

15.08.2021

**BSM211VERİTABANI**

**YÖNETİM SİSTEMLERİ PROJE**

**RAPORU**

**DERS SORUMLUSU:**

**Prof.Dr. CELAL ÇEKEN**

## ÖĞRENCİ: AYŞE AKTAĞ(s200110077)

## aayseaktag@gmail.com

Bu proje kapsamında bir kitap evi için PHP, CSS ve Bootstrap kullanarak bir web sitesi oluşturulmuştur. Kitap evi için geliştirilen web sitesinde müşteri takas yapmak için teklif verebilmekte ve takas etmek istediği kitap veya kitaplar için ekleme yapabilmektedir. Satıcı, işlemi onaylarsa alışveriş işlemi gerçekleşmektedir. Kitaplara, müşterilere, satışa ait bilgiler veritabanında tutulmaktadır. Proje kapsamında veritabanı yönetimi için Postgrsql veritabanını ve sunucu olarak pgAdmin programını kullanılmıştır.

PROBLEMİN TANITIMI:

Bu projeyi geliştirirken veritabanı yaşam döngüsü adımları izlenmiştir. Agile yazılım geliştirme yöntemi kullanılmıştır. Öncelikle problemi tanımlayıp, gereksinim analizini yapılmıştır. Problem senaryosundaki önemli noktalara ve gereksinimlere bakarak iş kurallarını oluşturulmuştur.

İŞ KURALLARI:

İşleyiş, kural veya yönetmeliğin özetlenmiş iş şekli bu kısımda maddeler halinde belirtilmiştir.

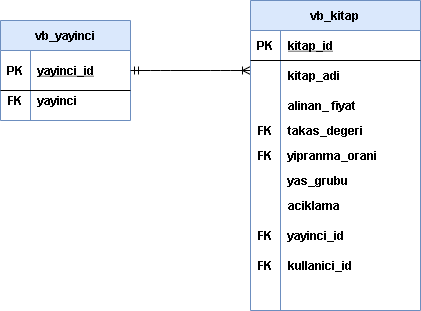
* Her kitap, kitap adı, alınan fiyatı, takas değeri, yıpranma oranı, hitap ettiği yaş grubu, kitap hakkında özetleyici bir açıklama kısmı, marka ve kullanıcı\_id , mesajlaşma bilgilerini bulundurmalıdır.
* Bir kitap bir basımevi tarafından basılır ve bir yayınevi vardır.
* Her yayın için farklı bir id tutulur.
* Her müşteri satın alma işlemi yapabilmek için sisteme kaydolmalıdır.
* Her müşteri sisteme kaydolurken kullanıcı adı ve şifre bilgisini girmek zorundadır.
* Bir müşteri çok sayıda kitap siparişi verebilir.
* Her müşterinin adı, soyadı, e-mail, telefon numarası, T.C. kimlik numarası ve adres bilgilerinin girilmesi istenir.
* Her müşteri satın aldığı kitap veya kitaplara yorum yapma hakkında sahiptir.
* Her müşteri yorum yaptığında her kullanıcı için farklı yorum id si tutulur.
* Her yorum için sistemde kitabın idsi ve kullanıcı idsi ile tutulur.
* Her müşteri satın aldığı kitap veya kitaplara puan verme hakkına sahiptir.
* Her kitap için bir puan id si bulunmalıdır. Kullanıcı puan verdiğinde sistemde kitap idsi ve kullanıcı id ile tutulur.
* Her kitap için bir resim sistemde bulunur. Kitaplar görseller ile müşteriye çağrışımda bulunur.
* Her kitap için farklı resim id si vardır.
* Kitaplar kategorilere ayrılmıştır. Her kategori için bir id tanımlıdır.
* Müşteriler kitapları takas edebilmektedir. Her müşteri takas işlemi yapmak istediğinde teklif vermelidir.
* Takas yapılabilmesi için her takas işlemine farklı id girilmelidir.
* Takas işleminde her müşteri teklifi için bir takas id tutulur.
* Teklif kabul edilirse kabul, teslim tarihi bilgisi her müşteri için tutulur.
* Her müşteri bir veya birden fazla kitap için fiyat teklifi verebilir.
* Her müşteri teklif vermek için isterse açıklama bilgisi girebilir.
* Her müşteri, her kitap için puan verebilir, yorum yapabilir, teklif verebilir.
* Her teklif için veritabanında farklı idler ile bilgiler tutulmalıdır.
* Her teklif için veritabanında teklifin tarihi, açıklama, iş durumu, teklifi gönderen kişinin id bilgi, alan kişinin id bilgisi ve kitabın id bilgisi bulunmalıdır.
* Her müşteri yaptığı her siparişi için kargo ücreti ödemelidir.
* Veritabanında her kargo firması için farklı idler ile kargo bilgileri tutulmalıdır.
* Her kargo firmasının kendine ait bir id si vardır.
* Her kargo için bir kargo\_id ve ücret bilgisi girilmelidir.
* Her müşteri her kitap için satıcıyla iletişime geçebilir. Web sitesinde her kitap için iletişim butonu bulunur.
* Kitaplar kategorilere ayrılmıştır. Bir kitap birçok farklı kategoriye ait olabilir.
* Gönderi ve kategorileri sadece yetkili kişi ekleyebilir.
* Bir kitap birçok yayınevi tarafından yayınlanabilir.

İş kuralları oluşturulduktan sonra, gerçekleştirilecek veritabanının modellenmesi aşamasına geçilmiştir.

* Bir kitap birçok yayınevi tarafından yayınlanabilir.

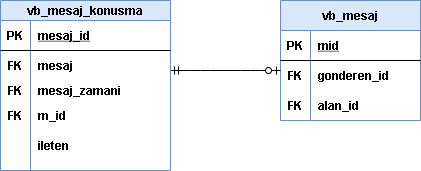
İLİŞKİSEL ŞEMA:

* Bir yayınevi birçok kitap yayınlayabilir.

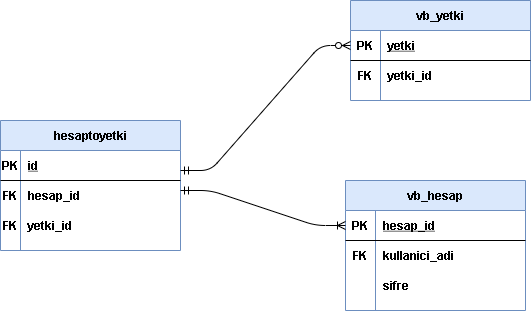


* Vb\_yayinci(yayinci :Character vaying(50), yayinci\_id: Integer)

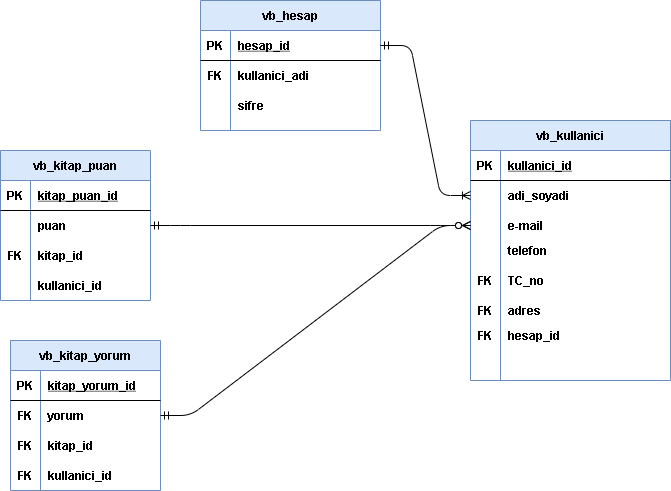
* Vb\_kitap(kitap\_id:Integer,kitap\_adi: Character varying(50), alinan\_fiyat: Numeric, takas\_degeri :Numeric, yipranma\_orani :Integer, yasgrubu: Character varying(6), aciklama : Character varying, yayinci\_id: Integer, kullanici\_id : Integer)
* Bir mesajın en az bir gönderen kişisi vardır.
* Bir mesajın en az bir alan kişisi vardır.



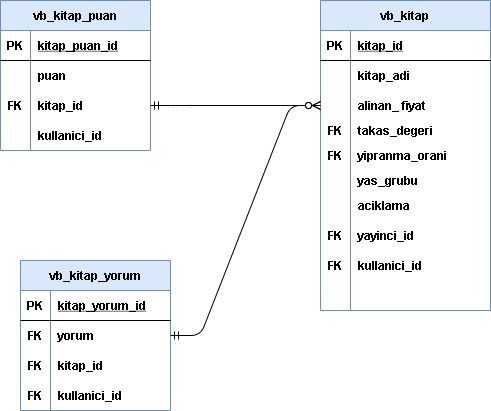
* vb\_mesaj(mid :integer, gonderen\_id: integer,alan\_id: integer)
* vb\_mesaj\_konusma(mesaj\_id: integer, mesaj character varying, mesaj\_zamani: timestamp without time zone, m\_id: integer, ileten integer)
* Her hesap en az bir veya çok yetkiye sahiptir.
* Yetki sahibi en az bir veya çok hesap vardır.
* Her hesabın en çok bir hesap\_id’si vardır.Bu id ler ile birbirlerinden ayrılırlar. Hesap olması için id olması zorunludur.( varolma bağımlılığı)
* Her hesap id en çok bir hesaba aittir.



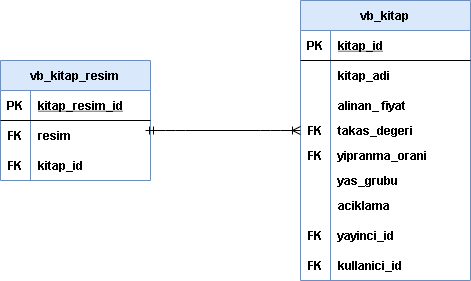
* Hesaptoyetki(id: integer, hesap\_id :integer, yetki\_id: integer)
* Vb\_yetki(yetki\_id: integer, yetki: character varying(20))
* Vb\_hesap(hesap\_id :integer, kullanici\_adi :character varying(30), sifre : character varying(30),)
* Her hesabın en çok bir kullanıcı adı-soyadı olmalıdır.
* Adı-soyadı olan her kullanıcının bir hesabı olmalıdır.
* Her kullanıcı e-mail ile kaydolarak bir veya çok kitaba puan verebilir.
* Bir veya çok puan verebilmek için her kullanıcı e-mail ile kaydolmalıdır.
* Bir kullanıcı e-mail ile kaydolarak bir veya çok kitaba yorum yapabilir.
* Bir veya çok yorum yapabilmek için her kullanıcı e-mail ile kaydolmalıdır.



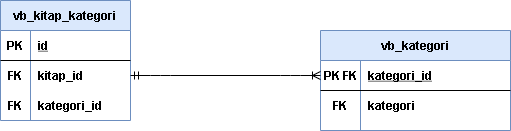
* Vb\_hesap(hesap\_id :integer, kullanici\_adi :character varying(30), sifre : character varying(30),)
* Vb\_kitap\_puan(kitap\_puan\_id :integer, puan: integer,kitap\_id: integer, kullanici\_id :integer)
* Vb\_kitap\_yorum(kitap\_yorum\_id: integer, yorum :character varying, kitap\_id :integer, kullanici\_id: integer)
* Vb\_kullanici(kullanici\_id :integer, adi\_soyadi: character varying(50), email : character varying(40),, telefon : character varying(11),, tc\_no: character varying(11),, adres: char, hesap\_id: integer)
* Her kitabın bir alış fiyatı vardır.
* Alış fiyatı her kitap için bir tanedir.
* Birçok kitap aynı fiyatta olabilir.
* Aynı fiyatta birçok kitap olabilir.
* Bir kitabın takası için kullanıcı birçok yorum yapabilir.



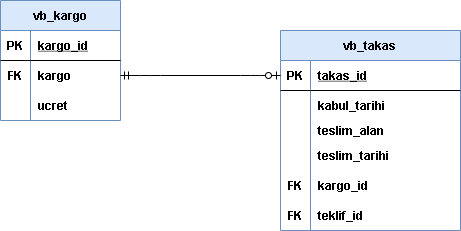
* Vb\_kitap\_puan(kitap\_puan\_id :integer, puan: integer,kitap\_id: integer, kullanici\_id :integer)
* Vb\_kitap\_yorum(kitap\_yorum\_id: integer, yorum :character varying, kitap\_id :integer, kullanici\_id: integer)
* Vb\_kitap(kitap\_id:Integer,kitap\_adi: Character varying(50), alinan\_fiyat: Numeric, takas\_degeri :Numeric, yipranma\_orani :Integer, yasgrubu: Character varying(6), aciklama : Character varying, yayinci\_id: Integer, kullanici\_id : Integer)
* Takas edilecek her kitabın bir resmi vardır.
* Resim olan her kitap takas edilebilir.



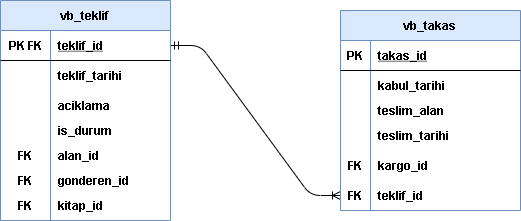
* Vb\_kitap(kitap\_id:Integer,kitap\_adi: Character varying(50), alinan\_fiyat: Numeric, takas\_degeri :Numeric, yipranma\_orani :Integer, yasgrubu: Character varying(6), aciklama : Character varying, yayinci\_id: Integer, kullanici\_id : Integer)
* Vb\_kitap\_resim(kitap\_resim\_id: integer, resim: character varying(50), kitap\_id :integer)
* Bir kitap birçok kategoride olabilir.
* Birçok kategori bir kitabı içerebilir.



* Vb\_kitap\_kategori(id: integer, kitap\_id: integer, kategori\_id :integer)
* Vb\_kategori(kategori\_id :integer, kategori :character varying(80))
* Takası kabul edilen kitaplar kargolanır.
* Bir kitabın kargolanması için bir kabul tarihi olmalıdır.



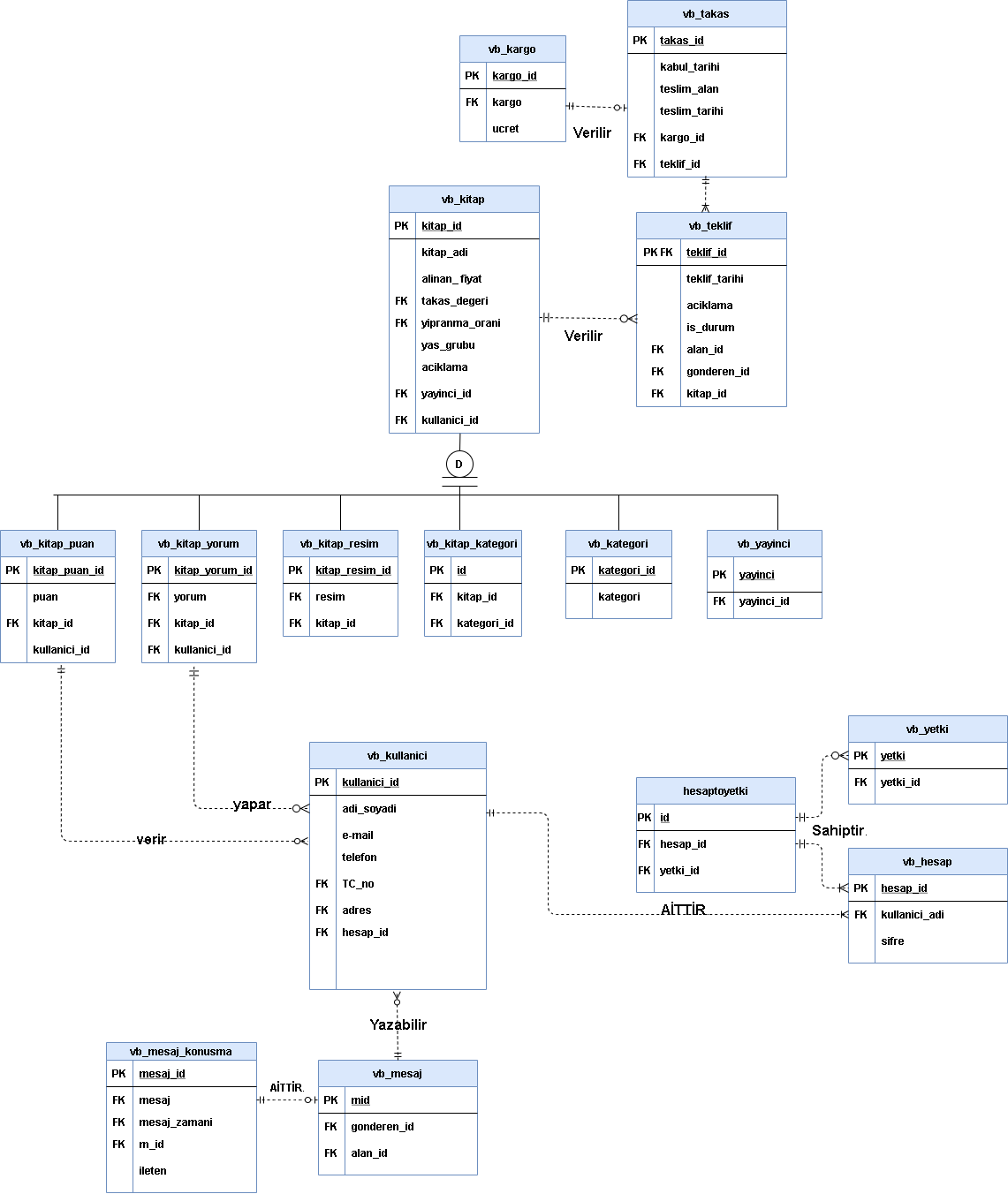
* Vb\_kargo(kargo\_id: integer, kargo :character varying(40), ucret: numeric)
* Vb\_takas(takas\_id: integer, kabul\_tarihi :timestamp without time zone, teslim\_alan :character varying(60), teslim\_tarihi :timestamp without time zone, kargo\_id :integer, teklif\_id :integer)
* Bir kitabın takas edilebilmesi için en az bir teklifi olmalıdır.
* Bir teklifi olan kitap bir kişiyle takas edilebilir.



* Vb\_takas(takas\_id: integer, kabul\_tarihi :timestamp without time zone, teslim\_alan :character varying(60), teslim\_tarihi :timestamp without time zone, kargo\_id :integer, teklif\_id :integer)
* Vb\_teklif(teklif\_id :integer, teklif\_tarihi :timestamp without time zone, aciklama character varying, is\_durum :integer, gonderen\_id :integer, alan\_id :integer,kitap\_id :integer)

VARLIK BAĞINTI MODELİ:

* i) Chen gösterimi ve ii) Crow’s Foot gösterimi sıkça kullanılan gösterim şekillerindendir.
* Proje kapsamında Crow’s Foot gösterimi kullanmamızı istemiştiniz.Bu sebeple Varlık Bağıntı Modeli kullanılmış ve Crow’s Foot gösterimi için draw.io programı tercih edilmiştir. Çizim aşağıdaki gibidir:



SQL KOMUTLARI

Hesaptoyetki sql komutları:

-- Table: public.hesaptoyetki

-- DROP TABLE public.hesaptoyetki;

CREATE TABLE public.hesaptoyetki

(

id integer NOT NULL DEFAULT nextval('hesaptoyetki\_id\_seq'::regclass),

hesap\_id integer,

yetki\_id integer,

CONSTRAINT hesaptoyetki\_pkey PRIMARY KEY (id),

CONSTRAINT hesaptoyetki\_hesap\_id\_fkey FOREIGN KEY (hesap\_id)

REFERENCES public.vb\_hesap (hesap\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT hesaptoyetki\_yetki\_id\_fkey FOREIGN KEY (yetki\_id)

REFERENCES public.vb\_yetki (yetki\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.hesaptoyetki

OWNER to postgres;

Vb\_hesap sql komutları:

-- Table: public.vb\_hesap

-- DROP TABLE public.vb\_hesap;

CREATE TABLE public.vb\_hesap

(

hesap\_id integer NOT NULL DEFAULT nextval('vb\_hesap\_hesap\_id\_seq'::regclass),

kullanici\_adi character varying(30) COLLATE pg\_catalog."default" NOT NULL,

sifre character varying(30) COLLATE pg\_catalog."default" NOT NULL,

CONSTRAINT vb\_hesap\_pkey PRIMARY KEY (hesap\_id)

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_hesap

OWNER to postgres;

Vb\_kargo sql komutları:

-- Table: public.vb\_kargo

-- DROP TABLE public.vb\_kargo;

CREATE TABLE public.vb\_kargo

(

kargo\_id integer NOT NULL DEFAULT nextval('vb\_kargo\_kargo\_id\_seq'::regclass),

kargo character varying(40) COLLATE pg\_catalog."default" NOT NULL,

ucret numeric DEFAULT 0,

CONSTRAINT vb\_kargo\_pkey PRIMARY KEY (kargo\_id)

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_kargo

OWNER to postgres;

Vb\_kategori sql komutları:

-- Table: public.vb\_kategori

-- DROP TABLE public.vb\_kategori;

CREATE TABLE public.vb\_kategori

(

kategori\_id integer NOT NULL DEFAULT nextval('vb\_kategori\_kategori\_id\_seq'::regclass),

kategori character varying(80) COLLATE pg\_catalog."default" NOT NULL,

CONSTRAINT vb\_kategori\_pkey PRIMARY KEY (kategori\_id)

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_kategori

OWNER to postgres;

-- Trigger: test2trigger

-- DROP TRIGGER test2trigger ON public.vb\_kategori;

CREATE TRIGGER test2trigger

AFTER INSERT

ON public.vb\_kategori

FOR EACH ROW

EXECUTE FUNCTION public.test2();

Vb\_kitap sql komutları:

-- Table: public.vb\_kitap

-- DROP TABLE public.vb\_kitap;

CREATE TABLE public.vb\_kitap

(

kitap\_id integer NOT NULL DEFAULT nextval('vb\_kitap\_kitap\_id\_seq'::regclass),

kitap\_adi character varying(50) COLLATE pg\_catalog."default" NOT NULL,

alinan\_fiyat numeric DEFAULT 0,

takas\_degeri numeric DEFAULT 0,

yipranma\_orani integer,

yasgrubu character varying(6) COLLATE pg\_catalog."default" NOT NULL,

aciklama character varying COLLATE pg\_catalog."default",

yayinci\_id integer,

kullanici\_id integer,

CONSTRAINT vb\_kitap\_pkey PRIMARY KEY (kitap\_id),

CONSTRAINT vb\_kitap\_kullanici\_id\_fkey FOREIGN KEY (kullanici\_id)

REFERENCES public.vb\_kullanici (kullanici\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_kitap\_yayinci\_id\_fkey FOREIGN KEY (yayinci\_id)

REFERENCES public.vb\_yayinci (yayinci\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_kitap\_yipranma\_orani\_check CHECK (yipranma\_orani >= 1 AND yipranma\_orani <= 100)

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_kitap

OWNER to postgres;

-- Trigger: testtrig

-- DROP TRIGGER testtrig ON public.vb\_kitap;

CREATE TRIGGER testtrig

AFTER INSERT

ON public.vb\_kitap

FOR EACH ROW

EXECUTE FUNCTION public.test();

-- Trigger: testtrigger

-- DROP TRIGGER testtrigger ON public.vb\_kitap;

CREATE TRIGGER testtrigger

AFTER INSERT

ON public.vb\_kitap

FOR EACH ROW

EXECUTE FUNCTION public.test();Vb\_kitap\_puan sql komutları:

-- Table: public.vb\_kitap\_puan

-- DROP TABLE public.vb\_kitap\_puan;

CREATE TABLE public.vb\_kitap\_puan

(

kitap\_puan\_id integer NOT NULL DEFAULT nextval('vb\_kitap\_puan\_kitap\_puan\_id\_seq'::regclass),

puan integer,

kitap\_id integer,

kullanici\_id integer,

CONSTRAINT vb\_kitap\_puan\_pkey PRIMARY KEY (kitap\_puan\_id),

CONSTRAINT vb\_kitap\_puan\_kitap\_id\_fkey FOREIGN KEY (kitap\_id)

REFERENCES public.vb\_kitap (kitap\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_kitap\_puan\_kullanici\_id\_fkey FOREIGN KEY (kullanici\_id)

REFERENCES public.vb\_kullanici (kullanici\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_kitap\_puan\_puan\_check CHECK (puan >= 1 AND puan <= 10)

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_kitap\_puan

OWNER to postgres;

Vb\_kitap\_resim sql komutları:

-- Table: public.vb\_kitap\_resim

-- DROP TABLE public.vb\_kitap\_resim;

CREATE TABLE public.vb\_kitap\_resim

(

kitap\_resim\_id integer NOT NULL DEFAULT nextval('vb\_kitap\_resim\_kitap\_resim\_id\_seq'::regclass),

resim character varying(50) COLLATE pg\_catalog."default" NOT NULL,

kitap\_id integer,

CONSTRAINT vb\_kitap\_resim\_pkey PRIMARY KEY (kitap\_resim\_id),

CONSTRAINT vb\_kitap\_resim\_kitap\_id\_fkey FOREIGN KEY (kitap\_id)

REFERENCES public.vb\_kitap (kitap\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_kitap\_resim

OWNER to postgres;

Vb\_kitap\_yorum sql komutları:

-- Table: public.vb\_kitap\_yorum

-- DROP TABLE public.vb\_kitap\_yorum;

CREATE TABLE public.vb\_kitap\_yorum

(

kitap\_yorum\_id integer NOT NULL DEFAULT nextval('vb\_kitap\_yorum\_kitap\_yorum\_id\_seq'::regclass),

yorum character varying COLLATE pg\_catalog."default",

kitap\_id integer,

kullanici\_id integer,

CONSTRAINT vb\_kitap\_yorum\_pkey PRIMARY KEY (kitap\_yorum\_id),

CONSTRAINT vb\_kitap\_yorum\_kitap\_id\_fkey FOREIGN KEY (kitap\_id)

REFERENCES public.vb\_kitap (kitap\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_kitap\_yorum\_kullanici\_id\_fkey FOREIGN KEY (kullanici\_id)

REFERENCES public.vb\_kullanici (kullanici\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_kitap\_yorum

OWNER to postgres;

Vb\_kitaptokategori sql komutları:

-- Table: public.vb\_kitaptokategori

-- DROP TABLE public.vb\_kitaptokategori;

CREATE TABLE public.vb\_kitaptokategori

(

id integer NOT NULL DEFAULT nextval('vb\_kitaptokategori\_id\_seq'::regclass),

kitap\_id integer,

kategori\_id integer,

CONSTRAINT vb\_kitaptokategori\_pkey PRIMARY KEY (id),

CONSTRAINT vb\_kitaptokategori\_kategori\_id\_fkey FOREIGN KEY (kategori\_id)

REFERENCES public.vb\_kategori (kategori\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_kitaptokategori\_kitap\_id\_fkey FOREIGN KEY (kitap\_id)

REFERENCES public.vb\_kitap (kitap\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_kitaptokategori

OWNER to postgres;

Vb\_kullanici sql komutları:

-- Table: public.vb\_kullanici

-- DROP TABLE public.vb\_kullanici;

CREATE TABLE public.vb\_kullanici

(

kullanici\_id integer NOT NULL DEFAULT nextval('vb\_kullanici\_kullanici\_id\_seq'::regclass),

adi\_soyadi character varying(50) COLLATE pg\_catalog."default" NOT NULL,

email character varying(40) COLLATE pg\_catalog."default" NOT NULL,

telefon character varying(11) COLLATE pg\_catalog."default" NOT NULL,

tc\_no character varying(11) COLLATE pg\_catalog."default" NOT NULL,

adres character varying COLLATE pg\_catalog."default",

hesap\_id integer,

CONSTRAINT vb\_kullanici\_pkey PRIMARY KEY (kullanici\_id),

CONSTRAINT vb\_kullanici\_hesap\_id\_fkey FOREIGN KEY (hesap\_id)

REFERENCES public.vb\_hesap (hesap\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_kullanici

OWNER to postgres;

Vb\_mesaj sql komutları:

-- Table: public.vb\_mesaj

-- DROP TABLE public.vb\_mesaj;

CREATE TABLE public.vb\_mesaj

(

mid integer NOT NULL DEFAULT nextval('vb\_mesaj\_mid\_seq'::regclass),

gonderen\_id integer,

alan\_id integer,

CONSTRAINT vb\_mesaj\_pkey PRIMARY KEY (mid),

CONSTRAINT vb\_mesaj\_alan\_id\_fkey FOREIGN KEY (alan\_id)

REFERENCES public.vb\_kullanici (kullanici\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_mesaj\_gonderen\_id\_fkey FOREIGN KEY (gonderen\_id)

REFERENCES public.vb\_kullanici (kullanici\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_mesaj

OWNER to postgres;

Vb\_mesaj\_konusma sql komutları:

-- Table: public.vb\_mesaj\_konusma

-- DROP TABLE public.vb\_mesaj\_konusma;

CREATE TABLE public.vb\_mesaj\_konusma

(

mesaj\_id integer NOT NULL DEFAULT nextval('vb\_mesaj\_konusma\_mesaj\_id\_seq'::regclass),

mesaj character varying COLLATE pg\_catalog."default" NOT NULL,

mesaj\_zamani timestamp without time zone DEFAULT CURRENT\_DATE,

m\_id integer,

ileten integer,

CONSTRAINT vb\_mesaj\_konusma\_pkey PRIMARY KEY (mesaj\_id),

CONSTRAINT vb\_mesaj\_konusma\_m\_id\_fkey FOREIGN KEY (m\_id)

REFERENCES public.vb\_mesaj (mid) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_mesaj\_konusma

OWNER to postgres;

Vb\_takas sql komutları:

-- Table: public.vb\_takas

-- DROP TABLE public.vb\_takas;

CREATE TABLE public.vb\_takas

(

takas\_id integer NOT NULL DEFAULT nextval('vb\_takas\_takas\_id\_seq'::regclass),

kabul\_tarihi timestamp without time zone DEFAULT CURRENT\_DATE,

teslim\_alan character varying(60) COLLATE pg\_catalog."default",

teslim\_tarihi timestamp without time zone,

kargo\_id integer,

teklif\_id integer,

CONSTRAINT vb\_takas\_pkey PRIMARY KEY (takas\_id),

CONSTRAINT vb\_takas\_teklif\_id\_key UNIQUE (teklif\_id),

CONSTRAINT vb\_takas\_kargo\_id\_fkey FOREIGN KEY (kargo\_id)

REFERENCES public.vb\_kargo (kargo\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_takas\_teklif\_id\_fkey FOREIGN KEY (teklif\_id)

REFERENCES public.vb\_teklif (teklif\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_takas

OWNER to postgres;

Vb\_teklif sql komutları:

-- Table: public.vb\_teklif

-- DROP TABLE public.vb\_teklif;

CREATE TABLE public.vb\_teklif

(

teklif\_id integer NOT NULL DEFAULT nextval('vb\_teklif\_teklif\_id\_seq'::regclass),

teklif\_tarihi timestamp without time zone DEFAULT CURRENT\_DATE,

aciklama character varying COLLATE pg\_catalog."default",

is\_durum integer DEFAULT 0,

gonderen\_id integer,

alan\_id integer,

kitap\_id integer,

CONSTRAINT vb\_teklif\_pkey PRIMARY KEY (teklif\_id),

CONSTRAINT vb\_teklif\_alan\_id\_fkey FOREIGN KEY (alan\_id)

REFERENCES public.vb\_kullanici (kullanici\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_teklif\_gonderen\_id\_fkey FOREIGN KEY (gonderen\_id)

REFERENCES public.vb\_kullanici (kullanici\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT vb\_teklif\_kitap\_id\_fkey FOREIGN KEY (kitap\_id)

REFERENCES public.vb\_kitap (kitap\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_teklif

OWNER to postgres;

Vb\_yayinci sql komutları:

-- Table: public.vb\_yayinci

-- DROP TABLE public.vb\_yayinci;

CREATE TABLE public.vb\_yayinci

(

yayinci\_id integer NOT NULL DEFAULT nextval('vb\_yayinci\_yayinci\_id\_seq'::regclass),

yayinci character varying(50) COLLATE pg\_catalog."default" NOT NULL,

CONSTRAINT vb\_yayinci\_pkey PRIMARY KEY (yayinci\_id)

)

TABLESPACE pg\_default;

ALTER TABLE public.vb\_yayinci

OWNER to postgres;

Vb\_yetki sql komutları:

-- Table: public.vb\_yetki

-- DROP TABLE public.vb\_yetki;

CREATE TABLE public.vb\_yetki

(

yetki\_id integer NOT NULL DEFAULT nextval('vb\_yetki\_yetki\_id\_seq'::regclass),

yetki character varying(20) COLLATE pg\_catalog."default",

CONSTRAINT vb\_yetki\_pkey PRIMARY KEY (yetki\_id)

)

TABLESPACE pg\_default;

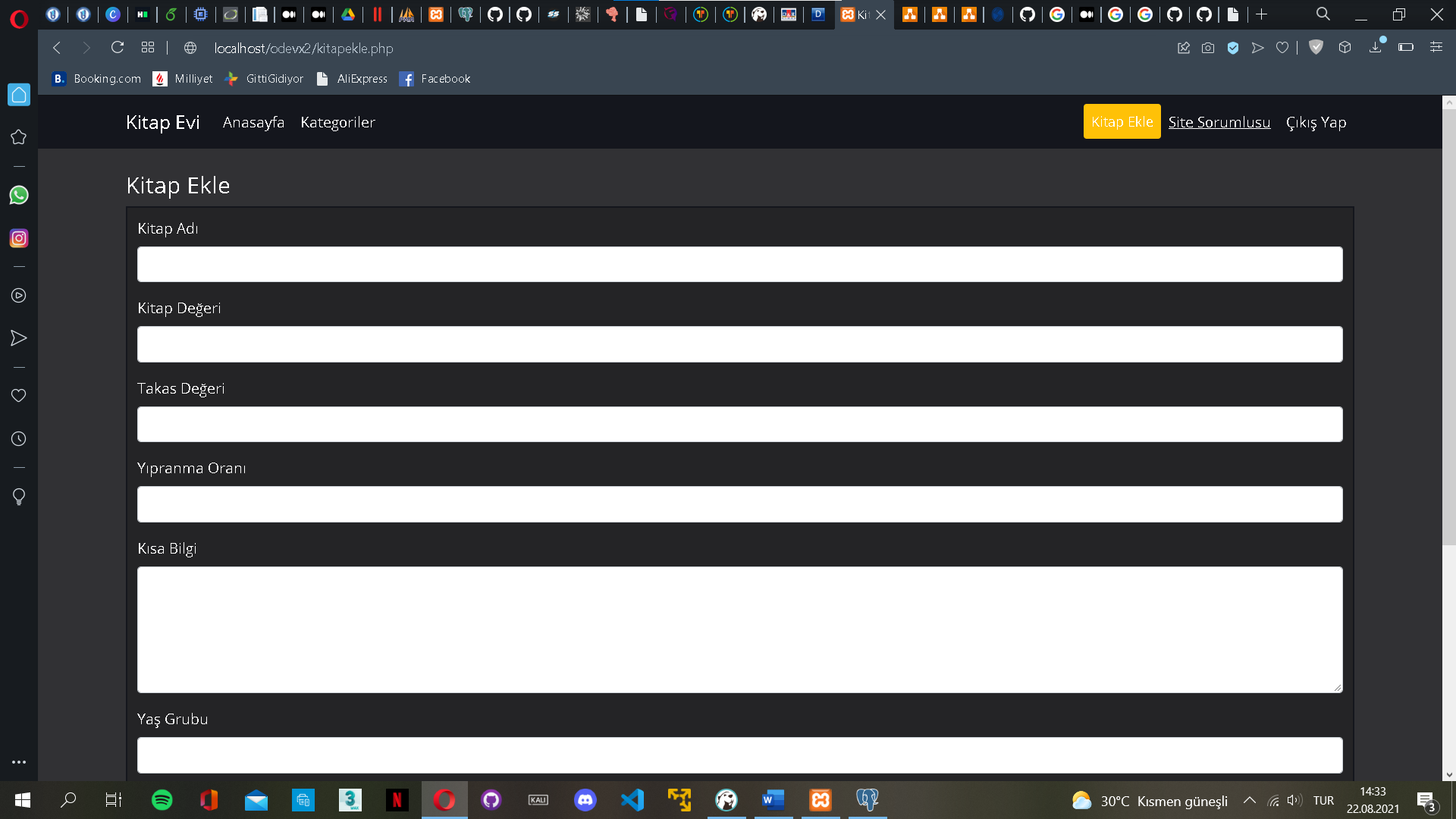
ALTER TABLE public.vb\_yetki

OWNER to postgres;

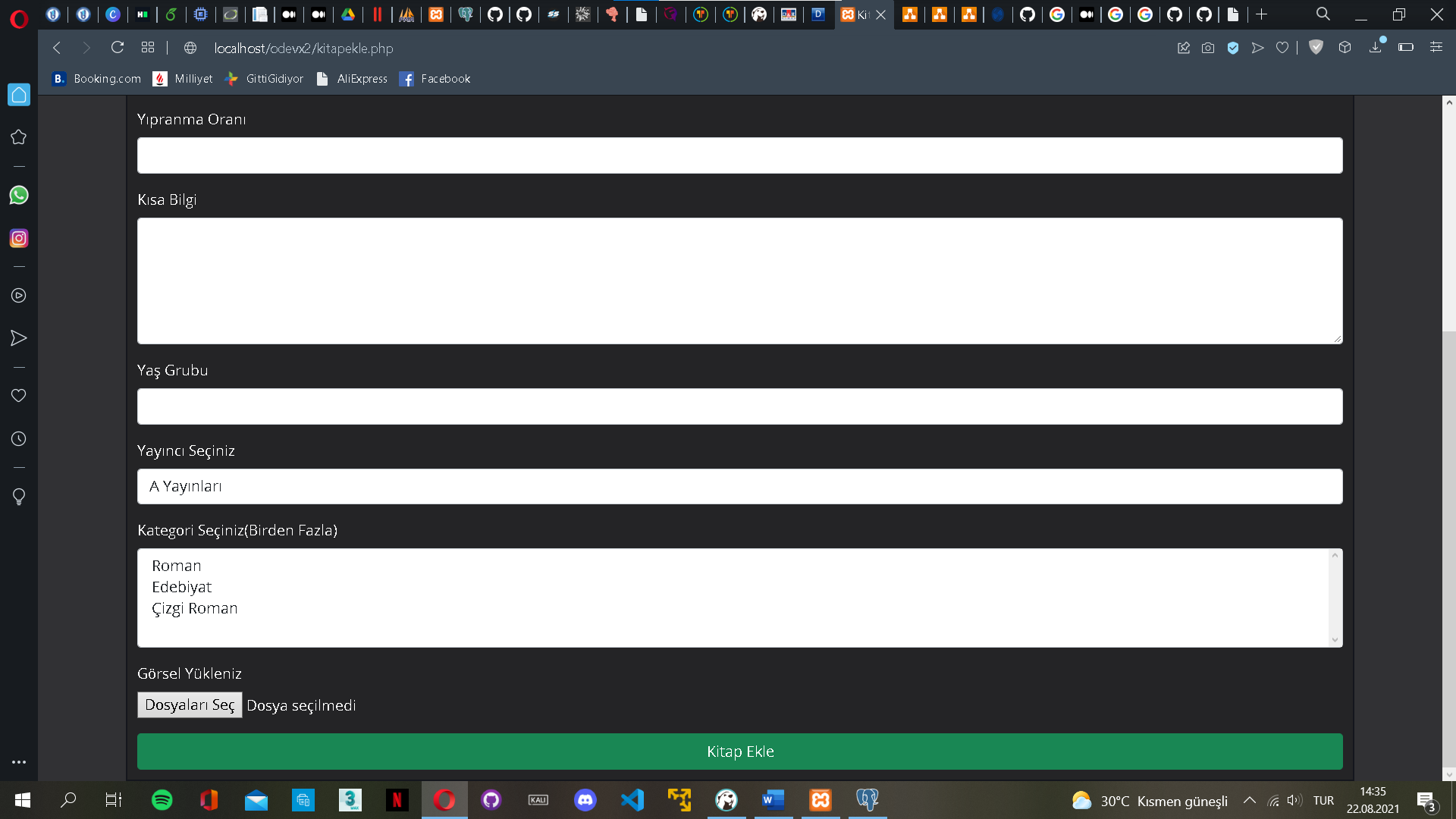
SAKLI YORDAM VE TETİKLEYİCİLER

**STORED PROCEDURE1**

Kitap bilgilerinin sisteme girileceği arayüz Şekil.1ve2’de verilmişitir.



Şekil1.Kitap Kayıt Ekranı



Şekil2.Kitap Kayıt Ekranı Devamı

Vb\_kitap tablosuna bilgi kaydetmek için kullanılan saklı yordam aşağıdaki gibidir.

Create procedure vb\_kitap\_kayıt( kitap\_id Integer, kitap\_adi Character varying(50), alinan\_fiyat Numeric, takas\_degeri Numeric, yipranma\_orani Integer, yasgrubu Character varying(6), aciklama Character varying, yayinci\_id Integer, kullanici\_id Integer)

language sql

As $$

insert into vb\_kitap(id,ad,fiyat,takasDegeri,yıpranmaOrani,yas,aciklama,yayinci,kullanici) values(kitap\_id,kitap\_adi, alinan\_fiyat,takas\_degeri,yipranma\_orani,yasgrubu,aciklama,yayinci\_id,kullanici\_id);

$$;

**STORED PROCEDURE2**

Create procedure vb\_hesap\_ekle( kategori\_id integer,kategori character varying(80))

Language SQL

As $$

insert into vb\_kategori(kategori\_id,kategori) values(kategori\_id,kategori);

$$;



Şekil.3.Saklıyordam ile Veri Ekleme

**STORED PROCEDURE3**

Create procedure vb\_kitap\_puan\_ekle( kitap\_puan\_id integer,puan integer,kitap\_id integer,kullanici\_id integer)

Language SQL

As $$

insert into vb\_kitap\_puan(kitap\_puan\_id,puan,kitap\_id,kullanici\_id) values(kitap\_puan\_id,puan,kitap\_id,kullanici\_id);

$$;



Şekil.4. Saklıyordam ile veri değiştirme

**STORED PROCEDURE4:**

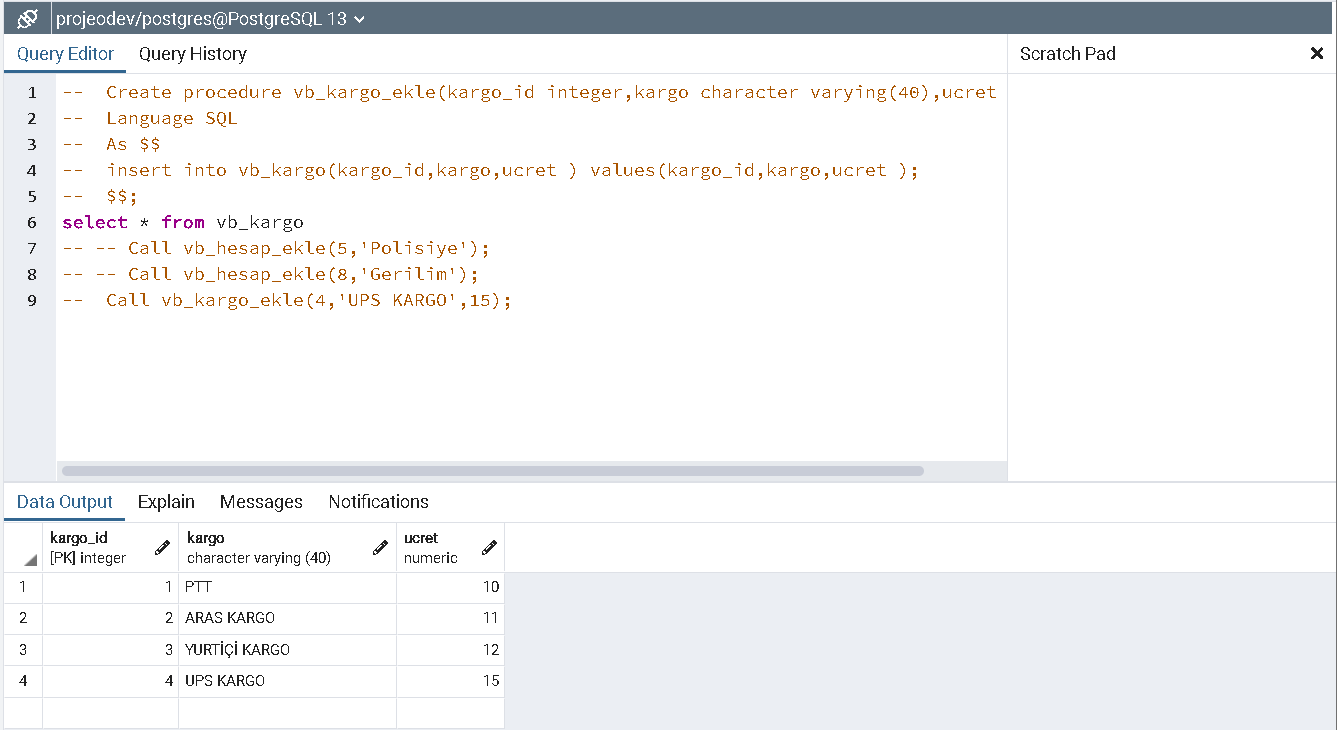
Create procedure vb\_kargo\_ekle(kargo\_id integer,kargo character varying(40),ucret numeric)

Language SQL

As $$

insert into vb\_kargo(kargo\_id,kargo,ucret ) values(kargo\_id,kargo,ucret );

$$;



Şekil.5. Saklıyordam ile vb\_kargo tablosuna veri ekleme

**TRIGGER1:**

Bu trigger için toplam\_kitap isimli tablo sonradan veritabanına eklenmiştir.

Vb\_kitap tablomuza herhangi bir veri kaydı yaptıktan sonra toplam\_kitap tablomuzda yer alan sayi sütunu 1 artsın ve bu otomatik olarak gerçekleşsin diye bu trigger oluşturuldu.

create or replace function test()

returns trigger

as

$$

begin

update toplam\_kitap set sayi=sayi+1;

return new;

end;

$$

language plpgsql;

---trigger olusturma

create trigger testtrigger

after insert

on vb\_kitap

for each row

execute procedure test();

//trigger,vb\_kitap tablosunda insert işlemi yapıldıktan sonra çalışacak.Her satır için vb\_kitap tablosuna ekleme yapıldıktan sonra test() prosedürünü çalıştıracak.

insert into vb\_kitap( kitap\_id ,kitap\_adi,alinan\_fiyat,takas\_degeri,yipranma\_orani ,yasgrubu ,aciklama ,yayinci\_id,kullanici\_id) values(27,'Yaşlı Adam ve Deniz',20,15,10,12,'Yaşlı Adam ve Deniz, HEMINGWAY ait en ölümsüz eserlerinden biridir',1,3)

Yukarıda yazılan komut ile vb\_kitap tablosuna ekleme yaparak toplam\_kitap tablosunun da +1 artması istenmiştir.

Select \* from toplam\_kitap yazarak değerin 22 iken +1 artarak 23 olduğu görülmüştür. Yazılan trigger başarılı bir şekilde çalışmıştır.

**TRIGGER2:**

Bu trigger için toplam\_kategori isimli tablo sonradan veritabanına eklenmiştir.

Vb\_kategori tablosuna yeni bir kategori eklediğimiz zaman bu kategorinin adı toplam kaç karakter ise toplam\_kategoriki sütunun o kadar artması için bir tetikleyici oluşturulmuştur. Dışarıdan bir parametreye bağlı olarak artacaktır.

--Önce fonksiyon oluşturulup ardından trigger oluşturulmuştur.

create or replace function test2()

returns trigger

as

$$

declare

uzunluk integer;

begin

uzunluk:=(select length(kategori) from vb\_kategori order by kategori\_id desc limit 1);

--Yukarıdaki komut ile limiti 1 seçerek son eklenen değeri çağırmış olduk.

update toplam\_kategori set sayi=sayi+uzunluk;

return new;

end;

$$

language plpgsql;

--Trigger oluşturma:

create trigger test2trigger

after insert

-- trigger insert işleminden sonra çalışacak.

on vb\_kategori

--vb\_kategori tablosu üzerinde çalışacak.

for each row

--her satır için

execute procedure test2();

--Triggerı oluşturduktan sonra:

select \* from toplam\_kategori

diyerek toplam\_kategorideki 0 değeri görüntülenmiştir.

insert into vb\_kategori(kategori\_id,kategori) values(9,'Bilim Kurgu');

Komutuyla vb\_kategoriye yeni veri ekleyerek sayının 0 iken ‘Bilim Kurgu’ daki karakter toplamı olan 11’e güncellenmesi beklenmiştir.



Şekil.6.Karakter Sayısı 11

insert into vb\_kategori(kategori\_id,kategori) values(10,'Dergi');

Komutuyla yeniden çalıştırıldığında, ‘Dergi’ kelimesindeki 5 karakter eklenecek ve 11+5=16 olması beklenecektir.



Şekil.7.Karakter Sayısı 16

Bu şekilde uzunluk değişkeninden gelen her değeri sayı değişkenine ekleyerek yazmış oluyor.

**TRIGGER3:**

Oluşturulan bu trigger ile vb\_kullanici tablosunda before ifadesiyle güncelleme işlemi yapılmıştır. Vb\_kullanici tablosundaki 5.id yi girilen değerler ile veriler değiştirilmiş, güncellenmiştir.

CREATE OR REPLACE FUNCTION "kullanici\_kayit"()

RETURNS TRIGGER

AS

$$

BEGIN

NEW."adi\_soyadi" = UPPER(NEW."adi\_soyadi"); -- büyük harfe dönüştürdükten sonra ekle

NEW."email" = LTRIM(NEW."email"); -- Önceki ve sonraki boşlukları temizle

IF NEW."adres" IS NULL THEN

RAISE EXCEPTION 'Adres alanı boş olamaz';

END IF;

RETURN NEW;

END;

$$

LANGUAGE "plpgsql";

CREATE TRIGGER "kayitKontrol"

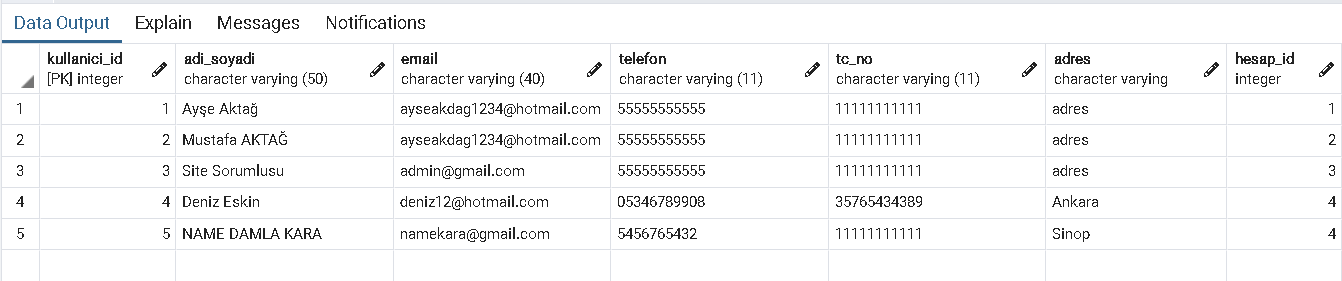
BEFORE INSERT OR UPDATE ON "vb\_kullanici" -- veriyi eklemeden/değiştirmeden önce üzerinde işlem yap

FOR EACH ROW

EXECUTE PROCEDURE "kullanici\_kayit"();

INSERT INTO "vb\_kullanici" ( "kullanici\_id", "adi\_soyadi", "email","telefon","tc\_no","adres","hesap\_id")

VALUES ( '5', 'Name Damla Kara', ' namekara@gmail.com ','5456765432','11111111111','Sinop', 4);



Şekil.8.Trigger 3 Update

Şekil.8.vb\_kullanici Tablosundaki Bilgilerin Değiştiğini Gösteren Tablo

**TRIGGER4:**

create or replace function toplamkargohesap()

returns trigger

as

$$

begin

update toplamkargo set sayi=sayi+1;

return new;

end;

$$

language plpgsql;

--trigger oluşturma

create trigger kargotrigger

after insert

on vb\_kargo

for each row

execute procedure toplamkargohesap();

//trigger,vb\_kargo tablosunda insert işlemi yapıldıktan sonra çalışacak.Her satır için vb\_kargo tablosuna ekleme yapıldıktan sonra toplamkargohesap () prosedürünü çalıştıracak.

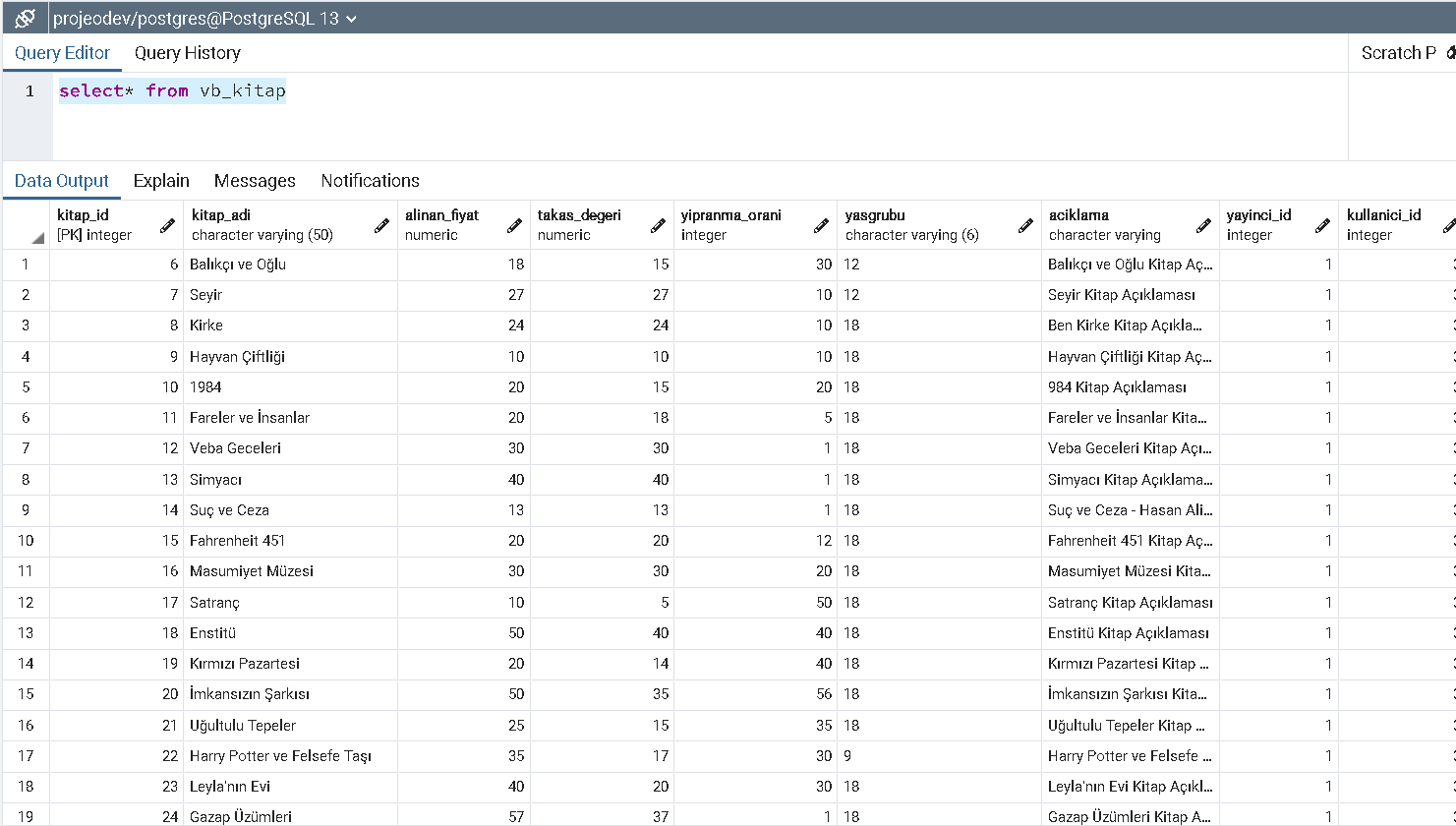
insert into vb\_kargo( kargo\_id ,kargo,ucret) values(5, 'Trendyol Kargo',10);

Yukarıda yazılan komut ile vb\_kargo tablosuna ekleme yaparak toplamkargo tablosunun da +1 artması istenmiştir.

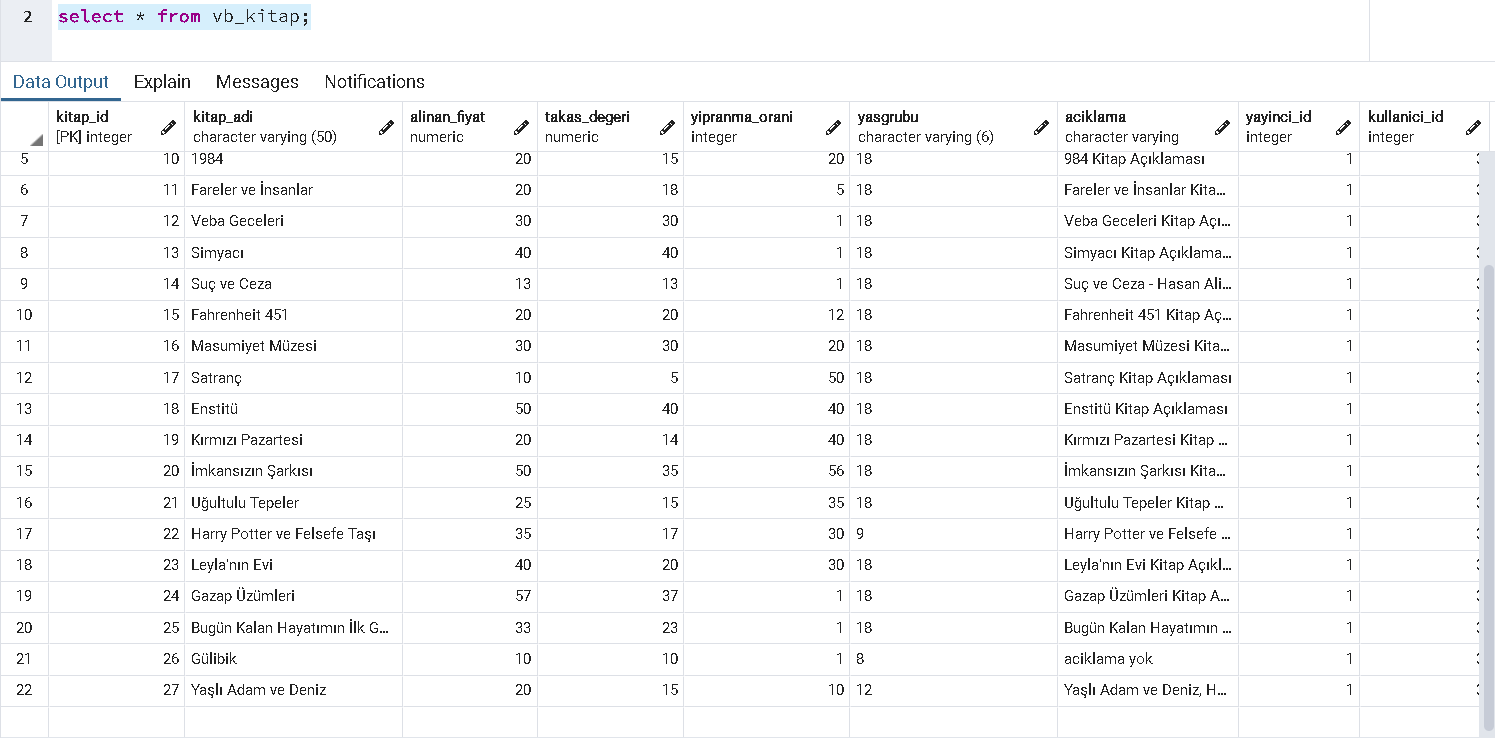
Select \* from toplamkargo yazarak değerin 1 arttığı görülmüştür. Yazılan trigger başarılı bir şekilde çalışmıştır.

**ARAMA İŞLEMİ:**

* Select Komutu veritabanından veri almak (arama/listeleme) için kullanılır.
* select\* from vb\_kitap;

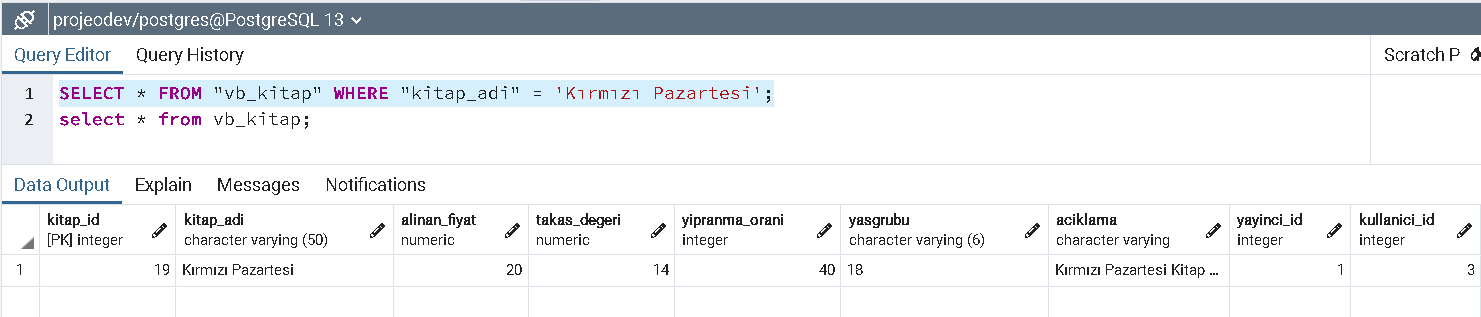


Şekil.9.Arama İşlemi Çıktısı1



Şekil.10.Arama İşlemi Çıktısı 2

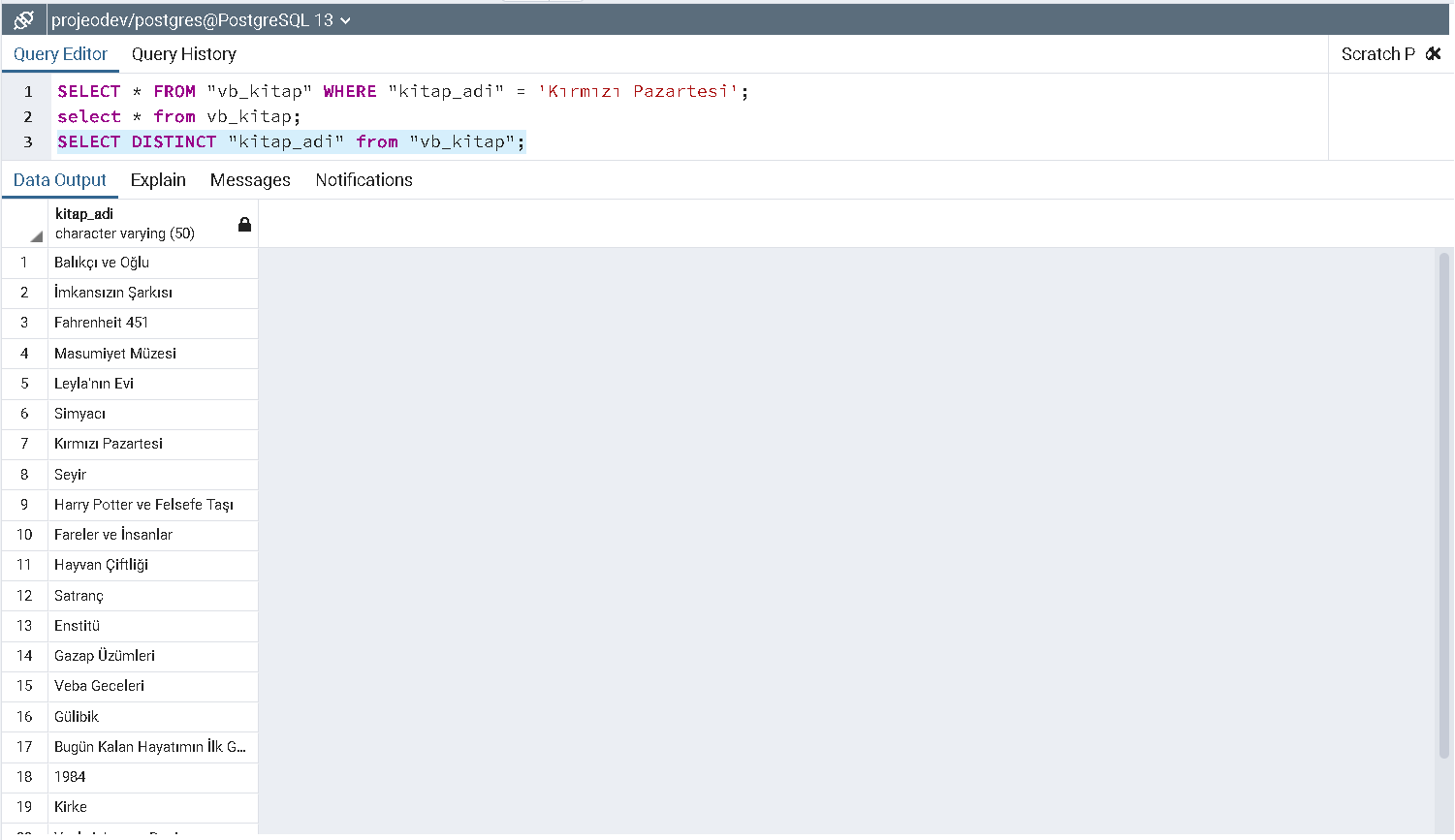
SELECT \* FROM "vb\_kitap" WHERE "kitap\_adi" = 'Kırmızı Pazartesi';



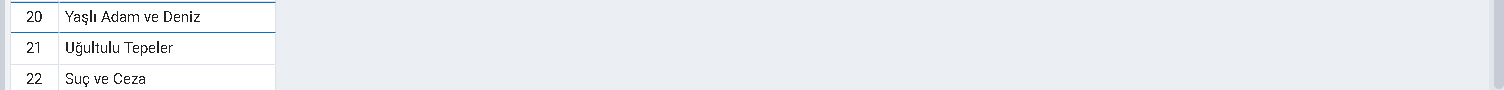
Şekil.11. Detaylı Arama İşlemi Çıktısı

* Discint ,sorgu sonucunda yer alan tekrarlı kayıtların (satırların), tek kayıt olarak getirilmesini sağlar.

SELECT DISTINCT "kitap\_adi" from "vb\_kitap";

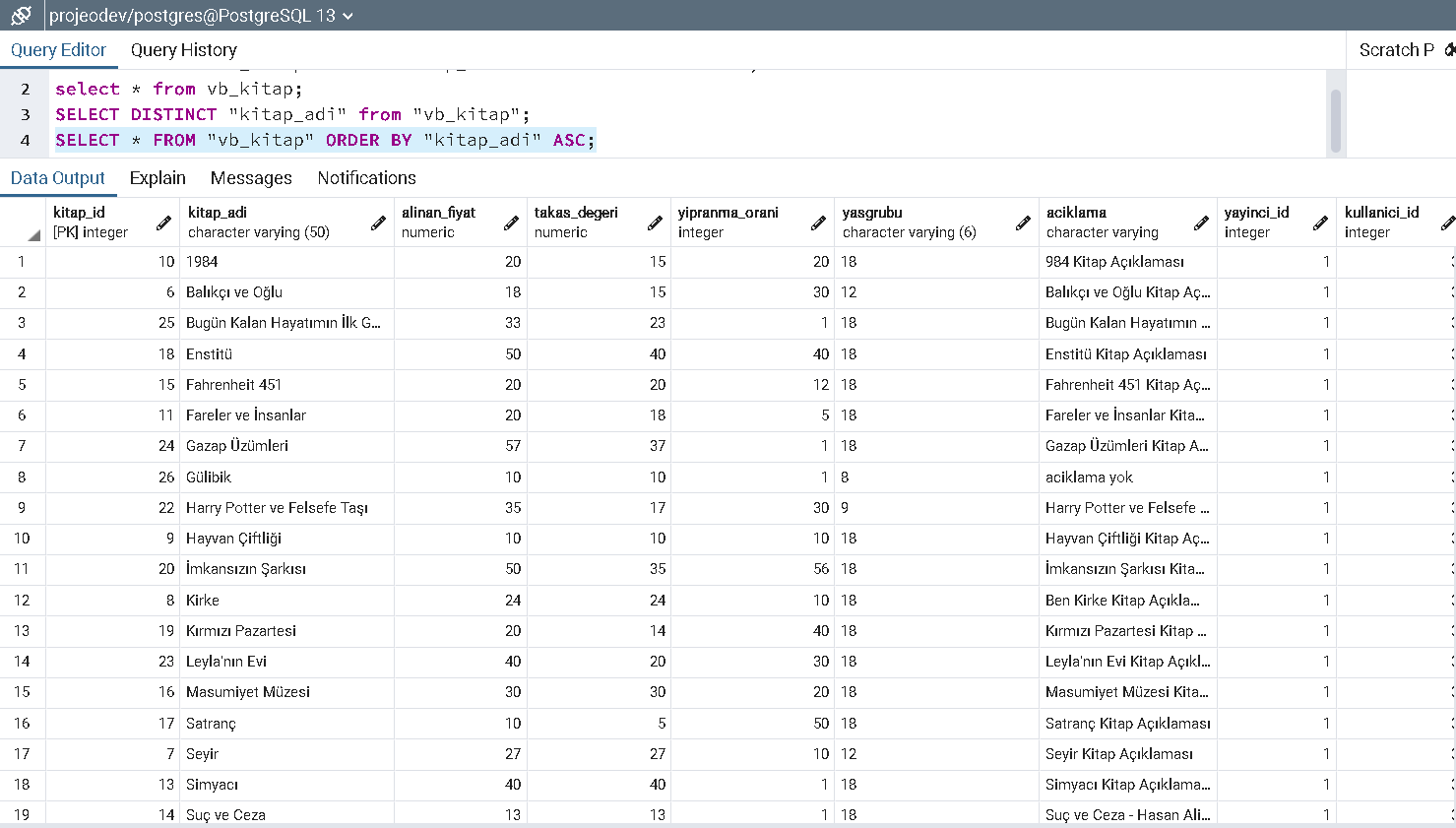


Şekil.12. Kitap Adı Sütunu

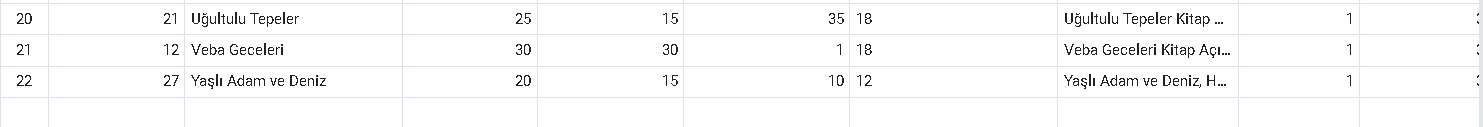


Şekil.13. Kitap Adı Sütunu Devamı

* Order by, sorgular sonucunda listelenen kayıtların belirli alanlara göre alfabetik veya sayısal olarak artan ya da azalan şeklinde sıralanması için "ORDER BY" komutu kullanılır.
* SELECT \* FROM "vb\_kitap" ORDER BY "kitap\_adi" ASC;



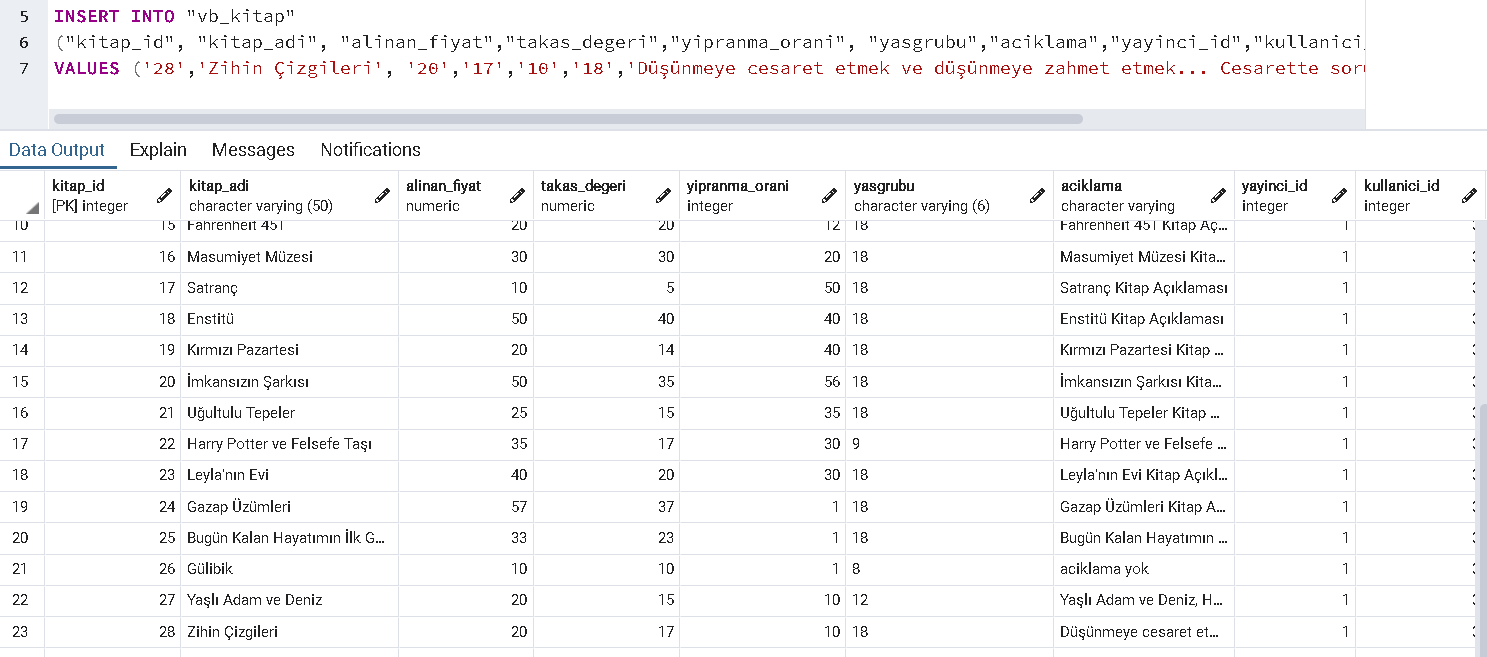
Şekil.14. Alfabetik Sırayla Arama



Şekil.15.Alfabetik Sırayla Arama Devamı

**EKLEME:**

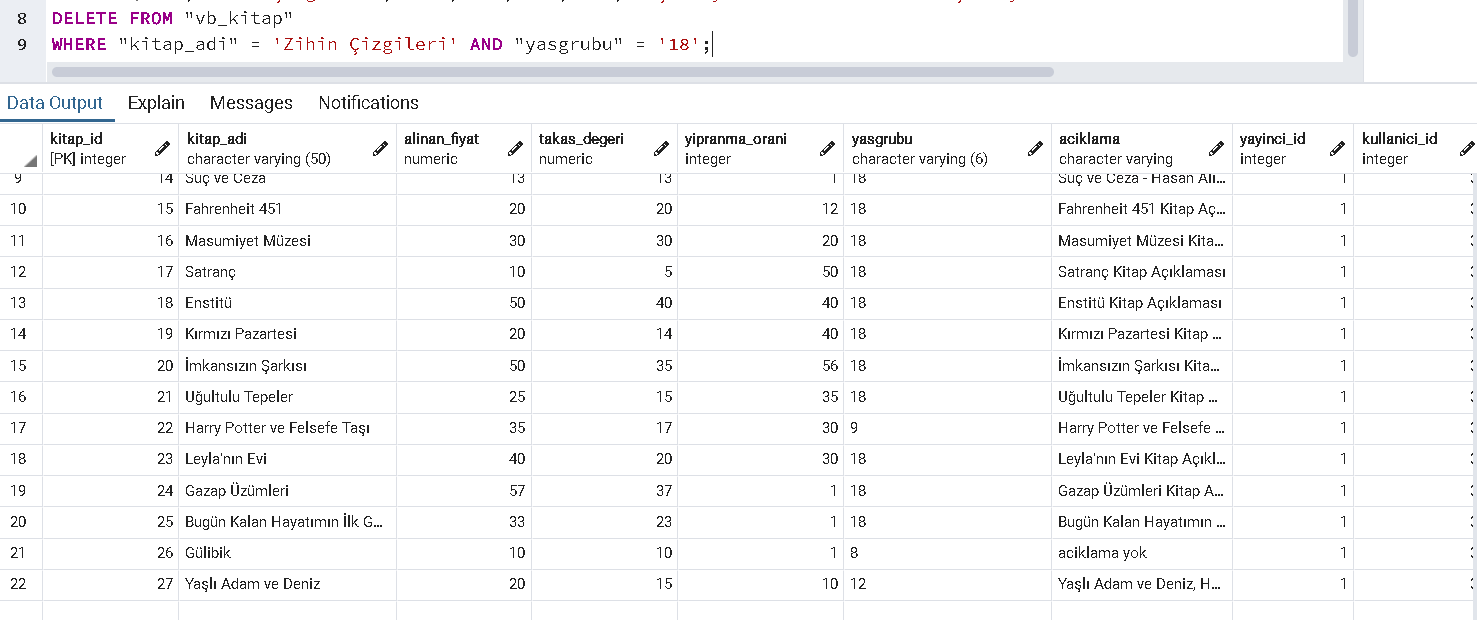
* INSERT komutu tabloya yeni kayıt eklemek için kullanılır.
* Ekleme işlemlerinde veri bütünlüğü kısıtları göz önüne alınır.
* Yalnızca bazı sütunlara veri eklememiz mümkündür.
* Veri eklenmeyen sütunlar NULL (boş) gözükecektir.
* INSERT INTO "vb\_kitap"
* ("kitap\_id", "kitap\_adi", "alinan\_fiyat","takas\_degeri","yipranma\_orani", "yasgrubu","aciklama","yayinci\_id","kullanici\_id")
* VALUES ('28','Zihin Çizgileri', '20','17','10','18','Düşünmeye cesaret etmek ve düşünmeye zahmet etmek... Cesarette sorun yok, sorun zahmette!','1','3');



Şekil.16.Ekleme

**SİLME:**

* DELETE ifadesi tablodaki kayıt veya kayıtların silinmesini sağlar.
* Silme işlemlerinde veri bütünlüğü kısıtları göz önüne alınır.
* DELETE FROM "vb\_kitap"
* WHERE "kitap\_adi" = 'Zihin Çizgileri' AND "yasgrubu" = '18';



Şekil.17.Silme

**GÜNCELLEME:**

* UPDATE komutu tablodaki kayıt(lar)ın değiştirilmesini sağlar.
* Güncelleme işlemlerinde veri bütünlüğü kısıtları göz önüne alınır.
* UPDATE "vb\_kitap" SET "alinan\_fiyat"= '15', "yasgrubu"= '18'
* WHERE "kitap\_adi" = 'Gülibik';
* Güncellemeden önce:
* 
* Güncellemeden sonra:
* 

**UYGULAMANIN KAYNAK KODLARI:**

https://github.com/ayseaktag/BSM211-AYSE-AKTAG-S200110077.git

**VİDEO ADRESİ:**

https://youtu.be/9O9h8pjm684