Relational Schema

airline(airlineID, revenue)

location(locID)

route(routeID)

airport(airportID, port_name, city, state, country, location fk[1])
Fk1: location -> location.locID

airplane(airlineID fk[2], tail_num, location fk[3], plane_type, props, skids, jets, seat_cap, speed)
Fk2: airlineID -> airline.airlineID
Fk3: location -> location.locID

flight(flightID, route_follow fk[4], cost, supportedID, supported_num fk[5], progress,
plane_status, next_time)
Fk4: route_follow -> route.routeID
Fk5: (supported_ID, supported_num) -> (airplane.airlineID, airplane.tail_num)

leg(legID, distance, departs fk[6], arrives fk[7])
Fk6: departs -> airport.airportID
Fk7: arrives -> airport.airportID

route_contains(routeID fk[8], legID fk[9], sequence)
Fk8: routeID -> route.routeID
Fk9: legID -> leg.legID

passenger(personID, fname, lname, funds, miles occupies fk[10])
Fk10: occupies -> location.locID

vacation(personID fk[11], destination, sequence)
Fk11: personID ->passenger.personID

pilot(pilotID, taxID, fname, lname, experience, commands fk[12], occupies fk[13]])
Fk12: commands -> flight.flightID
Fk13: occupies -> location.locID

license(pID fk[14], license_type)
Fk14: pID -> pilot.pilotID

Uncontained Constraints

- Ensure each pilot has at least one license to fly
- Ensure the sequence of legs is contiguous (i.e. if is supposed to go (Lx,1)->(Ly,2)->(Lz,3) using (name, sequence), there is (Lm, 4) before there is (Ln, 5))
- Similarly, a vacation sequence for a passenger should have the same property
- Each sequence must start at 1 and only increase; ensure there are no number < 1
- Ensure each leg distance is in miles
- Ensure each route is made up of legs (i.e. each route corresponds to at least one row in the route_contains table)

Side Nots and Logical Remarks about Uncontained Constraints

For any sequence, we will have it start at 1. This gives the interpretation that progress = 0 means that a flight has not yet started its sequences of legs.

Distance of the legs is assumed to be only in miles. We will make this assumption, so we can store the values as a number. This assumption could be overcome by adding another column called units that gives the units of the distance, but, for the sake of the project, this will not be implemented (though this would probably be more ideal in the real world).

Pilots have the option to have last name to default null since we were told that we are only required to store first names. This would probably not be the case in the real world; if you are flying a plane, we better know every bit of information about you including the last name.

In the person spreadsheet, we were given (flying_airline,flying_tail). This could be used in the pilot table, but we opted to use only (associated_flight) since this contains the information about (flying_airline,flying_tail) and would be redundant. Also, in the given ERD, pilots command flights not airplanes, so we should not expect the airplane identifier to pop up in the pilot table.