Simple Airline Management System (SAMS)

CS4400: Introduction to Database Systems
Course Project: Spring 2025 Semester

Version History

Version	Date	Notes
0	January 15, 2025	Initial release

Problem Description & Motivation

The following is a text description of the system you are being tasked with developing. The system requirements – explicit and implicit – are included in this document, and they need to be identified and reflected in your system (i.e., source code), and in the associated design documents as required by the assignment instructions.

You will develop, implement, and test a system for airlines to keep track of which airplanes are flown to different destinations, the pilots operating the airplanes, and the movement of passengers between destinations. There are two key sections of the document below:

- **Problem Requirements:** This section describes the main elements of the "airline management" problem domain and contains the essential information that you must include in your design model.
- **Sample Data Elements:** This section includes some data values that you can use to help determine if your design artifacts are as correct, consistent, comprehensive, and concise as reasonably possible.

Problem Requirements

The primary aim of your system is to track the overall status of passenger-based commercial aircraft across a large region with multiple airports. This means that it will have to manage the data and operations related to various types of entities – namely, the airplanes, airports, and other things relevant to our scenario. The single-most important resource in our scenario is people. You must track people in your system as users who are acting in various capacities. People can either be aircraft pilots, or they can be passengers riding on one or more flights (but not both). All people must be either pilots or passengers – your system is not required to keep track of people acting in any other capacities. All people must have distinct identifiers in this system. People will also have first and last names. First names will be required, but some celebrities might travel using only their first name (e.g., Madonna, Beyonce); thus, last names are optional.

Attributes that are used to identify entities in your system will normally consist of fifty (50) or fewer alphanumeric characters in some regular pattern/format. This will be the default format for entity-identifying and unique entity attributes in your system unless otherwise noted. Dates will be provided in the "yyyy-mm-dd" date format by default unless otherwise noted, but you may store them internally in a different format if required. Your system must provide the dates in their default/original format as required. Some users might have relatively long first and/or last names, so your system must be able to manage first and last names that have one hundred (100) or fewer characters. Our default size for storing "general purpose" descriptive attributes will be one hundred (100) or fewer characters unless otherwise noted. Please note that these "early" data type specifications might be superseded once we've received more sample data from the customer in later phases of the project.

The main providers in our problem domain are the airlines. An airline is an organization that helps to manage and/or provide flights for passengers. Airlines own one or more airplanes that can be used to accomplish these tasks. Each airline has a unique identifier. Within our system, we will also keep track of the total revenue of each airline.

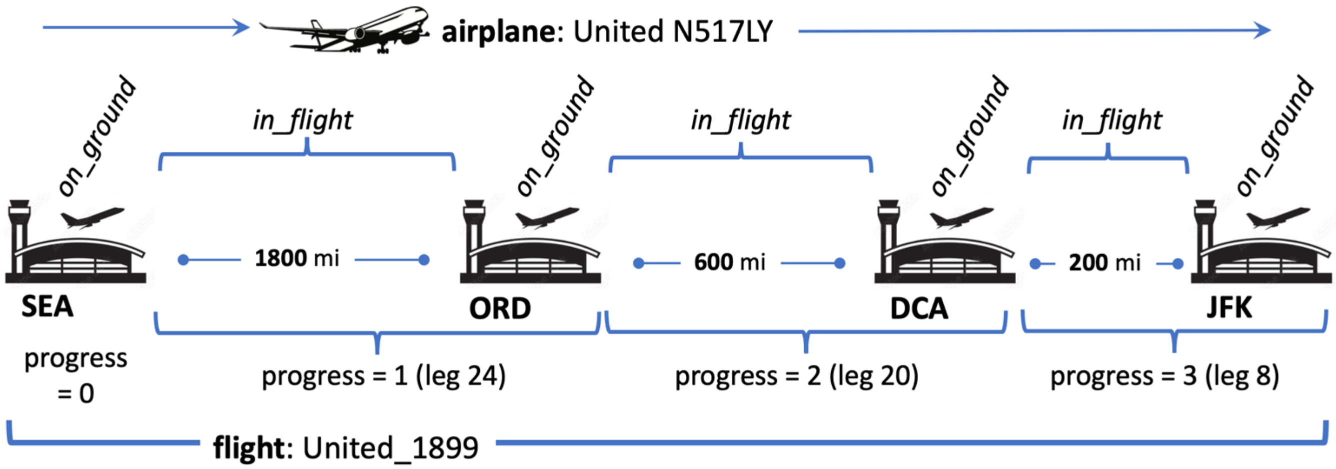
Airplanes are the craft used to transport passengers between airports. Airplanes can be identified uniquely by the combination of the airline that owns them and their tail number. Airplanes come in various types and capabilities, but most airplanes in this problem domain will be divided into two basic categories based on their manufacturer: Airbus airplanes and Boeing airplanes. Within these two types, they have different characteristics that must be monitored. Specifically, given recent events we are keeping track of the model of airplane for Boeing airplanes. Furthermore, it is important that we track whether the Boeing airplanes were serviced within the last month, not the specific date of service, to ensure passenger safety and for documentation with the FAA. To facilitate the maintenance of Airbus airplanes, we keep track of whether that airplane is a new engine option or 'neo' variant of the aircraft. Most airplanes in the system can be classified as a Boeing airplane or an Airbus airplane. Planes in the system that are not manufactured by either of these companies will not need specific data associated with them. All airplanes, regardless of manufacturer, will need a speed and a seating capacity.

Airplanes transport passengers from one "departure" airport to a different "arrival" airport. Airports have a three-letter identifier, along with a longer, more human-readable name – for example, "ATL" represents the Hartsfield-Jackson Atlanta International Airport. Each airport is in a specific city, state and country. Your system must track the location of each airport to determine which airports can be used to serve the same geographical regions. We will use the standard three-letter International Bank Account Number (i.e., IBAN) abbreviations for the various countries, e.g. USA, AUS, etc.

Airplanes will support flights on behalf of the airlines. Each flight will follow a specific route from some starting airport to some ending airport. Some flights are non-stop and go simply from the departure airport to the arrival airport. Other flights might follow routes that are made up of a specific sequence of multiple legs, where the airplane lands and then takes off again at various intermediate airports. Your system must track the distance for each leg of a flight – regardless of whether it is non-stop or multi-stop – so that it can determine the total duration of the flight based on the speed of the supporting airplane.

When an airplane is assigned to support a specific flight, your system must track the status of the airplane's progress along the flight's designated route. More specifically, it must track which leg of the route is currently being traversed; whether the airplane is on the ground or in the air; and when the next takeoff or landing will occur.

*Consider the three-leg route identified by the name **eastbound_north_milk_run**. This is fictional example of a route that might be flown by **United Flight 1899*** and *supported by the Boeing **airplane United N517LY** (tail number)*. This route has three legs and originates in Seattle, Washington. The first leg takes you to Chicago, Illinois; the second leg takes you to Washington, D.C.; and the third leg takes you to New York, New York. The distances for the legs are 1800 miles, 600 miles, and 200 miles, respectively. The legs are also identified using numbered labels for easy reference and reuse – legs #24, #20 and #8, respectively. When keeping track of flight status, the flight always starts in progress state 0. In this situation, the airplane supporting this flight is "on_ground" at the departing airport (**SEA**) for the first leg. Once the airplane takes off, then it is "in_flight" towards the first leg destination airport (**ORD**), and the progress state advances to 1. When that flight lands, the progress state remains at 1, but the airplane's status reverts to the "on_ground" value. This pattern of landing and takeoff transitions continues until the flight lands at the airport at the end of the route – in this case, the John F. Kennedy Airport (**JFK**) in New York.



Some people are pilots. The pilots in our system are qualified to command a flight. Flights that are supported by Airbus airplanes must have at least one assigned pilot before taking flight, while Boeing airplanes must have a minimum of two pilots. A flight does not require pilots to be assigned if it's not in the air. Your system must track that the correct number of pilots are assigned to command a flight before it is allowed to takeoff. Your system does not need to store or otherwise reference the pilot's specific roles such as Captain, First Officer, Flight Engineer, etc. Each pilot can have one or more licenses that confirm their flight skills, and gains experience that must be tracked for each leg of a flight when they are assigned as a member of the flight team. Your system must also keep track of each pilot's tax identifier to ensure that they are paid appropriately for their services. The tax-identifier will be stored using a "*xxx-xx-xxxx*" format.

The people who are not pilots will be passengers. Passengers are normally waiting at airports with "vacation intentions": namely, to travel to one or more other airports/destinations in a desired order/sequence. If a passenger waiting at an airport sees that an airplane is departing, and that airplane will fly to their next desired destination, then the passenger will attempt to pay the fare for that flight and board. If the passenger has the funds to cover the price of flight, and the airplane supporting that flight has the seating capacity, then your system must track the passenger boarding the flight and later getting off the flight at their desired destination. Your system must also track the number of miles earned by each passenger so that frequent flyer miles can be accounted for correctly. Once a passenger boards, the revenue of the airline managing the flight will also increase by the flight's fare.

Whether a person is waiting in an airport for their next flight or seated on an airplane waiting for it to reach its destination, that person must always be associated with one of those valid types of locations. Your system must keep track of where all persons – passengers and pilots – are located at all times. Your system does not require you to track people at other locations, such as traveling to/from the airport from home, staying in local hotels, etc.

Your system must also be able to display information that will help us keep track of the flow of airplanes and passengers throughout the system. Each airplane has a maximum capacity of the number of people that be carried at any one time, so your system must be able to determine how many people are on each airplane based on the number of people whose location is on that airplane. Similarly, your system must be able to determine how many people are at each airport. The main process of this "flight simulation" for our system is that airplanes will depart from airports and then land at the destination airports in time-sequential order. The "state of the simulation" will be configured initially to represent a 24-hour day of travel. As each airplane arrives at the next airport along its flight route, passengers who are due to depart the airplane at that stop will deplane the craft. As each airplane prepares to takeoff and depart towards the next airport along its route, passengers at that airport who need to board, will board the aircraft, and then the airplane will take off towards the next airport along its flight route until it reaches its (final) destination airport.

As an example of the simulation, consider the normal progression of flights as discussed above. ***For each flight, there is a next_time value that indicates when that flight will "logically change state"*** for our simulation purposes. If an airplane is in flight, then the next_time value indicates when the flight will land at the destination airport of the current leg. If an airplane is on the ground, then the next_time value indicates when the airplane will take off towards the arrival airport of the following leg. So, *the simulation cycle can be executed by **finding the "next flight" that will change its state chronologically*** – more specifically, that will land or take off – before the other flights based on having the smallest next_time value. *If the next flight is **landing***, then we must allow the flight to land, and allow passengers to disembark if their ticket destination matches the current airport. *If the next flight is **taking off***, then we must allow passengers to board the flight, and then allow for the flight to take off (if permitted). In the case where *the airplane is already **at the final airport** in the route*, we must recycle the crew so that they can leave the plane, get some rest, and prepare for their next flight. We must also remove the flight once it has reached the end of its route.

[1] **flight_landing()** & [2] **passengers_disembark()** [5] **passengers_board()** & [6] **flight_takeoff()**

[3] **recycle_crew()** & [4] **retire_flight()** [***if – and only if – this is the final airport on the route***]

Simulation Cycle Steps

Sample Data Elements

The following data is provided to assist you in visualizing and/or validating the system design you are tasked with developing. You are not required to submit this data. The intent is that you can use the data to check if your EERD can store the data values, relationships, etc. that we've provided in a reasonable manner. If there are elements of the data that can't be represented in an appropriate attribute, entity, or relationship, then perhaps you need to revise your design. Similarly, if there are attributes, entities, relationships, etc. that haven't been used after you've stored all the data, then perhaps your design has unnecessary elements. This exercise doesn't guarantee that your EERD is fully correct, but it does offer some validation that you are on the correct track.

Celia Cemre and Jason Tane are passengers currently located at the Fantasyland Airport (airport ID: FSL) in San Francisco, California, USA. Celia and Jason have 300 miles and 600 miles in frequent flyer miles, respectively. They will be riding on AirAces Flight 2340, which is supported by the AirAces airplane with tail number SB10. That airplane is a Boeing 777 which was serviced within the last month. That flight is being commanded by pilots Bo Feidhlim and Nadya Manjula. Bo and Nadya are also located at the FSL Airport. Bo’s tax ID is 235-71-1131, he has 10 legs of experience, and he has licenses for A380 and B777 aircraft. Nadya’s tax ID is 357-11-1317, she has 37 legs of experience, and she has licenses for B777, B787, and B737 aircraft. Flight 1761 is scheduled to depart at 5:30am.

Celia would like to go to the OmniTech Airport (airport ID: OTC) in Atlanta, Georgia, USA while Jason would like to go to the Bubba Gump Airport (airport ID: BGP) in Baton Rouge, Louisiana, USA. Both Celia and Jason notice that the AirAces Flight 2340 is about to depart and is heading to their desired destinations, so they purchase tickets for the flight. We must ensure that there’s enough space for both passengers and that both passengers also each have enough funds to afford the $300 cost for the flight. The funds collected for their tickets will be added into the revenue for AirAces, which is an airline with a revenue of $106B (i.e., Billion) dollars. They are a larger airline as compared to FlyFastFurious, which is a relatively smaller airline with a revenue of $23B dollars.

Meanwhile, AirAces Flight 2110 is being supported by an Airbus airplane with a neo engine (AirAces tail number SB16) and starting at the Bubba Gump Airport (airport ID: BGP) with a departure time of 6:30am. The flight will be commanded by Laverna Nare with a tax ID of 571-11-3171, 56 legs of experience, and licenses for A320 aircraft. Two other flights are the FlyFastFurious Flight 2200 starting at the Pommes Frites Airport (airport ID: PFS) in Colorado Springs, Colorado, USA and departing initially at 11:00am; and the AirAces Flight 2050, also starting at the FSL Airport and departing at 1:30pm.

The first activity occurs when AirAces Flight 2340 departs at 5:30am. The flight is following the three-leg route, *southern_states_hop*, from the FSL Airport to the Sandcastle Airport (airport ID: SCA) in Santa Fe, New Mexico, USA, then to the Bubba Gump Airport, and finally to the OmniTech Airport. The current status of AirAces Flight 2340 Flight is that it is on the ground at FSL, about to begin with leg #1 of its route, with the next activity/status change time as 5:30am. Any current passengers should stay on the plane, and new passengers – namely, Celia and Jason – must board the plane before it takes off towards the SCA Airport. Also, we must ensure that Bo & Nadya are assigned as part of the flight crew before the aircraft can be allowed to depart.

Suppose the distance between the FSL and SCA Airports is 1200 miles, and the speed of the AirAces SB10 airplane is 400 miles per hour. Then the airplane would reach the SCA Airport at 5:30am + (1200 / 400) hours = 8:30am. The status of AirAces Flight 2340 will then be changed to “in flight”, still on the first leg of the route, and with an updated activity/status change time of 8:30am.

Now, the earliest activity/status change time of the flights is the AirAces Flight 2110 at 6:30am. The AirAces Flight 2110 follows a two-leg route from the BGP Airport to the OTC Airport, and then back to the BGP Airport. If there are 500 miles between the BGP and OTC Airports, and if the AirAces SB16 airplane has a speed of 200 miles per hour, then the flight duration will be 2 hours and 30 minutes. There is a passenger at the BGP Airport named Mira – a high-profile actress – who wants to go to the OTC Airport. The current status of AirAces Flight 2110 is on the ground at BGP, about to begin the first leg of the route, with an activity/status change time of 6:30am. When this flight is processed, Laverna and Mira will board the airplane, and the flight status will be updated to “in flight”, still on the first leg of the route with an updated activity/status change time of 9:00am.

Let’s process one more event – namely, AirAces Flight 2340, which is currently “in flight” on the first leg of the route with an impending arrival time of 8:30am. As the flight lands, the system must check the tickets of people on the flight to ensure that all the passengers on board who need to deplane change their location from the airplane to the airport. There will be a time delay of 30 minutes while the airplane lands, refuels and prepares to take off again. So, the updated status of the AirAces Flight 2340 will be on the ground, on the second leg of the route with an impending takeoff time of 9:00am. Since this leg of the flight has concluded, the flight experience levels for Bo and Nadya should be updated to 11 and 38 legs, respectively, and the frequent flyer miles for Celia and Jason should be updated to 1500 and 1800 miles, respectively.

Note that the flight routes can have overlapping legs. For example, the FlyFastFurious Flight 2200 has a three-leg route starting from the PFS Airport to the SCA Airport, then on to the BGP Airport, and finally ending at the BigApple Airport (airport ID: BAP) in Islip, New York, USA. The AirAces Flight 2050 which begins at the FSL Airport, is a one-leg (i.e., nonstop) flight to the OTC Airport.