EGE UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

INTRODUCTION TO DATABASE 2023-2024 TERM PROJECT: COUNTRY TRANSPORTATION DATABASE

05200000086 - Kaan Durmuş

05200000737 – Cansu Aruk

05210000064 – Hüseyin Koç

05220000007 – Deniz Bahar

**Data req**

**Turkish Airlines**

* AIRPORT: Her havaalanı, bir şehirde yer alır ve benzersiz bir havaalanı koduna (örneğin: IATA kodu) sahiptir. Her havaalanı, bir veya birden fazla JOURNEY için kalkış ve varış noktası olarak kullanılır.
* TICKET: Her bilet için benzersiz bir kimlik numarası, koltuk numarası, satın alma tarihi, kategori ve fiyat bilgileri saklanır. Bilet kayıtları ayrıca yolcunun hayvan taşıyıp taşımadığı ve ekstra bagaj hakkı kullanıp kullanmadığı bilgilerini içerir. Her TICKET bir PERSON tarafından satın alınır ve bir JOURNEY ile ilişkilendirilir.
* PERSON: Her kişi için bir kayıt oluşturulur ve bu kayıt ad, soyad, TC kimlik numarası, cinsiyet ve doğum tarihi gibi özellikler içerir. Her PERSON bir veya birden fazla TICKET satın alabilir. Her kişinin kendine ait bir search history ve transaction tablosu vardır.
* JOURNEY: Her seyahat için bir kayıt oluşturulur ve bu kayıt kalkış ve varış havaalanlarını, seyahat süresini ve tarihi içerir. Her JOURNEY bir AIRPLANE ile ilişkilendirilir ve bir kalkış ve bir varış AIRPORT ile ilişkilendirilir.
* AIRPLANE: Her uçak için bir kayıt oluşturulur ve bu kayıt koltuk kapasitesi, menzil, hız ve model gibi özellikler içerir. Her AIRPLANE birden fazla JOURNEY için kullanılabilir.
* **SEARCH\_HISTORY**: Bu varlık, bir **PERSON**'ın yaptığı aramaların geçmişini tutar. Her kayıt, arama yapılan zaman ve aramanın yapıldığı tarih bilgilerini içerir. **SEARCH\_HISTORY** doğrudan bir **JOURNEY** ile ilişkilendirilir ve yolculukla ilgili arama kriterlerini (örneğin, kalkış ve varış havaalanları, tarihler) saklar.
* **TRANSACTION**: Bu, bir **TICKET** satın alımını veya bir hizmet için yapılan ödemeyi temsil eder. Her **TRANSACTION** kaydı, ilgili **TICKET** bilgilerini (bilet numarası, satın alma tarihi, fiyat) ve işlemle ilgili ek bilgileri (örneğin, ekstra bagaj, hayvan varlığı) içerir.

**High Speed Train**

PERSON: Her kişi için ad, soyad, TC kimlik numarası, cinsiyet, doğum tarihi, telefon ve e-posta bilgileri içeren bir kayıt bulunur. Her kişi birden fazla bilet alabilir ve birden fazla yolculuk yapabilir. Her kişinin kendine ait bir search history ve transaction tablosu vardır.

TICKET: Her bilet için bir kimlik numarası, kategori (sınıfı), fiyat, koltuk numarası, ek bagaj varlığı ve evcil hayvan varlığı gibi bilgiler içeren bir kayıt bulunur. Her bilet bir PERSON ve bir JOURNEY ile ilişkilendirilir.

JOURNEY: Her yolculuk için kalkış ve varış istasyonları, tarih ve zaman bilgileri içeren bir kayıt bulunur. Yolculuklar, birden fazla **durak** içerebilir ve bir TRAIN ile ilişkilendirilir.

TRAIN: Her tren için bir kimlik numarası, tren tipi (örneğin, hızlı tren, yavaş tren) ve vagon sayısı gibi bilgiler içeren bir kayıt bulunur. Her tren, birden fazla yolculukta kullanılabilir.

STATION: Her istasyon için bir isim ve şehir bilgisi içeren bir kayıt bulunur. İstasyonlar, bir veya daha fazla JOURNEY için durak olarak kullanılabilir.

SEARCH\_HISTORY: Her PERSON'ın etkinliklerini ve arama geçmişini izleyen kayıtlar bulunur. SEARCH\_HISTORY kullanıcıların yolculuk aramalarını kaydeder.

TRANSACTION\_HISTORY: Her bilet satın alım işlemi için tarih, saat ve işlem tutarı gibi bilgiler içeren bir kayıt bulunur. Her işlem bir TICKET ile ilişkilendirilir.

**Yolcu360**

PERSON: Sistemdeki her kişi için ad, soyad, TC kimlik numarası, adres, doğum tarihi, telefon, e-posta ve kredi kartı bilgileri içeren bir kayıt tutulur. Bir PERSON bir veya daha fazla araç kiralayabilir Her kişinin kendine ait bir search history ve transaction tablosu vardır.

VEHICLE: Her araç için araç ID, araç modeli, yakıt tipi, mevduat ücreti, kiralama ücreti, araç markası, sınıfı, fiyat ve mevcut olup olmadığı bilgileri içeren bir kayıt bulunur. Bir araç birden fazla RENT işlemine konu olabilir.

RENT: Her kiralama işlemi için kiralama ID, kiralama süresi ve zaman bilgileri içeren bir kayıt tutulur. Kiralama kaydı, teslim alma yeri, teslim etme yeri, teslim alma tarihi ve zamanı, teslim etme tarihi ve zamanı gibi detaylar içerir. Her RENT işlemi bir PERSON ve bir VEHICLE ile ilişkilendirilir.

COMPANY: Her şirket için isim ve konum bilgileri içeren bir kayıt bulunur. Bir COMPANY birden fazla VEHICLE'a sahip olabilir.

HISTORY: Her PERSON'ın geçmiş işlemlerini ve etkinliklerini izleyen kayıtlar bulunur. HISTORY kaydı, araç kiralama, araç iadesi ve arama gibi işlemlerin zaman damgalarını ve tarihlerini içerir.

SEARCH\_HISTORY: PERSON tarafından yapılan araç arama geçmişini izleyen kayıtlar bulunur. Her SEARCH\_HISTORY kaydı, arama yapılan tarih ve zaman bilgilerini içerir ve arama kriterlerine göre filtrelenen VEHICLE varlıkları ile ilişkilendirilebilir.

RENT\_HISTORY: Her PERSON'ın kiralama geçmişini izleyen kayıtlar bulunur. RENT\_HISTORY kaydı, kiralama zamanları ve tarihleri ile ilişkilendirilmiş RENT işlemlerinin özetlerini içerir.

**IZBAN(bunda bilet yok bizde ona göre sınavda başka bi durum olursa ona göre güncellersin)**

PERSON: Her kişi için ad, soyad, TC kimlik numarası ve kişisel iletişim bilgileri (adres, telefon, e-posta) gibi temel bilgileri içeren bir kayıt bulunur. Bir kişi birden fazla JOURNEY yapabilir. Her kişinin kendine ait bir search history ve transaction tablosu vardır.

TRAIN: Her tren için bir kimlik numarası, tren tipi ve vagon sayısı gibi özellikleri içeren bir kayıt bulunur. Bir tren birden fazla JOURNEY gerçekleştirebilir.

JOURNEY: Her yolculuk için kalkış ve varış istasyonları, tarih ve saat bilgileri içeren bir kayıt bulunur. Yolculuklarda bir TRAIN kullanılır ve her durak bir STATION ile ilişkilendirilir.

STATION: Her istasyon için bir isim ve lokasyon bilgisi içeren bir kayıt bulunur. İstasyonlar, bir veya daha fazla JOURNEY için durak olarak kullanılabilir.

HISTORY: Her PERSON'ın yolculuk ve bilet alım etkinliklerini izleyen kayıtlar bulunur. HISTORY kayıtları, işlem zamanları ve tarihleri içerir.

SEARCH\_HISTORY: Kullanıcıların yolculuk aramalarını izleyen kayıtlar bulunur. Her SEARCH\_HISTORY kaydı, arama tarih ve saatleri ile birlikte kullanıcı tarafından girilen arama kriterlerini içerir. Journey ile

TRANSACTION\_HISTORY: Her journey alım işlemi için bir kayıt bulunur ve bu kayıt işlem tarih ve saatleri ile işlemin mali detaylarını içerir.

**martı**

PERSON: Her birey için ad, soyad, TC kimlik numarası, cinsiyet, doğum tarihi, iletişim bilgileri (telefon ve e-posta) ve kredi kartı bilgilerini içeren bir kayıt bulunur. Kişilerin sistem içindeki etkinlikleri ve kiralama işlemleri HISTORY ile izlenir. Her kişinin kendine ait bir search history ve transaction tablosu vardır.

MARTI VEHICLE: Her elektrikli scooter için benzersiz bir kimlik numarası, model bilgisi, pil durumu ve anlık konum bilgileri (koordinatlar) içeren bir kayıt bulunur. Her araç, birden fazla RENT işlemi için kullanılabilir ve mevcut durumu RENT\_HISTORY ile izlenir.

RENT: Her kiralama işlemi için başlangıç ve bitiş zamanları, teslim alma ve bırakma yer koordinatları, kiralama süresi ve toplam fiyat bilgilerini içeren bir kayıt bulunur. Her RENT işlemi bir PERSON ile ilişkilendirilir ve işlem detayları TRANSACTION\_HISTORY içinde saklanır.

HISTORY: Her kullanıcının etkinlik kayıtlarını, kiralama ve iade işlemlerini içeren bir kayıt bulunur. Bu kayıt, işlemlerin zaman damgalarını ve tarihlerini içerir.

SEARCH\_HISTORY: Kullanıcıların araç arama etkinliklerini izleyen kayıtlar bulunur. Her kayıt, arama yapılan tarih ve saati içerir ve kullanıcı tarafından girilen arama kriterlerine göre filtrelenmiş araçları gösterebilir.

RENT\_HISTORY: Her kullanıcının kiralama geçmişini izleyen kayıtlar bulunur. Bu kayıtlar, her kiralama etkinliğinin zaman damgasını ve tarihi ile kiralanan MARTI VEHICLE bilgilerini içerir

**IDO**

PERSON: Her birey için ad, soyad, TC kimlik numarası, cinsiyet, doğum tarihi ve iletişim bilgileri (telefon, e-posta) ile kredi kartı bilgilerini içeren bir kayıt bulunur. Kişilerin satın alma ve seyahat etkinlikleri HISTORY ile izlenir. Her kişinin kendine ait bir search history ve transaction tablosu vardır.

TICKET: Her bilet için bilet numarası, satın alma tarihi, fiyat, kategori ve varsa araç bilgisi (arabasız/arabalı) içeren bir kayıt bulunur. Biletler, bir PERSON tarafından satın alınır ve bir JOURNEY ile ilişkilendirilir.

JOURNEY: Her yolculuk için, kalkış ve varış limanları, tarih ve zaman bilgileri ile süre bilgilerini içeren bir kayıt bulunur. Yolculuklar, birden fazla PORT'u içerebilir ve bir FERRY ile gerçekleştirilir.

FERRY: Her feribot için, tip, yakıt türü ve koltuk sayısı bilgilerini içeren bir kayıt bulunur. Feribotlar, birden fazla JOURNEY için kullanılabilir.

PORT: Her liman için, isim ve konum bilgilerini içeren bir kayıt bulunur. Limanlar, bir veya daha fazla JOURNEY için kalkış veya varış noktası olarak kullanılabilir.

HISTORY: PERSON'ın satın alma ve seyahat etkinliklerini içeren bir kayıt bulunur. HISTORY, işlem tarihleri ve zamanları ile tüm seyahatlerin ve işlemlerin özetini içerir.

SEARCH\_HISTORY: PERSON tarafından yapılan bilet arama geçmişini izleyen kayıtlar bulunur. Her SEARCH\_HISTORY kaydı, arama yapılan tarih ve saati içerir ve arama sonuçlarını gösterebilir.

TRANSACTION\_HISTORY: Her bilet satın alma işlemi için, işlem tarih ve saatleri ile mali detayları içeren bir kayıt bulunur.

**obilet**

o bilet için 4 tane yolculuk var detayında hepsi aynı req. Bunlar ido, araba kiralama, uçak ve otobüs bi tek otobüs yeni. Yukardaki data reqleri yazarak bu data reqi çıkarırsın sadece otobüsü ekliyorum.

BUS: Her otobüs için benzersiz bir kimlik numarası, yakıt tipi, koltuk sayısı, şanzıman tipi ve otobüs modeli bilgilerini içeren bir kayıt bulunur. Her otobüs, bir veya birden fazla BUS JOURNEY gerçekleştirebilir.

BUS JOURNEY: Her otobüs yolculuğu için, yolculuk adı, kalkış ve varış zamanları, kalkış ve varış durakları (bir veya daha fazla BUS STATION), ve yolculuğun içerdiği durak sayısını içeren bir kayıt bulunur. Her BUS JOURNEY, belirli bir BUS ile ilişkilendirilir ve bu yolculuk sırasında gerçekleştirilen duraklamaları içerebilir.

**DESIGN-CONCEPTUAL DESIGN**

**diyagram, renklilik, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Er to relational**

**1ST ITERATION**

***1-Regular Entity Types***

PERSON (Name, Surname, TC, Gender, Birth\_Date, Phone, E-mail)

CITIZEN\_CARD (Card\_ID, Money\_Amount)

JOURNEY (Journey\_ID)

COMPANY (Company\_Name, Location)

***2-Weak Entity Types***

TRANSACTION (Transaction\_ID, TC, Seat\_Number, Price, Scheduled\_Arr\_Date, Scheduled\_Arr\_Time, Scheduled\_Dep\_Date, Scheduled\_Dep\_Time, Ticket\_Category, Pet, Seat\_Rezervation, Extra\_Baggage\_Charges, Receive\_Place, Receive\_Place\_X, Receive\_Place\_Y, Departure\_Place, Departure\_Place\_X, Departure\_Place\_Y, Purchase\_Date, Purchase\_Time) **// BUY**

TRANSACTION (Transaction\_ID, TC, Journey\_ID, Seat\_Number, Total Price, Scheduled\_Arr\_Date, Scheduled\_Arr\_Time, Scheduled\_Dep\_Date, Scheduled\_Dep\_Time, Ticket\_Category, Pet, Seat\_Rezervation, Extra\_Baggage\_Charges, Receive\_Place, Receive\_Place\_X, Receive\_Place\_Y, Departure\_Place, Departure\_Place\_X, Departure\_Place\_Y ,Purchase\_Date, Purchase\_Time) **// INCLUDE**

HISTORY (History\_ID, TC) **// HOLD**

***3-Binary 1:1 Relation Types***

PERSON (Name, Surname, TC, Gender, Birth\_Date, Phone, E-mail, Card\_ID) **// HAS**

***4-Binary 1:N Relationship Types***

JOURNEY (Journey\_ID, Company\_Name) **// HAVE**

***5-Binary M:N Relationship Types***

**-**

***6-Multivalued Attributes***

LOCATION (Company\_Name, Location\_ID)

***7-N-ary Relationship Types***

**-**

***8-Specialization/Generalization***

***8.a-Multiple Relations: Superclass and Subclasses***

JOURNEY (Journey\_ID, Company\_Name)

PLANE\_JOURNEY (Journey\_ID)

BUS\_JOURNEY (Journey\_ID)

VEHICLE\_JOURNEY ((Journey\_ID)

FERRY\_JOURNEY (Journey\_ID)

METRO\_JOURNEY (Journey\_ID)

MARTI\_JOURNEY (Journey\_ID)

HIGH\_SPEED \_TRAIN\_JOURNEY (Journey\_ID)

***8.B-Multiple Relations: Subclasses***

SEARCH\_HISTORY (History\_ID, TC Search\_Time, Search\_Date)

TRANSACTION\_HISTORY (History\_ID, TC)

***9-Union Types***

**-**

**2ND ITERATION**

***1-Regular Entity Types***

**// PLANE JOURNEY**

AIRPLANE (Aircraft\_Type, Max\_Range, Heigh\_Max\_Speed, Wingspan\_Lenght)

AIRPORT (Airport\_Name, City)

**// BUS JOURNEY**

BUS (Bus\_ID, Fuel Type, Number of Seat, Transmission Type, Bus Model)

BUS\_STATION (Station\_Name)

**// FERRY JOURNEY**

FERRY (Ferry\_ID, Fuel\_Type, Ferry\_Type, Seat\_Count)

PORT (Port\_Name)

**// METRO JOURNEY**

TRAIN (Train\_Number, Train\_Type, Vagon\_Type, Seat\_Count)

METRO\_STATION (Station\_Name)

**// HIGH SPEED TRAIN JOURNEY**

TRAIN (Train\_Number, Train\_Type, Vagon\_Type, Seat\_Count)

HIGH\_SPEED\_TRAIN\_STATION (Station\_Name)

**// MARTI JOURNEY**

MARTI (Marti\_ID, Battery, Price, Location\_X, Location\_Y, Type)

**// VEHİCLE JOURNEY**

VEHICLE (Vehicle\_ID, Model, Fuel Type, Deposite Fees, Car Brand, Car Class, Location)

***2-Weak Entity Types***

**-**

***3-Binary 1:1 Relation Types***

TRANSACTION\_HISTORY (History\_ID, TC, Transaction\_ID, Journey\_ID) **// HOLD**

***4-Binary 1:N Relationship Types***

SEARCH\_HISTORY (History\_ID, TC, Search\_Time, Search\_Date, Journey\_ID) **// HOLD**

**// PLANE JOURNEY**

PLANE\_JOURNEY (Journey\_ID, Aircraft\_Type) **// INCLUDE**

PLANE\_JOURNEY (Journey\_ID, Aircraft\_Type, Departure\_Airport\_Name, Departure\_Time, Departure\_Date) **// DEPARTURE**

PLANE\_JOURNEY (Journey\_ID, Aircraft\_Type, Arrival\_Airport\_Name, Departure\_Time, Departure\_Date, Arrival\_Time, Arrival\_Date) **// ARRIVAL**

PLANE\_JOURNEY(J͟o͟u͟r͟n͟e͟y͟ ͟I͟d͟, Aircraft\_Type, Arrival\_Airport\_Name, Departure\_Time, Departure\_Date, Arrival\_Time, Arrival\_Date, Next\_Journey\_Id)//NEXT JOURNEY

**// BUS JOURNEY**

BUS\_JOURNEY (Journey\_ID, Bus\_ID) **// INCLUDE**

BUS\_JOURNEY (Journey\_ID, Bus\_ID, Arrival\_Time, Arrival\_Date, Arrival\_Bus\_Station) **// ARRIVAL**

BUS\_JOURNEY (Journey\_ID, Bus\_ID, Arrival\_Time, Arrival\_bus\_station, Departure\_Time, Departure\_Date, Departure\_Station) **// DEPARTURE**

**// FERRY JOURNEY**

FERRY\_JOURNEY (Journey\_ID, Ferry\_ID) **// OPERATE**

FERRY\_JOURNEY (Journey\_ID, Ferry\_ID, Arrival\_Time, Arrival\_Date, Arrival\_Port) **// ARRIVAL**

FERRY\_JOURNEY (Journey\_ID, Ferry\_ID, Arrival\_Time, Arrival\_Date, Arrival\_port, Departure\_Time, Departure\_Date, Departure\_Port) **// DEPARTURE**

**// METRO JOURNEY**

METRO\_JOURNEY (Journey\_ID, Train\_Number) **// OPERATE**

METRO\_JOURNEY (Journey\_ID, Train\_Number, Arrival\_Time, Arrival\_Date, Arrival\_Station) **// ARRIVAL**

METRO\_JOURNEY (Journey\_ID, Train\_Number, Arrival\_Time, Arrival\_Date, Arrival\_station, Departure\_Time, Departure\_Date, Departure\_Station) **// DEPARTURE**

**// HIGH SPEED TRAIN JOURNEY**

HIGH\_SPEED\_TRAIN\_JOURNEY (Journey\_ID, Train\_Number) **// OPERATE**

HIGH\_SPEED\_TRAIN\_JOURNEY (Journey\_ID, Train\_Number, Arrival\_Time, Arrival\_Date, Arrival\_Station) **// ARRIVAL**

HIGH\_SPEED\_TRAIN\_JOURNEY (J͟o͟u͟r͟n͟e͟y͟ ͟I͟d͟, Train\_Number, Arrival\_Time, Arrival\_Station, Departure\_Time, Departure\_Date, Departure\_Station) **// DEPARTURE**

**// MARTI JOURNEY**

MARTI\_JOURNEY (Journey\_ID, MARTI\_ID)

**// VEHICLE JOURNEY**

VEHICLE\_JOURNEY (Journey\_ID, Vehicle\_ID)

***5-Binary M:N Relationship Types***

BUS\_STOPS (Station\_Name, Bus\_Journey\_ID, Arrival\_Time, Arrival\_Date, Departure\_Time, Departure\_Date) **// STOPS**

FERRY\_STOPS (Ferry Journey ID, Port\_Name, Arrival\_Time, Arrival\_Date, Departure\_Time, Departure\_Date) **// STOPS**

METRO\_STOPS (Metro\_Journey\_ID, Station \_Name, Arrival\_Time, Arrival\_Date, Departure\_Time, Departure\_Date) **// STOPS**

HIGH\_SPEED\_TRAIN\_STOPS (High\_Speed\_Train\_Journey\_ID, Station\_Name, Arrival\_Time, Arrival\_Date, Departure\_Time, Departure\_Date)

**sql**

TABLE

CREATE TABLE CITIZEN\_CARD (

Card\_ID INT PRIMARY KEY NOT NULL,

Money\_Amount DECIMAL(10, 2) NOT NULL CHECK (Money\_Amount >= 0)

);

CREATE TABLE PERSON (

Name NVARCHAR(50) NOT NULL,

Surname NVARCHAR(50) NOT NULL,

TC CHAR(11) PRIMARY KEY NOT NULL,

Gender NVARCHAR(10) NOT NULL,

Birth\_Date DATE NOT NULL,

Phone NVARCHAR(15),

Email NVARCHAR(100),

Card\_ID INT NOT NULL DEFAULT 0,

FOREIGN KEY (Card\_ID) REFERENCES CITIZEN\_CARD(Card\_ID) ON DELETE SET DEFAULT ON UPDATE CASCADE

);

CREATE TABLE COMPANY (

Company\_Name NVARCHAR(255) PRIMARY KEY NOT NULL

);

CREATE TABLE LOCATIONS (

Location\_Id INT NOT NULL,

Company\_Name NVARCHAR(255) NOT NULL DEFAULT 'NO NAME',

Address NVARCHAR(255) NOT NULL,

FOREIGN KEY (Company\_Name) REFERENCES COMPANY(Company\_Name) ON DELETE SET DEFAULT ON UPDATE CASCADE,

PRIMARY KEY (Location\_Id, Company\_Name)

);

CREATE TABLE JOURNEY (

Journey\_Id INT PRIMARY KEY NOT NULL,

Company\_Name NVARCHAR(255) NOT NULL DEFAULT 'NO NAME',

FOREIGN KEY (Company\_Name) REFERENCES COMPANY(Company\_Name) ON DELETE SET DEFAULT ON UPDATE CASCADE

);

CREATE TABLE TRANSACTIONS (

Transaction\_Id INT NOT NULL,

TC CHAR(11) NOT NULL DEFAULT '00000000000',

Journey\_Id INT NOT NULL DEFAULT 0000,

Seat\_Number INT,

Total\_Price DECIMAL(10, 2) NOT NULL,

Scheduled\_Arr\_Date DATE,

Scheduled\_Arr\_Time TIME,

Scheduled\_Dep\_Date DATE,

Scheduled\_Dep\_Time TIME,

Ticket\_Category NVARCHAR(50),

Departure\_Place NVARCHAR(255),

Receive\_Place NVARCHAR(255),

Departure\_Place\_X DECIMAL(9,6),

Departure\_Place\_Y DECIMAL(9,6),

Receive\_Place\_X DECIMAL(9,6),

Receive\_Place\_Y DECIMAL(9,6),

Pet BIT,

Seat\_Reservation BIT,

Extra\_Baggage\_Charges DECIMAL(10, 2),

Purchase\_Date DATE NOT NULL,

Purchase\_Time TIME NOT NULL,

FOREIGN KEY (TC) REFERENCES PERSON(TC) ON DELETE SET DEFAULT ON UPDATE CASCADE,

FOREIGN KEY (Journey\_Id) REFERENCES JOURNEY(Journey\_Id) ON DELETE SET DEFAULT ON UPDATE CASCADE,

PRIMARY KEY (Transaction\_Id, TC, Journey\_Id)

);

CREATE TABLE TRANSACTION\_HISTORY (

History\_Id INT NOT NULL,

TC CHAR(11) NOT NULL DEFAULT '00000000000',

Transaction\_Id INT NOT NULL DEFAULT 0,

Journey\_Id INT NOT NULL DEFAULT 0000,

FOREIGN KEY (Transaction\_Id, TC, Journey\_Id) REFERENCES TRANSACTIONS(Transaction\_Id,TC, Journey\_Id) ON DELETE SET DEFAULT ON UPDATE CASCADE,

PRIMARY KEY (History\_Id, TC)

);

CREATE TABLE SEARCH\_HISTORY (

History\_Id INT NOT NULL,

TC CHAR(11) NOT NULL DEFAULT '00000000000',

Journey\_Id INT NOT NULL DEFAULT 0000,

Search\_Time TIME NOT NULL,

Search\_Date DATE NOT NULL,

PRIMARY KEY (History\_Id, TC),

FOREIGN KEY (TC) REFERENCES PERSON(TC) ON DELETE SET DEFAULT ON UPDATE CASCADE,

FOREIGN KEY (Journey\_Id) REFERENCES JOURNEY(Journey\_Id) ON DELETE SET DEFAULT ON UPDATE CASCADE,

);

CREATE TABLE AIRPORT (

Airport\_Name NVARCHAR(255) PRIMARY KEY NOT NULL,

City NVARCHAR(100) NOT NULL

);

CREATE TABLE AIRPLANE (

Aircraft\_Type NVARCHAR(255) PRIMARY KEY NOT NULL ,

Max\_Range INT NOT NULL,

High\_Max\_Speed INT NOT NULL,

Wingspan\_Length INT NOT NULL

);

CREATE TABLE PLANE\_JOURNEY (

Journey\_Id INT PRIMARY KEY NOT NULL CHECK (Journey\_Id >= 1000 AND Journey\_Id < 2000),

Aircraft\_Type NVARCHAR(255) NOT NULL,

Arrival\_Airport\_Name NVARCHAR(255) NOT NULL,

Departure\_Airport\_Name NVARCHAR(255) NOT NULL,

Departure\_Time TIME NOT NULL,

Departure\_Date DATE NOT NULL,

Arrival\_Time TIME NOT NULL,

Arrival\_Date DATE NOT NULL,

Next\_Journey\_ID INT,

FOREIGN KEY (Aircraft\_Type) REFERENCES AIRPLANE(Aircraft\_Type),

FOREIGN KEY (Arrival\_Airport\_Name) REFERENCES AIRPORT(Airport\_Name),

FOREIGN KEY (Departure\_Airport\_Name) REFERENCES AIRPORT(Airport\_Name),

FOREIGN KEY (Next\_Journey\_ID) REFERENCES PLANE\_JOURNEY(Journey\_Id),

FOREIGN KEY (Journey\_Id) REFERENCES Journey(Journey\_Id)

);

CREATE TABLE BUS (

Bus\_ID INT PRIMARY KEY NOT NULL,

Fuel\_Type NVARCHAR(50) NOT NULL,

Number\_of\_Seats INT NOT NULL,

Transmission\_Type NVARCHAR(50) NOT NULL,

Bus\_Model NVARCHAR(50) NOT NULL

);

CREATE TABLE BUS\_STATION (

Station\_Name NVARCHAR(255) PRIMARY KEY NOT NULL

);

CREATE TABLE BUS\_JOURNEY (

Journey\_Id INT PRIMARY KEY NOT NULL CHECK (Journey\_Id >= 2000 AND Journey\_Id < 3000),

Bus\_Id INT NOT NULL,

Arrival\_Time TIME,

Arrival\_Date Date,

Arrival\_Bus\_Station NVARCHAR(255),

Departure\_Time TIME NOT NULL,

Departure\_Date DATE NOT NULL,

Departure\_Station NVARCHAR(255) NOT NULL,

FOREIGN KEY (Bus\_Id) REFERENCES BUS(Bus\_ID),

FOREIGN KEY (Departure\_Station) REFERENCES BUS\_STATION(Station\_Name),

FOREIGN KEY (Arrival\_Bus\_Station) REFERENCES BUS\_STATION(Station\_Name),

FOREIGN KEY (Journey\_Id) REFERENCES Journey(Journey\_Id),

);

CREATE TABLE BUS\_STOPS (

Station\_Name NVARCHAR(255) NOT NULL,

Bus\_Journey\_Id INT NOT NULL,

Arrival\_Time TIME,

Arrival\_Date DATE,

Departure\_Time TIME,

Departure\_Date DATE,

PRIMARY KEY (Station\_Name, Bus\_Journey\_Id),

FOREIGN KEY (Station\_Name) REFERENCES BUS\_STATION(Station\_Name),

FOREIGN KEY (Bus\_Journey\_Id) REFERENCES BUS\_JOURNEY(Journey\_Id)

);

CREATE TABLE VEHICLE (

Vehicle\_ID INT PRIMARY KEY NOT NULL,

Model NVARCHAR(50) NOT NULL,

Fuel\_Type NVARCHAR(50) NOT NULL,

Deposite\_Fees DECIMAL(10, 2) NOT NULL,

Car\_Brand NVARCHAR(50) NOT NULL,

Car\_Class NVARCHAR(50) NOT NULL,

Locations NVARCHAR(255) NOT NULL

);

CREATE TABLE VEHICLE\_JOURNEY (

Journey\_Id INT PRIMARY KEY NOT NULL CHECK (Journey\_Id >= 3000 AND Journey\_Id < 4000),

Vehicle\_Id INT NOT NULL,

FOREIGN KEY (Vehicle\_Id) REFERENCES VEHICLE(Vehicle\_Id),

FOREIGN KEY (Journey\_Id) REFERENCES Journey(Journey\_Id),

);

CREATE TABLE FERRY (

Ferry\_Id INT PRIMARY KEY NOT NULL,

Fuel\_Type NVARCHAR(50) NOT NULL,

Ferry\_Type NVARCHAR(50) NOT NULL,

Seat\_Count INT NOT NULL

);

--Port Kelimesi Uygulamanın Kendi içinde tanımlı olduğu için HARBOR olarak tanımladık.

CREATE TABLE HARBOR (

Port\_Name NVARCHAR(255) PRIMARY KEY NOT NULL

);

CREATE TABLE FERRY\_JOURNEY (

Journey\_ID INT PRIMARY KEY NOT NULL CHECK (Journey\_Id >= 4000 AND Journey\_Id < 5000),

Ferry\_ID INT NOT NULL,

Arrival\_Time TIME NOT NULL,

Arrival\_Date DATE NOT NULL,

Arrival\_Port NVARCHAR(255) NOT NULL,

Departure\_Time TIME NOT NULL,

Departure\_Date DATE NOT NULL,

Departure\_Port NVARCHAR(255) NOT NULL,

FOREIGN KEY (Arrival\_Port) REFERENCES HARBOR(Port\_Name),

FOREIGN KEY (Departure\_Port) REFERENCES HARBOR(Port\_Name),

FOREIGN KEY (Ferry\_ID) REFERENCES FERRY(Ferry\_Id),

FOREIGN KEY (Journey\_Id) REFERENCES Journey(Journey\_Id),

);

CREATE TABLE FERRY\_STOPS (

Ferry\_Journey\_ID INT NOT NULL,

Port\_Name NVARCHAR(255) NOT NULL,

Arrival\_Time TIME NOT NULL,

Arrival\_Date DATE NOT NULL,

Departure\_Time TIME NOT NULL,

Departure\_Date DATE NOT NULL,

PRIMARY KEY (Ferry\_Journey\_ID, Port\_Name),

FOREIGN KEY (Ferry\_Journey\_ID) REFERENCES FERRY\_JOURNEY(Journey\_ID),

FOREIGN KEY (Port\_Name) REFERENCES HARBOR(Port\_Name)

);

CREATE TABLE METRO\_TRAIN (

Train\_Number INT PRIMARY KEY NOT NULL,

Train\_Type NVARCHAR(50) NOT NULL,

Vagon\_Type NVARCHAR(50) NOT NULL,

Seat\_Count INT NOT NULL

);

CREATE TABLE METRO\_STATION (

Station\_Name NVARCHAR(255) PRIMARY KEY NOT NULL

);

CREATE TABLE METRO\_JOURNEY (

Journey\_ID INT PRIMARY KEY NOT NULL CHECK (Journey\_Id >= 5000 AND Journey\_Id < 6000),

Train\_Number INT NOT NULL,

Arrival\_Time TIME NOT NULL,

Arrival\_Date TIME NOT NULL,

Arrival\_Station NVARCHAR(255) NOT NULL,

Departure\_Time TIME NOT NULL,

Departure\_Date DATE NOT NULL,

Departure\_Station NVARCHAR(255) NOT NULL,

FOREIGN KEY (Train\_Number) REFERENCES METRO\_TRAIN(Train\_Number),

FOREIGN KEY (Arrival\_Station) REFERENCES METRO\_STATION(Station\_Name),

FOREIGN KEY (Departure\_Station) REFERENCES METRO\_STATION(Station\_Name),

FOREIGN KEY (Journey\_Id) REFERENCES Journey(Journey\_Id)

);

CREATE TABLE METRO\_STOPS (

Metro\_Journey\_ID INT NOT NULL,

Station\_Name NVARCHAR(255) NOT NULL,

Arrival\_Time TIME,

Arrival\_Date DATE,

Departure\_Time TIME,

Departure\_Date DATE,

PRIMARY KEY (Metro\_Journey\_ID, Station\_Name),

FOREIGN KEY (Metro\_Journey\_ID) REFERENCES METRO\_JOURNEY(Journey\_ID),

FOREIGN KEY (Station\_Name) REFERENCES METRO\_STATION(Station\_Name)

);

CREATE TABLE MARTI (

Marti\_ID INT PRIMARY KEY NOT NULL,

Battery INT NOT NULL CHECK (Battery BETWEEN 0 AND 100),

Price DECIMAL(10, 2) NOT NULL,

Location\_X DECIMAL(9,6) NOT NULL,

Location\_Y DECIMAL(9,6) NOT NULL,

Type NVARCHAR(50) NOT NULL

);

CREATE TABLE MARTI\_JOURNEY (

Journey\_ID INT PRIMARY KEY NOT NULL CHECK (Journey\_Id >= 7000 AND Journey\_Id < 8000),

Marti\_ID INT NOT NULL,

FOREIGN KEY (Marti\_ID) REFERENCES MARTI(Marti\_ID),

FOREIGN KEY (Journey\_Id) REFERENCES Journey(Journey\_Id)

);

CREATE TABLE HIGH\_SPEED\_TRAIN (

Train\_Number INT PRIMARY KEY NOT NULL,

Train\_Type NVARCHAR(50) NOT NULL,

Vagon\_Type NVARCHAR(50) NOT NULL,

Seat\_Count INT NOT NULL

);

CREATE TABLE HIGH\_SPEED\_TRAIN\_STATION (

Station\_Name NVARCHAR(255) PRIMARY KEY NOT NULL

);

CREATE TABLE HIGH\_SPEED\_TRAIN\_JOURNEY (

Journey\_ID INT PRIMARY KEY NOT NULL CHECK (Journey\_Id >= 6000 AND Journey\_Id < 7000),

Train\_Number INT NOT NULL,

Arrival\_Time TIME NOT NULL,

Arrival\_Date TIME NOT NULL,

Arrival\_Station NVARCHAR(255) NOT NULL,

Departure\_Time TIME NOT NULL,

Departure\_Date DATE NOT NULL,

Departure\_Station NVARCHAR(255) NOT NULL,

FOREIGN KEY (Train\_Number) REFERENCES HIGH\_SPEED\_TRAIN(Train\_Number),

FOREIGN KEY (Arrival\_Station) REFERENCES HIGH\_SPEED\_TRAIN\_STATION(Station\_Name),

FOREIGN KEY (Departure\_Station) REFERENCES HIGH\_SPEED\_TRAIN\_STATION(Station\_Name),

FOREIGN KEY (Journey\_Id) REFERENCES Journey(Journey\_Id)

);

CREATE TABLE HIGH\_SPEED\_TRAIN\_STOPS (

HIGH\_SPEED\_TRAIN\_Journey\_ID INT NOT NULL,

Station\_Name NVARCHAR(255) NOT NULL,

Arrival\_Time TIME,

Arrival\_Date DATE,

Departure\_Time TIME,

Departure\_Date DATE,

PRIMARY KEY (HIGH\_SPEED\_TRAIN\_Journey\_ID, Station\_Name),

FOREIGN KEY (HIGH\_SPEED\_TRAIN\_Journey\_ID) REFERENCES HIGH\_SPEED\_TRAIN\_JOURNEY(Journey\_ID),

FOREIGN KEY (Station\_Name) REFERENCES HIGH\_SPEED\_TRAIN\_STATION(Station\_Name)

);

--CITIZEN CARD dan veri silindiğinde persondaki foreign keye yansıması

/\*

DELETE

FROM CITIZEN\_CARD

WHERE Card\_ID = 6;

SELECT \*

FROM PERSON

\*/

--Company Name izban olan yolculukların şriket ismi silindi, default değer NO Name atandı.

/\*

DELETE

FROM COMPANY

WHERE Company\_Name = 'İzban'

SELECT \*

FROM JOURNEY

\*/

--Company den Location\_Id = 10 değerini sildik.

/\*

DELETE

FROM LOCATIONS

WHERE Company\_Name = 'Garenta Araç Kiralama' And Address = 'Beşiktaş Mah., İstanbul'

SELECT \*

FROM LOCATIONS

\*/

-- CITIZEN CARD dan veri güncellendiğinde persondaki foreign keye yansıması

/\*

UPDATE CITIZEN\_CARD

SET Card\_ID = 30

WHERE Card\_ID = 3;

SELECT \*

FROM PERSON

\*/

-- Plane\_Journey uçak veri tipi update örneği

/\*

UPDATE PLANE\_JOURNEY

SET Aircraft\_Type = 'Boeing 737'

WHERE Journey\_Id = 1002;

SELECT \*

FROM PLANE\_JOURNEY

\*/

-- Update Örneği. Aktarmalı uçuşların ini sürelerini günceller.

-- Direkt Where IS NOT NULL ile de yapılabilirdi, çeşitlilik olsun diye detaylandırıldı.

/\*

UPDATE PLANE\_JOURNEY

SET Arrival\_Date = '2013-11-01'

WHERE Next\_Journey\_ID in (

Select Next\_Journey\_ID

From PLANE\_JOURNEY

Where Next\_Journey\_ID IS NOT NULL

)

SELECT \*

FROM PLANE\_JOURNEY

\*/

-- 2 Tablolu SELECT örneği

/\*

SELECT \*

FROM PERSON AS P, TRANSACTIONS AS T

WHERE P.TC = T.TC AND P.Name = 'Elif'

\*/

-- 2li Sorgu Örneği

-- Kadın kullanıcıların Arama geçmişi

/\*

SELECT DISTINCT P.Name, P.Surname, S.\*

FROM PERSON AS P, SEARCH\_HISTORY AS S

WHERE P.TC = S.TC AND P.Gender = 'Kadın'

\*/

-- 3 Tablolu Örnek

-- Ferry Journey yapan kişileri döndürür.

/\*

SELECT

P.Name, P.Surname , P.TC

FROM

PERSON AS P

WHERE

EXISTS (

SELECT

1

FROM

TRANSACTIONS AS T, FERRY\_JOURNEY AS FJ

WHERE

T.TC = P.TC and T.Journey\_Id = FJ.Journey\_Id

);

\*/

-- 3 Tablolu Örnek

-- Antalya Havalimanında 2023-07-11 tarihinde 100 lira üstündeki işlemin bilgileri.

/\*

SELECT

T.Transaction\_Id,

T.Total\_Price,

T.Purchase\_Date,

J.Journey\_Id,

J.Company\_Name,

P.Name,

P.Surname,

P.TC

FROM

PERSON AS P,

TRANSACTIONS AS T

INNER JOIN

JOURNEY AS J ON T.Journey\_Id = J.Journey\_Id

WHERE

T.Purchase\_Date = '2023-07-11' AND

T.Total\_Price > 100.00 AND

J.Journey\_Id IN (

SELECT

Journey\_Id

FROM

PLANE\_JOURNEY

WHERE

Departure\_Airport\_Name = 'Antalya Havalimanı' And P.TC = T.TC

);\*/

-- 3 Tablolu Örnek

--Türk Havayollarınının 7000 altında max renge sahip uçaklarının bilgileri

/\*

SELECT A.\*

FROM JOURNEY AS J, AIRPLANE AS A

WHERE J.Company\_Name = 'Türk Hava Yolları' AND J.Journey\_Id IN (

SELECT P.Journey\_Id

FROM PLANE\_JOURNEY AS P

WHERE P.Aircraft\_Type = A.Aircraft\_Type AND A.Max\_Range < 7000

)

\*/

------------------ View örneği

CREATE VIEW ALL\_TICKET

AS SELECT P.Name,

P.Surname,

P.Phone,

T.Journey\_Id,

T.Seat\_Number

FROM PERSON AS P, TRANSACTIONS AS T

WHERE P.TC = T.TC

------------------

SELECT PERSON.Name ,PERSON.Surname ,SH.\*

FROM PERSON, TRANSACTIONS AS T

JOIN SEARCH\_HISTORY AS SH

ON T.TC = SH.TC

WHERE PERSON.TC = T.TC

AND T.Journey\_Id

IN

(SELECT T.Journey\_Id

FROM HIGH\_SPEED\_TRAIN\_STOPS as HS

JOIN TRANSACTIONS AS T

ON HS.HIGH\_SPEED\_TRAIN\_JOURNEY\_ID =T.Journey\_Id

WHERE HS.Station\_Name = 'Polatlı Gar')

TRİGGER

CREATE SEQUENCE Seq\_TransactionHistoryId

AS INT

START WITH 1

INCREMENT BY 1;

CREATE TRIGGER trg\_InsertTransactionHistory

ON TRANSACTIONS

AFTER INSERT

AS

BEGIN

INSERT INTO TRANSACTION\_HISTORY (History\_Id, TC, Transaction\_Id, Journey\_Id)

SELECT

NEXT VALUE FOR Seq\_TransactionHistoryId, -- Sequence'dan sonraki değeri alır

i.TC,

i.Transaction\_Id,

i.Journey\_Id

FROM inserted i;

END;

-----------------

CREATE TRIGGER trg\_UpdateCitizenCardBalance

ON TRANSACTIONS

AFTER INSERT

AS

BEGIN

UPDATE CITIZEN\_CARD

SET Money\_Amount = Money\_Amount - i.Total\_Price

FROM inserted i

INNER JOIN PERSON p ON i.TC = p.TC

INNER JOIN CITIZEN\_CARD cc ON p.Card\_ID = cc.Card\_ID

WHERE cc.Card\_ID = p.Card\_ID;

END;

----------------

CREATE TRIGGER trg\_vehiclekonum

ON TRANSACTIONS

AFTER INSERT

AS

BEGIN

UPDATE VEHICLE

SET VEHICLE.Locations = i.Receive\_Place

FROM VEHICLE V

INNER JOIN VEHICLE\_JOURNEY VJ ON V.Vehicle\_ID = VJ.Vehicle\_ID

INNER JOIN inserted i ON VJ.Journey\_Id = i.Journey\_Id;

END;

--------------------

CREATE TRIGGER trg\_martıkonum

ON TRANSACTIONS

AFTER INSERT

AS

BEGIN

UPDATE MARTI

SET MARTI.Location\_X = i.Receive\_Place\_X, MARTI.Location\_Y = i.Receive\_Place\_Y

FROM MARTI

INNER JOIN MARTI\_JOURNEY ON MARTI.Marti\_ID = MARTI\_JOURNEY.Marti\_ID

INNER JOIN inserted i ON MARTI\_JOURNEY.Journey\_Id = i.Journey\_Id;

END;

-----------------------

CREATE TRIGGER trg\_DecreaseMartiBattery4

ON TRANSACTIONS

AFTER INSERT

AS

BEGIN

-- Batarya seviyesini rastgele bir miktarla azalt

UPDATE MARTI

SET Battery = CASE

-- RAND() ile 5 ile 100 arasında rastgele bir değer üret

WHEN Battery - CAST((RAND(CHECKSUM(NEWID())) \* 95 + 5) AS INT) < 0 THEN 0

ELSE Battery - CAST((RAND(CHECKSUM(NEWID())) \* 95 + 5) AS INT)

END

FROM MARTI

INNER JOIN MARTI\_JOURNEY ON MARTI.Marti\_ID = MARTI\_JOURNEY.Marti\_ID

INNER JOIN inserted i ON MARTI\_JOURNEY.Journey\_Id = i.Journey\_Id;

END;

------------------------------