Online Airline Reservation System

Part 1

Assumptions

1. There are only one-way round trips
2. The details of the cost of the ticket are already known to the customer
3. On a particular date one passenger can only book one ticket
4. There can be only one airport in the same country and the same state
5. Same Airline cannot have the same scheduled dates
6. There are no multivalued and composite Attributes

About inserted data on the "Sheduled_Dates" column In the Flight table

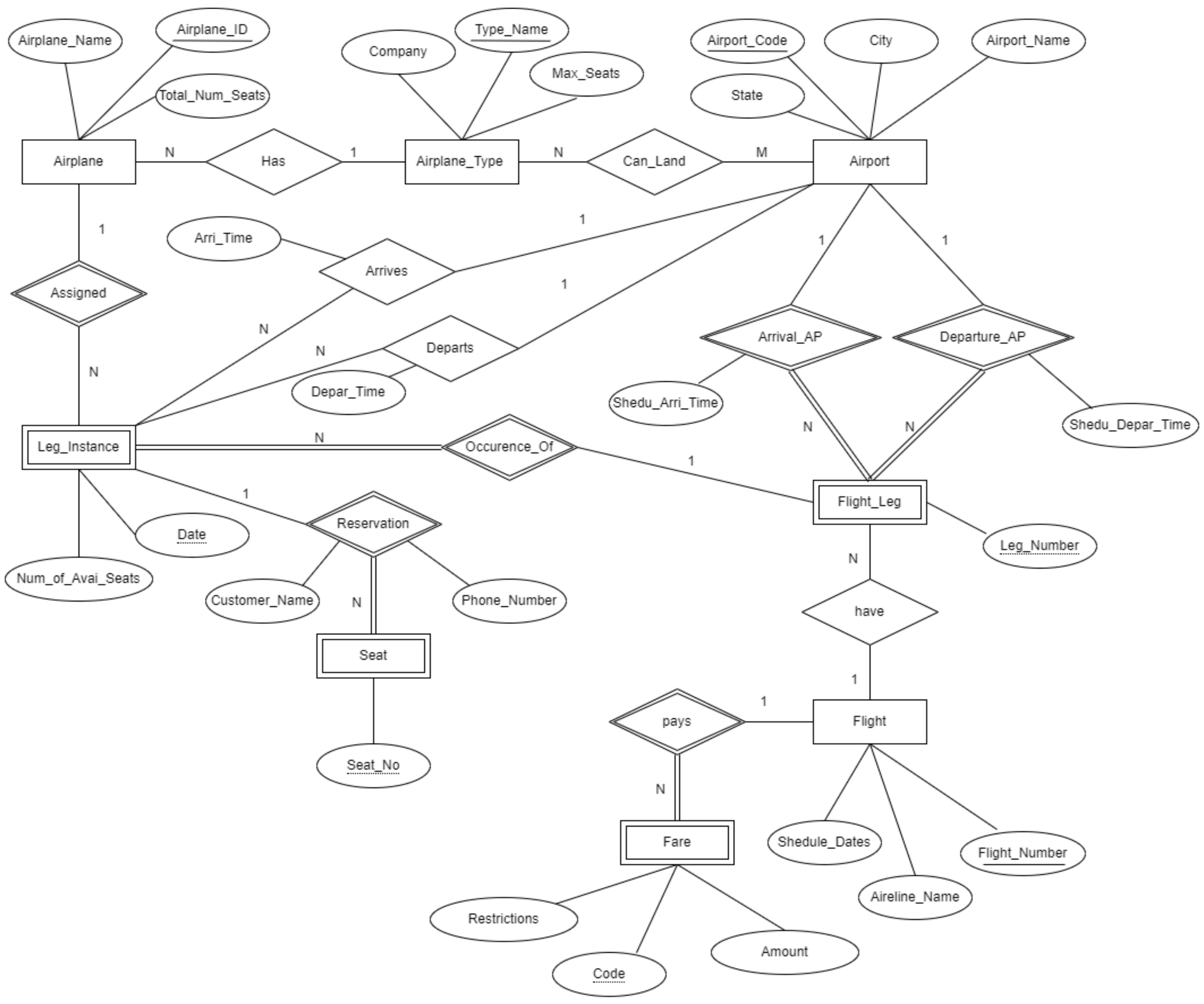
X8 means flight scheduled for every day (WD+WE)

X7 means flight scheduled for every day except Sunday

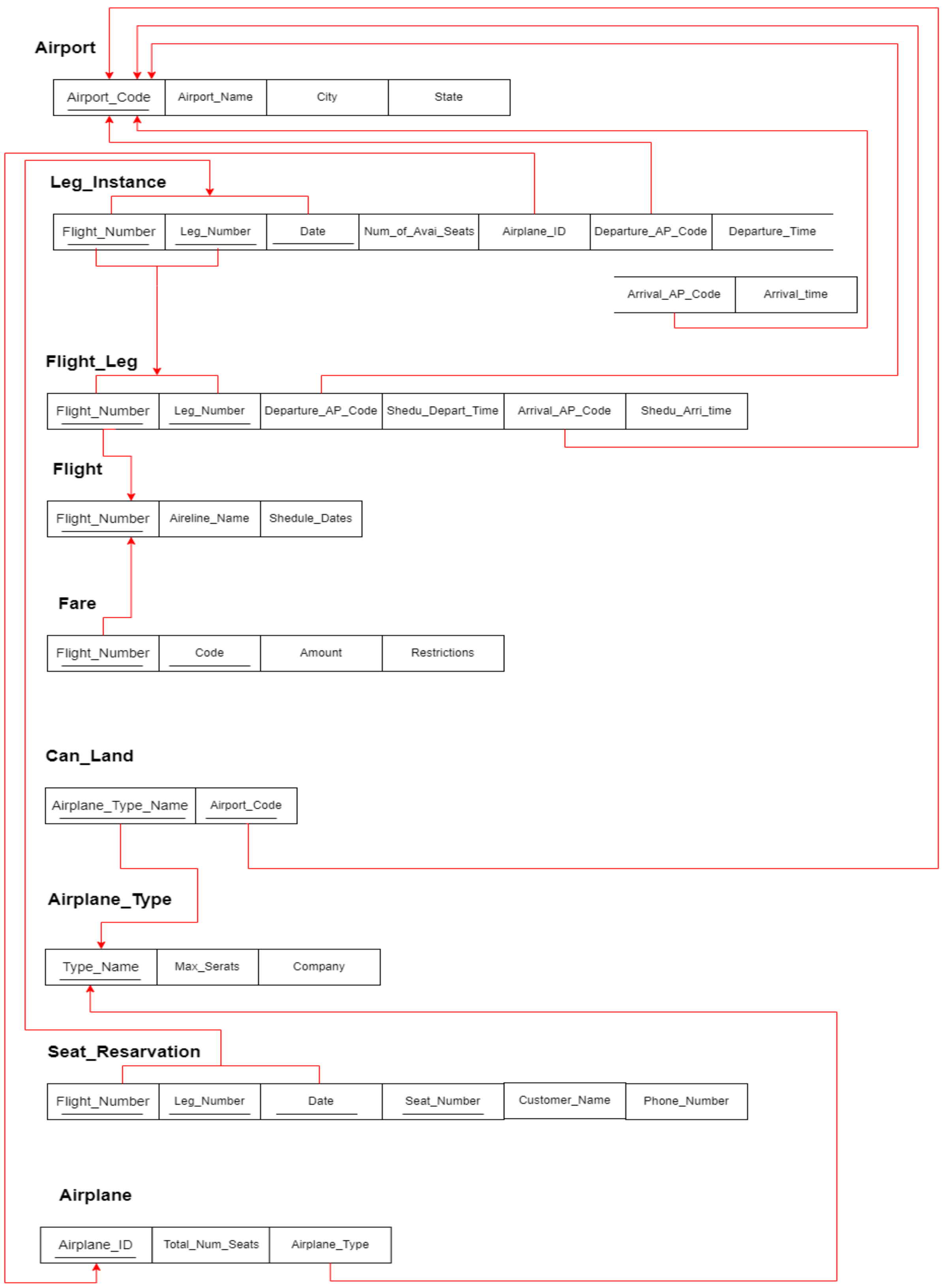
X1 means flight scheduled for every day except Monday

And So on

Entity Relationship Diagram



Normalized Logical Model



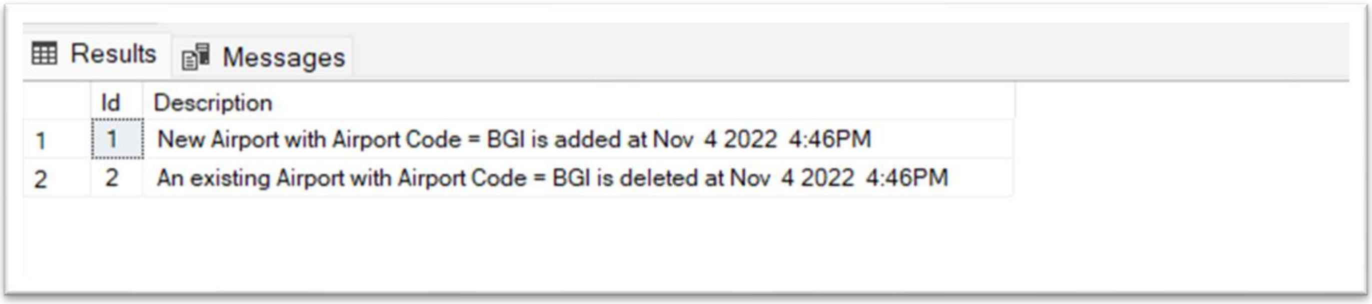
Functions

```
--functions 1

CREATE FUNCTION AirplaneCount(@Airplane_company VARCHAR(20))
RETURNS integer
AS
BEGIN
    DECLARE @planeCount integer
    SELECT @planeCount = count(*)
    FROM Airplane_Type A
    WHERE A.Company = @Airplane_company
    RETURN @planeCount
END

Declare @count integer
Exec @count = AirplaneCount 'Boing'
Print @count

--functions 2

CREATE FUNCTION MaxSeats(@Airline VARCHAR(20))
RETURNS integer
AS
BEGIN
    DECLARE @SeatCount integer
    SELECT @SeatCount = max(Maximum_Seats)
    FROM Airplane_Type
    WHERE Company = @Airline
    RETURN @SeatCount
END

Declare @Seatcount integer
Exec @Seatcount = MaxSeats'Airbus'
Print @Seatcount
```

Triggers

```
--trigger
CREATE TABLE Airport_Audit
(
Id int IDENTITY,
Description text
)



CREATE TRIGGER Insert_Airport
ON Airport
FOR INSERT
AS
BEGIN
  Declare @Id varchar(20)
  SELECT @Id = Airport_Code from inserted
  INSERT INTO Airport_Audit
  VALUES ('New Airport with Airport Code = ' + CAST(@Id AS VARCHAR(20)) + ' is added
at ' + CAST(Getdate() AS VARCHAR(22)))
END
```

This trigger is created to record newly added airports, added time, and date

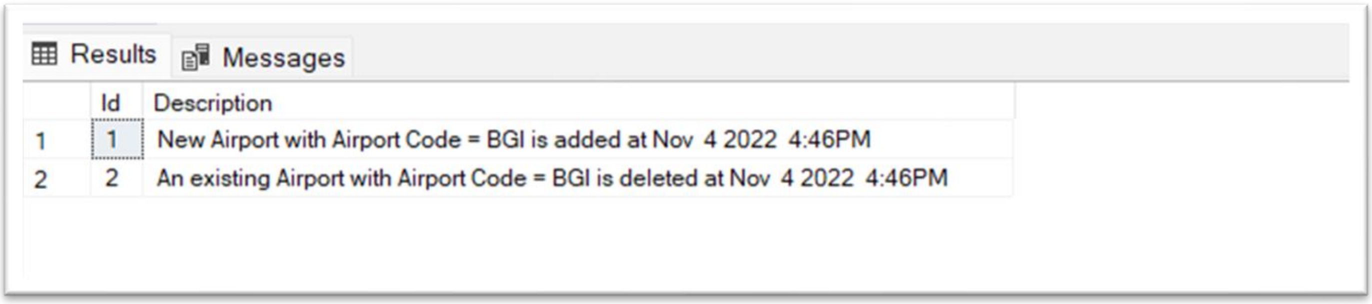


Results Messages

	Id	Description
1	1	New Airport with Airport Code = BGI is added at Nov 4 2022 4:46PM
2	2	An existing Airport with Airport Code = BGI is deleted at Nov 4 2022 4:46PM

```
--delete trigger

CREATE TRIGGER Delete_Airport
ON Airport
FOR DELETE
AS
BEGIN
  Declare @Id varchar(20)
  SELECT @Id = Airport_Code from deleted
  INSERT INTO Airport_Audit
  VALUES ('An existing Airport with Airport Code = ' + CAST(@Id AS VARCHAR(20)) + ' is
deleted at ' + CAST(Getdate() AS VARCHAR(22)))
```

```
END

This trigger is created to record deleted airports, deleted time, and date
```

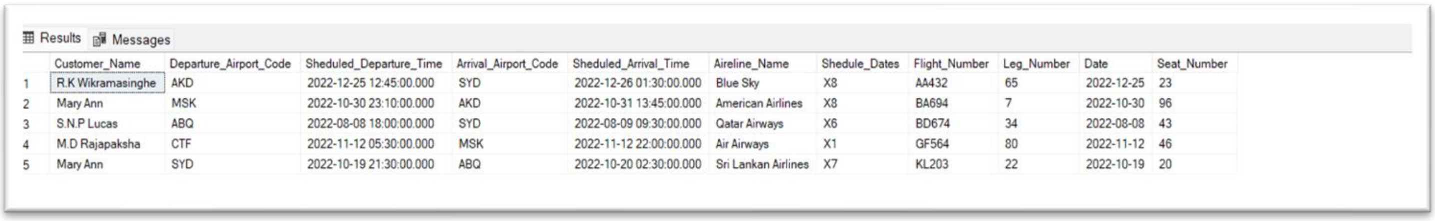


```
   Id  Description
1  1   New Airport with Airport Code = BGI is added at Nov  4 2022  4:46PM
2  2   An existing Airport with Airport Code = BGI is deleted at Nov  4 2022  4:46PM
```

Views

1. Passenger (Who has paid the ticket price)

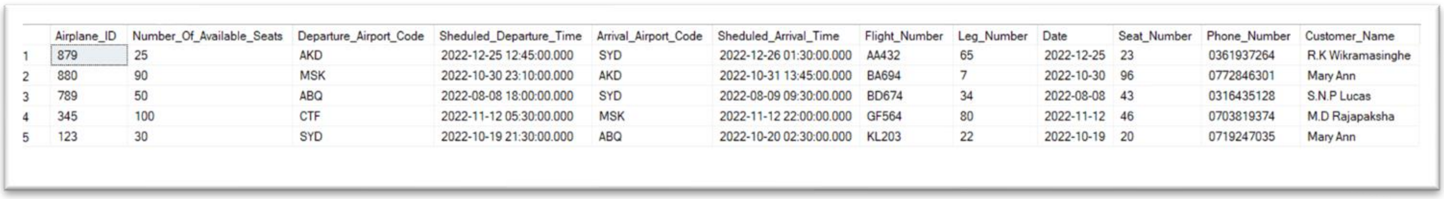
```sql
CREATE VIEW [GET FLIGHT DETAILS] AS
SELECT SR.Customer_Name,Departure_Airport_Code, FL.Sheduled_Departure_Time,
FL.Arrival_Airport_Code, FL.Sheduled_Arrival_Time, F.Aireline_Name, F.Shedule_Dates,
SR.Flight_Number, SR.Leg_Number, SR.Date, SR.Seat_Number
FROM Flight F, Seat_Resavation SR, Leg_Instance LI, Flight_Leg FL
WHERE FL.Flight_Number = F.Flight_Number AND LI.Flight_Number = FL.Flight_Number AND
LI.Flight_Number = SR.Flight_Number
```


Results Messages

	Customer_Name	Departure_Airport_Code	Sheduled_Departure_Time	Arrival_Airport_Code	Sheduled_Arrival_Time	Aireline_Name	Shedule_Dates	Flight_Number	Leg_Number	Date	Seat_Number
1	R.K Wikramasinghe	AKD	2022-12-25 12:45:00.000	SYD	2022-12-26 01:30:00.000	Blue Sky	X8	AA432	65	2022-12-25	23
2	Mary Ann	MSK	2022-10-30 23:10:00.000	AKD	2022-10-31 13:45:00.000	American Airlines	X8	BA694	7	2022-10-30	96
3	S.N.P Lucas	ABQ	2022-08-08 18:00:00.000	SYD	2022-08-09 09:30:00.000	Qatar Airways	X6	BD674	34	2022-08-08	43
4	M.D Rajapaksha	CTF	2022-11-12 05:30:00.000	MSK	2022-11-12 22:00:00.000	Air Airways	X1	GF564	80	2022-11-12	46
5	Mary Ann	SYD	2022-10-19 21:30:00.000	ABQ	2022-10-20 02:30:00.000	Sri Lankan Airlines	X7	KL203	22	2022-10-19	20

2. Seat Reservation Administrator (Who is in charge of checking the available seats and allocating passengers for available seats)

```sql
CREATE VIEW [GET SEAT RESERVATION DETAILS] AS
SELECT LI.Airplane_ID, LI.Number_Of_Available_Seats, FL.Departure_Airport_Code,
FL.Sheduled_Departure_Time, FL.Arrival_Airport_Code, FL.Sheduled_Arrival_Time,
SR.Flight_Number, SR.Leg_Number, SR.Date, SR.Seat_Number, SR.Phone_Number,
SR.Customer_Name
FROM Seat_Resavation SR, Leg_Instance LI, Flight_Leg FL
WHERE LI.Flight_Number = FL.Flight_Number AND LI.Flight_Number = SR.Flight_Number
```



	Airplane_ID	Number_Of_Available_Seats	Departure_Airport_Code	Sheduled_Departure_Time	Arrival_Airport_Code	Sheduled_Arrival_Time	Flight_Number	Leg_Number	Date	Seat_Number	Phone_Number	Customer_Name
1	879	25	AKD	2022-12-25 12:45:00.000	SYD	2022-12-26 01:30:00.000	AA432	65	2022-12-25	23	0361937264	R.K Wikramasinghe
2	880	90	MSK	2022-10-30 23:10:00.000	AKD	2022-10-31 13:45:00.000	BA694	7	2022-10-30	96	0772846301	Mary Ann
3	789	50	ABQ	2022-08-08 18:00:00.000	SYD	2022-08-09 09:30:00.000	BD674	34	2022-08-08	43	0316435128	S.N.P Lucas
4	345	100	CTF	2022-11-12 05:30:00.000	MSK	2022-11-12 22:00:00.000	GF564	80	2022-11-12	46	0703819374	M.D Rajapaksha
5	123	30	SYD	2022-10-19 21:30:00.000	ABQ	2022-10-20 02:30:00.000	KL203	22	2022-10-19	20	0719247035	Mary Ann

Indexes

1. For Fare Table

```
CREATE INDEX Fare_Amount_idx
ON Fare(Flight_Number)
```

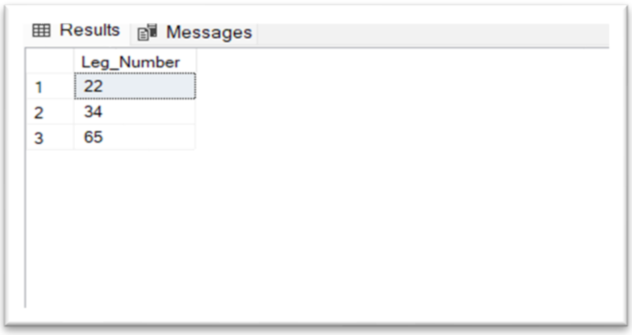
2. For Seat_Resavation_Table

```
CREATE INDEX Seat_Resavation_Customer_Name_idx
ON Seat_Resavation(Customer_Name)
```

Procedures

1. List all flight legs that depart or arrive at “Sydney” airport.

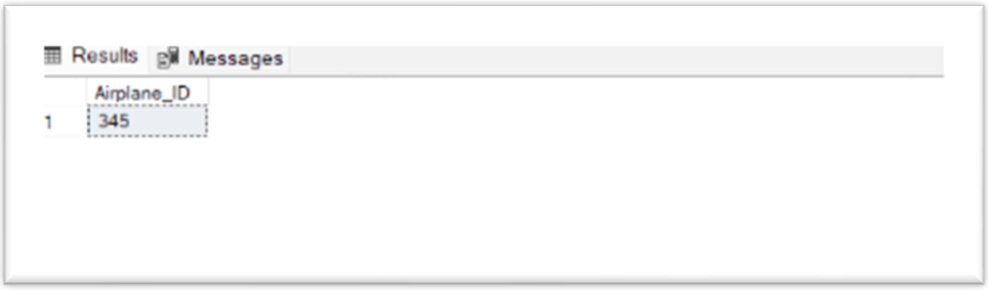
```
CREATE PROCEDURE DEPART_OR_ARRIVAL_AIRPORTS(@AirportName varchar(50))
AS
BEGIN
       SELECT DISTINCT Leg_Number
       FROM Flight_Leg F ,Airport A
       WHERE Departure_Airport_Code=@AirportName OR
Arrival_Airport_Code=@AirportName AND A.Airport_Code=@AirportName
END

EXEC DEPART_OR_ARRIVAL_AIRPORTS SYD
```



```
   Leg_Number
1  22
2  34
3  65
```

2. List all airplanes that can land at “Singapore” airport

```
CREATE PROCEDURE GET_AIRPLANE_IDS (@AirportName varchar(20))
AS
BEGIN
       SELECT A.Airplane_ID
       FROM   Airplane A,Leg_Instance L ,Airport D,Can_Land C
       WHERE   A.Airplane_ID=L.Airplane_ID AND
L.Depature_Airport_Code=D.Airport_Code  AND C.Airport_Code=D.Airport_Code  AND
C.Airport_Code=@AirportName

END

EXEC GET_Airplane_IDS CTF
```

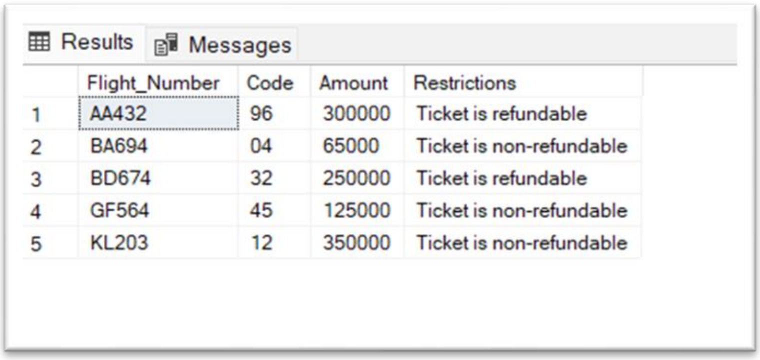


	Airplane_ID
1	345

3. Increase the fare of all tickets on flight "KL203" by 20%

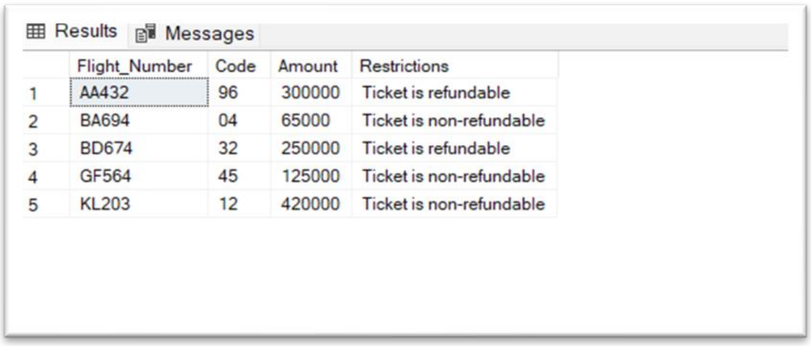
```
CREATE PROCEDURE INCREASE_TICKET_VALUES (@Percentage float,@Flight_Number
varchar(20))
AS
BEGIN
    UPDATE Fare
    SET    Amount=Amount+Amount*(@Percentage/100)
    WHERE Flight_Number=@Flight_Number
END

EXEC INCREASE_TICKET_VALUES 20,KL203
```

Before



	Flight_Number	Code	Amount	Restrictions
1	AA432	96	300000	Ticket is refundable
2	BA694	04	65000	Ticket is non-refundable
3	BD674	32	250000	Ticket is refundable
4	GF564	45	125000	Ticket is non-refundable
5	KL203	12	350000	Ticket is non-refundable

After

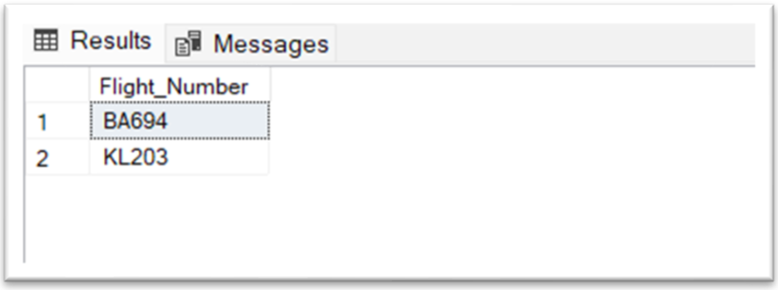


Results Messages

	Flight_Number	Code	Amount	Restrictions
1	AA432	96	300000	Ticket is refundable
2	BA694	04	65000	Ticket is non-refundable
3	BD674	32	250000	Ticket is refundable
4	GF564	45	125000	Ticket is non-refundable
5	KL203	12	420000	Ticket is non-refundable

4. List all the flights taken by passenger “Mary Ann”

```
CREATE PROCEDURE GET_FLIGHT_DETAILS_OF_A_CUSTOMER (@Customer_Name varchar(20))
AS
BEGIN
       SELECT Flight_Number
       FROM   Seat_Resavation
       WHERE   Customer_Name=@Customer_Name

END

EXEC GET_FLIGHT_DETAILS_OF_A_CUSTOMER 'Mary Ann'
```



Results Messages

	Flight_Number
1	BA694
2	KL203

Full Query

```
create table Airport(

	Airport_Code varchar(20) NOT NULL,
	Airport_Name varchar(60) NOT NULL,
	City varchar(20) NOT NULL,
	State varchar(20) NOT NUll,

	primary key(Airport_Code)

)

create table Airplane_Type
(
	Type_Name varchar(50) NOT NULL,
	Maximum_Seats int NOT NULL,
	Company varchar(10)  NOT NULL,
	primary key (Type_Name)

);


create table Airplane
(
	Airplane_ID varchar(20) NOT NULL,
	Total_Number_Of_Seats int NOT NULL,
	Airplane_Type varchar(50) NOT NULL,
	primary key (Airplane_ID),

	 constraint fk1_type_name foreign key (Airplane_Type) references
Airplane_Type(Type_name)

);

create table Flight_Leg(

	Flight_Number varchar(20) NOT NULL,
	Leg_Number int NOT NULL,
	Departure_Airport_Code varchar(20) NOT NULL,
	Sheduled_Departure_Time datetime NOT NULL,
	Arrival_Airport_Code varchar(20) NOT NULL,
	Sheduled_Arrival_Time datetime NOT NULL,

	constraint Flight_Leg_PK primary key (Flight_Number,Leg_Number),
	constraint Flight_Leg_FK1 foreign key (Departure_Airport_Code) references
Airport(Airport_Code),
	constraint Flight_Leg_FK2 foreign key (Arrival_Airport_Code) references
Airport(Airport_Code)


)
```

```
create table Leg_Instance(
	Flight_Number varchar(20) NOT NULL,
	Leg_Number int NOT NULL,
	Date date NOT NULL,
	Airplane_ID varchar(20) NOT NULL,
	Number_Of_Available_Seats int,
	Depature_Airport_Code varchar(20),
	Depature_Time  datetime,
	Arrival_Airport_Code varchar(20),
	Arrival_Time datetime

	constraint Leg_Instance_PK primary key (Flight_Number,Leg_Number,Date),
	constraint Leg_Instance_FK1 foreign key (Flight_Number,Leg_Number)  references
Flight_Leg (Flight_Number,Leg_Number),
	constraint Leg_Instance_FK2 foreign key (Depature_Airport_Code)  references
Airport (Airport_Code),
	constraint Leg_Instance_FK3 foreign key (Arrival_Airport_Code)  references
Airport (Airport_Code),
	constraint Leg_Instance_FK4 foreign key (Airplane_ID)  references
Airplane(Airplane_ID),

)

create table Seat_Resavation(

	Flight_Number varchar(20) NOT NULL,
	Leg_Number int NOT NULL,
	Date date,
	Seat_Number int,
	Customer_Name varchar(20),
	Phone_Number char(10) check (Phone_Number like '[0-9][0-9][0-9][0-9][0-9][0-
9][0-9][0-9][0-9][0-9]')

	constraint Seat_Resavation_PK primary key
(Flight_Number,Leg_Number,Date,Seat_Number),
	constraint Seat_resavation_FK foreign key (Flight_Number,Leg_Number,Date)
references Leg_Instance (Flight_Number,Leg_Number,Date)

)



create table Can_Land
(
	Airplane_Type_Name   varchar(50)   NOT NULL,
	Airport_Code varchar(20) NOT NULL,
	primary key (Airplane_Type_Name,Airport_Code),

	constraint fk1_airplane_type_namme foreign key (Airplane_Type_Name) references
Airplane_Type (Type_Name),
	constraint fk2_airport_code foreign key (Airport_Code) references Airport
(Airport_Code)

);


CREATE TABLE Flight(
```

```sql
Flight_Number varchar(20) NOT NULL,
Aireline_Name varchar(50) NOT NULL,
Shedule_Dates varchar(20) default 'X8',

CONSTRAINT flight_pk PRIMARY KEY (Flight_Number),
);



CREATE TABLE Fare (
Flight_Number varchar(20) NOT NULL,
Code varchar(20) NOT NULL,
Amount float,
Restrictions VARCHAR(200),

CONSTRAINT fare_pk PRIMARY KEY (Flight_Number, Code),
CONSTRAINT fare_fk FOREIGN KEY(Flight_Number) references Flight(Flight_Number)
);




insert into Airplane_Type
values('Airbus 707','500','Airbus')
insert into Airplane_Type
values('Boeing 757','300','Boing')
insert into Airplane_Type
values('Boeing 747','650','Boing')
insert into Airplane_Type
values('Embraer 175','200','Embraer')
insert into Airplane_Type
values('Airbus A350','350','Airbus')


insert into Airplane
values('123','200','Airbus 707')
insert into Airplane
values('789','100','Boeing 757')
insert into Airplane
values('345','350','Boeing 747')
insert into Airplane
values('879','50','Embraer 175')
insert into Airplane
values('880','170','Airbus A350')


insert into Airport
values('SYD','Kingsford Smith Airport','SYDNEY','AUSTRALIA')
insert into Airport
values('ABQ','ALBUQUERQUE INTERNATIONAL SUNPORT','AlBuQuerque.','USA')
insert into Airport
values('CTF',' Singapore Changi Airport','Changi','Singapore')
insert into Airport
values('AKD',' London Heathrow Airport','Heathrow','London')
insert into Airport
values('MSK',' Dallas Fort Worth International Airport','Dallas','US')

insert into Can_Land
values('Airbus 707','SYD')
```

```
insert into Can_Land
values('Boeing 757','ABQ')
insert into Can_Land
values('Boeing 747','CTF')
insert into Can_Land
values('Embraer 175','AKD')
insert into Can_Land
values('Airbus A350','MSK')



insert into Flight
values('KL203','Sri Lankan Airlines','X7')
insert into Flight
values('BD674','Qatar Airways','X6')
insert into Flight
values('GF564','Air Airways','X1')
insert into Flight
values('AA432','Blue Sky','X8')
insert into Flight
values('BA694','American Airlines','X8')

insert into Flight_Leg
values('KL203','22','SYD','2022-10-19 21:30:00','ABQ','2022-10-20 02:30:00')
insert into Flight_Leg
values('BD674','34','ABQ','2022-08-08 18:00:00','SYD','2022-08-09 09:30:00')
insert into Flight_Leg
values('GF564','80','CTF','2022-11-12 05:30:00','MSK','2022-11-12 22:00:00')
insert into Flight_Leg
values('AA432','65','AKD','2022-12-25 12:45:00','SYD','2022-12-26 01:30:00')
insert into Flight_Leg
values('BA694','07','MSK','2022-10-30 23:10:00','AKD','2022-10-31 13:45:00')


insert into Leg_Instance
values('KL203','22','2022-10-19','123',30,'SYD','2022-10-19 21:30:45','ABQ','2022-10-
20 02:30:40')
insert into Leg_Instance
values('BD674','34','2022-08-08','789',50,'ABQ','2022-08-08 05:30:45','SYD','2022-08-
19 23:00:30')
insert into Leg_Instance
values('GF564','80','2022-11-12','345',100,'CTF','2022-11-12 05:30:00','MSK','2022-11-
12 22:35:10')
insert into Leg_Instance
values('AA432','65','2022-12-25','879',25,'AKD','2022-12-25 12:45:00','SYD','2022-12-
26 12:00:55')
insert into Leg_Instance
values('BA694','07','2022-10-30','880',90,'MSK','2022-10-30 23:10:00','AKD','2022-10-
31 14:40:33')



insert into  Fare
values('KL203','12','350000','Ticket is non-refundable')
insert into  Fare
values('BD674','32','250000','Ticket is refundable')
insert into  Fare
values('GF564','45','125000','Ticket is non-refundable')
insert into  Fare
```

```
values('AA432','96','300000', 'Ticket is refundable')
insert into  Fare
values('BA694','04','65000','Ticket is non-refundable')



insert into Seat_Resavation
values('KL203','22','2022-10-19','20','Mary Ann','0719247035')
insert into Seat_Resavation
values('BD674','34','2022-08-08','43','S.N.P Lucas','0316435128')
insert into Seat_Resavation
values('GF564','80','2022-11-12','46','M.D Rajapaksha','0703819374')
insert into Seat_Resavation
values('AA432','65','2022-12-25','23','R.K Wikramasinghe','0361937264')
insert into Seat_Resavation
values('BA694','07','2022-10-30','96','Mary Ann','0772846301')


select * from Airplane
select * from Airplane_Type
select * from Airport
select * from Can_Land
select * from Fare
select * from Flight
select * from Flight_Leg
select * from Leg_Instance
select * from Seat_Resavation


--functions

CREATE FUNCTION AirplaneCount(@Airplane_company VARCHAR(20))
RETURNS integer
AS
BEGIN
    DECLARE @planeCount integer
    SELECT @planeCount = count(*)
    FROM Airplane_Type A
    WHERE A.Company = @Airplane_company
    RETURN @planeCount
END

Declare @count integer
Exec @count = AirplaneCount 'Boing'
Print @count


CREATE FUNCTION MaxSeats(@Airline VARCHAR(20))
RETURNS integer
AS
BEGIN
    DECLARE @SeatCount integer
    SELECT @SeatCount = max(Maximum_Seats)
    FROM Airplane_Type
    WHERE Company = @Airline
    RETURN @SeatCount
END

Declare @Seatcount integer
```

```
Exec @Seatcount = MaxSeats'Airbus'
Print @Seatcount



--trigger
CREATE TABLE Airport_Audit
(
Id int IDENTITY,
Description text
)




CREATE TRIGGER Insert_Airport
ON Airport
FOR INSERT
AS
BEGIN
  Declare @Id varchar(20)
  SELECT @Id = Airport_Code from inserted
  INSERT INTO Airport_Audit
  VALUES ('New Airport with Airport Code = ' + CAST(@Id AS VARCHAR(20)) + ' is added
at ' + CAST(Getdate() AS VARCHAR(22)))
END


--delete trigger
CREATE TRIGGER Delete_Airport
ON Airport
FOR DELETE
AS
BEGIN
  Declare @Id varchar(20)
  SELECT @Id = Airport_Code from deleted
  INSERT INTO Airport_Audit
  VALUES ('An existing Airport with Airport Code = ' + CAST(@Id AS VARCHAR(20)) + ' is
deleted at ' + CAST(Getdate() AS VARCHAR(22)))
END
drop trigger Delete_Airport


--view 1 for passenger
CREATE VIEW [GET FLIGHT DETAILS] AS
SELECT SR.Customer_Name,Departure_Airport_Code, FL.Sheduled_Departure_Time,
FL.Arrival_Airport_Code, FL.Sheduled_Arrival_Time, F.Aireline_Name, F.Shedule_Dates,
SR.Flight_Number, SR.Leg_Number, SR.Date, SR.Seat_Number
FROM Flight F, Seat_Resavation SR, Leg_Instance LI, Flight_Leg FL
WHERE FL.Flight_Number = F.Flight_Number AND LI.Flight_Number = FL.Flight_Number AND
LI.Flight_Number = SR.Flight_Number

SELECT *
FROM [GET FLIGHT DETAILS]




--view 2 for seat reservation
```

```
CREATE VIEW [GET SEAT RESERVATION DETAILS] AS
SELECT LI.Airplane_ID, LI.Number_Of_Available_Seats, FL.Departure_Airport_Code,
FL.Sheduled_Departure_Time, FL.Arrival_Airport_Code, FL.Sheduled_Arrival_Time,
SR.Flight_Number, SR.Leg_Number, SR.Date, SR.Seat_Number, SR.Phone_Number,
SR.Customer_Name
FROM Seat_Resavation SR, Leg_Instance LI, Flight_Leg FL
WHERE LI.Flight_Number = FL.Flight_Number AND LI.Flight_Number = SR.Flight_Number

SELECT *
FROM [GET SEAT RESERVATION DETAILS]

CREATE INDEX Fare_Amount_idx
ON Fare(Flight_Number)


CREATE INDEX Seat_Resavation_Customer_Name_idx
ON Seat_Resavation(Customer_Name)



CREATE PROCEDURE DEPART_OR_ARRIVAL_AIRPORTS(@AirportName varchar(50))
AS
BEGIN
       SELECT DISTINCT Leg_Number
       FROM Flight_Leg F ,Airport A
       WHERE Departure_Airport_Code=@AirportName OR Arrival_Airport_Code=@AirportName
AND A.Airport_Code=@AirportName
END

EXEC DEPART_OR_ARRIVAL_AIRPORTS SYD

CREATE PROCEDURE GET_AIRPLANE_IDS (@AirportName varchar(20))
AS
BEGIN
       SELECT A.Airplane_ID
       FROM   Airplane A,Leg_Instance L ,Airport D,Can_Land C
       WHERE   A.Airplane_ID=L.Airplane_ID AND  L.Depature_Airport_Code=D.Airport_Code
AND C.Airport_Code=D.Airport_Code  AND C.Airport_Code=@AirportName

END

EXEC GET_Airplane_IDS CTF


CREATE PROCEDURE INCREASE_TICKET_VALUES (@Percentage float,@Flight_Number varchar(20))
AS
BEGIN
       UPDATE Fare
       SET    Amount=Amount+Amount*(@Percentage/100)
       WHERE Flight_Number=@Flight_Number
END

EXEC INCREASE_TICKET_VALUES 20,KL203


CREATE PROCEDURE GET_FLIGHT_DETAILS_OF_A_CUSTOMER (@Customer_Name varchar(20))
AS
BEGIN
       SELECT Flight_Number
       FROM   Seat_Resavation
       WHERE   Customer_Name=@Customer_Name

END

EXEC GET_FLIGHT_DETAILS_OF_A_CUSTOMER 'Mary Ann'


drop table Seat_Resavation
drop table Leg_Instance
drop table Airplane
drop table Flight_Leg
drop table Fare
drop table Can_Land
drop table Flight
drop table Airplane_Type
drop table Airport


drop view [GET FLIGHT DETAILS]
drop view [GET SEAT RESERVATION DETAILS]
DROP INDEX Fare_Amount_idx ON Fare
DROP INDEX Seat_Resavation_Customer_Name_idx ON Seat_Resavation
DROP PROCEDURE DEPART_OR_ARRIVAL_AIRPORTS
DROP PROCEDURE GET_AIRPLANE_IDS
DROP PROCEDURE INCREASE_TICKET_VALUES
DROP PROCEDURE GET_FLIGHT_DETAILS_OF_A_CUSTOMER
```

Part 2

1. SQL injection

SQL injection is web base security vulnerability that allows an attacker to use malicious code to interfere with the queries that an application sends to its database. it enables the actor to retrieve sensitive information that is not disclosed to retrieve by database administrators. a successful attack may result in viewing a user list, deletion of the entire table, and might gain administrative privileges or modification of database data which is extremely valuable to business and cause unintentional behavior to application content. attackers can improvise SQL injection attacks to compromise the underlying server or other back-end infrastructure or perform DDos or dos.

Examples of SQL injections

- Retrieving hidden data.
- Subverting application logic.
- Examining the database.

Retrieving hidden data.

This method involves retrieving data by modifying the query. in a scenario of a user using a shopping application, the user clicks on the gift category the browser sends a request URL this makes the application make an SQL request from the database to get the necessary information. in the SQL query, it is supposed to return only the released product information, but an attacker can modify this URL request and get the information of the unreleased product as well as modify that application to display all the products in any category including publicly censored categories.

Subverting application logic

When an application lets a user log into it.it lets users enter their name and password then the application connects to the database and checks credentials by using a query. if the query returns the details of a user the application let the user log in otherwise reject. attackers can manipulate the SQL query so that it bypasses the checking constraint in SQL query and lets an attacker in without a password. attackers even can log in as administrators.

Examining the database

These methods involve obtaining some information about the database. this paves the way for the actor to further exploit the database. depending on the database type using the appropriate query can execute the SQL command to display the database version and its infrastructure. also, can view the information about the tables and the columns of the database that they contain.

Countermeasures SQL injection vulnerabilities.

Some SQL injection countermeasures can be used to prevent database damage. SQL injection usually occurred due to mostly poor coding and website administration

- Using prepared statements
- Object-relational mapping (ORM) framework
- Input validation
- Database permissions
- Evaluate all data input
- Use parameterized queries

Using prepared statements

using precompiled SQL prepare statements command inside the program to use several times throughout the application’s lifecycle.ps input commands are blind statements blind inputs parameters prevent any command alterations during SQL injections. This enables not to cause any unwanted behavior of the program.

Object-relational mapping (ORM) framework.

This is a technique involving tables of the database being converted into a structure of the database. programmers and developers can safely communicate with the tables to run a command like UPDATE, SERCH, or DELETE an example of this can hibernate in java using these parameters like ‘username’ and ‘password’ can be passed as a string rather than with the SQL command which is much safer.

Input validation

Validation plays a key role in preventing SQLi. Before inputs are submitted and processed, they are validated by the server. Email validation is one input validation that function as an email validator. There are two types of validations client-side validation and server-side.

- Client-side

  Before submitting data to the database validate the data.

- Server-side

  After the data is submitted from the applications validate the data before submitting it to the database.

Database permissions

Setting database permission on the database username and password credentials as possible. If only the data that need to be displayed eliminates the permissions to use an update, insert, and delete permissions. having two different users one with administrative privileges and the other as a guest with minimum privileges.

Evaluate all data input

All data and URL query strings should be evaluated. If data is being transmitted using a query by the users testing whether it is an integer or a string using an inbuilt function such as is numeric.

Use parameterized queries

Using parameterized queries eliminates the potential of string concatenation when connecting to the database. Always using parameterized queries enables safety rather than creating the SQL.

2. Weak Authentication.

Weak authentications allow attackers to assume the identity of ligament database users by stealing or obtaining login credentials. An attacker may go to any number of lengths to obtain these credentials.

- **Brute force-** the attacker repeatedly inputs the user credentials combinations until finding the one that succeeds. Brute force might involve simple guessing or systematic enumeration of all possible key combinations often archived by an automated software program.
- **Social engineering-** or honey pot attacker leverage the natural human tendency to trust to convince others to provide victims user credentials. this can be achieved by pretending to be its manager and requesting login credentials or using fraud calling.
- **Direct credential theft-** attacker steals login credentials by copying or stealing victims' identities. This can be copying post-IT notes, and passwords, once the attacker succeeds attacker gains the same privileges as the victims.

Countermeasures Weak Authentication.

Proacting against password cracking.

It is a necessity to protect the application against password cracking this can be accomplished by Introducing increased delays whenever credentials are entered incorrectly numerous times introducing a time delay.

Strong authentication.

The strongest practice for authentication technologies and policies must be implemented. Two FA authentication are much safer approach. Cost the other hand makes 2FA impractical.

Directory integration.

Stronger authentication mechanisms should implement with enterprise directory infrastructure. This can enable the user to use a single set of passwords across multiple databases and applications. It is much easier for users to keep it in memory. And overall, this makes 2fa more cost-effective.

Secure authentication protections.

Despite the best effort in strong authentication break downs occur. password policies are ignored, and a lucky attacker might brute force the much decent passwords as well. to deal with this dynamic profiling, failed login detection and authentication assessment must be imposed to authenticate protection.

References

[1] PortSqigger, "web security academy," PortSwigger Ltd, 2022. [Online]. Available: https://portswigger.net/web-security/sql-injection.

[2] I. Imperva, "Top ten database security threats - schell," imperva, [Online]. Available: https://schell.com/Top_Ten_Database_Threats.pdf.

[3] Affinity, "INCREASE SECURITY. MAINTAIN COMPLIANCE. RETAIN CONTROL.," Affinity security servies, [Online]. Available: https://affinity-it-security.com/what-is-weak-authentication/.