

# VERİTABANI 2 BONUS ÖDEVİ RAPORU

## GENEL BİLGİ

**Veritabanı Adı:** `flightcontrol1`

**Amaç:** Bu proje, PostgreSQL kullanarak üç ana varlık (yolcular, uçuşlar, rezervasyonlar) üzerine kurulmuş bir Uçak Bilet Rezervasyon Sistemi tasarlamak, sorgular geliştirmek, tetikleyiciler, fonksiyonlar, prosedürler ve güvenlik yapılarını uygulayarak Veritabanı 2 dersinde işlenmiş tüm kavramları pratikte kullanabilmektir. Rapor sonunda örnek görseller ve kişisel kullanım için rehber adım mevcuttur.

## VARLIKLAR VE İLİŞKİLER

### 1. `passengers`

- Yolcu bilgilerini tutar.
- `passenger_id` PRIMARY KEY.

### 2. `flights`

- Uçuş bilgilerini tutar.
- `flight_id` PRIMARY KEY.

### 3. bookings

- Yolcu ile uçuş arasındaki ilişkiyi kurar.
- `flight_id` ve `passenger_id` FOREIGN KEY.
- `booking_id` PRIMARY KEY.

### 4. booking\_log

- Trigger tarafından beslenen log tablosu.

## VERİ OLUŞTURMA

### Tablo Oluşturma:

Projede üç temel tabloya ek olarak bir log tablosu oluşturulmuştur.

### Veri Ekleme:

- **8 Yolcu** ve **6 Uçuş** eklenmiştir.
- Çeşitli yolcular için **10'dan fazla rezervasyon** kaydı girilmiştir.

## SQL SORGULARI (TOPLAM 33+)

### 1. Basit SELECT Sorguları

- Tüm yolcuları görüntüleme
- Fiyat, kalkış yeri, doğum tarihi şartlarıyla filtreleme

### 2. JOIN İçeren Sorgular

- Uçuş ve yolcu bilgilerini rezervasyonlar ile birleştirme

- Gruplama ile sayım ve toplam gelir

### 3. ORDER BY & LIMIT

- En pahalı uçuşlar, en eski rezervasyonlar

### 4. UPDATE / DELETE

- Belirli rezervasyonların statüsünü değiştirme
- Yolcu silme işlemleri

### 5. VIEW Kullanımı

- `passenger_booking_summary` adlı VIEW ile rezervasyon özetleri oluşturulmuştur.
- View üzerinden Business yolcular ve belirli rotalar filtrelenmiştir.

### 6. FUNCTION

- `flight_duration(f_id)` fonksiyonu ile bir uçuşun toplam süre farkı hesaplanmıştır.

### 7. PROCEDURE

- `update_booking_status()` prosedürü ile rezervasyon statüsü değiştirilmiştir.

### 8. TRIGGER

- `log_new_booking()` fonksiyonu ve `trg_booking_insert` trigger'ı ile yeni rezervasyonlar `booking_log` tablosuna otomatik kaydedilmiştir.

### 9. TRANSACTION

- Birden fazla rezervasyon INSERT işlemi `BEGIN...COMMIT` bloğu ile gerçekleştirilmiştir.

### 10. SECURITY (Yetkilendirme)

- `readonly_user` rolü oluşturulmuş, yalnızca SELECT yetkisi verilmiştir.

## 11. Bonus Sorgular

- `LIKE`, `BETWEEN`, `IN`, `CAST`, `HAVING` gibi özelliklerle filtreleme ve analiz sorguları yapılmıştır.

## TRIGGER VE VIEW KULLANIMINA ÖRNEK

### Trigger ile Log Kontrolü:

- Yeni bir rezervasyon eklendiğinde log kaydı otomatik yazılmıştır.
- Bu loglar tarih sırasına göre listelenerek denetlenmiştir.

### View ile Raporlama:

- "Business" sınıfı yolcuları ve belli rota sorguları `passenger_booking_summary` üzerinden çok daha okunabilir biçimde sorgulanmıştır.

## ÖRNEK SORGULAR

```
-- 28. View üzerinden tüm rezervasyon özetlerini çek
SELECT * FROM passenger_booking_summary;
```

first_name	last_name	seat_no	travel_class	departure_airport	arrival_airport
Ahmet	Yılmaz	12A	Economy	IST	ANK
Mehmet	Kaya	8C	Economy	SAW	ADB
Zeynep	Şahin	7D	Economy	ADB	IST
Ali	Çelik	1A	Business	ESB	SAW
Elif	Aydın	9E	Economy	IST	AYT
Burak	Koç	10F	Economy	IST	ANK
Fatma	Bulut	2B	Business	IST	ESB
Ahmet	Yılmaz	6A	Economy	SAW	ADB
Ayşe	Demir	3C	Economy	ADB	IST
Ayşe	Demir	5B	Business	IST	ESB
Mehmet	Kaya	6F	Economy	ESB	SAW
Zeynep	Şahin	9D	Economy	IST	AYT
Zeynep	Şahin	11C	Economy	SAW	ADB

X

```
-- 29. Sadece "Business" sınıfında seyahat eden yolcular
```

first_name	last_name	seat_no	travel_class	departure_airport	arrival_airport
Fatma	Bulut	2B	Business	IST	ESB
Ayşe	Demir	5B	Business	IST	ESB
Ali	Çelik	1A	Business	ESB	SAW

132 -- 9. Her uçuşun toplam gelirini hesapla

Data Output Messages Notifications

≡+ 📄 ▼ 📋 ▼ 🗑️ 🗄️ ⬇️ 📈 SQL

	flight_id [PK] integer	total_revenue numeric
1	3	1799.70
2	5	1400.00
3	4	1300.00
4	6	1450.00
5	2	1719.00
6	1	1599.98

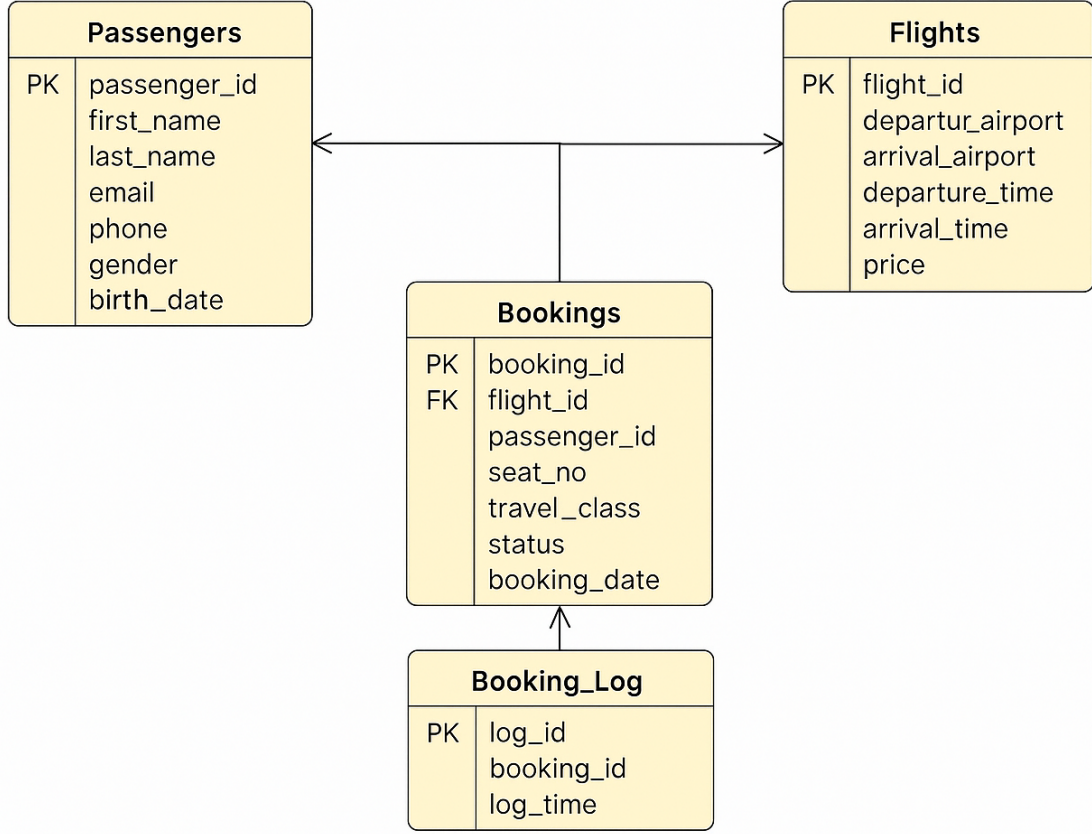
-- 6. Tüm rezervasyonları yolcu ve uçuş bilgileriyle birlikte getir

Output Messages Notifications

📄 ▼ 📋 ▼ 🗑️ 🗄️ ⬇️ 📈 SQL

booking_id integer	passenger text	route text	departure_time timestamp without time zone	seat_no character varying (5)	travel_class character varying (20)
1	Ahmet Yilm...	IST → ANK	2025-06-01 08:00:00	12A	Economy
3	Mehmet Ka...	SAW → ADB	2025-06-02 12:00:00	8C	Economy
4	Zeynep Şahin	ADB → IST	2025-06-03 17:00:00	7D	Economy
5	Ali Çelik	ESB → SAW	2025-06-04 20:00:00	1A	Business
6	Elif Aydın	IST → AYT	2025-06-05 09:00:00	9E	Economy
7	Burak Koç	IST → ANK	2025-06-01 08:00:00	10F	Economy
8	Fatma Bulut	IST → ESB	2025-06-01 15:00:00	2B	Business
9	Ahmet Yilm...	SAW → ADB	2025-06-02 12:00:00	6A	Economy
10	Ayşe Demir	ADB → IST	2025-06-03 17:00:00	3C	Economy
2	Ayşe Demir	IST → ESB	2025-06-01 15:00:00	5B	Business
11	Mehmet Ka...	ESB → SAW	2025-06-04 20:00:00	6F	Economy
12	Zeynep Şahin	IST → AYT	2025-06-05 09:00:00	9D	Economy
13	Zeynep Şahin	SAW → ADB	2025-06-02 12:00:00	11C	Economy

ER DIYAGRAM ORNEĞİ



## SİZ DE QUERY'LERİ KULLANARAK KENDİ VERİTABANINIZA EKLEYEBİLİRSİNİZ

Bu proje kapsamında kullanılan tüm SQL sorguları ve veritabanı yapıları, PostgreSQL tabanlı bir sistemde doğrudan çalıştırılabilir biçimdedir. Eğer siz de bu örneği kendi bilgisayarınızda veya okul bilgisayarlarında denemek isterseniz, aşağıdaki adımları takip ederek kısa sürede uçak bilet rezervasyon sisteminizi kurabilirsiniz:

# 1. pgAdmin veya PostgreSQL Arayüzüne Erişim Sağlayın

- Bilgisayarınızda **pgAdmin 4** veya başka bir PostgreSQL arayüzü kurulu olmalıdır.
- Arayüzden **Query Tool**'u açarak komutları yapıştırabileceğiniz editöre ulaşın.

## 2. Yeni Veritabanı Oluşturun


Aşağıdaki komutu kullanarak sıfırdan yeni bir veritabanı yaratabilirsiniz:

```
CREATE DATABASE flightcontrol1 TEMPLATE template0;
```

-Alternatif olarak pgAdmin üzerinden GUI yoluyla da veritabanı oluşturabilirsiniz.

## 3. SQL Script'ini Tek Parça Halinde Yapıştırın

Bu projeye ait SQL script tüm tablo oluşturma, veri ekleme, sorgulama, VIEW, TRIGGER, FUNCTION ve diğer yapılandırmaları **tek bir blok** halinde içermektedir. Yapmanız gereken:

- Hazırlanan SQL dosyasını veya .txt içeriğini Query Tool'a yapıştırmak
- Ardından **"Execute/Çalıştır"** () butonuna basmak

## 4. Her Şey Hazır!

Script çalıştıktan sonra:

- Tüm tablolar ve örnek veriler oluşur.
- Fonksiyonlar ve prosedürler tanımlanır.



- Otomatik loglama ve güvenlik kontrolleri hazır hale gelir.
- Bonus olarak VIEW ve TRIGGER kullanım örneklerini test edebilirsiniz.

Öneri: Script'teki tablo adları, sütunlar ve veriler İngilizce yapıda, ancak yolcu bilgileri Türkçe olarak düzenlenmiştir.

- Bu sistemi daha da geliştirmek isteyenler, ek tablolar (havayolu şirketi, terminal, kalkış kapısı vb.) ekleyerek genişletebilir.

Kendi sorgularınızı test ederken veri bütünlüğünü bozmadığınızdan emin olun. Hataları geri almak için ROLLBACK veya DELETE komutlarını bilinçli kullanın.

Hazırlanan bu proje, veritabanı öğreniminizi pekiştirmeniz için birebir gerçek dünyaya yakın bir örnektir. Kendi veritabanınızı kurarak deneyiminizi artırabilirsiniz.

## SONUÇ VE YORUM

Bu proje, Veritabanı 2 dersinde teorik olarak görülen tüm konuların uygulamaya dökülmüş halidir. Hem çekirdek SQL sorguları hem de ileri seviye yapılar (trigger, view, procedure vb.) kullanılarak veritabanı sisteminin bütünsel çalışması sağlanmıştır.

Sistem, gerçek hayatta bir havayolu rezervasyon altyapısı olarak şekillenebilir ve daha fazla tablo ile geliştirilebilir (hava yolları, kalkış kapıları, bagaj takibi, vb.).

Toplam 33+ sorgu ve yapı işlemi içerir. Tüm gereksinimler eksiksiz olarak uygulanmıştır.

Hazırlayan: **1247008055 Osman Yetkin.**

Ders: Veritabanı 2

Tarih: Mayıs 2025