--Kullanıcılar

CREATE TABLE Kullanici (

kullanici\_id SERIAL PRIMARY KEY,

kullanici\_adi VARCHAR(100) NOT NULL UNIQUE,

email VARCHAR(100) NOT NULL UNIQUE,

sifre VARCHAR(255) NOT NULL,

favori\_oyunlar TEXT

);

-- Oyunlar

CREATE TABLE Oyunlar (

oyun\_id SERIAL PRIMARY KEY,

oyun\_adi VARCHAR(100) NOT NULL UNIQUE

);

-- Takımlar

CREATE TABLE Takimlar (

takim\_id SERIAL PRIMARY KEY,

takim\_adi VARCHAR(100) NOT NULL,

uyeler TEXT,

kaptan VARCHAR(100)

);

select uyeler from Takimlar

-- Turnuvalar

CREATE TABLE Turnuvalar (

turnuva\_id SERIAL PRIMARY KEY,

turnuva\_adi VARCHAR(100) NOT NULL,

turu VARCHAR(50),

tarih DATE,

aciklama TEXT

);

-- Turnuva Katılım

CREATE TABLE TurnuvaKatilim (

katilim\_id SERIAL PRIMARY KEY,

kullanici\_id INT NOT NULL,

turnuva\_id INT NOT NULL,

takim\_id INT,

bireysel BOOLEAN DEFAULT FALSE,

FOREIGN KEY (kullanici\_id) REFERENCES Kullanici(kullanici\_id),

FOREIGN KEY (turnuva\_id) REFERENCES Turnuvalar(turnuva\_id),

FOREIGN KEY (takim\_id) REFERENCES Takimlar(takim\_id)

);

-- Turnuva Ödülleri

CREATE TABLE TurnuvaOdulleri (

odul\_id SERIAL PRIMARY KEY,

turnuva\_id INT NOT NULL,

odul\_miktari INT NOT NULL,

odul\_turu VARCHAR(100),

FOREIGN KEY (turnuva\_id) REFERENCES Turnuvalar(turnuva\_id)

);

-- Turnuva Sonuçları

CREATE TABLE TurnuvaSonuclari (

turnuva\_sonuc\_id SERIAL PRIMARY KEY,

turnuva\_id INT NOT NULL,

takim\_id INT,

skor INT,

derece INT,

FOREIGN KEY (turnuva\_id) REFERENCES Turnuvalar(turnuva\_id),

FOREIGN KEY (takim\_id) REFERENCES Takimlar(takim\_id)

);

-- Maçlar

CREATE TABLE Maclar (

mac\_id SERIAL PRIMARY KEY,

takim1\_id INT NOT NULL,

takim2\_id INT NOT NULL,

tarih DATE,

saat TIME,

harita VARCHAR(100),

turnuva\_id INT,

FOREIGN KEY (takim1\_id) REFERENCES Takimlar(takim\_id),

FOREIGN KEY (takim2\_id) REFERENCES Takimlar(takim\_id),

FOREIGN KEY (turnuva\_id) REFERENCES Turnuvalar(turnuva\_id)

);

-- İstatistikler

CREATE TABLE Istatistikler (

istatistik\_id SERIAL PRIMARY KEY,

mac\_id INT NOT NULL,

takim\_id INT NOT NULL,

skor INT,

kazanan BOOLEAN,

FOREIGN KEY (mac\_id) REFERENCES Maclar(mac\_id),

FOREIGN KEY (takim\_id) REFERENCES Takimlar(takim\_id)

);

-- İstatistik Log

CREATE TABLE IstatistikLog (

log\_id SERIAL PRIMARY KEY,

mac\_id INT,

takim\_id INT,

islem\_zamani TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Log Fonksiyonu

CREATE OR REPLACE FUNCTION IstatistikEkleLog()

RETURNS TRIGGER AS $$

BEGIN

INSERT INTO IstatistikLog (mac\_id, takim\_id)

VALUES (NEW.mac\_id, NEW.takim\_id);

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Trigger

CREATE TRIGGER IstatistikEklendigindeLogla

AFTER INSERT ON Istatistikler

FOR EACH ROW

EXECUTE FUNCTION IstatistikEkleLog();

-- Kullanıcılar

INSERT INTO Kullanici (kullanici\_adi, email, sifre, favori\_oyunlar) VALUES

('Ali', 'ali@example.com', 'hashed\_password\_1', 'Valorant, League of Legends'),

('Mert', 'mert@example.com', 'hashed\_password\_2', 'CS:GO, PUBG'),

('janet46', 'lisa85@cardenas.org', 'hashed\_password\_0', 'League of Legends'),

('mwilkins', 'katie14@taylor-patterson.net', 'hashed\_password\_1', 'PUBG'),

('brooke77', 'mfox@garcia.com', 'hashed\_password\_2', 'Dota 2'),

('rjohns', 'donna49@gmail.com', 'hashed\_password\_3', 'League of Legends'),

('alexandermichael', 'stacy98@henderson.org', 'hashed\_password\_4', 'CS:GO'),

('hectorgonzalez', 'amandasmith@baxter.net', 'hashed\_password\_5', 'Dota 2'),

('abrown', 'abrown@gmail.com', 'hashed\_password\_6', 'Valorant'),

('emilyj', 'emilyj@outlook.com', 'hashed\_password\_7', 'PUBG');

-- Oyunlar

INSERT INTO Oyunlar (oyun\_adi) VALUES

('Valorant'),

('League of Legends'),

('Dota 2'),

('CS:GO'),

('PUBG'),

('Overwatch'),

('Apex Legends'),

('Rocket League'),

('Rainbow Six Siege'),

('Fortnite');

-- Takımlar

INSERT INTO Takimlar (takim\_adi, uyeler, kaptan) VALUES

('Şimşekler', 'Ali, Mert', 'Ali'),

('Kartallar', 'Mert, Janet', 'Mert'),

('Zaferler', 'Ahmet, Zeynep, Hasan', 'Ahmet'),

('Galaksiler', 'Mehmet, Elif, Selim', 'Mehmet'),

('Yıldızlar', 'Brooke, Lisa, Amanda', 'Lisa'),

('Fırtına', 'John, Emily, Hector', 'Emily'),

('Karanlıklar', 'Alex, Katie, Donna', 'Katie'),

('Aslanlar', 'Sam, Leo, Max', 'Sam'),

('Ejderhalar', 'Sophie, Emma, Mia', 'Emma'),

('Volkanlar', 'Jack, Ryan, Noah', 'Jack');

-- Turnuvalar

INSERT INTO Turnuvalar (turnuva\_adi, turu, tarih, aciklama) VALUES

('Yaz Kupası', 'Online', '2025-08-01', 'Yaz turnuvası'),

('Kış Kupası', 'LAN', '2025-12-15', 'Kış temalı turnuva'),

('Bahar Şampiyonası', 'Online', '2025-04-20', 'Bahar dönemi karşılaşmaları'),

('Sonbahar Meydan Okuması', 'LAN', '2025-10-10', 'Sonbahar turnuvası'),

('Pro League', 'Online', '2025-09-05', 'Profesyonel takımların mücadelesi'),

('Amatör Turnuva', 'Online', '2025-06-10', 'Yeni başlayanlar için'),

('Gençler Ligi', 'Online', '2025-07-01', 'Genç oyuncular turnuvası'),

('Kadınlar Ligi', 'LAN', '2025-05-30', 'Kadın oyunculara özel'),

('1v1 Challenge', 'Online', '2025-11-11', 'Birebir mücadele'),

('Takım Turnuvası', 'LAN', '2025-03-03', 'En iyi takım ödülü');

-- Turnuva Ödülleri

INSERT INTO TurnuvaOdulleri (turnuva\_id, odul\_miktari, odul\_turu) VALUES

(1, 1000, 'Para Ödülü'),

(2, 800, 'Ekipman'),

(3, 500, 'Ekipman'),

(4, 300, 'Para Ödülü'),

(5, 1200, 'Sponsorluk'),

(6, 200, 'Teşekkür Belgesi'),

(7, 400, 'Ödül Çeki'),

(8, 600, 'Kupa'),

(9, 250, 'Madalya'),

(10, 700, 'Eşya Seti');

-- Maçlar

INSERT INTO Maclar (takim1\_id, takim2\_id, tarih, saat, harita, turnuva\_id) VALUES

(1, 2, '2025-08-01', '15:00', 'Harita A', 1),

(2, 3, '2025-08-02', '16:00', 'Harita B', 1),

(3, 4, '2025-08-03', '17:00', 'Harita C', 1),

(4, 5, '2025-08-04', '18:00', 'Harita D', 1),

(5, 6, '2025-08-05', '19:00', 'Harita E', 1),

(6, 7, '2025-08-06', '20:00', 'Harita F', 1),

(7, 8, '2025-08-07', '21:00', 'Harita G', 1),

(8, 9, '2025-08-08', '22:00', 'Harita H', 1),

(9, 10, '2025-08-09', '23:00', 'Harita I', 1),

(10, 1, '2025-08-10', '14:00', 'Harita J', 1);

-- İstatistikler

INSERT INTO Istatistikler (mac\_id, takim\_id, skor, kazanan) VALUES

(1, 1, 10, TRUE),

(1, 2, 8, FALSE),

(2, 2, 12, TRUE),

(2, 3, 7, FALSE),

(3, 3, 9, TRUE),

(3, 4, 6, FALSE),

(4, 4, 11, TRUE),

(4, 5, 8, FALSE),

(5, 5, 13, TRUE),

(5, 6, 7, FALSE),

(6, 6, 14, TRUE),

(6, 7, 10, FALSE),

(7, 7, 12, TRUE),

(7, 8, 9, FALSE),

(8, 8, 11, TRUE),

(8, 9, 8, FALSE),

(9, 9, 15, TRUE),

(9, 10, 12, FALSE),

(10, 10, 14, TRUE),

(10, 1, 13, FALSE);

-- Turnuva Katılım Kayıtları

INSERT INTO TurnuvaKatilim (kullanici\_id, turnuva\_id, takim\_id, bireysel) VALUES

(1, 1, 1, FALSE),

(2, 2, 2, FALSE),

(3, 3, 3, FALSE),

(4, 4, 4, FALSE),

(5, 5, 5, FALSE),

(6, 6, 6, FALSE),

(7, 7, 7, FALSE),

(8, 8, 8, FALSE),

(9, 9, 9, FALSE),

(10, 10, 10, FALSE);

-- Turnuva Sonuçları Kayıtları

INSERT INTO TurnuvaSonuclari (turnuva\_id, takim\_id, skor, derece) VALUES

(1, 1, 10, 1),

(1, 2, 8, 2),

(2, 2, 12, 1),

(2, 3, 7, 2),

(3, 3, 9, 1),

(3, 4, 6, 2),

(4, 4, 11, 1),

(4, 5, 8, 2),

(5, 5, 13, 1),

(5, 6, 7, 2);

-- Takım Eklenince log

CREATE OR REPLACE FUNCTION TakimEkleLog()

RETURNS TRIGGER AS $$

BEGIN

INSERT INTO IstatistikLog (mac\_id, takim\_id)

VALUES (NEW.mac\_id, NEW.takim\_id);

RAISE NOTICE 'Yeni takım eklendi: %', NEW.takim\_adi;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER TakimEkleTrigger

AFTER INSERT ON Takimlar

FOR EACH ROW

EXECUTE FUNCTION TakimEkleLog();

-- İstatistik Güncellenince log

CREATE OR REPLACE FUNCTION IstatistikUpdateLog()

RETURNS TRIGGER AS $$

BEGIN

INSERT INTO IstatistikLog (mac\_id, takim\_id)

VALUES (NEW.mac\_id, NEW.takim\_id);

RAISE NOTICE 'İstatistik güncellendi: Maç ID % Takım ID %', NEW.mac\_id, NEW.takim\_id;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER IstatistikUpdateTrigger

AFTER UPDATE ON Istatistikler

FOR EACH ROW

EXECUTE FUNCTION IstatistikUpdateLog();

-- Fonksiyonlar

-- Turnuva Adına Göre Arama

CREATE OR REPLACE FUNCTION TurnuvaAra(turnuva\_adi\_param VARCHAR)

RETURNS TABLE(turnuva\_adi VARCHAR, tarih DATE) AS $$

BEGIN

RETURN QUERY

SELECT turnuva\_adi, tarih

FROM Turnuvalar

WHERE turnuva\_adi LIKE '%' || turnuva\_adi\_param || '%';

END;

$$ LANGUAGE plpgsql;

-- Kullanıcı Puanına Göre Arama

CREATE OR REPLACE FUNCTION KullaniciPuanAra(min\_puan INT)

RETURNS TABLE(kullanici\_adi VARCHAR, favorite\_oyunlar TEXT) AS $$

BEGIN

RETURN QUERY

SELECT k.kullanici\_adi, k.favorite\_oyunlar

FROM Kullanici k

JOIN TurnuvaKatilim tk ON k.kullanici\_id = tk.kullanici\_id

JOIN TurnuvaSonuclari ts ON tk.turnuva\_id = ts.turnuva\_id

WHERE ts.skor > min\_puan;

END;

$$ LANGUAGE plpgsql;

-- Kullanıcı Turnuva Ödülleri

CREATE OR REPLACE FUNCTION KullaniciTurnuvaOdul(kullanici\_id\_param INT)

RETURNS TABLE(turnuva\_adi VARCHAR, odul\_miktari INT, odul\_turu VARCHAR) AS $$

BEGIN

RETURN QUERY

SELECT t.turnuva\_adi, tod.odul\_miktari, tod.odul\_turu

FROM TurnuvaKatilim tk

JOIN TurnuvaSonuclari ts ON tk.turnuva\_id = ts.turnuva\_id

JOIN TurnuvaOdulleri tod ON ts.turnuva\_id = tod.turnuva\_id

JOIN Turnuva t ON ts.turnuva\_id = t.turnuva\_id

WHERE tk.kullanici\_id = kullanici\_id\_param;

END;

$$ LANGUAGE plpgsql;

-- Turnuva Sonuçları ve Ödülleri Görüntüleme

CREATE VIEW TurnuvaSonucOdul AS

SELECT

t.turnuva\_adi,

ts.derece,

ts.skor,

tod.odul\_miktari,

tod.odul\_turu

FROM

Turnuvalar t

JOIN TurnuvaSonuclari ts ON t.turnuva\_id = ts.turnuva\_id

JOIN TurnuvaOdulleri tod ON t.turnuva\_id = tod.turnuva\_id;

-- Toplam Skoru 50'yi Geçen Turnuvaları Listeleme

SELECT turnuva\_id, SUM(skor) AS toplam\_skor

FROM TurnuvaSonuclari

GROUP BY turnuva\_id

HAVING SUM(skor) > 50;

-- RECORD örneği

CREATE OR REPLACE FUNCTION kullanici\_bilgi\_goster()

RETURNS VOID AS $$

DECLARE

r RECORD;

BEGIN

FOR r IN SELECT \* FROM Kullanici LOOP

RAISE NOTICE 'Kullanici Adi: %, Email: %', r.kullanici\_adi, r.email;

END LOOP;

END;

$$ LANGUAGE plpgsql;

-- CURSOR örneği

CREATE OR REPLACE FUNCTION kullanici\_cursor()

RETURNS VOID AS $$

DECLARE

cur CURSOR FOR SELECT \* FROM Kullanici;

r RECORD;

BEGIN

OPEN cur;

LOOP

FETCH cur INTO r;

EXIT WHEN NOT FOUND;

RAISE NOTICE 'Kullanici Adi: %, Email: %', r.kullanici\_adi, r.email;

END LOOP;

CLOSE cur;

END;

$$ LANGUAGE plpgsql;