T.C.
FIRAT ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Yemekhane Otomasyonu Projesi

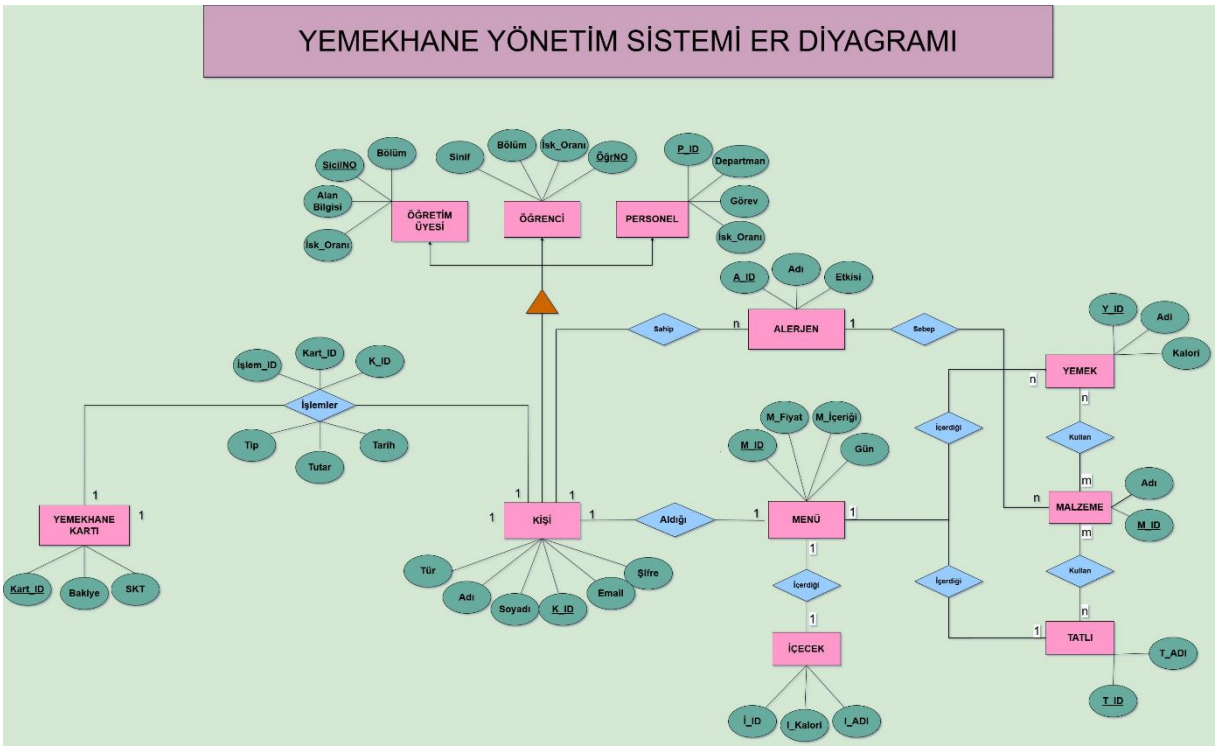
Veri Tabanı Projenin

Son Raporu

Ayşe Ceren Tuncer : 220260011
Azra Şahin : 220260029
Ömer Faruk Yelman : 220260007

Bu proje, bir yemekhane otomasyon sistemi geliştirmeyi amaçlamaktadır. Sistemde, üniversite bünyesindeki öğrenci, öğretim üyesi ve personelin yemek hizmetlerinden faydalanmasını sağlamak için kayıtlı kişilerin işlemleri kolayca takip edilebilir. Her kullanıcının kişisel bilgileri, kart bakiyeleri ve işlemleri yönetilirken, farklı indirim oranları ve işlem türleri (yemek alımı, bakiye yükleme, iade) desteklenir. Sistem, kullanıcıların yemek alışverişlerini kolaylaştırmak, mali kayıtları düzenli tutmak ve yemekhane yönetiminin verimli bir şekilde çalışmasına yardımcı olmak için tasarlanmıştır.

1.ER DİYAGRAMI :



2. İLİŞKİ ŞEMASI OLUŞTURMA AŞAMASI

ER Diyagramından ilişki şeması türetilmiştir. Bu, her varlık ve ilişkiyi açıkça ifade eder.

Varlıklar ve İlişkiler:

ER diyagramından ilişki şeması çıkarıldı:

- Kişi(K_ID, Ad, Soyad, Email, Şifre)
- Öğrenci(ÖğrNO, K_ID ,Bölüm, Sınıf, İsk_Oranı)
- ÖğretimÜyesi(SicilNO, K_ID, AlanBilgisi, Bölüm, İsk_Oranı)
- Personel(P_ID,K_ID, Departman, Görev, İsk_Oranı)
- YemekhaneKartı(Kart_ID, K_ID, Bakiye, SKT)
- İşlemler(İşlem_ID, Kart_ID,K_ID, Tip, Tutar, Tarih)
- Menü(M_ID, Gün, M_Fiyat, M_İçerik)
- Yemek(Y_ID, Adı, Kalori)

- Menü_Yemek(M_ID, Y_ID)
- Malzeme(M_ID, Adı)
- Yemek_Malzeme(Y_ID, M_ID)
- Alerjen(A_ID, Adı, Etkisi)
- Yemek_Alerjen(Y_ID, A_ID, Sebep)
- Tatlı(T_ID, Adı)
- Tatlı_Malzeme(T_ID, M_ID)
- İçecek(I_ID, Adı, Kalori)
- Menü_İçecek(M_ID, I_ID)
- Alerjen_Malzeme(A_ID, M_ID)
- Alerjen_Kişi(A_ID, K_ID)

3. NORMALİZASYON YAPMA AŞAMASI

Adım 1: 1NF (Birinci Normal Form)

1NF'ye göre, tüm hücreler atomik olmalı ve her hücrede yalnızca bir değer olmalıdır.
Bu tablo zaten 1NF'ye uygundur, çünkü tüm hücrelerde atomik değerler bulunmaktadır.

Adım 2: 2NF (İkinci Normal Form)

2NF'ye göre, kısmi bağımlılıklar kaldırılmalıdır. Bu tabloda `Kart_ID → Adı` bağımlılığı bulunmaktadır. Bu bağımlılık kısmi bağımlılığa yol açmaktadır. Bu nedenle tablo ikiye bölünür:

Adım 3: 3NF (Üçüncü Normal Form)

3NF'ye göre transitif bağımlılıklar kaldırılmalıdır. `Kart_ID → K_ID → Adı` transitif bir bağımlılıktır.

Bunları yaptıktan sonra yeni ilişki şeması aşağıdaki gibi oluyor

- Kişi(K_ID, Ad, Soyad, Email, Şifre,Kart_ID)
- Öğrenci(ÖğrNO, K_ID ,Bölüm, Sınıf, İsk_Oranı)
- ÖğretimÜyesi(SicilNO, K_ID, AlanBilgisi, Bölüm, İsk_Oranı)
- Personel(P_ID,K_ID, Departman, Görev, İsk_Oranı)
- YemekhaneKartı(Kart_ID, K_ID, Bakiye, SKT)
- İşlemler(İşlem_ID, Kart_ID, Tip, Tutar, Tarih)
- Menü(M_ID, Gün, M_Fiyat, M_İçerik)
- Yemek(Y_ID, Adı, Kalori)
- Menü_Yemek(M_ID, Y_ID)
- Malzeme(M_ID, Adı)
- Yemek_Malzeme(Y_ID, M_ID)
- Alerjen(A_ID, Adı, Etkisi)
- Yemek_Alerjen(Y_ID, A_ID, Sebep)
- Tatlı(T_ID, Adı)

- Tatlı_Malzeme(T_ID, M_ID)
- İçecek(I_ID, Adı, Kalori)
- Menü_İçecek(M_ID, I_ID)
- Alerjen_Malzeme(A_ID, M_ID)
- Alerjen_Kişi(A_ID, K_ID)

Önce

ALTER TABLE Kisi

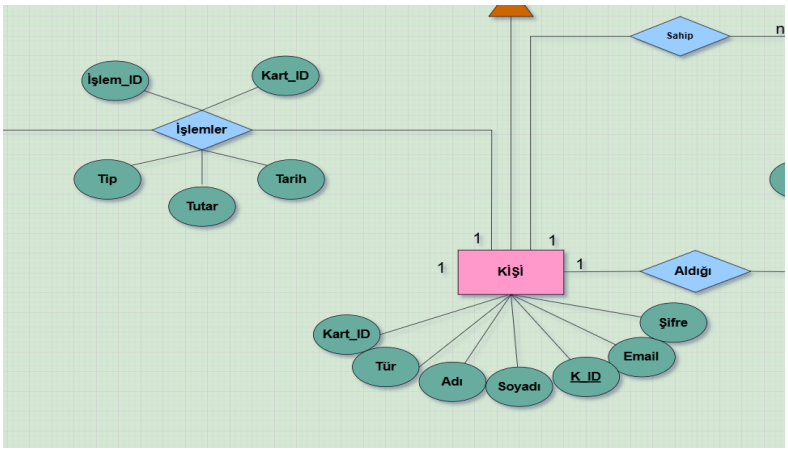
ADD KartId INT;

FOREIGN KEY (KartId) REFERENCES YemekhaneKarti (KartId); Kodları ile Kisi tablosuna KartId ekledik

Sonra

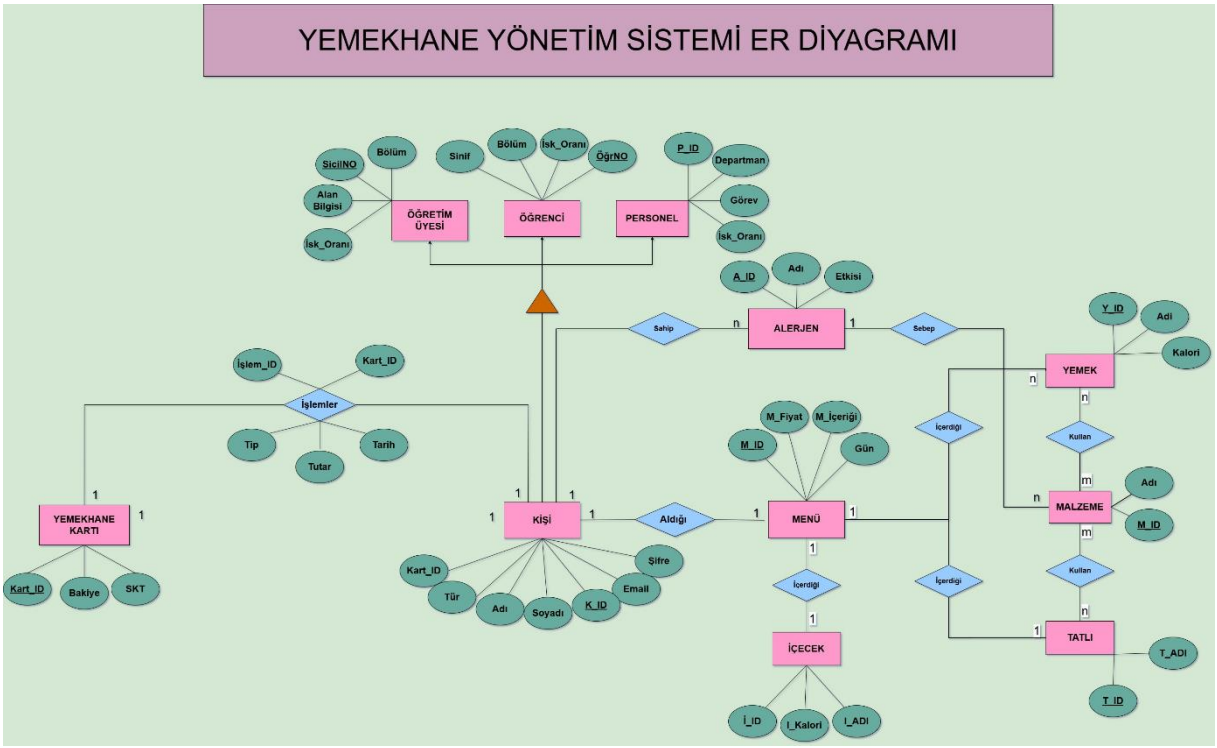
ALTER TABLE Islemler

DROP COLUMN K_ID; Kodları ile de İşlemler tablosundan K_ID sütununu kaldırdık.



Normalizasyondan sonra değişiklik olan kısmı gösteriyor

- **Normalizasyondan sonra ER diyagramının son hali**



4. VERİTABANI TABLOLARI OLUŞTURMA AŞAMASI

Veritabanı Oluşturma

```
Create Database YemekhaneYönetimSistemi
use YemekhaneYönetimSistemi
```

- **Malzeme Tablosu:**

```
Create Table Malzeme(
malzemeId INT IDENTITY(1,1) PRIMARY KEY,
malzemeAdi NVARCHAR(50),
);
```

- **Yemek Tablosu:**

```
Create Table Yemek(
yemekId INT IDENTITY(1,1) PRIMARY KEY,
yemekAdi NVARCHAR(50),
kalori FLOAT);
```

- **Yemek-Malzeme İlişkisi Tablosu:**

```
Create Table YemekMalzeme(
yemekId INT,
malzemeId INT,
PRIMARY KEY (yemekId, malzemeId),
FOREIGN KEY (yemekId) REFERENCES Yemek(yemekId),
```

```
FOREIGN KEY (malzemeId) REFERENCES Malzeme(malzemeId),
);
```

- **Tatlı Tablosu:**

```
Create Table Tatli(
tatliId INT IDENTITY(1,1) PRIMARY KEY,
tatliAdi NVARCHAR(50),
kalori FLOAT
);
```

- **Tatlı-Malzeme İlişkisi Tablosu:**

```
Create Table TatliMalzeme(
tatliId INT,
malzemeId INT,
PRIMARY KEY (tatliId, malzemeId),
FOREIGN KEY (tatliId) REFERENCES Tatli(tatliId),
FOREIGN KEY (malzemeId) REFERENCES Malzeme(malzemeId),
);
```

- **İçecek Tablosu:**

```
Create Table Icecek(
icecekId INT IDENTITY(1,1) PRIMARY KEY,
icecekAdi NVARCHAR(50),
kalori FLOAT
);
```

- **Alerjen Tablosu:**

```
Create Table Alerjen(
alerjenId INT IDENTITY(1,1) PRIMARY KEY,
alerjenAdi NVARCHAR(50),
alerjenEtkisi NVARCHAR(100)
);
```

- **Alerjen-Malzeme İlişkisi Tablosu:**

```
Create Table AlerjenMalzeme(
alerjenId INT,
malzemeId INT,
PRIMARY KEY (alerjenId, malzemeId),
FOREIGN KEY (alerjenId) REFERENCES Alerjen(alerjenId),
FOREIGN KEY (malzemeId) REFERENCES Malzeme(malzemeId),
);
```

- **Menü Tablosu:**

```
Create Table Menu(
menuId INT IDENTITY(1,1) PRIMARY KEY,
Menuiçeriği NVARCHAR(50),
```

```
gun DATE,
fiyat FLOAT
);
```

- **Menü-Yemek İlişkisi Tablosu:**

```
Create Table MenuYemek(
menuId INT,
yemekId INT,
PRIMARY KEY (menuId, yemekId),
FOREIGN KEY (menuId) REFERENCES Menu(menuId),
FOREIGN KEY (yemekId) REFERENCES Yemek(yemekId),
);
```

- **Menü-Tatlı İlişkisi Tablosu:**

```
Create Table MenuTatli(
menuId INT,
tatliId INT,
PRIMARY KEY (menuId, tatliId),
FOREIGN KEY (menuId) REFERENCES Menu(menuId),
FOREIGN KEY (tatliId) REFERENCES Tatli(tatliId),
);
```

- **Menü-İçecek İlişkisi Tablosu:**

```
Create Table MenuIcecek(
menuId INT,
icecekId INT,
PRIMARY KEY (menuId, icecekId),
FOREIGN KEY (menuId) REFERENCES Menu(menuId),
FOREIGN KEY (icecekId) REFERENCES Icecek(icecekId),
);
```

- **Yemekhane Kartı Tablosu:**

```
Create Table YemekhaneKarti (
 kartId INT IDENTITY(1,1) PRIMARY KEY,
 bakiye FLOAT,
 skt DATE
);
```

- **Kişi Tablosu:**

```
Create Table Kisi(
kisiId INT IDENTITY(1,1) PRIMARY KEY,
adi NVARCHAR(50),
soyadi NVARCHAR(50),
email NVARCHAR(50),
sifre NVARCHAR(20),
Tur NVARCHAR(20),
kartId INT,
FOREIGN KEY(kartId) REFERENCES YemekhaneKarti(kartId)
```

);

- **Öğrenci Tablosu:**

```
Create Table Ogrenci(
ogrenciNo INT PRIMARY KEY,
bolum NVARCHAR(50),
fakulte NVARCHAR(50),
sinif INT,
iskontoOrani FLOAT,
kisiId INT,
FOREIGN KEY(kisiId) REFERENCES Kisi(kisiId)
);
```

- **Öğretim Üyesi Tablosu:**

```
Create Table OgretimUyesi(
sicilNo INT PRIMARY KEY,
bolum NVARCHAR(50),
fakulte NVARCHAR(50),
iskontoOrani FLOAT,
kisiId INT,
FOREIGN KEY(kisiId) REFERENCES Kisi(kisiId)
);
```

- **Personel Tablosu:**

```
Create Table Personel(
personelId INT PRIMARY KEY,
departman NVARCHAR(50),
gorev NVARCHAR(50),
iskontoOrani FLOAT,
kisiId INT,
FOREIGN KEY(kisiId) REFERENCES Kisi(kisiId)
);
```

- **Alerjen-Kişi İlişkisi Tablosu:**

```
Create Table AlerjenKisi(
alerjenId INT,
kisiId INT,
PRIMARY KEY (alerjenId, kisiId ),
FOREIGN KEY (alerjenId) REFERENCES Alerjen(alerjenId),
FOREIGN KEY (kisiId ) REFERENCES Kisi(kisiId ),
);
```

- **İşlemler Tablosu:**

```
Create Table Islemler (
islemId INT IDENTITY(1,1)  PRIMARY KEY,
kartId INT NOT NULL,
tip BIT NOT NULL,
```

```
    tutar FLOAT NOT NULL,
    tarih DATE NOT NULL,
    FOREIGN  KEY (kartId) REFERENCES YemekhaneKarti(kartId),
    FOREIGN KEY (kisiId) REFERENCES Kisi(kisiId)
);
```

5. VERİTABANI TABLOLARININ İÇERİKLERİNİ DOLDURMA:

Tabloların oluşturma komutları kısaca verilmiş ve örnek olarakta tabloların bir kısmı fotoğraf olarak gösterilmiştir.

- **Kisi Tablosu:**

```
INSERT INTO Kisi (Kisi_ıd ,adi, soyadi, email, sifre, kartId,tur)
VALUES
(Sibel, Deniz, 'sibel.deniz@example.com', 'password1', 1),
(Yusuf, Güler, 'yusuf.guler@example.com', 'password2', 2),
```

...

kisiId	adi	soyadi	email	sifre	kartId	tur
23	Sibel	Deniz	sibel.deniz@example.com	password23	23	Ogrenci
24	Yusuf	Güler	yusuf.guler@example.com	password24	24	Ogrenci
25	Derya	Şimşek	derya.simsek@example.com	password25	25	Ogrenci
26	Mert	Kaplan	mert.kaplan@example.com	password26	26	Ogrenci
27	Pelin	Er	pelin.er@example.com	password27	27	Ogrenci
28	Ozan	Çakır	ozan.cakir@example.com	password28	28	Ogrenci
29	Hande	Kaya	hande.kaya@example.com	password29	29	Ogrenci
30	Çağlar	Kılıç	caglar.kilic@example.com	password30	30	Ogrenci
31	Betül	Çoban	betul.coban@example.com	password31	31	Personel
32	Eray	Gül	eray.gul@example.com	password32	32	Personel
33	Tuğçe	Doğan	tugce.dogan@example.com	password33	33	Personel
34	Kerem	Yıldız	kerem.yildiz@example.com	password34	34	Personel
35	Selin	Orhan	selin.orhan@example.com	password35	35	Personel
36	Gökhan	Tan	gokhan.tan@example.com	password36	36	Personel
37	Ebru	Kılıç	ebru.kilic@example.com	password37	37	Personel
38	Volkan	Yüce	volkan.yuce@example.com	password38	38	Personel
39	Filiz	Türkm...	filiz.turkmen@example.com	password39	39	Personel
40	Harun	Erol	harun.erol@example.com	password40	40	Personel
41	Esra	Kumru	esra.kumru@example.com	password41	41	Personel
42	Cenk	Şen	cenk.sen@example.com	password42	42	Personel
43	Dilara	Alkan	dilara.alkan@example.com	password43	43	Personel
44	Furkan	Boz	furkan.boz@example.com	password44	44	Personel

- **Personel Tablosu:**

```
INSERT INTO Personel (personelId, departman, gorev, iskontoOrani, kisiId)
VALUES
(1, 'İdari İşler', 'Ofis Yöneticisi', 0.10, 31),
(2, 'Mali İşler', 'Muhasebe', 0.10, 32),
(3, 'Destek Hizmetleri', 'Temizlik Personeli', 0.10, 33),
```

...

personelId	departman	gorev	iskontoOrani	kisiId
1	İdari İşler	Ofis Yöneticisi	0,1	31
2	Mali İşler	Muhasebe	0,1	32
3	Destek Hizmetleri	Temizlik Personeli	0,1	33
4	Teknik Servis	Elektrikçi	0,1	34
5	Bilgi İşlem	Sistem Uzmanı	0,1	35
6	İdari İşler	Ofis Yöneticisi	0,1	36
7	Mali İşler	Muhasebe	0,1	37
8	Destek Hizmetleri	Temizlik Personeli	0,1	38
9	Teknik Servis	Elektrikçi	0,1	39
10	Bilgi İşlem	Sistem Uzmanı	0,1	40
11	İdari İşler	Ofis Yöneticisi	0,1	41
12	Mali İşler	Muhasebe	0,1	42
13	Destek Hizmetleri	Temizlik Personeli	0,1	43
14	Teknik Servis	Elektrikçi	0,1	44
15	Bilgi İşlem	Sistem Uzmanı	0,1	45
16	İdari İşler	Ofis Yöneticisi	0,1	46

- **OgretimUyesi Tablosu:**

```
INSERT INTO OgretimUyesi (sicilNo, bolum, fakulte, iskontoOrani, kisiId)
VALUES
(1, 'Bilgisayar Mühendisliği', 'Mühendislik Fakültesi', 0.30, 1),
(2, 'Elektrik Mühendisliği', 'Mühendislik Fakültesi', 0.30, 2),
```

.....

sicilNo	bolum	fakulte	iskontoOrani	kisiId
1	Bilgisayar Mühendisliği	Mühendislik Fakültesi	0,3	1
2	Elektrik Mühendisliği	Mühendislik Fakültesi	0,3	2
3	Makine Mühendisliği	Mühendislik Fakültesi	0,3	3
4	İnşaat Mühendisliği	Mühendislik Fakültesi	0,3	4
5	İnşaat Mühendisliği	Mühendislik Fakültesi	0,3	5
6	Endüstri Mühendisliği	Mühendislik Fakültesi	0,3	6
7	Kimya Mühendisliği	Mühendislik Fakültesi	0,3	7
8	Bilgisayar Mühendisliği	Mühendislik Fakültesi	0,3	8
9	Elektrik Mühendisliği	Mühendislik Fakültesi	0,3	9
10	Makine Mühendisliği	Mühendislik Fakültesi	0,3	10

- **Ogrenci Tablosu:**

```
INSERT INTO Ogrenci (ogrenciNo, bolum, fakulte, sinif, iskontoOrani, kisiId)
VALUES
(1001, 'Bilgisayar Mühendisliği', 'Mühendislik Fakültesi', 1, 0.20, 11),
(1002, 'Elektrik Mühendisliği', 'Mühendislik Fakültesi', 2, 0.20, 12),
```

...

	OgrenciNo	bolum	fakulte	sinif	İskontoOrani	kisild
1	1001	Bilgisayar Mühendisliği	Mühendislik Fakültesi	1	0,2	11
2	1002	Elektrik Mühendisliği	Mühendislik Fakültesi	2	0,2	12
3	1003	Makine Mühendisliği	Mühendislik Fakültesi	1	0,2	13
4	1004	İnşaat Mühendisliği	Mühendislik Fakültesi	3	0,2	14
5	1005	İnşaat Mühendisliği	Mühendislik Fakültesi	2	0,2	15
6	1006	Endüstri Mühendisliği	Mühendislik Fakültesi	1	0,2	16
7	1007	Kimya Mühendisliği	Mühendislik Fakültesi	2	0,2	17
8	1008	Bilgisayar Mühendisliği	Mühendislik Fakültesi	1	0,2	18
9	1009	Elektrik Mühendisliği	Mühendislik Fakültesi	3	0,2	19
10	1010	Makine Mühendisliği	Mühendislik Fakültesi	2	0,2	20
11	1011	Endüstri Mühendisliği	Mühendislik Fakültesi	1	0,2	21
12	1012	Kimya Mühendisliği	Mühendislik Fakültesi	3	0,2	22
13	1013	Bilgisayar Mühendisliği	Mühendislik Fakültesi	2	0,2	23
14	1014	Elektrik Mühendisliği	Mühendislik Fakültesi	1	0,2	24
15	1015	Makine Mühendisliği	Mühendislik Fakültesi	2	0,2	25
16	1016	İnşaat Mühendisliği	Mühendislik Fakültesi	3	0,2	26

- **Malzeme Tablosu**

```
INSERT INTO Malzeme (malzemeAdi)
VALUES
('Un'),
('Şeker'),
('Tuz'),
('Zeytinyağı'),
('Pirinç'),
```

...

	malzemeId	malzemeAdi
1	1	Un
2	2	Şeker
3	3	Tuz
4	4	Zeytinyağı
5	5	Pirinç
6	6	Bulgur
7	7	Makarna
8	8	Tavuk
9	9	Dana Eti
10	10	Balık
11	11	Süt
12	12	Yumurta
13	13	Yoğurt
14	14	Tereyağı
15	15	Patates
16	16	Soğan
17	17	Domates

- **Yemek Tablosu:**

INSERT INTO Yemek (yemekAdi, kalori)
VALUES
('Mercimek Çorbası', 120),
('Domates Çorbası', 100),
...
('Yoğurtlu Patates Salatası',150)

Y_ID	Adi	Kalori
1	Mercimek Çorbası	120
2	Domates Çorbası	100
3	Ezogelin Çorbası	130
4	Tarhana Çorbası	120
5	Sebze Çorbası	110
6	Etli Kuru Fasulye	350
7	Tavuklu Şehriyeli Pilav	290
8	Karnıyarık	420
9	Taze Fasulye	180
10	Zeytinyağlı Dolma	230
11	Etli Nohut	350
12	Sebzeli Güveç	220
13	Lahmacun	310
14	Tereyağlı Pilav	320
15	Mantı	480
16	Fırında Köfte	400
17	Bulgur Pilavı	280
18	Fırında Tavuk	300
19	Kremalı Makarna	450

- **Icecek Tablosu:**

INSERT INTO Icecek (icecekAdi, kalori)
VALUES
('Su', 0),
('Çay', 20),
...

I_ID	I_Adi	I_Kalori
1	Su	0
2	Çay	20
3	Kahve	30
4	Ayran	50
5	Limonata	100
6	Gazoz	120
7	Kola	150
8	Fanta	160
9	Meyve Suyu	120
10	Soda	0
11	Bitki Çayı	15
12	Sıcak Çikola...	300
13	Maden Suyu	0

- **Tatlı Tablosu:**

INSERT INTO Tatli (tatliAdi, kalori)

VALUES
('Baklava', 600),
('Sütlaç', 300),

....

T_ID	T_Adi	Kalori
1	Baklava	600
2	Sütlaç	300
3	Revani	450
4	Künefe	700
5	Şekerpare	500
6	Tulumba Tat...	1000
7	Helva	500

- **Menu Tablosu:**

INSERT INTO Menu (MenüİçeriĞi, Gun, M_Fiyat)
VALUES
-- 1. Hafta
('Mercimek Çorbası, Etli Kuru Fasulye, Tereyağlı Pilav, Çoban Salata, Kola, Baklava', 'Pazartesi', 50),
('Domates Çorbası, Karnıyarık, Bulgur Pilavı, Mevsim Salatası, Ayran, Sütlaç', 'Salı', 50),
('Ezogelin Çorbası, Tavuklu Şehriyeli Pilav, Çoban Salata, Limonata, Revani', 'Çarşamba', 50),
('Tarhana Çorbası, Zeytinyağlı Dolma, Pirinç Pilavı, Rus Salatası, Gazoz, Künefe', 'Perşembe', 50),
('Sebze Çorbası, Fırında Köfte, Sebzeli Türlü, Çoban Salata, Fanta, Şekerpare', 'Cuma', 50),

-- 2. Hafta
('Mercimek Çorbası, Kuzu Tandır, Tereyağlı Pilav, Çoban Salata, Meyve Suyu, Tulumba Tatlısı', 'Pazartesi', 50),
('Domates Çorbası, Hünkar Beğendi, Bulgur Pilavı, Mevsim Salatası, Soda, Helva', 'Salı', 50),
('Ezogelin Çorbası, Fırında Tavuk, Çoban Salata, Kola, Baklava', 'Çarşamba', 50),
('Tarhana Çorbası, Etli Nohut, Pirinç Pilavı, Rus Salatası, Gazoz, Sütlaç', 'Perşembe', 50),
('Sebze Çorbası, Sebzeli Kebap, Çoban Salata, Ayran, Revani', 'Cuma', 50),

...

M_ID	MenüİçeriĞi	Gun	M_Fiyat
1	Mercimek Çorbası, Etli Kuru Fasulye, Tereyağlı Pilav, Çoban Salata, Kola, Baklava	Pazartesi	50.00
2	Domates Çorbası, Karnıyarık, Bulgur Pilavı, Mevsim Salatası, Ayran, Sütlaç	Salı	50.00
3	Ezogelin Çorbası, Tavuklu Şehriyeli Pilav, Çoban Salata, Limonata, Revani	Çarşamba	50.00
4	Tarhana Çorbası, Zeytinyağlı Dolma, Pirinç Pilavı, Rus Salatası, Gazoz, Künefe	Perşembe	50.00
5	Sebze Çorbası, Fırında Köfte, Sebzeli Türlü, Çoban Salata, Fanta, Şekerpare	Cuma	50.00
6	Mercimek Çorbası, Kuzu Tandır, Tereyağlı Pilav, Çoban Salata, Meyve Suyu, Tulumba Tatlısı	Pazartesi	50.00
7	Domates Çorbası, Hünkar Beğendi, Bulgur Pilavı, Mevsim Salatası, Soda, Helva	Salı	50.00
8	Ezogelin Çorbası, Fırında Tavuk, Çoban Salata, Kola, Baklava	Çarşamba	50.00
9	Tarhana Çorbası, Etli Nohut, Pirinç Pilavı, Rus Salatası, Gazoz, Sütlaç	Perşembe	50.00
10	Sebze Çorbası, Sebzeli Kebap, Çoban Salata, Ayran, Revani	Cuma	50.00
11	Mercimek Çorbası, Tavuk Şiş, Tereyağlı Pilav, Çoban Salata, Fanta, Künefe	Pazartesi	50.00
12	Domates Çorbası, Izgara Köfte, Bulgur Pilavı, Mevsim Salatası, Meyve Suyu, Şekerpare	Salı	50.00
13	Ezogelin Çorbası, Sebzeli Güveç, Çoban Salata, Limonata, Tulumba Tatlısı	Çarşamba	50.00
14	Tarhana Çorbası, Etli Bezelye, Pirinç Pilavı, Rus Salatası, Soda, Helva	Perşembe	50.00
15	Sebze Çorbası, Fırında Köfte, Çoban Salata, Kola, Baklava	Cuma	50.00

- **MenuTatli Tablosu:**

```
INSERT INTO MenuTatli (menuID, tatliID)
VALUES
-- 1. Hafta

(1, 1), -- Pazartesi: Baklava
(2, 2), -- Salı: Sütlaç
(3, 3), -- Çarşamba: Revani
(4, 4), -- Perşembe: Künefe
(5, 5), -- Cuma: Şekerpare
```

...

	M_ID	T_ID
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	1
9	9	2
10	10	3
11	11	4
12	12	5
13	13	6
14	14	7
15	15	1

- **MenuYemek Tablosu**

```
INSERT INTO MenuYemek (M_ID, Y_ID)
VALUES
 (1, 1), -- Pazartesi: Mercimek Çorbası
 (1, 6), -- Pazartesi: Etli Kuru Fasulye
 (1, 10), -- Pazartesi: Tereyağlı Pilav
```

.....

	M_ID	Y_ID
1	1	1
2	1	6
3	1	10
4	1	36
5	2	4
6	2	13
7	2	16
8	2	37
9	3	3
10	3	12
11	3	22
12	3	38

- **Menu İçecek Tablosu**

INSERT INTO MenuİcecekM_ID,I_ID),
VALUES

--1. Hafta
(1,4) --Pazartesi=Ayran
(2,5) --Salı=Limonata
(3, 6), -- Çarşamba: Gazoz
(4, 1), -- Perşembe: Su
(5, 8), -- Cuma: Fanta
.....

	M_ID	I_ID
1	1	4
2	2	5
3	3	6
4	4	1
5	5	8
6	6	4
7	7	6
8	8	7
9	9	5
10	10	8
11	11	4
12	12	6

- **Tatlı Malzeme**

INSERT INTO TatliMalzeme (tatliId, malzemeId)
VALUES
(1, 1), -- Baklava: Un
(1, 2), -- Baklava: Şeker
.....

	T_ID	M_ID
1	1	1
2	1	2
3	1	12
4	1	13
5	1	14
6	1	40
7	2	2
8	2	7
9	2	11
10	2	14
11	3	1
12	3	2
13	3	12
14	3	14

- **Yemek Malzeme Tablosu**

INSERT INTO YemekMalzeme (yemekId, malzemeId)
VALUES

-- Mercimek Çorbası
(1, 28), -- Mercimek
(1, 16), -- Soğan
(1, 3), -- Tuz
(1, 4), -- Zeytinyağı
(1, 17), -- Domates

-- Domates Çorbası
(2, 17), -- Domates
(2, 3), -- Tuz
(2, 4), -- Zeytinyağı

	Y_ID	M_ID
1	1	3
2	1	4
3	1	16
4	1	17
5	1	28
6	2	3
7	2	4
8	2	17
9	3	16
10	3	17
11	3	28
12	3	29
13	4	3
14	4	4
15	4	16
16	4	30

- **Yemekhane Kartı Tablosu**

```
INSERT INTO YemekhaneKarti (bakiye, skt)
VALUES
(100.0, '2025-01-01'),
(50.0, '2025-02-01'),
(75.5, '2025-03-01'),
```

kartId	bakiye	skt
1	100	2025-01-01
2	50	2025-02-01
3	75,5	2025-03-01
4	120	2025-04-01
5	200	2025-05-01
6	150	2025-06-01
7	80	2025-07-01
8	110	2025-08-01
9	90	2025-09-01
10	60	2025-10-01
11	70	2026-01-01
12	95	2026-02-01
13	130	2026-03-01
14	85	2026-04-01
15	115	2026-05-01

- **Alerjen Tablosu:**

```
INSERT INTO Alerjen (alerjenAdi, alerjenEtkisi)
VALUES
('Gluten', 'Çölyak hastalığına neden olabilir.'),
('Laktoz', 'Laktoz intoleransına yol açabilir.'),
('Fıstık', 'Ağır alerjik reaksiyonlara neden olabilir.'),
```

...

A_ID	Adi	Etkisi
1	Gluten	Çölyak hastalığına neden olabilir.
2	Laktoz	Laktoz intoleransına yol açabilir.
3	Fıstık	Ağır alerjik reaksiyonlara neden olabilir.
4	Yumurta	Yumurta alerjisine neden olabilir.
5	Deniz Ürünleri	Deniz ürünü alerjisine yol açabilir.
6	Soya	Soya alerjisine neden olabilir.
7	Susam	Hassasiyet veya alerjik reaksiyona neden olabilir.
8	Hardal	Alerjik semptomlara neden olabilir.
9	Sarımsak	Sindirim sorunlarına neden olabilir.
10	Bal	Alerjik reaksiyonlara neden olabilir.

- **AlerjenMalzeme Tablosu:**

```
INSERT INTO AlerjenMalzeme (alerjenId, malzemeId)
VALUES
(1, 1),  -- Gluten -> Un
(1, 6),  -- Gluten -> Bulgur
(1, 7),  -- Gluten -> Makarna
(2, 11), -- Laktoz -> Süt
```

...

	A_ID	M_ID
1	1	1
2	1	6
3	1	7
4	2	11
5	2	13
6	2	14
7	2	37
8	3	38
9	3	39
10	3	40
11	4	12
12	5	9
13	6	18
14	7	41

- **AlerjenKisi Tablosu:**

```
INSERT INTO AlerjenKisi (alerjenId, kisiId)
VALUES
(1, 1),
(2, 2),
(3, 3),
```

...

	A_ID	K_ID
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10

- **Islemler Tablosu:**

```
INSERT INTO Islemler (Kart_ID, tip, tutar, tarih)
VALUES
(1,1,50.0, '2025-01-01'),
```

(2,0,20.0, '2025-01-05'),
(3,1,30.0,,'2025-02-01'),
(4,0,15.0, '2025-02-10'),

...

Islem_ID	kartId	tip	tutar	tarih
1	1	1	50	2025-01-01
2	1	0	35	2025-01-05
3	2	1	30	2025-02-01
4	2	0	35	2025-02-10
5	3	1	60	2025-03-01
6	3	0	35	2025-03-15
7	4	1	70	2025-04-01
8	4	0	35	2025-04-10
9	5	1	80	2025-05-01
10	5	0	35	2025-05-15
11	6	1	40	2025-06-01
12	6	0	35	2025-06-20
13	7	1	50	2025-07-01
14	7	0	35	2025-07-15
15	8	1	60	2025-08-01
16	8	0	35	2025-08-20
17	9	1	70	2025-09-01
18	9	0	35	2025-09-15
19	10	1	80	2025-10-01
20	10	0	35	2025-10-20

SQL Procedur:

Bu prosedür kişi yemekhane kartına para yükleme işlemini gerçekleştiriyor

```
CREATE PROCEDURE ParaYukle (
   @kisiId INT,
   @miktar FLOAT)
AS
BEGIN
   BEGIN TRANSACTION;
 BEGIN TRY
      DECLARE @kartId INT;
      SELECT @kartId = kartId FROM Kisi WHERE kisiId = @kisiId;

      UPDATE YemekhaneKarti
      SET bakiye = bakiye + @miktar
      WHERE kartId = @kartId;

      INSERT INTO Islemler (kartId, kisiId, tip, tutar, tarih)
      VALUES (@kartId, @kisiId, 1, @miktar, GETDATE()); -- tip=1 para yükleme
 PRINT 'İşlem başarılı: Para yüklendi.';
     COMMIT TRANSACTION;
   END TRY
   BEGIN CATCH
      ROLLBACK TRANSACTION;
      THROW;
   END CATCH
END;
```

SQL serverdaki kod gösterimi

SQLQuery44.sql - CE...(ceren\cerent (77))* | SQLQuery43.sql - CE...(ceren\cerent (85)) | SQLQuery39.sql

```sql
-- Para yükleme işlemi öncesinde bakiyeyi kontrol et
DECLARE @oncekiBakiye FLOAT, @guncelBakiye FLOAT;

-- Kişinin mevcut bakiyesini al
SELECT @oncekiBakiye = bakiye
FROM YemekhaneKarti
WHERE kartId = (SELECT kartId FROM Kisi WHERE kisiId = 1);

PRINT 'Para yüklemeden önceki bakiye: ' + CAST(@oncekiBakiye AS VARCHAR);

-- Para yükleme işlemi
EXEC ParaYukle @kisiId = 1, @miktar = 100;

-- Para yükleme sonrası bakiyeyi al
SELECT @guncelBakiye = bakiye
FROM YemekhaneKarti
WHERE kartId = (SELECT kartId FROM Kisi WHERE kisiId = 1);

PRINT 'Para yükleme sonrası bakiye: ' + CAST(@guncelBakiye AS VARCHAR);
```

110 %

Messages

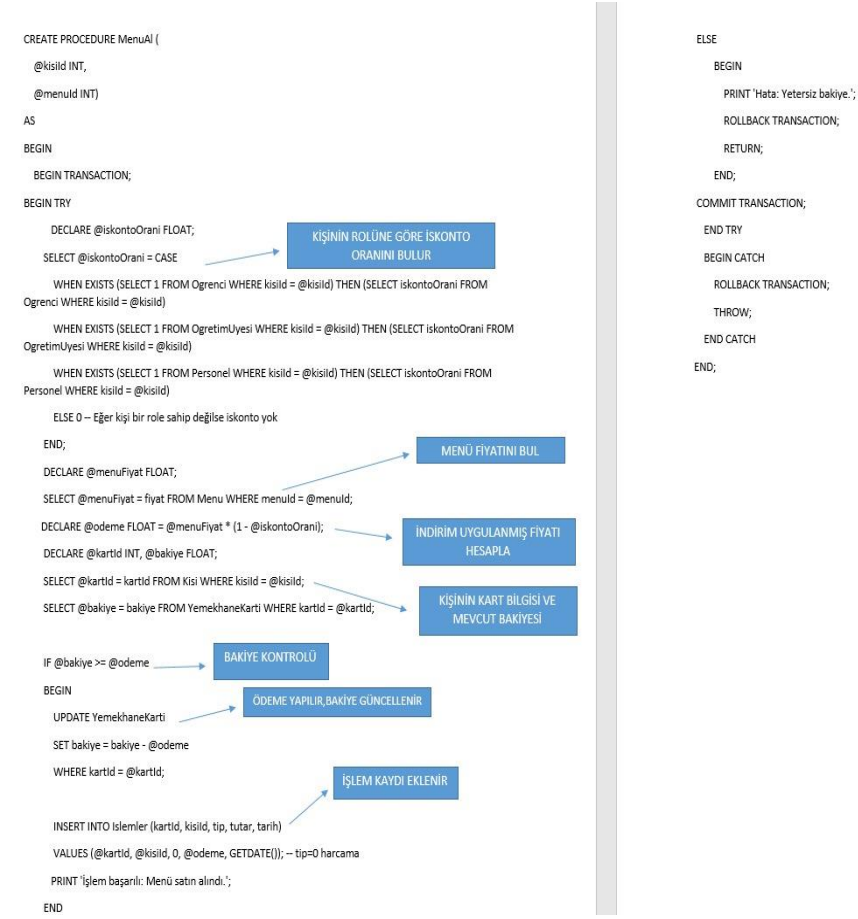
```
Para yüklemeden önceki bakiye: 355

(1 row affected)

(1 row affected)
İşlem başarılı: Para yüklendi.
Para yükleme sonrası bakiye: 455

Completion time: 2025-01-05T17:27:15.9874679+03:00
```

Bu prosedür kişi yemekhanede menü satın alma işlemini gerçekleştiriyor

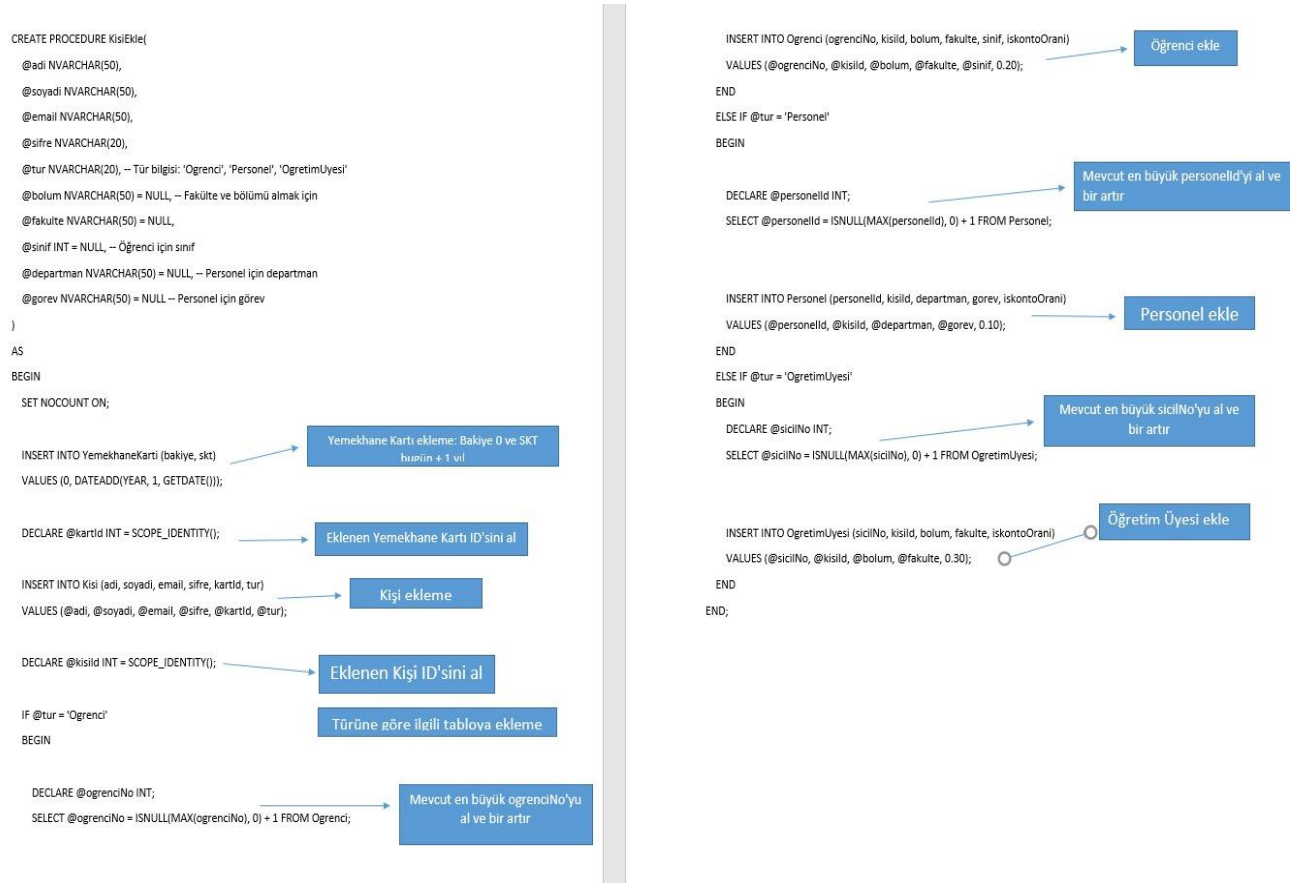



```
CREATE PROCEDURE MenuAl (
    @kisiId INT,
    @menuId INT)
AS
BEGIN
    BEGIN TRANSACTION;
BEGIN TRY
        DECLARE @iskontoOrani FLOAT;
    SELECT @iskontoOrani = CASE
        WHEN EXISTS (SELECT 1 FROM Ogrenci WHERE kisiId = @kisiId) THEN (SELECT iskontoOrani FROM
Ogrenci WHERE kisiId = @kisiId)
        WHEN EXISTS (SELECT 1 FROM OgretimUyesi WHERE kisiId = @kisiId) THEN (SELECT iskontoOrani FROM
OgretimUyesi WHERE kisiId = @kisiId)
        WHEN EXISTS (SELECT 1 FROM Personel WHERE kisiId = @kisiId) THEN (SELECT iskontoOrani FROM
Personel WHERE kisiId = @kisiId)
        ELSE 0 -- Eğer kişi bir role sahip değilse iskonto yok
    END;
    DECLARE @menuFiyat FLOAT;
    SELECT @menuFiyat = fiyat FROM Menu WHERE menuId = @menuId;
    DECLARE @odeme FLOAT = @menuFiyat * (1 - @iskontoOrani);
    DECLARE @kartId INT, @bakiye FLOAT;
    SELECT @kartId = kartId FROM Kisi WHERE kisiId = @kisiId;
    SELECT @bakiye = bakiye FROM YemekhaneKarti WHERE kartId = @kartId;

    IF @bakiye >= @odeme
    BEGIN
      UPDATE YemekhaneKarti
      SET bakiye = bakiye - @odeme
      WHERE kartId = @kartId;

      INSERT INTO Islemler (kartId, kisiId, tip, tutar, tarih)
      VALUES (@kartId, @kisiId, 0, @odeme, GETDATE()); -- tip=0 harcama
     PRINT 'İşlem başarılı: Menü satın alındı.';
    END
    ELSE
        BEGIN
          PRINT 'Hata: Yetersiz bakiye.';
          ROLLBACK TRANSACTION;
          RETURN;
        END;
COMMIT TRANSACTION;
    END TRY
    BEGIN CATCH
      ROLLBACK TRANSACTION;
      THROW;
    END CATCH
END;
```

Bu prosedür sisteme kişi ekleme işlemini gerçekleştiriyor



Saklı Yordam ve Tetikleyici (Stored Procedures and Triggers):

"trigger" (tetikleyici), belirli bir veritabanı olayı gerçekleştiğinde otomatik olarak çalışan, önceden tanımlanmış SQL kodu parçasıdır.

Yemekhane kartındaki bakiyeyi günceller

```
CREATE TRIGGER BakiyeKontrolTrigger
ON YemekhaneKarti
AFTER UPDATE
AS
BEGIN
  IF EXISTS (
    SELECT 1
    FROM YemekhaneKarti
    WHERE bakiye < 0
  )
  BEGIN
    PRINT 'UYARI: Negatif bakiye tespit edildi. İşlem iptal ediliyor.';
    ROLLBACK TRANSACTION;
  END;
END;
DECLARE @oncekiBakiye FLOAT, @guncelBakiye FLOAT;

SELECT @oncekiBakiye = bakiye
FROM YemekhaneKarti
WHERE kartId = (SELECT kartId FROM Kisi WHERE kisiId = 1);

PRINT 'Para yüklemeden önceki bakiye: ' + CAST(@oncekiBakiye AS VARCHAR);
EXEC ParaYukle @kisiId = 1, @miktar = 100;

SELECT @guncelBakiye = bakiye
FROM YemekhaneKarti
WHERE kartId = (SELECT kartId FROM Kisi WHERE kisiId = 1);

PRINT 'Para yükleme sonrası bakiye: ' + CAST(@guncelBakiye AS VARCHAR);
```

Para yükleme işlemi öncesinde bakiyeyi kontrol et

Kişinin mevcut bakiyesini al

Para yükleme işlemi

Para yükleme sonrası bakiyeyi al

Önceki bakiye

Results | Messages

	kartId	bakiye	skt
1	1	420	2025-01-01
2	2	50	2025-02-01
3	3	75,5	2025-03-01
4	4	120	2025-04-01
5	5	200	2025-05-01
6	6	150	2025-06-01
7	7	80	2025-07-01
8	8	110	2025-08-01
9	9	90	2025-09-01
10	10	60	2025-10-01
11	11	70	2026-01-01
12	12	95	2026-02-01

Sonraki Bakiye

	kartId	bakiye	skt
1	1	385	2025-01-01
2	2	50	2025-02-01
3	3	75,5	2025-03-01
4	4	120	2025-04-01
5	5	200	2025-05-01
6	6	150	2025-06-01
7	7	80	2025-07-01
8	8	110	2025-08-01
9	9	90	2025-09-01
10	10	60	2025-10-01
11	11	70	2026-01-01
12	12	95	2026-02-01

Kişinin yemekhane kartındaki bakiyesi menü alma işleminden önce 420 TL.Kişi öğretim üyesidir.Öğretim üyesinin iskonto oranı 0.3.Menü fiyatı 50TL.İndirim uygulandığında 50-(0,3*50)=35 sonucu elde edilir.Yani öğretim üyesinin menü için ödeyeceği fiyat 35 TL'dir.Bakiyedeki son durum 420-35=385TL olarak güncellenir.

Transaction Ve Rollback-Commit İşlemleri:

Birden fazla SQL ifadesinin bir arada çalıştığı bir işlem içinde, tüm ifadelerin başarılı bir şekilde çalışması durumunda işlemin onaylanması, aksi takdirde geri alınması sağlanmış oluyor.

--Eğer aşağıda yazılan tüm komutlar doğruysa, hepsi başarılı bir şekilde çalışacaktır; ancak, yanlış olan bir tanesi dahi olsa, hiçbiri çalışmayacaktır.

```
CREATE PROCEDURE ParaYukle (
   @kisiId INT,
   @miktar FLOAT
)
AS
BEGIN
   --   BEGIN TRANSACTION;
   BEGIN TRY
      DECLARE @kartId INT;
      SELECT @kartId = kartId FROM Kisi WHERE kisiId = @kisiId;
      UPDATE YemekhaneKarti
      SET bakiye = bakiye + @miktar
      WHERE kartId = @kartId;
      INSERT INTO Islemler (kartId, kisiId, tip, tutar, tarih)
      VALUES (@kartId, @kisiId, 1, @miktar, GETDATE()); -- tip=1 para yükleme

      PRINT 'İşlem başarılı: Para yüklendi.';

      COMMIT TRANSACTION;
   END TRY
   BEGIN CATCH
         ROLLBACK TRANSACTION;
      THROW;
   END CATCH
END;
```

Transaction başlatılır

Hata durumunda işlem geri alınır

Rollback işlemi sırasında hata meydana geldiğinde, işlemin başlatılmasından önceki duruma geri döner. Bu veri tabanında güncellenmiş bir durumda bırakılmayı korur ve verilerin bütünlüğünü korur.

Eğer hata yok ise de **Commit** işlemi atılır ve kod tamamen kaydedilir.

```sql
-- Menü satın alma işlemi (Yetersiz bakiye ile hata senaryosu)
BEGIN TRY
    EXEC MenuAl @kisiId = 1, @menuId = 9999; -- Geçersiz bir menuId seçerek hata oluştur
END TRY
BEGIN CATCH
    PRINT 'İşlem başarısız: ' + ERROR_MESSAGE();
END CATCH
```

0 %

Messages

```
Hata: Yetersiz bakiye.

Completion time: 2025-01-05T23:10:53.4437926+03:00
```

Prosedür,Trigger Rollback ve Commit kodları:

```
CREATE PROCEDURE ParaYukle (
   @kisiId INT,
   @miktar FLOAT
)
AS
BEGIN
   -- Transaction başlatılır
   BEGIN TRANSACTION;

   BEGIN TRY
      -- Kişinin kart bilgisi
      DECLARE @kartId INT;
      SELECT @kartId = kartId FROM Kisi WHERE kisiId = @kisiId;

      -- Bakiyeyi güncelle
      UPDATE YemekhaneKarti
      SET bakiye = bakiye + @miktar
      WHERE kartId = @kartId;

      -- İşlem kaydı eklenir
      INSERT INTO Islemler (kartId, kisiId, tip, tutar, tarih)
      VALUES (@kartId, @kisiId, 1, @miktar, GETDATE()); -- tip=1 para yükleme

      PRINT 'İşlem başarılı: Para yüklendi.';

      COMMIT TRANSACTION;
   END TRY
   BEGIN CATCH
      -- Hata durumunda işlem geri alınır
      ROLLBACK TRANSACTION;
      THROW;
   END CATCH
END;
```

```
CREATE PROCEDURE MenuAl (
   @kisiId INT,
   @menuId INT
)
AS
BEGIN
   -- Transaction başlatılır
   BEGIN TRANSACTION;

   BEGIN TRY
      -- Kişinin rolüne göre iskonto oranını bul
      DECLARE @iskontoOrani FLOAT;
      SELECT @iskontoOrani = CASE
         WHEN EXISTS (SELECT 1 FROM Ogrenci WHERE kisiId = @kisiId) THEN (SELECT
iskontoOrani FROM Ogrenci WHERE kisiId = @kisiId)
         WHEN EXISTS (SELECT 1 FROM OgretimUyesi WHERE kisiId = @kisiId) THEN (SELECT
iskontoOrani FROM OgretimUyesi WHERE kisiId = @kisiId)
         WHEN EXISTS (SELECT 1 FROM Personel WHERE kisiId = @kisiId) THEN (SELECT
iskontoOrani FROM Personel WHERE kisiId = @kisiId)
         ELSE 0 -- Eğer kişi bir role sahip değilse iskonto yok
      END;

      -- Menü fiyatını bul
      DECLARE @menuFiyat FLOAT;
      SELECT @menuFiyat = fiyat FROM Menu WHERE menuId = @menuId;

      -- İndirim uygulanmış fiyatı hesapla
      DECLARE @odeme FLOAT = @menuFiyat * (1 - @iskontoOrani);

      -- Kişinin kart bilgisi ve mevcut bakiyesi
      DECLARE @kartId INT, @bakiye FLOAT;
      SELECT @kartId = kartId FROM Kisi WHERE kisiId = @kisiId;
      SELECT @bakiye = bakiye FROM YemekhaneKarti WHERE kartId = @kartId;

      -- Bakiye kontrolü
      IF @bakiye >= @odeme
      BEGIN
         -- Ödeme yapılır, bakiye güncellenir
         UPDATE YemekhaneKarti
         SET bakiye = bakiye - @odeme
         WHERE kartId = @kartId;

         -- İşlem kaydı eklenir
         INSERT INTO Islemler (kartId, kisiId, tip, tutar, tarih)
         VALUES (@kartId, @kisiId, 0, @odeme, GETDATE()); -- tip=0 harcama
```

```
            PRINT 'İşlem başarılı: Menü satın alındı.';
        END
        ELSE
        BEGIN
            -- Yetersiz bakiye uyarısı
            PRINT 'Hata: Yetersiz bakiye.';
            ROLLBACK TRANSACTION;
            RETURN;
        END;

        COMMIT TRANSACTION;
    END TRY
    BEGIN CATCH
        -- Hata durumunda işlem geri alınır
        ROLLBACK TRANSACTION;
        THROW;
    END CATCH
END;
```

```
CREATE TRIGGER BakiyeKontrolTrigger
ON YemekhaneKarti
AFTER UPDATE
AS
BEGIN
   IF EXISTS (
      SELECT 1
      FROM YemekhaneKarti
      WHERE bakiye < 0
   )
   BEGIN
      PRINT 'UYARI: Negatif bakiye tespit edildi. İşlem iptal ediliyor.';
      ROLLBACK TRANSACTION;
   END;
END;



-- Para yükleme işlemi öncesinde bakiyeyi kontrol et
DECLARE @oncekiBakiye FLOAT, @guncelBakiye FLOAT;

-- Kişinin mevcut bakiyesini al
SELECT @oncekiBakiye = bakiye
FROM YemekhaneKarti
WHERE kartId = (SELECT kartId FROM Kisi WHERE kisiId = 1);

PRINT 'Para yüklemeden önceki bakiye: ' + CAST(@oncekiBakiye AS VARCHAR);

-- Para yükleme işlemi
EXEC ParaYukle @kisiId = 1, @miktar = 100;

-- Para yükleme sonrası bakiyeyi al
SELECT @guncelBakiye = bakiye
FROM YemekhaneKarti
WHERE kartId = (SELECT kartId FROM Kisi WHERE kisiId = 1);

PRINT 'Para yükleme sonrası bakiye: ' + CAST(@guncelBakiye AS VARCHAR);



-- Menü satın alma işlemi öncesinde bakiyeyi kontrol et
DECLARE @oncekiBakiyeMenu FLOAT, @guncelBakiyeMenu FLOAT;

-- Kişinin mevcut bakiyesini al
```

```
SELECT @oncekiBakiyeMenu = bakiye
FROM YemekhaneKarti
WHERE kartId = (SELECT kartId FROM Kisi WHERE kisiId = 1);

PRINT 'Menü satın almadan önceki bakiye: ' + CAST(@oncekiBakiyeMenu AS VARCHAR);

-- Menü satın alma işlemi
EXEC MenuAl @kisiId = 1, @menuId = 1;

-- Menü satın alma sonrası bakiyeyi al
SELECT @guncelBakiyeMenu = bakiye
FROM YemekhaneKarti
WHERE kartId = (SELECT kartId FROM Kisi WHERE kisiId = 1);

PRINT 'Menü satın alma sonrası bakiye: ' + CAST(@guncelBakiyeMenu AS VARCHAR);
```

```
CREATE PROCEDURE KisiEkle(
    @adi NVARCHAR(50),
    @soyadi NVARCHAR(50),
    @email NVARCHAR(50),
    @sifre NVARCHAR(20),
    @tur NVARCHAR(20), -- Tür bilgisi: 'Ogrenci', 'Personel', 'OgretimUyesi'
    @bolum NVARCHAR(50) = NULL, -- Fakülte ve bölümü almak için
    @fakulte NVARCHAR(50) = NULL,
    @sinif INT = NULL, -- Öğrenci için sınıf
    @departman NVARCHAR(50) = NULL, -- Personel için departman
    @gorev NVARCHAR(50) = NULL -- Personel için görev
)
AS
BEGIN
    SET NOCOUNT ON;

    -- Yemekhane Kartı ekleme: Bakiye 0 ve SKT bugün + 1 yıl
    INSERT INTO YemekhaneKarti (bakiye, skt)
    VALUES (0, DATEADD(YEAR, 1, GETDATE()));

    -- Eklenen Yemekhane Kartı ID'sini al
    DECLARE @kartId INT = SCOPE_IDENTITY();

    -- Kişi ekleme
    INSERT INTO Kisi (adi, soyadi, email, sifre, kartId, tur)
    VALUES (@adi, @soyadi, @email, @sifre, @kartId, @tur);

    -- Eklenen Kişi ID'sini al
    DECLARE @kisiId INT = SCOPE_IDENTITY();

    -- Türüne göre ilgili tabloya ekleme
    IF @tur = 'Ogrenci'
    BEGIN
        -- Mevcut en büyük ogrenciNo'yu al ve bir artır
        DECLARE @ogrenciNo INT;
        SELECT @ogrenciNo = ISNULL(MAX(ogrenciNo), 0) + 1 FROM Ogrenci;

        -- Öğrenci ekle
        INSERT INTO Ogrenci (ogrenciNo, kisiId, bolum, fakulte, sinif, iskontoOrani)
        VALUES (@ogrenciNo, @kisiId, @bolum, @fakulte, @sinif, 0.20);
    END
    ELSE IF @tur = 'Personel'
    BEGIN
        -- Mevcut en büyük personelId'yi al ve bir artır
        DECLARE @personelId INT;
        SELECT @personelId = ISNULL(MAX(personelId), 0) + 1 FROM Personel;
```

```
        -- Personel ekle
        INSERT INTO Personel (personelId, kisiId, departman, gorev, iskontoOrani)
        VALUES (@personelId, @kisiId, @departman, @gorev, 0.10);
    END
    ELSE IF @tur = 'OgretimUyesi'
    BEGIN
        -- Mevcut en büyük sicilNo'yu al ve bir artır
        DECLARE @sicilNo INT;
        SELECT @sicilNo = ISNULL(MAX(sicilNo), 0) + 1 FROM OgretimUyesi;

        -- Öğretim Üyesi ekle
        INSERT INTO OgretimUyesi (sicilNo, kisiId, bolum, fakulte, iskontoOrani)
        VALUES (@sicilNo, @kisiId, @bolum, @fakulte, 0.30);
    END
END;
GO


EXEC KisiEkle
    @adi = 'Mehmet',
    @soyadi = 'Kara',
    @email = 'mehmet@example.com',
    @sifre = '9012',
    @tur = 'OgretimUyesi',
    @bolum = 'Bilgisayar Mühendisliği',
    @fakulte = 'Teknoloji Fakültesi';
```