Travel Agency Management System and Database

Group 118

Eric Cheng, Gordon Ng, Samuel Ren, Greta Zu

I. Requirement Analysis

Introduction

Purpose

This application is a simplified version of a multipurpose travel booking website, where users have the ability to book flights, hotel rooms or rental cars. Its primary objective is to serve as a common point of service for multiple areas of tourism industries, streamlining the travel experience of its users.

Scope and Special Requirements

The scope of our application is global, with a focus on major cities and tourist destinations. Requirements for the database portion includes language support, payment processing and the ability to handle peak travel loads by making the database operate as efficiently as possible. Additionally, scalability is a key factor in order to be able to add new destinations, services and carriers as demand shifts or grows.

Description of the Data Requirements

Entities:

- **Users**: A user can be anyone using the application to book either a hotel reservation, a flight, or a car rental. All users need to provide a *phone number*, their *credit card information*, their *name*, *address* and *email*. Each user is assigned and identified by a unique *user ID*.
- **Registered**: Some users can choose to be registered instead of being a guest user. They have all information stored as well as additional information such as their *booking history*, *language* preference, *password* and *username*.
- **Cities**: Cities are identified by their *city name* and *country* together. Cities also have the *airport name* as an attribute. Flight departures and arrivals can take place in various cities. Hotels and cars are in specific cities as well.
- **Flights**: Flights are identified by a *flight number*. For each flight, the system keeps track of the *airline policy*, the *airline* name, the *number of seats available per fare class*, the *arrival* and *departure* dates and time, the *ticket costs per fare class* and the *airplane model*.

- **Flight Booking**: A flight booking is identified by a *flight reference number*, which must be associated with the specific flight the user booked. The system additionally keeps track of the *seat numbers*, the *plane ticket cost*, the *plane ticket surcharge*, the *fare class*, the *passenger names*, the *flight booking date*, the *flight booking fees* and the *plane ticket tax*.
- **Hotel Booking**: Users can also make reservations for hotels. A hotel booking is identified by the *hotel reference number* associated with the hotel chosen by the user. The system also keeps track of the *room cost*, the *hotel tax*, if *breakfast* is included, the *hotel booking fee* and the *hotel booking date*.
- **Hotel**: A hotel is identified by the *brand affiliation* and its *address*. The system also keeps track of the *hotel's policy*, the *restaurants* and *business facilities* in the hotel, and the *listing name*. It also indicates whether the following services are offered: *on-site parking*, *pet allowance*, *pool*, *fitness center* and *airport shuttle*.
- **Room**: A room inside a hotel is identified by a *room number*. The system keeps track of the *room price*, the *room capacity*, the *size*, the *beds*, the *room name* and the *view* available. It also indicates if the room contains a *minibar* or a *private bathroom*, if *smoking* is allowed and whether there is *free wifi*.
- **Car rental booking**: Finally, users can also book cars. A car rental booking is identified by the *car rental reference number* associated with the car chosen by the user. The system keeps track of the *car rental cost*, the *insurance*, the *car rental booking fees*, the *car rental tax*, the *car rental booking date*, the *pickup location*, the *return location*, the *pickup* and *return* dates and time.
- **Car**: A car is identified by the *car license plate*. The system keeps track of the *company policy*, the *car model*, the *transmission type*, the *car rental cost,* the *car engine type*, the *number of seats* and the *car rental agency*. It also indicates if the car has air conditioning (*AC*) and *CarPlay*.

Functional Requirements

Relationships:

- **flight reservation**: Users can make flight reservations. Each flight reservation has a total cost.
    - Many-to-one Relationship: Many flight bookings can be associated with a single User.
    - Participation Constraint: Each flight booking must be associated with a single User
- **hotel reservation**: Users can also make hotel reservations. Each hotel reservation has a total cost.
    - Many-to-one Relationship: Many hotel bookings can be associated with a single User.
    - Participation Constraint: Each hotel booking must be associated to a single User

- **car rental reservation**: Finally, users can also make car rental reservations. Each car rental reservation has a total cost.
    - Many-to-one Relationship: Many car rental bookings can be associated with a single User.
    - Participation Constraint: Each car rental booking must be associated to a single User
- **flight booked**:
    - Many-to-one Relationship: Each flight has many bookings, but a flight booking is only associated with a single flight.
    - Participation Constraint: Each flight booking is associated to a single flight containing more specific information on the flight booked.
- **room booked**:
    - One-to-one Relationship: Each room can only be booked by one user at a time.
    - Participation Constraint: Each room booking is associated to a single room containing specific information on the room booked.
- **car booked**:
    - One-to-one Relationship: since a car can only be booked by a single user.
    - Participation Constraint: Each car rental booking is associated with a single car with specific information on the car being rented.
- **route**: Each flight has route information in relation to cities, indicating the departure city and the arrival city.
    - Many-to-many Relationship: A city can be the departure city for many flights and also the arrival city for many flights.
    - Participation Constraint: Each flight must be associated with a single route since it defines departure and arrival locations of the flight.
- **located in**:
    - Many-to-one Relationship: Each city may have many hotels while a specific hotel can only be located in a single city
    - Participation Constraint: Each hotel must be associated with a single city.
- **belongs to**:
    - Many-to-one Relationship: Each city can have many cars up for rental.
    - Participation Constraint: Each car must be associated with a city.
- **part of**: A room is a weak entity, since rooms in different hotels may have the same room number.

- Many-to-one Relationship: Each room is a part of a specific hotel, while a hotel can hold many rooms for rental.
- Participation Constraint: Each room is part of a specific hotel and the system uses the hotel keys to help identify a specific room.

II. E/R Diagram

See separate file

III. Relations

Entities:

Users (<u>user ID</u>, phone number, credit card information, name, address, email)

- NOT NULL: phone number, credit card information, name, address, email

Registered (<u>user ID</u>, history, language, password, username)

- Foreign Key: user ID references Users
- NOT NULL: password, username

Flight Booking (<u>flight reference number</u>, seat numbers, plane ticket cost, plane ticket surcharge, fare class, passenger names, flight booking date, flight booking fees, plane ticket tax, flight total cost, user ID, flight number)

- Foreign Key: user ID references Users (related through *flight reservation*), flight number references Flights (related through *flight booked*)
- NOT NULL: seat numbers, plane ticket cost, plane ticket surcharge, fare class, passenger names, flight booking date, flight booking fees, plane ticket tax, flight total cost, user ID, flight number

Flights (<u>flight number</u>, airline policy, airline, seats available per fare class, arrival date/time, departure date/time, ticket cost per fare class, airplane model, departure city name, departure country, arrival city name, arrival country)

- Foreign Key: departure city name and arrival city name reference Cities (city name); departure country and arrival country reference Cities (country) (all related through *route*)
- NOT NULL: airline policy, airline, seats available per fare class, arrival date/time, departure date/time, ticket cost per fare class, airplane model, departure city name, departure country, arrival city name, arrival country

Hotel Booking (<u>hotel reference number</u>, room cost, hotel tax, breakfast inclusion, hotel booking date, hotel booking fees, hotel total cost, user ID, room number, brand affiliation, hotel address)

- Foreign Key: user ID references Users (related through *hotel reservation*), room number references Room (related through *room booked*), brand affiliation and hotel address reference Hotel (weak entity relation)
- NOT NULL: room cost, hotel tax, hotel booking date, hotel booking fees, hotel total cost, user ID, room number, brand affiliation, hotel address

Hotel (brand affiliation, hotel address, hotel policy, airport shuttle, business facilities, restaurants, listing name, fitness center, on-site parking, pet allowance, pool, city name, country)

- Foreign Key: city name and country reference Cities (related through *located in*)
- NOT NULL: hotel policy, listing name, city name, country

Cities (city name, country, airport name)

Car Rental Booking (car rental reference number, car rental cost, insurance, car rental booking fees, car rental tax, car rental booking date, pickup location, return location, pickup date/time, return date/time, car license plate, user ID, car rental total cost)

- Foreign Key: car license plate references Car (related through *car booked*), user ID references Users (related through *car rental reservation*)
- NOT NULL: car rental cost, insurance, car rental booking fees, car rental tax, car rental booking date, pickup location, return location, pickup date/time, return date/time, car license plate, user ID, car rental total cost

Car (car license plate, car rental agency, AC, number of seats, car engine type, car rental cost, CarPlay, transmission type, car model, company policy, city name, country)

- Foreign Key: city name and country reference Cities (related through *belongs to*)
- NOT NULL: car rental agency, number of seats, car engine type, car rental cost, transmission type, car model, company policy, city name, country

Weak Entity:

Room (hotel Address, brand affiliation, room number, room price, room capacity, minibar, private bathroom, smoking, room name, view, beds, availability, free wifi, size)

- Foreign Key: brand affiliation and hotel address reference Hotel (related through *part of*)
- NOT NULL: room price, room capacity, room name, beds, availability

Relationships:

Route (departure city name, departure country, arrival city name, arrival country)

We do not feel the need to combine any of the entities. For example, the different types of bookings each have many unique attributes and it is simply better to have them separated in order to not compromise the efficiency of the booking actions and the data relevant to each entity.

IV. Application Design

Application Description

Our application will function as a comprehensive search engine for hotel, flight and car bookings. Within our database system, we will store important information for each type of booking choices to help users find and compare diverse options, enabling them to select the most fitting option according to their interests. For example, if a user is looking for an affordable hotel in Montreal, our system will facilitate the comparison of prices across various hotels in the city. Since our system holds a lot of entities with many attributes and complex relations (an is-a hierarchy, a weak entity set and a ternary relationship set), there will be many sophisticated ways to manipulate the data stored in our system. Moreover, there are three main functional branches which will result in very different user interaction experiences once the application is built.

V. Inspiration and Work Process

Website:

https://www.expedia.ca/

Expedia works by interaction with other databases to retrieve offerings for its different types of services or to populate its own database. For our purposes, the booking options will be manually populated to have a usable database.

Work Process:

We split the project into 2 main parts:

- Finding an idea and producing the E/R diagram as well as the relations
- Redacting the project document

In an online group chat, we brainstormed ideas and came to an agreement. The group was then split into two to handle the different parts of the project. Over two online meetings, Greta and Eric worked on the diagram and schema. Once that was complete, Gordon and Sam wrote the report in a single online meeting.