James Bloor

CS669 – Lab 1

1/25/2021

Project Direction Overview

I would like to develop an application for airlines can use to track where the aircrafts are to book flights, see how many passengers can fit on a plane and even watch where aircrafts have been to. The user(s) for this app can load their planes onto the app, giving the location, type of plane and # of passengers that are able to fly on this plane. As people request flights a user can log onto the application and see if they have a flight available to be booked across the nation. This application can help a small airline company track where their planes are and make it very convenient to book flights as well as add planes as wanted.

An example of someone using this application could be to see if a plane is available to be book out of Atlanta. An authorized user could then log onto their application and see what planes are in or near Atlanta. From there they would be able to see how many passengers could fly on the plane and then book the flight. Then the necessary staff would be notifying to be ready for a flight at a certain time, a log would be updated that their #5 plane is moving from Atlanta to Boston on the 25th of January and all the necessary changes would be made on the database side. AS airlines add planes to their fleet or retire planes, these changes can be made in the application to make sure that all needed updates are easy and accessible.

The project I will work on for the coming weeks will focus on the database side of this creation. I will be able to track aircrafts for multiple companies that request the services of this application. I will record each flight that takes place, keep track of what aircrafts are in use at any given time, and have the ability to retire planes and add them as companies find needed.

Use Cases and Fields

*Add/delete planes as necessary*

1. Determine what airline we are making changes to
2. ask for all the needed information to ensure that when a plane is in our database.
3. Have all the information that might be wanted by a user (model, year, available passengers) or a possible booker implemented into the correct table.

|  |  |  |
| --- | --- | --- |
| **Field** | **What it Stores** | **Why it is Needed** |
| AirlineName | This field will tell us who the owner of the plane is. | This is necessary so when a specific user for an airline is only allowed to see their flight data/where their planes reside. |
| PlaneModel | The model of the plane and name in which the owners gave to it. | Good to have so that a plane can be tracked overtime and through multiple trips. |
| PassangerSize | The number of passages that can fit on a desired plane. | This information is useful when someone wants to book a flight and they can tell if a certain plane fits their requirements. |
| PlaneYear | See when the plane was manufactured | Good to know when a plane was made so we can see when parts/pieces might be outdated. |
| PlaneNumber | A unique identifier for each plane | This differentiates each aircraft from one another. |

*Search for a flight*

1. A user enters our application and chooses the location in which they desire to fly out of.
2. I use the desired information to look at my table and see what flight are available to fly out of a certain location.
3. The database will also make sure that the plane will be there on the requested date of a certain possible flight

|  |  |  |
| --- | --- | --- |
| **Field** | **What it Stores** | **Why it is Needed** |
| PlaneLocation | The location for a certain plane | This will be used to determine where a certain plane is and if it would be useful in our search. |
| DateOfArrival | The date in which our plane arrived at the location above. | This can make sure that the plane will be at the above location at the certain time in which the booker want to book the flight for |
| PlaneNumber | A unique identifier for each plane | This differentiates each aircraft from one another. |

3. *Book a flight*

1) After a user has found a plane, he/she would like to book, they may request the plane to be booked.

2) The database will then mark this plane as booked for a certain day

3)The application will also request the user to input the location in which they want to fly to; thus placing this information in the database.

|  |  |  |
| --- | --- | --- |
| **Field** | **What it Stores** | **Why it is Needed** |
| BookDate | The date in which a specific airplane is booked for. | This will make sure that no plane can be book on the same day to multiple people or to be sent to different locations. |
| Destination | The location in which our plane will fly to. | This info will be used to determine when a plane will be in the future and allow other users to book the plane in the future from this location. |
| FlightNumber | The unique transactions of this flight. | This ensures that each flight has its own unique identifier. |

*Look up past flights that have occurred*

1. A user enters the airline they are associated with.
2. The user can enter if they would like to see the data from a certain flight number of aircraft/to a location/from a location.

|  |  |  |
| --- | --- | --- |
| **Field** | **What it Stores** | **Why it is Needed** |
| FlightNumber | The unique transactions of this flight. | This ensures that each flight has its own unique identifier. |
| FlightDeparture | Location in which this flight left. | Know the leaving location for a flight. |
| FlightArivial | Location in which this flight will land. | Knowing the arriving location for a flight. |

*Adding an airline into our database*

1. A new user will add their airline info in our app.
2. The user will need to add their name.
3. This will make their airline/company a reference in our database now.

|  |  |  |
| --- | --- | --- |
| **Field** | **What it Stores** | **Why it is Needed** |
| AirlineName | The airline name in which we are holding their plane data. | This data is needed so we can keep track whose data we are holding and for attaching plane number to. |
| DateAirlineAdded | The date in which the airline joined our database. | This is good information to hold to know when people have joined our database and application. |

Summary and Reflection

My database will have the ability to hold multiple airlines flight data as well as keep track of aircrafts across the country. A smaller airline might find it cumbersome and stressful to keep all this information in an easily accessible location so I want to make a database in which they can use and track all necessary information. This database will have the ability to be searched through, book a plane, and see wat works best for the requesters of a plane in certain locations.

A worry that I have now is who I allow to access certain information and who is allowed to add/delete planes from a certain airline. Also, flights get delayed/canceled/moved while in the air at certain times. I want to make sure I have something in the future to address these situations, so my database is always accurate.

Coming from a city, Atlanta, where we have flights that stretch around the world, I look forward to thinking through problems that I will encounter in the coming weeks. Becoming familiar with how a database is implemented into business is a necessary tool in when I want to create my path and I hope that by creating this database I can build upon my knowledge and ready myself for SQL and the road ahead.

Thank you in advance for reviewing this document and for your feedback.

James Bloor

CS669 – Iteration 2

2/1/2021

Aircraft ERD Model

First, I will start by defining some structural database rules that my database must follow. These will help me create my initial ERD model for this project.

Each flight number lands at an arrival location, each arrival location can have many planes land there.

Each flight number takes off at a departure location, each departure location can have many planes take off there.

Each flight number gains a flight distance, each distance can have many flights.

A flight number leaves on a certain date, each date can have many planes leaving.

A flight data can change many times, a unique change to a flight can only be done once.

Each plane is located at a location, each location can have many planes.

A plane location can change many times, a unique location change can only happen once.

Each airline has airline info, info must be attached to a unique airline.

Each plane has airline info, Airline info can be attributed to many planes.

A Plane can only be one of the following, A very light jet, a light jet, a mid-sized jet, a super mid-sized jet, a large jet, a long-range jet, or a VIP airliner.

A very light jet is created by a very light jet model, a very light jet model crates a very light jet.

A light jet is created by a light jet model, a light jet model crates a light jet.

A mid-size jet is created by a mid-size jet model, a mid-size jet model crates a mid-size jet.

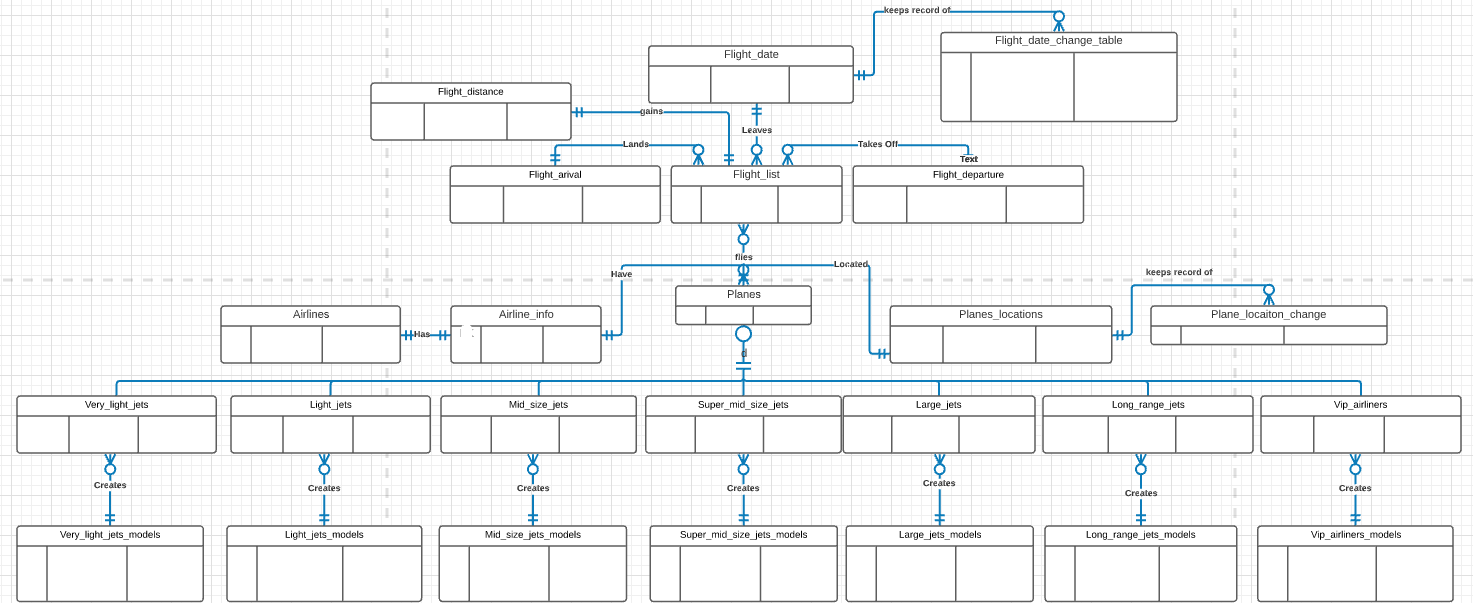
A super mid-size jet is created by a super mid-size jet model, a super mid-size jet model crates a super mid-size jet.

A large jet is created by a large jet model, a large jet model crates a large jet.

A long-range jet is created by a long-range jet model, a long-range jet model crates a long-range jet.

A VIP airliner is created by a VIP airliners model, a VIP airliners model crates a VIP airliners.

I feel comfortable in these relationships and rules that I have above. I will now move to Luicid and create a ERD and copy it too here below.



As a note, I am having trouble what is mandatory and not. For example, a DateAirlineAdded notes the year an airline enters my database. I believe that its entity relationship would be each DateAirlineAdded can have many AirlineName. This is because I can have the year 2014 but no airline joined my database at that time meaning that 2014 would not appear in this table. So does this make it optional because a year does not have to be in my database or is it mandatory because every year that will be in my table has at least one airline that is aligned with it?

I have also implemented the recommended additions that you had noted in my previous iteration of this project.

Thank you again for your feedback.

-James

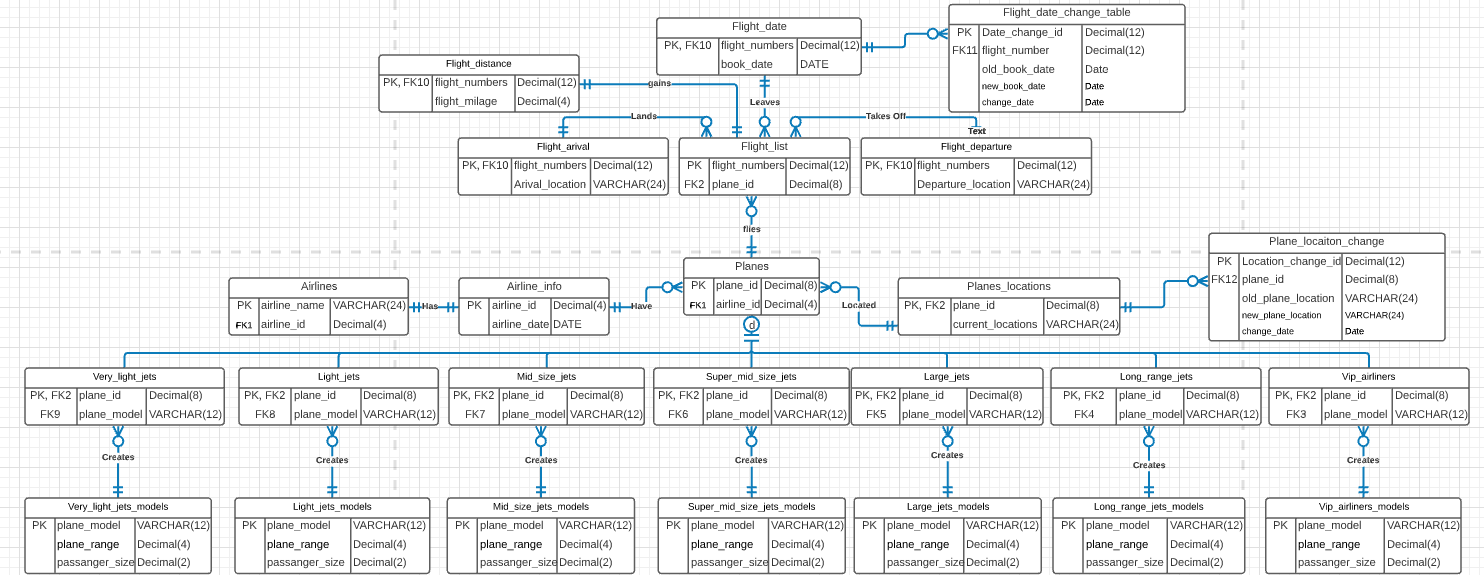
James Bloor

CS669 – Iteration 3

2/8/2021

In the area above I have taken the feedback that I received on my previous version of this project and updated my ERD. The major changes were taking attributes and moving them away from my ERD for now. While these attributes will become useful in iteration 3 of project. After doing some research I have found a specialization to place in my ERD. This specialization will be for my plane models, private jets fall in seven different categories so, now you can see above that I have list these in my ERD above. These specializations will help with finding the correct plane for a specific passenger size as well as distance a plane can fly.

On the next pace I have placed my DBMS physical ERD for my aircraft tracking application.



Updated Project Summary for Iteration 3:

My database is an aircraft tracker for private airline business. My database will work with an application so airline companies can input their aircraft information and then use my app to track their planes, have clients book flights, and update any info as needed. I will be tracking every flight made by any aircraft I have in my system, total milage by a plane, and even have filters that allow a user to input a possible location/passenger size/location to then book a suitable flight.

I have recently updated my iteration 2 ERD. At first, I was bringing into many attributes and classifying them as entities. Now that I have updated my ERD, it has allowed me to create a DBMS physical ERD. This design maps out how my database tables with work with one another to make my application possible. BY having specific plane model data in my database, an airline will only need to ass in what model a plane is, and I will already have all the standard info ready for any specific jet. The Planes table will work as my central location to pulling needed information, whether that be where the plane is currently traveling or if the plane is reaching its end of life.

With my DBMS physical ERD created, I now have a visual idea of how all my tables will work with one another and communicate when needed. I also notice that it will take time to place in the correct model information for each jet specification I will have; however, I believe I have found a great list that will help me with all the information I will need for these steps.

Thank you in advance for your feedback.

-James Bloor

James Bloor

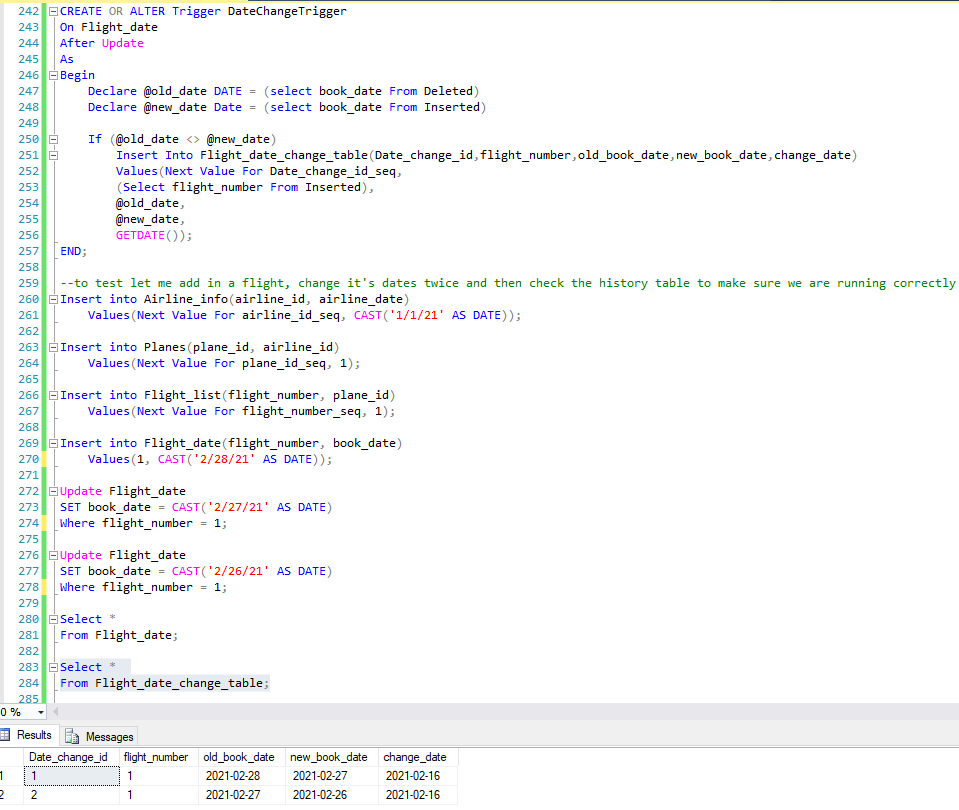
CS669 – Iteration 4

2/15/2021

Below is a screen shot of my initial table creation and sequences running correctly.

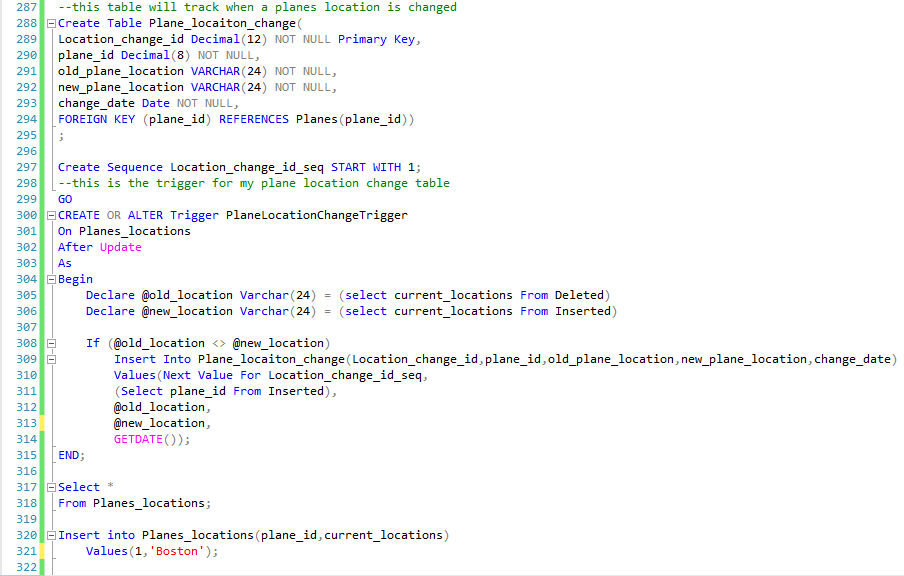


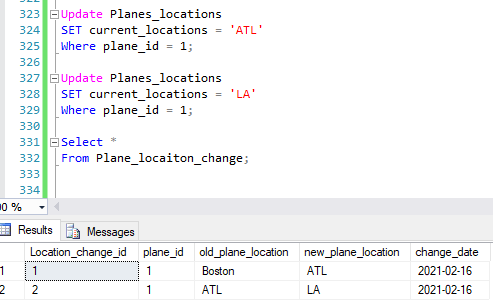
Below is my first trigger that will track the date change for flights:



The above history will be useful to see when I flight was originally booked and if the data has changed for an initial booking of a flight. With flights being decayed or canceled become of weather or other extenuating circumstances, this table will prove to be useful.

My next trigger will track when a plane changes locations:

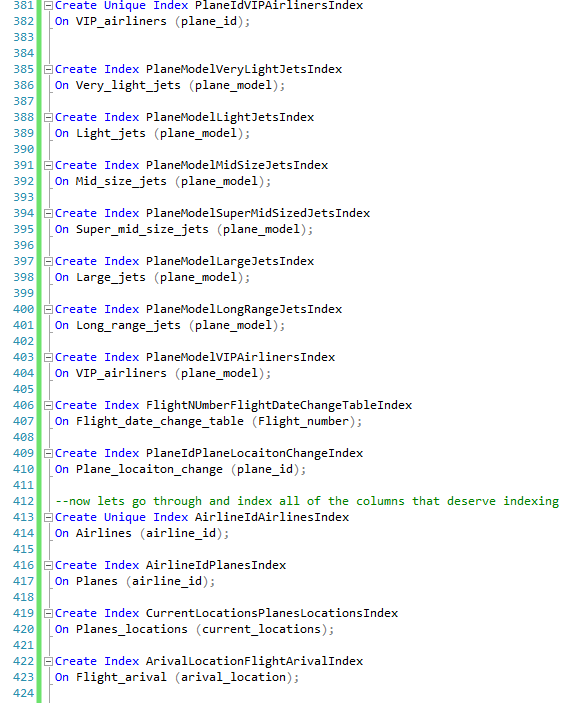


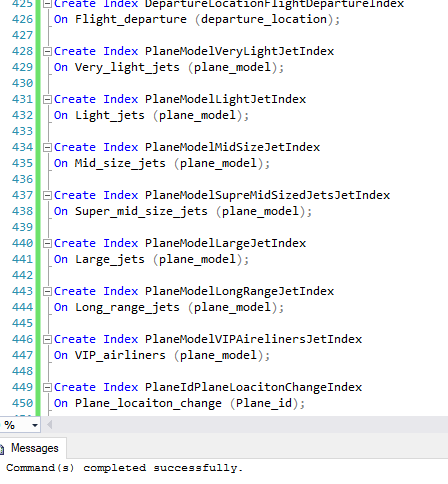


The trigger above will track when a plane moves locations. This is useful so users or airline workers will be able to see when a plane has traveled to in its entire history of being in my database. This can also be useful to see when a plane has gone when it is not on an official flight. The flight data will not be having it move from Boston to Atlanta for example, however, this flight history will be able to be shown in this history tables.

Below are the screen shots for my indexes, I start with the Foreign Keys in my database and then move to the columns in my tables that I believe deserve indexing for future joins and where statements.







In this iteration I ma happy to see my work starting to come to life> After these past few weeks I am beginning to fell more confidant that my tables will be able to interact with one another in the way that I envisioned them. A part of this week that slowed me down were the triggers, some of their format did place me in a bit of a rut but I am thrilled to finally have them working and see that my history tables will be able to run with ease.

I have updated my previous iterations to reflect my current work and stay up to date with what my final product will looks like. I am have done research this past week to know what types of aircrafts belong in which subsection of private planes to get me ready to place real data in these tables this coming week.

I would again like to thank you for your guidance during these past few weeks and constructive feedback.

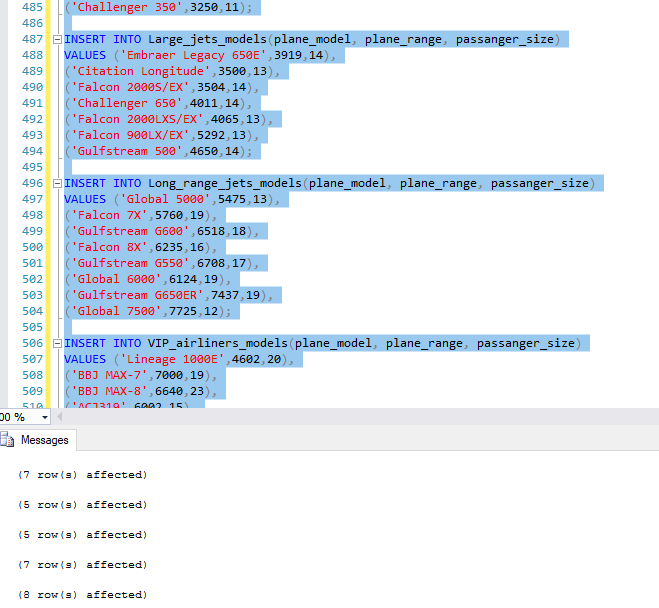
James Bloor

James Bloor

CS669 – Iteration 5

2/25/2021

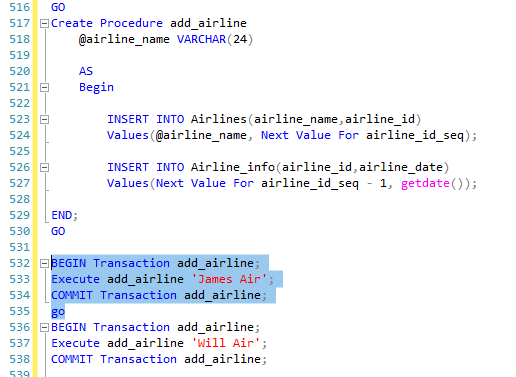
Before I start with my procedures, let me add in all the private jet models that exist. Some of these tables do not have 10 rows because there are not 10 possible models that exist in each model type.



First, I will add a Procedure that will allow us to add Airlines into our database.

*Adding an airline into our database*

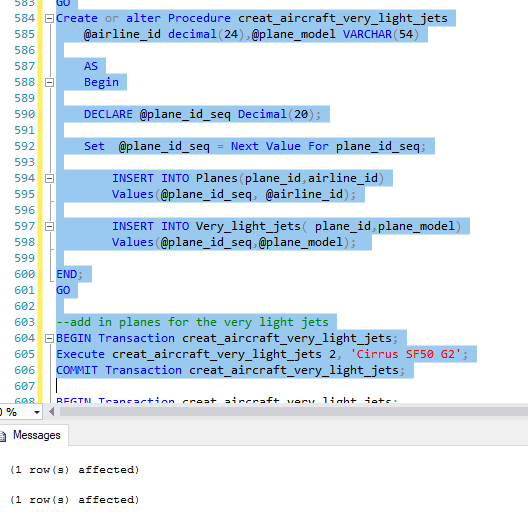
1. A new user will add their airline info in our app.
2. The user will need to add their name.
3. This will make their airline/company a reference in our database now.



Now I will add in planes to be used for flights:

*Add/delete planes as necessary.*

1. Determine what airline we are making changes to
2. ask for all the needed information to ensure that when a plane is in our database.



And my last procedure will add flights to my database

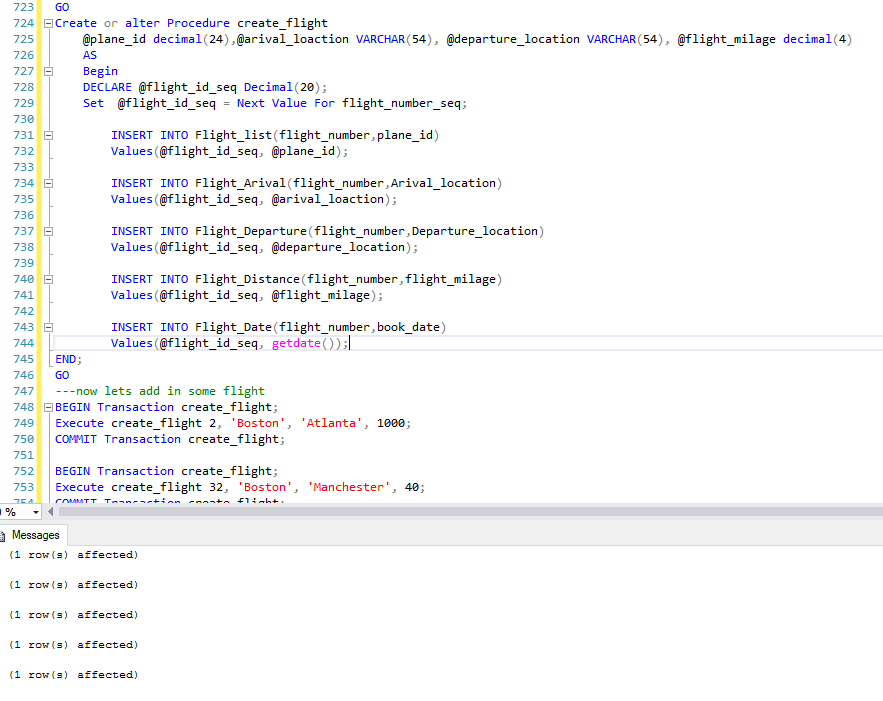
*Book a flight*

1) After a user has found a plane, he/she would like to book, they may request the plane to be booked.

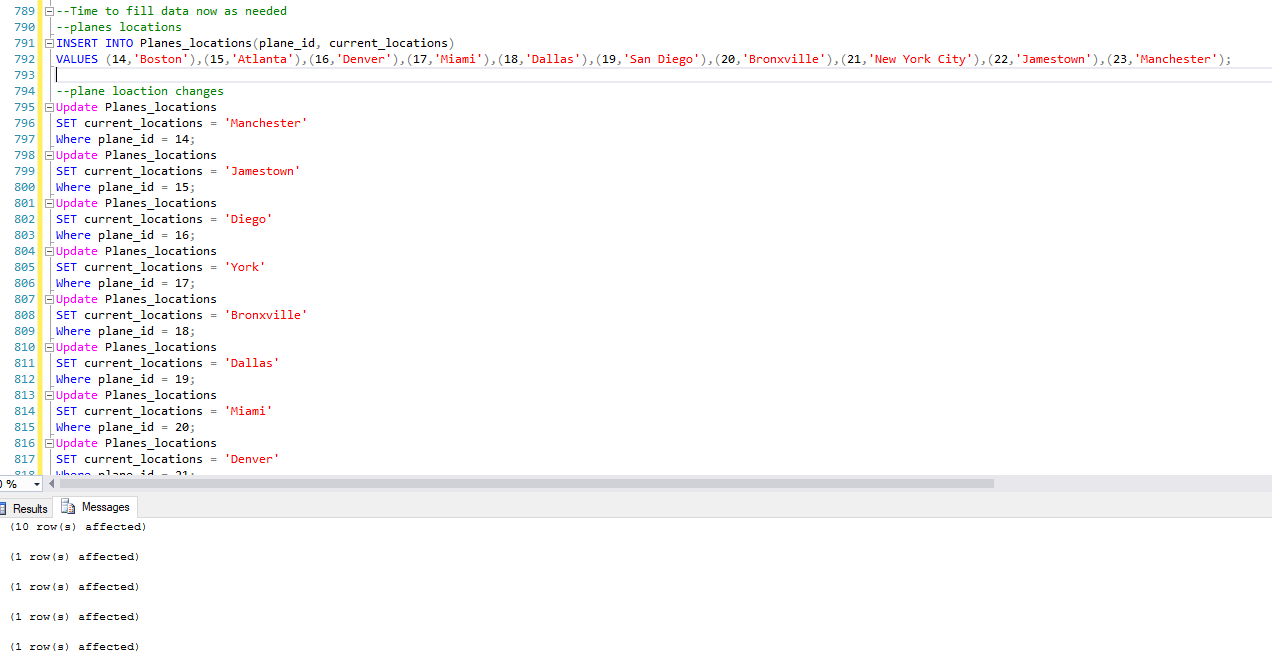
2) The database will then mark this plane as booked for a certain day

3)The application will also request the user to input the location in which they want to fly to; thus placing this information in the database.

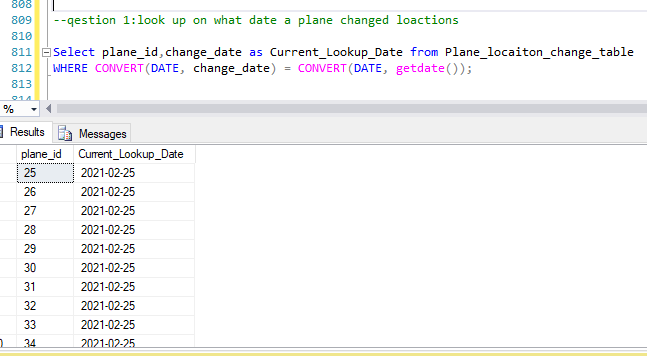
|  |  |  |
| --- | --- | --- |
| **Field** | **What it Stores** | **Why it is Needed** |
| BookDate | The date in which a specific airplane is booked for. | This will make sure that no plane can be book on the same day to multiple people or to be sent to different locations. |
| Destination | The location in which our plane will fly to. | This info will be used to determine when a plane will be in the future and allow other users to book the plane in the future from this location. |
| FlightNumber | The unique transactions of this flight. | This ensures that each flight has its own unique identifier. |
|  |  |  |



Now I will move to my 5 questions to address with Queries, with all my tables being filled with at least 10 rows of data.

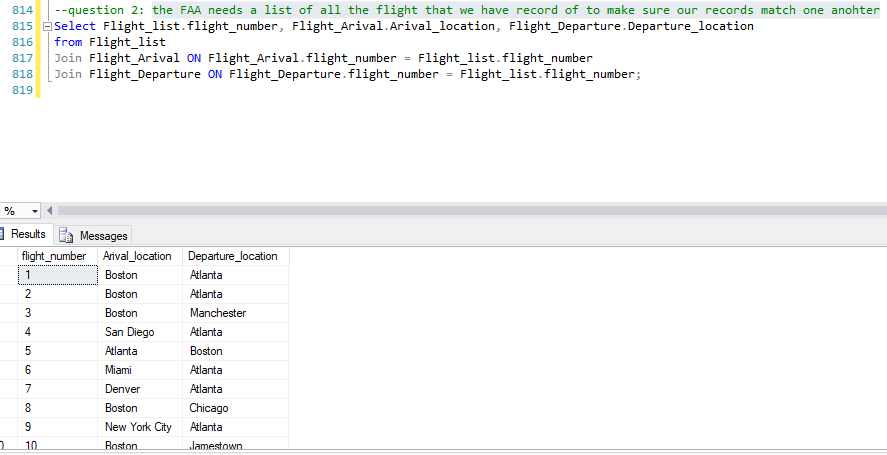


Question 1: How many Planes changed location on a certain day? We can use the location change table to see this information.



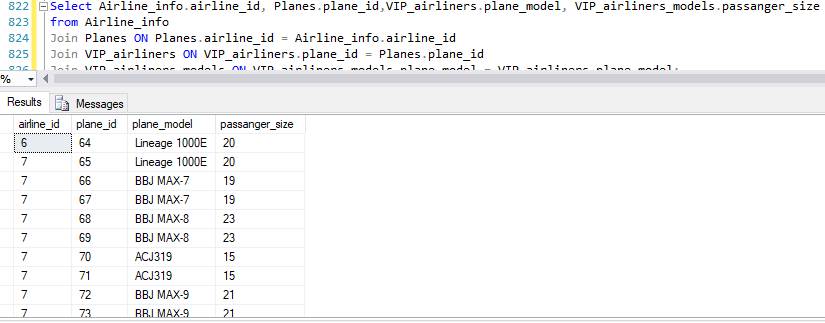
In the photo above I have created a look to see what planes have changed locations today. This data can be valuable to make sure that all planes make it to their desired locations and to make sure that database is doing its job when planes take off and move locations.

Question 2: The FAA needs a list of all the flight that we have record of to make sure our records match one another. The query below will display all the data needed.



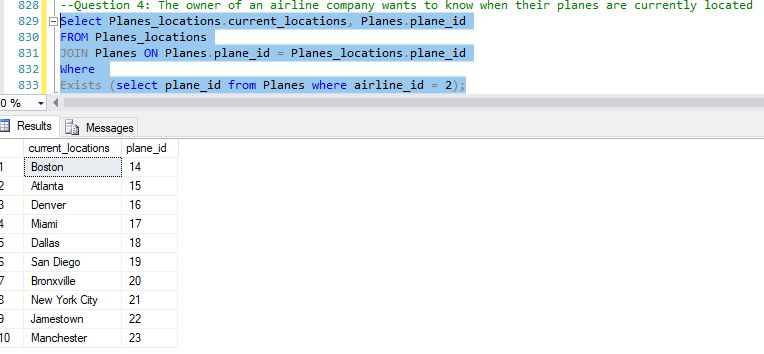
In the photo above we can see that I have pulled flight numbers and arrival and departure location data to give to the FAA. This is a good list to make sure they have the data that I am tracking as well in case anything goes wrong or maybe any funny business is going on behind the sense.

Question 3: A customer is looking for all the VIP airliners that we have in our system and wants to know how many people each plane it can seat. After receiving this information, they say they can decide on which plane would be best for them.



And now we have the perfect set of data to send along to our interested customer. They can pick a plane in which they would like to fly on and we can see what we can work out from where he/she would like to take off from.

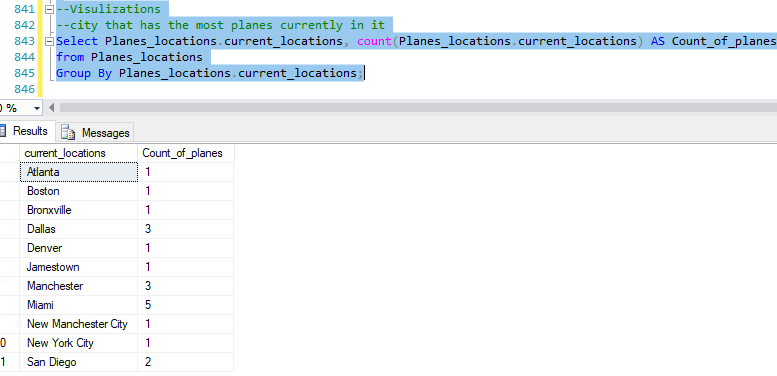
Question 4: The owner of an airline company, id = 2, wants to know where their planes are currently location, I have created the query below to help us create a data set to send to the owner.



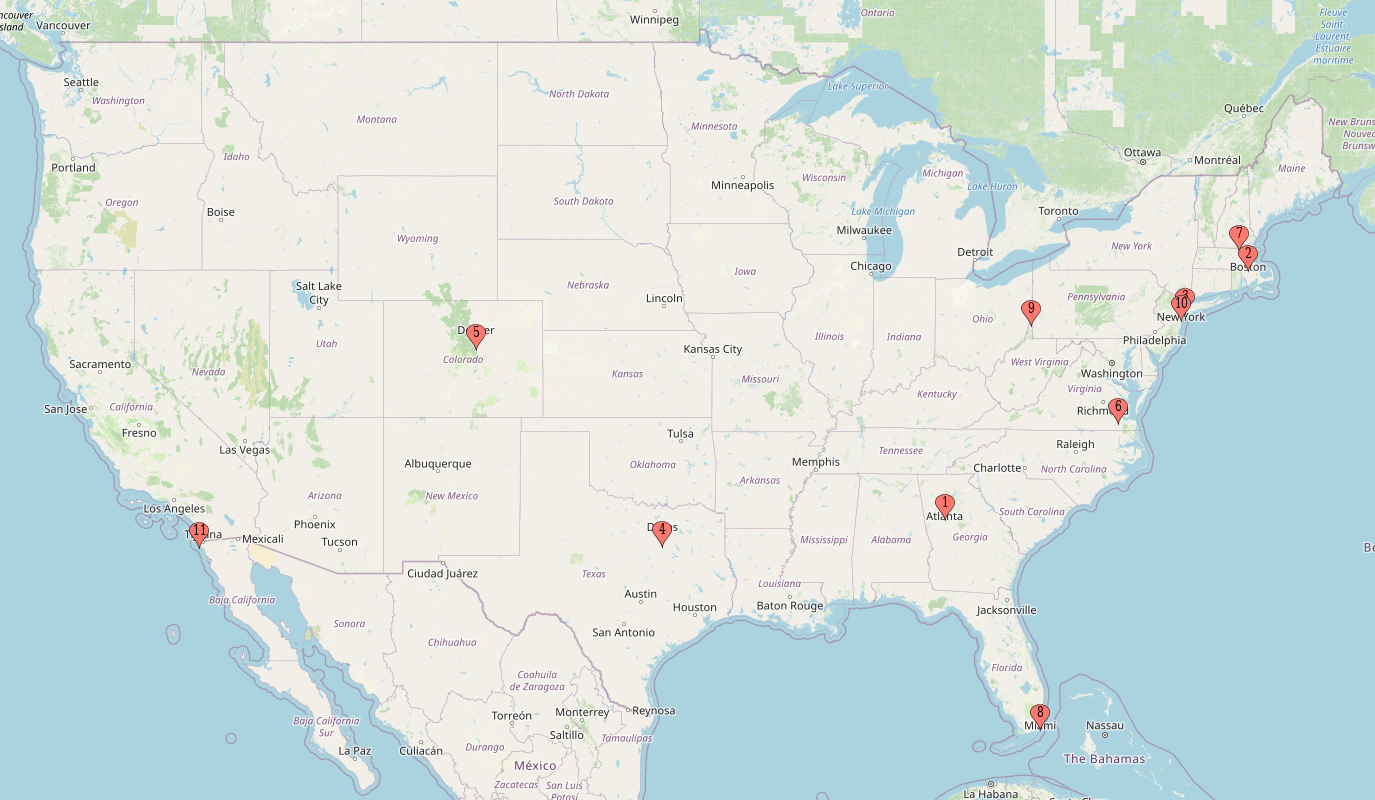
Now for this last question, which we will use for one of our data visualizations, we get a clean list of the planes owned by airline 2. This list gives us the plane’s ID as well as the location. This is a good way to make sure all of our planes aren’t at one single location and not working efficiently as well.

Now let us move on to visualize some data.

First, we will look at how many planes are in each city that has a plane in. This can help decide on if planes need to move to a different city to make sure flights can depart from all location.



From this photo above we can see that Dallas, Miami and Manchester have more planes than a lot of other cities. Luckily, these cities are all located in different parts of the country so there is not a load up in one certain are of the country. However, with these being the only planes in our system I would recommend we spread some out to ensure people can book trips and have a plane to fly on as they please.

Now I want to use the data we had in my question 4. This is where we were asked to find where the owner’s planes were across the nation. Instead of giving a list I would like to represent the data in a different way. 

Getting a list is nice to see where items, in this instance’s aircrafts, are but with a map this data can be visualized in a nice format.

Overall, I am please with what I have created. This past iteration what difficult for me to complete but I am happy that data is filled in my tables. My interest in crating this database started as a minor thought but by taking the steps that we have in this project it has been interesting to thing though the problems and realize how much data can be held for a small group of aircrafts.

I have added instances where someone can reach out to my team and ask to add their aircrafts to my database to be tracked. From there we can add their airlines in, aircrafts in and start booking flights immediately.

My ERD represents what I have created in my SQL and can act as a great representation of how my database and tables act with one another in SQL. I have created valuable tools in my code to make sure that all primary and foreign keys are in alignment with one another. It has taken time but I very happy to call this project my own.

Moving on from here, I look forward to enhancing this database and continuing to add entities in my free time. I am now aware that my past work needs to be continuously updated with my database so everything ins covered from start to finish. While I may never truly use this database for what it is built for, I do use a pharmaceutical database in my work environment, and I look forward to utilizing the skills I have learned through this project to have that database and put it to use.

James Bloor