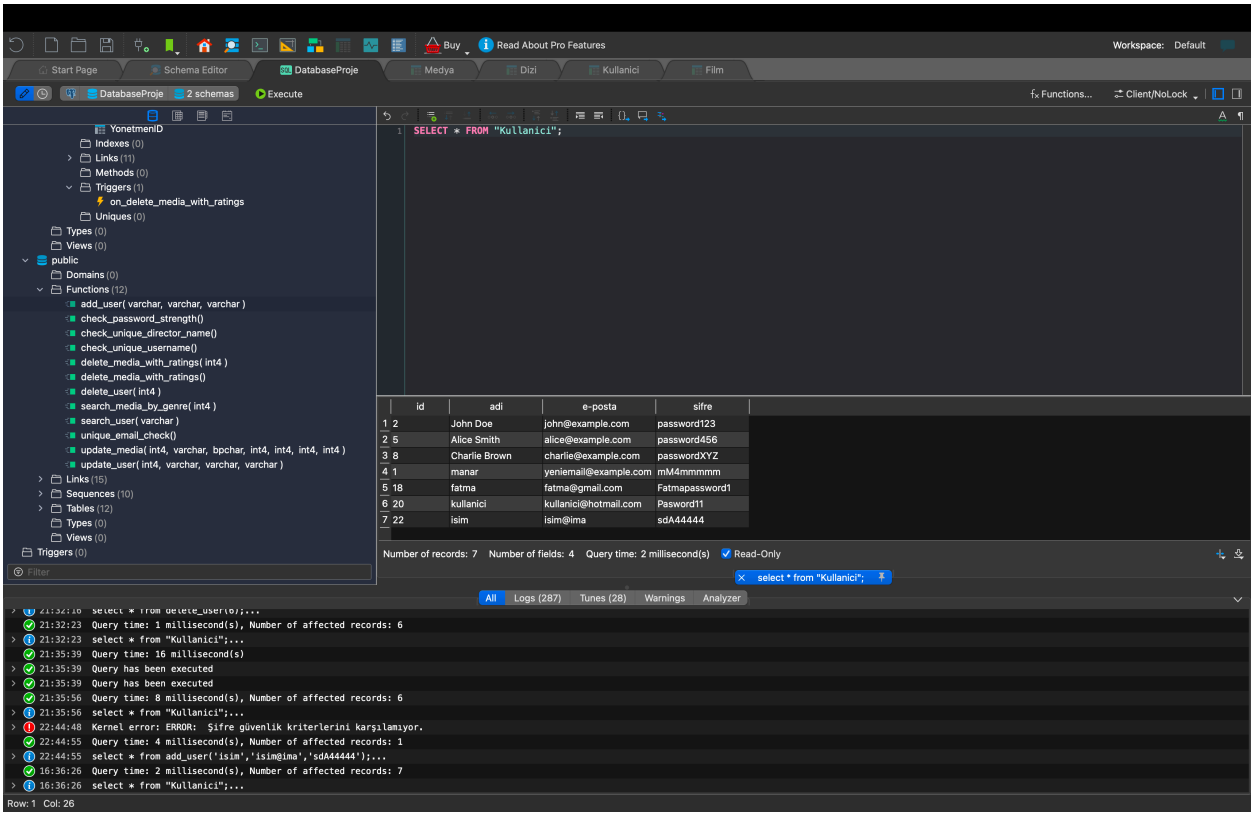
```
CREATE OR REPLACE FUNCTION public.check_unique_director_name()
 RETURNS trigger
 LANGUAGE plpgsql
AS $function$
BEGIN
    IF NEW."director_name" IS NOT NULL THEN
        IF EXISTS (SELECT 1 FROM "Yonetmen" WHERE "director_name" = NEW."director_name") THEN
            RAISE EXCEPTION 'Bu yönetmen adı zaten kullanılmaktadır.';
        END IF;
    END IF;
    RETURN NEW;
END;
$function$
```

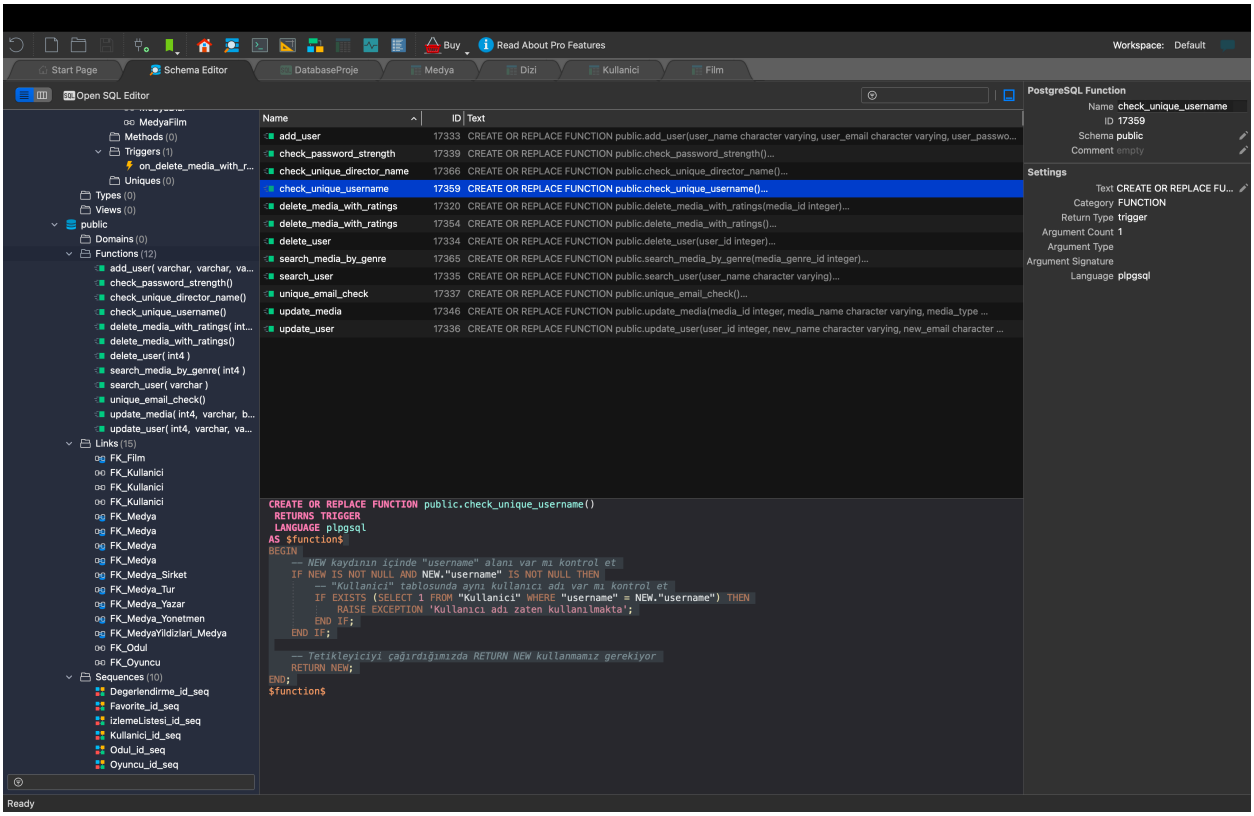
3.Medya ve yıldızları silme:

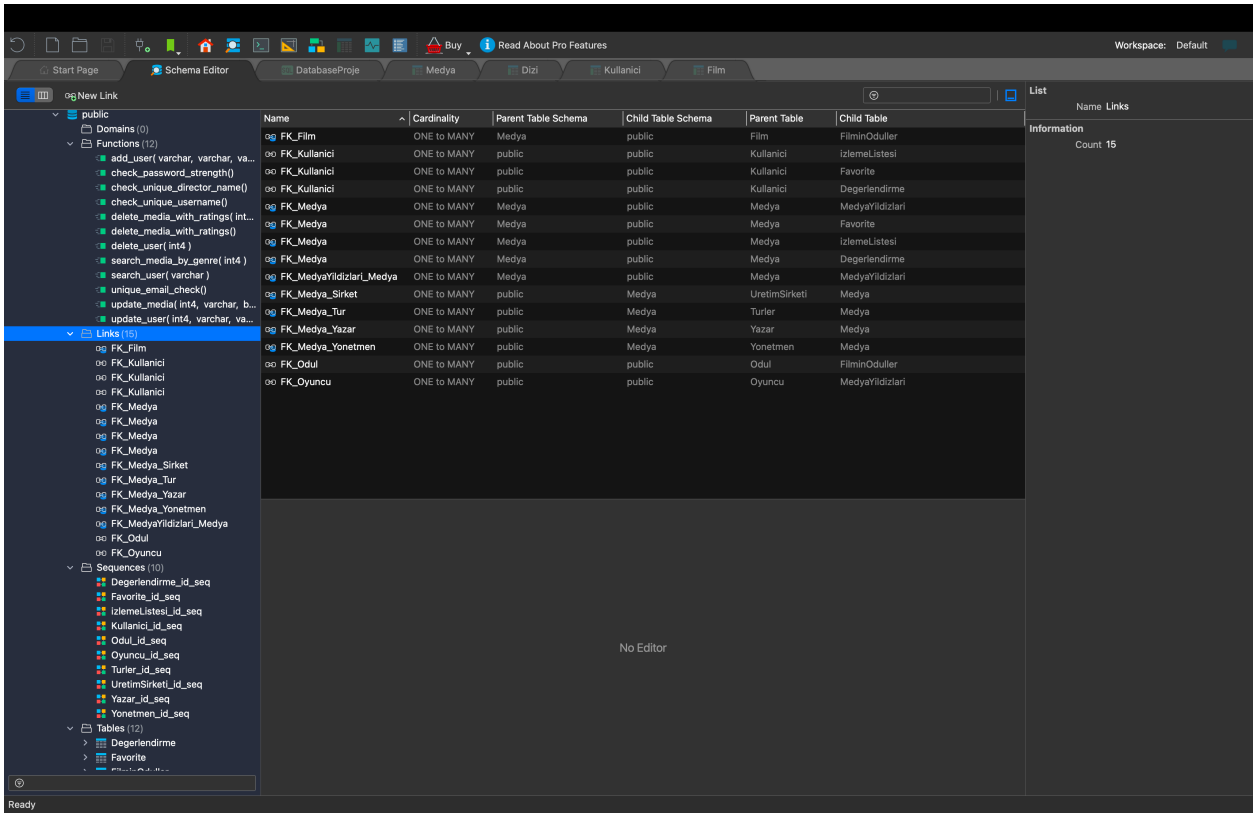
```
CREATE OR REPLACE FUNCTION public.delete_media_with_ratings()
 RETURNS trigger
 LANGUAGE plpgsql
AS $function$
BEGIN
    DELETE FROM "MedyaYildizlari" WHERE "MedyaID" = OLD."id";
    RETURN OLD;
END;
$function$
```

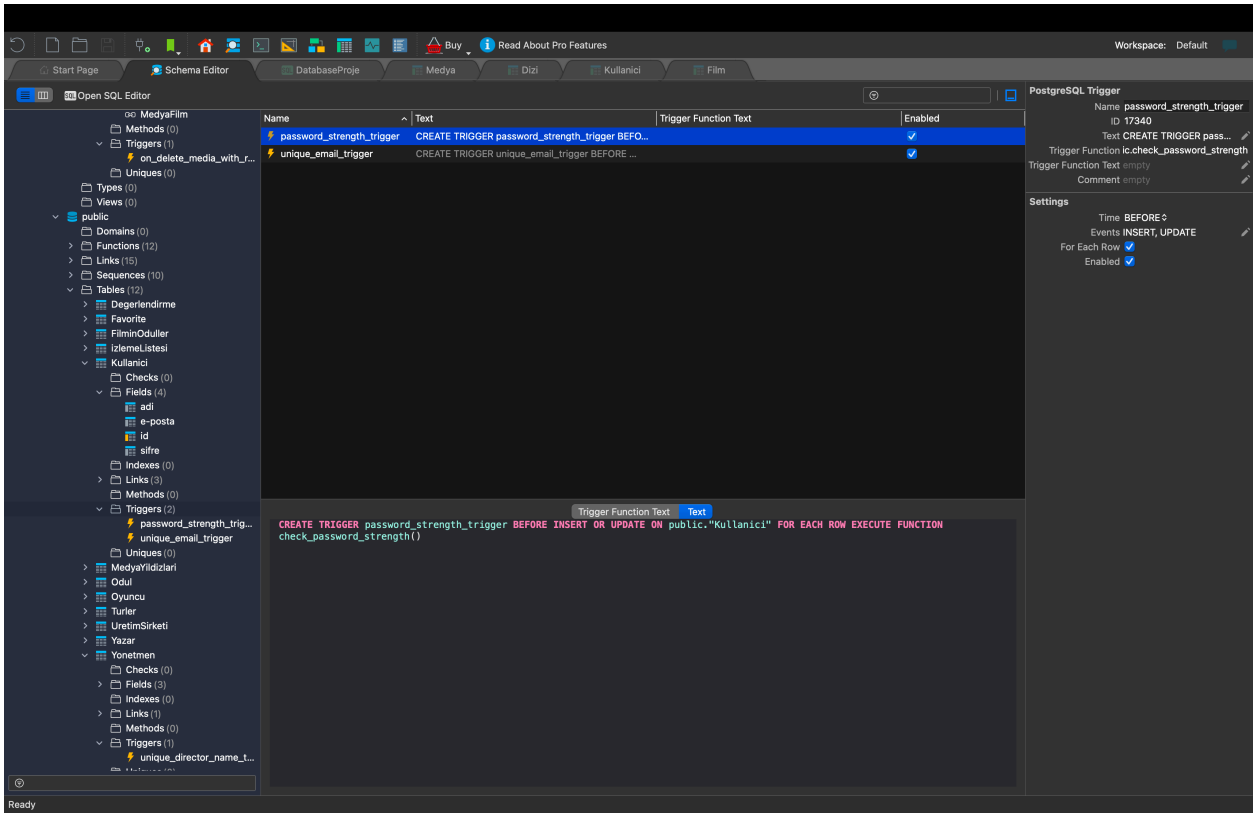
4.Benzersiz e-mail:

```
CREATE OR REPLACE FUNCTION public.unique_email_check()
 RETURNS trigger
 LANGUAGE plpgsql
AS $function$
BEGIN
    IF NEW."e-posta" IS NOT NULL THEN
        IF EXISTS (
            SELECT 1
            FROM "Kullanici"
            WHERE "e-posta" = NEW."e-posta"
            AND "id" <> NEW."id"  -- Mevcut kullanıcının güncellenmesi durumunda kendi e-postasını kontrol
etme
        ) THEN
            RAISE EXCEPTION 'Bu e-posta zaten kullanımda.';
        END IF;
    END IF;
    RETURN NEW;
END;
$function$
```


SELECT * FROM "Kullanici";
id adi e-posta sifre
1 2 John Doe john@example.com password123
2 5 Alice Smith alice@example.com password456
3 8 Charlie Brown charlie@example.com passwordXYZ
4 1 manar yeniemail@example.com mM4mmmmm
5 18 fatma fatma@gmail.com Fatmapassword1
6 20 kullanici kullanici@hotmail.com Pasword11
7 22 isim isim@ima sdA44444
Number of records: 7 Number of fields: 4 Query time: 2 millisecond(s) Read-Only
21:32:23 Query time: 1 millisecond(s), Number of affected records: 6
21:32:23 select * from "Kullanici";...
21:35:39 Query time: 16 millisecond(s)
21:35:39 Query has been executed
21:35:39 Query has been executed
21:35:56 Query time: 8 millisecond(s), Number of affected records: 6
21:35:56 select * from "Kullanici";...
22:44:48 Kernel error: ERROR: Şifre güvenlik kriterlerini karşılamıyor.
22:44:55 Query time: 4 millisecond(s), Number of affected records: 1
22:44:55 select * from add_user('isim','isim@ima','sdA44444');...
16:36:26 Query time: 2 millisecond(s), Number of affected records: 7
16:36:26 select * from "Kullanici";...


CREATE OR REPLACE FUNCTION public.check_unique_username()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $function$
BEGIN
-- NEW kaydının içinde "username" alanı var mı kontrol et
IF NEW IS NOT NULL AND NEW."username" IS NOT NULL THEN
-- "Kullanici" tablosunda aynı kullanıcı adı var mı kontrol et
IF EXISTS (SELECT 1 FROM "Kullanici" WHERE "username" = NEW."username") THEN
RAISE EXCEPTION 'Kullanıcı adı zaten kullanılmakta';
END IF;
END IF;
-- Tetikleyiciyi çağırdığımızda RETURN NEW kullanmamız gerekiyor
RETURN NEW;
END;
$function$


Buy
Read About Pro Features
Workspace: Default
Start Page
Schema Editor
DatabaseProje
Medya
Dizi
Kullanici
Film
New Link
public
Domains (0)
Functions (12)
add_user(varchar, varchar, va...
check_password_strength()
check_unique_director_name()
check_unique_username()
delete_media_with_ratings(int...
delete_media_with_ratings()
delete_user(int4)
search_media_by_genre(int4)
search_user(varchar)
unique_email_check()
update_media(int4, varchar, b...
update_user(int4, varchar, va...
Links (15)
FK_Film
FK_Kullanici
FK_Kullanici
FK_Kullanici
FK_Medya
FK_Medya
FK_Medya
FK_Medya
FK_Medya_Sirket
FK_Medya_Tur
FK_Medya_Yazar
FK_Medya_Yonetmen
FK_MedyaYildizlari_Medya
FK_Odul
FK_Oyuncu
Sequences (10)
Degerlendirme_id_seq
Favorite_id_seq
izlemeListesi_id_seq
Kullanici_id_seq
Odul_id_seq
Oyuncu_id_seq
Turler_id_seq
UretimSirketi_id_seq
Yazar_id_seq
Yonetmen_id_seq
Tables (12)
Degerlendirme
Favorite
Name
Cardinality
Parent Table Schema
Child Table Schema
Parent Table
Child Table
FK_Film ONE to MANY Medya public Film FilminOduller
FK_Kullanici ONE to MANY public public Kullanici izlemeListesi
FK_Kullanici ONE to MANY public public Kullanici Favorite
FK_Kullanici ONE to MANY public public Kullanici Degerlendirme
FK_Medya ONE to MANY Medya public Medya MedyaYildizlari
FK_Medya ONE to MANY Medya public Medya Favorite
FK_Medya ONE to MANY Medya public Medya izlemeListesi
FK_Medya ONE to MANY Medya public Medya Degerlendirme
FK_MedyaYildizlari_Medya ONE to MANY Medya public Medya MedyaYildizlari
FK_Medya_Sirket ONE to MANY public Medya UretimSirketi Medya
FK_Medya_Tur ONE to MANY public Medya Turler Medya
FK_Medya_Yazar ONE to MANY public Medya Yazar Medya
FK_Medya_Yonetmen ONE to MANY public Medya Yonetmen Medya
FK_Odul ONE to MANY public public Odul FilminOduller
FK_Oyuncu ONE to MANY public public Oyuncu MedyaYildizlari
No Editor
List
Name Links
Information
Count 15
Ready


Buy
Read About Pro Features
Workspace: Default
Start Page
Schema Editor
DatabaseProje
Medya
Dizi
Kullanici
Film
Open SQL Editor
MedyaFilm
Methods (0)
Triggers (1)
on_delete_media_with_r...
Uniques (0)
Types (0)
Views (0)
public
Domains (0)
Functions (12)
Links (15)
Sequences (10)
Tables (12)
Degerlendirme
Favorite
FilminOduller
izlemeListesi
Kullanici
Checks (0)
Fields (4)
adi
e-posta
id
sifre
Indexes (0)
Links (3)
Methods (0)
Triggers (2)
password_strength_trig...
unique_email_trigger
Uniques (0)
MedyaYildizlari
Odul
Oyuncu
Turler
UretimSirketi
Yazar
Yonetmen
Checks (0)
Fields (3)
Indexes (0)
Links (1)
Methods (0)
Triggers (1)
unique_director_name_t...
Name
Text
Trigger Function Text
Enabled
password_strength_trigger CREATE TRIGGER password_strength_trigger BEFO...
unique_email_trigger CREATE TRIGGER unique_email_trigger BEFORE ...
Trigger Function Text
Text
CREATE TRIGGER password_strength_trigger BEFORE INSERT OR UPDATE ON public."Kullanici" FOR EACH ROW EXECUTE FUNCTION
check_password_strength()
PostgreSQL Trigger
Name password_strength_trigger
ID 17340
Text CREATE TRIGGER pass...
Trigger Function ic.check_password_strength
Trigger Function Text empty
Comment empty
Settings
Time BEFORE
Events INSERT, UPDATE
For Each Row
Enabled
Ready