Phuc Nguyen
INFO 330 BB
Completed Project (2-4)
Note - I am doing this part of the project **myself**, due to the group's unresponsiveness, so do not include me in the group grade.

See the Logical Diagram Attached

Project Phase 1 (Contains ER diagram) -
https://docs.google.com/document/d/156wqq49IKq0Rsa4dRHLwClbuYIB5r27wG_FI6gJV1W8/edit?usp=sharing

Keep in mind that data is generated, and some situations are unlikely to occur in the real world.

Relational Schema (For Logical Model)

Passenger (**Passenger_ID** INT, First_Name VARCHAR, Last_Name VARCHAR, Phone VARCHAR, Email VARCHAR, Date_Of_Birth DATE)
Books (**Passenger_ID** INT, **Booking_ID** INT)
Booking (**Booking_ID** INT, Price DECIMAL(10, 2), Booking_Date DATE)
MultiFlightBooking (**Booking_ID** INT, **Flight_ID** INT, Flight_Sequence TINYINT)
Flight (**Flight_ID** INT, Departure_Date_Time DATETIME, Arrival_Date_Time DATETIME, Departure_Airport VARCHAR, Arrival_Airport VARCHAR)
FlightAssignment (**Flight_ID** INT, **Airline_ID** VARCHAR, **Aircraft_ID** VARCHAR)
Aircraft (**Aircraft_ID** VARCHAR, **Airline_ID** VARCHAR, Model VARCHAR, Capacity INT)
Airline (**Airline_ID** VARCHAR, Airline_Name VARCHAR)

Size specification in Dump Script

Business Rules (Refined)

1. A passenger can have many bookings and bookings can have multiple passengers, therefore "Books" is defined as allowing multiple passengers to have multiple bookings.
2. A booking can be associated with many flights and a flight can have multiple bookings from different passengers. MultiFlightBooking is used to correlate sequence if the passenger is going on connecting flights.
3. Flight uses FlightAssignment to assign to an airline and aircraft, which can be assigned to different/multiple airlines and aircraft.
4. Aircraft can only have one airline, while an airline can have many aircraft
5. Each entry in the Books table associates a passenger with a booking.
6. Each entry in the FlightAssignment table associates a flight with an airline and aircraft.
7. Passenger has to be at least 18 years of age and less than 100 years of age from 12/07/2023
8. FlightAssignment connects flight ID with airline and aircraft. Flight, Airline, and Aircraft can have multiple flight assignments but every flight assignment must only contain one of each.

Associative Entities/Problem Solving

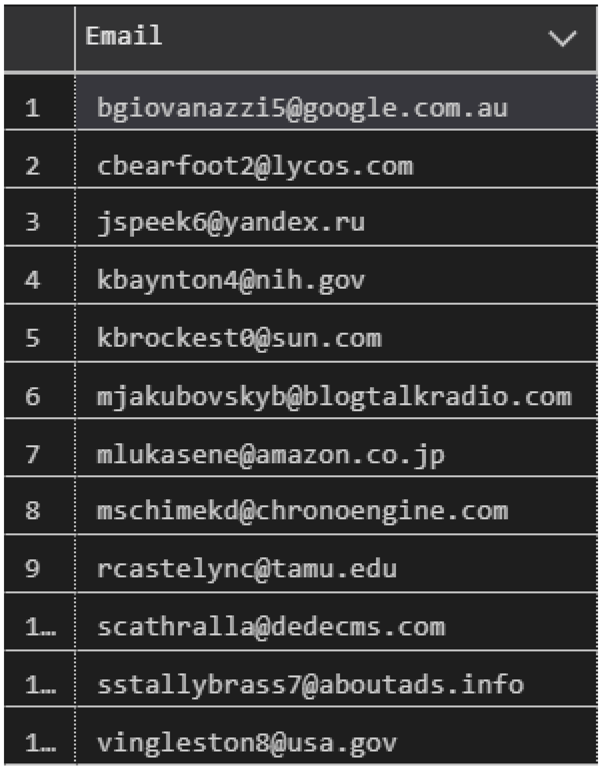
The problem occurs when a passenger wants to book a flight that connects (multiple flights), to resolve this obstacle, I added both associative entities for booking multiple flights and flights having multiple airlines. One solution is Flight_Sequence, basically, the TINYINT that shows the order of flights taken if multiple, else it would just be 1. (For this table I went in and found some connecting values to make connecting flight sequences possible as examples).

A similar problem occurs when a passenger wants to have multiple bookings or a booking has multiple passengers, although the complexity of this diagram and model can go on, I kept it at a minimum mainly focusing on a passenger being able to have multiple bookings, through 'Books' this can happen. If a passenger wants to have multiple members, another associative entity attribute of member count and identification should be placed.

FlightAssigment assesses the problem of having multiple airlines in one trip, to make it simpler, FlightAssignment also contains aircraft ID which can help with determining the right planes to board during the trip even if the airlines differ.

The focus of this project is solely based on solving the many-to-many relationships, such as the examples above. Keep in mind, that there are still a vast amount of complex obstacles that can occur in real-world situations, such as flaws in the data due to it being generated instead of collected from the real world.

Queries & Explanation

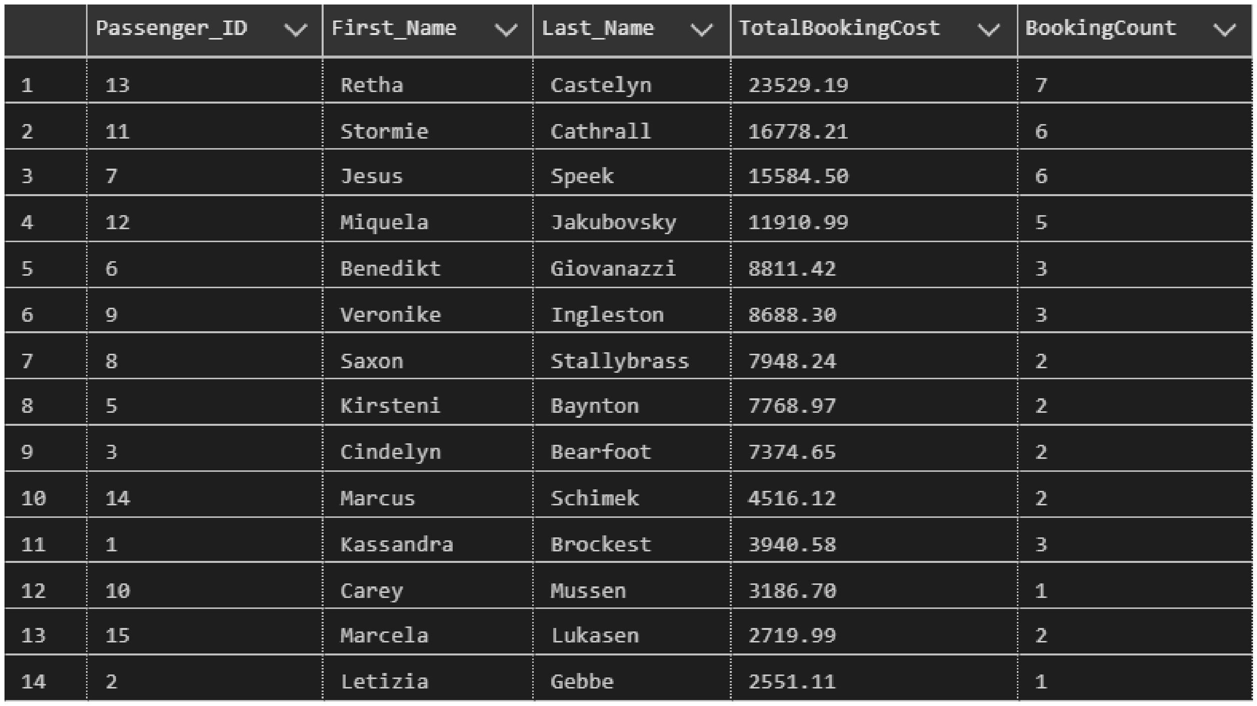
1. 

	Email
1	bgiovanazzi5@google.com.au
2	cbearfoot2@lycos.com
3	jspeek6@yandex.ru
4	kbaynton4@nih.gov
5	kbrockest0@sun.com
6	mjakubovskyb@blogtalkradio.com
7	mlukasene@amazon.co.jp
8	mschimekd@chronoengine.com
9	rcastelync@tamu.edu
1…	scathralla@dedecms.com
1…	sstallybrass7@aboutads.info
1…	vingleston8@usa.gov

```
-- Query returns all customers that have multiple bookings
-- Can be set up to use email as contact information to
-- notify the customer of their bookings, HAVING can be changed
-- to = 1, or 0, to show those customers. Returns email.
```

Returns email of customers that have multiple bookings can be altered to return other information based on that customer, email can be used for contacting purposes/changes.

2.

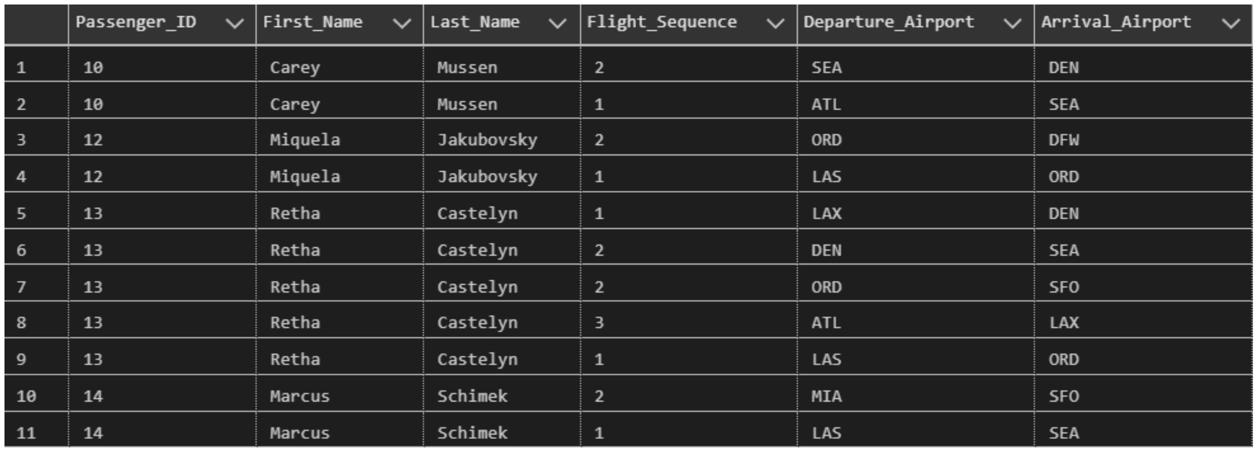


	Passenger_ID	First_Name	Last_Name	TotalBookingCost	BookingCount
1	13	Retha	Castelyn	23529.19	7
2	11	Stormie	Cathrall	16778.21	6
3	7	Jesus	Speek	15584.50	6
4	12	Miquela	Jakubovsky	11910.99	5
5	6	Benedikt	Giovanazzi	8811.42	3
6	9	Veronike	Ingleston	8688.30	3
7	8	Saxon	Stallybrass	7948.24	2
8	5	Kirsteni	Baynton	7768.97	2
9	3	Cindelyn	Bearfoot	7374.65	2
10	14	Marcus	Schimek	4516.12	2
11	1	Kassandra	Brockest	3940.58	3
12	10	Carey	Mussen	3186.70	1
13	15	Marcela	Lukasen	2719.99	2
14	2	Letizia	Gebbe	2551.11	1

```
-- Total Price of Passenger Bookings
-- Returns Passenger id, First name, Last name, calculates total booking
-- cost of all bookings that the passenger booked, and a count of the bookings.
-- Only Includes passengers with bookings, ordered by total cost descending.
```

Overlook of this can help the management roles track transactions and misconducts that can happen within the system, also this can help implement a leveling system based on booking count and/or costs.

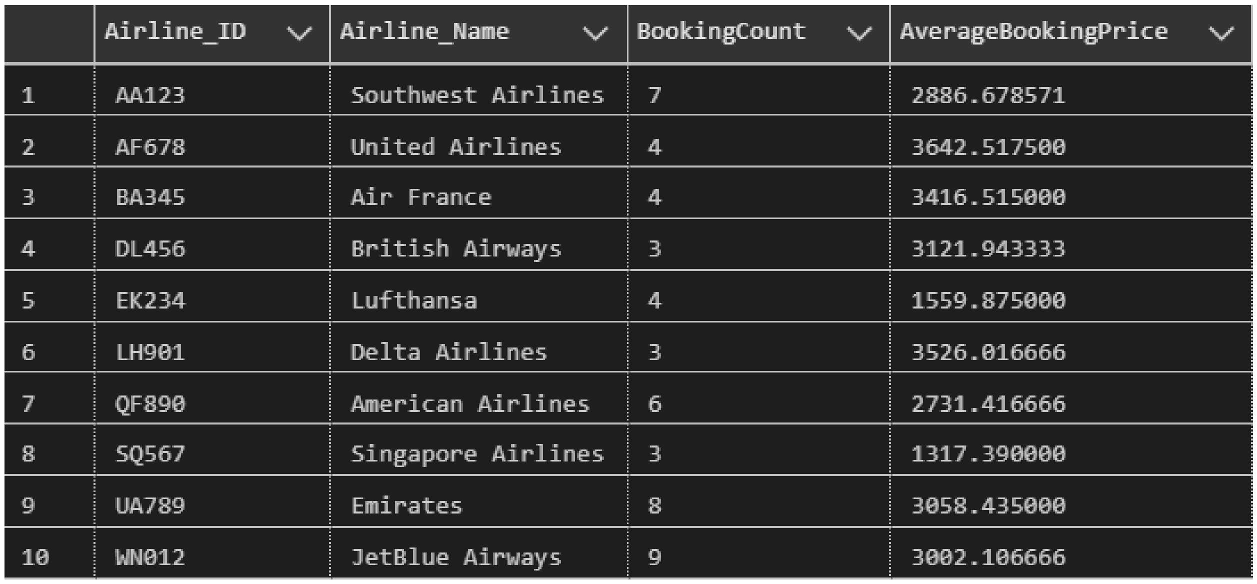
3.



	Passenger_ID	First_Name	Last_Name	Flight_Sequence	Departure_Airport	Arrival_Airport
1	10	Carey	Mussen	2	SEA	DEN
2	10	Carey	Mussen	1	ATL	SEA
3	12	Miquela	Jakubovsky	2	ORD	DFW
4	12	Miquela	Jakubovsky	1	LAS	ORD
5	13	Retha	Castelyn	1	LAX	DEN
6	13	Retha	Castelyn	2	DEN	SEA
7	13	Retha	Castelyn	2	ORD	SFO
8	13	Retha	Castelyn	3	ATL	LAX
9	13	Retha	Castelyn	1	LAS	ORD
10	14	Marcus	Schimek	2	MIA	SFO
11	14	Marcus	Schimek	1	LAS	SEA

```
-- Returns Passenger id, First name, Last name, Flight Sequence, Departure
Airport,
-- and Arrival Airport of passengers that have connecting flights in their
booking,
-- meaning multiple flight sequences, 1 being first and second connecting etc.
```

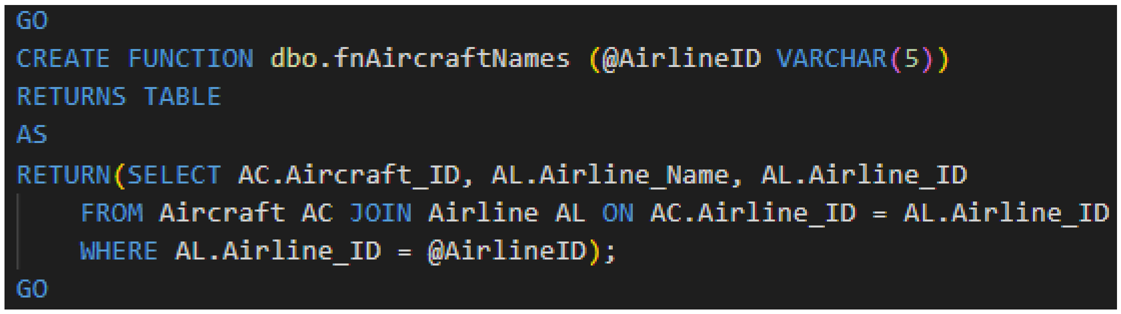
This solves one of the problems that occurred in phase 1, it is important to keep track of these things to make sure that customers with connecting flights are able to connect without missing their next flight(s).

4. 

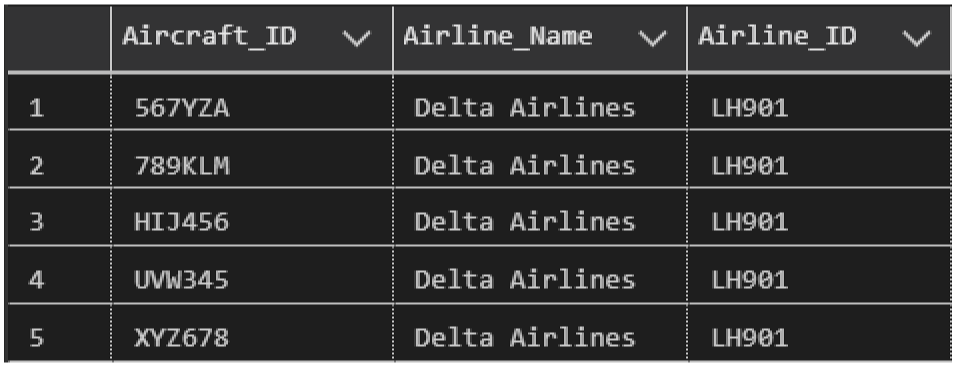
	Airline_ID	Airline_Name	BookingCount	AverageBookingPrice
1	AA123	Southwest Airlines	7	2886.678571
2	AF678	United Airlines	4	3642.517500
3	BA345	Air France	4	3416.515000
4	DL456	British Airways	3	3121.943333
5	EK234	Lufthansa	4	1559.875000
6	LH901	Delta Airlines	3	3526.016666
7	QF890	American Airlines	6	2731.416666
8	SQ567	Singapore Airlines	3	1317.390000
9	UA789	Emirates	8	3058.435000
10	WN012	JetBlue Airways	9	3002.106666

```
-- Airline Average Revenue Query
-- This query uses both a CTE and SELECT statement to
-- connect the tables and returns the Airline ID,
-- Airline Name, Booking Count, and Airline Revenue Average.
```

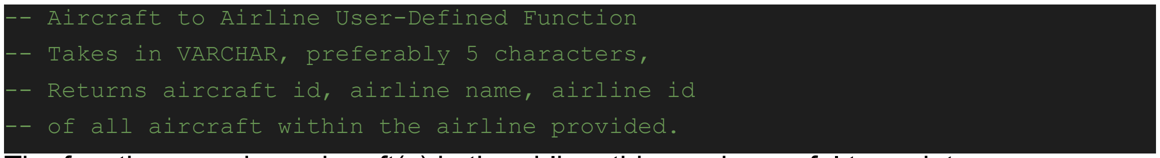
This is a query using a CTE which returns the count and total of the booking per airline, given that the data in this database is fairly low populated compared to the real world, this query would contain more and larger values in a sense.

5. 

```
GO
CREATE FUNCTION dbo.fnAircraftNames (@AirlineID VARCHAR(5))
RETURNS TABLE
AS
RETURN(SELECT AC.Aircraft_ID, AL.Airline_Name, AL.Airline_ID
    FROM Aircraft AC JOIN Airline AL ON AC.Airline_ID = AL.Airline_ID
    WHERE AL.Airline_ID = @AirlineID);
GO
```



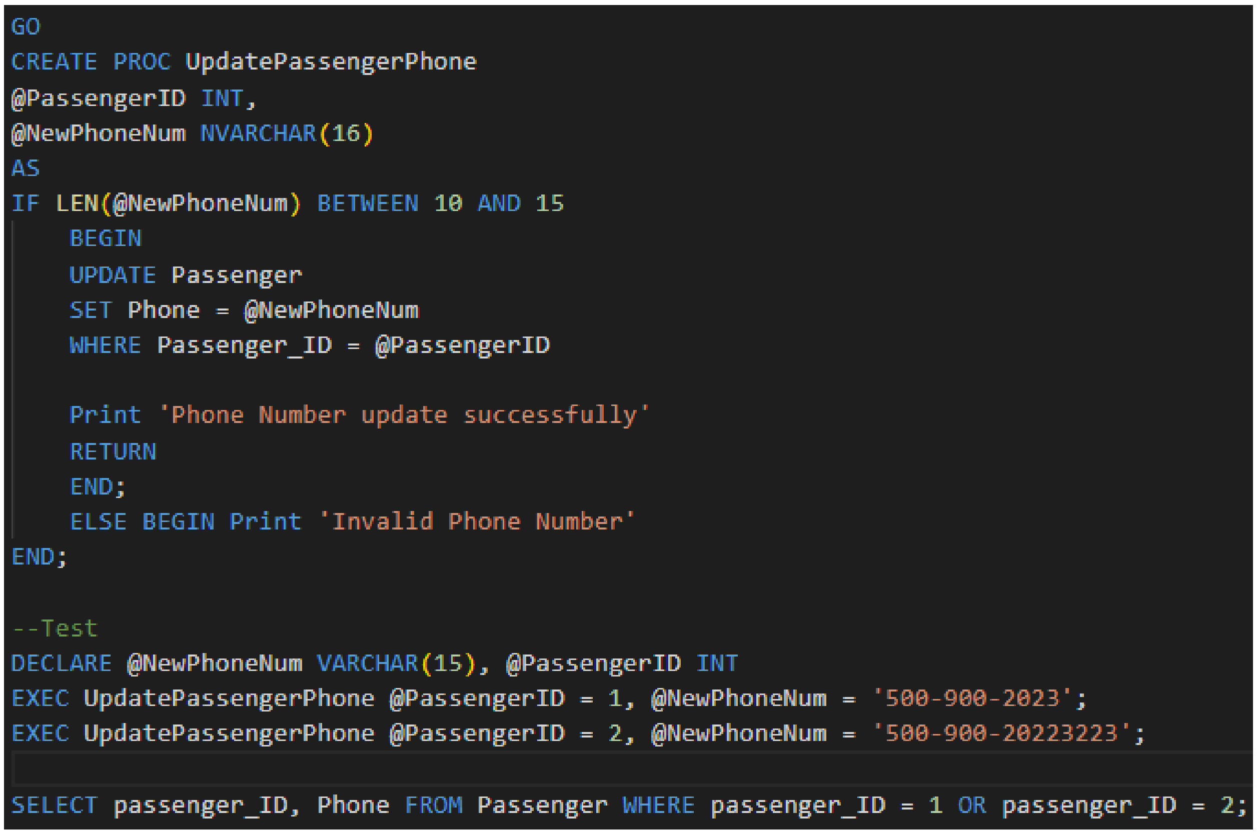
	Aircraft_ID	Airline_Name	Airline_ID
1	567YZA	Delta Airlines	LH901
2	789KLM	Delta Airlines	LH901
3	HIJ456	Delta Airlines	LH901
4	UVW345	Delta Airlines	LH901
5	XYZ678	Delta Airlines	LH901



```
-- Aircraft to Airline User-Defined Function
-- Takes in VARCHAR, preferably 5 characters,
-- Returns aircraft id, airline name, airline id
-- of all aircraft within the airline provided.
```

The function can show aircraft(s) in the airline, this can be useful to update or remove aircraft, or to update the capacity or model.

6.

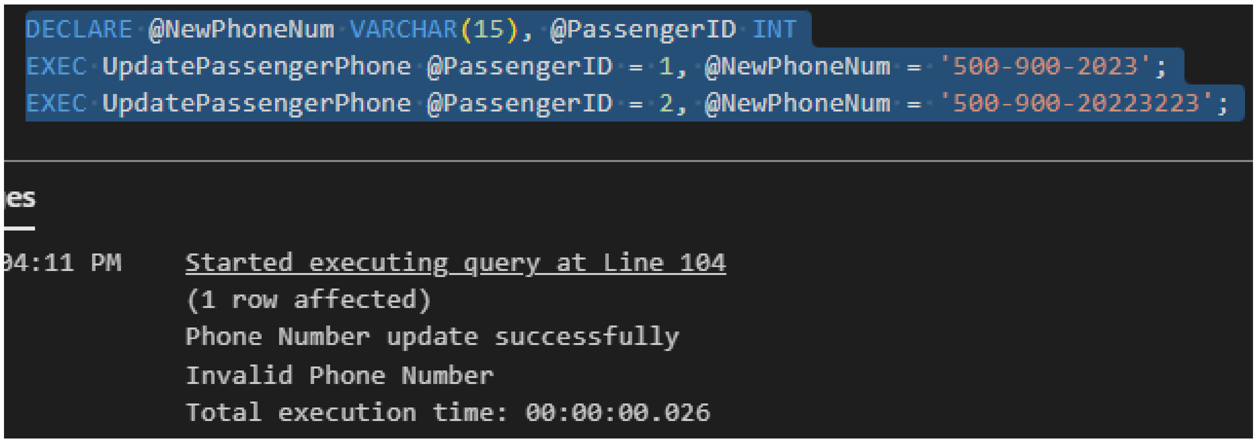


```
GO
CREATE PROC UpdatePassengerPhone
@PassengerID INT,
@NewPhoneNum NVARCHAR(16)
AS
IF LEN(@NewPhoneNum) BETWEEN 10 AND 15
    BEGIN
    UPDATE Passenger
    SET Phone = @NewPhoneNum
    WHERE Passenger_ID = @PassengerID

    Print 'Phone Number update successfully'
    RETURN
    END;
    ELSE BEGIN Print 'Invalid Phone Number'
END;

--Test
DECLARE @NewPhoneNum VARCHAR(15), @PassengerID INT
EXEC UpdatePassengerPhone @PassengerID = 1, @NewPhoneNum = '500-900-2023';
EXEC UpdatePassengerPhone @PassengerID = 2, @NewPhoneNum = '500-900-20223223';

SELECT passenger_ID, Phone FROM Passenger WHERE passenger_ID = 1 OR passenger_ID = 2;
```



```
DECLARE @NewPhoneNum VARCHAR(15), @PassengerID INT
EXEC UpdatePassengerPhone @PassengerID = 1, @NewPhoneNum = '500-900-2023';
EXEC UpdatePassengerPhone @PassengerID = 2, @NewPhoneNum = '500-900-20223223';
```

```
es

04:11 PM    Started executing query at Line 104
            (1 row affected)
            Phone Number update successfully
            Invalid Phone Number
            Total execution time: 00:00:00.026
```

```
-- Update Passenger's Phone Number Stored Procedure
-- Minimum Length is 10 Max Length is 15
-- May not be accurate in terms of real world phone numbers
-- The procedure can be easily altered to fit the requirements
-- of a real world situation
```

The stored procedure shows just one of the many ways that it can also be implemented on the other attributes and entities in the database. The usefulness of this stored procedure comes in handy when the user provides an invalid phone number, so both the user and program can respond accordingly.