DATABASE MANAGEMENT

Extending a Conceptual Design

25 February 20221

Schema diagram for AIRLINE relational database schema.

AIRPORT

Airport_code	Name	City	State

FLIGHT

Flight_number	Airline	Weekdays

FLIGHT_LEG

Flight_number	Leg_number	Departure_airport_code	Scheduled_departure_time	Arrival_airport_code	Scheduled_arrival_time

LEG_INSTANCE

Flight_number	Leg_number	Date	Num_of_available_seats	Airplane_id	Departure_airport_code	Departure_time	Arrival_airport_code	Arrival_time

FARE

Flight_number	Fare_code	Amount	Restrictions

AIRPLANE_TYPE

Airplane_type_name	Max_seats	Company

CAN_LAND

Airplane_type_name	Airport_code

AIRPLANE

Airplane_id	Total_number_of_seats	Airplane_type

SEAT_RESERVATION

Flight_number	Leg_number	Date	Seat_number	Passport_number	Country

COSTUMER

Passport_number	Country	Costumer_name	Customer_phone	Address	Email

FFC

Passport_number	Country	Mileage

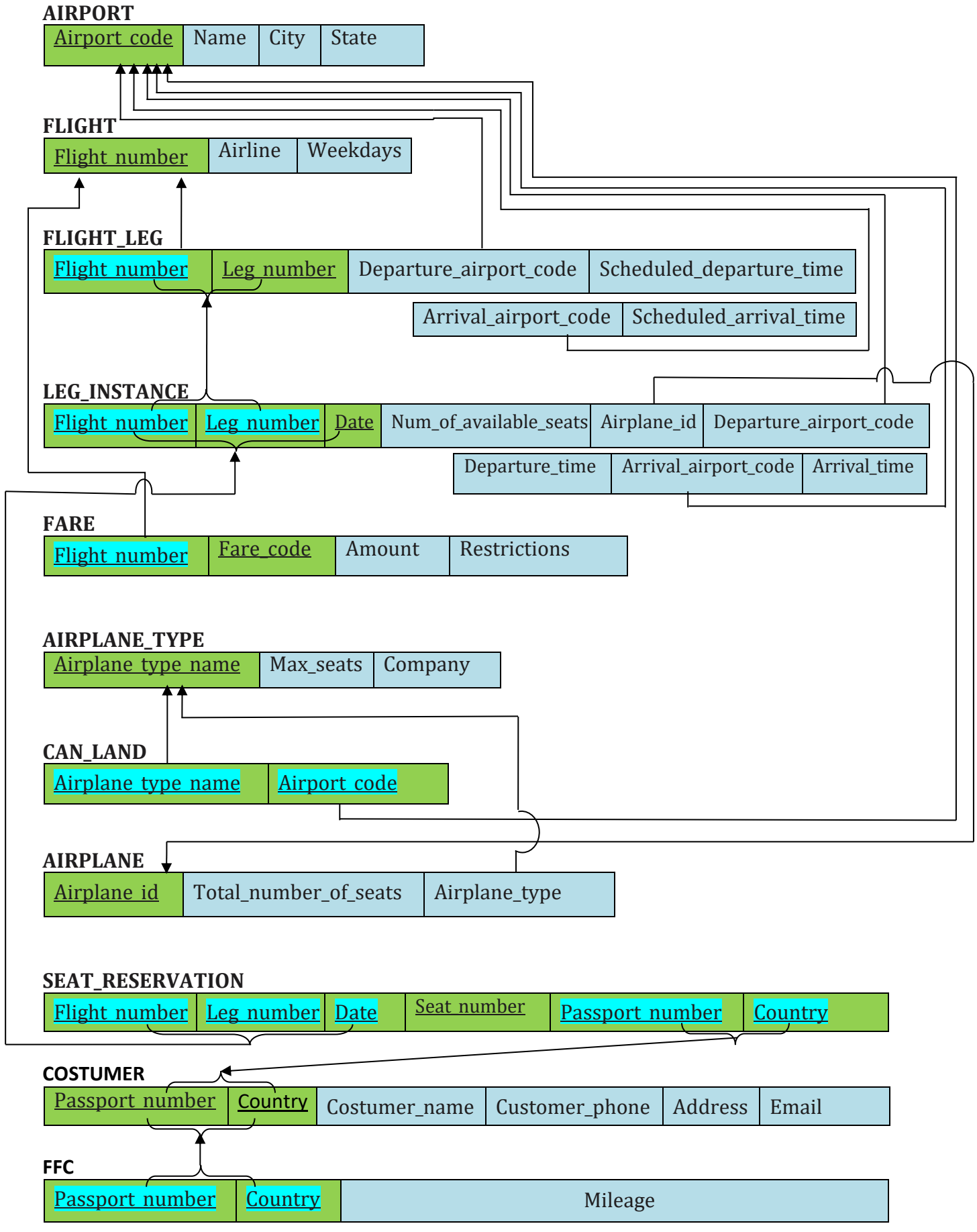
According to model, the following requirements are satisfied:

The AIRLINE relational database schema shown in the previous page's figure describes a database for airline flight information. Each FLIGHT is identified by a Flight number and consists of one or more FLIGHT_LEGs with Leg numbers 1, 2, 3, and so on. Each FLIGHT_LEG has scheduled arrival and departure times, airports, and one or more LEG_INSTANCEs—one for each Date on which the flight travels. FAREs are kept for each FLIGHT. For each FLIGHT_LEG instance, SEAT_RESERVATIONs are kept, as are the AIRPLANE used on the leg and the actual arrival and departure times and airports. An AIRPLANE is identified by an Airplane_id and is of a particular AIRPLANE_TYPE. CAN_LAND relates AIRPLANE_TYPEs to the AIRPORTs at which they can land. An Airport_code identifies an AIRPORT.
Passport number and country identify each customer, customer's personal information such as name, phone number, address and email are also kept here in this entity. We also have an entity called FFC (stands for frequent flyer customer), which keep track of the customer's flight information; If a customer has checked-in physically a flight create a transaction record with the mileage information assigned to that flight leg. There is a COMPANY entity for both AIRLINE and AIRPLANE, in which we store information about the aircraft manufacturers (BOENG, AIRBUS…) and airline companies (Lufthansa, Turkish Airlines, Qatar Airways…). We have two separate entities as a subclass for both aircraft manufacturer and airline company (using generalization/ specialization hierarchy), we used disjoint there, which obviously means a company cannot be an airline and airplane manufacturer at the same time…

Specialization/ Generalization:

Bir Company entity'si yarattık. Company entitysi iki farklı entity'i ifade ediyor. Bunlar Airplane_manufucturer ve Airline entity'leri. Burada generalization/specialization hiyerarşisini kullandık. Bu bağlantıda Airplane_manufucturer da bir Company'dir, Airline entity'si de bir Company'i ifade etmektedir. Fakat bir Company aynı anda hem Airplane_manufucturer hem de Airline için sağlanmıyor. Bu yüzden bu hiyerarşiyi disjoint olarak belirledik. Airplane_manufucturer ve Airplane'in de diyagramımızda devam eden bağlantıları vardır.

Referential integrity constraints displayed on the AIRLINE relational database schema.



Draw an EER diagram for AIRLINE relational database. (Database reverse engineering)

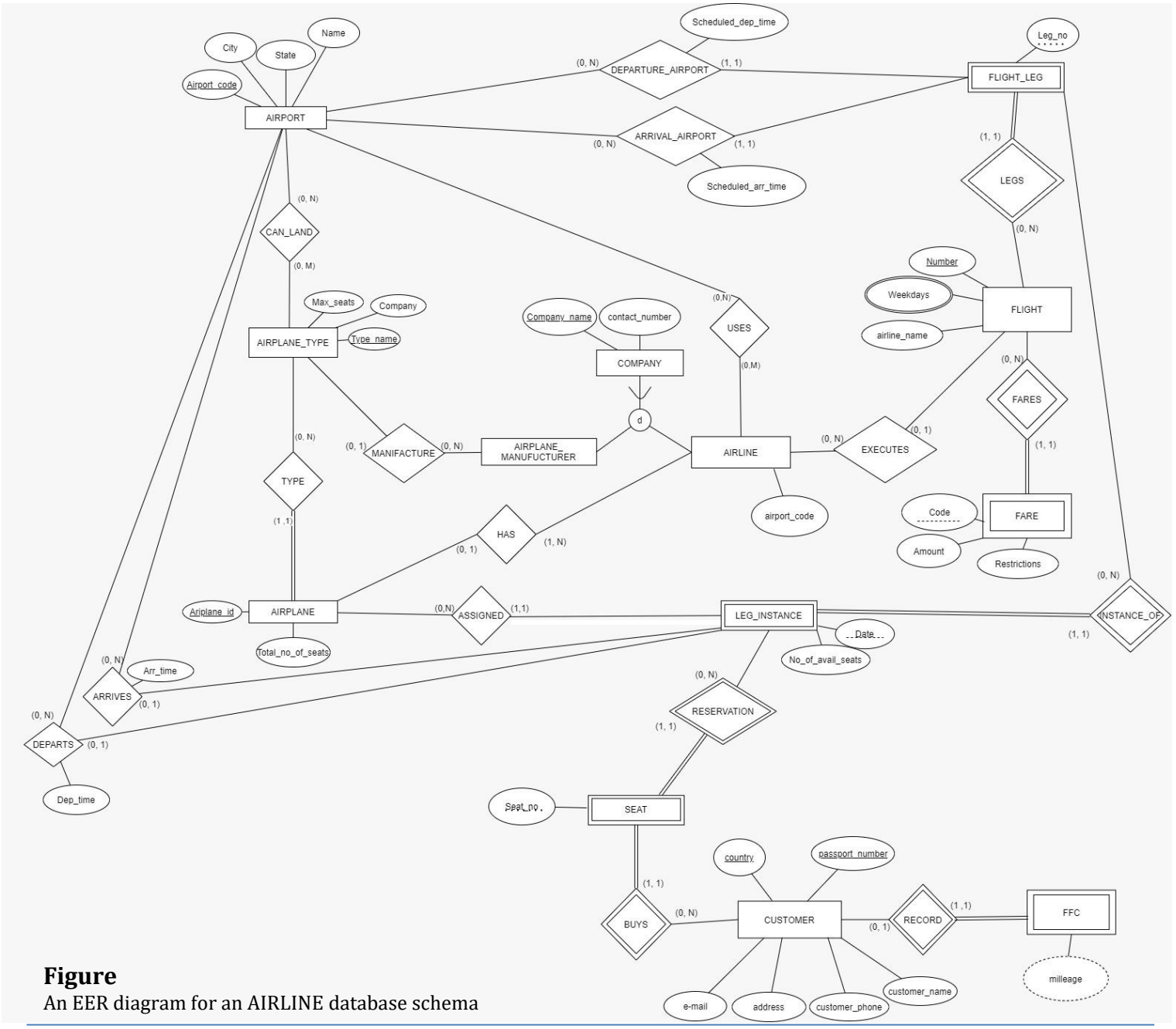


Figure
An EER diagram for an AIRLINE database schema

Türkiye’de 56 tane havaalanı bulunmaktadır. Havaalanları ise onlarca havayolu şirketine sahiplik eder. Türkiye’de 12 tane havayolu şirketi bulunuyor ve Türkiye’deki havaalanlarına 50’den fazla havayolu şirketi iniş ve kalkış gerçekleştiriyor. Dünyada ise bu havayolu şirketlerine uçak kiralayan veya satan onlarca uçak üreticisi vardır. Bunlardan en ünlü ikisi ise Boeing ve Airbus. 2019 yılı sonunda yapılan bir araştırmaya göre, 2019 yılında toplam 4.4 milyar kişi sivil havacılık ile seyahat etti ve bu uçuşlar için toplam 845 milyar dolar harcadılar. İstanbul Havalimanında ise 2019 yılı içinde 86 günlük (6 Nisan- 1 Temmuz) bir aralıkta 74 saniyede bir uçak seferi gerçekleşti.

Genellikle ülkeler arasında olan uçuşlarda aktarmalı uçuşlar bulunmaktadır. Bu uçuşlar müşteriler için zorluk çıkarsa da havayolu şirketleri için daha az maliyetli tüketimi ve en optimum uçuş takvimini sağlar. Bu takvimlerin oluşturulabilmesi için havayolu şirketleri hem kendi aralarında hem de havaalanı yönetimleri ile korelasyon içinde olmak zorundadır. Bu zorlu ve büyük verileri düzenleyebilmek için güçlü bir veri yönetimi sistemi ve kaliteli iletişimin sağlanması gerekir.

Biz de grupça bu veri yönetimini, en optimum şekilde gerçekleştirebilmek adına bizden istenilen temel gereksinimleri ve sivil havacılık sektöründeki gereksinimleri göz önüne alarak bir veri yönetimi sisteminin EER diyagramını hazırladık.

Bu diyagramın bazı mantıksal eksiklikleri bulunmaktadır ve sivil havacılık seferlerinin bütün ihtiyaçlarını karşılayabilecek bir düzeyde değildir. Bunlar:

- Normal hayatta aynı uçakta uçuş yapan insanların uçuş numaraları aynı olur fakat bizim tasarladığımız konseptte eğer ki içinde bulunulan uçuştan sonra devam eden uçuşu bulunan yolcular için(aktarmalı uçuşlu bilete sahip yolcular) farklı bir numara atanır. Bu da aynı uçakta bulunan yolcular için ufak bir karışıklığa sebep olabilir. Fakat işleyiş açısından herhangi bir problem barındırmamaktadır.
- Genellikle yurt dışı uçuşlarda karşılaşılan, aktarmadan aktarmaya havayolu şirketi değişimi durumu modelimizde desteklenmemektedir. Bir yolcu hangi havayolu şirketi ile uçuşa başladıysa o şirket ile aktarmalarını tamamlamak zorundadır.
- Normal hayatta bilet satın alımı sırasında koltuk numarası seçimi yapılmamaktadır. Bu işlem biletin yolcuya teslim edildiğin anda gerçekleştirilir ve genellikle rastgele şekilde atama yapılır(Ekonomi sınıfı yolcular için karşılaşılan bir problem). Bizim modelimizde ise böyle bir şey mümkün değildir. Bilet ödemesi alındığı anda yolcuya bir koltuk numarası atanır. Yani yolcunun bilete sahip olabilmesi için bir koltuk numarasına sahip olması zorunludur.
- Çizilen diyagram aynı uçuşa sahip iki insanın aynı numaralı koltuğa atanmasını engelleyemiyor.
- Aynı Leg_number'a ve Date'e sahip bir uçak bir noktadan bir noktaya gittiğini düşünelim. Sistemde Bu uçağı 2 kez ekleyip aynı uçağı 2 farklı noktaya göndermek mümkün. Gerçek hayatta bu mümkün değildir.Örneğin: İzmirden Ankaraya uçan bir uçak sisteme eklendi. Aynı Leg_number'a, Date'e ve Airplane_id'e sahip ve farklı Flight_number'a sahip bir uçak yine İzmirden Ankaraya uçacak şekilde sisteme ekleyebiliriz. Böyle aynı uçak aynı tarihte 2 kere uçmuş oluyor.

İlişki Tipleri:

COMPANY, AIRLINE ve AIRLANE MANUFACTURER'in üst sınıfıdır.

FLIGHT_LEG, FLIGHT'a göre weak entity'dir.

LEG_INSTANCE, FLIGHT_LEG'e göre weak entity'dir.

FLIGHT_LEG, LEG_INSTANCE'e göre strong entity'dir.

SEAT, LEG_INSTANCE'a göre weak entity'dir.

LEG_INSTANCE, SEAT'a göre strong entity'dir.

CUSTOMER, SEAT'e göre strong entity'dir.

CUSTOMER, FFC'ye göre strong entity'dir.