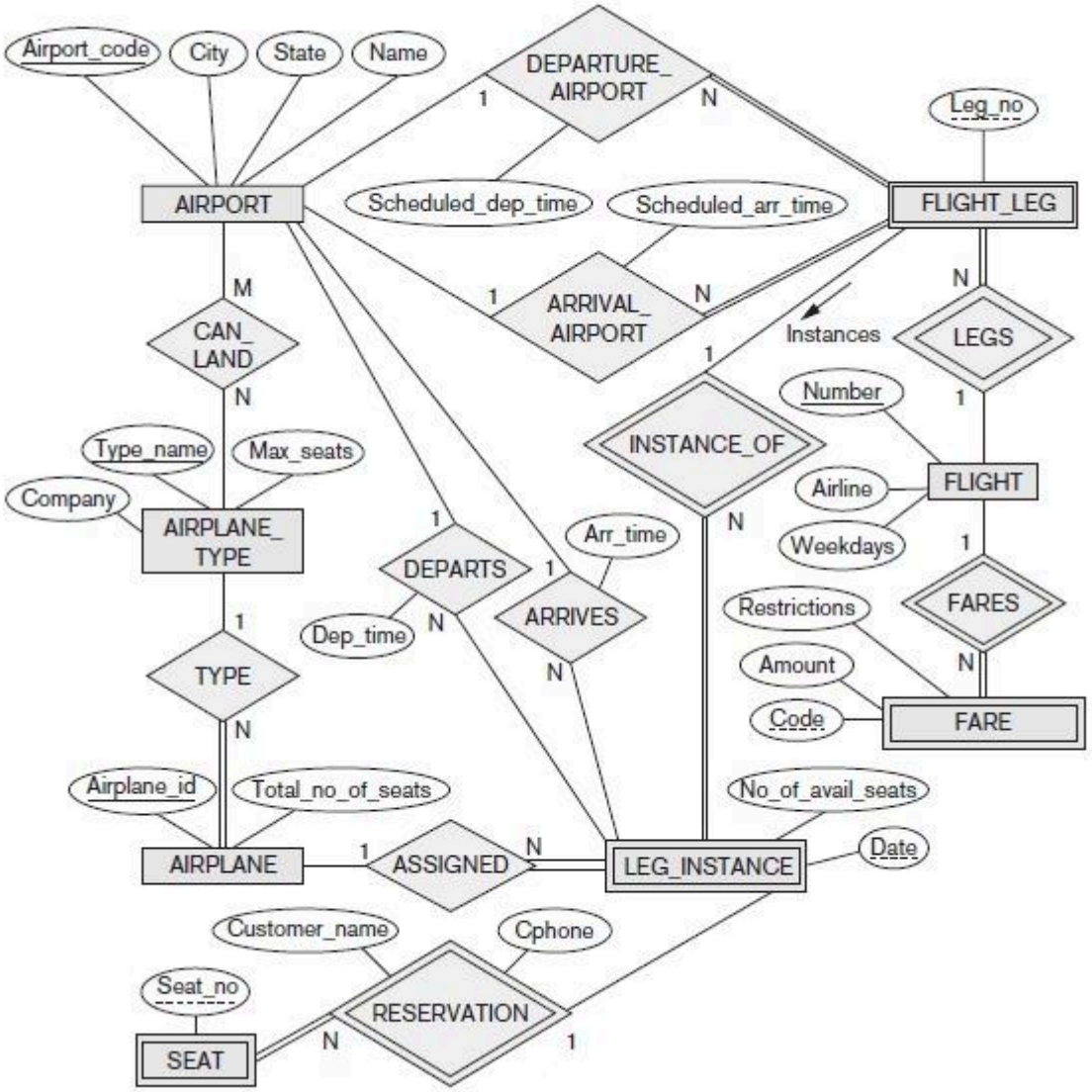
Question: 01



Firstly Creating the Database **Airline_Reservation_System** using these SQL Queries:

```
-- Create the Database
use master
go
create database Airline_Reservation_System

-- Use the Database
use [Airline_Reservation_System]
```

After creating the Database in SQL Server, Creating the Table **Airport:**

```sql
-- Create Table Airport
CREATE TABLE Airport (
    airport_code VARCHAR(10) PRIMARY KEY,
    city VARCHAR(100) NOT NULL,
    state VARCHAR(100) NOT NULL,
    name VARCHAR(255) NOT NULL
);
```

This will Create the Table with Columns:

airport_code	city	state	name

Here **airport_code** is a **Primary Key.** Now Creating the Table **Flight**:

```sql
-- Create Table Flight
CREATE TABLE Flight (
number INT PRIMARY KEY,
airline VARCHAR(100) NOT NULL,
weekdays VARCHAR(50) NOT NULL
);
```

This will create **Flight** table with columns:

number	airline	weekdays

Here, **number** is a **Primary Key**. Creating the table for Weak Entity **Flight_Leg** and partial key (discriminator) **Leg_no:**

```sql
-- Create Table Flight_Leg
CREATE TABLE Flight_Leg (
Leg_no INT NOT NULL,
number INT NOT NULL, -- This is connected to Flight Number from Flight Table
FOREIGN KEY (number) REFERENCES Flight(number),
PRIMARY KEY (Leg_no, number)
);
```

This will create **Flight_Leg** table:

number	Leg_no

Here number is **Foreign key** from **Flight** table and **Leg_no** is **partial key** that depends upon the **number (primary key)** from **Flight** table. This creates a final composite key.

Creating a table for **Departure_Airport** relation:

```
-- Create Table Departure_Airport
CREATE TABLE Departure_Airport (
scheduled_dep_time DATETIME NOT NULL,
airport_code VARCHAR(10) NOT NULL, -- This is connected to Airport Code from
Airport Table
Leg_no INT NOT NULL, -- This is connected to Leg Number from Flight_Leg
Table
number INT NOT NULL, -- This is connected to Flight Number from Flight Table
FOREIGN KEY (airport_code) REFERENCES Airport(airport_code),
FOREIGN KEY (Leg_no, number) REFERENCES Flight_Leg(Leg_no, number), --
Referencing both columns in Flight_Leg
PRIMARY KEY (airport_code, Leg_no)
);
```

It isn't possible to only get **Leg_no** from **Flight_Leg** table because it is making a composite key with Flight **number** from **Flight** table. So, you have to Reference them both.

scheduled_dep_time	airport_code	Leg_no	number

Now, making a table for **Arrival_Airport** relation.

```
-- Create Table Departure_Airport
CREATE TABLE Arival_Airport (
    scheduled_arr_time DATETIME NOT NULL,
    airport_code VARCHAR(10) NOT NULL, -- This is connected to Airport Code
from Airport Table
    Leg_no INT NOT NULL, -- This is connected to Leg Number from Flight_Leg
Table
    number INT NOT NULL, -- This is connected to Flight Number from Flight
Table
    FOREIGN KEY (airport_code) REFERENCES Airport(airport_code),
    FOREIGN KEY (Leg_no, number) REFERENCES Flight_Leg(Leg_no, number), --
Referencing both columns in Flight_Leg
    PRIMARY KEY (airport_code, Leg_no)
);
```

This will give us the table:

scheduled_arr_time	airport_code	Leg_no	number

Creating the table **Airplane_Type:**

```sql
-- Create Table Airplane_Type
CREATE TABLE Airplane_Type (
    Type_name VARCHAR(100) PRIMARY KEY,
    Company VARCHAR(100) NOT NULL,
    Max_Seats INT NOT NULL
);
```

Type_name	Company	Max_Seats

Here **Type_name** is a **Primary Key**. Now for table **Airplane:**

```sql
-- Create Table Airplane
CREATE TABLE Airplane (
    Airplane_ID INT PRIMARY KEY,
    Total_Seats INT NOT NULL,
);
```

Airplane_ID	Total_Seats

Here, **Airplane_ID** is a **Primary Key.** For the Weak Entity **Fare** with the relation **Fares:**

```sql
-- Create Table Flight_Leg
CREATE TABLE Fare (
    Code INT NOT NULL,
    Restrictions VARCHAR(100) NOT NULL,
    Amount INT NOT NULL,
    number INT NOT NULL, -- This is connected to the Flight Number from the
Flight Table
    FOREIGN KEY (number) REFERENCES Flight(number),
    PRIMARY KEY (Code, number)
);
```

Code	Restrictions	Amount	number

Here, **Code** and **number** are a **Composite Primary Key**. Creating table **Leg_Instance:**

```sql
-- Create Table Leg_Instance
CREATE TABLE Leg_Instance (
    No_of_seats INT NOT NULL,
    Date DATE NOT NULL,
    Leg_no INT NOT NULL,
    number INT NOT NULL, -- This is connected to the Flight Number from the
```

```
Flight Table
    FOREIGN KEY (number) REFERENCES Flight(number),
    FOREIGN KEY (Leg_no, number) REFERENCES Flight_Leg(Leg_no, number), --
Referencing both columns in Flight_Leg
    PRIMARY KEY (Leg_no, number, Date)
);
```

This creates table:

No_of_seats	Date	Leg_no	number

To show the **Can_Land** attribute, we create the table:

```
-- Create Can_Land Table
CREATE TABLE Can_Land (
    Type_name VARCHAR(100) NOT NULL, -- This is connected to Type Name from
Airplane Type Table
    airport_code VARCHAR(10) NOT NULL, -- This is connected to Airport Code
from Airport Table
    FOREIGN KEY (Type_name) REFERENCES Airplane_Type(Type_name),
    FOREIGN KEY (airport_code) REFERENCES Airport(airport_code),
    PRIMARY KEY (Type_name, airport_code)
);
```

Type_name	airport_code

After this table, we edit the **Airplane** table to put the **Type_name** key from **Airplane Type** table as a foreign key inside the **Airplane** table**.**

```
-- Alter Table Airplane to add Type_name as Foreign Key
ALTER TABLE Airplane
ADD Type_name VARCHAR(100) NOT NULL,
FOREIGN KEY (Type_name) REFERENCES Airplane_Type(Type_name);
```

Now the Airplane Table schema has changed to:

Airplane_ID	Total_Seats	Type_Name

Here **Airplane_ID** is **Primary key** and **Type_Name** is **Foreign Key.**

```
-- Alter Table Leg_Instance to add Airplane_ID as Foreign Key
```

```
ALTER TABLE Leg_Instance
ADD Airplane_ID INT NOT NULL,
FOREIGN KEY (Airplane_ID) REFERENCES Airplane(Airplane_ID);
```

Now the **Leg_Instance** table is:

No of Seats	Date	Leg_no	number	Airplane_ID

Here, **Leg_no, number, Date** are **composite primary keys** and **Airplane_ID** is **Foreign Key.** For the Departs Relation:

```
-- Create Table Departs
CREATE TABLE Departs (
    Dep_time TIME NOT NULL,
    Airport_code VARCHAR(10) NOT NULL, -- This is connected to Airport Code
from Airport Table
    Date DATE NOT NULL, -- This is connected to Date from Leg_Instance Table
    Leg_no INT NOT NULL, -- This is connected to Leg Number from Flight_Leg
Table
    number INT NOT NULL, -- This is connected to Flight Number from Flight
Table
    FOREIGN KEY (Airport_code) REFERENCES Airport(airport_code),
    FOREIGN KEY (Leg_no, number, Date) REFERENCES Leg_Instance(Leg_no,
number, Date), -- Referencing all columns in Leg_Instance
    PRIMARY KEY (Airport_code, Date, Leg_no, number)
);
```

Dep_time	Airport_Code	Date	Leg_no	number

Here, all underlined column names are **composite primary keys.** Only, Dep_time is a Normal key. Similarly, for **Arrives:**

```
-- Create Table Arrives
CREATE TABLE Arrives (
    Arr_time TIME NOT NULL,
    Airport_code VARCHAR(10) NOT NULL, -- This is connected to Airport Code
from Airport Table
    Date DATE NOT NULL, -- This is connected to Date from Leg_Instance Table
    Leg_no INT NOT NULL, -- This is connected to Leg Number from Flight_Leg
Table
    number INT NOT NULL, -- This is connected to Flight Number from Flight
Table
    FOREIGN KEY (Airport_code) REFERENCES Airport(airport_code),
    FOREIGN KEY (Leg_no, number, Date) REFERENCES Leg_Instance(Leg_no,
```

```
number, Date), -- Referencing all columns in Leg_Instance
    PRIMARY KEY (Airport_code, Date, Leg_no, number)
);
```

Arr_time	Airport_Code	Date	Leg_no	number

Here, all underlined column names are **composite primary keys.** Only, Arr_time is a Normal key. For the weak entity, **Seat:**

```
-- Create table Seat
CREATE TABLE Seat (
    Seat_no INT NOT NULL,
    Customer_name VARCHAR(100) NOT NULL,
    Cphone INT NOT NULL,
    DATE DATE NOT NULL, -- This is connected to Date from Leg_Instance Table
    Leg_no INT NOT NULL, -- This is connected to Leg Number from Flight_Leg
Table
    number INT NOT NULL, -- This is connected to Flight Number from Flight
Table
    FOREIGN KEY (Leg_no, number, Date) REFERENCES Leg_Instance(Leg_no,
number, Date), -- Referencing all columns in Leg_Instance
    PRIMARY KEY (Seat_no, Date, Leg_no, number)
);
```

Seat_no	name	Cphone	Date	Leg_no	number

Here all underlined ones are **composite** primary keys. Now, for the **Reservation** table:

```
-- Create table Reservation
CREATE TABLE Reservation (
    Seat_no INT NOT NULL,
    Customer_name VARCHAR(100) NOT NULL,
    Cphone INT NOT NULL,
    DATE DATE NOT NULL, -- This is connected to Date from Leg_Instance Table
    Leg_no INT NOT NULL, -- This is connected to Leg Number from Flight_Leg
Table
    number INT NOT NULL, -- This is connected to Flight Number from Flight
Table
    FOREIGN KEY (Leg_no, number, Date) REFERENCES Leg_Instance(Leg_no,
number, Date), -- Referencing all columns in Leg_Instance
    PRIMARY KEY (Seat_no, Date, Leg_no, number)
);
```

Seat_no	name	Cphone	Date	Leg_no	number

Here underlined, keys are composite primary keys.

Assumptions (Extra):

- Whenever, a composite key is used. I used it because there is an 1 to N relation between a strong and an (N) relation weak entity.
- In a normal 1 to N relation, you make a primary key of 1 Entity as a foreign key in N entity. However, when we met the Course Instructor in her office she said we can make a separate table for both 1 to N and N to M relations, so I did that instead.
- I carried the Composite keys after already made tables as primary keys to other tables. Because, there can't be 2 primary keys in a table (without composition).

The SQL Code to Drop the Tables and Database is given below:

```
-- Drop the Tables
DROP TABLE Airport;
DROP TABLE Flight;
DROP TABLE Flight_Leg;
DROP TABLE Departure_Airport;
DROP TABLE Arival_Airport;
DROP TABLE Airplane_Type;
DROP TABLE Airplane;
DROP TABLE Fare;
DROP TABLE Leg_Instance;
DROP TABLE Can_Land;
DROP TABLE Departs;
DROP TABLE Arrives;
DROP TABLE Seat;
DROP TABLE Reservation;

-- Drop the Database
DROP DATABASE Airline_Reservation_System;
```

I uploaded the original SQL File to Google Classroom as `Airline_Reservation_System.sql` and you can view all the Code i used (from creating Database to Final Schema) in one place there. Moreover, the Schema Diagram for this Database is given on the Next Page: