SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

VERİTABANI YÖNETİM SİSTEMLERİ DERSİ

PROJE ÖDEVİ

B201210100

BEYZANUR ODABAŞ

beyzanur.odabas@ogr.sakarya.edu.tr

İçerik

** Uygulamanın kısa tanıtımı, iş kuralları, ilişkisel şema

**Varlık Bağıntı modeli

**SQL ifadeleri

**Saklı yordamlar , tetikleyiciler

** Uygulamaya ait ekran görüntüleri

**Uygulamanın kaynak kodlarının bulunduğu Github adresi

**Çalışmamı anlattığım videonun adresi

Uygulamanın Tanıtımı

Projemde bir hastanenin veritabanı gerçeklemesini yaptım. Veritabanı yönetim sistemi olarak postgreSQL kullanılmıştır. C# ile uygulamamı hazırladım.

İŞ KURALLARI

Hastalar randevu alır. Bir randevu varsa bu en az / en çok bir hastaya aittir. Bir hastanın en az 0 en çok çok randevusu olabilir.

Doktorların randevuları vardır. Bir doktorun hiç randevusu olmayacağı gibi çok randevusu da olabilir. Bir randevu varsa bu en az / en çok bir doktora aittir.

Doktorların branşları vardır. Bir doktor en az / en çok 1 branşa sahiptir. Bir branşta hiç doktor olmayadabilir çok sayıda doktor da olabilir.

Bir hasta muayene olur. Bir muayene en az /en çok bir hastaya aittir. Bir hasta hiç muayene olmayabilir, en çok çok sayıda muayene olur.

Doktorlar hastaları muayene ederler. Bir doktor hiç hasta muayene etmeyedebilir , çok sayıda muayene de edebilir. Bir muayene varsa bu en az /en çok bir doktora aittir.

Bir hastanede en az 1 en çok çok sayıda doktor olabilir. Bir doktor hiç hastanede olmaz veya en çok bir hastanede olabilir.

Hastaneler şehirde bulunur. Bir şehirde en az 0 en çok çok sayıda hastane vardır. Bir hastane en az / en çok 1 şehirde bulunur.

Şehirlerde ilçeler bulunur. Bir şehrin en az 1 en çok çok sayıda ilçesi bulunur. Bir ilçe en az 1 en çok da 1 şehre aittir.

Doktorlar reçete yazarlar. Bir doktor hiç reçete yazmayadabilir çok sayıda da reçete yazabilir. Bir reçete varsa bu en az / en çok 1 doktora aittir.

Bir reçetede en az 1 en çok çok sayıda ilaç bulunur. Bir ilaç hiçbir reçeteye ait olmayadabilir. Bir ilaç en çok çok sayıda reçeteye aittir.

Hastalar reçeteye sahiptir. Bir hastanın hiç reçetesi olmayacağı gibi en çok çok sayıda reçetesi olabilir. Bir reçete en az /en çok bir hastaya aittir.

Randevunun tarih bilgisi bulunur. Bir tarihte en az 0 en çok çok sayıda randevu olabilir. Bir randevu yalnızca bir tarihte gerçekleşir.

Hastanede odalar bulunur. Bir hastanede en az 1 en çok çok sayıda oda bulunur. Bir oda en az en çok 1 hastanede bulunur.

Odalarda hastalar yatar. Bir hasta en az 0 en çok 1 odada kalır. Bir odada en az 0 en çok çok sayıda hasta kalabilir.

Bir hastanenin yöneticisi bulunur. Bir hastanede en az 1 en çok çok sayıda yönetici bulunur. Bir yönetici yalnızca bir hastanede bulunabilir.

Hastanede en az 1 en çok çok sayıda personel bulunur. Bir personel yalnızca bir hastanede bulunabilir.

Hasta tablosu tc_no, ad, soyad, telefon ve oda_no alanlarını içermektedir.

Muayene tablosu muayene_id , hasta_tc, doktor_id alanlarını içermektedir.

Randevu tablosu randevu_id, hasta_tc, doktor_id, tarih_no alanlarını içerir.

Doktor tablosu ad, soyad, telefon, branş, doktor_id, hastane_no alanlarını içerir.

Hastane tablosu hastane_Adi, plaka_no, hastane_no alanlarını içerir.

Branş tablosu brans_adi, branş_no alanlarını içerir.

Şehir tablosu plaka_no, il_ismi alanlarını içerir.

İlçe tablosu ilce_kodu, ilce_ismi, plaka_no alanlarını içerir.

Tarih tablosu tarih_no, tarıh, saat alanlarını içerir.

İlaç tablosu ilac_adi, kullanim alanlarını içerir.

Reçete tablosu reçete_no, doktor_id, hasta_tc alanlarını içerir.

ilaç_recete tablosu reçete_no, ilac_adi alanlarını içerir.

Yönetici tablosu yönetici_no, yönetici_adi, yönetici_soyadi, hastane_no alanlarını içerir.

Personel tablosu personelNo, personelAdi, personelSoyadi, personelYasi, hastane_no alanlarını içerir.

Oda tablosu oda_no, kati, hastane_no alanlarını içerir.

İLİŞKİSEL ŞEMA

hasta(**tc_no: char(11)**, ad: varchar, soyad: varchar , telefon:varchar, oda_no:integer)

muayene(**muayene_id: integer** , hasta_tc: char(11), doktor_id: integer)

randevu(**randevu_id: integer**, hasta_tc: char(11), doktor_id: integer, tarih_no: integer)

doktor(**doktor_id: integer**, ad: varchar, soyad: varchar, telefon: varchar, brans_adi: text, hastane_no: integer)

hastane(**hastane_no: integer**, hastane_adi: varchar, plaka_no: varchar,)

brans(**brans_adi: text**, brans_no:integer)

sehir(**plaka_no: varchar**, il_ismi: varchar)

ilce(**ilce_kodu: integer**, ilce_ismi: varchar, plaka_no:varchar)

tarih(**tarih_no:integer**, tarih:date, saat:time)

ilac(**ilac_adi:varchar** , kullanim:text)

recete(**reçete_no:varchar**, doktor_id: integer, hasta_tc: char(11))

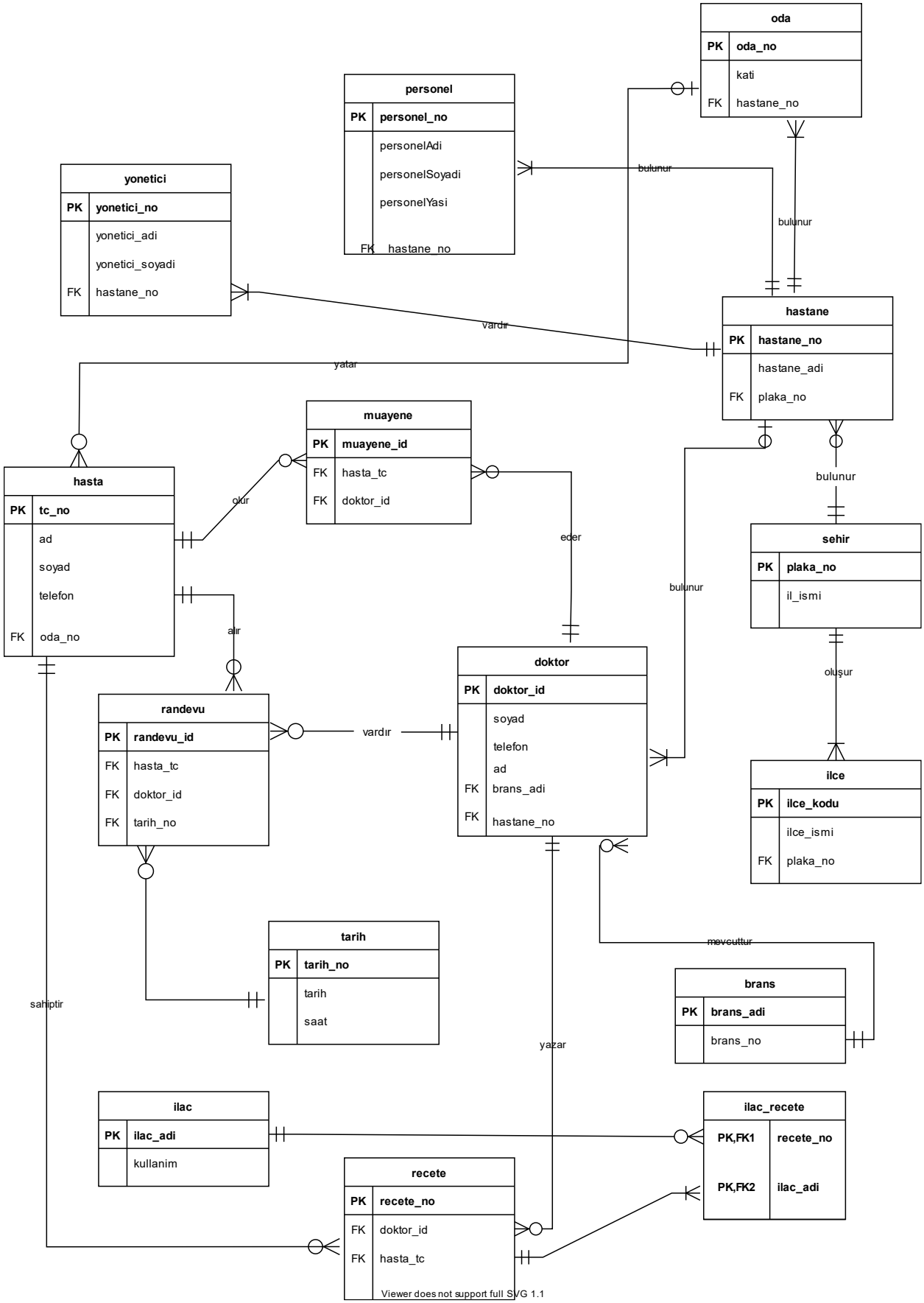
ilac_recete(**reçete_no:varchar, ilac_adi: varchar**)

yonetici(**yönetici_no:integer**, yönetici_adi:varchar, yönetici_soyadi:varchar, hastane_no :integer)

personel(**personelNo: integer**, personelAdi:varchar, personelSoyadi: varchar, personelYasi: smallint, hastane_no: integer)

oda(**oda_no: integer**, kati: smallint, hastane_no : integer)

VARLIK BAĞINTI DİYAGRAMI



SQL İFADELERİ

```
--
-- PostgreSQL database dump
--


-- Dumped from database version 14.0
-- Dumped by pg_dump version 14.0

SET statement_timeout = 0;
SET lock_timeout = 0;
SET idle_in_transaction_session_timeout = 0;
SET client_encoding = 'UTF8';
SET standard_conforming_strings = on;
SELECT pg_catalog.set_config('search_path', '', false);
SET check_function_bodies = false;
SET xmloption = content;
SET client_min_messages = warning;
SET row_security = off;


--
-- Name: doktorDegisikligi(); Type: FUNCTION; Schema: public; Owner: postgres
--


CREATE FUNCTION public."doktorDegisikligi"() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
BEGIN
            IF NEW."brans_adi" <> OLD."brans_adi" THEN
            INSERT INTO "doktorDegisikligiIzle"("doktor_no","eskiBransNo",
           "yeniBransNo", "degisiklikTarihi")
           VALUES(OLD."doktor_no",OLD."brans_adi",NEW."brans_adi",
```

```
                        CURRENT_TIMESTAMP::TIMESTAMP);

    END IF;

   RETURN NEW;
END;
$$;



ALTER FUNCTION public."doktorDegisikligi"() OWNER TO postgres;


--
-- Name: doktor_brans_listele(); Type: FUNCTION; Schema: public; Owner: postgres
--

CREATE FUNCTION public.doktor_brans_listele() RETURNS TABLE(doktor_id integer, brans_adi text)
   LANGUAGE plpgsql
   AS $$
BEGIN
  RETURN QUERY SELECT
            doktor_id,
     brans_adi
  FROM
     doktor
  WHERE
     doktor_id=1 and 2 and 3 and 4 and 5
      order by brans_adi asc;
END;
$$;
```

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


--
-- Name: doktor_sayisi(); Type: FUNCTION; Schema: public; Owner: postgres
--

CREATE FUNCTION public.doktor_sayisi() RETURNS integer
    LANGUAGE plpgsql
    AS $$
            DECLARE
              doktor_sayi integer;
            BEGIN
              SELECT count("doktor_id") into doktor_sayi from "doktor";
              RETURN doktor_sayi;
            END;
            $$;


ALTER FUNCTION public.doktor_sayisi() OWNER TO postgres;


--
-- Name: hastaKayitEkle(); Type: FUNCTION; Schema: public; Owner: postgres
--

CREATE FUNCTION public."hastaKayitEkle"() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
BEGIN
    NEW."soyad"=UPPER(NEW."soyad");

            IF NEW."ad" IS NULL THEN
```

```
            RAISE EXCEPTION 'Ad alanı boş bırakılamaz.';
    END IF;

   RETURN NEW;
END;
$$;


ALTER FUNCTION public."hastaKayitEkle"() OWNER TO postgres;

--
-- Name: hasta_sayisi(); Type: FUNCTION; Schema: public; Owner: postgres
--

CREATE FUNCTION public.hasta_sayisi() RETURNS integer
   LANGUAGE plpgsql
   AS $$
            DECLARE
              hasta_sayi integer;
            BEGIN
              SELECT count("tc_no") into hasta_sayi from "hasta";
              RETURN hasta_sayi;
           END;
           $$;


ALTER FUNCTION public.hasta_sayisi() OWNER TO postgres;

--
-- Name: ilacKayitEkle(); Type: FUNCTION; Schema: public; Owner: postgres
--
```

```
CREATE FUNCTION public."ilacKayitEkle"() RETURNS trigger
    LANGUAGE plpgsql
    AS $$
BEGIN
    NEW."ilac_adi"=UPPER(NEW."ilac_adi");

    RETURN NEW;
END;
$$;



ALTER FUNCTION public."ilacKayitEkle"() OWNER TO postgres;


--
-- Name: kayitdolanimi(); Type: FUNCTION; Schema: public; Owner: postgres
--


CREATE FUNCTION public.kayitdolanimi() RETURNS text
    LANGUAGE plpgsql
    AS $$
DECLARE
    muayenedegisken muayene%ROWTYPE;
    sonuc TEXT;
BEGIN
    sonuc := '';
    FOR muayenedegisken IN SELECT * FROM muayene LOOP
        sonuc := sonuc || muayenedegisken."muayene_id" || E'\t' || muayenedegisken."hasta_tc" ||
E'\t' || muayenedegisken."doktor_id" || E'\r\n';
    END LOOP;
    RETURN sonuc;
```

```
END;
$$;


ALTER FUNCTION public.kayitdolanimi() OWNER TO postgres;

--
-- Name: randevuDegisikligi(); Type: FUNCTION; Schema: public; Owner: postgres
--

CREATE FUNCTION public."randevuDegisikligi"() RETURNS trigger
   LANGUAGE plpgsql
   AS $$
BEGIN
            IF NEW."tarih_no" <> OLD."tarih_no" THEN
            INSERT INTO "randevuTarihDegisikligi"("randevu_no","eskiTarihNo",
           "yeniTarihNo", "degisiklikTarihi")
           VALUES(OLD."randevu_no",OLD."tarih_no",NEW."tarih_no",
                        CURRENT_TIMESTAMP::TIMESTAMP);

    END IF;

   RETURN NEW;
END;
$$;


ALTER FUNCTION public."randevuDegisikligi"() OWNER TO postgres;

SET default_tablespace = '';
```

```
SET default_table_access_method = heap;

--
-- Name: brans; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.brans (
    brans_adi text NOT NULL,
    brans_no integer NOT NULL
);


ALTER TABLE public.brans OWNER TO postgres;

--
-- Name: doktor; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.doktor (
    doktor_id integer NOT NULL,
    ad character varying NOT NULL,
    soyad character varying NOT NULL,
    telefon character varying NOT NULL,
    brans_adi text NOT NULL,
    hastane_no integer NOT NULL
);


ALTER TABLE public.doktor OWNER TO postgres;

--
```

```
-- Name: doktorDegisikligiIzle; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public."doktorDegisikligiIzle" (
    doktor_no smallint NOT NULL,
    "kayitNo" integer NOT NULL,
    "eskiBransNo" integer NOT NULL,
    "yeniBransNo" integer NOT NULL,
    "degisiklikTarihi" timestamp without time zone NOT NULL
);


ALTER TABLE public."doktorDegisikligiIzle" OWNER TO postgres;

--
-- Name: doktorDegisikligiIzle_kayitNo_seq; Type: SEQUENCE; Schema: public; Owner: postgres
--

CREATE SEQUENCE public."doktorDegisikligiIzle_kayitNo_seq"
    AS integer
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE public."doktorDegisikligiIzle_kayitNo_seq" OWNER TO postgres;

--
```

```
-- Name: doktorDegisikligiIzle_kayitNo_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres
--

ALTER SEQUENCE public."doktorDegisikligiIzle_kayitNo_seq" OWNED BY public."doktorDegisikligiIzle"."kayitNo";



--
-- Name: hasta; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.hasta (
    tc_no "char" NOT NULL,
    ad character varying NOT NULL,
    soyad character varying NOT NULL,
    telefon character varying(11) NOT NULL,
    oda_no integer NOT NULL
);


ALTER TABLE public.hasta OWNER TO postgres;

--
-- Name: hastane; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.hastane (
    hastane_no integer NOT NULL,
    hastane_adi character varying NOT NULL,
    plaka_no character varying NOT NULL
);
```

```
ALTER TABLE public.hastane OWNER TO postgres;

--
-- Name: ilac; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.ilac (
    ilac_adi character varying NOT NULL,
    kullanim text
);


ALTER TABLE public.ilac OWNER TO postgres;

--
-- Name: ilac_recete; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.ilac_recete (
    recete_no character varying NOT NULL,
    ilac_adi character varying NOT NULL
);


ALTER TABLE public.ilac_recete OWNER TO postgres;

--
-- Name: ilce; Type: TABLE; Schema: public; Owner: postgres
--
```

```
CREATE TABLE public.ilce (
    ilce_kodu integer NOT NULL,
    ilce_ismi character varying NOT NULL,
    plaka_no character varying(2) NOT NULL
);


ALTER TABLE public.ilce OWNER TO postgres;

--
-- Name: muayene; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.muayene (
    muayene_id integer NOT NULL,
    hasta_tc "char" NOT NULL,
    doktor_id integer NOT NULL
);


ALTER TABLE public.muayene OWNER TO postgres;

--
-- Name: oda; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.oda (
    oda_no integer NOT NULL,
    kati smallint NOT NULL,
    hastane_no integer NOT NULL
```

```
);


ALTER TABLE public.oda OWNER TO postgres;

--
-- Name: personel; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.personel (
    "personelNo" integer NOT NULL,
    "personelAdi" character varying NOT NULL,
    "personelSoyadi" character varying NOT NULL,
    "personelYasi" smallint NOT NULL,
    hastane_no integer NOT NULL
);


ALTER TABLE public.personel OWNER TO postgres;

--
-- Name: randevu; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.randevu (
    randevu_id integer NOT NULL,
    hasta_tc "char" NOT NULL,
    doktor_id integer NOT NULL,
    tarih_no integer NOT NULL
);
```

```
ALTER TABLE public.randevu OWNER TO postgres;

--
-- Name: randevuTarihDegisikligi; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public."randevuTarihDegisikligi" (
    randevu_no integer NOT NULL,
    "eskiTarihNo" integer NOT NULL,
    "yeniTarihNo" integer NOT NULL,
    "degisiklikTarihi" timestamp without time zone NOT NULL
);


ALTER TABLE public."randevuTarihDegisikligi" OWNER TO postgres;

--
-- Name: recete; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.recete (
    recete_no character varying NOT NULL,
    doktor_id integer NOT NULL,
    hasta_tc "char" NOT NULL
);


ALTER TABLE public.recete OWNER TO postgres;

--
```

```
-- Name: sehir; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.sehir (
    plaka_no character varying(2) NOT NULL,
    il_ismi character varying(20) NOT NULL
);


ALTER TABLE public.sehir OWNER TO postgres;

--
-- Name: tarih; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.tarih (
    tarih_no integer NOT NULL,
    tarih date,
    saat time without time zone
);


ALTER TABLE public.tarih OWNER TO postgres;

--
-- Name: yonetici; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.yonetici (
    yonetici_no integer NOT NULL,
    yonetici_adi character varying NOT NULL,
```

```
    yonetici_soyadi character varying NOT NULL,
    hastane_no integer NOT NULL
);


ALTER TABLE public.yonetici OWNER TO postgres;

--
-- Name: doktorDegisikligiIzle kayitNo; Type: DEFAULT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public."doktorDegisikligiIzle" ALTER COLUMN "kayitNo" SET DEFAULT
nextval('public."doktorDegisikligiIzle_kayitNo_seq"'::regclass);


--
-- Data for Name: brans; Type: TABLE DATA; Schema: public; Owner: postgres
--

INSERT INTO public.brans (brans_adi, brans_no) VALUES ('kulak-burun-boğaz', 1);
INSERT INTO public.brans (brans_adi, brans_no) VALUES ('göz', 2);
INSERT INTO public.brans (brans_adi, brans_no) VALUES ('cildiye', 3);
INSERT INTO public.brans (brans_adi, brans_no) VALUES ('diş', 4);


--
-- Data for Name: doktor; Type: TABLE DATA; Schema: public; Owner: postgres
--

INSERT INTO public.doktor (doktor_id, ad, soyad, telefon, brans_adi, hastane_no) VALUES (2,
'demet', 'ak', '05351345678', 'cildiye', 2);
```

INSERT INTO public.doktor (doktor_id, ad, soyad, telefon, brans_adi, hastane_no) VALUES (3, 'eda', 'kırmızı', '05351345600', 'diş', 3);

INSERT INTO public.doktor (doktor_id, ad, soyad, telefon, brans_adi, hastane_no) VALUES (4, 'ahmet', 'cetin', '05351345656', 'kulak-burun-boğaz', 4);

INSERT INTO public.doktor (doktor_id, ad, soyad, telefon, brans_adi, hastane_no) VALUES (1, 'ilker', 'al', '05312345789', 'kulak-burun-boğaz', 1);

INSERT INTO public.doktor (doktor_id, ad, soyad, telefon, brans_adi, hastane_no) VALUES (5, 'gh', 'g', '6', 'kulak-burun-boğaz', 2);

--

-- Data for Name: doktorDegisikligiIzle; Type: TABLE DATA; Schema: public; Owner: postgres

--

--

-- Data for Name: hasta; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public.hasta (tc_no, ad, soyad, telefon, oda_no) VALUES ('1', 'ali', 'al', '05301375544', 1);

INSERT INTO public.hasta (tc_no, ad, soyad, telefon, oda_no) VALUES ('2', 'ayse', 'can', '05301378989', 2);

INSERT INTO public.hasta (tc_no, ad, soyad, telefon, oda_no) VALUES ('3', 'fatma', 'ak', '05301378980', 3);

INSERT INTO public.hasta (tc_no, ad, soyad, telefon, oda_no) VALUES ('4', 'ahmet', 'can', '05301378955', 4);

INSERT INTO public.hasta (tc_no, ad, soyad, telefon, oda_no) VALUES ('5', 'f', 'F', '5', 4);

--

-- Data for Name: hastane; Type: TABLE DATA; Schema: public; Owner: postgres

--

```
INSERT INTO public.hastane (hastane_no, hastane_adi, plaka_no) VALUES (1, 'marmara', '34');
INSERT INTO public.hastane (hastane_no, hastane_adi, plaka_no) VALUES (2, 'kartal devlet', '54');
INSERT INTO public.hastane (hastane_no, hastane_adi, plaka_no) VALUES (3, 'kocaeli darıca', '55');
INSERT INTO public.hastane (hastane_no, hastane_adi, plaka_no) VALUES (4, 'ersoy özel', '60');



--
-- Data for Name: ilac; Type: TABLE DATA; Schema: public; Owner: postgres
--

INSERT INTO public.ilac (ilac_adi, kullanim) VALUES ('parol', 'su ile alınız');
INSERT INTO public.ilac (ilac_adi, kullanim) VALUES ('aferin', 'içiniz');
INSERT INTO public.ilac (ilac_adi, kullanim) VALUES ('aspirin', 'ıcınız');
INSERT INTO public.ilac (ilac_adi, kullanim) VALUES ('arveles', 'yutunuz');



--
-- Data for Name: ilac_recete; Type: TABLE DATA; Schema: public; Owner: postgres
--

INSERT INTO public.ilac_recete (recete_no, ilac_adi) VALUES ('a', 'parol');
INSERT INTO public.ilac_recete (recete_no, ilac_adi) VALUES ('b', 'aferin');
INSERT INTO public.ilac_recete (recete_no, ilac_adi) VALUES ('c', 'arveles');
INSERT INTO public.ilac_recete (recete_no, ilac_adi) VALUES ('d', 'aspirin');



--
-- Data for Name: ilce; Type: TABLE DATA; Schema: public; Owner: postgres
--

INSERT INTO public.ilce (ilce_kodu, ilce_ismi, plaka_no) VALUES (123, 'sultanbeyli', '34');
```

```
INSERT INTO public.ilce (ilce_kodu, ilce_ismi, plaka_no) VALUES (345, 'bafra', '55');
INSERT INTO public.ilce (ilce_kodu, ilce_ismi, plaka_no) VALUES (567, 'serdivan', '54');
INSERT INTO public.ilce (ilce_kodu, ilce_ismi, plaka_no) VALUES (456, 'almus', '60');


--
-- Data for Name: muayene; Type: TABLE DATA; Schema: public; Owner: postgres
--

INSERT INTO public.muayene (muayene_id, hasta_tc, doktor_id) VALUES (54, '1', 1);
INSERT INTO public.muayene (muayene_id, hasta_tc, doktor_id) VALUES (78, '2', 2);
INSERT INTO public.muayene (muayene_id, hasta_tc, doktor_id) VALUES (12, '3', 3);
INSERT INTO public.muayene (muayene_id, hasta_tc, doktor_id) VALUES (90, '4', 4);
INSERT INTO public.muayene (muayene_id, hasta_tc, doktor_id) VALUES (1, '2', 1);
INSERT INTO public.muayene (muayene_id, hasta_tc, doktor_id) VALUES (2, '4', 2);
INSERT INTO public.muayene (muayene_id, hasta_tc, doktor_id) VALUES (3, '1', 3);
INSERT INTO public.muayene (muayene_id, hasta_tc, doktor_id) VALUES (4, '1', 4);


--
-- Data for Name: oda; Type: TABLE DATA; Schema: public; Owner: postgres
--

INSERT INTO public.oda (oda_no, kati, hastane_no) VALUES (1, 1, 1);
INSERT INTO public.oda (oda_no, kati, hastane_no) VALUES (2, 2, 2);
INSERT INTO public.oda (oda_no, kati, hastane_no) VALUES (3, 6, 3);
INSERT INTO public.oda (oda_no, kati, hastane_no) VALUES (4, 7, 4);


--
-- Data for Name: personel; Type: TABLE DATA; Schema: public; Owner: postgres
```

--

INSERT INTO public.personel ("personelNo", "personelAdi", "personelSoyadi", "personelYasi", hastane_no) VALUES (1, 'ayse', 'tok', 12, 1);

INSERT INTO public.personel ("personelNo", "personelAdi", "personelSoyadi", "personelYasi", hastane_no) VALUES (2, 'fatma', 'cin', 16, 2);

INSERT INTO public.personel ("personelNo", "personelAdi", "personelSoyadi", "personelYasi", hastane_no) VALUES (3, 'ali', 'yal', 36, 3);

INSERT INTO public.personel ("personelNo", "personelAdi", "personelSoyadi", "personelYasi", hastane_no) VALUES (4, 'tugba', 'sal', 25, 4);

--

-- Data for Name: randevu; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public.randevu (randevu_id, hasta_tc, doktor_id, tarih_no) VALUES (1, '1', 1, 1);

INSERT INTO public.randevu (randevu_id, hasta_tc, doktor_id, tarih_no) VALUES (2, '2', 2, 2);

INSERT INTO public.randevu (randevu_id, hasta_tc, doktor_id, tarih_no) VALUES (3, '3', 3, 3);

INSERT INTO public.randevu (randevu_id, hasta_tc, doktor_id, tarih_no) VALUES (4, '4', 4, 4);

--

-- Data for Name: randevuTarihDegisikligi; Type: TABLE DATA; Schema: public; Owner: postgres

--

--

-- Data for Name: recete; Type: TABLE DATA; Schema: public; Owner: postgres

--

```
INSERT INTO public.recete (recete_no, doktor_id, hasta_tc) VALUES ('a', 1, '1');
INSERT INTO public.recete (recete_no, doktor_id, hasta_tc) VALUES ('b', 2, '2');
INSERT INTO public.recete (recete_no, doktor_id, hasta_tc) VALUES ('c', 3, '3');
INSERT INTO public.recete (recete_no, doktor_id, hasta_tc) VALUES ('d', 4, '4');


--
-- Data for Name: sehir; Type: TABLE DATA; Schema: public; Owner: postgres
--

INSERT INTO public.sehir (plaka_no, il_ismi) VALUES ('34', 'istanbul');
INSERT INTO public.sehir (plaka_no, il_ismi) VALUES ('55', 'samsun');
INSERT INTO public.sehir (plaka_no, il_ismi) VALUES ('54', 'sakarya');
INSERT INTO public.sehir (plaka_no, il_ismi) VALUES ('60', 'tokat');


--
-- Data for Name: tarih; Type: TABLE DATA; Schema: public; Owner: postgres
--

INSERT INTO public.tarih (tarih_no, tarih, saat) VALUES (1, NULL, NULL);
INSERT INTO public.tarih (tarih_no, tarih, saat) VALUES (2, NULL, NULL);
INSERT INTO public.tarih (tarih_no, tarih, saat) VALUES (3, NULL, NULL);
INSERT INTO public.tarih (tarih_no, tarih, saat) VALUES (4, NULL, NULL);


--
-- Data for Name: yonetici; Type: TABLE DATA; Schema: public; Owner: postgres
--
```

```
INSERT INTO public.yonetici (yonetici_no, yonetici_adi, yonetici_soyadi, hastane_no) VALUES (1, 'ece', 'tekşn', 1);

INSERT INTO public.yonetici (yonetici_no, yonetici_adi, yonetici_soyadi, hastane_no) VALUES (2, 'ali', 'cetin', 2);

INSERT INTO public.yonetici (yonetici_no, yonetici_adi, yonetici_soyadi, hastane_no) VALUES (3, 'ahmet', 'al', 3);

INSERT INTO public.yonetici (yonetici_no, yonetici_adi, yonetici_soyadi, hastane_no) VALUES (4, 'betül', 'tutıl', 4);



--

-- Name: doktorDegisikligiIzle_kayitNo_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg_catalog.setval('public."doktorDegisikligiIzle_kayitNo_seq"', 1, false);



--

-- Name: brans brans_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.brans

    ADD CONSTRAINT brans_pkey PRIMARY KEY (brans_adi);



--

-- Name: doktor doktor_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.doktor

    ADD CONSTRAINT doktor_pkey PRIMARY KEY (doktor_id);
```

```
--
-- Name: doktorDegisikligiIzle doktorpk; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public."doktorDegisikligiIzle"
    ADD CONSTRAINT doktorpk PRIMARY KEY (doktor_no);


--
-- Name: hasta hasta_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.hasta
    ADD CONSTRAINT hasta_pkey PRIMARY KEY (tc_no);


--
-- Name: hastane hastane_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.hastane
    ADD CONSTRAINT hastane_pkey PRIMARY KEY (hastane_no);


--
-- Name: ilac ilac_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.ilac
    ADD CONSTRAINT ilac_pkey PRIMARY KEY (ilac_adi);
```

```
--
-- Name: ilac_recete ilac_recete_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.ilac_recete
    ADD CONSTRAINT ilac_recete_pkey PRIMARY KEY (recete_no, ilac_adi);


--
-- Name: ilce ilce_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.ilce
    ADD CONSTRAINT ilce_pkey PRIMARY KEY (ilce_kodu);


--
-- Name: muayene muayene_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.muayene
    ADD CONSTRAINT muayene_pkey PRIMARY KEY (muayene_id);


--
-- Name: oda oda_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.oda
```

```
    ADD CONSTRAINT oda_pkey PRIMARY KEY (oda_no);


--
-- Name: personel personel_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.personel
    ADD CONSTRAINT personel_pkey PRIMARY KEY ("personelNo");


--
-- Name: randevu randevu_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.randevu
    ADD CONSTRAINT randevu_pkey PRIMARY KEY (randevu_id);


--
-- Name: randevuTarihDegisikligi randevupk; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public."randevuTarihDegisikligi"
    ADD CONSTRAINT randevupk PRIMARY KEY (randevu_no);


--
-- Name: recete recete_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--
```

```
ALTER TABLE ONLY public.recete
    ADD CONSTRAINT recete_pkey PRIMARY KEY (recete_no);


--
-- Name: sehir sehir_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.sehir
    ADD CONSTRAINT sehir_pkey PRIMARY KEY (plaka_no);


--
-- Name: tarih tarih_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.tarih
    ADD CONSTRAINT tarih_pkey PRIMARY KEY (tarih_no);


--
-- Name: yonetici yonetici_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.yonetici
    ADD CONSTRAINT yonetici_pkey PRIMARY KEY (yonetici_no);


--
-- Name: fki_brans_adi; Type: INDEX; Schema: public; Owner: postgres
--
```

```
CREATE INDEX fki_brans_adi ON public.doktor USING btree (brans_adi);


--
-- Name: fki_doktor_id; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_doktor_id ON public.randevu USING btree (doktor_id);


--
-- Name: fki_doktor_id_fk; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_doktor_id_fk ON public.muayene USING btree (doktor_id);


--
-- Name: fki_doktor_id_fk2; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_doktor_id_fk2 ON public.recete USING btree (doktor_id);


--
-- Name: fki_hasta_tc; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_hasta_tc ON public.randevu USING btree (hasta_tc);
```

```
--
-- Name: fki_hasta_tc_fk; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_hasta_tc_fk ON public.muayene USING btree (hasta_tc);


--
-- Name: fki_hasta_tc_fk2; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_hasta_tc_fk2 ON public.recete USING btree (hasta_tc);


--
-- Name: fki_hastane_no; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_hastane_no ON public.doktor USING btree (hastane_no);


--
-- Name: fki_hastane_no_fk1; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_hastane_no_fk1 ON public.oda USING btree (hastane_no);


--
-- Name: fki_hastane_no_fk2; Type: INDEX; Schema: public; Owner: postgres
```

```
--

CREATE INDEX fki_hastane_no_fk2 ON public.personel USING btree (hastane_no);


--
-- Name: fki_hastane_no_fk3; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_hastane_no_fk3 ON public.yonetici USING btree (hastane_no);


--
-- Name: fki_ilac_adi; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_ilac_adi ON public.ilac_recete USING btree (ilac_adi);


--
-- Name: fki_oda_no; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_oda_no ON public.hasta USING btree (oda_no);


--
-- Name: fki_plaka_no; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX fki_plaka_no ON public.hastane USING btree (plaka_no);
```

--

-- Name: fki_recete_no; Type: INDEX; Schema: public; Owner: postgres

--

CREATE INDEX fki_recete_no ON public.ilac_recete USING btree (recete_no);

--

-- Name: fki_tarih_no; Type: INDEX; Schema: public; Owner: postgres

--

CREATE INDEX fki_tarih_no ON public.randevu USING btree (tarih_no);

--

-- Name: doktor doktorBransDegistiginde; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "doktorBransDegistiginde" BEFORE UPDATE ON public.doktor FOR EACH ROW EXECUTE FUNCTION public."doktorDegisikligi"();

--

-- Name: hasta hastaKayitKontrol; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "hastaKayitKontrol" BEFORE INSERT OR UPDATE ON public.hasta FOR EACH ROW EXECUTE FUNCTION public."hastaKayitEkle"();

```
--

-- Name: ilac ilacTrig; Type: TRIGGER; Schema: public; Owner: postgres

--


CREATE TRIGGER "ilacTrig" BEFORE INSERT OR UPDATE ON public.ilac FOR EACH ROW EXECUTE
FUNCTION public."ilacKayitEkle"();




--

-- Name: randevu randevuTarihDegistiginde; Type: TRIGGER; Schema: public; Owner: postgres

--


CREATE TRIGGER "randevuTarihDegistiginde" BEFORE UPDATE ON public.randevu FOR EACH ROW
EXECUTE FUNCTION public."randevuDegisikligi"();




--

-- Name: doktor brans_adi; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--


ALTER TABLE ONLY public.doktor

    ADD CONSTRAINT brans_adi FOREIGN KEY (brans_adi) REFERENCES public.brans(brans_adi) NOT
VALID;




--

-- Name: randevu doktor_id; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--


ALTER TABLE ONLY public.randevu

    ADD CONSTRAINT doktor_id FOREIGN KEY (doktor_id) REFERENCES public.doktor(doktor_id) NOT
VALID;
```

```
--

-- Name: muayene doktor_id_fk; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--


ALTER TABLE ONLY public.muayene

    ADD CONSTRAINT doktor_id_fk FOREIGN KEY (doktor_id) REFERENCES public.doktor(doktor_id)
NOT VALID;




--

-- Name: recete doktor_id_fk2; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--


ALTER TABLE ONLY public.recete

    ADD CONSTRAINT doktor_id_fk2 FOREIGN KEY (doktor_id) REFERENCES public.doktor(doktor_id)
NOT VALID;




--

-- Name: randevu hasta_tc; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--


ALTER TABLE ONLY public.randevu

    ADD CONSTRAINT hasta_tc FOREIGN KEY (hasta_tc) REFERENCES public.hasta(tc_no) NOT VALID;




--

-- Name: muayene hasta_tc; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--
```

```
ALTER TABLE ONLY public.muayene
    ADD CONSTRAINT hasta_tc FOREIGN KEY (hasta_tc) REFERENCES public.hasta(tc_no) NOT VALID;


--
-- Name: muayene hasta_tc_fk; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.muayene
    ADD CONSTRAINT hasta_tc_fk FOREIGN KEY (hasta_tc) REFERENCES public.hasta(tc_no) NOT
VALID;


--
-- Name: recete hasta_tc_fk2; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.recete
    ADD CONSTRAINT hasta_tc_fk2 FOREIGN KEY (hasta_tc) REFERENCES public.hasta(tc_no) NOT
VALID;


--
-- Name: doktor hastane_no; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.doktor
    ADD CONSTRAINT hastane_no FOREIGN KEY (hastane_no) REFERENCES
public.hastane(hastane_no) NOT VALID;


--
```

```
-- Name: oda hastane_no_fk1; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.oda
    ADD CONSTRAINT hastane_no_fk1 FOREIGN KEY (hastane_no) REFERENCES
public.hastane(hastane_no) NOT VALID;


--

-- Name: personel hastane_no_fk2; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.personel
    ADD CONSTRAINT hastane_no_fk2 FOREIGN KEY (hastane_no) REFERENCES
public.hastane(hastane_no) NOT VALID;


--

-- Name: yonetici hastane_no_fk3; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.yonetici
    ADD CONSTRAINT hastane_no_fk3 FOREIGN KEY (hastane_no) REFERENCES
public.hastane(hastane_no) NOT VALID;


--

-- Name: ilac_recete ilac_adi; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public.ilac_recete
    ADD CONSTRAINT ilac_adi FOREIGN KEY (ilac_adi) REFERENCES public.ilac(ilac_adi) NOT VALID;
```

```
--

-- Name: hasta oda_no; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--


ALTER TABLE ONLY public.hasta

    ADD CONSTRAINT oda_no FOREIGN KEY (oda_no) REFERENCES public.oda(oda_no) NOT VALID;




--

-- Name: hastane plaka_no; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--


ALTER TABLE ONLY public.hastane

    ADD CONSTRAINT plaka_no FOREIGN KEY (plaka_no) REFERENCES public.sehir(plaka_no) NOT
VALID;




--

-- Name: ilce plaka_no_fk; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--


ALTER TABLE ONLY public.ilce

    ADD CONSTRAINT plaka_no_fk FOREIGN KEY (plaka_no) REFERENCES public.sehir(plaka_no) NOT
VALID;




--

-- Name: ilac_recete recete_no; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--
```

```
ALTER TABLE ONLY public.ilac_recete
    ADD CONSTRAINT recete_no FOREIGN KEY (recete_no) REFERENCES public.recete(recete_no) NOT VALID;


--
-- Name: randevu tarih_no; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.randevu
    ADD CONSTRAINT tarih_no FOREIGN KEY (tarih_no) REFERENCES public.tarih(tarih_no) NOT VALID;


--
-- PostgreSQL database dump complete
--
```

FONKSİYONLAR

1. FONKSİYON

```
CREATE FUNCTION public.doktor_brans_listele() RETURNS TABLE(doktor_id integer, brans_adi text)
    LANGUAGE plpgsql
    AS $$
BEGIN
  RETURN QUERY SELECT
          doktor_id,
     brans_adi
  FROM
     doktor
```

```
  WHERE
    doktor_id=1 and 2 and 3 and 4 and 5
     order by brans_adi asc;
END;
$$;


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

2. FONKSİYON

```
CREATE FUNCTION public.doktor_sayisi() RETURNS integer
   LANGUAGE plpgsql
   AS $$
         DECLARE
           doktor_sayi integer;
         BEGIN
           SELECT count("doktor_id") into doktor_sayi from "doktor";
           RETURN doktor_sayi;
         END;
         $$;
ALTER FUNCTION public.doktor_sayisi() OWNER TO postgres;
```

3. FONKSİYON

```
CREATE FUNCTION public.hasta_sayisi() RETURNS integer
   LANGUAGE plpgsql
   AS $$
         DECLARE
           hasta_sayi integer;
         BEGIN
```

```
    SELECT count("tc_no") into hasta_sayi from "hasta";
    RETURN hasta_sayi;
END;
$$;


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

4. FONKSİYON

```
CREATE FUNCTION public.kayitdolanimi() RETURNS text
    LANGUAGE plpgsql
    AS $$
DECLARE
    muayenedegisken muayene%ROWTYPE;
    sonuc TEXT;
BEGIN
    sonuc := '';
    FOR muayenedegisken IN SELECT * FROM muayene LOOP
        sonuc := sonuc || muayenedegisken."muayene_id" || E'\t' ||
muayenedegisken."hasta_tc" || E'\t' || muayenedegisken."doktor_id" || E'\r\n';
    END LOOP;
    RETURN sonuc;
END;
$$;


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

TETİKLEYİCİLER

1. TETİKLEYİCİ

```
CREATE OR REPLACE FUNCTION "ilacKayitEkle"()
RETURNS TRIGGER
AS
$$
BEGIN
   NEW."ilac_adi"=UPPER(NEW."ilac_adi");


   RETURN NEW;
END;
$$
LANGUAGE "plpgsql";


CREATE TRIGGER "ilacTrig"
BEFORE INSERT OR UPDATE ON "ilac"
FOR EACH ROW
EXECUTE PROCEDURE "ilacKayitEkle"();
```

2. TETİKLEYİCİ

```
CREATE OR REPLACE FUNCTION "hastaKayitEkle"()
RETURNS TRIGGER
AS
$$
BEGIN
   NEW."soyad"=UPPER(NEW."soyad");


        IF NEW."ad" IS NULL THEN
        RAISE EXCEPTION 'Ad alanı boş bırakılamaz.';
```

```
    END IF;


   RETURN NEW;
END;
$$
LANGUAGE "plpgsql";


CREATE TRIGGER "hastaKayitKontrol"
BEFORE INSERT OR UPDATE ON "hasta"
FOR EACH ROW
EXECUTE PROCEDURE "hastaKayitEkle"();
```

3. TETİKLEYİCİ

```
CREATE TABLE "public"."randevuTarihDegisikligi"(

        "randevu_no" integer NOT NULL ,
        "eskiTarihNo" integer NOT NULL,
        "yeniTarihNo" integer NOT NULL,
        "degisiklikTarihi" TIMESTAMP NOT NULL,
        CONSTRAINT "randevupk" PRIMARY KEY ("randevu_no")

);


CREATE OR REPLACE FUNCTION "randevuDegisikligi"()
RETURNS TRIGGER
AS
$$
BEGIN
         IF NEW."tarih_no" <> OLD."tarih_no" THEN
```

```
        INSERT INTO "randevuTarihDegisikligi"("randevu_no","eskiTarihNo",
        "yeniTarihNo", "degisiklikTarihi")
        VALUES(OLD."randevu_no",OLD."tarih_no",NEW."tarih_no",
                CURRENT_TIMESTAMP::TIMESTAMP);

    END IF;

   RETURN NEW;
END;
$$
LANGUAGE "plpgsql";

CREATE TRIGGER "randevuTarihDegistiginde"
BEFORE UPDATE ON "randevu"
FOR EACH ROW
EXECUTE PROCEDURE "randevuDegisikligi"();
```

4. TETİKLEYİCİ

```
CREATE TABLE "public"."doktorDegisikligiIzle"(

        "doktor_no" smallint NOT NULL ,
        "kayitNo" serial,
        "eskiBransNo" integer NOT NULL,
        "yeniBransNo" integer NOT NULL,
        "degisiklikTarihi" TIMESTAMP NOT NULL,
        CONSTRAINT "doktorpk" PRIMARY KEY ("doktor_no")

);
```

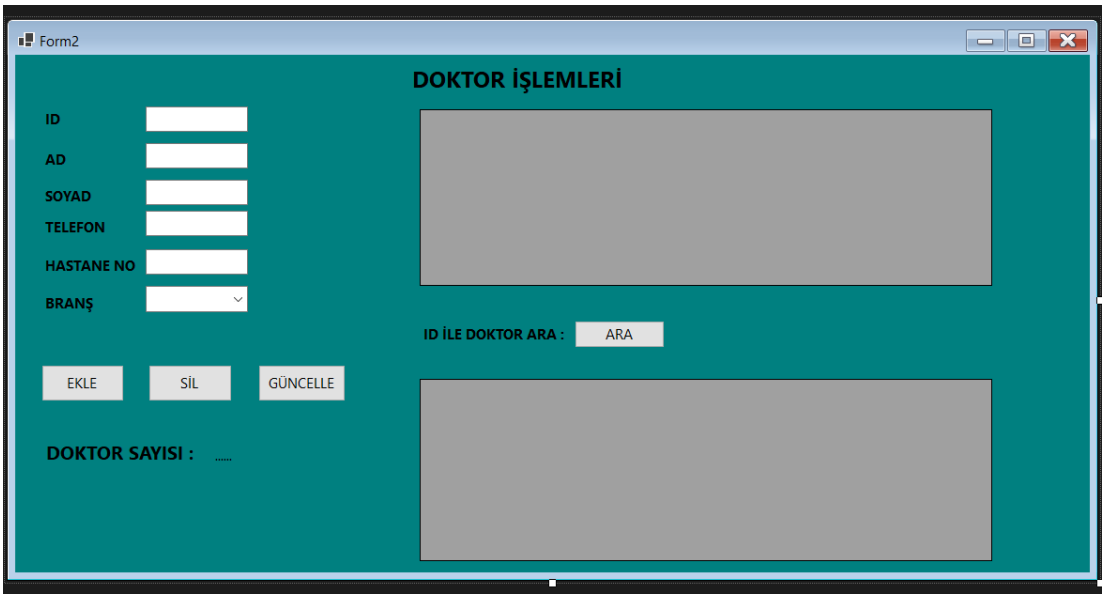
```
CREATE OR REPLACE FUNCTION "doktorDegisikligi"()
RETURNS TRIGGER
AS
$$
BEGIN
        IF NEW."brans_adi" <> OLD."brans_adi" THEN
        INSERT INTO "doktorDegisikligiIzle"("doktor_no","eskiBransNo",
       "yeniBransNo", "degisiklikTarihi")
       VALUES(OLD."doktor_no",OLD."brans_adi",NEW."brans_adi",
                CURRENT_TIMESTAMP::TIMESTAMP);

  END IF;

 RETURN NEW;
END;
$$
LANGUAGE "plpgsql";

CREATE TRIGGER "doktorBransDegistiginde"
BEFORE UPDATE ON "doktor"
FOR EACH ROW
EXECUTE PROCEDURE "doktorDegisikligi"();
```

EKRAN GÖRÜNTÜLERİ

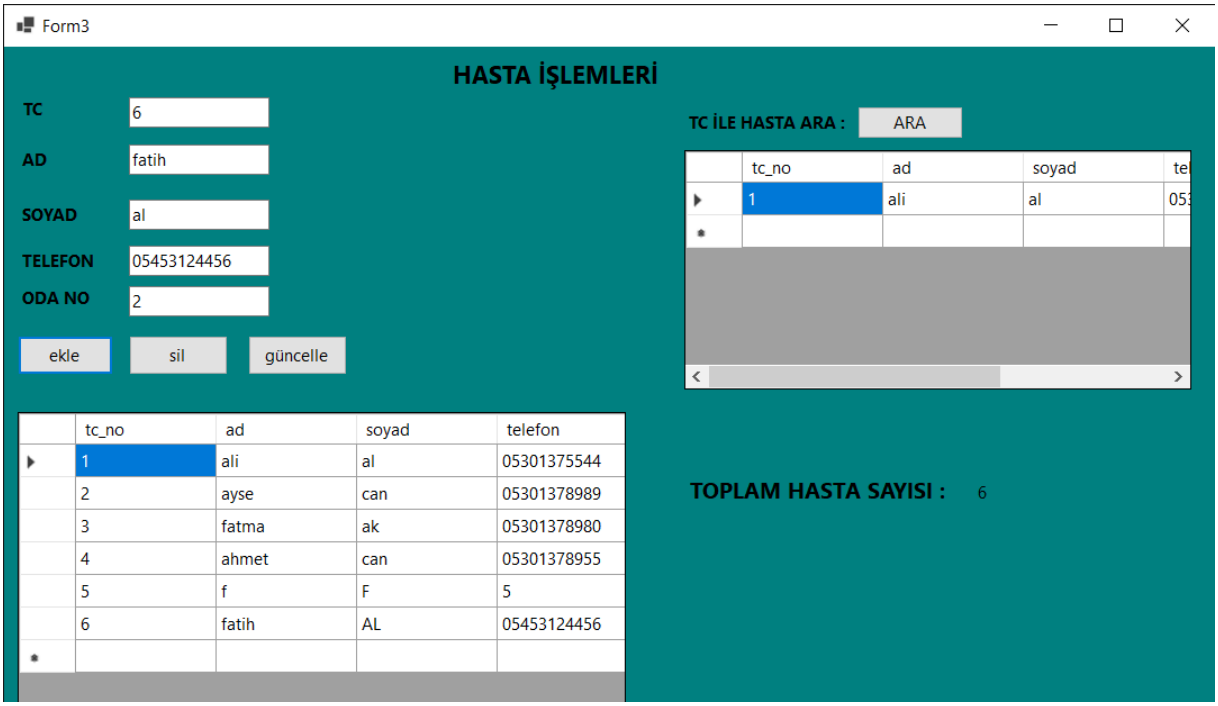
ANASAYFA:

HASTA İŞLEMLERİ SAYFASI:

DOKTOR İŞLEMLERİ SAYFASI:

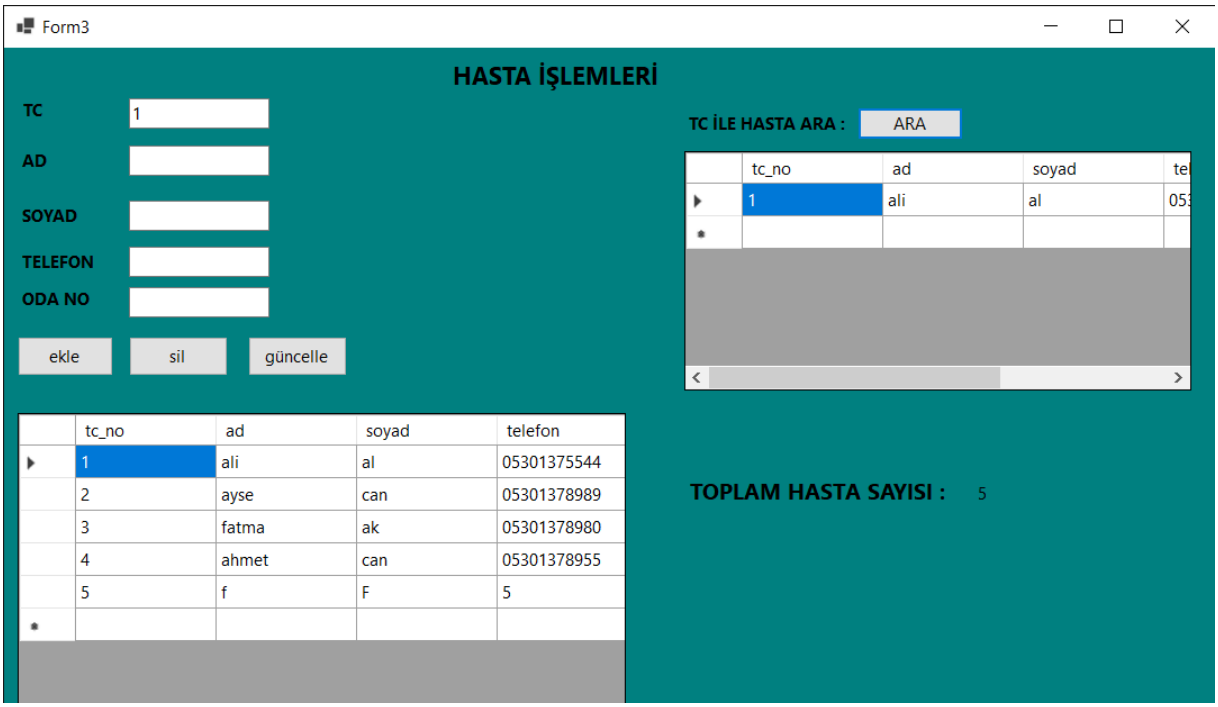


HASTA İŞLEMLERİ SAYFASI **ARAMA VE EKLEME** ÖRNEĞİ:

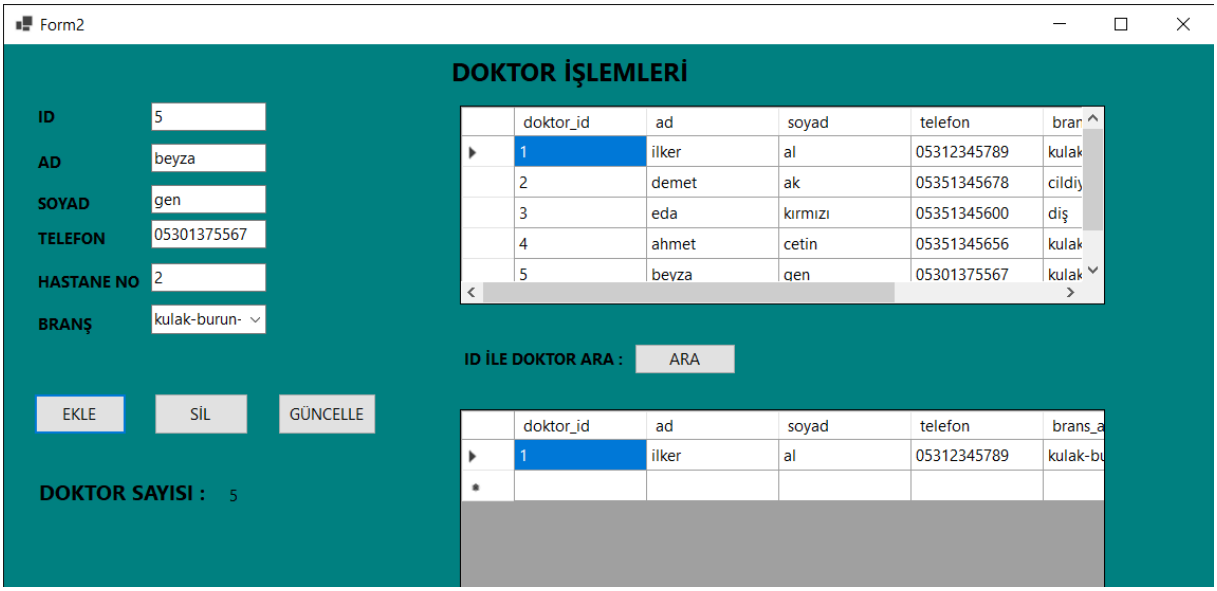


HASTA İŞLEMLERİ SAYFASI **SİLME** İŞLEMİ ÖRNEĞİ:

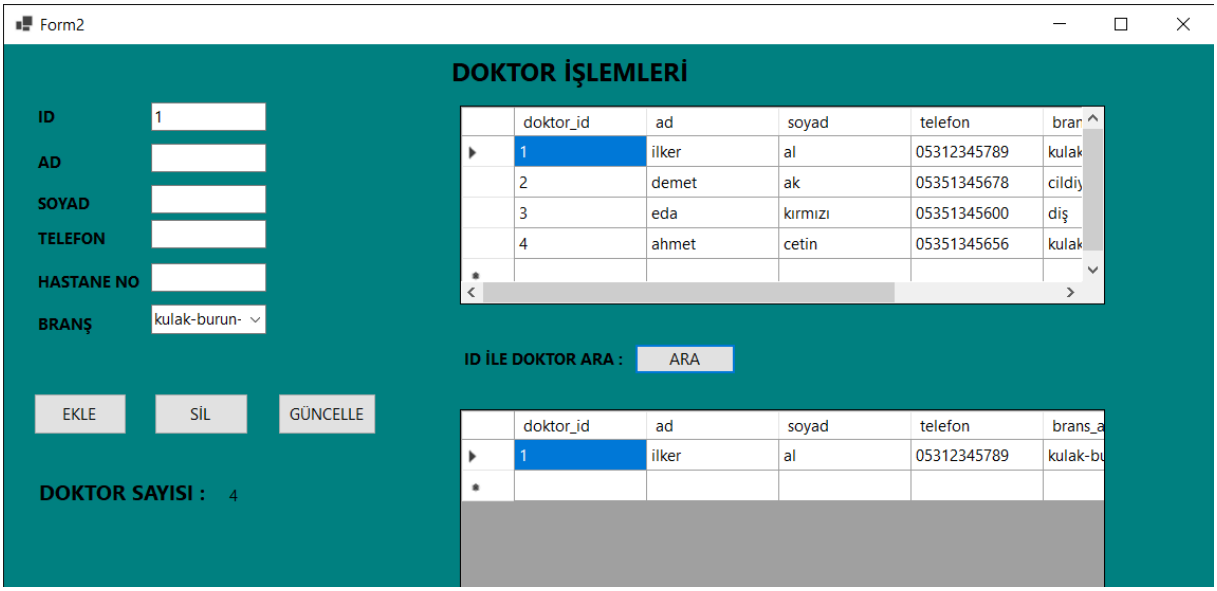
***Toplam hasta sayısı 5'e düşmüştür. Aşağıda ekran görüntüsünde gösterilmektedir. Bu fonksiyon kullanılarak yapılmıştır.



EKLEME İŞLEMİ EKRAN GÖRÜNTÜSÜ:



ID İLE DOKTOR **ARAMA** EKRAN GÖRÜNTÜSÜ:



VİDEO ADRESİ

https://youtu.be/gX1eOp9G3BU

UYGULAMA KAYNAK KODLARI

https://github.com/beyzanurodabas/vtys_proje