Senaryo

Gerçek dünya problemini çözen yazılım geliştirmesi istenmektedir. Geliştirilen uygulamanın veri tabanı içerisinde işlev (fonksiyon)/saklı yordam (stored procedure) ve tetikleyici (trigger) kullanılmalıdır. Veri tabanı içerisinde en az 15 tablo yer almalıdır. En az 4 adet işlev/saklı yordam (tetikleyiciler için tanımlananlar hariç) ve 4 adet tetikleyici tanımlanmalıdır.

Visual Studio'da C# kullanarak diş hastanesi doktor arayüzü yaptım.

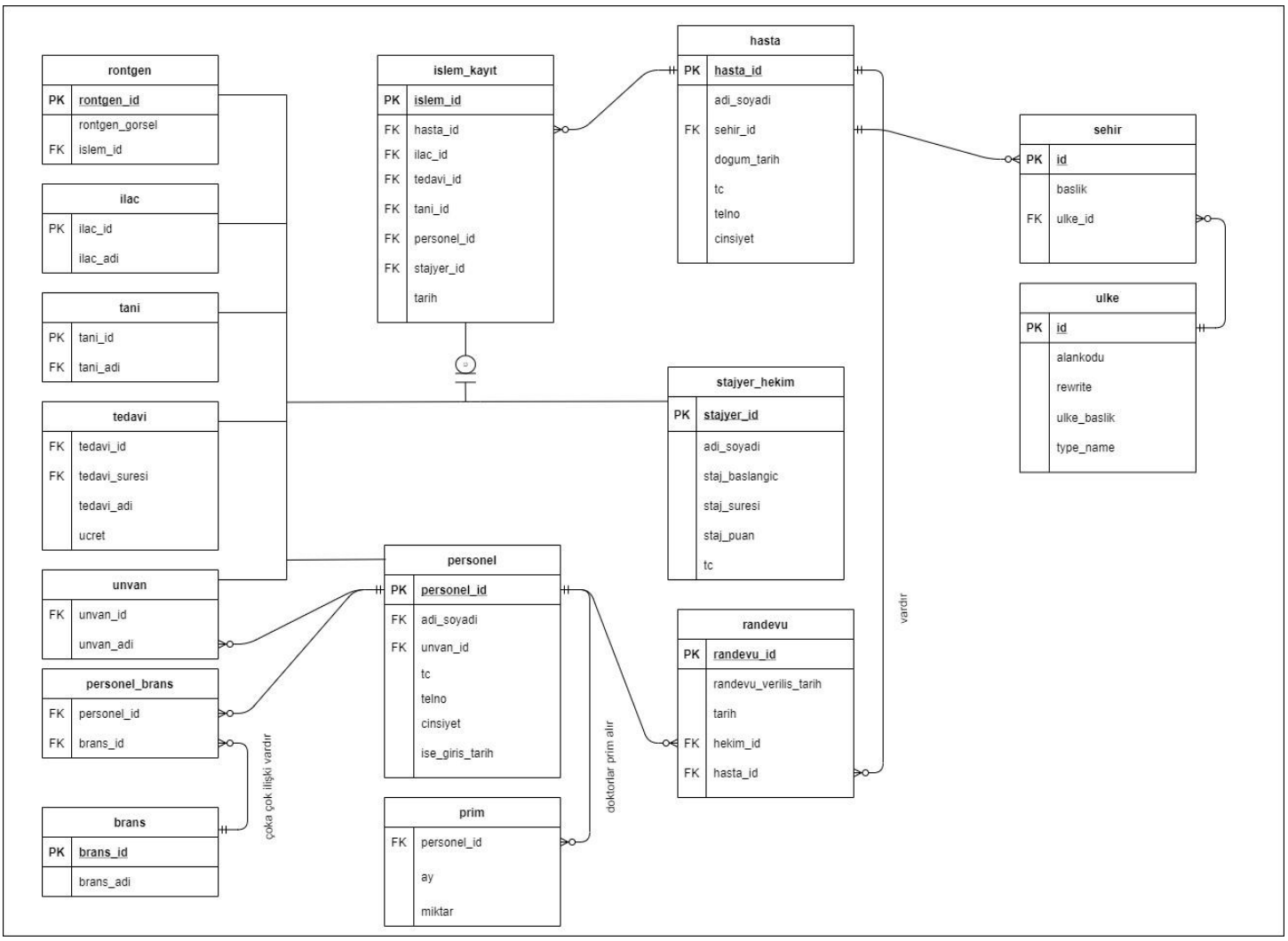
İş Kuralları

- Bu veritabanında her kullanıcının hastanın bir id numarası vardır. Hasta kaydı yapılırken ad, tc, telno, vb. bilgilerini girmesi beklenir.
- Hasta birden çok muayene olabilir.
- Bir hastaya birden çok doktor bakabilir.
- Bir tedavi birden çok işlemde olabilir ama bir işlemde birden çok tedavi kaydı yapılamaz.
- Bir tanı birden çok işlemde olabilir ama bir işlemde birden çok tanı kaydı yapılamaz.
- İşlem kaydı yapılırken ilaç adı hariç boş alan bırakılmaması beklenir.
- Hasta randevuya gelmezse, randevu kaydı silinir.
- Doktor veya stajyer randevu verebilir, iptal edebilir.
- Yeni hasta kaydı eklenebilir.
- Olan hasta kaydı üzerinde güncelleme yapılabilir.
- Hasta kaydı silinebilir.
- Doktorların birden çok branşı olabilir.
- Bir branş birden çok doktorda bulunabilir.
- İşlemlerde uygulanan tedavilerin süreleri vardır. Randevular bu sürelere göre verilir.
- Tedavilerin ücretleri vardır.
- Her uygulanan işlemde doktor ücrete göre prim alır. Ay sonuna kadar hesaplanır.
- Her uygulanan işlemde stajyer puan alır.
- Stajyerlerin staj süresi bitince kayıtları silinir.
- Geriye dönük randevu verilemez.

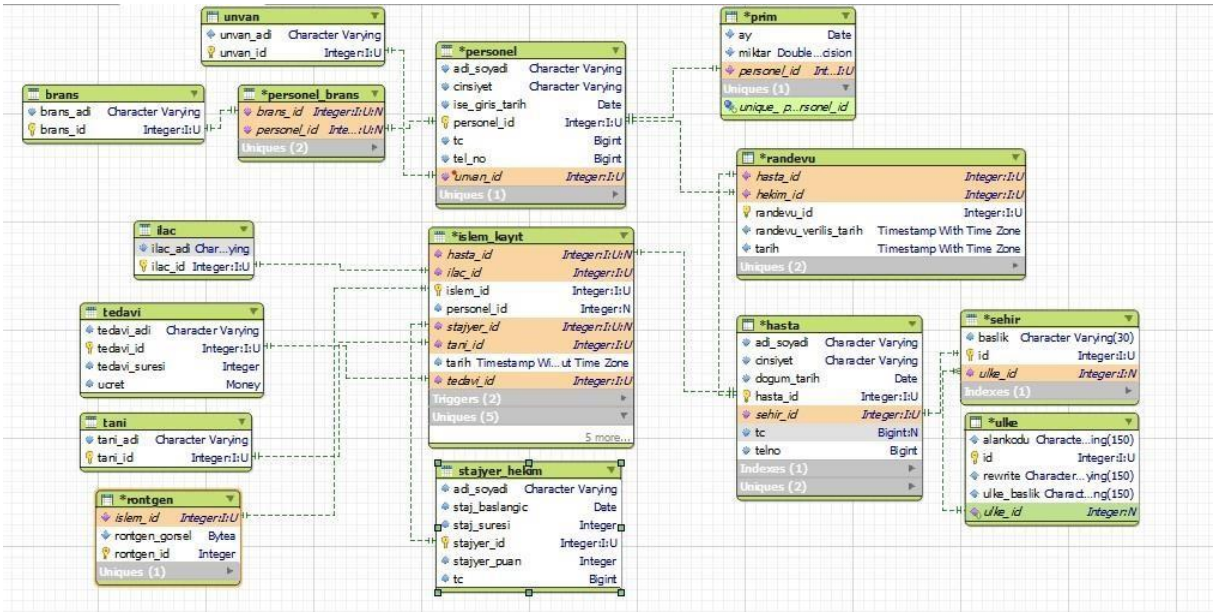
İlişkisel Şema (metinsel gösterim)

1. personel(**personel_id:int**, adi_soyadi: varchar ,unvan_id:int ,tc :bigint ,telno :varchar , cinsiyet: varchar , ise_giris_tarih: timestamp)
2. hasta(**hasta_id: int,sehir_id:int ,**adi_soyadi: varchar,tc :bigint ,telno :varchar , cinsiyet: varchar , dogum _tarih: timestamp)
3. stajyer_hekim(**stajyer_id: int ,**_ adi_soyadi: varchar ,time :date, staj_ baslangic :int, staj_suresi:int, staj_puan:int, tc:bigint)
4. işlem_kayıt (**işlem_id :int , hasta_id:int personel_id:int,tani_id:int,tedavi_id:int,ilaç_id:int,** tarih: timestamp)
5. ilac(**ilac_id: int,** ilaç_adi:varchar)
6. tani(**tani_id: int,**tani_id:int)
7. tedavi(**tedavi_id: int ,** tedavi_suresi:int,tedavi_adi:varchar,ucret:int)
8. unvan (**unvan_id: int,** unvan_adi:varchar)
9. brans(**brans_id: int,**branş_adi:varchar)
10. **personel_brans(brans_id: int,personel_id: int**)
11. sehir(**id: int, ulke_id: int, sehir_adi:varchar**)
12. ulke(**id: int ,alan_kodu: varchar, rewrite**: varchar,ulke_baslik: varchar,type_name:varchar)
13. randevu(**randevu_id_id: int, hekim_id:int,hasta_id:int** ,randevu_verilis_tarih:timestamp,tarih:timestamp)
14. rontgen(**röntgen_id: int, işlem_id:int** ,röntgen_gorsel:bytea)
15. prim (**personel_id: int**, ay:date,miktar:int)

Veri Bağıntı Diyagramı



Valentina Studioda tablolar arası ilişkiler.



Veri Bağıntını Oluşturan Sql Kodları

Sql kodları en alttaki linktedir.

Triggerlar ve Saklı Yordamlar

```
BEGIN;

-- CREATE FUNCTION "doktor_bul( int4 )" ------------------------
CREATE OR REPLACE FUNCTION public.doktor_bul(aranan_id INTEGER)
 RETURNS TABLE(id INTEGER, isim CHARACTER VARYING)
 LANGUAGE plpgsql
AS $function$
BEGIN
    RETURN QUERY SELECT "personel_id", "adi_soyadi" FROM personel
                 WHERE "personel_id" = aranan_id;
END;
$function$;
-- -------------------------------------------------------------

COMMIT;
```

İd si verilen doktorun bulan fonksiyon

```
BEGIN;
CREATE OR REPLACE FUNCTION public.islem_kaydet(p_hasta_id INTEGER, p_tani_adi CHARACTER VARYING, p_tedavi_adi CHARACTER
VARYING, p_tarih date, p_ilac_adi CHARACTER VARYING, p_personel_adi CHARACTER VARYING, p_stajyer_adi CHARACTER VARYING)
 RETURNS TEXT
 LANGUAGE plpgsql
AS $function$
 DECLARE p_hasta_id INT;
 DECLARE p_tani_id INT ;
 DECLARE p_tedavi_id  INT ;
 DECLARE p_ilac_id INT ;
 DECLARE p_personel_id  INT ;
 DECLARE p_stajyer_id INT  ;

BEGIN

p_hasta_id := (SELECT "hasta_id" FROM "hasta" WHERE "hasta_id"=p_hasta_id);
p_tani_id := (SELECT "tani_id" FROM "tani" WHERE "tani_adi"=p_tani_adi);
p_tedavi_id := (SELECT "tedavi_id" FROM "tedavi" WHERE "tedavi_adi"=p_tedavi_adi);
p_ilac_id := (SELECT "ilac_id" FROM "ilac" WHERE "ilac_adi"=p_ilac_adi);
p_personel_id := (SELECT "personel"."personel_id" FROM "personel" WHERE "personel"."adi_soyadi"=p_personel_adi);
p_stajyer_id := (SELECT "stajyer_hekim"."stajyer_id" FROM "stajyer_hekim" WHERE "stajyer_hekim"."adi_soyadi"=p_stajyer_adi);

INSERT INTO "islem_kayıt"("hasta_id","tani_id","tedavi_id","tarih","ilac_id","personel_id","stajyer_id") VALUES (p_hasta_id,
p_tani_id,p_tedavi_id,p_tarih,p_ilac_id,p_personel_id,p_stajyer_id );

 RETURN 'hasta kaydı oluşturuldu';
END;
```

Verilen argümanlar ile işlem_kayıt tablosuna işlem kaydını yapar. İsim olarak olan argümanların id leri bulunur daha sonra insert işlemini gerçekleştirir.

```
BEGIN;

 -- CREATE FUNCTION "islemleri_getir( int4 )" --------------------
 CREATE OR REPLACE FUNCTION public.islemleri_getir(personelno INTEGER)
  RETURNS TABLE(hastaisim CHARACTER VARYING, adi CHARACTER VARYING,
 uygulanan_tedavi CHARACTER VARYING, islem_tarih TIMESTAMP WITHOUT TIME ZONE,
 tedavi_ucret money)
  LANGUAGE plpgsql
 AS $function$
BEGIN
 RETURN QUERY SELECT "hasta"."adi_soyadi","personel"."adi_soyadi","tedavi".
 "tedavi_adi","islem_kayıt"."tarih" ,"tedavi"."ucret" FROM islem_kayıt INNER
 JOIN personel ON "personel"."personel_id"="islem_kayıt"."personel_id" INNER
 JOIN tedavi ON "tedavi"."tedavi_id"="islem_kayıt"."tedavi_id"  INNER JOIN
 hasta ON "hasta"."hasta_id"="islem_kayıt"."hasta_id"   WHERE "islem_kayıt".
 "personel_id" = personelNo;

END;
 $function$;
 -- -----------------------------------------------------------------

 COMMIT;
```

İd si verilen personelin tüm tedavi işlemlerini ve doktorlarını getiren fonksiyon.

```
BEGIN;

-- CREATE FUNCTION "prim_getir( int4 )" ------------------------
CREATE OR REPLACE FUNCTION public.prim_getir(personelno INTEGER)
 RETURNS TABLE(numara INTEGER, adi CHARACTER VARYING, tarih date,
tutar money)
 LANGUAGE plpgsql
AS $function$
BEGIN
RETURN QUERY SELECT "personel"."personel_id","personel".
"adi_soyadi","prim"."ay","prim"."miktar" FROM "personel" INNER
JOIN "prim" ON "personel"."personel_id"="prim"."personel_id" WHERE
 "personel"."personel_id" = personelNo;
END;
$function$;
-- ------------------------------------------------------------

COMMIT;
```

İdsi verilen personelin primlerini gösterir

Arama, Silme, Ekleme ve Güncelleme Ekran Görüntüleri

Saklı Yordamlar İle Beraber

1.Arama

```
BEGIN;

-- CREATE FUNCTION "hasta_kayit_ara( int4 )" -------------------
CREATE OR REPLACE FUNCTION public.hasta_kayit_ara(aranan_id
INTEGER)
 RETURNS TABLE(tc BIGINT, adi_soyadi CHARACTER VARYING, sehir_adi
CHARACTER VARYING, telno BIGINT)
 LANGUAGE plpgsql
AS $function$
BEGIN
    RETURN QUERY SELECT "hasta"."tc","hasta"."adi_soyadi","sehir".
"baslik","hasta"."telno" FROM "hasta" INNER JOIN sehir ON
hasta.sehir_id = sehir.id INNER JOIN ulke ON sehir.ulke_id =
ulke.id WHERE  "hasta"."hasta_id"=aranan_id;
END;
$function$;
-- ------------------------------------------------------------

COMMIT;
```

2.Ekleme

```
CREATE OR REPLACE FUNCTION public.hasta_ekle(p_adi_soyadi
CHARACTER VARYING, p_tc BIGINT, p_sehir_adi CHARACTER VARYING,
p_cinsiyet CHARACTER VARYING, p_telno BIGINT, p_dogum_tarih
date)
 RETURNS TEXT
 LANGUAGE plpgsql
AS $function$
 DECLARE p_sehir_id INT;

BEGIN

p_sehir_id := (SELECT "id" FROM "sehir" WHERE "baslik"=
p_sehir_adi);
 INSERT INTO "public"."hasta" ("adi_soyadi","tc","sehir_id",
"cinsiyet","telno","dogum_tarih")
 VALUES (p_adi_soyadi,  p_tc ,  p_sehir_id,  p_cinsiyet ,
p_telno, p_dogum_tarih );

 RETURN 'hasta kaydı oluşturuldu';
END;
$function$;
```

3.Güncelleme

```
BEGIN;

-- CREATE FUNCTION "hasta_guncelle( int4,  varchar,  int8,
varchar,  varchar,  int8,  date )"
CREATE OR REPLACE FUNCTION public.hasta_guncelle(p_id INTEGER,
p_adi_soyadi CHARACTER VARYING, p_tc BIGINT, p_sehir_adi
CHARACTER VARYING, p_cinsiyet CHARACTER VARYING, p_telno BIGINT
, p_dogum_tarih date)
 RETURNS TEXT
 LANGUAGE plpgsql
AS $function$
 DECLARE p_sehir_id INT;

BEGIN

p_sehir_id := (SELECT "id" FROM "sehir" WHERE "baslik"=
p_sehir_adi);
 UPDATE "public"."hasta" SET "adi_soyadi"=p_adi_soyadi,"tc"=
p_tc,"sehir_id"=p_sehir_id,"cinsiyet"=p_cinsiyet,"telno"=
p_telno,"dogum_tarih"=p_dogum_tarih WHERE "hasta_id"=p_id;

 RETURN 'hasta kaydı güncellendi';
END;
```

4.Silme

```
BEGIN;

-- CREATE FUNCTION "hasta_sil( int4 )" -------------------------
CREATE OR REPLACE FUNCTION public.hasta_sil(p_id INTEGER)
 RETURNS TEXT
 LANGUAGE plpgsql
AS $function$

BEGIN

 DELETE FROM "public"."hasta" WHERE "hasta_id"=p_id;

 RETURN 'hasta kaydı silindi';
END;
$function$;
-- ---------------------------------------------------------------
```

Proje ve Sql Kodları için : https://github.com/rabiaabdioglu/veritabani

Video için : https://youtu.be/ihgSWHIgxXY