Dersin Kodu: BIMU3064

Dersin Adı: Veri Tabanı Yönetim Sistemleri

Öğrenci Numaram: 1306170114

Adım: Enver

Soyadım: Usta

Ödev Numarası: Ödev4

1. Fis tablosundaki toplam alanını gerektiğinde güncelleyen trigger(lar)ı yazınız. İpucu: fisUrunleri tablosuna bir kayıt eklendiğinde, silindiğinde veya bir kayıttaki (barkod veya miktar veya birimFiyat alanları) guncellendiğinde çalışacak olan bu trigger fiş toplamı değerini fiş ürünlerindeki “miktar x birimFiyat” değerlerini toplayarak bulur.

CREATE OR REPLACE FUNCTION func\_fis\_toplam\_guncelle()

RETURNS TRIGGER AS $fis\_toplam\_guncelle$

BEGIN

IF(TG\_OP = 'DELETE')    THEN

        update fis set toplam=toplam-(NEW.miktar \* NEW.birimfiyat) WHERE OLD.fisno=fis.fisno;

        RETURN OLD;

    ELSIF (TG\_OP = 'INSERT') THEN

        update fis set toplam=toplam+(NEW.miktar \* NEW.birimfiyat) WHERE NEW.fisno=fis.fisno;

        RETURN NEW;

    ELSIF(TG\_OP = 'UPDATE') THEN

        update fis set toplam=toplam-(OLD.miktar \* OLD.birimfiyat)+(NEW.miktar \* NEW.birimfiyat) WHERE NEW.fisno=fis.fisno;

        RETURN NEW;

    END IF;

END;

$fis\_toplam\_guncelle$

LANGUAGE plpgsql;

CREATE TRIGGER trigger\_fis\_toplam\_guncelle

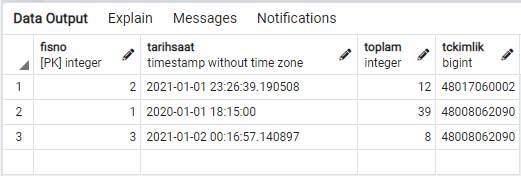
AFTER INSERT OR UPDATE OR DELETE

ON fisurunleri

FOR EACH ROW

EXECUTE PROCEDURE func\_fis\_toplam\_guncelle();

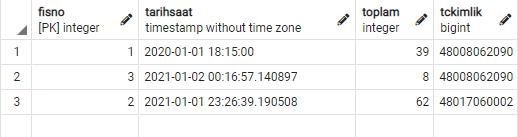
Fis tablosundaki ilk görünümüm:





Tablomuzdaki Değişimler:

INSERT INTO fisurunleri VALUES(2, 124, 5, 'kg', 10);

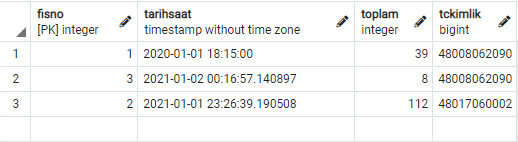




--Az önceki eklediğim ürünün 5kg olan miktarını 10kg olarak güncelleyeceğim

--Böylece 2 numaralı fişteki 62tl’nin tekrar değişmesi gerekmekte.

UPDATE fisurunleri SET miktar=10 WHERE fisno=2 and barkod=124;

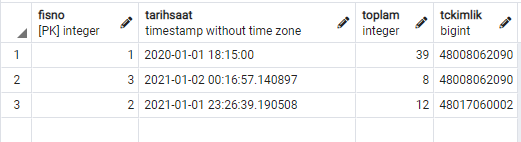




--Şimdi ise az önce ekleyip hem de güncellediğim 124 barkod numaralı ürünü

--sileceğim. En baştaki görünüme geri gelmesi gerekiyor.

DELETE FROM fisurunleri WHERE barkod=124 AND fisno=2;





2. Urun tablosundaki stokMiktarı alaninı gerektiğinde güncelleyen trigger(lar)ı yazınız. İpucu: stokMiktarı bir ürünün tüm alımlardaki(fatura) miktarlar toplamından tüm satişlardaki miktarlar toplamının çıkarılmasıyla hesaplanabilir. Bu hesaplamadan birim alanı kullanılmaz.

Fiş Ürünleri için yazdığım trigger:

CREATE OR REPLACE FUNCTION func\_fis\_urun\_stok\_guncelle()

RETURNS TRIGGER AS $fis\_urun\_stok\_guncelle$

    BEGIN

        IF(TG\_OP = 'INSERT') THEN

            UPDATE urun SET stokmiktari = (stokmiktari - NEW.miktar)

WHERE  barkod=NEW.barkod;

            RETURN NEW;

        ELSIF(TG\_OP = 'DELETE') THEN

            UPDATE urun SET stokmiktari = (stokmiktari + OLD.miktar)

WHERE barkod=OLD.barkod;

            RETURN OLD;

        ELSIF(TG\_OP = 'UPDATE') THEN

            UPDATE urun SET stokmiktari =

(stokmiktari + OLD.miktar - NEW.miktar)

WHERE barkod=OLD.barkod;

            RETURN NEW;

        END IF;

    END;

$fis\_urun\_stok\_guncelle$ LANGUAGE plpgsql;

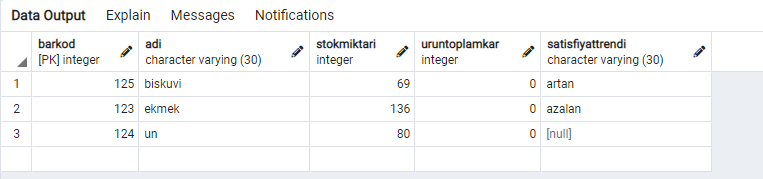
CREATE TRIGGER  trigger\_fis\_urun\_stok\_guncelle

AFTER INSERT OR UPDATE OR DELETE

ON fisurunleri

FOR EACH ROW EXECUTE PROCEDURE func\_fis\_urun\_stok\_guncelle();

Ürün tablomdaki ilk görünüm:



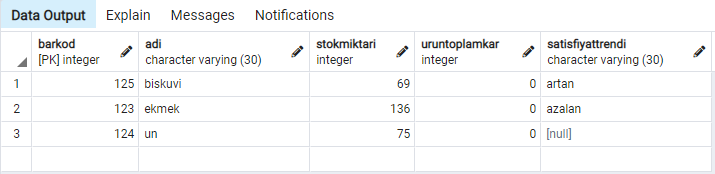


Tablomuzdaki Değişimler:

--5kg miktarındaki un’u fisurunlerine ekliyorum.

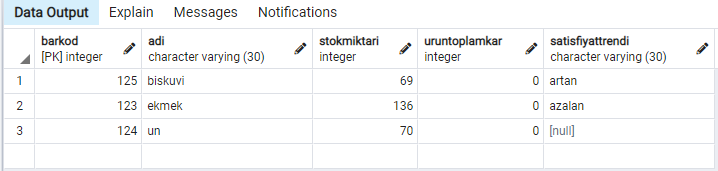
--stok miktarının 5kg azalması bekleniyor.

INSERT INTO fisurunleri VALUES(2, 124, 5, 'kg', 10);



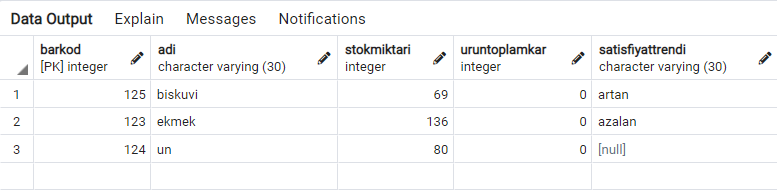


UPDATE fisurunleri SET miktar=10 WHERE fisno=2 and barkod=124;





DELETE FROM fisurunleri WHERE barkod=124 AND fisno=2;





Fatura Ürünleri için yazdığım trigger:

CREATE OR REPLACE FUNCTION func\_fatura\_urun\_stok\_guncelle()

RETURNS TRIGGER AS $fatura\_urun\_stok\_guncelle$

    BEGIN

        IF(TG\_OP = 'INSERT') THEN

            UPDATE urun SET stokmiktari = (stokmiktari + NEW.miktar)

            WHERE  barkod=NEW.barkod;

            RETURN NEW;

        ELSIF(TG\_OP = 'DELETE') THEN

            UPDATE urun SET stokmiktari = (stokmiktari - OLD.miktar)

            WHERE barkod=OLD.barkod;

            RETURN OLD;

        ELSIF(TG\_OP = 'UPDATE') THEN

            UPDATE urun SET stokmiktari = (stokmiktari - OLD.miktar + NEW.miktar)

            WHERE barkod=OLD.barkod;

            RETURN NEW;

        END IF;

    END;

$fatura\_urun\_stok\_guncelle$ LANGUAGE plpgsql;

CREATE TRIGGER  trigger\_fatura\_urun\_stok\_guncelle

AFTER INSERT OR UPDATE OR DELETE

ON faturaurunleri

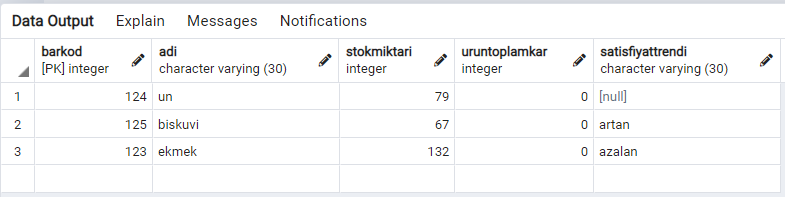
FOR EACH ROW EXECUTE PROCEDURE func\_fatura\_urun\_stok\_guncelle();

-- Faturaurunlerindeki değişimlerde bizim stok miktarımızı etkileyecektir.

-- 2 numarali faturamiza biraz un ekleyelim ve stoktaki değişimi gözlemleyelim.

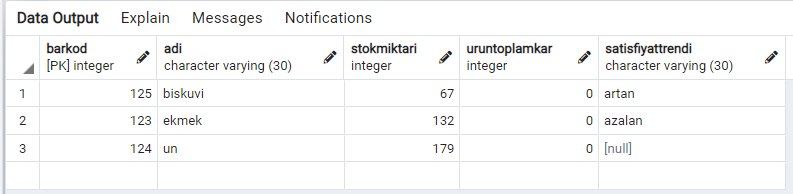
NOT: Aralarda denemeler yaptığım için tablolarımda sorular arasında geçiş yaparken de değişimler gözlenebilir, lütfen benim önceki şeklinde gösterdiğim tabloyu dikkate alınız.

Ürün Tablom:





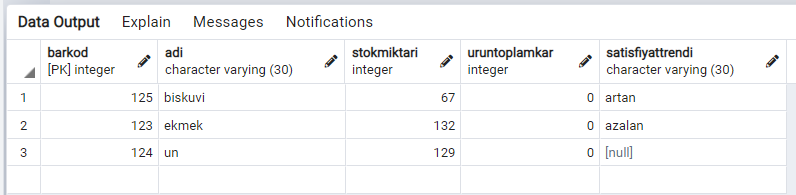
INSERT INTO faturaurunleri VALUES(2, 124, 100, 'kg', 1);





--100kg un fazla gelmiş gibi duruyor, bunu 50kg olarak güncelleyelim.

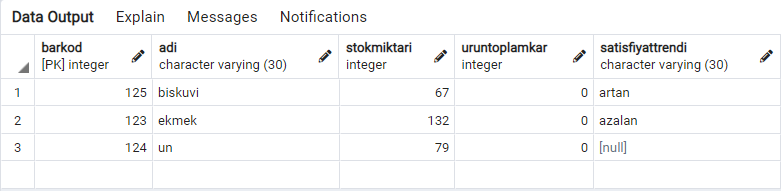
UPDATE faturaurunleri SET miktar=50 WHERE faturano=2 AND barkod=124;





--faturaurunlerinden kaldırıyoruz. Birdaha kg’ını 1tl’den un nasıl buluruz hiç bilmiyorum :(

DELETE FROM faturaurunleri WHERE faturano=2 AND barkod=124;



3. Urun tablosundaki satisFiyatTrendi alaninı gerektiğinde güncelleyen trigger(lar)ı yazınız. İpucu: satisFiyatTrendi alanı “artan” veya “azalan” olabilir. Bu değer ürünün tarihSaat’e göre son iki satışındaki (fisUrunlari) birimFiyat değerlerinin karşılaştırılmasıyla bulunur: Son satıştaki fiyat bir öncesinden büyükse trend “artan”, değilse “azalan” olacaktır.

Önemli Not!

Şayet kendisinden önce eklenen bir değer yoksa satisfiyattrendini NULL olarak eklemektedir.

CREATE OR REPLACE FUNCTION func\_urun\_satis\_fiyat\_trendi\_guncelle()

RETURNS TRIGGER AS $urun\_satis\_fiyat\_trendi\_guncelle$

    DECLARE

        prevurunbirimfiyat integer;

    BEGIN

        IF(TG\_OP = 'INSERT') THEN

            prevurunbirimfiyat = (SELECT birimfiyat FROM fisurunleri

                                 WHERE NEW.barkod=barkod AND fisno=(

                                                      SELECT fisno FROM fis

                                                      ORDER BY tarihsaat DESC

                                                       LIMIT 1 OFFSET 1

                                                        )

);

            IF prevurunbirimfiyat is NULL THEN

                UPDATE urun SET satisfiyattrendi=NULL

WHERE barkod=NEW.barkod;

            ELSIF(prevurunbirimfiyat < NEW.birimfiyat) THEN

                UPDATE urun SET satisfiyattrendi='artan'

WHERE barkod=NEW.barkod;

            ELSIF(prevurunbirimfiyat > NEW.birimfiyat) THEN

                UPDATE urun SET satisfiyattrendi='azalan'

WHERE barkod=NEW.barkod;

            END IF;

            RETURN NEW;

        END IF;

    END;

$urun\_satis\_fiyat\_trendi\_guncelle$ LANGUAGE plpgsql;

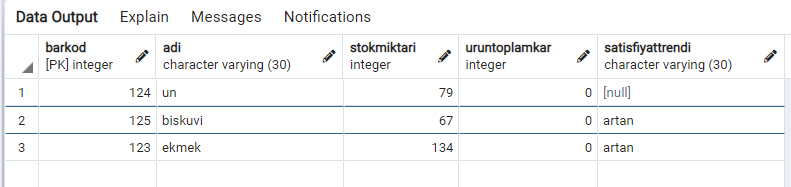
CREATE TRIGGER trigger\_urun\_satis\_fiyat\_trendi\_guncelle

AFTER INSERT

ON fisurunleri

FOR EACH ROW EXECUTE PROCEDURE func\_urun\_satis\_fiyat\_trendi\_guncelle();

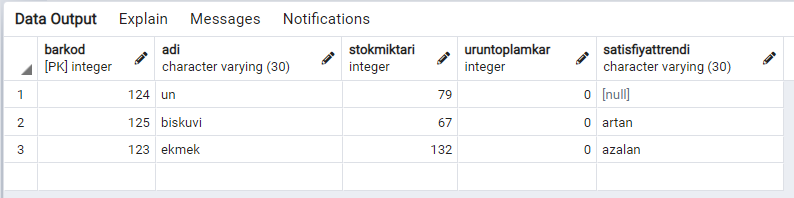
Ürün tablomdaki ilk görünüm:





-- Fisnosu 5 olan yeni bir fiş oluşturalım (4 numaralı fişte 10tl gibi fahiş bir fiyattan satılan ekmeği 5 numaralı fişte insancıl değer olan 1tl’den satalım. Fırsatçılara yer yok!!!)

INSERT INTO fis VALUES(5, CURRENT\_TIMESTAMP, 0, 48008062090);

INSERT INTO fisurunleri VALUES(5, 123, 2, 'adet', 1);



4. Fatura tablosundaki toplan alanını gerektiğinde güncelleyen trigger(lar)ı yazınız. İpucu: faturaUrunleri tablosuna bir kayıt eklendiğinde, silindiğinde veya bir kayıttaki (barkod veya miktar veya birimFiyat alanları) guncellendiğinde çalışacak olan bu trigger toplam değerini fatura ürünlerindeki “miktar x birimFiyat” değerlerini toplayarak bulur.

CREATE OR REPLACE FUNCTION func\_fatura\_toplam\_guncelle()

RETURNS TRIGGER AS $fatura\_toplam\_guncelle$

BEGIN

    IF(TG\_OP = 'DELETE')    THEN

        update fatura set toplam=toplam-(OLD.miktar \* OLD.birimfiyat) WHERE OLD.faturano=fatura.faturano;

        RETURN OLD;

    ELSIF (TG\_OP = 'INSERT') THEN

        update fatura set toplam=toplam+(NEW.miktar \* NEW.birimfiyat) WHERE NEW.faturano=fatura.faturano;

        RETURN NEW;

    ELSIF(TG\_OP = 'UPDATE') THEN

        update fatura set toplam=toplam-(OLD.miktar \* OLD.birimfiyat)+(NEW.miktar \* NEW.birimfiyat) WHERE NEW.faturano=fatura.faturano;

        RETURN NEW;

    END IF;

END;

$fatura\_toplam\_guncelle$

LANGUAGE plpgsql;

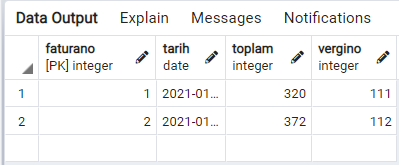
CREATE TRIGGER trigger\_fatura\_toplam\_guncelle

AFTER INSERT OR UPDATE OR DELETE

ON faturaurunleri

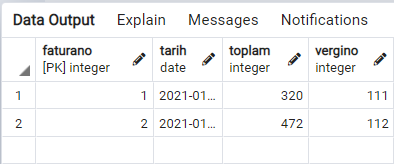
FOR EACH ROW

EXECUTE PROCEDURE func\_fatura\_toplam\_guncelle();

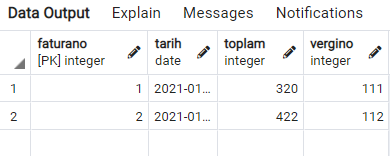
Fatura tablosundaki ilk görünümüm:

Tablomuzdaki Değişimler:

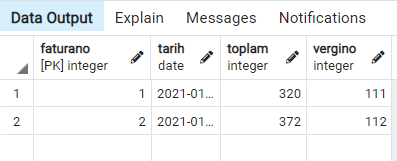
INSERT INTO faturaurunleri VALUES(2, 124, 100, 'kg', 1);



UPDATE faturaurunleri SET miktar=50 WHERE faturano=2 AND barkod=124;



DELETE FROM faturaurunleri WHERE faturano=2 AND barkod=124;



5. Tedarikçi tablosundaki faturaSayisi, faturaToplamı, faturaOrtalaması alanlarını gerektiğinde güncelleyen trigger(lar)ı yazınız. İpucu: fatura tablosuna kayıt eklenmesi, silinmesi veya bu tablodaki toplam değerinin güncellenmesi durumunda çalışacak bu trigger, soru 4’teki trigger’I dolaylı olarak kullanmalıdır. Çünkü faturaUrunleri tablosuna bir kayıt eklenmesi, silinmesi veya güncelleme durumunda tedarikci tablosundaki FaturaToplami ve faturaOrtalaması değerlerinin yeniden hesaplanması gerekecektir.

CREATE OR REPLACE FUNCTION func\_tedarikci\_guncelle()

RETURNS TRIGGER AS $tedarikci\_guncelle$

BEGIN

    IF (TG\_OP = 'INSERT') THEN

        UPDATE tedarikci

        SET faturasayisi= faturasayisi + 1, faturatoplami= faturatoplami+NEW.toplam, faturaortalamasi=( (faturatoplami+NEW.toplam)/(faturasayisi+1) )

        WHERE NEW.vergino=tedarikci.vergino;

        RETURN NEW;

    ELSIF(TG\_OP = 'DELETE') THEN

        UPDATE tedarikci

        SET faturasayisi= faturasayisi - 1, faturatoplami= faturatoplami-OLD.toplam, faturaortalamasi=( (faturatoplami-OLD.toplam)/(faturasayisi-1) )

        WHERE NEW.vergino=tedarikci.vergino;

        RETURN OLD;

    ELSIF(TG\_OP = 'UPDATE') THEN

        UPDATE tedarikci

        SET faturatoplami= faturatoplami-OLD.toplam+NEW.toplam, faturaortalamasi=( (faturatoplami-faturatoplami-OLD.toplam+NEW.toplam)/(faturasayisi) )

        WHERE NEW.vergino=tedarikci.vergino;

        RETURN NEW;

    END IF;

END;

$tedarikci\_guncelle$

LANGUAGE plpgsql;

CREATE TRIGGER trigger\_tedarikci\_guncelle

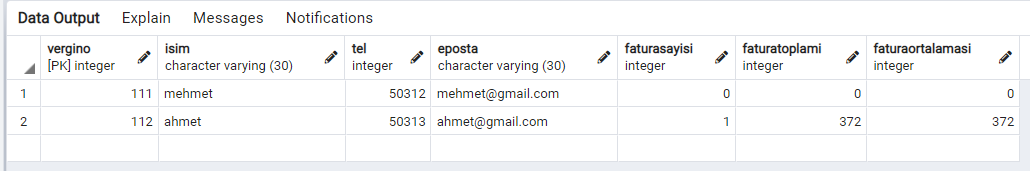
AFTER INSERT OR UPDATE OR DELETE

ON fatura

FOR EACH ROW

EXECUTE PROCEDURE func\_tedarikci\_guncelle();

Tedarikci tablosundaki ilk görünümüm

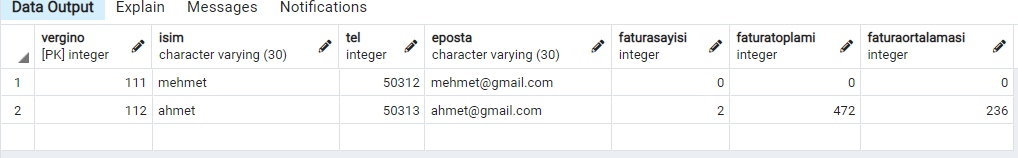




Tablomuzdaki Değişimler:

--Yeni bir fatura ekleyelim 112 nolu Ahmet tedarikcisine(toplam değerini elle 100 giriyorum 0 yerine).

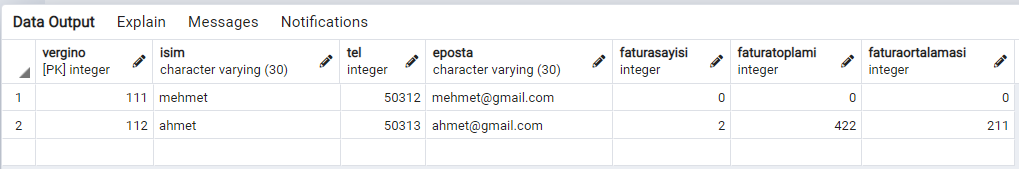
INSERT INTO fatura VALUES(3, CURRENT\_TIMESTAMP, 100, 112);





--Az önceki 100 olan toplam değerini 50 yapalim.

UPDATE fatura SET toplam=50 WHERE faturano=3;

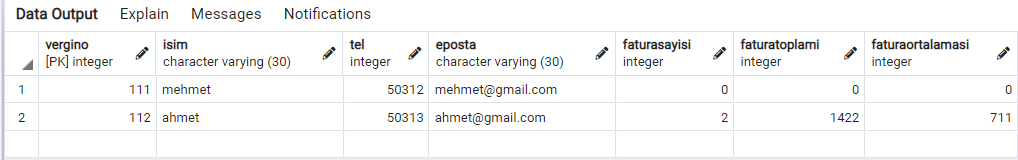




--Bu yeni oluşturduğumuz faturaya(faturanosu 3 olan) yeni bir faturaurunu ekleyelim ve bakalim önceki triggerimiz bu triggerimizi tetikliyor mu.

--1000adet ekmek ekliyoruz yeni faturamiza.

INSERT INTO faturaurunleri VALUES(3, 123, 1000, 'adet', 1);

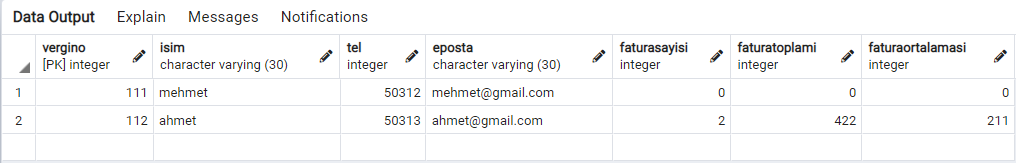




--Eklediğimiz 1000 ekmeği kaldıralım.

--Unutma! Faturayı tamamen silmiyoruz sadece içindeki faturaurununu kaldırıyoruz.

DELETE FROM faturaurunleri WHERE faturano=3 AND barkod=123;





6. “… satislar(barkod, tarihSaat) returns table…” gibi tanımlanan barkodu verilen bir ürünün verilen bir tarihSaatten sonraki tüm satış kayıtlarını (fisUrunlari tablosundaki kayıtları) döndüren stored function’I yazınız ve bu fonksiyonu çalıştığını gösterebilmek için bir SQL komutunda kullanınız. İpucu: “… satislar(barkod, tarihSaat) returns table…”

CREATE OR REPLACE FUNCTION satislar(barkodumuz integer, tarihsaatimiz timestamp)

RETURNS TABLE (

fisno integer,

barkod integer,

miktar integer,

birim varchar(10),

birimfiyat integer

) AS $$

BEGIN

RETURN QUERY SELECT fisurunleri.\*

FROM fisurunleri, fis

WHERE fisurunleri.fisno=fis.fisno AND

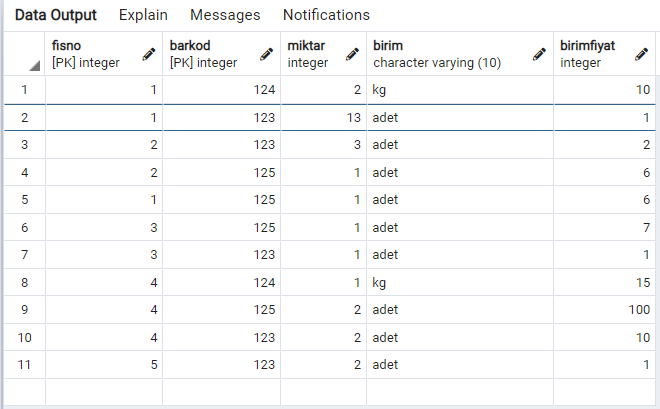
fisurunleri.barkod=barkodumuz AND

fis.tarihsaat > tarihsaatimiz;

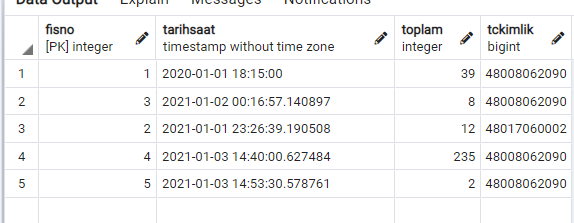
END;

$$ LANGUAGE plpgsql;

Fisurunleri tablosundaki görünümüm:

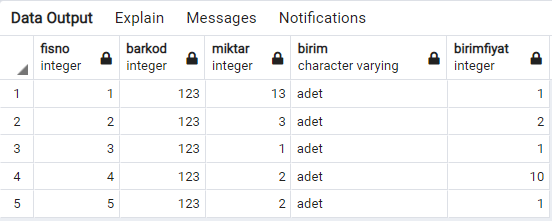


Fis tablosundaki görünümüm:



Sonuçlar:

--1999’dan sonraki ekmekleri getirelim. SELECT \* FROM satislar(123, ‘1999-01-08 04:05:06');



-- 2021-01-02 00:17:00’dan sonraki ekmekler. SELECT \* FROM satislar(123, '2021-01-02 00:17:00');

