BSc (Hons) in Information Technology

Specializing in Cyber Security

Year 2, Semester 1

IE2042 - Database Management Systems for Security

June 2022

Group Assignment

Database Design, Implementation and Security

Group Members Details

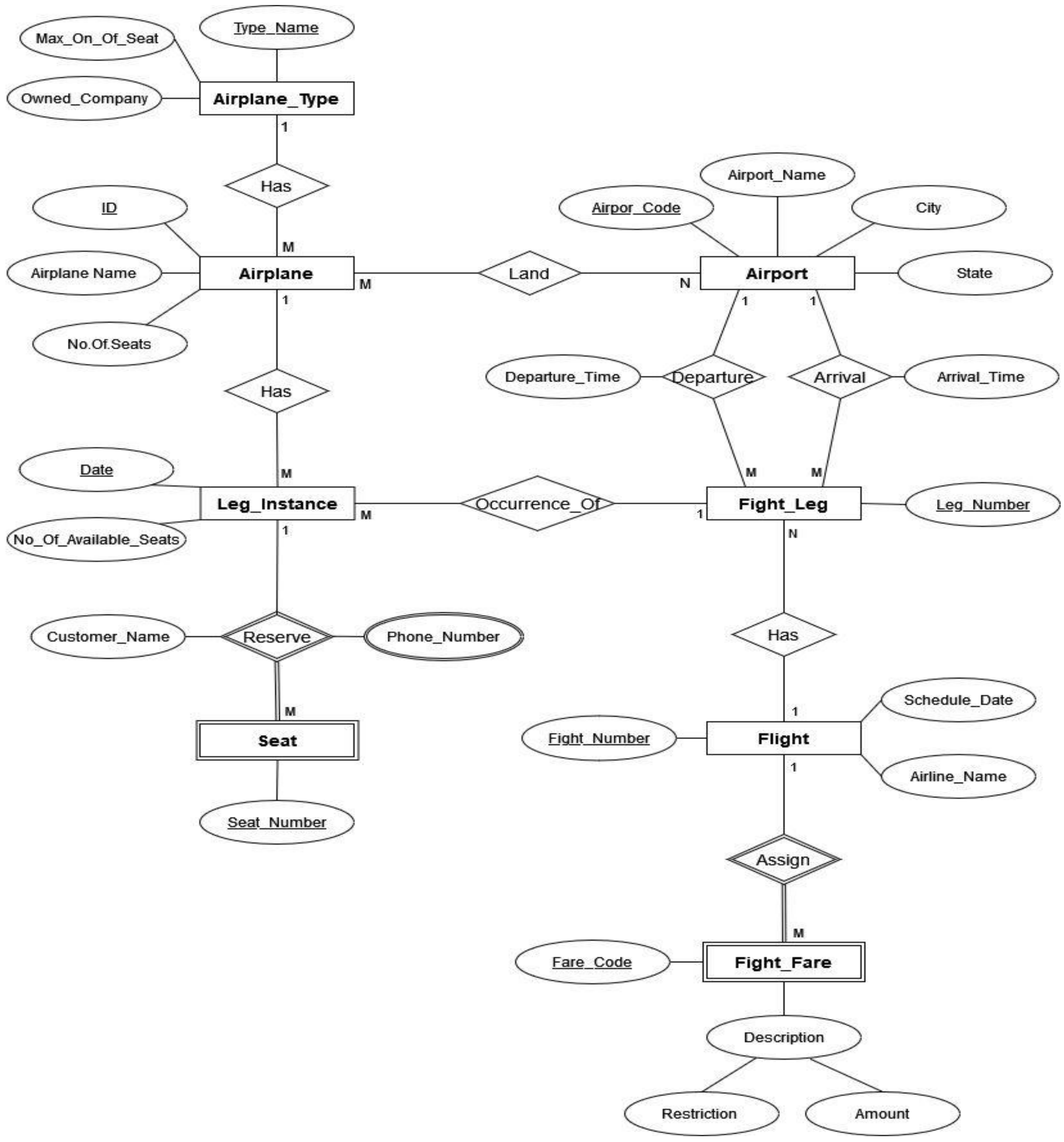
No	Student ID	Student Name	E-mail	Contact Number
01	IT21195402	Gunathilaka D.J.V	it21195402@my.sliit.lk	+94 714911894
02	IT21172496	Suhaif A.M.	It21172496@my.sliit.lk	+94 763816708
03	IT21163890	W.C.Gihan	it21163890@my.sliit.lk	+94 753146404
04	IT21261978	Sriskandarajah J.P	it21261978@my.sliit.lk	+94 76 1140271

Part 1

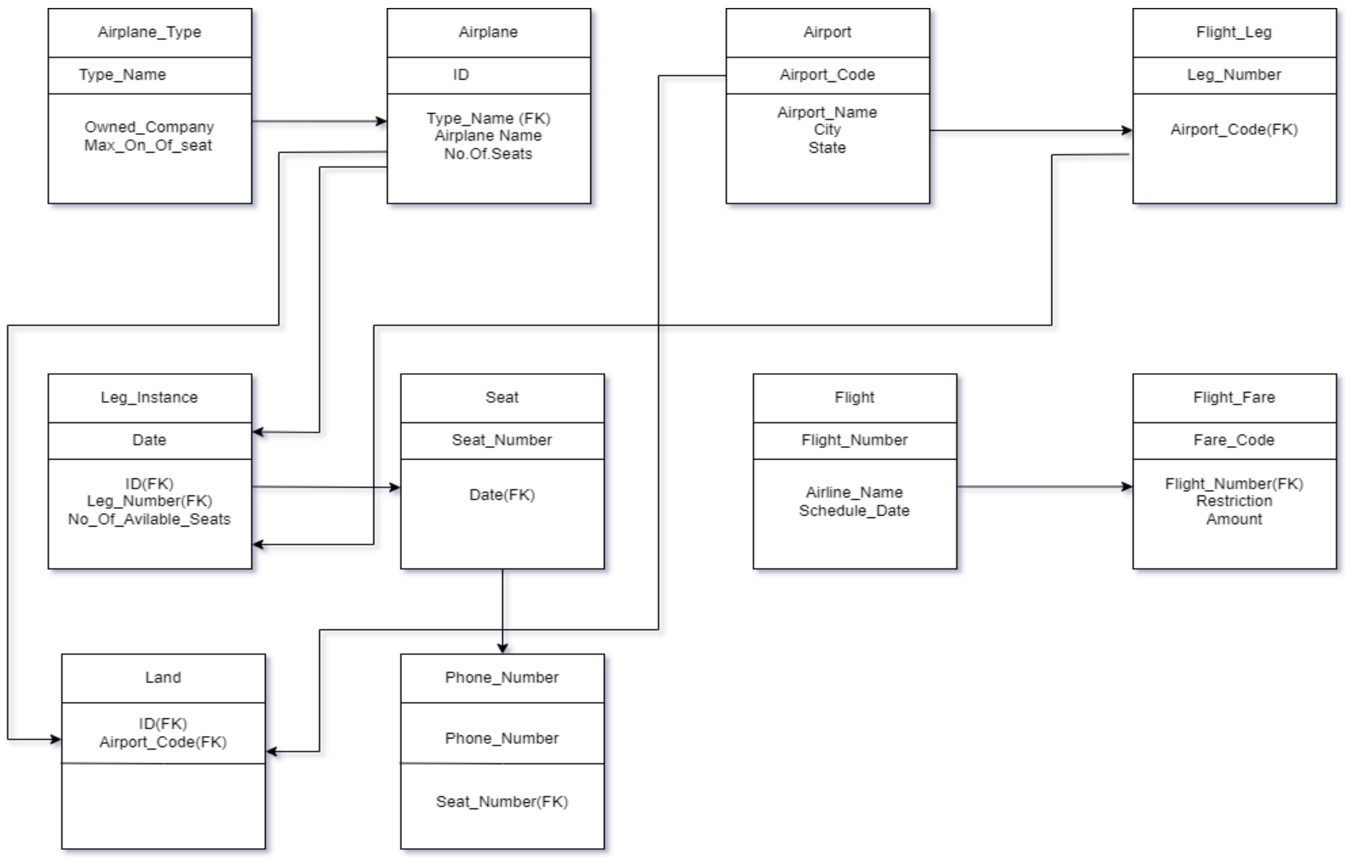
1. Assumptions

Seat and Flight Fare are weak entities.
Description is a composite attribute.
Phone Number is a multi-valued attribute.
Reserve, Departure and Arrival are descriptive attributes.

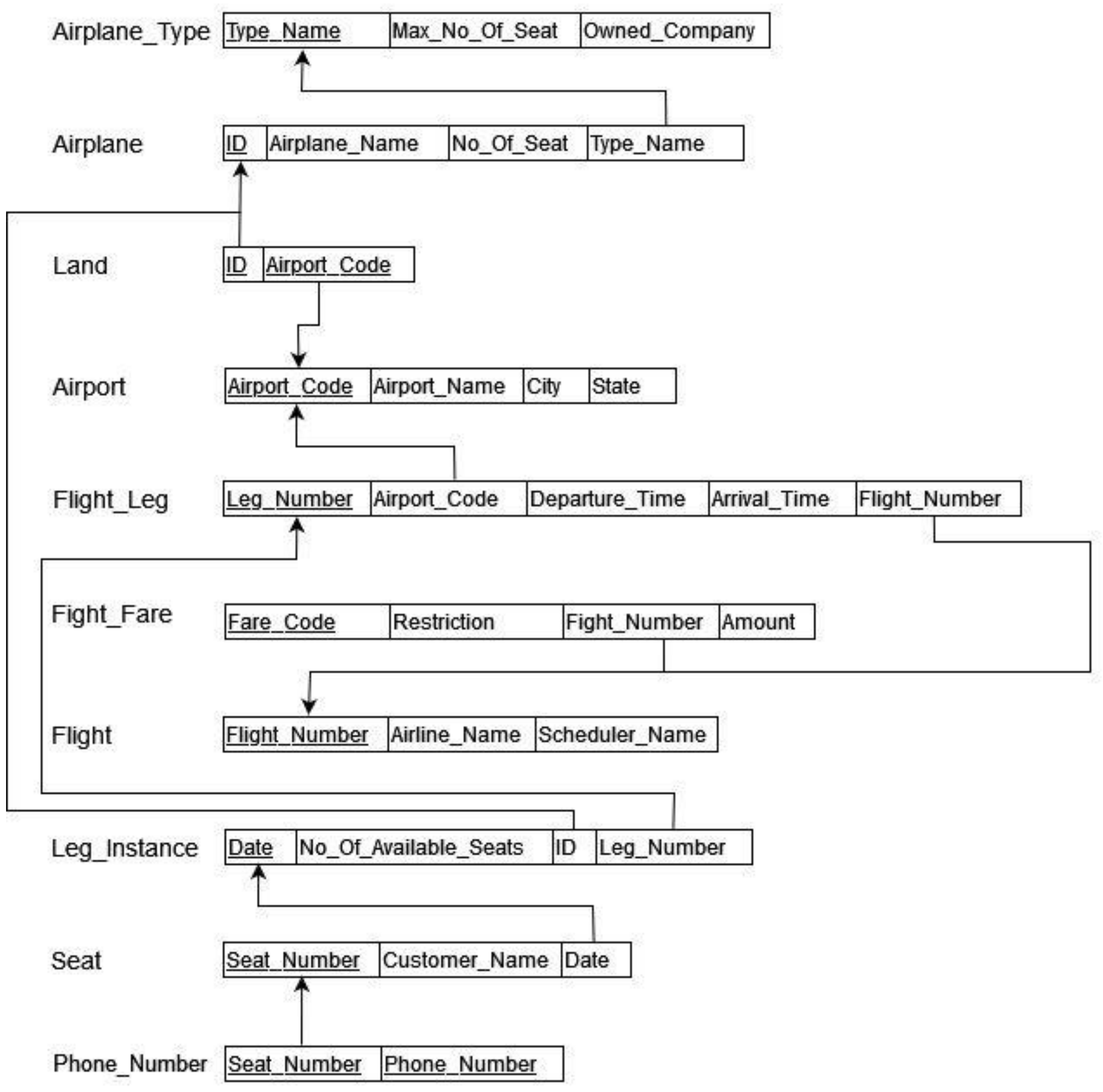
2. ER (Entity Relationship) Diagram



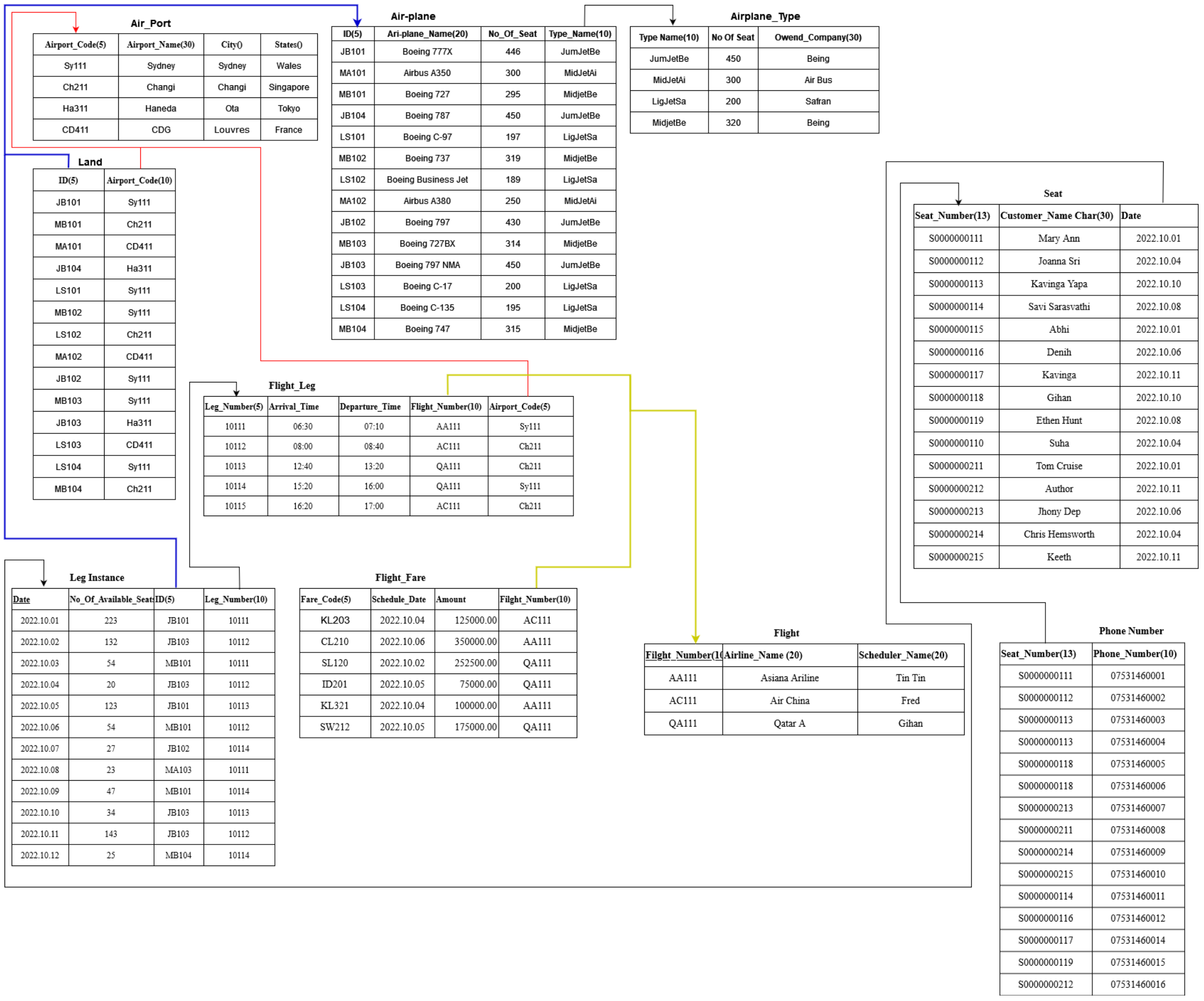
Logical Model



Relational Schema 3rd Normal Form



Logical Model in MS SQL



Air_Port

Airport_Code(5)	Airport_Name(30)	City()	States()
Sy111	Sydney	Sydney	Wales
Ch211	Changi	Changi	Singapore
Ha311	Haneda	Ota	Tokyo
CD411	CDG	Louvres	France

Air-plane

ID(5)	Ari-plane_Name(20)	No_Of_Seat	Type_Name(10)
JB101	Boeing 777X	446	JumJetBe
MA101	Airbus A350	300	MidJetAi
MB101	Boeing 727	295	MidjetBe
JB104	Boeing 787	450	JumJetBe
LS101	Boeing C-97	197	LigJetSa
MB102	Boeing 737	319	MidjetBe
LS102	Boeing Business Jet	189	LigJetSa
MA102	Airbus A380	250	MidJetAi
JB102	Boeing 797	430	JumJetBe
MB103	Boeing 727BX	314	MidjetBe
JB103	Boeing 797 NMA	450	JumJetBe
LS103	Boeing C-17	200	LigJetSa
LS104	Boeing C-135	195	LigJetSa
MB104	Boeing 747	315	MidjetBe

Airplane_Type

Type Name(10)	No Of Seat	Owend_Company(30)
JumJetBe	450	Being
MidJetAi	300	Air Bus
LigJetSa	200	Safran
MidjetBe	320	Being

Land

ID(5)	Airport_Code(10)
JB101	Sy111
MB101	Ch211
MA101	CD411
JB104	Ha311
LS101	Sy111
MB102	Sy111
LS102	Ch211
MA102	CD411
JB102	Sy111
MB103	Sy111
JB103	Ha311
LS103	CD411
LS104	Sy111
MB104	Ch211

Flight_Leg

Leg_Number(5)	Arrival_Time	Departure_Time	Flight_Number(10)	Airport_Code(5)
10111	06:30	07:10	AA111	Sy111
10112	08:00	08:40	AC111	Ch211
10113	12:40	13:20	QA111	Ch211
10114	15:20	16:00	QA111	Sy111
10115	16:20	17:00	AC111	Ch211

Seat

Seat_Number(13)	Customer_Name Char(30)	Date
S0000000111	Mary Ann	2022.10.01
S0000000112	Joanna Sri	2022.10.04
S0000000113	Kavinga Yapa	2022.10.10
S0000000114	Savi Sarasvathi	2022.10.08
S0000000115	Abhi	2022.10.01
S0000000116	Denih	2022.10.06
S0000000117	Kavinga	2022.10.11
S0000000118	Gihan	2022.10.10
S0000000119	Ethen Hunt	2022.10.08
S0000000110	Suha	2022.10.04
S0000000211	Tom Cruise	2022.10.01
S0000000212	Author	2022.10.11
S0000000213	Jhony Dep	2022.10.06
S0000000214	Chris Hemsworth	2022.10.04
S0000000215	Keeth	2022.10.11

Leg Instance

Date	No_Of_Available_Seats	ID(5)	Leg_Number(10)
2022.10.01	223	JB101	10111
2022.10.02	132	JB103	10112
2022.10.03	54	MB101	10111
2022.10.04	20	JB103	10112
2022.10.05	123	JB101	10113
2022.10.06	54	MB101	10112
2022.10.07	27	JB102	10114
2022.10.08	23	MA103	10111
2022.10.09	47	MB101	10114
2022.10.10	34	JB103	10113
2022.10.11	143	JB103	10112
2022.10.12	25	MB104	10114

Flight_Fare

Fare_Code(5)	Schedule_Date	Amount	Filght_Number(10)
KL203	2022.10.04	125000.00	AC111
CL210	2022.10.06	350000.00	AA111
SL120	2022.10.02	252500.00	QA111
ID201	2022.10.05	75000.00	QA111
KL321	2022.10.04	100000.00	AA111
SW212	2022.10.05	175000.00	QA111

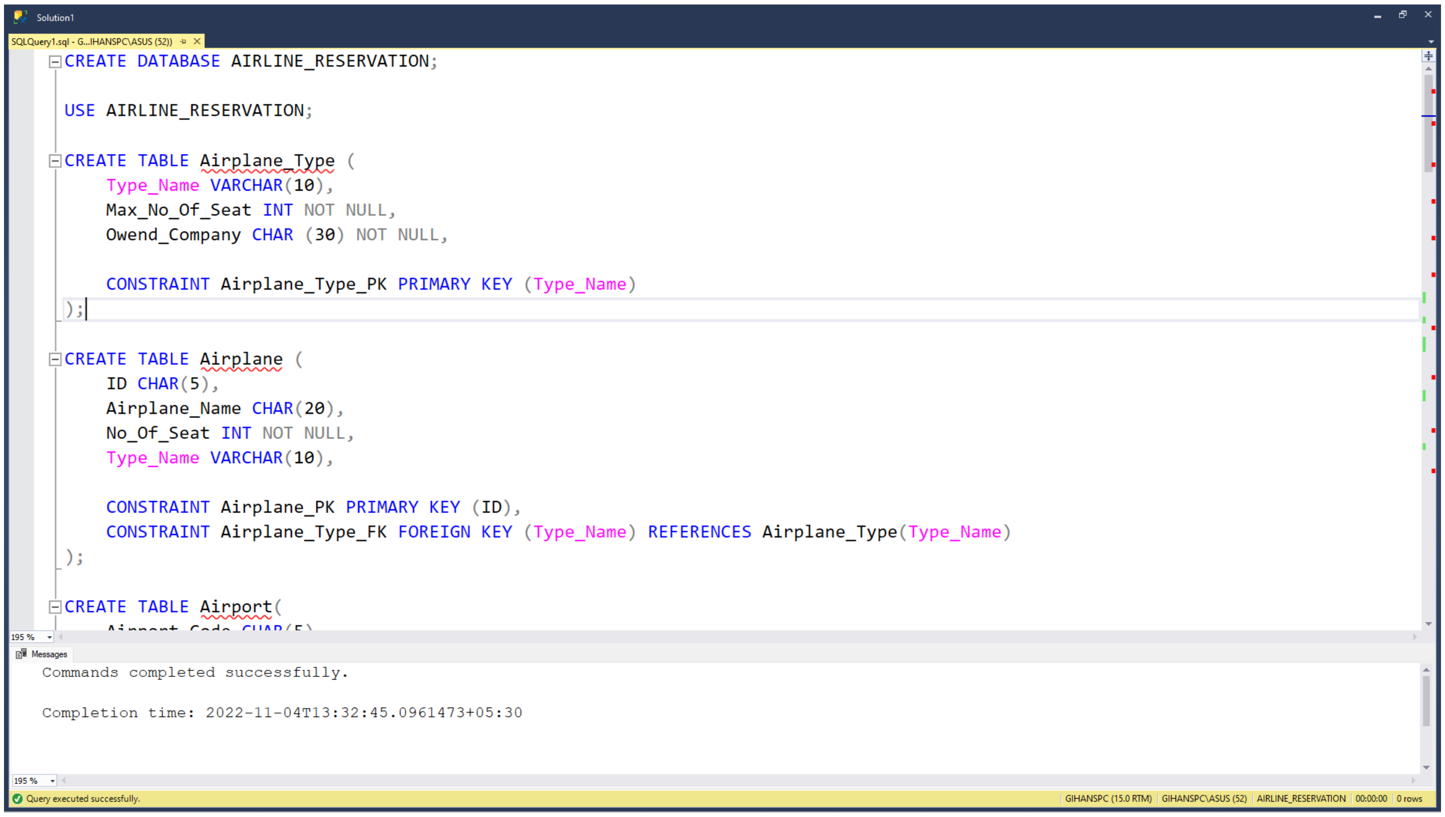
Flight

Filght_Number(1	Airline_Name (20)	Scheduler_Name(20)
AA111	Asiana Ariline	Tin Tin
AC111	Air China	Fred
QA111	Qatar A	Gihan

Phone Number

Seat_Number(13)	Phone_Number(10)
S0000000111	07531460001
S0000000112	07531460002
S0000000113	07531460003
S0000000113	07531460004
S0000000118	07531460005
S0000000118	07531460006
S0000000213	07531460007
S0000000211	07531460008
S0000000214	07531460009
S0000000215	07531460010
S0000000114	07531460011
S0000000116	07531460012
S0000000117	07531460014
S0000000119	07531460015
S0000000212	07531460016

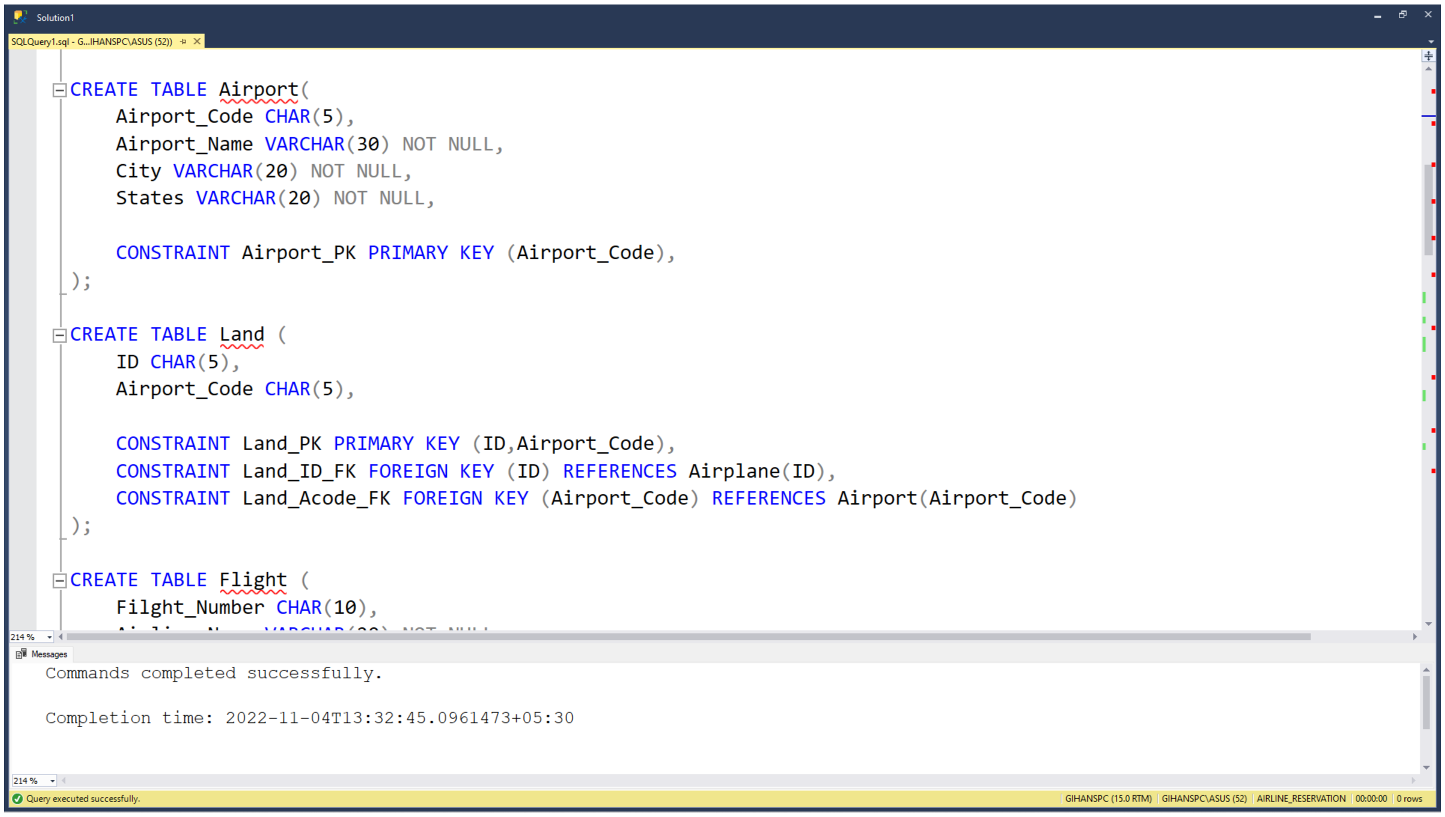
3. Data Base Implement



```sql
CREATE DATABASE AIRLINE_RESERVATION;

USE AIRLINE_RESERVATION;

CREATE TABLE Airplane_Type (
    Type_Name VARCHAR(10),
    Max_No_Of_Seat INT NOT NULL,
    Owend_Company CHAR (30) NOT NULL,

    CONSTRAINT Airplane_Type_PK PRIMARY KEY (Type_Name)
);

CREATE TABLE Airplane (
    ID CHAR(5),
    Airplane_Name CHAR(20),
    No_Of_Seat INT NOT NULL,
    Type_Name VARCHAR(10),

    CONSTRAINT Airplane_PK PRIMARY KEY (ID),
    CONSTRAINT Airplane_Type_FK FOREIGN KEY (Type_Name) REFERENCES Airplane_Type(Type_Name)
);

CREATE TABLE Airport(
```

```
Commands completed successfully.

Completion time: 2022-11-04T13:32:45.0961473+05:30
```

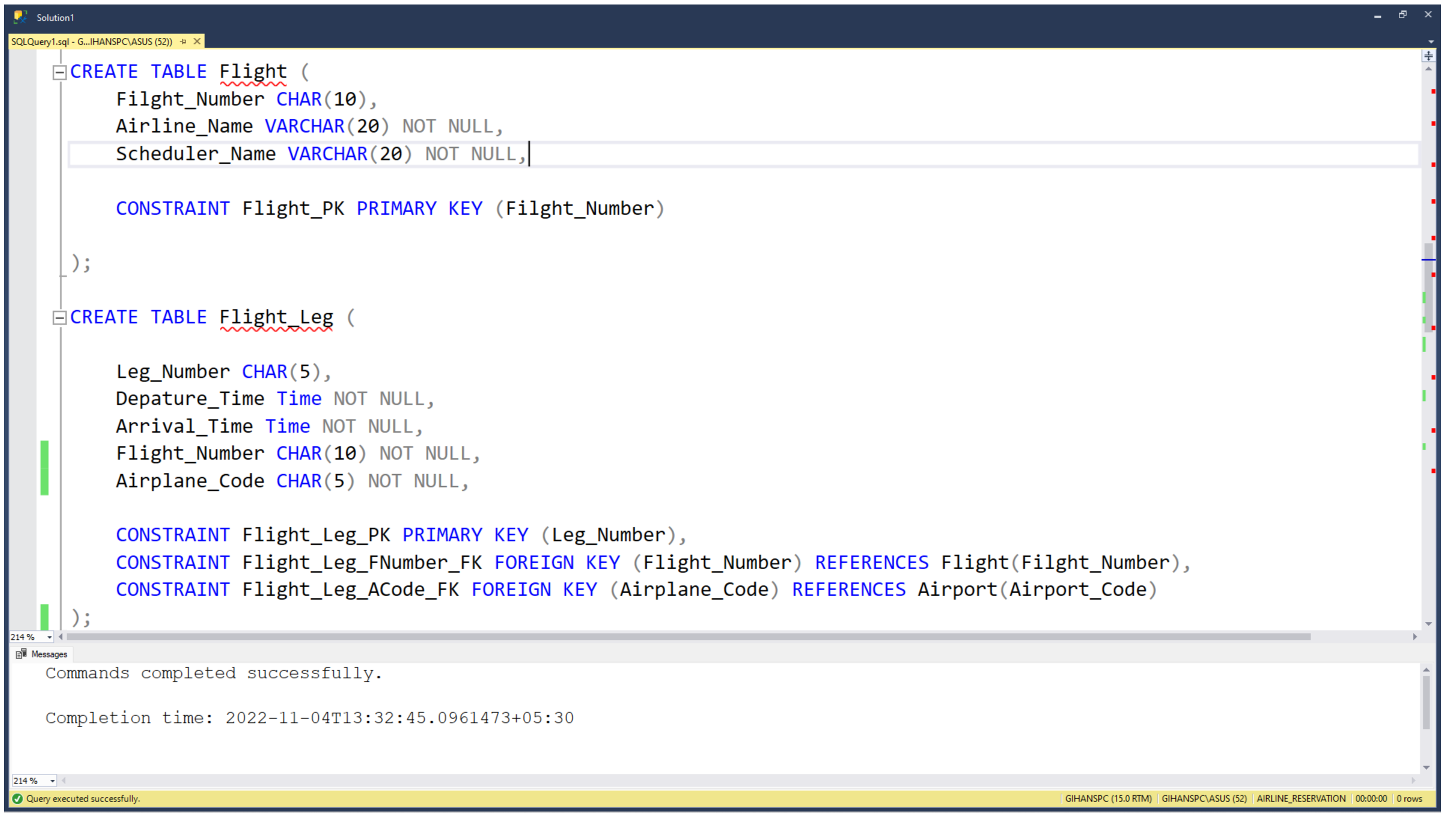
Query executed successfully. GIHANSPC (15.0 RTM) | GIHANSPC\ASUS (52) | AIRLINE_RESERVATION | 00:00:00 | 0 rows



```sql
CREATE TABLE Airport(
    Airport_Code CHAR(5),
    Airport_Name VARCHAR(30) NOT NULL,
    City VARCHAR(20) NOT NULL,
    States VARCHAR(20) NOT NULL,

    CONSTRAINT Airport_PK PRIMARY KEY (Airport_Code),
);

CREATE TABLE Land (
    ID CHAR(5),
    Airport_Code CHAR(5),

    CONSTRAINT Land_PK PRIMARY KEY (ID,Airport_Code),
    CONSTRAINT Land_ID_FK FOREIGN KEY (ID) REFERENCES Airplane(ID),
    CONSTRAINT Land_Acode_FK FOREIGN KEY (Airport_Code) REFERENCES Airport(Airport_Code)
);

CREATE TABLE Flight (
    Filght_Number CHAR(10),
```

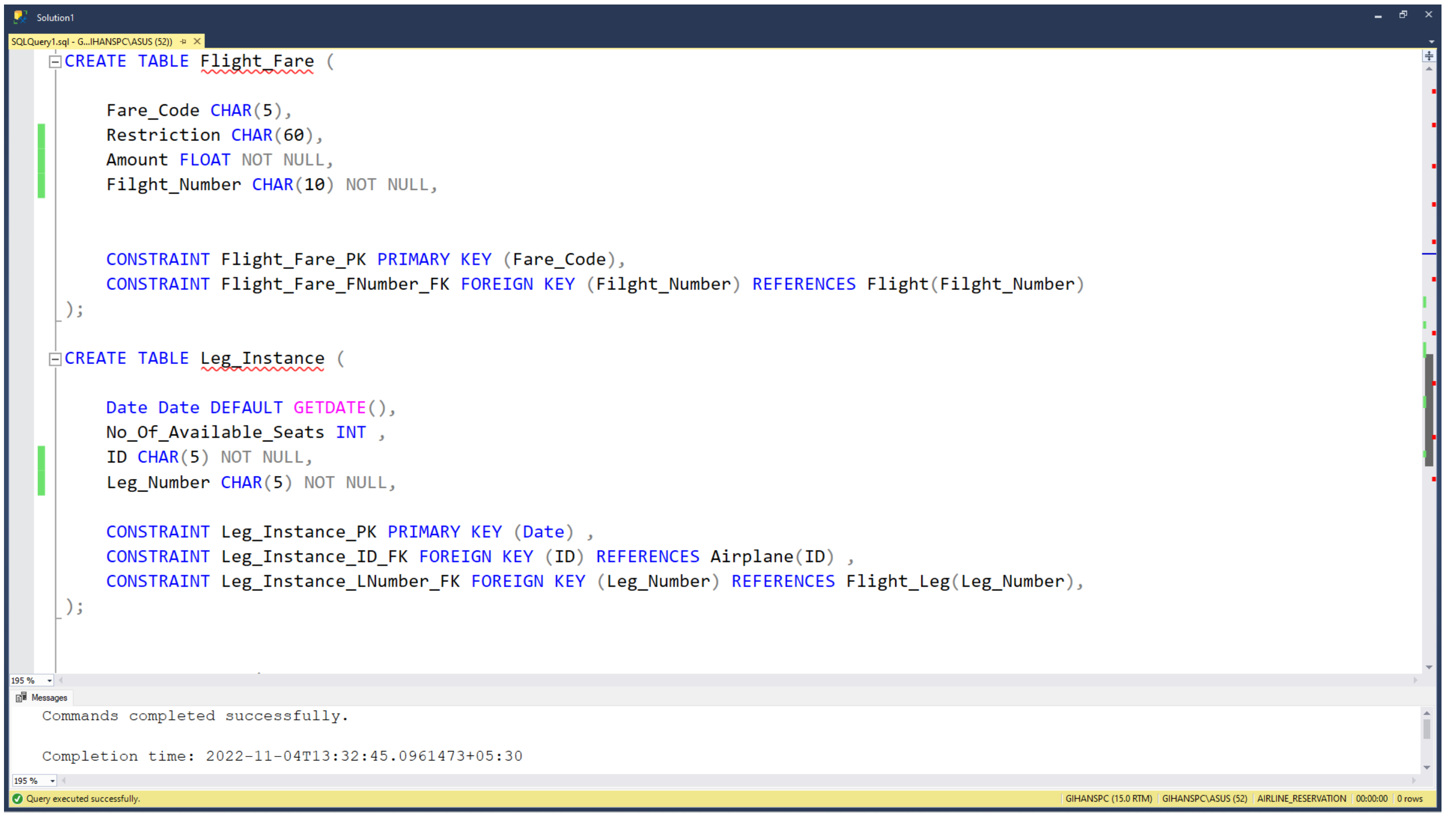
```
Commands completed successfully.

Completion time: 2022-11-04T13:32:45.0961473+05:30
```

Query executed successfully. GIHANSPC (15.0 RTM) | GIHANSPC\ASUS (52) | AIRLINE_RESERVATION | 00:00:00 | 0 rows



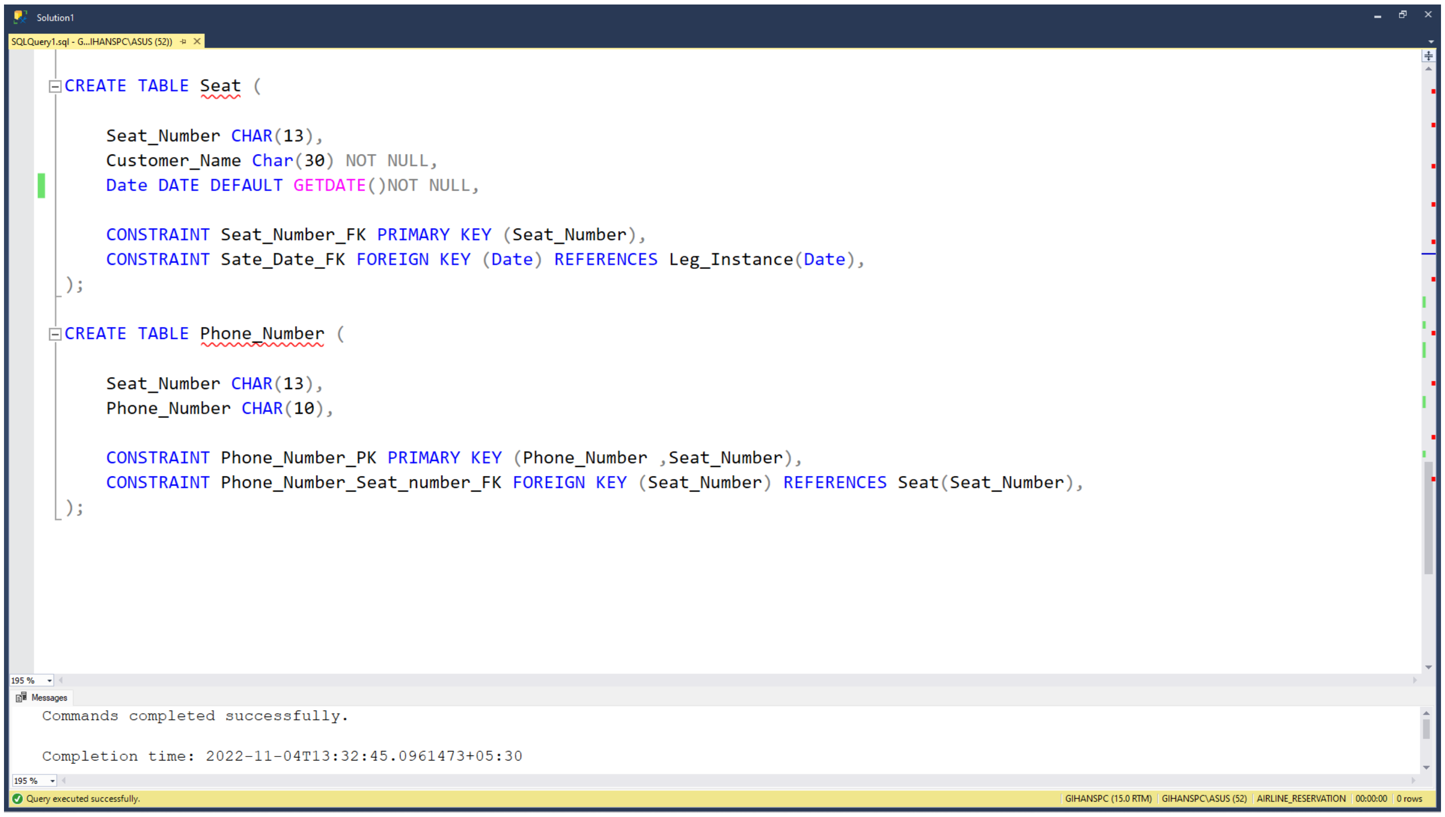
```
CREATE TABLE Flight (
    Filght_Number CHAR(10),
    Airline_Name VARCHAR(20) NOT NULL,
    Scheduler_Name VARCHAR(20) NOT NULL,

    CONSTRAINT Flight_PK PRIMARY KEY (Filght_Number)

);

CREATE TABLE Flight_Leg (

    Leg_Number CHAR(5),
    Depature_Time Time NOT NULL,
    Arrival_Time Time NOT NULL,
    Flight_Number CHAR(10) NOT NULL,
    Airplane_Code CHAR(5) NOT NULL,

    CONSTRAINT Flight_Leg_PK PRIMARY KEY (Leg_Number),
    CONSTRAINT Flight_Leg_FNumber_FK FOREIGN KEY (Flight_Number) REFERENCES Flight(Filght_Number),
    CONSTRAINT Flight_Leg_ACode_FK FOREIGN KEY (Airplane_Code) REFERENCES Airport(Airport_Code)
);
```

```
Commands completed successfully.

Completion time: 2022-11-04T13:32:45.0961473+05:30
```



```
CREATE TABLE Flight_Fare (

    Fare_Code CHAR(5),
    Restriction CHAR(60),
    Amount FLOAT NOT NULL,
    Filght_Number CHAR(10) NOT NULL,


    CONSTRAINT Flight_Fare_PK PRIMARY KEY (Fare_Code),
    CONSTRAINT Flight_Fare_FNumber_FK FOREIGN KEY (Filght_Number) REFERENCES Flight(Filght_Number)
);

CREATE TABLE Leg_Instance (

    Date Date DEFAULT GETDATE(),
    No_Of_Available_Seats INT ,
    ID CHAR(5) NOT NULL,
    Leg_Number CHAR(5) NOT NULL,

    CONSTRAINT Leg_Instance_PK PRIMARY KEY (Date) ,
    CONSTRAINT Leg_Instance_ID_FK FOREIGN KEY (ID) REFERENCES Airplane(ID) ,
    CONSTRAINT Leg_Instance_LNumber_FK FOREIGN KEY (Leg_Number) REFERENCES Flight_Leg(Leg_Number),
);
```

```
Commands completed successfully.

Completion time: 2022-11-04T13:32:45.0961473+05:30
```

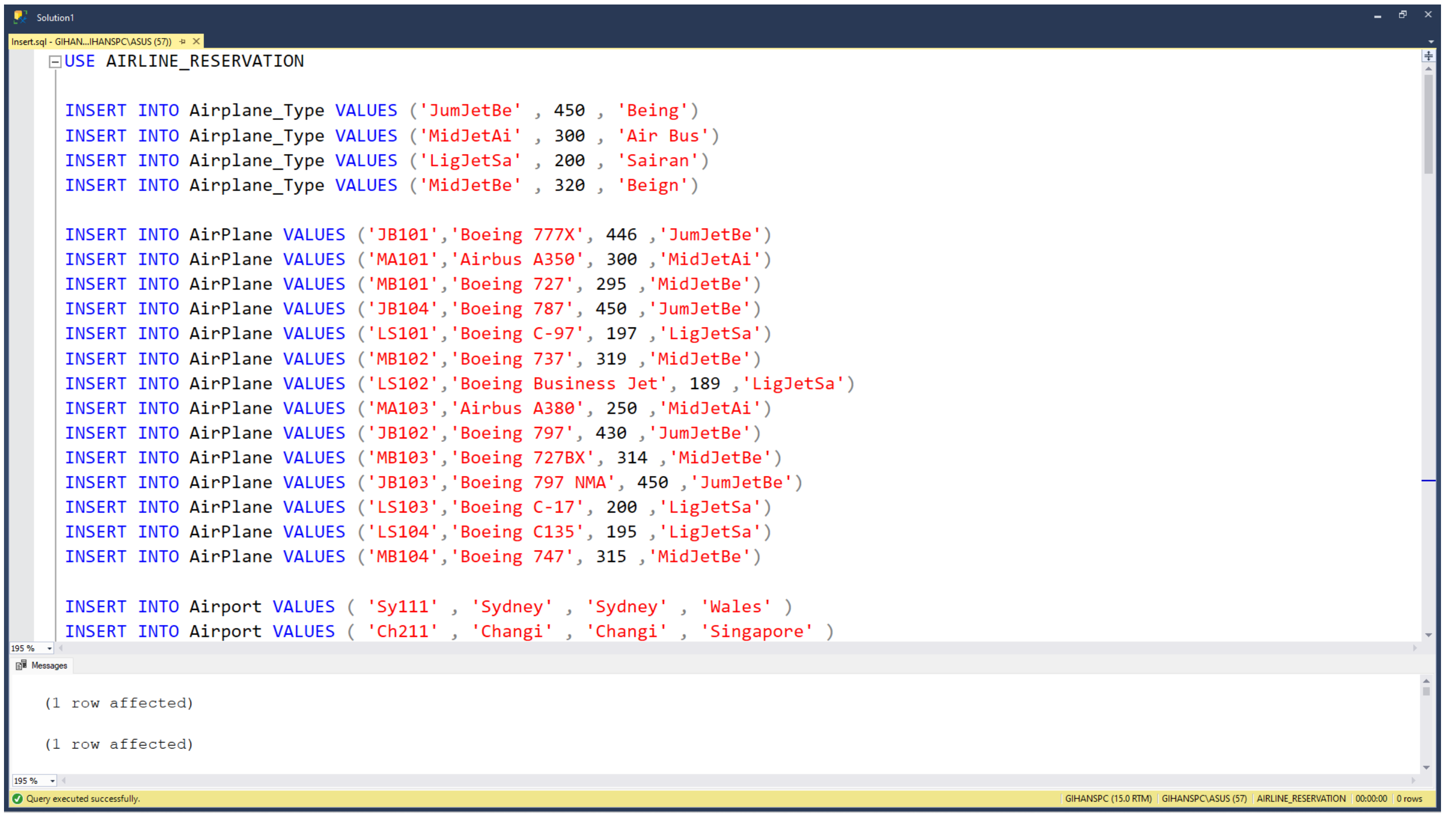


```
CREATE TABLE Seat (

    Seat_Number CHAR(13),
    Customer_Name Char(30) NOT NULL,
    Date DATE DEFAULT GETDATE()NOT NULL,

    CONSTRAINT Seat_Number_FK PRIMARY KEY (Seat_Number),
    CONSTRAINT Sate_Date_FK FOREIGN KEY (Date) REFERENCES Leg_Instance(Date),
);

CREATE TABLE Phone_Number (

    Seat_Number CHAR(13),
    Phone_Number CHAR(10),

    CONSTRAINT Phone_Number_PK PRIMARY KEY (Phone_Number ,Seat_Number),
    CONSTRAINT Phone_Number_Seat_number_FK FOREIGN KEY (Seat_Number) REFERENCES Seat(Seat_Number),
);
```

```
Commands completed successfully.

Completion time: 2022-11-04T13:32:45.0961473+05:30
```

Query executed successfully. | GIHANSPC (15.0 RTM) | GIHANSPC\ASUS (52) | AIRLINE_RESERVATION | 00:00:00 | 0 rows

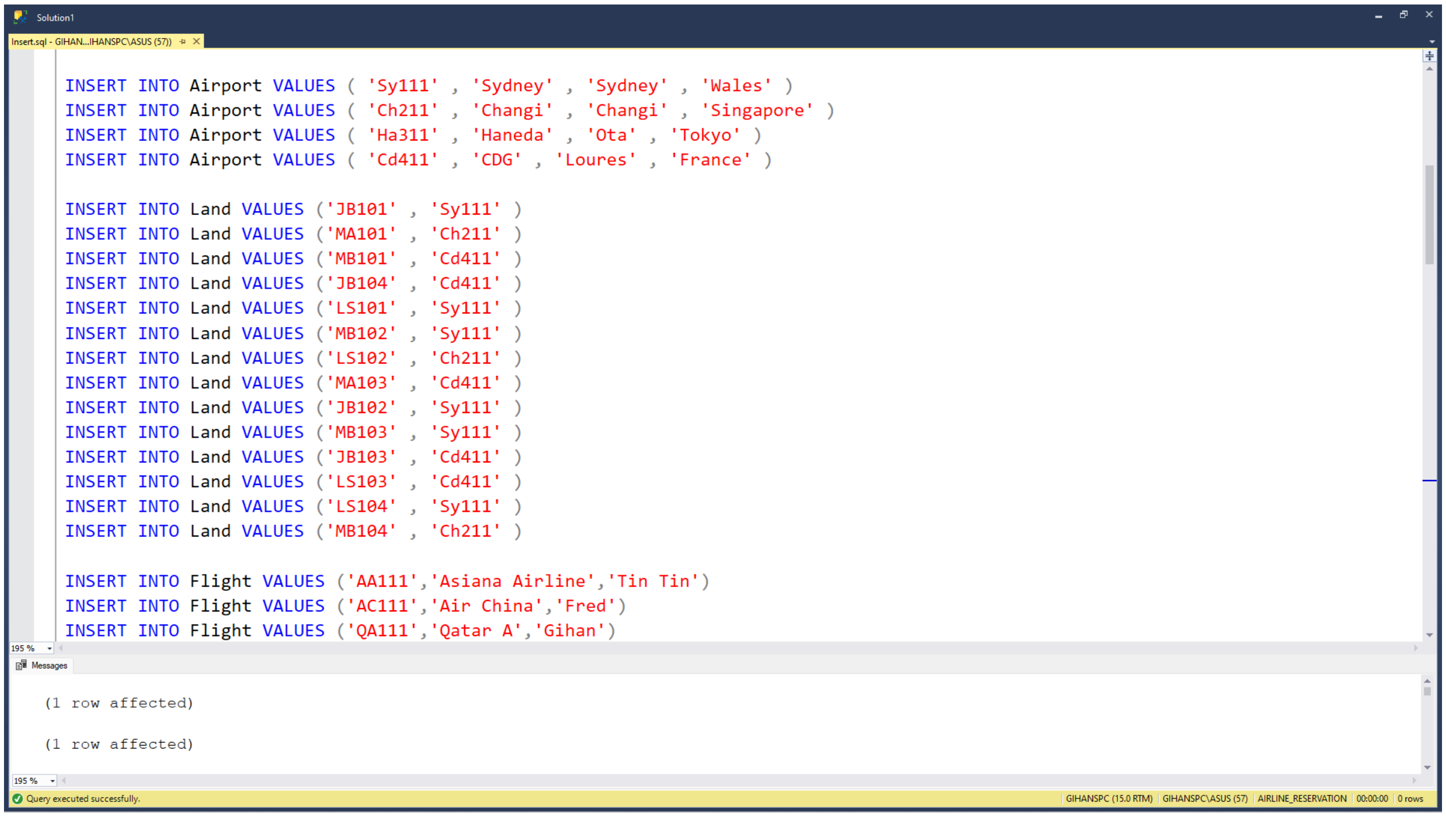
4. Insert data into tables



```
USE AIRLINE_RESERVATION

INSERT INTO Airplane_Type VALUES ('JumJetBe' , 450 , 'Being')
INSERT INTO Airplane_Type VALUES ('MidJetAi' , 300 , 'Air Bus')
INSERT INTO Airplane_Type VALUES ('LigJetSa' , 200 , 'Sairan')
INSERT INTO Airplane_Type VALUES ('MidJetBe' , 320 , 'Beign')

INSERT INTO AirPlane VALUES ('JB101','Boeing 777X', 446 ,'JumJetBe')
INSERT INTO AirPlane VALUES ('MA101','Airbus A350', 300 ,'MidJetAi')
INSERT INTO AirPlane VALUES ('MB101','Boeing 727', 295 ,'MidJetBe')
INSERT INTO AirPlane VALUES ('JB104','Boeing 787', 450 ,'JumJetBe')
INSERT INTO AirPlane VALUES ('LS101','Boeing C-97', 197 ,'LigJetSa')
INSERT INTO AirPlane VALUES ('MB102','Boeing 737', 319 ,'MidJetBe')
INSERT INTO AirPlane VALUES ('LS102','Boeing Business Jet', 189 ,'LigJetSa')
INSERT INTO AirPlane VALUES ('MA103','Airbus A380', 250 ,'MidJetAi')
INSERT INTO AirPlane VALUES ('JB102','Boeing 797', 430 ,'JumJetBe')
INSERT INTO AirPlane VALUES ('MB103','Boeing 727BX', 314 ,'MidJetBe')
INSERT INTO AirPlane VALUES ('JB103','Boeing 797 NMA', 450 ,'JumJetBe')
INSERT INTO AirPlane VALUES ('LS103','Boeing C-17', 200 ,'LigJetSa')
INSERT INTO AirPlane VALUES ('LS104','Boeing C135', 195 ,'LigJetSa')
INSERT INTO AirPlane VALUES ('MB104','Boeing 747', 315 ,'MidJetBe')

INSERT INTO Airport VALUES ( 'Sy111' , 'Sydney' , 'Sydney' , 'Wales' )
INSERT INTO Airport VALUES ( 'Ch211' , 'Changi' , 'Changi' , 'Singapore' )
```

```
(1 row affected)

(1 row affected)
```

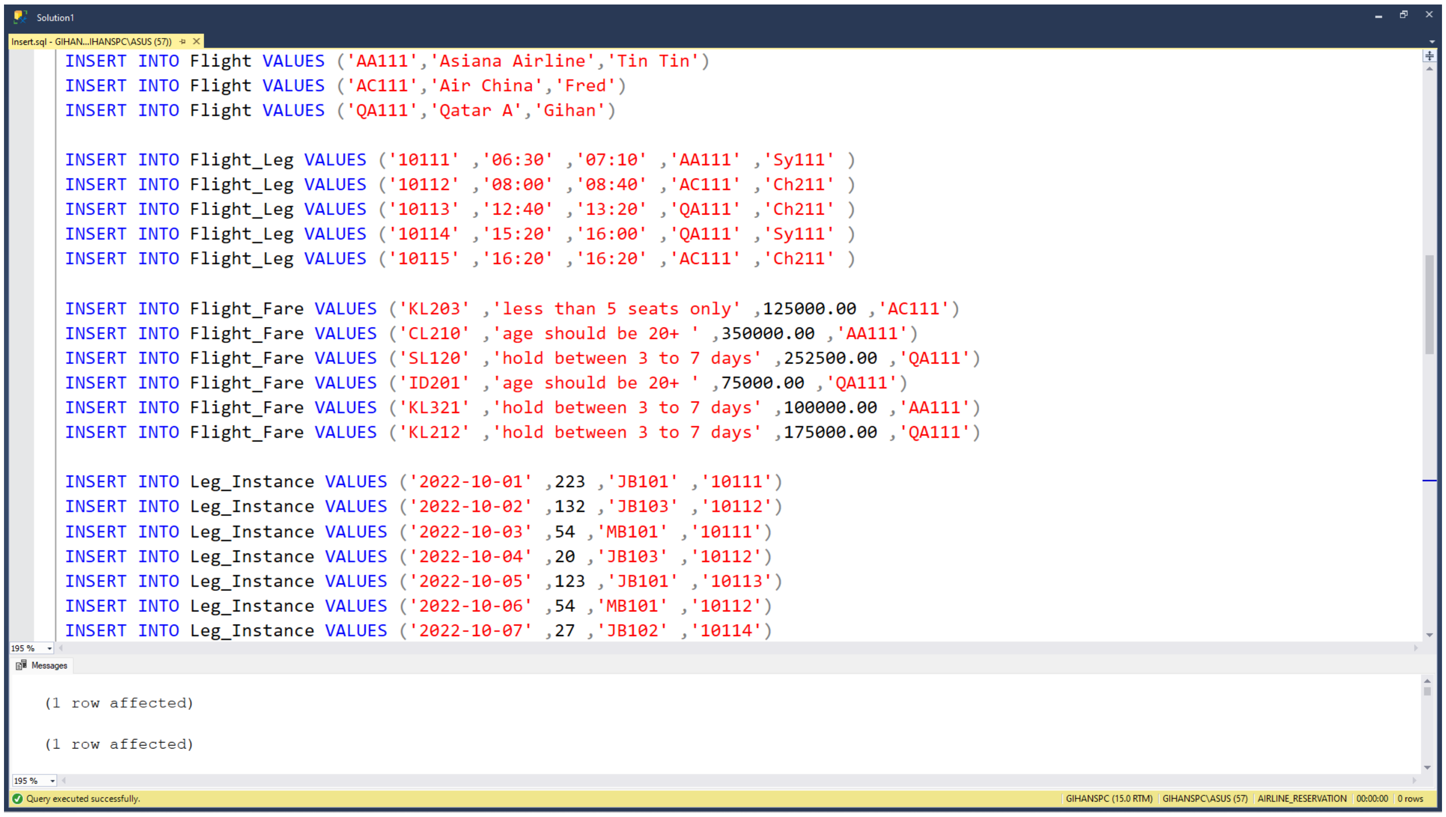
Query executed successfully. GIHANSPC (15.0 RTM) | GIHANSPC\ASUS (57) | AIRLINE_RESERVATION | 00:00:00 | 0 rows



```
INSERT INTO Airport VALUES ( 'Sy111' , 'Sydney' , 'Sydney' , 'Wales' )
INSERT INTO Airport VALUES ( 'Ch211' , 'Changi' , 'Changi' , 'Singapore' )
INSERT INTO Airport VALUES ( 'Ha311' , 'Haneda' , 'Ota' , 'Tokyo' )
INSERT INTO Airport VALUES ( 'Cd411' , 'CDG' , 'Loures' , 'France' )

INSERT INTO Land VALUES ('JB101' , 'Sy111' )
INSERT INTO Land VALUES ('MA101' , 'Ch211' )
INSERT INTO Land VALUES ('MB101' , 'Cd411' )
INSERT INTO Land VALUES ('JB104' , 'Cd411' )
INSERT INTO Land VALUES ('LS101' , 'Sy111' )
INSERT INTO Land VALUES ('MB102' , 'Sy111' )
INSERT INTO Land VALUES ('LS102' , 'Ch211' )
INSERT INTO Land VALUES ('MA103' , 'Cd411' )
INSERT INTO Land VALUES ('JB102' , 'Sy111' )
INSERT INTO Land VALUES ('MB103' , 'Sy111' )
INSERT INTO Land VALUES ('JB103' , 'Cd411' )
INSERT INTO Land VALUES ('LS103' , 'Cd411' )
INSERT INTO Land VALUES ('LS104' , 'Sy111' )
INSERT INTO Land VALUES ('MB104' , 'Ch211' )

INSERT INTO Flight VALUES ('AA111','Asiana Airline','Tin Tin')
INSERT INTO Flight VALUES ('AC111','Air China','Fred')
INSERT INTO Flight VALUES ('QA111','Qatar A','Gihan')
```

```
(1 row affected)

(1 row affected)
```

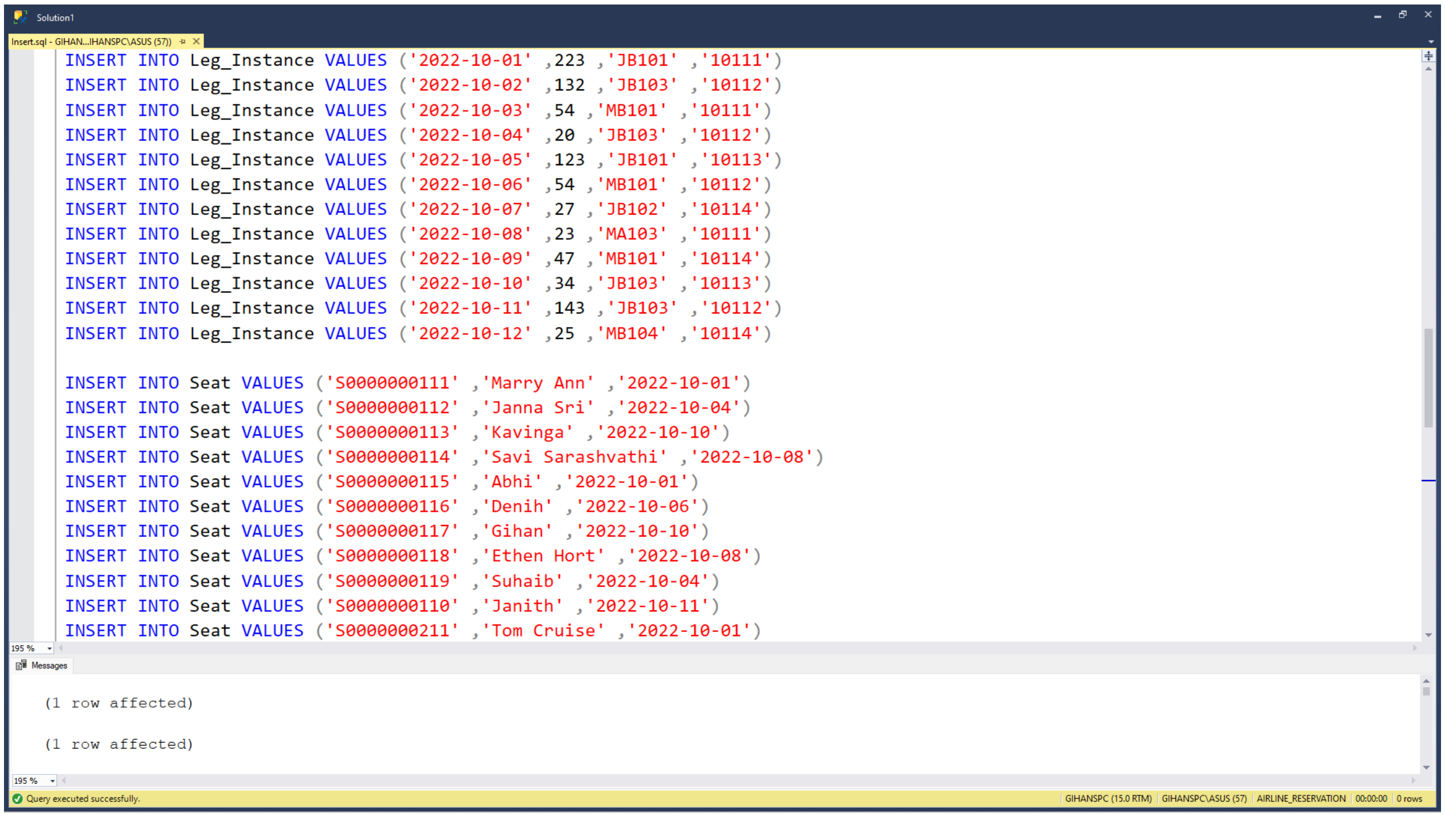
Query executed successfully. GIHANSPC (15.0 RTM) | GIHANSPC\ASUS (57) | AIRLINE_RESERVATION | 00:00:00 | 0 rows



```
INSERT INTO Flight VALUES ('AA111','Asiana Airline','Tin Tin')
INSERT INTO Flight VALUES ('AC111','Air China','Fred')
INSERT INTO Flight VALUES ('QA111','Qatar A','Gihan')

INSERT INTO Flight_Leg VALUES ('10111' ,'06:30' ,'07:10' ,'AA111' ,'Sy111' )
INSERT INTO Flight_Leg VALUES ('10112' ,'08:00' ,'08:40' ,'AC111' ,'Ch211' )
INSERT INTO Flight_Leg VALUES ('10113' ,'12:40' ,'13:20' ,'QA111' ,'Ch211' )
INSERT INTO Flight_Leg VALUES ('10114' ,'15:20' ,'16:00' ,'QA111' ,'Sy111' )
INSERT INTO Flight_Leg VALUES ('10115' ,'16:20' ,'16:20' ,'AC111' ,'Ch211' )

INSERT INTO Flight_Fare VALUES ('KL203' ,'less than 5 seats only' ,125000.00 ,'AC111')
INSERT INTO Flight_Fare VALUES ('CL210' ,'age should be 20+ ' ,350000.00 ,'AA111')
INSERT INTO Flight_Fare VALUES ('SL120' ,'hold between 3 to 7 days' ,252500.00 ,'QA111')
INSERT INTO Flight_Fare VALUES ('ID201' ,'age should be 20+ ' ,75000.00 ,'QA111')
INSERT INTO Flight_Fare VALUES ('KL321' ,'hold between 3 to 7 days' ,100000.00 ,'AA111')
INSERT INTO Flight_Fare VALUES ('KL212' ,'hold between 3 to 7 days' ,175000.00 ,'QA111')

INSERT INTO Leg_Instance VALUES ('2022-10-01' ,223 ,'JB101' ,'10111')
INSERT INTO Leg_Instance VALUES ('2022-10-02' ,132 ,'JB103' ,'10112')
INSERT INTO Leg_Instance VALUES ('2022-10-03' ,54 ,'MB101' ,'10111')
INSERT INTO Leg_Instance VALUES ('2022-10-04' ,20 ,'JB103' ,'10112')
INSERT INTO Leg_Instance VALUES ('2022-10-05' ,123 ,'JB101' ,'10113')
INSERT INTO Leg_Instance VALUES ('2022-10-06' ,54 ,'MB101' ,'10112')
INSERT INTO Leg_Instance VALUES ('2022-10-07' ,27 ,'JB102' ,'10114')
```

```
(1 row affected)

(1 row affected)
```

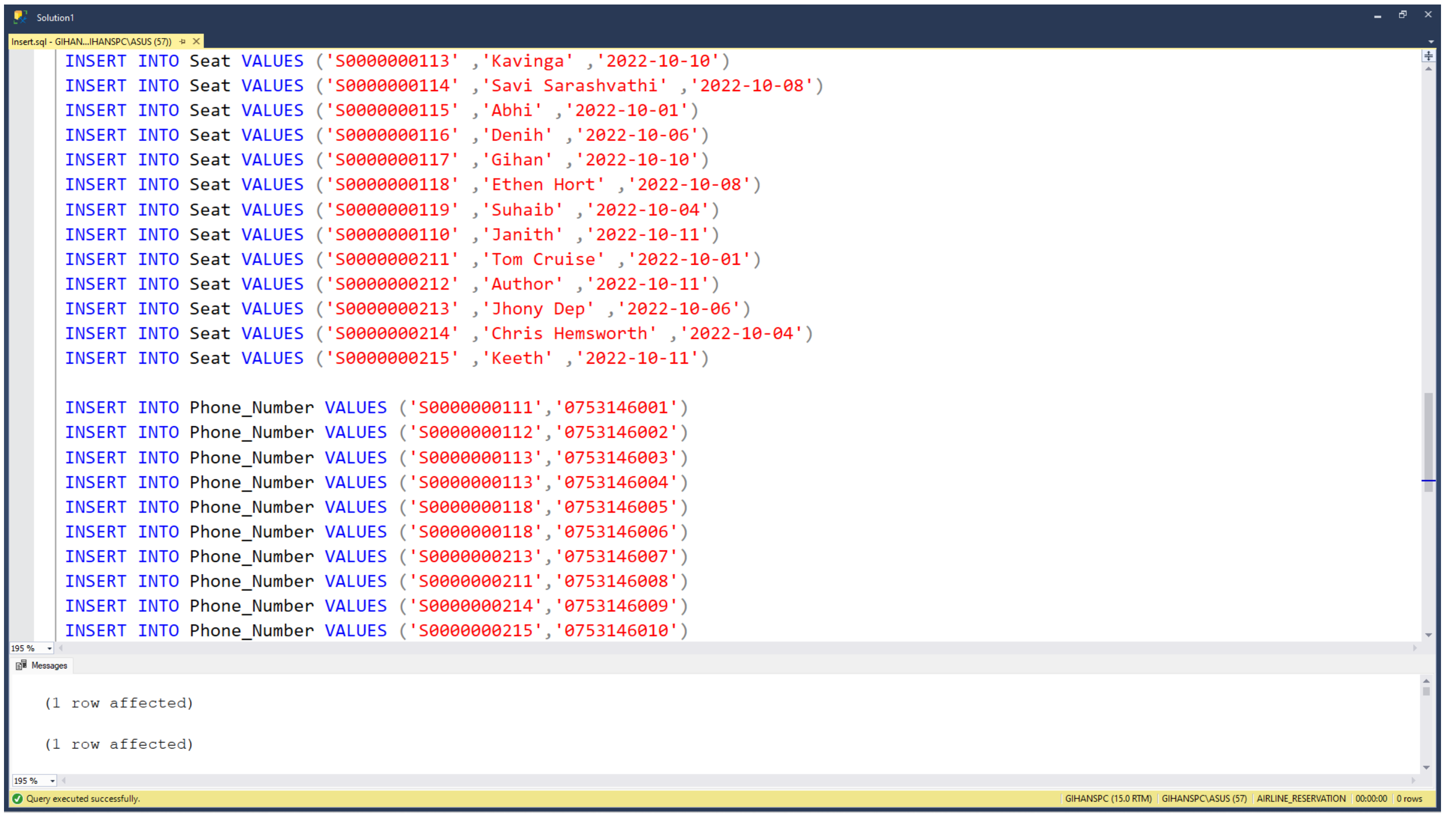
Query executed successfully. | GIHANSPC (15.0 RTM) | GIHANSPC\ASUS (57) | AIRLINE_RESERVATION | 00:00:00 | 0 rows



```
INSERT INTO Leg_Instance VALUES ('2022-10-01' ,223 ,'JB101' ,'10111')
INSERT INTO Leg_Instance VALUES ('2022-10-02' ,132 ,'JB103' ,'10112')
INSERT INTO Leg_Instance VALUES ('2022-10-03' ,54 ,'MB101' ,'10111')
INSERT INTO Leg_Instance VALUES ('2022-10-04' ,20 ,'JB103' ,'10112')
INSERT INTO Leg_Instance VALUES ('2022-10-05' ,123 ,'JB101' ,'10113')
INSERT INTO Leg_Instance VALUES ('2022-10-06' ,54 ,'MB101' ,'10112')
INSERT INTO Leg_Instance VALUES ('2022-10-07' ,27 ,'JB102' ,'10114')
INSERT INTO Leg_Instance VALUES ('2022-10-08' ,23 ,'MA103' ,'10111')
INSERT INTO Leg_Instance VALUES ('2022-10-09' ,47 ,'MB101' ,'10114')
INSERT INTO Leg_Instance VALUES ('2022-10-10' ,34 ,'JB103' ,'10113')
INSERT INTO Leg_Instance VALUES ('2022-10-11' ,143 ,'JB103' ,'10112')
INSERT INTO Leg_Instance VALUES ('2022-10-12' ,25 ,'MB104' ,'10114')

INSERT INTO Seat VALUES ('S0000000111' ,'Marry Ann' ,'2022-10-01')
INSERT INTO Seat VALUES ('S0000000112' ,'Janna Sri' ,'2022-10-04')
INSERT INTO Seat VALUES ('S0000000113' ,'Kavinga' ,'2022-10-10')
INSERT INTO Seat VALUES ('S0000000114' ,'Savi Sarashvathi' ,'2022-10-08')
INSERT INTO Seat VALUES ('S0000000115' ,'Abhi' ,'2022-10-01')
INSERT INTO Seat VALUES ('S0000000116' ,'Denih' ,'2022-10-06')
INSERT INTO Seat VALUES ('S0000000117' ,'Gihan' ,'2022-10-10')
INSERT INTO Seat VALUES ('S0000000118' ,'Ethen Hort' ,'2022-10-08')
INSERT INTO Seat VALUES ('S0000000119' ,'Suhaib' ,'2022-10-04')
INSERT INTO Seat VALUES ('S0000000110' ,'Janith' ,'2022-10-11')
INSERT INTO Seat VALUES ('S0000000211' ,'Tom Cruise' ,'2022-10-01')
```

```
(1 row affected)

(1 row affected)
```

Query executed successfully. | GIHANSPC (15.0 RTM) | GIHANSPC\ASUS (57) | AIRLINE_RESERVATION | 00:00:00 | 0 rows


Solution1

Insert.sql - GIHAN...IHANSPC\ASUS (57))

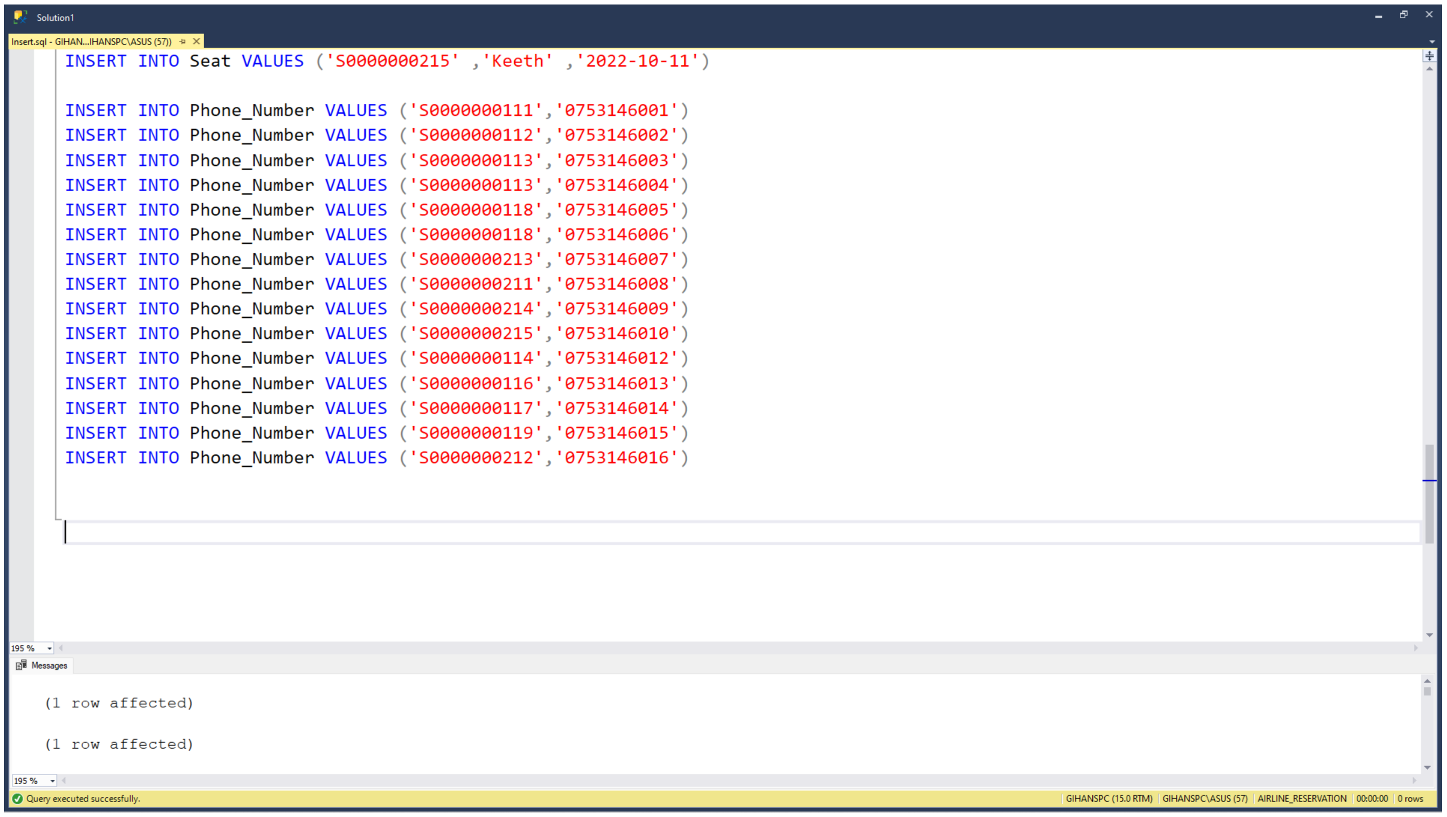
```
INSERT INTO Seat VALUES ('S0000000113' ,'Kavinga' ,'2022-10-10')
INSERT INTO Seat VALUES ('S0000000114' ,'Savi Sarashvathi' ,'2022-10-08')
INSERT INTO Seat VALUES ('S0000000115' ,'Abhi' ,'2022-10-01')
INSERT INTO Seat VALUES ('S0000000116' ,'Denih' ,'2022-10-06')
INSERT INTO Seat VALUES ('S0000000117' ,'Gihan' ,'2022-10-10')
INSERT INTO Seat VALUES ('S0000000118' ,'Ethen Hort' ,'2022-10-08')
INSERT INTO Seat VALUES ('S0000000119' ,'Suhaib' ,'2022-10-04')
INSERT INTO Seat VALUES ('S0000000110' ,'Janith' ,'2022-10-11')
INSERT INTO Seat VALUES ('S0000000211' ,'Tom Cruise' ,'2022-10-01')
INSERT INTO Seat VALUES ('S0000000212' ,'Author' ,'2022-10-11')
INSERT INTO Seat VALUES ('S0000000213' ,'Jhony Dep' ,'2022-10-06')
INSERT INTO Seat VALUES ('S0000000214' ,'Chris Hemsworth' ,'2022-10-04')
INSERT INTO Seat VALUES ('S0000000215' ,'Keeth' ,'2022-10-11')

INSERT INTO Phone_Number VALUES ('S0000000111','0753146001')
INSERT INTO Phone_Number VALUES ('S0000000112','0753146002')
INSERT INTO Phone_Number VALUES ('S0000000113','0753146003')
INSERT INTO Phone_Number VALUES ('S0000000113','0753146004')
INSERT INTO Phone_Number VALUES ('S0000000118','0753146005')
INSERT INTO Phone_Number VALUES ('S0000000118','0753146006')
INSERT INTO Phone_Number VALUES ('S0000000213','0753146007')
INSERT INTO Phone_Number VALUES ('S0000000211','0753146008')
INSERT INTO Phone_Number VALUES ('S0000000214','0753146009')
INSERT INTO Phone_Number VALUES ('S0000000215','0753146010')
```

195 %

Messages

```
(1 row affected)

(1 row affected)
```

Query executed successfully. GIHANSPC (15.0 RTM) | GIHANSPC\ASUS (57) | AIRLINE_RESERVATION | 00:00:00 | 0 rows


Solution1

Insert.sql - GIHAN...IHANSPC\ASUS (57))

```
INSERT INTO Seat VALUES ('S0000000215' ,'Keeth' ,'2022-10-11')

INSERT INTO Phone_Number VALUES ('S0000000111','0753146001')
INSERT INTO Phone_Number VALUES ('S0000000112','0753146002')
INSERT INTO Phone_Number VALUES ('S0000000113','0753146003')
INSERT INTO Phone_Number VALUES ('S0000000113','0753146004')
INSERT INTO Phone_Number VALUES ('S0000000118','0753146005')
INSERT INTO Phone_Number VALUES ('S0000000118','0753146006')
INSERT INTO Phone_Number VALUES ('S0000000213','0753146007')
INSERT INTO Phone_Number VALUES ('S0000000211','0753146008')
INSERT INTO Phone_Number VALUES ('S0000000214','0753146009')
INSERT INTO Phone_Number VALUES ('S0000000215','0753146010')
INSERT INTO Phone_Number VALUES ('S0000000114','0753146012')
INSERT INTO Phone_Number VALUES ('S0000000116','0753146013')
INSERT INTO Phone_Number VALUES ('S0000000117','0753146014')
INSERT INTO Phone_Number VALUES ('S0000000119','0753146015')
INSERT INTO Phone_Number VALUES ('S0000000212','0753146016')
```

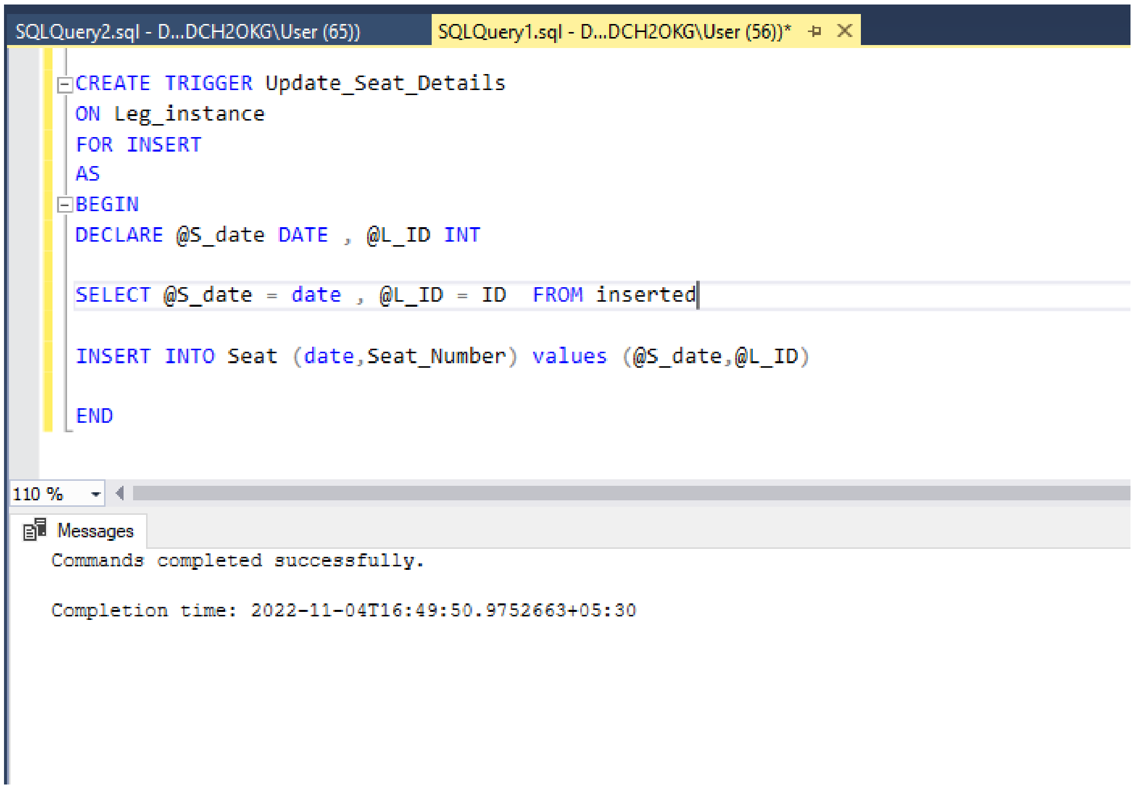
195 %

Messages

```
(1 row affected)

(1 row affected)
```

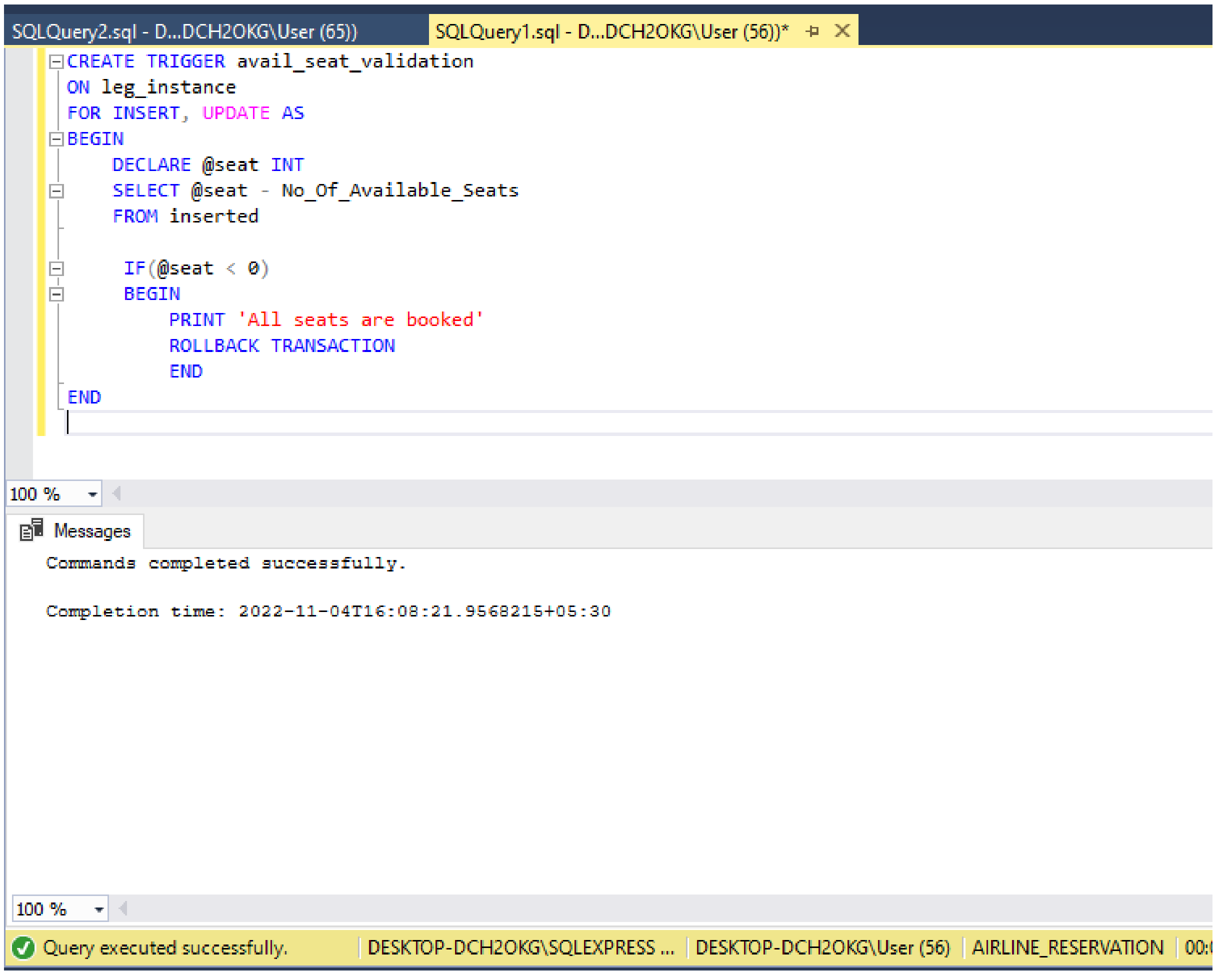
Query executed successfully. GIHANSPC (15.0 RTM) | GIHANSPC\ASUS (57) | AIRLINE_RESERVATION | 00:00:00 | 0 rows

Triggers


SQLQuery2.sql - D...DCH2OKG\User (65))
SQLQuery1.sql - D...DCH2OKG\User (56))*
110 %
Messages

```sql
CREATE TRIGGER Update_Seat_Details
ON Leg_instance
FOR INSERT
AS
BEGIN
DECLARE @S_date DATE , @L_ID INT

SELECT @S_date = date , @L_ID = ID  FROM inserted

INSERT INTO Seat (date,Seat_Number) values (@S_date,@L_ID)

END
```

```
Commands completed successfully.

Completion time: 2022-11-04T16:49:50.9752663+05:30
```

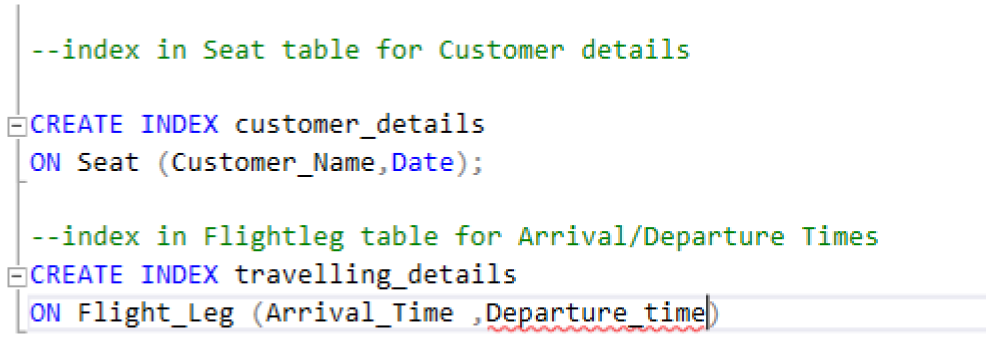

SQLQuery2.sql - D...DCH2OKG\User (65))
SQLQuery1.sql - D...DCH2OKG\User (56))*
100 %
Messages
100 %

```sql
CREATE TRIGGER avail_seat_validation
ON leg_instance
FOR INSERT, UPDATE AS
BEGIN
    DECLARE @seat INT
    SELECT @seat - No_Of_Available_Seats
    FROM inserted

     IF(@seat < 0)
     BEGIN
         PRINT 'All seats are booked'
         ROLLBACK TRANSACTION
         END
END
```

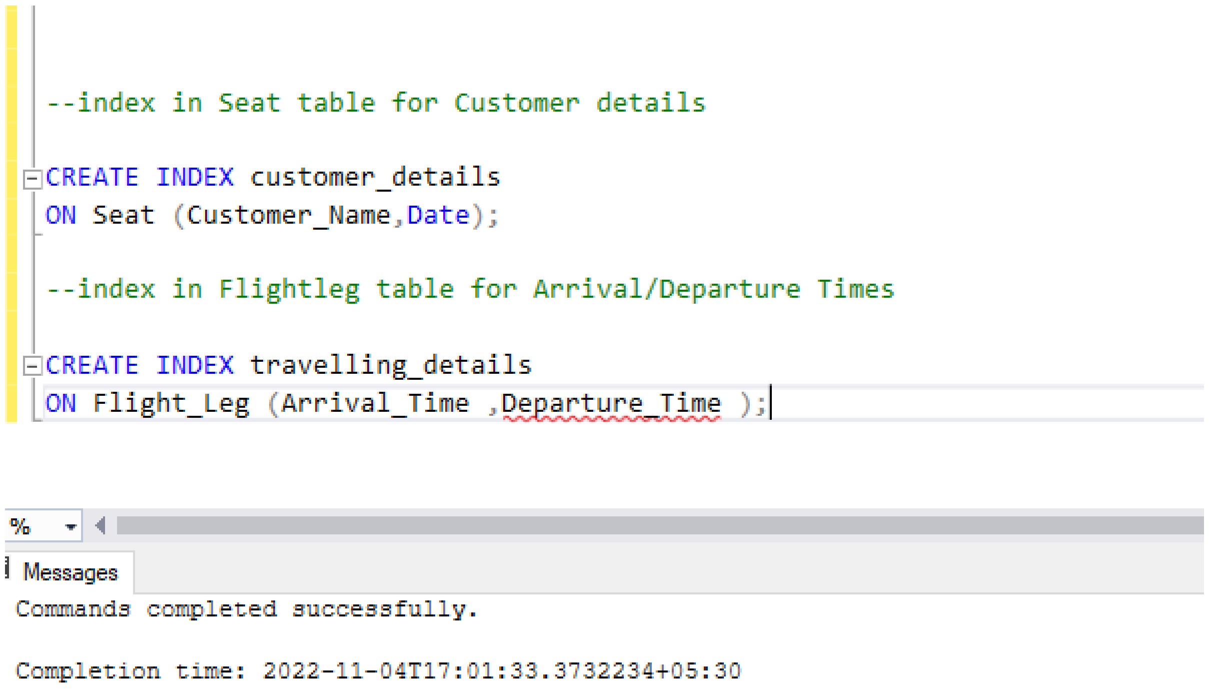
```
Commands completed successfully.

Completion time: 2022-11-04T16:08:21.9568215+05:30
```

Query executed successfully. | DESKTOP-DCH2OKG\SQLEXPRESS ... | DESKTOP-DCH2OKG\User (56) | AIRLINE_RESERVATION | 00:0

Indexes



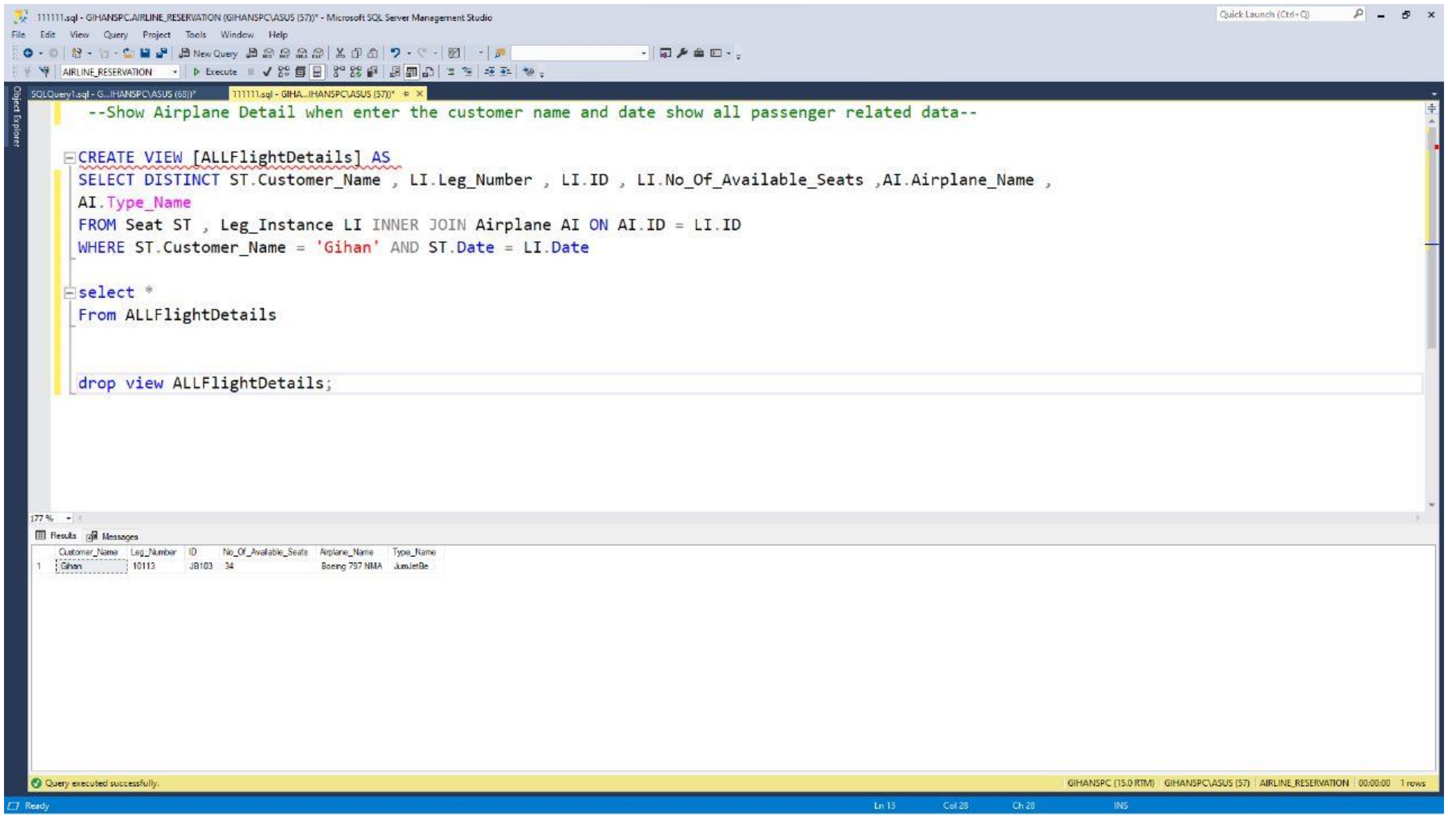
```
--index in Seat table for Customer details

CREATE INDEX customer_details
ON Seat (Customer_Name,Date);

--index in Flightleg table for Arrival/Departure Times
CREATE INDEX travelling_details
ON Flight_Leg (Arrival_Time ,Departure_time)
```



```
--index in Seat table for Customer details

CREATE INDEX customer_details
ON Seat (Customer_Name,Date);

--index in Flightleg table for Arrival/Departure Times

CREATE INDEX travelling_details
ON Flight_Leg (Arrival_Time ,Departure_Time );
```

```
Commands completed successfully.

Completion time: 2022-11-04T17:01:33.3732234+05:30
```

Views

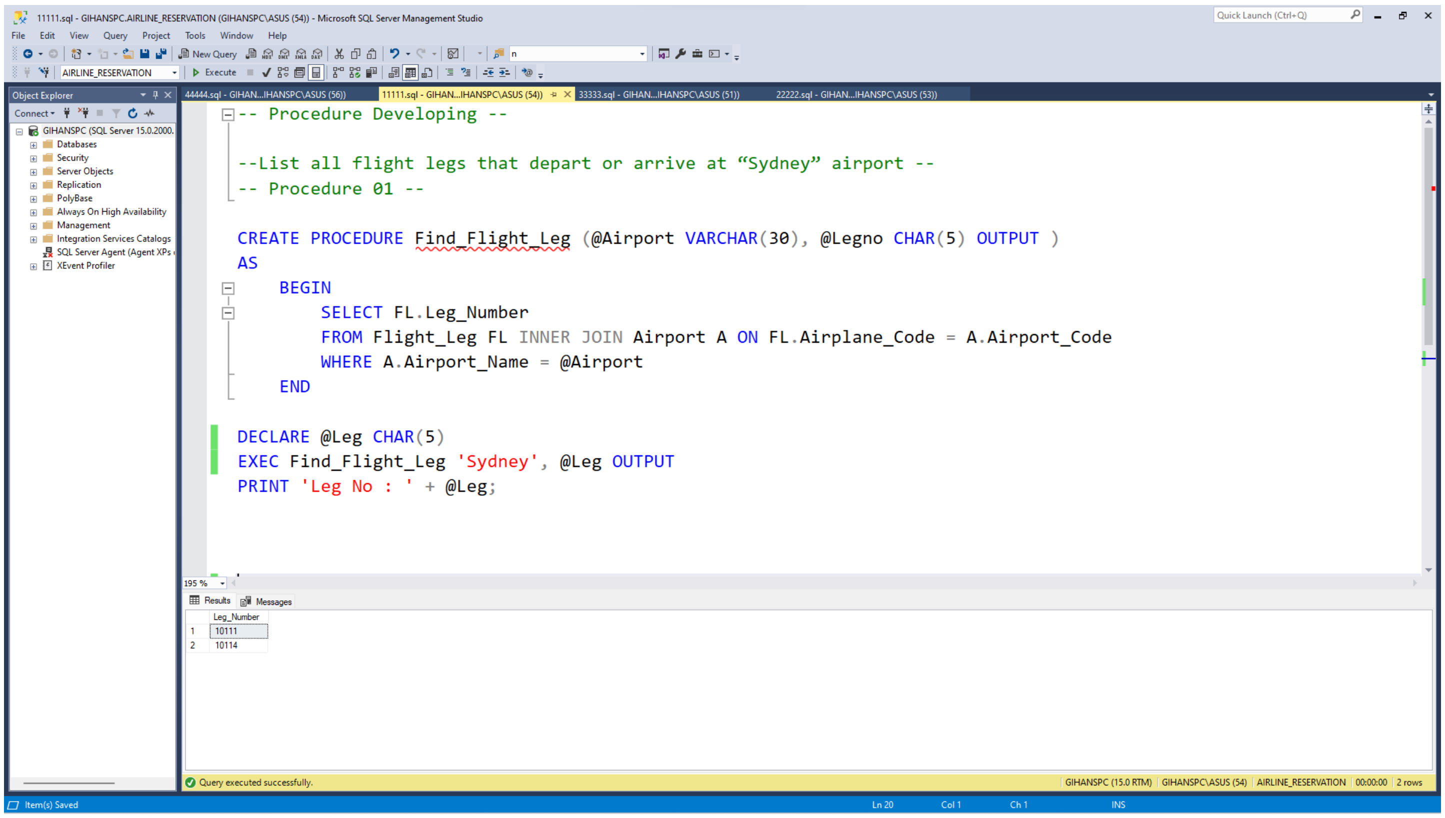


```sql
--Show Airplane Detail when enter the customer name and date show all passenger related data--

CREATE VIEW [ALLFlightDetails] AS
SELECT DISTINCT ST.Customer_Name , LI.Leg_Number , LI.ID , LI.No_Of_Available_Seats ,AI.Airplane_Name ,
AI.Type_Name
FROM Seat ST , Leg_Instance LI INNER JOIN Airplane AI ON AI.ID = LI.ID
WHERE ST.Customer_Name = 'Gihan' AND ST.Date = LI.Date

select *
From ALLFlightDetails


drop view ALLFlightDetails;
```

	Customer_Name	Leg_Number	ID	No_Of_Available_Seats	Airplane_Name	Type_Name
1	Gihan	10113	JB103	34	Boeing 797 NMA	JumJetBe

Query executed successfully.

1. Procedures

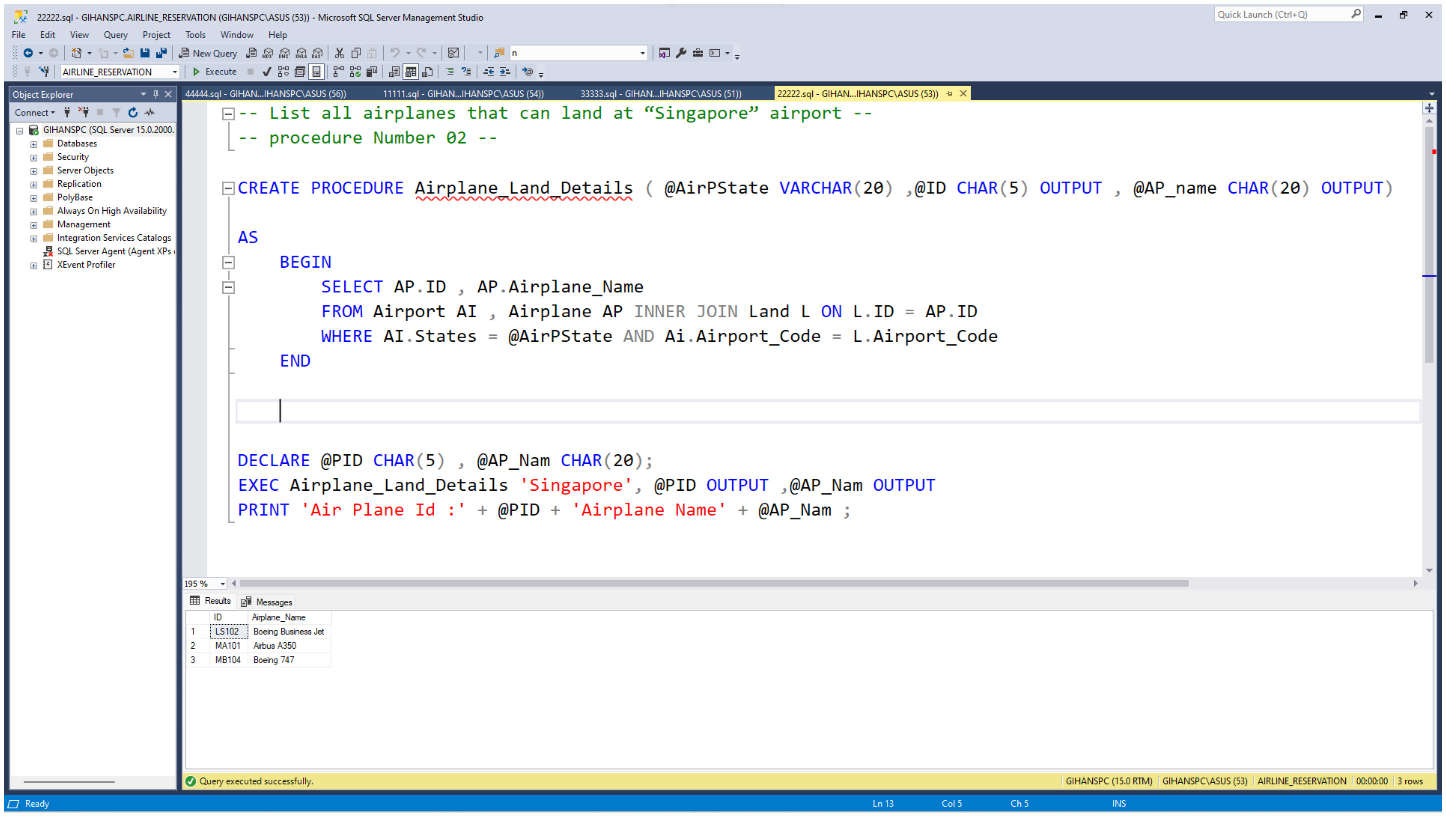


11111.sql - GIHANSPC.AIRLINE_RESERVATION (GIHANSPC\ASUS (54)) - Microsoft SQL Server Management Studio

```sql
-- Procedure Developing --

--List all flight legs that depart or arrive at “Sydney” airport --
-- Procedure 01 --

CREATE PROCEDURE Find_Flight_Leg (@Airport VARCHAR(30), @Legno CHAR(5) OUTPUT )
AS
    BEGIN
        SELECT FL.Leg_Number
        FROM Flight_Leg FL INNER JOIN Airport A ON FL.Airplane_Code = A.Airport_Code
        WHERE A.Airport_Name = @Airport
    END

DECLARE @Leg CHAR(5)
EXEC Find_Flight_Leg 'Sydney', @Leg OUTPUT
PRINT 'Leg No : ' + @Leg;
```

	Leg_Number
1	10111
2	10114

Query executed successfully.

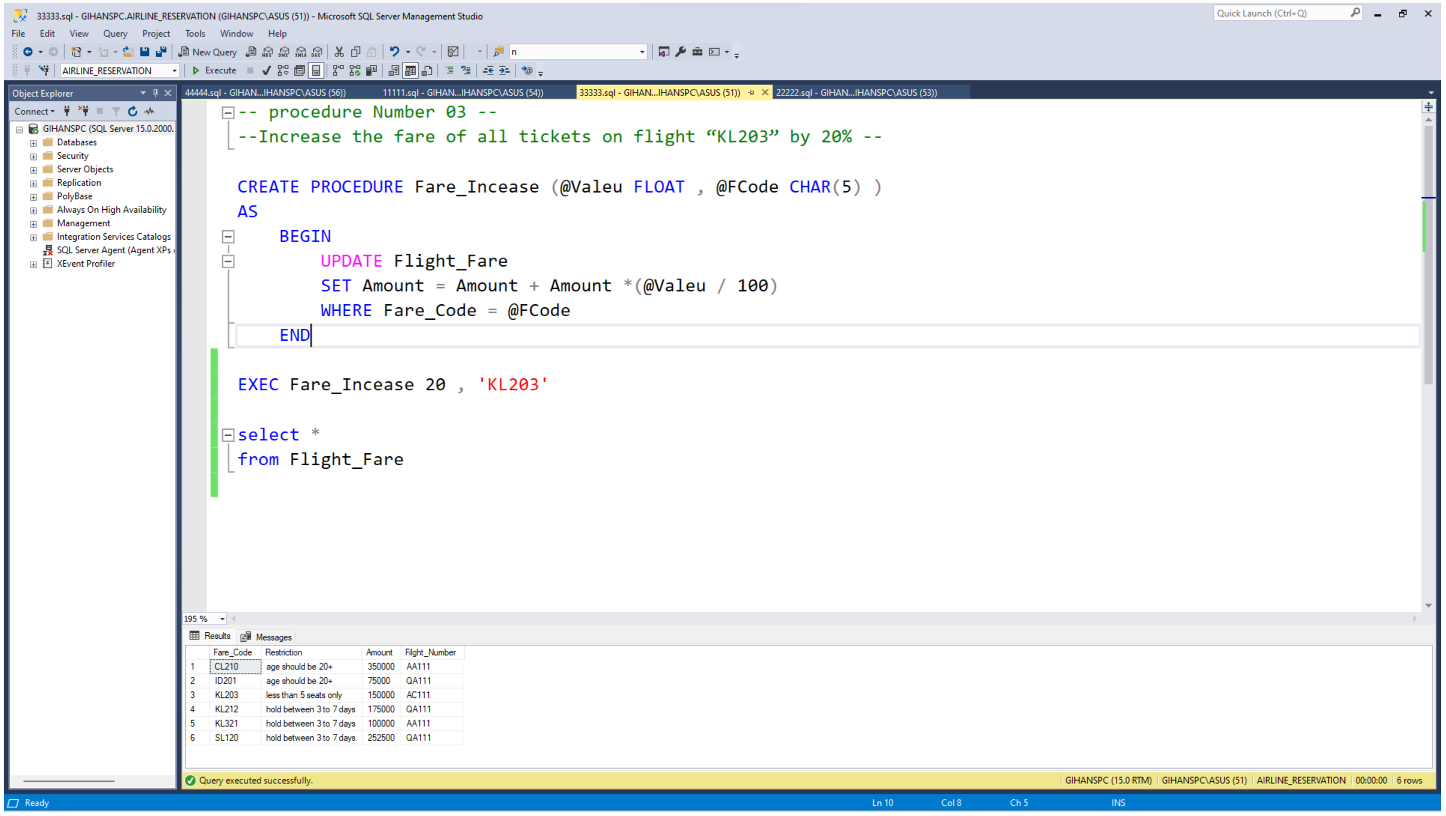


22222.sql - GIHANSPC.AIRLINE_RESERVATION (GIHANSPC\ASUS (53)) - Microsoft SQL Server Management Studio

```sql
-- List all airplanes that can land at “Singapore” airport --
-- procedure Number 02 --

CREATE PROCEDURE Airplane_Land_Details ( @AirPState VARCHAR(20) ,@ID CHAR(5) OUTPUT , @AP_name CHAR(20) OUTPUT)

AS
    BEGIN
        SELECT AP.ID , AP.Airplane_Name
        FROM Airport AI , Airplane AP INNER JOIN Land L ON L.ID = AP.ID
        WHERE AI.States = @AirPState AND Ai.Airport_Code = L.Airport_Code
    END


DECLARE @PID CHAR(5) , @AP_Nam CHAR(20);
EXEC Airplane_Land_Details 'Singapore', @PID OUTPUT ,@AP_Nam OUTPUT
PRINT 'Air Plane Id :' + @PID + 'Airplane Name' + @AP_Nam ;
```

	ID	Airplane_Name
1	LS102	Boeing Business Jet
2	MA101	Airbus A350
3	MB104	Boeing 747

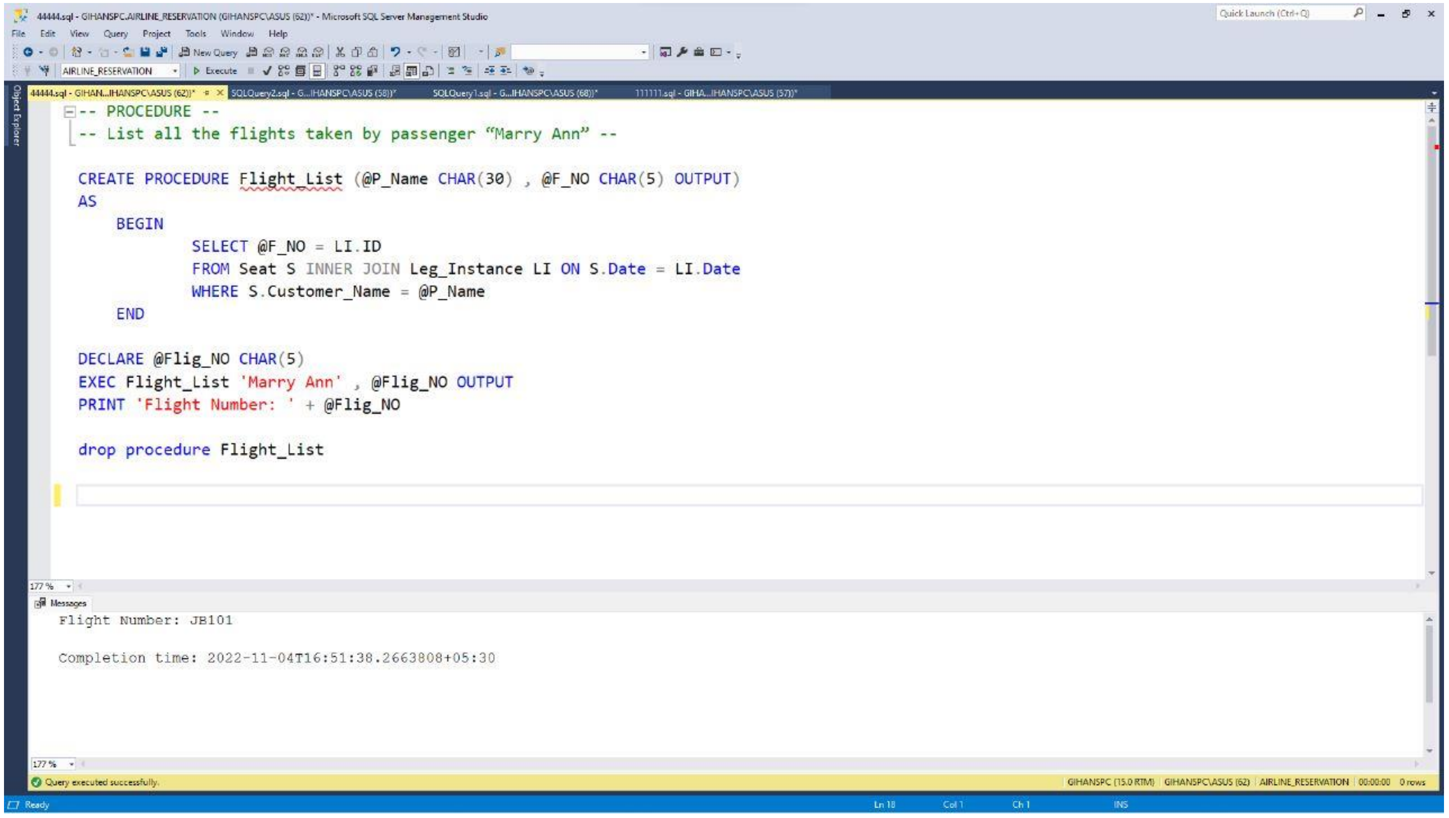
Query executed successfully.



```sql
-- procedure Number 03 --
--Increase the fare of all tickets on flight “KL203” by 20% --

CREATE PROCEDURE Fare_Incease (@Valeu FLOAT , @FCode CHAR(5) )
AS
    BEGIN
        UPDATE Flight_Fare
        SET Amount = Amount + Amount *(@Valeu / 100)
        WHERE Fare_Code = @FCode
    END

EXEC Fare_Incease 20 , 'KL203'

select *
from Flight_Fare
```

	Fare_Code	Restriction	Amount	Filght_Number
1	CL210	age should be 20+	350000	AA111
2	ID201	age should be 20+	75000	QA111
3	KL203	less than 5 seats only	150000	AC111
4	KL212	hold between 3 to 7 days	175000	QA111
5	KL321	hold between 3 to 7 days	100000	AA111
6	SL120	hold between 3 to 7 days	252500	QA111

Query executed successfully.



```sql
-- PROCEDURE --
-- List all the flights taken by passenger “Marry Ann” --

CREATE PROCEDURE Flight_List (@P_Name CHAR(30) , @F_NO CHAR(5) OUTPUT)
AS
    BEGIN
            SELECT @F_NO = LI.ID
            FROM Seat S INNER JOIN Leg_Instance LI ON S.Date = LI.Date
            WHERE S.Customer_Name = @P_Name
    END

DECLARE @Flig_NO CHAR(5)
EXEC Flight_List 'Marry Ann' , @Flig_NO OUTPUT
PRINT 'Flight Number: ' + @Flig_NO

drop procedure Flight_List
```

```
Flight Number: JB101

Completion time: 2022-11-04T16:51:38.2663808+05:30
```

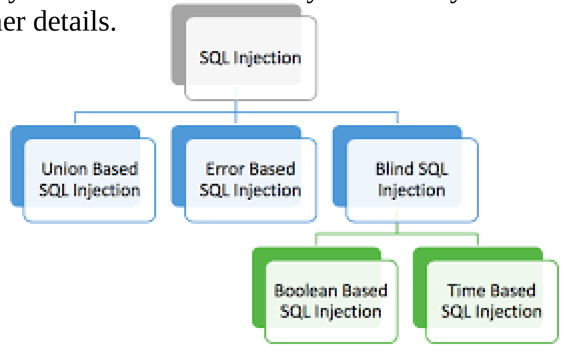
Query executed successfully.

Part 2

Two database vulnerabilities

1. SQL Injection

SQL (Structured Query Language) is a common attack vector that uses malicious SQL code for backend database manipulation to access information that was not intended to be displayed. This information may include any number of items, including sensitive company data, user lists, or private customer details.


SQL Injection
Union Based SQL Injection
Error Based SQL Injection
Blind SQL Injection
Boolean Based SQL Injection
Time Based SQL Injection

Types of SQL Injection

1. Error-Based SQL Injection
2. Blind-Based SQL Injection
3. Union-Based SQL Injection

1. Error-based SQL injection

The Error based technique, when an attacker tries to insert a malicious query in input fields and gets some error which is regarding SQL syntax or database.
How to detect error-based SQL injection
In the input field parameter add a single quote (‘), and double quote (“) as well as can try some SQL keywords like ‘AND’, and ‘OR’ for the test.

2. Blind-based SQL injection

Attacking through a normal SQLI application gives a normal error message saying that the syntax of the SQL query is incorrect. Blind SQLI is a type of SQLI technique that works on injecting SQLI query to the database blindly and identifying the output based on the change in the behavior of the response.

Blind SQLI is not like ERROR based in which the user inserts some SQL queries against the database where the user gets a specified error message. The attacker tries to get information by asking the database true or false query. So based on the prediction we need to define the output.

It has 2 types:

a. Boolean-based SQL Injection
b. Time-based SQL Injection

a. Boolean-based SQL Injection

Boolean-based SQL injection is one in which the attacker is sending an SQL query to the database based on true and false conditions and according to that response is getting changed. Helpful to find database name character by character.

b. Time-based SQL Injection

Time-based SQI in which attackers insert SQL query causing database pause for a specified amount of time and then returning the results (just delaying the output). This is helpful when the attacker does not have any kind of answer (error/output) from the application because the input validation has been sanitized.

3. Union-based SQL injection

Union-based SQL injection involves the use of the UNION operator that combines the results of multiple SELECT statements to fetch data from multiple tables as a single result set. The malicious UNION operator query can be sent to the database via the website URL or the user input field.

Impact of SQL Injection

Once an attacker realizes that a system is vulnerable to SQL Injection, he is able to inject SQL Query / Commands through an input form field. This is equivalent to handing the attacker your database and allowing him to execute any SQL command including DROP TABLE to the database!

An attacker may execute arbitrary SQL statements on the vulnerable system. This may compromise the integrity of your database and/or expose sensitive information. Depending on the back-end database in use, SQL injection vulnerabilities lead to varying levels of data/system access for the attacker. It may be possible to manipulate existing queries, to UNION (used to select related information from two tables) arbitrary data, use sub-selects, or append additional queries.

In some cases, it may be possible to read in or write out to files, or to execute shell commands on the underlying operating system. Certain SQL Servers such as Microsoft SQL Server contain stored and extended procedures (database server functions). If an attacker can obtain access to these procedures, unfortunately, the impact of SQL Injection is only uncovered when the theft is discovered. Data is being unwittingly stolen through various hack attacks all the time. The more expert hackers rarely get caught.

How to Mitigate SQL Injection

SQL injection is a very dangerous attack that steals your data, modifies causing the attacker to view unauthorized user lists, and delete entire tables.

So, it is necessary to prevent this from happening. There are several techniques that a developer can implement in code this might help to reduce taking advantage of SQLI ad performing harmful tasks.

1. **Stored procedure:** In this SQL code a stored procedure is defined and stored in the database and then called from the application when the user inserts input. However, this is not always safe.
2. **Whitelist Input Validation:** Avoid external malicious input and only allow accepted input. Whitelist validation checks an external input against a set of known (approved input). This is very much useful because the application knows what to allow and reject.
3. **Least Privilege:** Limit the privilege assigned to every user that is accessing the application. Do not assign database administrator or admin-type access rights to your application accounts.

Countermeasures to overcome SQL Injection

Preventing SQL injection attacks is a complex and rigorous process since prevention techniques vary according to the programming language used, the SQL database engine, and the SQL injection subtype. This section explores the tools and best practices to prevent SQL Injection vulnerabilities.

Regular Scanning

Attackers inject malicious input through vulnerabilities they discover in the system code. Therefore, security teams should undertake a SQL injection scanner. Using the right tools is relevant to find any possible SQLi vulnerabilities before attackers can take advantage of them.

Training & Awareness

everyone involved in developing and managing the application should understand the risk and impacts of SQL injections. Training should also be extended to users to understand why it is important only to include valid inputs when prompted.

Filter user input

A database administrator should never trust user input. Internal and public user inputs should all be filtered and validated before being exposed to the database server.

Use Whitelist based filters

Attackers will always develop clever methods to circumvent blacklists. Whitelisting prevents attacks using a list that only allows certain users to access the protected system. In addition, malicious payloads deployed by SQLi injections cannot execute when they don't exist in the whitelist.

Use updated web technologies

Software updates often include patches for discovered vulnerabilities. Hackers typically rely on these vulnerabilities to deploy malicious payloads. Using the latest patched versions of development environments and frameworks will meet compliance standards and keep the web application safe from exploits as most software organizations try to stay ahead of hackers.

2. DATABASE BACKUP

Database threats might originate both within and externally. And a lot of the time, internal dangers are given more consideration by businesses than external ones. The loyalty of a company's employees is never completely predictable by business owners. The Dark Web is a market where data may be stolen and sold by almost anybody with uncontrolled access. Typically, when individuals think about database protection, they focus on the primary database they want to keep secure and overlook database backups, whose security demands the same level of seriousness and attention. We move on to the point that follows from this one.

The backups are divided into two types

1. Physical backups
2. Logical backups

- Physical backups
  - The physical backups of your databases' data files, control files, archived redo logs, and log files are known as physical backups.
  - It is a duplicate of the database information files that also are maintained in another location, such as a drive or offline storage like magnetic tape.
  - Physical backups are the foundation of the recovery mechanism in the database.
  - Physical backup provides minute details about the transaction and modification to the database

- Logical backup
  - The logical data in a logical backup is taken from a database.
  - It also contains a recovery of logical data, such as views, functions, procedures, and tables.
  - In many cases, it is a helpful addition to physical backups, but as logical backups only include structural data, they are insufficient as a means of protecting against data loss on their own.

Importance Of Backups

- A major data crash can be prevented by planning and testing backups against failures of the media, operating system, software, and any other sort of failure.
- Data from physical storage is extracted using a physical backup (usually from disc to tape). One instance of physical backup is using the system.
- Logical backup obtains information from the database and saves it as a binary file using SQL.
- To recover the database objects, a logical backup is used. The database administrator (DBA) can thus back up and restore a portion of the database's objects according to the logical backup tools.
- It affects how quickly and well a person recovers.

Methods of Backup techniques

There are several ways to back up a database, including:

- Full Backup

  Using this method requires a lengthy period since the complete database, including the data and transaction records, must be duplicated.

- Transaction Log

  This limits the storage of the backup to the transaction logs.
  To make the backup file as small as possible, the previous transaction log contents are deleted after establishing a new backup record.

- Differential Backup

  It preserves both the data and the transaction records, which makes it akin to full backup. However, the backup only contains the data that has changed since the last complete backup. Therefore, differential backup results in smaller files.

WHY Plan Backups?

Having a backup and recovery plan is crucial since a database might fail for a variety of reasons. To name a few of them:

- An enormous amount of data and transactions are included in databases.
- In the case of a system malfunction or calamity, recovering the database is extremely difficult.

Impact of Database backups

- ➢ When you back up data to third-party online servers if you manage valuable and confidential, private data such as medical reports, etc. it’s not very ethical because you are handing over your data to another party.
- ➢ When you choose offline methods such as portable hard drives, they can be easily stolen by someone.
- ➢ When you backup and restore the database you need technical assistance, and you have to update your backups with your latest data often get a better advantage of backing up but it will cost more time and money

How to mitigate Database backups

- Include backup in your security strategy.

  Incorporate backup-related systems into your security policy, as necessary. Almost all security regulations, including those governing physical security, access control, system monitoring, virus protection, and notably encryption, immediately apply to data backups.

- Include a backup system in your DR strategy

  As part of your incident response and disaster recovery strategies, mention your data backup methods. In cases like a ransomware epidemic, employee break-ins, or natural disasters like floods or hurricanes, data backups may be hacked, corrupted, or even destroyed. If this happens, solid backups may suffer, so you need to have a strategy in place for what you'll do if it does.

- Limit access rights to data backups

  Only let people who need to be involved in the backup process for business purposes access permissions to the backup system. Both the actual backup files and the backup program are subject to this. System access to backups is provided by both local networks and cloud-based technologies, so don't ignore them.

- Consider different database backups

  Keep your backups offsite, or at the very least in another building. Your data center and your backups might be destroyed all at once by a natural disaster, fire, or another uncommon but significant occurrence.

- Limit physical access to data backups

  Make sure that access is properly managed in those locations regardless of how you decide to store your backups: on backup servers, NAS, or even external discs or tapes. Treat your backups the same way you would any other important devices. SOC audit reports, reports from independent security assessments, or your audits may allow you to verify this.

- Ensure your network is secure

  Regardless of how you choose to store your backups—on backup servers, NAS, or even external discs or tapes—make sure that access is adequately managed in those locations. The same care should be taken with your backups as with any other vital equipment. You might confirm this by looking at SOC audit reports, reports from independent security assessments, or your audits.

Countermeasures to overcome Database backups

Making backups of exclusive databases at predetermined intervals is a recommended practice. Surprisingly, database backup files are frequently not even remotely secured against attack. Due to database backup leaks, there are countless security breaches occurring.

Countermeasures :

- Backups and databases should both be encrypted. Database production and backup copies may be securely stored when data is encrypted. The best method for doing it is Data Sunrise Data Encryption.
- Examine the backups and the database together. This allows you to identify everyone who has attempted to access sensitive information.