

**GAZİ ÜNİVERSİTESİ**

**TEKNOLOJİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**BMT-311 VERİ TABANI YÖNETİM SİSTEMLERİ**

**2024-2025 GÜZ DÖNEMİ**

**PROJE RAPORU**

**GRUP ÜYELERİ:**

**22181616033- İsmail Emre Candan**

**22181616051– Şevval Özdemir**

## **İÇİNDEKİLER**

[**İÇİNDEKİLER** 2](#_heading=h.gjdgxs)

[1. GİRİŞ 3](#_heading=h.30j0zll)

[2. PROJE İÇERİĞİ 4](#_heading=h.1fob9te)

[2.1. PROJE TANITIMI 4](#_heading=h.3znysh7)

[2.2. PROJE TASARIM VE GELİŞTİRME AŞAMALARI 4](#_heading=h.2et92p0)

[2.3. KULLANILAN TEKNOLOJİLER 4](#_heading=h.tyjcwt)

[3. VERİ TABANI TASARIM AŞAMALARI 5](#_heading=h.3dy6vkm)

[4. GÜNCEL VERİTABANI VE ER DİYAGRAMI 6](#_heading=h.1t3h5sf)

[5. SONUÇ 7](#_heading=h.4d34og8)

[6. KAYNAKÇA 8](#_heading=h.2s8eyo1)

1. **GİRİŞ**

Projemizin konusu bir otel otomasyonu oluşturmak ve bu oteli en iyi şartlar altında sanal ortamda yönetmek,verileri gözlemlemek.Otel operasyonlarını kolaylaştırmak için bir yönetim sistemi geliştirmeyi hedefledik. Günümüzde otellerde rezervasyon, müşteri takibi, oda fiyatlandırması ve muhasebe gibi işlemler genellikle manuel ya da dağınık sistemler üzerinden yürütülmektedir. Bu durum verimsizliklere ve hatalara yol açabilmektedir.

Projenin amacı, bu süreçleri tek bir platform altında toplayarak otel çalışanlarının işlerini daha hızlı ve hatasız yapmasını sağlamaktır. Sistem, müşteri bilgilerini kaydetmek, rezervasyon işlemlerini düzenlemek, oda harcamalarını ve muhasebe işlemlerini yönetmek için tasarlanmıştır.

**Hedef Kitle**

**Küçük ve Orta Ölçekli Oteller:**

* + Rezervasyon yönetimi, müşteri takibi ve muhasebe işlemleri için etkili bir çözüme ihtiyaç duyan oteller.
  + Düşük maliyetle, pratik ve kolay entegre edilebilir bir sistem arayan işletmeler.

**Otel Yöneticileri:**

* + Operasyonları daha düzenli yönetmek ve iş süreçlerini analiz etmek isteyen karar vericiler.
  + Gelir-gider takibini doğru ve hızlı bir şekilde yapmak isteyen yöneticiler.

Projenizin GitHub veya Gitlab public repo adresini yazınız.

1. **PROJE İÇERİĞİ**

Bu kısımda aşağıda verilen maddeleri düz yazı olarak açıklamanız gerekmektedir.

* 1. **PROJE TANITIMI**

Projenizin amaçlarını, yeteneklerini, çözümü olduğunu düşündüğünüz problemi/problemleri, çözüm önerisini tasarlarken nasıl bir yaklaşım sergilediğinizi detaylı olarak açıklayınız.

Projenin amacı, otel rezervasyon sistemi için veritabanı altyapısı oluşturarak muhasebe, müşteri, rezervasyon ve oda yönetimi gibi süreçlerin düzenli ve hızlı bir şekilde yürütülmesini sağlamaktır.

* **Problem:** Geleneksel rezervasyon sistemlerinde veri karmaşıklığı ve tekrarlayan veri girişleri nedeniyle hatalar ve karışıklıklar meydana gelmektedir. Bu, veri analizi ve raporlamada zorluk yaratmaktadır.Yöneticilerin verilerinin ele geçirilmesi güvenlik problemleri yaratabilir
* **Çözüm:** Veriler normalizasyon teknikleri ile yapılandırılmış, güçlü ilişkisel bir veritabanı tasarımı ile saklanmıştır. Bu yapı, rezervasyonların, müşterilerin, oda ve masrafların merkezi bir sistem üzerinden yönetilmesini kolaylaştırır.Yönetici verileri güvenli bir şekilde saklanmıştır.
* **Yaklaşım:** PostgreSQL veritabanı kullanılarak, tablolardaki veriler güçlü anahtar-kısıtlar (primary key ve foreign key) ile ilişkilendirilmiştir. Normalizasyon, veri tekrarını önlemek ve verimliliği artırmak için uygulanmıştır.Yönetici verileri şifrelenerek saklanmıştır.
  1. **PROJE TASARIM VE GELİŞTİRME AŞAMALARI**

**Tasarım Aşaması**

Tasarım aşaması, projenin temel yapısının oluşturulduğu ve gereksinimlere uygun bir veri modeli ile kullanıcı arayüzlerinin planlandığı bölümdür.

a. Gereksinim Analizi

1. Hedef Belirleme:
   * Müşteri, oda, rezervasyon, masraf ve finansal işlemleri kaydedebilen bir otel yönetim sistemi oluşturulması hedeflenmiştir.
2. İş Süreçlerinin Belirlenmesi:
   * Rezervasyon oluşturma, masraf ekleme, müşteri bilgilerini yönetme ve finansal raporlamanın sistemde olması gerektiği belirlenmiştir.

b. Veri Modelleme

1. Varlıkların ve İlişkilerin Belirlenmesi:
   * Kavramsal tasarımda belirlenen varlıklar ve ilişkiler doğrultusunda veri modeli oluşturulmuştur.
   * Normalizasyon süreci tamamlanarak veritabanı yapısındaki veri tekrarlarının önüne geçilmiştir.
2. ER Diyagramı Hazırlama:
   * Varlıklar (örneğin, customers\_normalized, room\_normalized) ve ilişkiler (örneğin, müşteri-rezervasyon, oda-rezervasyon ilişkileri) görselleştirilmiştir.

c. Kullanıcı Arayüzü Tasarımı

1. Formlar ve Fonksiyonlar:
   * Rezervasyon Formu: Kullanıcının yeni rezervasyon ekleyebilmesi için tasarlandı.
   * Masraf Yönetim Formu: Rezervasyon veya oda masraflarının girilebilmesi için bir ekran oluşturuldu.
   * Raporlama Arayüzü: Finansal işlemlerin ve rezervasyon durumlarının raporlanması için bir alan tasarlandı.
2. Kullanıcı Deneyimi:
   * Arayüzler kullanıcı dostu ve basit bir yapıda tasarlandı.

**Geliştirme Aşaması**

Geliştirme aşamasında sistemin kodlanması ve işlevselliklerinin uygulamaya geçirilmesi sağlandı.

a. Veritabanı Oluşturma

* PostgreSQL kullanılarak normalleştirilmiş bir veritabanı oluşturuldu.
* Tablolar arası ilişkiler FOREIGN KEY kısıtlamaları ile tanımlandı.
* Trigger veya Stored Procedure gibi mekanizmalar gereklilik halinde kullanıldı.

b. Kodlama

1. Backend (Arka Plan) Geliştirme:
   * Rezervasyon işlemleri ve masraf yönetimi gibi işlemleri yöneten fonksiyonlar yazıldı.
   * PostgreSQL ile etkileşimli sorgular oluşturuldu.
   * Transaction mekanizması kullanılarak veri bütünlüğü sağlandı.
2. Frontend (Ön Yüz) Geliştirme:
   * Tüm panelleri bir arada tutan giriş ekranı oluşturuldu.
   * Kullanıcıların kolayca rezervasyon yapabileceği ve masraf ekleyebileceği bir grafik arayüz (GUI) oluşturuldu.
   * Adminlerin otel verilerini gözlemleyebileceği panel
   * Kullanıcıların rezervasyon iptali yapabileceği panel eklendi

c. Test Süreci

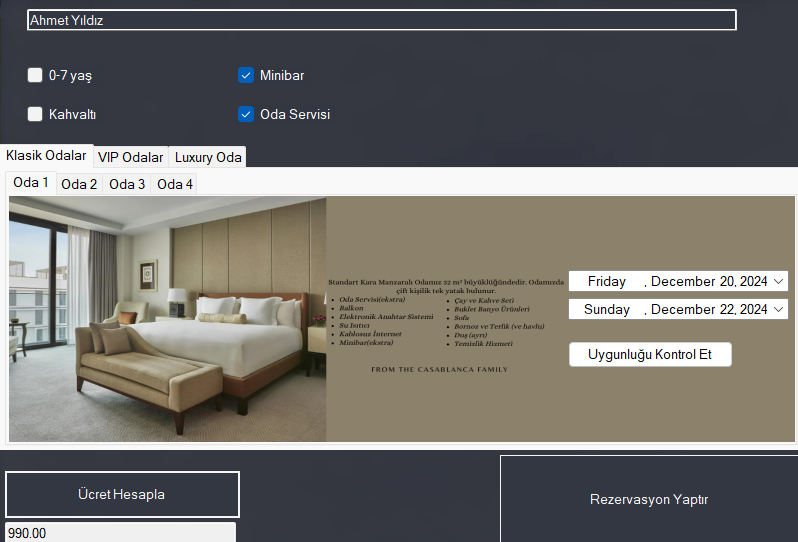
* Veritabanı sorguları ve fonksiyonlar bir dizi testten geçirildi.
* Arayüz ve veritabanı etkileşimi üzerinde yapılan testlerle hatalar tespit edilip giderildi.

d. Karşılaşılan Problemler ve Çözümleri

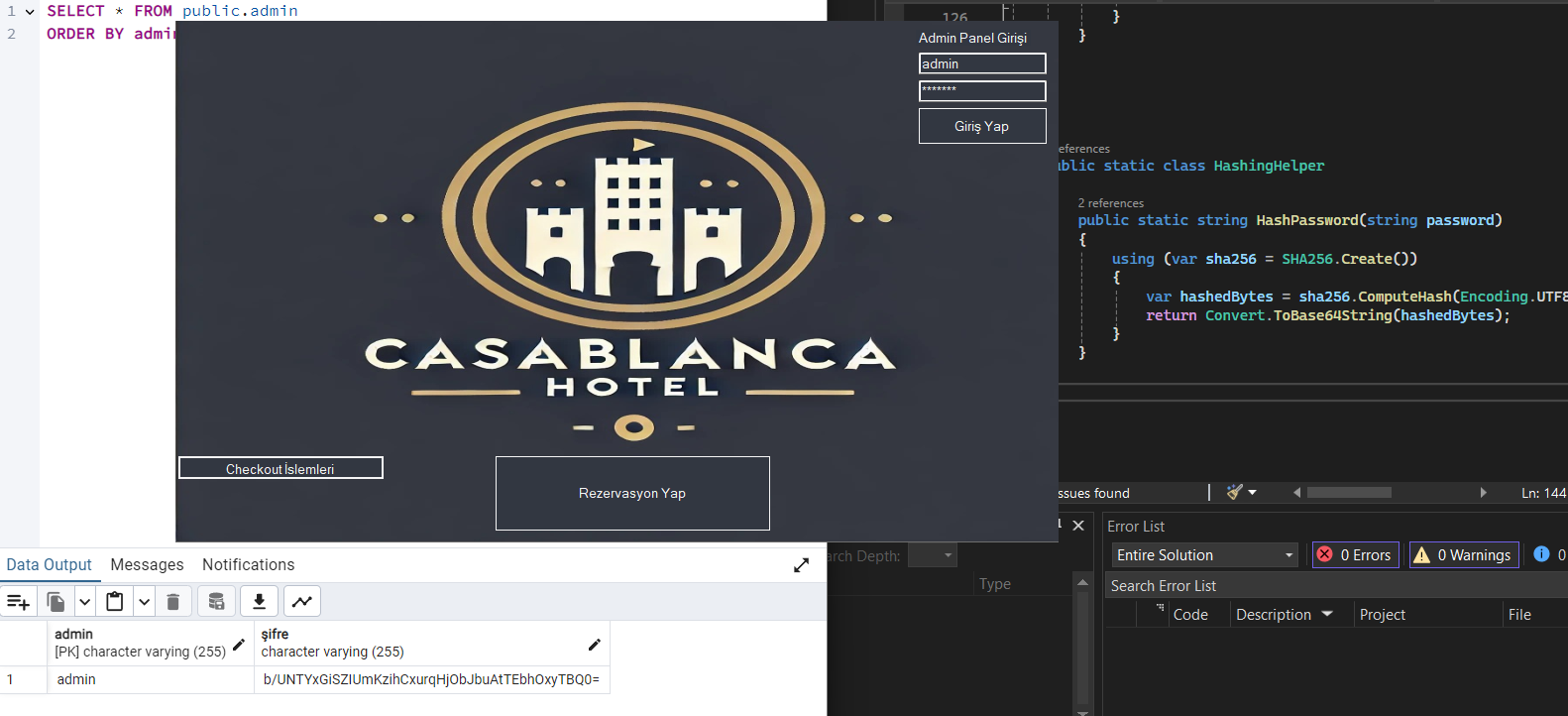
1. Problem: İlişkiler arasındaki karmaşıklık.
   * Çözüm: Kavramsal tasarımda belirlenen ilişkiler doğrultusunda detaylı bir normalizasyon yapıldı.
2. Problem: Masraf türlerinin ve tutarlarının yönetimi.
   * Çözüm: Masraflar için ayrı tablolar (reservation\_expenses\_normalized ve room\_expenses\_normalized) oluşturuldu.
3. Problem: Toplam fiyat hesabının yapılması
   * Çözüm: Veritabanında fee\_calculator ve dynamic\_price adlı iki fonksiyon oluşturuldu bu sayede kullanıcının girdiği verilere göre oda masrafları ve odanın kendi dinamik fiyatı ayrı ayrı hesaplanarak en son toplandı.
4. Problem: Aynı tarihe birden fazla rezervasyon yapılması
   * Çözüm: Veritabanında kullanılan check\_availablity fonksiyonuyla seçilen odanın müsaitlik durumuna göre rezervasyon izni verildi
5. Problem: Aynı Müşterinin farklı rezervasyonlarında customer tablosuna aynı müşteriyi yazdırma
   * Çözüm:C# üzerinde önce müşterinin customer tablosundaki varlığı kontrol ediliyor ardından bu duruma göre customer tablosuna insert yapılıyor.
   1. **KULLANILAN TEKNOLOJİLER**

Projede PostgreSQL veritabanı için,C# kullanıcı arayüzü ve uygulamanın genel işleyişi için kullanıldı.  
PostgreSQL tercih edilmesinin sebebi açık kaynaklı olması sayesinde bir problemle karşılaşıldığında çözümünü kolaylıkla araştırarak bulunabilecek olması.

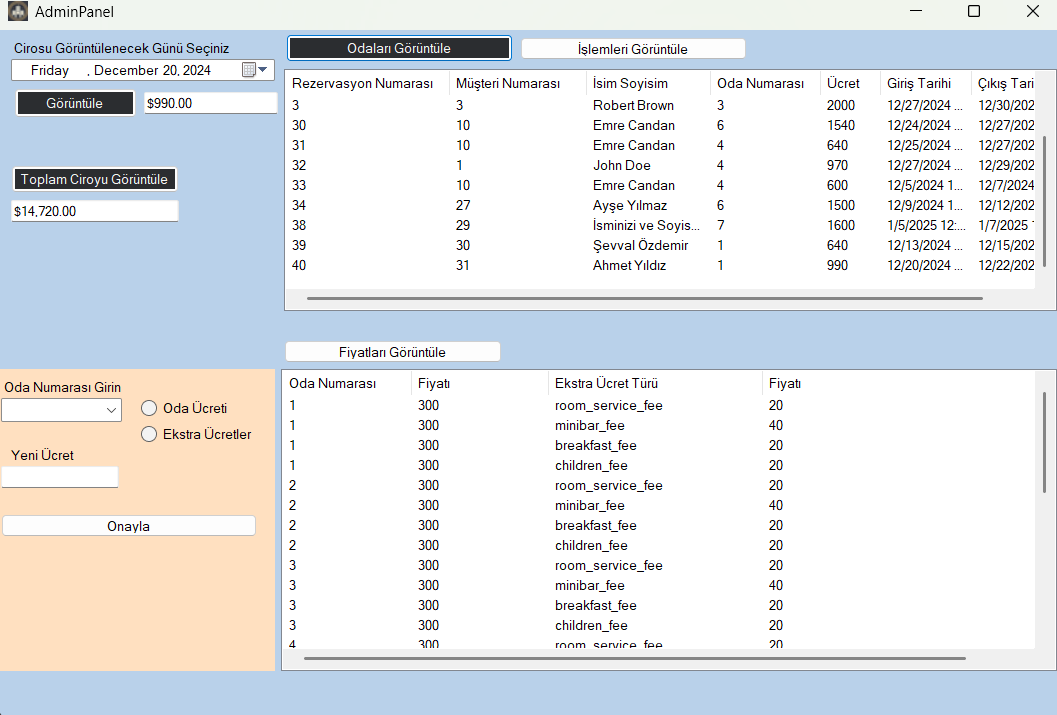
* 1. **Bazı Arayüz Görüntüleri**

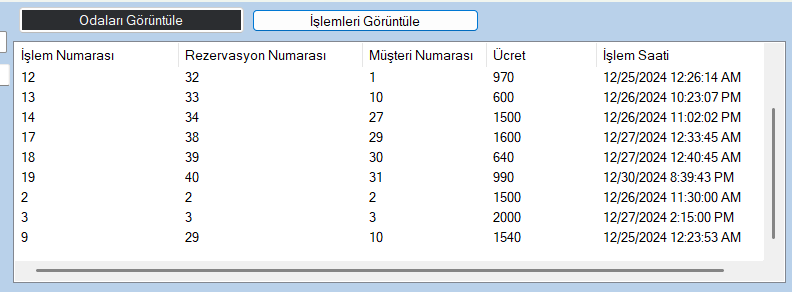


Öncelikle sağ taraftan seçilen tarih ile uygunluk kontrol et butonu check\_availablity fonksiyonu ile müsaitlik durumuna bakıyor ve daha sonra ücret hesaplanıyor.Ücret Hesaplanırken önce dinamik olarak yoğunluğa göre oda fiyatı hesaplanıyor örneğin burada oda 1in fiyatı 300 ve oda 1 in seçilen hafta yoğunluğu %10 dan fazla bu durumda oda fiyatı 1.55 ile çarpılıyor ve ortaya 930 rakamı çıkıyor ardından fee\_calculatordan odanın ekstra ücretlerinin fiyatı alınarak dinamik oda ücretiyle toplanıyor.

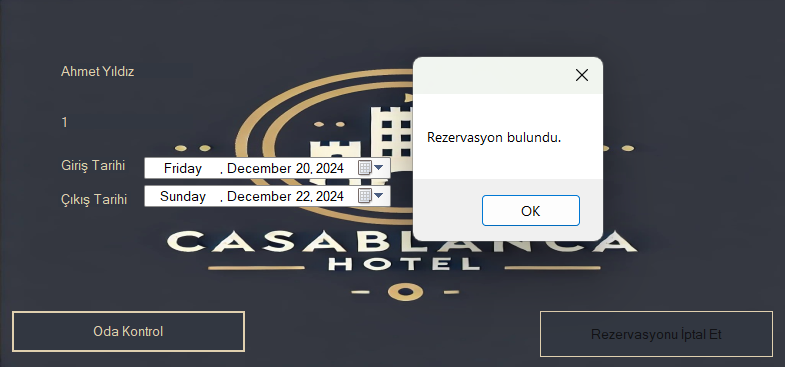


Arayüzün sağ üst tarafından admin panele giriş yapılıyor bu kısımda şifre SHA256 ile şifrelenerek veritabanına atıldı bu sayede bu panelin güvenliği sağlandı.SHA256 ile yazılan şifre:asdas41





Bu panelde admin kullanıcı bir günlük ciroyu toplam ciroyu,tüm odaların verilerini tüm transfer işlemlerini ve oda fiyatlarını görüntülerken sol alt taraftaki panel sayesinde ekstra ücretleri ve oda fiyatlarını güncelleyebiliyor



Bu panelde rezervasyon yapan müşteri ad soyad oda numarası ve giriş çıkış tarihlerini girerek önce rezervasyonların eşleşmesini kontrol ediyor ardından rezervasyonunu iptal ederek rezervasyonun verisinin veritabanından silinmesini sağlıyor.Ancak eğer müşterinin farklı bir rezervasyonu daha varsa customer table daki verileri ile diğer rezervasyonunun verileri silinmiyor.

1. **VERİ TABANI TASARIM AŞAMALARI**

**3.1 Kavramsal Tasarım**

**3.1.1 Temel Gereksinimler:**

1. Müşteri bilgileri tutulmalıdır (isim-soyisim gibi).
2. Oda bilgileri (oda numarası ve fiyat gibi) kayıt altında olmalıdır.
3. Rezervasyonlar ve bu rezervasyonların giriş-çıkış tarihleri kaydedilmelidir.Aynı tarihte aynı odaya rezervasyon yapılması engellenmelidir.
4. Odalar ve rezervasyonlara ait masraflar yönetilmelidir (örneğin, kahvaltı veya minibar ücretleri).
5. Finansal işlemler (müşteri ödemeleri gibi) kayıt altına alınmalıdır.
6. Sisteme yönetici girişi için gerekli bilgiler hashlenerek şifreleme yöntemiyle tutulmalıdır (kullanıcı adı ve şifre).
7. Odanın yoğunluk durumuna göre rezervasyon fiyatın dinamik hale gelmesi gerekmektedir.
8. Yönetici için günlük,toplam fiyat bilgileri,hangi odada kimin kaldığı,hangi tarihte kimin transfer işlemi yaptığı görüntülenebilir hale gelmelidir.

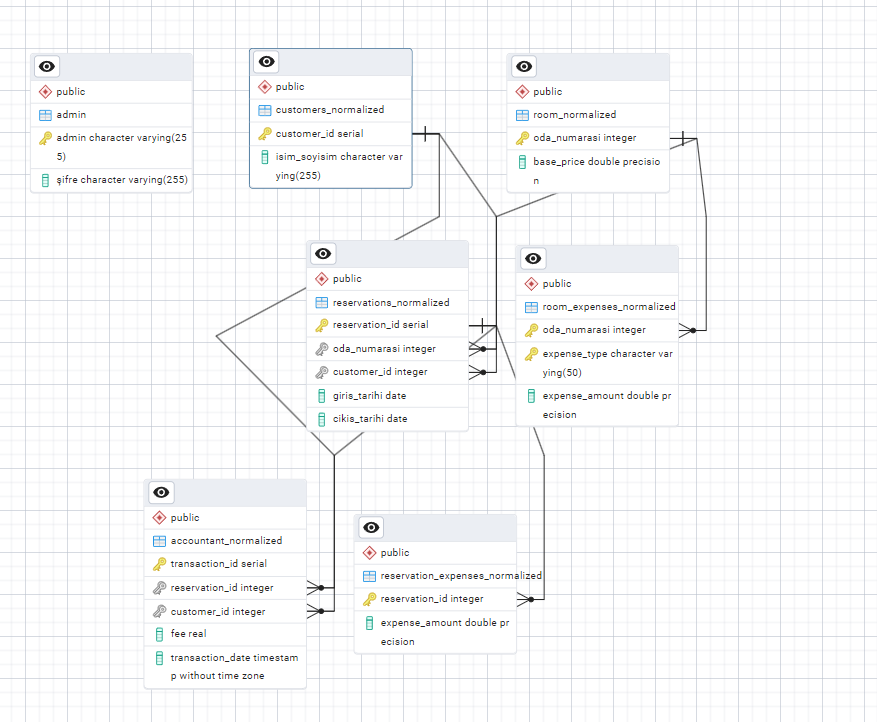
**3.1.2 Varlıklar ve Öznitelikler:**

|  |  |
| --- | --- |
| customers\_normalized (Müşteriler) | customer\_id, isim\_soyisim |
| room\_normalized (Odalar) | oda\_numarasi, base\_price |
| reservations\_normalized (Rezervasyonlar) | reservation\_id, oda\_numarasi, customer\_id, giris\_tarihi, cikis\_tarihi |
| reservation\_expenses\_normalized (Rezervasyon Masrafları) | reservation\_id, expense\_amount |
| room\_expenses\_normalized (Oda Masrafları) | oda\_numarasi, expense\_type, expense\_amount |
| accountant\_normalized (Finansal İşlemler) | transaction\_id, reservation\_id, customer\_id, fee, transaction\_date |
| admin (Yönetici) | |  | | --- | |  |  |  | | --- | | admin,şifre | |

**3.1.3 İlişkiler:**

|  |  |  |  |
| --- | --- | --- | --- |
| **customers\_normalized** | **rezervations\_normalized** | **1'e Çok** | **Bir müşteri birden fazla rezervasyon yapabilir.** |
| **room\_normalized** | **reservations\_normalized** | **1'e Çok** | **Bir oda farklı zamanlarda birden fazla rezervasyona konu olabilir.** |
| **reservations\_normalized** | **reservation\_expenses\_normalized** | **1'e Çok** | **Bir rezervasyona birden fazla masraf kaydedilebilir (örneğin, minibar,kahvaltı).** |
| **room\_normalized** | **room\_expenses\_normalized** | **1’e Çok** | **Bir odaya farklı masraf türleri uygulanabilir.** |
| **reservations\_normalized** | **accountant\_normalized** | **1'e 1 veya 1'e Çok** | **Bir rezervasyon bir veya birden fazla finansal işlemle ilişkilendirilebilir.** |
| **customers\_normalized** | **accountant\_normalized** | **1'e 1 veya 1'e Çok** | **Bir müşteri bir veya daha fazla finansal işlem gerçekleştirebilir (rezervasyon ödemeleri).** |

**3.1.4 ER Diyagramı:**

****

**3.2 Mantıksal Tasarım**

**3.2.1 Güçlü ve Zayıf Varlıklar:**

* **Güçlü Varlıklar:**
  + customers\_normalized, room\_normalized, reservations\_normalized
* **Zayıf Varlıklar:**
  + reservation\_expenses\_normalized, room\_expenses\_normalized (diğer tablolara bağımlıdır).

**3.2.2 İlişki Tipleri:**

1.customers\_normalized ile reservations\_normalized (1'e Çok İlişki)

* Açıklama:  
  Bir müşteri (tek bir customer\_id) birden fazla rezervasyon yapabilir, ancak her rezervasyon bir müşteriyle ilişkilendirilir.
* reservations\_normalized tablosundaki customer\_id sütunu, customers\_normalized tablosundaki customer\_id ile bir **yabancı anahtar** ilişkisine sahiptir.

2. room\_normalized ile reservations\_normalized (1'e Çok İlişki)

* Açıklama:  
  Her oda (oda\_numarasi), birden fazla rezervasyonda yer alabilir, ancak her rezervasyon bir odaya bağlanır.
* reservations\_normalized tablosundaki oda\_numarasi sütunu, room\_normalized tablosundaki oda\_numarasi ile bir **yabancı anahtar** ilişkisine sahiptir.

3. reservations\_normalized ile reservation\_expenses\_normalized (1'e Çok İlişki)

* Açıklama:  
  Bir rezervasyon (reservation\_id), bir veya birden fazla masraf kalemiyle ilişkilendirilebilir. Ancak her masraf kalemi yalnızca bir rezervasyona bağlıdır.
* reservation\_expenses\_normalized tablosundaki reservation\_id sütunu, reservations\_normalized tablosundaki reservation\_id ile bir **yabancı anahtar** ilişkisine sahiptir.

4. room\_normalized ile room\_expenses\_normalized (1'e Çok İlişki)

* Açıklama:  
  Her oda (oda\_numarasi), bir veya birden fazla masraf türüne sahip olabilir (örneğin, "temizlik", "onarım"). Ancak her masraf kaydı yalnızca bir odaya bağlıdır.
* room\_expenses\_normalized tablosundaki oda\_numarasi sütunu, room\_normalized tablosundaki oda\_numarasi ile bir yabancı anahtar ilişkisine sahiptir.

5. accountant\_normalized ile reservations\_normalized ve customers\_normalized (Çoklu İlişkiler)

* Açıklama:
  + Rezervasyon İlişkisi: Bir finansal işlem (transaction\_id), belirli bir rezervasyonla ilişkilendirilebilir.
  + Müşteri İlişkisi: Aynı zamanda bu işlem bir müşteriyle ilişkilendirilir.
  + accountant\_normalized tablosundaki reservation\_id sütunu, reservations\_normalized tablosundaki reservation\_id ile bir yabancı anahtar ilişkisine sahiptir.
  + accountant\_normalized tablosundaki customer\_id sütunu, customers\_normalized tablosundaki customer\_id ile bir yabancı anahtar ilişkisine sahiptir.

**3.2.3 Normalizasyon**

Normalizasyon öncesi:

**CREATE TABLE IF NOT EXISTS room\_expenses** (

oda\_numarasi INT PRIMARY KEY,

base\_price DOUBLE PRECISION,

children\_fee DOUBLE PRECISION,

breakfast\_fee DOUBLE PRECISION,

minibar\_fee DOUBLE PRECISION,

room\_service\_fee DOUBLE PRECISION);

**CREATE TABLE IF NOT EXISTS musteriler**(

no SERIAL,

oda\_numarasi INT REFERENCES room\_expenses(oda\_numarasi),

isim\_soyisim VARCHAR(255),

ucret REAL,

giris\_tarihi DATE,

cikis\_tarihi DATE);

**CREATE TABLE IF NOT EXISTS ACCOUNTANT**(

NAME VARCHAR(255) NOT NULL,

FEE REAL NOT NULL,

TRANSACTION\_DATE TIMESTAMP NOT NULL);

**CREATE TABLE IF NOT EXISTS admin**(

admin VARCHAR(255) NOT NULL PRIMARY KEY,

sifre VARCHAR(255) NOT NULL,);

**Normalizasyon sonrası:**

**room\_expenses** tablosunda her oda için kahvaltı ücreti, minibar ücreti gibi farklı ücret

türleri ayrı sütunlarda tutulmaktadır.Tek bir satırda birçok ücret detayının saklanması, veri

tekrarına yol açıyor ve NULL değerler oluşmasına neden oluyor.

• Bu, yeni bir ücret türü eklemek gerektiğinde tabloya sütun eklemeyi zorunlu hale getirir ve

**1NF'ye uygun değildir**.

**Her bir ücret türü bir sütun yerine ayrı bir satırda saklanmalıdır**.

**CREATE TABLE room\_normalized** (

oda\_numarasi INT PRIMARY KEY,

base\_price DOUBLE PRECISION NOT NULL);

**CREATE TABLE IF NOT EXISTS room\_expenses\_normalized** (

oda\_numarasi INT NOT NULL references room\_normalized(oda\_numarasi),

expense\_type VARCHAR(50) NOT NULL,

expense\_amount DOUBLE PRECISION NOT NULL,

PRIMARY KEY (oda\_numarasi, expense\_type));

Bu tabloda yer alan her satır bize oda\_numarasi ve expense\_type’ın benzersiz bir

kombinasyonunu anlattığı için PRIMARY KEY oda\_numarasi ve expense\_type

kombinasyonudur. Bu şekilde tam bağımlı ve 2NF’ye uygun hale gelir. Geçişli bağımlılık

olmadığı için **3NF’ye de uygundur.**

Müşteriler tablosunda oda numarası ve müşteri bilgileri birleştirilmiş durumda. Bu, müşteri ve

odalar arasında birebir bir ilişki kuruyor gibi görünüyor. Ancak bir müşteri farklı tarihlerde aynı

odada kalabilir, bu da veritabanında gereksiz tekrarlar yaratabilir ve **1NF’ye uygun değildir.**

• musteriler tablosundan müşteri bilgileri ayrı bir tabloya taşınmalıdır.

**CREATE TABLE IF NOT EXISTS customers\_normalized** (

customer\_id SERIAL PRIMARY KEY,

isim\_soyisim VARCHAR(255) NOT NULL );

**CREATE TABLE IF NOT EXISTS reservations\_normalized**(

reservation\_id SERIAL PRIMARY KEY,

oda\_numarasi INT NOT NULL REFERENCES room(oda\_numarasi),

customer\_id INT NOT NULL REFERENCES customers(customer\_id),

ucret REAL NOT NULL,

giris\_tarihi DATE NOT NULL,

cikis\_tarihi DATE NOT NULL);

**reservation\_id** birincil anahtardır ve tüm diğer sütunlar (oda\_numarasi, customer\_id,

ucret, giris\_tarihi, cikis\_tarihi) doğrudan bu anahtara bağlıdır bu da tablomuzu

2NF’ye uygun hale getirmiştir. Hiçbir sütun geçişli bağımlılık göstermez. Örneğin:

• oda\_numarasi veya customer\_id başka bir sütuna bağımlı değil, doğrudan

reservation\_id ile ilişkilidir. Dolayısıyla geçişli bağımlılık yoktur, **bu tablo 3NF’ye**

**uygundur.**

Accountant tablosunda muhasebe işlemleri, herhangi bir rezervasyon veya müşteriyle

ilişkilendirilmemiştir. Bu eksiklik raporlama ve veri tutarlılığı açısından sorunlara yol açabilir.

**CREATE TABLE IF NOT EXISTS accountant\_normalized** (

transaction\_id SERIAL PRIMARY KEY,

reservation\_id INT REFERENCES reservations(reservation\_id),

customer\_id INT REFERENCES(customer\_id),

fee REAL NOT NULL,

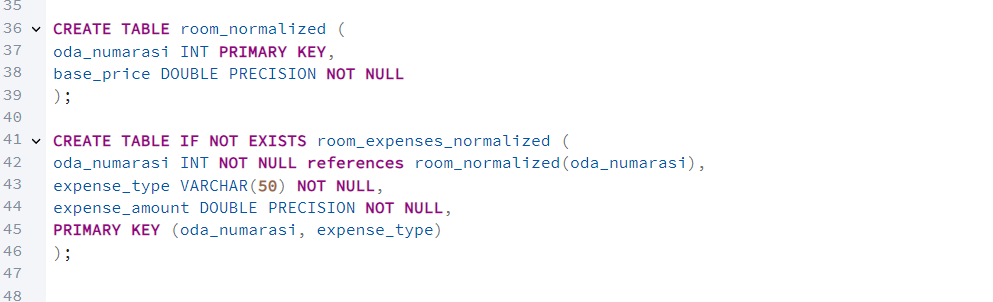
transaction\_date TIMESTAMP NOT NULL);

**Bu tasarım, normalizasyon kurallarına uygun, tutarlı ve esnek bir veritabanı sağlar.**

**3.3 FİZİKSEL TASARIM**

**3.3.1 TABLOLARIN HAZIRLANMASI**



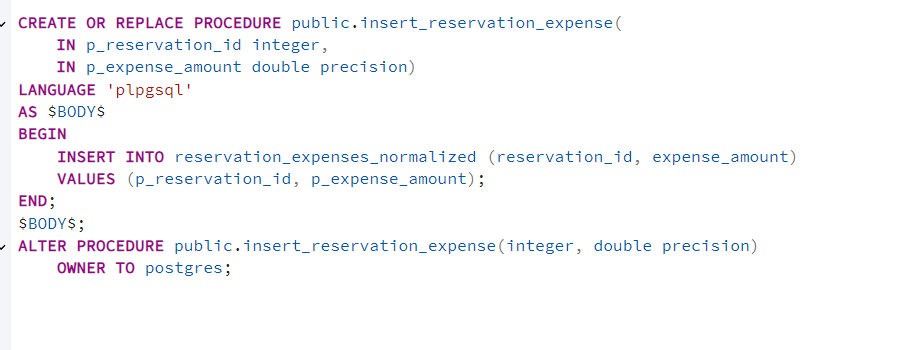


**3.3.2 PROSEDÜRLERİN HAZIRLANMASI**

Müşteri checkout işlemi yaptıktan sonra accountant log tablosundan müşteri bilgilerini silme işlemi



Rezervasyon masraflarını rezervasyon işlemi sırasında tabloya yazdırma işlemi

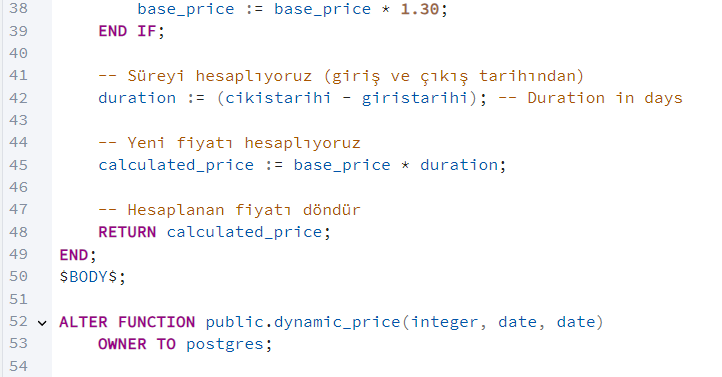
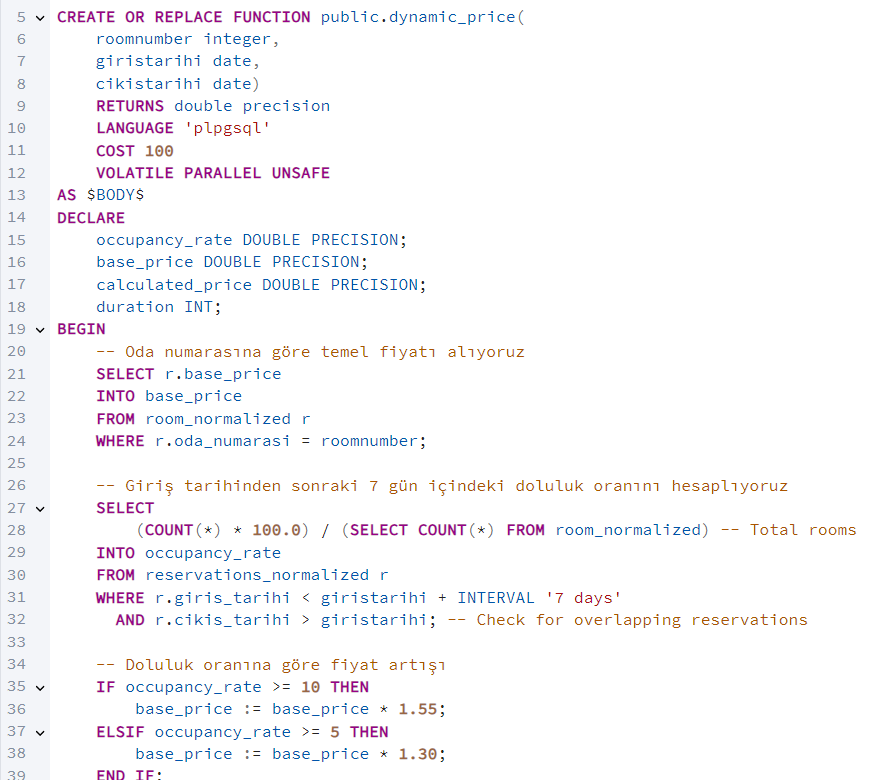


**3.3.3 FONKSİYONLARIN HAZIRLANMASI**

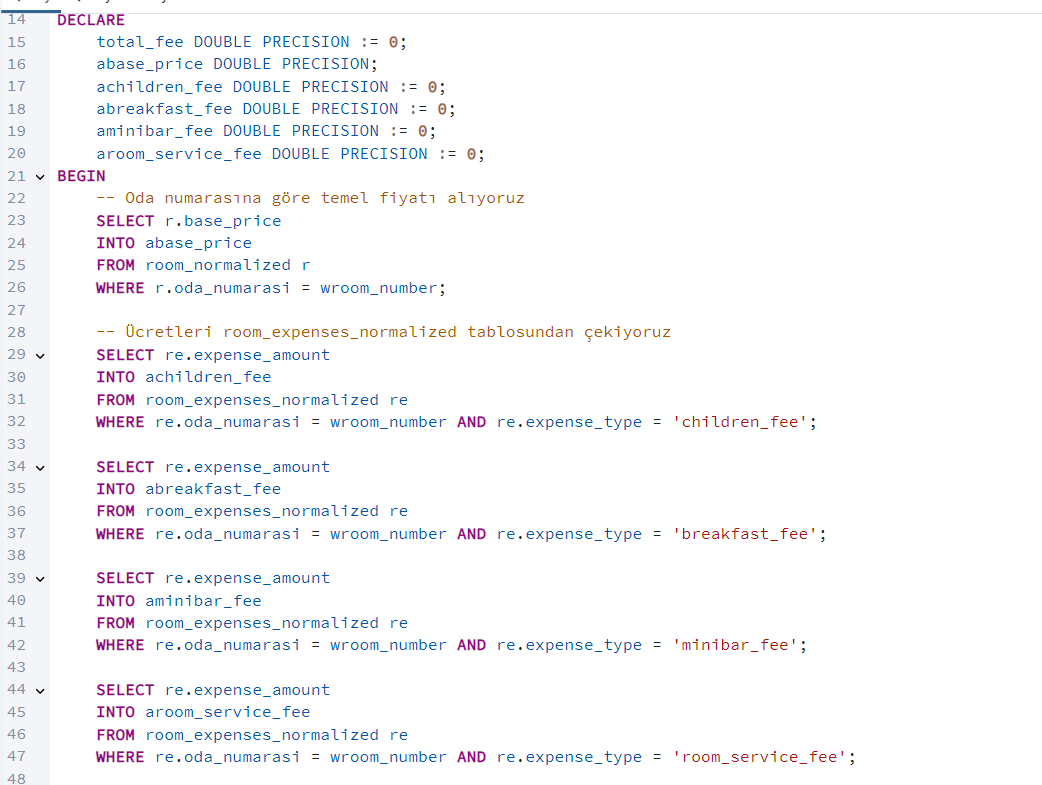
1-Odanın dolu olup olmadığını buna göre rezervasyona izin verilip verilmeyeceğini denetleyen fonksiyon

****

2-Otelin müşterinin seçtiği haftaya göre doluluk oranını hesaplayan ve oda fiyatını bu orana göre dinamik olarak güncelleyen fonksiyon

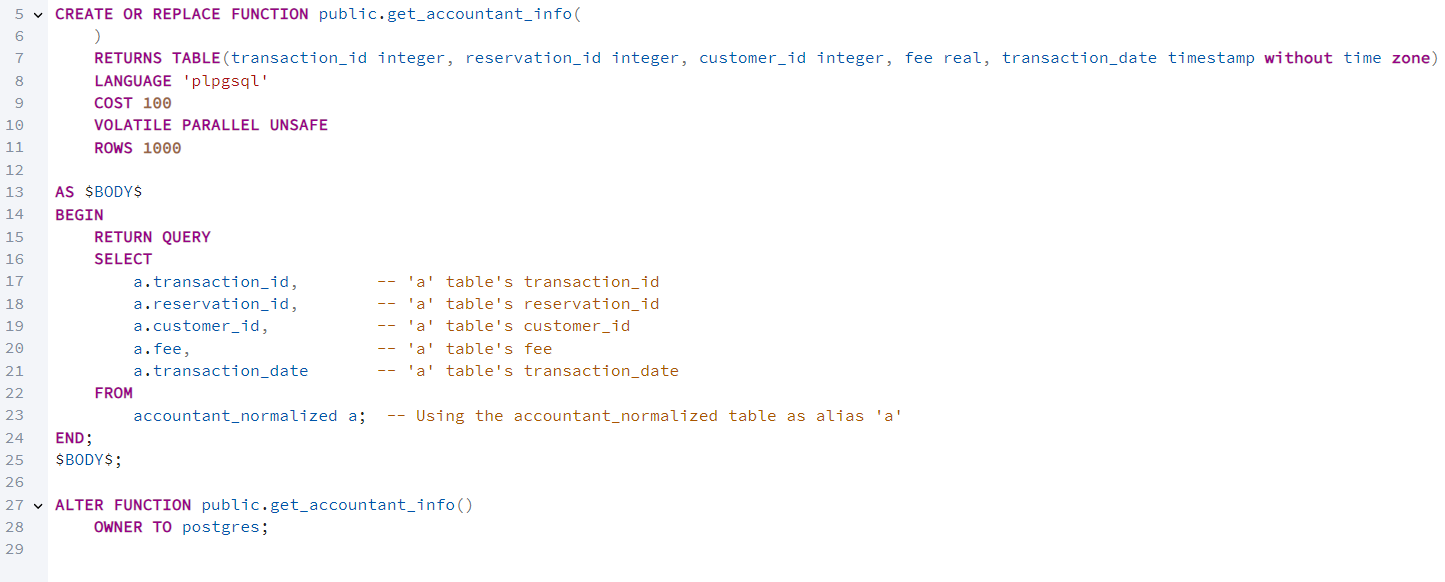
****

3-Ayrı tabloda tutulan ekstra masrafları uygulamadaki checkboxların seçilip seçilmeme durumuna göre hesaplayıp önceden hesaplanan dinamik ücrete ekleyen fonksiyon

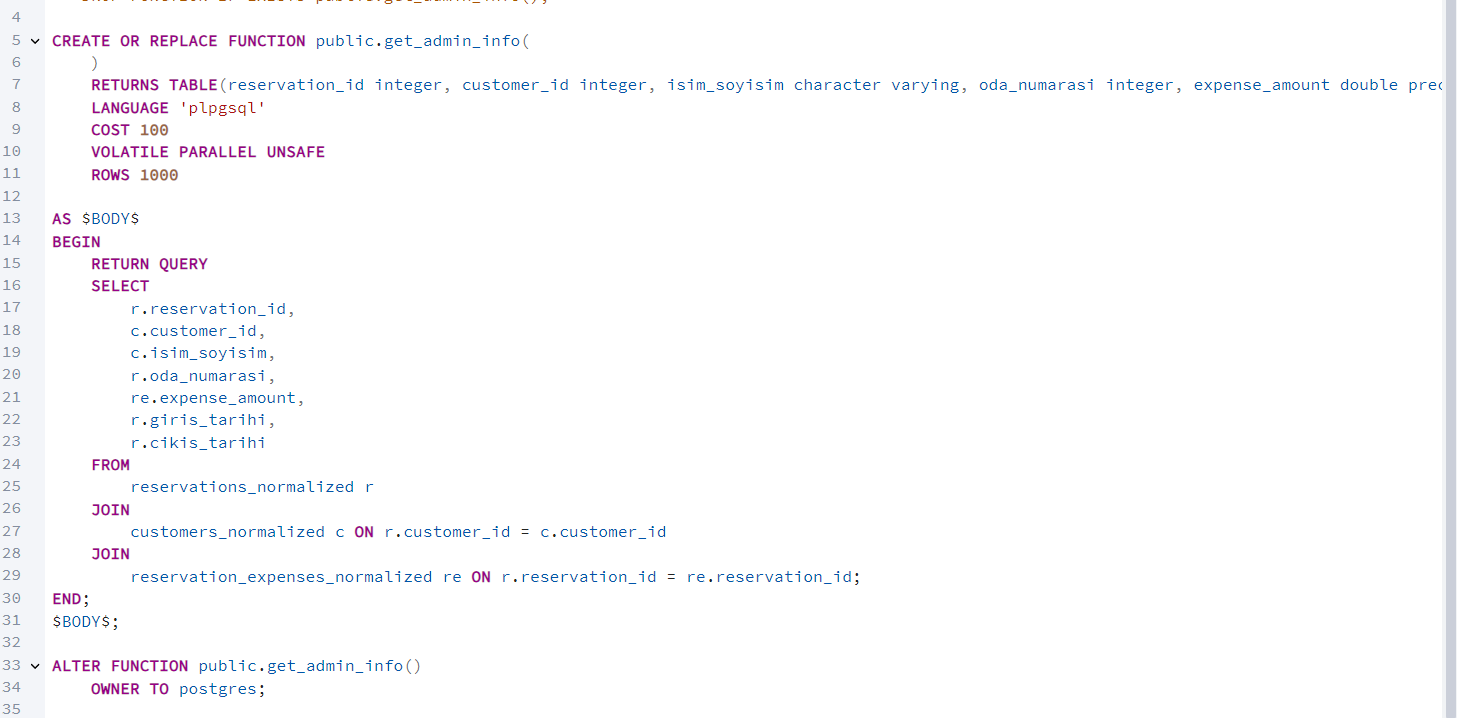




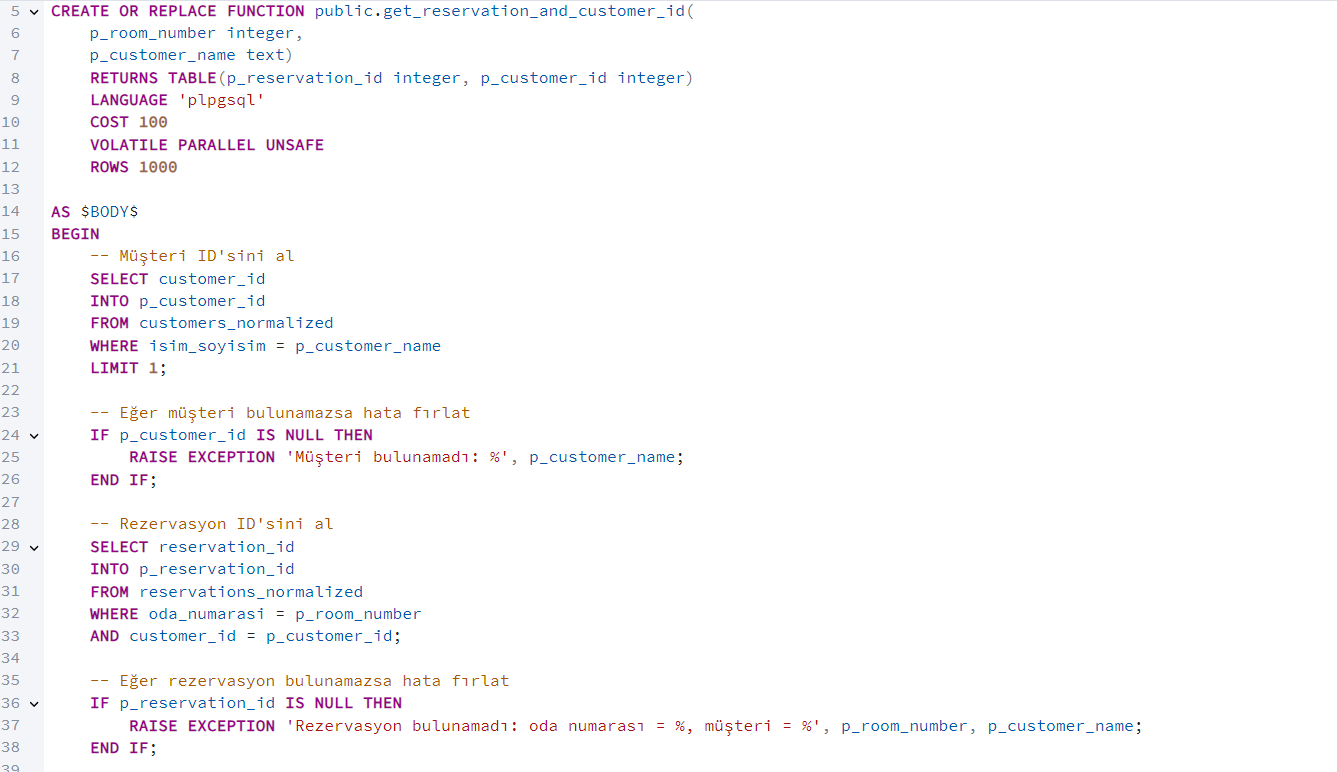
4-Accountant tablosundaki verileri çekerek uygulamadaki admin panele bu bilgileri yansıtan fonksiyon



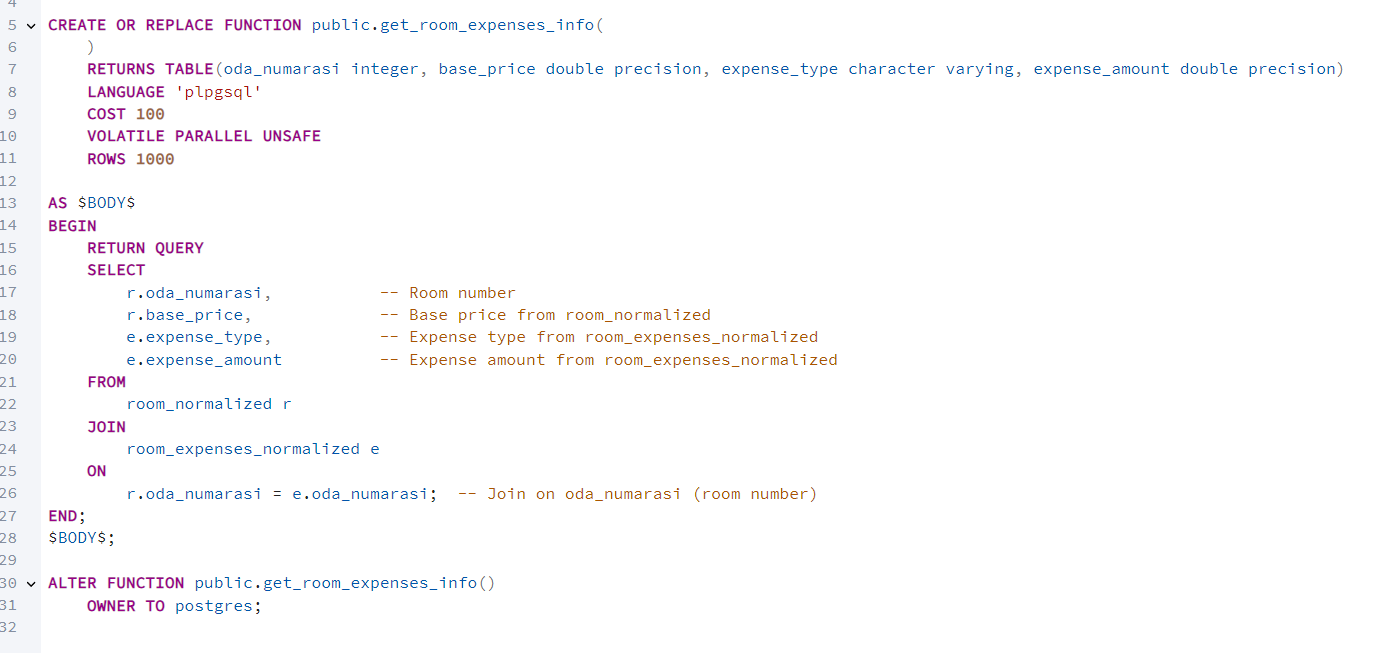
5-Müşteri bilgileri içeren normalize edilmiş tablolardaki tüm verileri birleştirerek uygulamadaki admin panele bu bilgileri yansıtan fonksiyon



6-Rezervasyon iptali işleminde kullanıcının girdiği bilgilerle veritabanında eşleşen bir rezervasyonun olup olmadığını kontrol eden fonksiyon



7-Admin panelde oda fiyatlarını görüntülemeyi sağlayan fonksiyon



8-Müşterinin rezervasyon bilgilerini rezervasyon tablosuna aktaran fonksiyon



**3.3.3 TRIGGERLARIN HAZIRLANMASI**

Müşteri rezervasyon yaptıktan sonra müşterinin bilgileriyle birlikte rezervasyonu gerçekleştiği tarih ve saati accountant tablosuna yazdıran fonksiyon



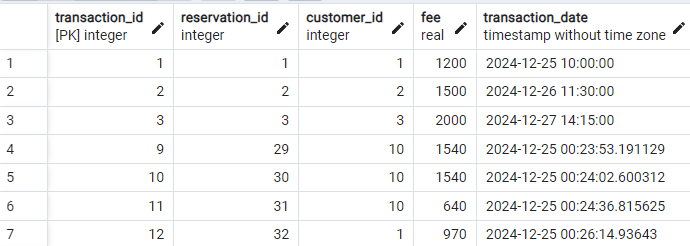
1. **GÜNCEL VERİTABANI VE UML DİYAGRAMI**

Admin tablosu

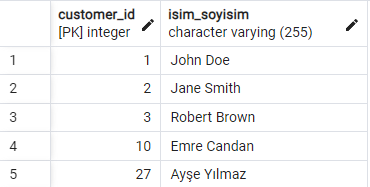
metin, yazı tipi, çizgi, sayı, numara içeren bir resim

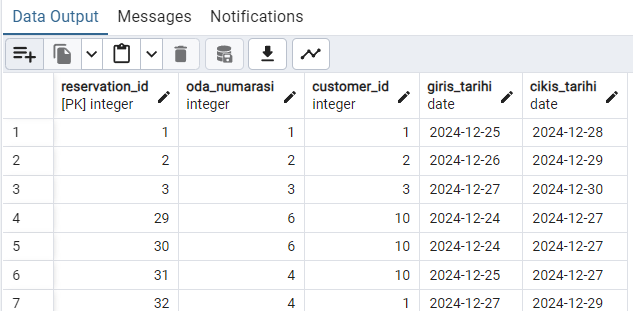
Açıklama otomatik olarak oluşturuldu

Accountant\_normalized tablosu

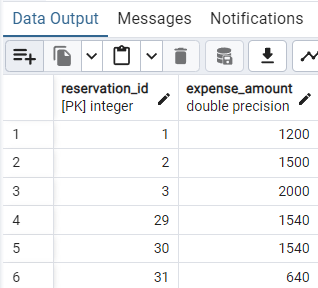
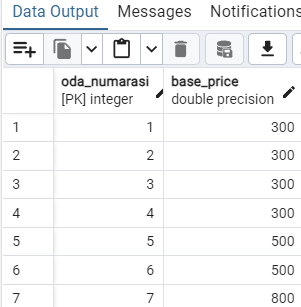


customer\_normalized tablosu

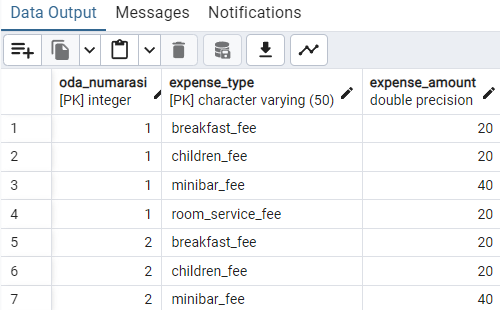


Reservation\_normalized tablosu

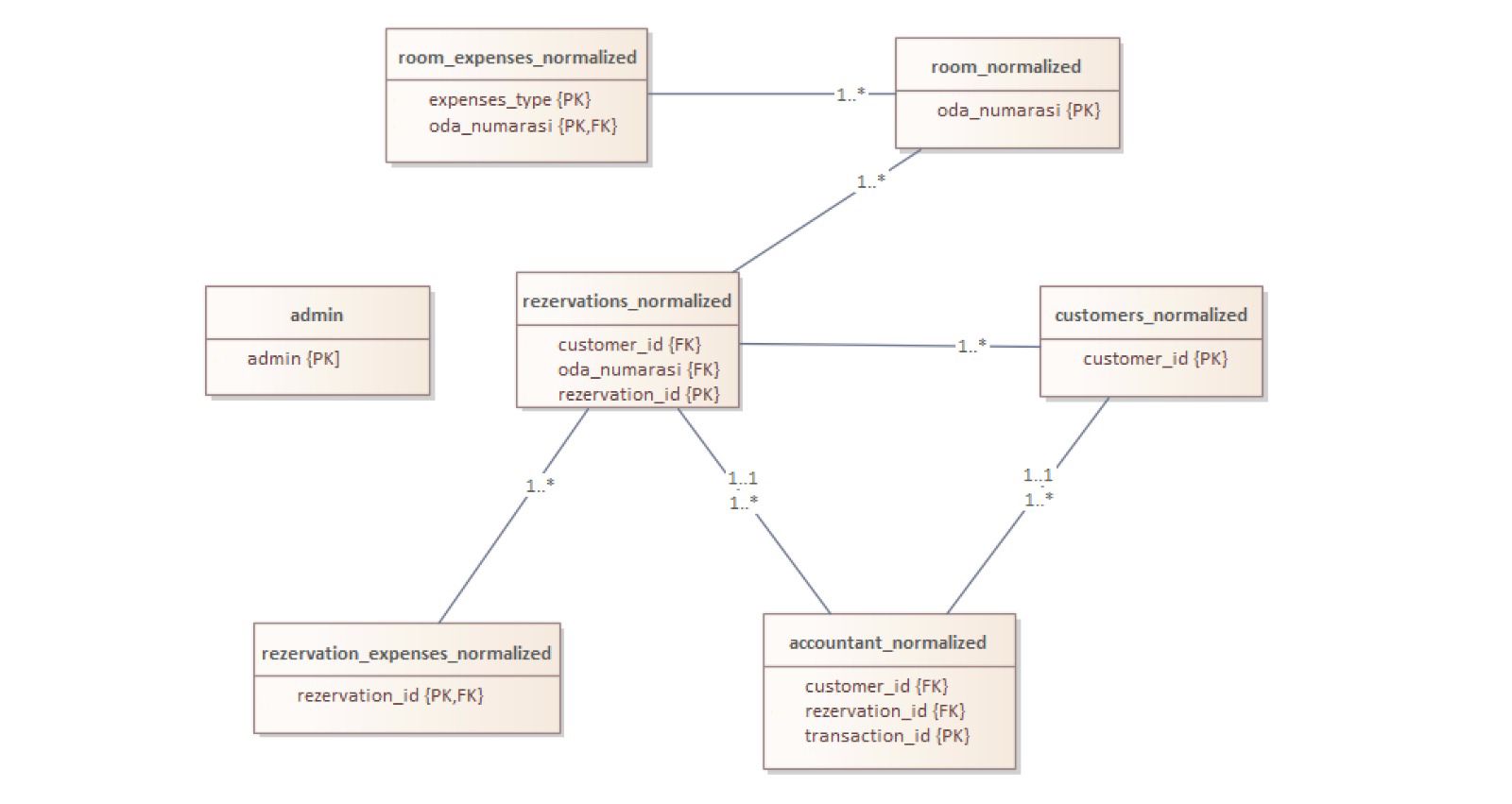
Reservation\_expenses\_normalized tablosu room\_normalized tablosu

Room\_Expenses\_normalized tablosu



**UML DİYAGRAMI**



1. **SONUÇ**

Projemiz oldukça başarılı olup hedefine ulaştı.

Hali hazırda bulunan projelerden artı yönleri dinamik fiyatlandırma ve çoğu fonksiyonel işlemin veritabanında yapılması bu sayede de hızlı olması söylenebilir.

Eksi yön olarak da projemizde herhangi bir ödeme ekranı bulunmuyor.Rezervasyon yaptır butonuna tıklandıktan sonra bir ödeme ekranı eklenebilir.

Projeyi yaparken kendimizi veritabanı alanında oldukça ilerlettiğimizi düşünüyoruz.Bunun dışında arayüz tasarımı ve veritabanı uygulama ilişkisini iyi bir şekilde kavradığımızı düşünüyoruz.

1. **KAYNAKÇA**

<https://www.postgresql.org/docs/>

<https://www.geeksforgeeks.org/>

<https://www.w3schools.com/>

<https://stackoverflow.com/>