

Giriş

Parti Dünyası - Organizasyon Planlama & Satın Alma Sistemi isimli projemiz, organizasyonların bir veritabanında tutulmasını kolaylaştırmak ve özelleştirmek amacıyla tasarlanmış kapsamlı bir veritabanı projesidir. Bu projenin temel odak noktası, kullanıcıların sistem üzerinden **kayıt olmalarını, giriş yapmalarını** ve seçtikleri organizasyon türüne uygun firmalardan **teklif alabilmelerini** sağlamaktır. Veritabanımız, kullanıcıların girdiği bilgileri ve tercihlerini depolayarak, organizasyon sürecini adım adım yönetmelerine olanak tanımaktadır.

Organizasyon türlerine özgü olarak listelenen firmalar, veritabanında kategori bazlı olarak sınıflandırılmıştır. Kullanıcılar, kişi sayısı, zaman dilimi gibi kriterlere göre filtreleme yaparak, ihtiyaçlarına en uygun firmalardan teklif alabilirler. Bu teklifler, veritabanında kullanıcı sayfasında detaylı bir şekilde listelenir ve güncellenir. Aynı zamanda, şirketlerin teklif verme sayıları da veritabanında tutularak kullanıcılar için anlık ve güncel bilgiler sunulur.

Projemizin ek özelliklerinden biri olan alışveriş modülü, kullanıcıların ürünleri veritabanımız üzerinden satın almalarına imkan sağlar. Bu özellik, kullanıcıların organizasyonlarına dair ihtiyaçları doğrultusunda ürün seçeneklerini keşfetmelerine ve kolayca satın almalarına olanak tanır

.

Veritabanımız, kullanıcıların organizasyon planlama ve satın alma süreçlerini etkili bir şekilde yönetmelerini sağlamak üzere tasarlanmış olup, güçlü bir altyapıya sahiptir. Bu sayede, kullanıcılarımızın deneyimini iyileştirerek, özel günlerini daha unutulmaz kılmak için gereken veri tabanı desteğini sağlamaktadır.

Proje Planı

Proje yapılmadan önce bir plan dahilinde gitmenin faydalı olacağını düşündü ve projem yapılırken şu sırayla ilerlendi.

E-R Diyagram Tasarımı:

Bu projeye başlamadan önce elde bir şemanın bulunması gerektiği düşünüldü ve hiç kod yazmadan hangi varlıkların hangi ilişkilerin olması gerektiği kararlaştırıldı.

SQL Kod İmplementasyonu (Tablo, Fonksiyonlar, Tetikleyiciler)

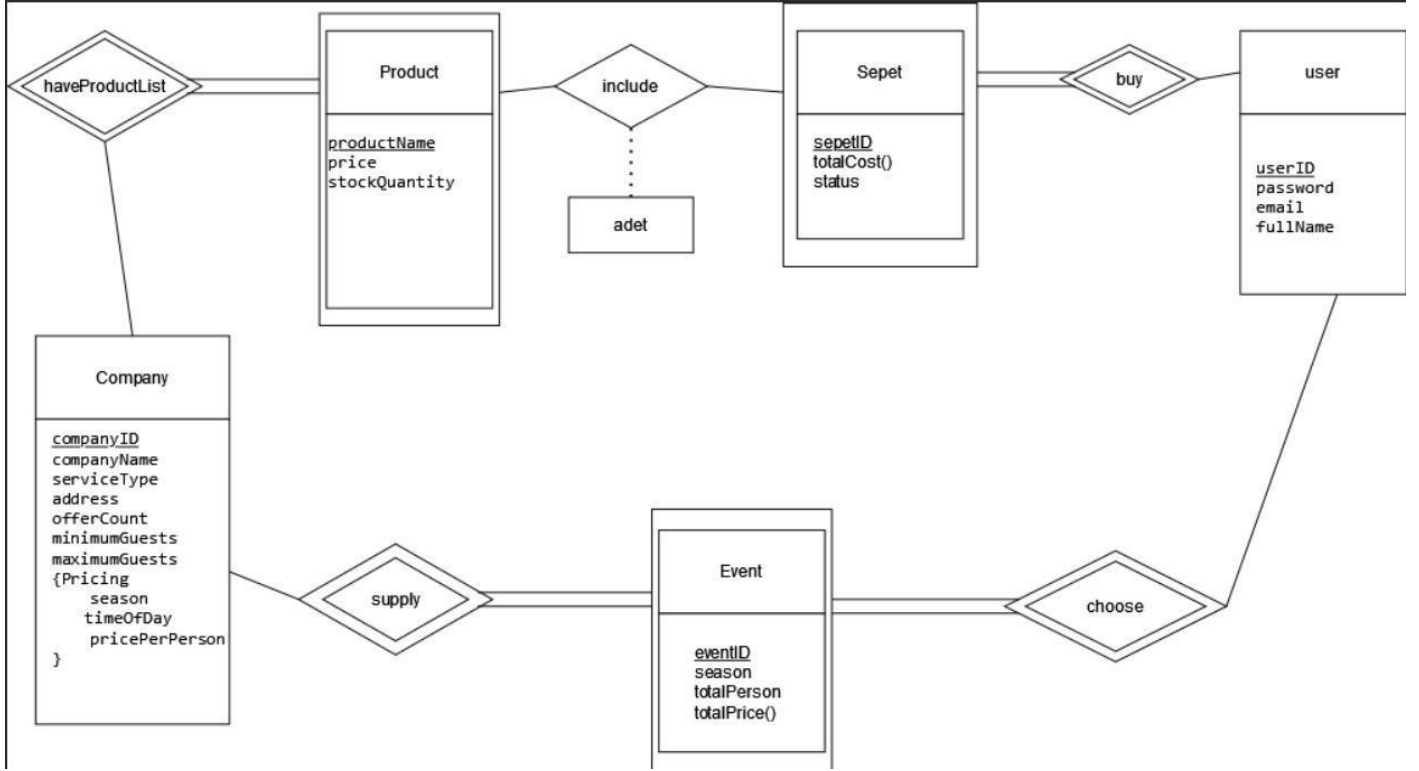
Şema göz önünde bulundurularak tablolar oldukça hızlı bir şekilde oluşturuldu. Tablo eklendikten sonra gerekli fonksiyonlar, tetikleyiciler ve kısıtlar oluşturuldu. Bu aşamada fonksiyonlar ve kısıtlar eklenirken tabloda mantık hataları oluştu ve bu hatalar sonucu E-R şemasında da değişikliğe gidildi.

Arayüz Tasarımı

İşin büyük bölümü olan arkayüz tasarımı bitti son olarak hangi programlama dilinde arayüz yazılacağı kararlaştırıldı ve SQL kodları yazılan programla bağlandı. Program üzerinden SQL fonksiyonları çağrılarak SQL tablolarına ekle, sil, görüntüle gibi işlemler yapılabildi

E-R Diyagramı

Şema:



İlişkiler:

1. haveProductList (Ürün Listesine Sahip):

1. Şirketin birden çok ürün listesine sahip olabileceğini belirtir.

2. include (İçerir):

1. Sepetin, birden fazla ürün içerebileceğini gösterir. "adet" etiketi, bir sepete eklenen ürün sayısını temsil edebilir.

3. buy (Satın Al):

1. Kullanıcının bir sepeti satın alabileceğini ifade eder.

4. supply (Tedarik Et):

1. Şirket kullanıcının istediği organizasyonu tedarik edebilir.

5. choose (Seç):

1. Kullanıcıların bir etkinlik seçebileceğini ifade eder.

E-R Diyagramı

Varlıklar:

1.Product (Ürün):

1. productName (Ürün Adı)
2. price (Fiyat)
3. stockQuantity (Stok Miktarı)

2.Sepet (Basket):

1. sepetID (Sepet ID)
2. totalCost ;Türetilmiş bir öznitelik
3. status (Durum)

3.User (Kullanıcı):

1. userID (Kullanıcı ID)
2. password (Şifre)
3. email (E-Posta)
4. fullName (Tam Adı)

4.Company (Şirket):

1. companyID (Şirket ID)
2. companyName (Şirket Adı)
3. serviceType (Hizmet Tipi)
4. address (Adres)
5. offerCount (Teklif Sayısı)
6. minimumGuests (Minimum Misafir)
7. maximumGuests (Maksimum Misafir)
8. {Pricing (Fiyatlandırma)}:
 1. season (Sezon)
 2. timeOfDay (Günün Zamanı)
 3. pricePerPerson (Kişi Başına Fiyat)

5.Event (Etkinlik):

1. eventID (Etkinlik ID)
2. season (Sezon)
3. totalPerson (Toplam Kişi Sayısı)
4. totalPrice() (Toplam Fiyat)

Tablolar

Kullanıcı Tablosu

	userid [PK] integer	parola character varying (255)	email character varying (255)	fullname character varying (255)
1	1	123456	ahmet.yilmaz@example.com	Ahmet Yılmaz
2	2	parola123	ayse.kaya@example.com	Ayşe Kaya
3	3	sifre321	mehmet.ali@example.com	Mehmet Ali
4	4	654321	elif.basak@example.com	Elif Başak
5	5	parola12345	fatma.gul@example.com	Fatma Gül
6	6	123456789	emre.kurt@example.com	Emre Kurt
7	7	asdfghjkl	deniz.tekin@example.com	Deniz Tekin
8	8	zxcvbnm	yasemin.uzun@example.com	Yasemin Uzun
9	9	qweasdzxc	kemal.sunal@example.com	Kemal Sunal
10	10	123qweasd	leyla.aydemir@example.com	Leyla Aydemir

Şirket Tablosu

	companyid [PK] integer	companyname character varying (255)	servicetype character varying (255)	adress character varying (255)	offercount integer	minimumguest integer	maximumguest integer
1	2	Güneş Nişan	Engagement	Ankara	4	20	500
2	3	Ay Nişan Salonu	Engagement	Izmir	9	5	100
3	5	Dağ Organizasyon	Birthday	Kocaeli	5	15	150
4	6	Göl Kır Düğün Salonu	Wedding	Istanbul	8	12	250
5	7	Orman Düğün Salonu	Wedding	Balıkesir	3	10	60
6	8	Nehir Nişan Organizasyon	Engagement	Sakarya	6	20	400
7	9	Okyanus Düğün Salonu	Wedding	Ankara	4	8	80
8	10	Bulut Kafe	Birthday	Batman	10	10	100
9	4	Deniz Kafe	Birthday	Batman	8	30	300
10	1	Yıldız Catering	Wedding	Istanbul	7	10	200
11	11	Davutpaşa	Wedding	Istanbul	1	20	200
12	12	Kutupyıldızı Nişan	Engagement	Ankara	2	20	400
13	13	Mutlu Nişan Salonu	Engagement	Izmir	8	15	100
14	14	Dolunay Kafe	Birthday	Batman	6	30	250
15	15	Kartepe Organizasyon	Birthday	Kocaeli	4	15	100
16	16	İpek Düğün Salonu	Wedding	Istanbul	5	25	250
17	17	Ayvatlar Düğün Salonu	Wedding	Balıkesir	1	10	50
18	18	Sapanca Nişan Organizasyon	Engagement	Sakarya	0	20	200
19	19	Aspava Düğün Salonu	Wedding	Ankara	3	25	150
20	20	Petrol Kafe	Birthday	Batman	9	10	150

Tablolar

Ürün Tablosu

	comid [PK] integer	productname [PK] character varying (255)	productprice numeric	stockquantity integer
1	1	Tavuk Döner	75	100
2	2	Et Döner	150	200
3	3	Lokum	37	10
4	4	Tatil Paketi	3000	15
5	5	Keşkek	150	1000
6	6	Kurabiye	25	50
7	7	Çay	5	20
8	8	Konfeti	100	30
9	9	Akide Şekeri	75	40
10	1	Soğuk Çay	15	25
11	7	Kavurma	175	40
12	4	Kahve	45	25
13	6	Islak kek	45	50
14	1	Oralet	5	20
15	8	Havai fişek	550	30
16	9	Süs balonu	75	40
17	1	Kuru-Pilav	150	25
18	5	Limonata	50	40
19	7	Kola Turka	45	25
20	4	Gelin başı	1500	50

Tablolar

Etkinlik Tablosu

	comid [PK] integer	userid [PK] integer	etkinlikid [PK] integer	season character varying (255)	timeofday character varying (255)	totalperson integer
1	1	1	1001	Yaz	Öğle	50
2	2	2	1002	Kış	Akşam	100
3	3	3	1003	İlkbahar	Sabah	30
4	4	4	1004	Sonbahar	Öğle	40
5	5	5	1005	Yaz	Akşam	70
6	6	6	1006	Kış	Sabah	60
7	7	7	1007	İlkbahar	Öğle	55
8	8	8	1008	Sonbahar	Akşam	80
9	9	9	1009	Yaz	Sabah	45
10	10	10	1010	Kış	Öğle	65

Pricing Tablosu

	comid [PK] integer	season [PK] character varying (255)	timeofday [PK] character varying (255)	priceperperson numeric
1	1	Summer	Afternoon	50
2	2	Winter	Evening	100
3	3	Spring	Morning	30
4	4	Autumn	Afternoon	40
5	5	Summer	Evening	70
6	6	Winter	Morning	60
7	7	Spring	Afternoon	55
8	8	Autumn	Evening	80
9	9	Summer	Morning	45
10	10	Winter	Afternoon	65

Tablolar

Sepet Tablosu

	userid integer	status integer	sepetid [PK] integer	sepettotalcost numeric
1	1	0	1	0
2	2	0	2	0
3	3	0	3	0
4	4	0	4	0
5	5	0	5	0
6	6	0	6	0
7	7	0	7	0
8	8	0	8	0
9	9	0	9	0
10	10	0	10	0

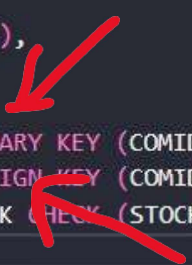
SepetIcindekiler Tablosu

sepid [PK] integer	comid [PK] integer	productname [PK] character varying (255)	adet integer	productprice numeric
-----------------------	-----------------------	---	-----------------	-------------------------

(Sepet içine ürün eklenince doluyor.)



SQL ~ Primary Key- Foreign Key

```
-- URUN Tablosu
CREATE TABLE URUN(
  COMID INT,
  PRODUCTNAME VARCHAR(255),
  PRODUCTPRICE NUMERIC,
  STOCKQUANTITY INT,
  CONSTRAINT PK_URUN PRIMARY KEY (COMID, PRODUCTNAME),
  CONSTRAINT FK_URUN FOREIGN KEY (COMID) REFERENCES SIRKET(COMPANYID) ON DELETE CASCADE,
  CONSTRAINT URUN_STOCK_CK CHECK (STOCKQUANTITY >= 0)
);
```




SQL ~ Sayı Kısıtı :

```
-- PRICING Tablosu
CREATE TABLE PRICING(
  COMID INT,
  SEASON VARCHAR(255),
  TIMEOFDAY VARCHAR(255),
  PRICEPERPERSON NUMERIC,
  CONSTRAINT PK_PRICING PRIMARY KEY (COMID, SEASON, TIMEOFDAY),
  CONSTRAINT FK_PRICING FOREIGN KEY (COMID) REFERENCES SIRKET(COMPANYID) ON DELETE CASCADE,
  CONSTRAINT PRICING_PRICE_CK CHECK (PRICEPERPERSON > 0)
);
```



SQL ~ Silme Kısıtı:

```
-- PRICING Tablosu
CREATE TABLE PRICING(
  COMID INT,
  SEASON VARCHAR(255),
  TIMEOFDAY VARCHAR(255),
  PRICEPERPERSON NUMERIC,
  CONSTRAINT PK_PRICING PRIMARY KEY (COMID, SEASON, TIMEOFDAY),
  CONSTRAINT FK_PRICING FOREIGN KEY (COMID) REFERENCES SIRKET(COMPANYID) ON DELETE CASCADE,
  CONSTRAINT PRICING_PRICE_CK CHECK (PRICEPERPERSON > 0)
);
```



SQL ~ View



View Oluşturma

```
Create View user_uruns
As
Select si.sepid, si.comid, si.productname, si.adet * si.productprice as cost,
si.adet, status, s.userid
From sepeticindekiler si, sepet s
where si.sepid = s.sepetid;

create or replace function trig_fonk_sepeteEkle()
Returns trigger As $$
Declare
    stock int;
Begin
    Select stockquantity into stock
    From urun u
    where u.productname = new.productname and u.comid = new.comid;
    if(stock < new.adet) then
        return old;
    End if;

    Return new;
End;
$$ Language 'plpgsql';
```

SQL ~ Sequences

1.3 Sequences (4)

1.3 etkinlik_etkinlikid_seq

1.3 sepet_sepetid_seq

1.3 sirket_companyid_seq

1.3 userid_seq

Sequence Ekleme

```
-- Sequence oluşturma  
CREATE SEQUENCE USERID_SEQ  
START WITH 1 INCREMENT BY 1  
NO MINVALUE NO MAXVALUE CACHE 1;
```

Sequence Kullanımı

```
-- KULLANICI Tablosu  
CREATE TABLE KULLANICI(  
    USERID INT DEFAULT NEXTVAL('userid_seq') PRIMARY KEY,  
    PAROLA VARCHAR(255),  
    EMAIL VARCHAR(255),  
    FULLNAME VARCHAR(255)  
);
```

SQL ~ Intersect

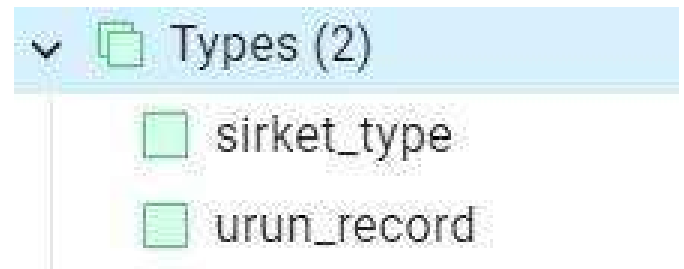
```
Create or replace function list_sirket(ser_type varchar(255), total_person int,
Returns sirket_type[] As $$
Declare
    sirkets sirket_type[];
    cur_sirkets cursor for (select companyid, companyname, priceperperson, offer
                            where s.companyid = p.comid and ser_type = s.servicet
                            and total_person between s.minimumguest and s.maximum
                            intersect
                            select companyid, companyname, priceperperson, offer
                            where adress like sirket_location and s.companyid = p
                            );
    i int default 1;
Begin
    for sirket in cur_sirkets loop
        sirkets[i] = sirket;
        i = i + 1;
    End loop;
    Return sirkets;
End;
$$ Language 'plpgsql';
```

SQL ~ Aggregate (Having)

```
Select servicetype, count(*) , sum(totalPriceEtkinlik) as tmaliyet, avg(totalPriceEtkinlik) as ortmaliyet,
sum(totalperson) as tkisi, avg(totalperson) as ortkisi
From (Select servicetype, totalperson, totalperson * priceperperson as totalPriceEtkinlik
      From sirket s, etkinlik e, pricing pr
      where s.companyid = e.comid and pr.comid = e.comid and ? = userid)
Group by servicetype
having servicetype = ?
```


SQL ~ Record / Cursor

Record - Cursor Örnek 1



```
create type urun_record as (pro_name varchar(255), productprice numeric, comID int, stockQuantity int);

Create or replace function list_urun(pro_name varchar(255), pro_adet int)
returns urun_record[] AS $$
Declare
    uruns urun_record[];
    i int default 1;
    cur_uruns cursor for (Select productname, productprice, comID, stockquantity
                        from urun
                        where productname like CONCAT('%',pro_name,'%') and pro_adet <= stockQuantity);

Begin
    for urun in cur_uruns loop
        uruns[i] = urun;
        i = i + 1;
    end loop;
    Return uruns;
End;
$$ Language 'plpgsql';
```

Record - Cursor Örnek 2

```
create type sirket_type as (comid int, comname varchar(255), priceperperson numeric, offerCount int);

Create or replace function list_sirket(ser_type varchar(255), total_person int, season varchar(255),timeofday varchar(255))
Returns sirket_type[] As $$
Declare
    sirkets sirket_type[];
    cur_sirkets cursor for (select companyid, companyname, priceperperson, offercount from sirket s, pricing p
                        where s.companyid = p.comid and ser_type = s.servicetype and list_sirket.season = p.season
                        and total_person between s.minimumguest and s.maximumguest and list_sirket.timeofday = p.timeofday
                        intersect
                        select companyid, companyname, priceperperson, offercount from sirket s, pricing p
                        where adress like sirket_location and s.companyid = p.comid
                        );
    i int default 1;

Begin
    for sirket in cur_sirkets loop
        sirkets[i] = sirket;
        i = i + 1;
    End loop;
    Return sirkets;
End;
$$ LAnguage 'plpgsql';
```

SQL ~ Fonksiyonlar

Functions (3)

- list_sirket(ser_type character varying, total_person integer, season character varying, timeofday character varying, sirket_location character varying)
- list_urun(pro_name character varying, pro_adet integer)
- totalpriceevent(cid integer, seas character varying, timeday character varying, total integer)

Fonksiyon Örnek 1

```
152 create type sirket_type as (comid int, comname varchar(255), priceperperson numeric, offerCount int);
153
154 Create or replace function list_sirket(ser_type varchar(255), total_person int, season varchar(255),timeofday varchar(255), sirket_location varchar(255))
155 Returns sirket_type[] As $$
156 Declare
157     sirkets sirket_type[];
158     cur_sirkets cursor for (select companyid, companyname, priceperperson, offercount from sirket s, pricing p
159                             where s.companyid = p.comid and ser_type = s.servicetype and list_sirket.season = p.season
160                             and total_person between s.minimumguest and s.maximumguest and list_sirket.timeofday = p.timeofday
161                             intersect
162                             select companyid, companyname, priceperperson, offercount from sirket s, pricing p
163                             where adress like sirket_location and s.companyid = p.comid
164                             );
165     i int default 1;
166 Begin
167     for sirket in cur_sirkets loop
168         sirkets[i] = sirket;
169         i = i + 1;
170     end loop;
171     Return sirkets;
172 End;
173 $$ Language 'plpgsql';
```

Fonksiyon Örnek 2

```
create type urun_record as (pro_name varchar(255), productprice numeric, comID int, stockQuantity int);

Create or replace function list_urun(pro_name varchar(255), pro_adet int)
returns urun_record[] AS $$
Declare
    uruns urun_record[];
    i int default 1;
    cur_uruns cursor for (Select productname, productprice, comID, stockquantity
                           from urun
                           where productname like CONCAT('%',pro_name,'%') and pro_adet <= stockQuantity);

Begin
    for urun in cur_uruns loop
        uruns[i] = urun;
        i = i + 1;
    end loop;
    Return uruns;
End;
$$ Language 'plpgsql';
```


SQL ~ Triggers

- Trigger Functions (6)
 - trig_fonk_buy_urun()
 - trig_fonk_create_user_sepet()
 - trig_fonk_incrementoffercount()
 - trig_fonk_remove_procdut()
 - trig_fonk_sepeteeikle()
 - trig_fonk_sepetetotalcostguncelle()

Trigger örnek 1

```
Create or Replace Function trig_fonk_create_user_sepet()
returns trigger AS $$
Begin
    Insert INTO sepet (userid, status, sepettotalcost) Values (new.userid, 0,0);
    Return new;
End;
$$ Language 'plpgsql';

Create Trigger user_sepet
After Insert
on kullanıcı
for each row execute procedure trig_fonk_create_user_sepet();
```

Trigger örnek 2

```
Create or replace function trig_fonk_buy_urun()
Returns Trigger As $$
Declare
    cur_sepet_uruns cursor for (Select u.comid, s.productname, adet, stockquantity from sepeticindekiler s, urun u
                                where sepid = new.sepid and u.productname = s.productname and s.comid = u.comid);
Begin
    for urun in cur_sepet_uruns loop
        if(urun.adet > urun.stockquantity) then
            Return old;
        End if;
    end loop;
    Insert INTO sepet (userid, status, sepettotalcost) Values (new.userid, 0,0);
    for i in cur_sepet_uruns loop
        update urun set stockquantity = stockquantity - i.adet where comid = i.comid and productname = i.productname;
    end loop;

    Return new;
End;
$$ Language 'plpgsql';

Create trigger buy_urun
Before Update of status
on sepet
for each row execute procedure trig_fonk_buy_urun();
```

SQL ~ Triggers

Trigger örnek 3

```
create or replace function trig_fonk_sepeteTotalCostGuncelle()
Returns trigger As $$
Begin
    if(TG_OP = 'DELETE') then
        update sepet
        set sepettotalcost = sepettotalcost - old.adet * old.productprice
        where sepetid = old.sepid;





    elsif (TG_OP = 'INSERT') then
        update sepet
        set sepettotalcost = sepettotalcost + new.adet * new.productprice
        where sepetid = new.sepid;

    else
        update sepet
        set sepettotalcost = sepettotalcost - old.adet * old.productprice + new.adet * new.productprice
        where sepetid = new.sepid;
    End if;

    Return new;
End;
$$ LAnguage 'plpgsql';

create trigger sepeteTotalCost
After insert or update or delete
on sepeticindekiler
for each row execute procedure trig_fonk_sepeteTotalCostGuncelle();
```

Arayüz :

 **Kaydol**   

Ad:

Soyad:

Mail Adresi:

Parola:

Parola(tekrar):

Kayıt Ol

 **Giriş**   

User ID:

Parola

Giriş Yap

Kayıt Oluştur

Arayüz :

Panel

Organizasyon

Ürün

Geçmiş Organizasyonlarım

Geçmiş Siparişlerim

Organizasyon Tipi

Seçiniz...

Gün Dilimi

Seçiniz...

Kişi Sayısı

Lokasyon

Seçiniz...

Mevsim

Seçiniz...

Filtrele

Sipariş Ver

ComID	Name	Type	Price	Offer Count
-------	------	------	-------	-------------

Panel

Organizasyon

Ürün

Geçmiş Organizasyonlarım

Geçmiş Siparişlerim

Ürün İsmi :

Ürün adeti:

Ürünleri Listele

Ürünler

ComID	Product...	Price	Stock
-------	------------	-------	-------

Miktarı Güncelle

comID:

...

Sepete Ekle

Product Name:

...

Sepetten sil

Quantity:

...

Sipariş Ver

Sepetim

SepetID	ComID	Produc...	Quantity	Price
---------	-------	-----------	----------	-------

Arayüz :

Panel

Organizasyon

Ürün

Geçmiş Organizasyonlarım

Geçmiş Siparişlerim

SepetID	Total Cost
1	18.0

Sepet ID	Product ...	Quantity	Cost
----------	-------------	----------	------

Panel

Organizasyon

Ürün

Geçmiş Organizasyonlarım

Geçmiş Siparişlerim

Event	Company N...	Season	Timeofday	Person Count	Total Price
-------	--------------	--------	-----------	--------------	-------------

Wedding

Listele

Toplam Maliyet
...

Toplam Kişi Sayısı
...

Ortalama Maliyeti
...

Ortalama Kişi Sayısı
...