Conestoga College – PROG2111 Relational Databases
Final Project – Phase 1: Project Idea & Use Case

Project Title: Air Condor Airline Reservation System

Student Name: Aliaksandr Kazeika (9061886)

Project Overview

This project will design and implement a relational database for an airline reservation system used by the airline **Air Condor** and its brands (Air Condor, Air Condor Express, Air Condor Red). The system will support two main areas:

1. **System provisioning** – managing aircraft types, individual aircraft, airline brands, airports, and scheduled flights.
2. **Customer service** – allowing customers to search for flights, create bookings, add passengers, assign seats, and issue tickets.

The purpose of this project is to build a clear and well-normalized relational database that stores essential information about flights, customers, passengers, bookings, tickets, and seat assignments.

Use Case Description

A **customer** can search for available flights by origin, destination, and date. After selecting a flight, the customer creates a **booking**. Each booking can include one or more **passengers**. Passengers are not the same as customers—one customer can purchase tickets for other people (family members, friends, etc.).

For every passenger on a flight, the system must assign a **specific seat**. The system also tracks the **aircraft** used for each flight, departure and arrival **airports**, and the **ticket** issued to each passenger.

The database will be connected with a C# program using **ADO.NET MySQL API**. The C# application will perform CRUD operations such as:

- Creating flights
- Registering customers
- Creating bookings
- Adding passengers
- Assigning seats
- Viewing bookings and tickets

This project will include:

- An ERD diagram
- Normalization from UNF → 1NF → 2NF → 3NF
- SQL DDL scripts (CREATE TABLE statements)
- A C# application demonstrating CRUD functionality
- Final documentation and GitHub repository

GitHub Repository

A GitHub repository will be created for this project and will contain:

- ERD (PDF)

- SQL DDL script
- C# project with CRUD code
- Final project documentation

Phase 2 — ERD Entities and Relationships

The following section defines all entities and relationships that will appear in the ERD for the *Air Condor Airline Reservation System*. These entities were designed to support aircraft management, flight scheduling, customer bookings, passengers, and seat assignments.

1. AirlineBrand

Represents the airline brands operated by Air Condor.

Attributes:

- BrandID (PK)
- Name
- Code

2. Airport

Represents airports used in the flight network.

Attributes:

- AirportCode (PK)
- Name
- City
- Country

3. AircraftType

Represents the type/model of an aircraft.

Attributes:

- AircraftTypeID (PK)
- Manufacturer
- Model
- TotalSeats

4. Aircraft

Represents a specific aircraft in the fleet.

Attributes:

- AircraftID (PK)
- AircraftTypeID (FK → AircraftType)
- BrandID (FK → AirlineBrand)

5. Seat

Represents an individual seat on a specific aircraft.

Attributes:

- SeatID (PK)
- AircraftID (FK → Aircraft)
- SeatNumber
- SeatClass

6. Flight

Represents a scheduled flight operated by Air Condor.

Attributes:

- FlightID (PK)
- FlightNumber
- BrandID (FK → AirlineBrand)
- AircraftID (FK → Aircraft)
- DepartureAirportCode (FK → Airport)
- ArrivalAirportCode (FK → Airport)
- DepartureDateTime

- ArrivalDateTime
- BasePrice

7. Customer

Represents the person who pays for the booking.

Attributes:

- CustomerID (PK)
- FirstName
- LastName
- Email
- Phone

8. Passenger

Represents the person who actually travels.
Passengers are not the same as customers.

Attributes:

- PassengerID (PK)
- FirstName
- LastName
- DateOfBirth
- PassportNumber

9. Booking

Represents a booking created by a customer for a specific flight.

Attributes:

- BookingID (PK)
- CustomerID (FK → Customer)
- FlightID (FK → Flight)
- BookingDate
- PaymentStatus
- TotalAmount

10. BookingPassenger

Represents which passengers belong to a specific booking.
This resolves the many-to-many relationship between Booking and Passenger.

Attributes:

- BookingPassengerID (PK)
- BookingID (FK → Booking)
- PassengerID (FK → Passenger)

11. Ticket (SeatAssignment)

Represents the assigned seat and ticket information for each passenger in a booking.

Attributes:

- TicketID (PK)
- BookingPassengerID (FK → BookingPassenger)
- SeatID (FK → Seat)
- TicketNumber
- PricePaid

Summary of Key Relationships

- A **Brand** can operate many **Aircraft** and **Flights**.
- An **AircraftType** can be used by many **Aircraft**.
- An **Aircraft** contains many **Seats**.
- A **Flight** uses one **Aircraft** and involves two **Airports** (departure and arrival).
- A **Customer** creates **Bookings**.
- A **Booking** is for one **Flight** but can contain multiple **Passengers**.
- A **Passenger** can appear in multiple **Bookings** through **BookingPassenger**.
- Each passenger in a booking receives a **Ticket** linked to a specific **Seat**.

ERD Overview

The Entity-Relationship Diagram (ERD) for the *Air Condor Airline Reservation System* was designed based on the entities and relationships defined in Phase 2. The goal of the ERD is to clearly show how airlines, aircraft, flights, customers, passengers, bookings, and seat assignments are connected in the database.

The ERD uses crow's foot notation to represent cardinalities (one-to-many and many-to-many relationships). Each entity has a primary key (PK), and foreign keys (FK) are used to enforce relationships between tables.

Main Relationships

The key relationships in the ERD are the following:

1. **AirlineBrand → Aircraft (1:N)**
   One airline brand can operate many aircraft. Each aircraft belongs to exactly one brand.
2. **AircraftType → Aircraft (1:N)**
   One aircraft type (for example, Airbus A330-300) can be used by many specific

aircraft.
Each aircraft is of exactly one aircraft type.

3. **Aircraft → Seat (1:N)**
   One aircraft contains many seats.
   Each seat is defined for a specific aircraft (for example, AircraftID = "AC-330-01" with seats 1A, 1B, 2A, 2B, etc.).
4. **AirlineBrand → Flight (1:N)**
   One airline brand can operate many flights.
   Each flight is associated with exactly one brand.
5. **Airport → Flight (Departure and Arrival) (1:N for each role)**
   An airport can be the departure airport for many flights and the arrival airport for many flights.
   Each flight has exactly one departure airport and one arrival airport.
6. **Aircraft → Flight (1:N)**
   One aircraft can be scheduled for many flights over time.
   Each flight is operated by one aircraft.
7. **Customer → Booking (1:N)**
   One customer can create many bookings.
   Each booking is created by exactly one customer.
8. **Flight → Booking (1:N)**
   One flight can appear in many bookings (different customers booking the same flight).
   Each booking is associated with exactly one flight.
   This relationship is critical and ensures that every booking contains information about the specific flight.
9. **Booking ↔ Passenger (M:N via BookingPassenger)**
   a. One booking can include multiple passengers.
   b. The same passenger can appear in multiple bookings (for example, a frequent traveler).
      This many-to-many relationship is resolved using the **BookingPassenger** junction table.
10. **BookingPassenger → Ticket (1:N)**
    One record in BookingPassenger represents a passenger in a specific booking.
    Each such passenger can have one or more tickets (for example, different segments), but in this project we assume **one ticket per passenger per booking**.
11. **Seat → Ticket (1:N)**
    One seat can be assigned to many tickets over time (for different flights and dates),

but for a single flight and booking it is assigned to only one passenger.
Each ticket references exactly one seat.

ERD Diagram

The ERD was created using a standard diagramming tool (for example, draw.io / diagrams.net, MySQL Workbench or Lucidchart).

The diagram includes the following entities:

- AirlineBrand
- Airport
- AircraftType
- Aircraft
- Seat
- Flight
- Customer
- Passenger
- Booking
- BookingPassenger
- Ticket

Each entity shows its primary key (PK) at the top, followed by attributes. Foreign keys (FK) are clearly marked to indicate relationships between tables.

The ERD diagram is exported as a PDF file and attached to this document as **Figure 1: Air Condor Airline Reservation System – ERD**.

Phase 4 – Normalization (UNF → 1NF → 2NF → 3NF)

UNF – Unnormalized Form

The following table contains repeating groups, multi-valued attributes, and non-atomic values. This represents how booking information might appear before normalization.

BookingID	CustomerName	CustomerEmail	FlightNumber	DepartureAirport	ArrivalAirport	PassengerNames	SeatNumbers	TotalAmount
B001	John Smith	john@mail.com	AC120	YYZ	ZAG	John Smith; Anna Smith	12A; 12B	1450.00
B002	Mary Brown	mary@mail.com	AC200	YYZ	FRA	Mary Brown	14C	800.00

Problems in UNF:

- PassengerNames contain multiple values.
- SeatNumbers contain multiple values.
- Customer information mixed together with flight and passenger data.
- Repeated data causes anomalies.

1NF – First Normal Form

Each row must contain atomic (single) values.
We split repeating values into separate rows.

BookingID	CustomerName	CustomerEmail	FlightNumber	DepartureAirport	ArrivalAirport	PassengerName	SeatNumber	TotalAmount
B001	John Smith	john@mail.com	AC120	YYZ	ZAG	John Smith	12A	1450.00
B001	John Smith	john@mail.com	AC120	YYZ	ZAG	Anna Smith	12B	1450.00
B002	Mary Brown	mary@mail.com	AC200	YYZ	FRA	Mary Brown	14C	800.00

Issues remaining in 1NF:

- Customer attributes depend only on Customer, not on BookingID.
- Flight details depend only on FlightNumber.
- Passenger details are repeated and mixed with booking information.

2NF – Second Normal Form

Remove partial dependencies by separating data into multiple entities:

Customer

| CustomerID | FirstName | LastName | Email |

Passenger

| PassengerID | FirstName | LastName | DateOfBirth | PassportNumber |

Flight

| FlightID | FlightNumber | DepartureAirport | ArrivalAirport | DepartureDateTime | ArrivalDateTime | BasePrice |

Booking

| BookingID | CustomerID | FlightID | BookingDate | PaymentStatus | TotalAmount |

BookingPassenger

| BookingPassengerID | BookingID | PassengerID |

Why this is 2NF:

- No attribute depends only on part of a composite key.
- Customer, Flight, Booking, and Passenger data are separated correctly.

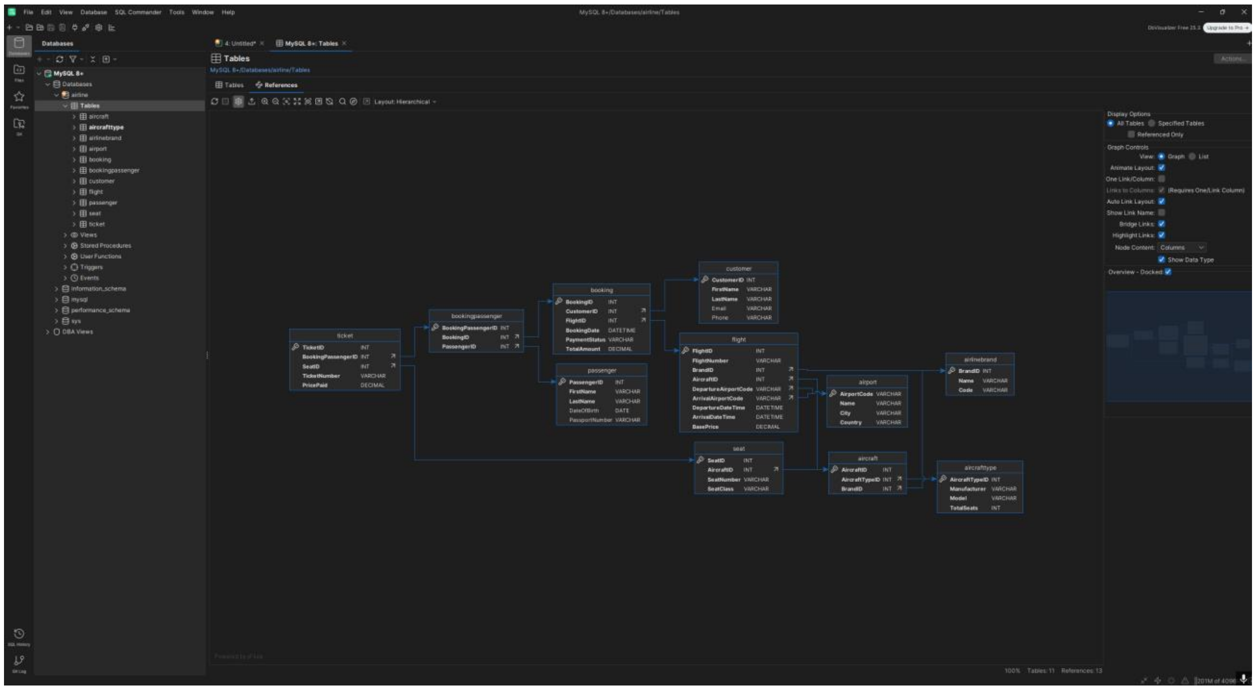
3NF – Third Normal Form

Remove all transitive dependencies:

- Airport data moved to **Airport** table.
- Aircraft and Brand data moved to **Aircraft**, **AircraftType**, **AirlineBrand**.
- Seats moved to **Seat**.
- Tickets moved to **Ticket**.

This results in the final list of normalized tables (same as ERD):

- AirlineBrand
- AircraftType
- Aircraft
- Seat
- Airport
- Flight
- Customer
- Passenger
- Booking
- BookingPassenger
- Ticket



All attributes now depend only on the key, the whole key, and nothing but the key.

IR CONDOR AIRLINE RESERVATION SYSTEM

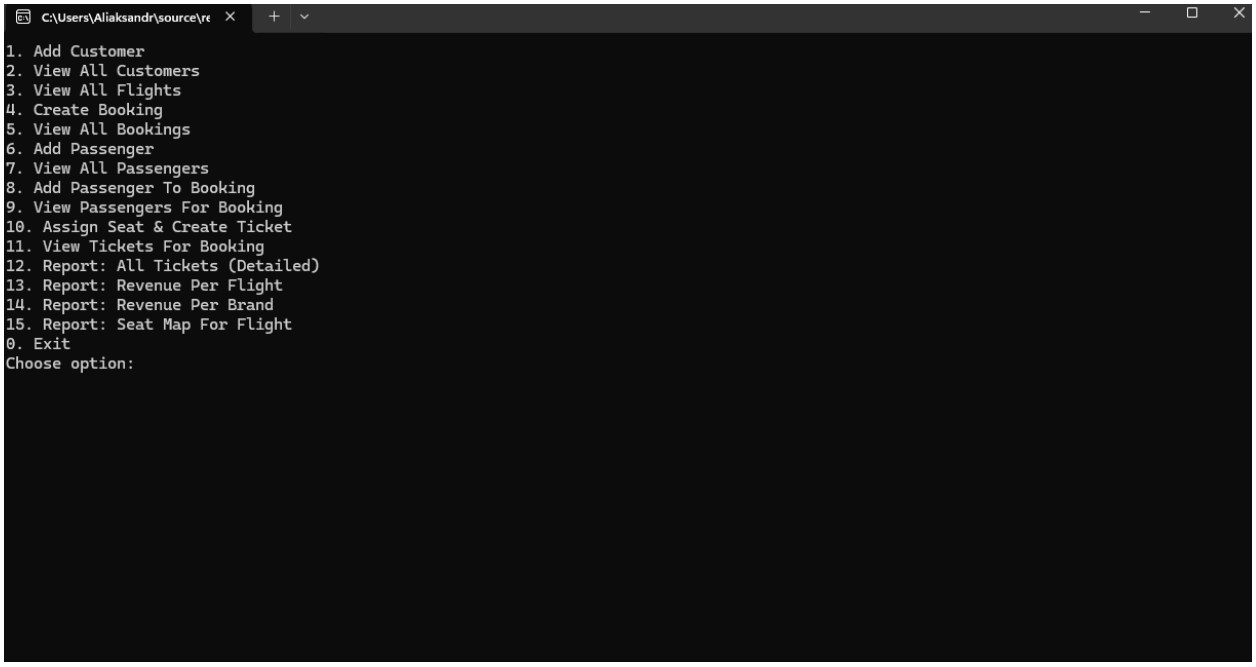
Phase 5 – C# Console Application (ADO.NET MySQL) Implementation

1. Overview of the C# Application

The C# console application was developed to demonstrate full CRUD functionality and SQL interaction with the Air Condor Airline Reservation System database.
The application communicates with MySQL using **ADO.NET (MySql.Data)** and implements:

- Customer management
- Passenger management
- Flight browsing
- Booking creation
- Passenger → Booking assignment
- Seat assignment and ticket generation
- Reporting and analytics

This application proves that the relational database schema is fully functional and supports all important business operations.



```
1. Add Customer
2. View All Customers
3. View All Flights
4. Create Booking
5. View All Bookings
6. Add Passenger
7. View All Passengers
8. Add Passenger To Booking
9. View Passengers For Booking
10. Assign Seat & Create Ticket
11. View Tickets For Booking
12. Report: All Tickets (Detailed)
13. Report: Revenue Per Flight
14. Report: Revenue Per Brand
15. Report: Seat Map For Flight
0. Exit
Choose option:
```

2. Architecture of the Application

The project uses a **service-based architecture**.
Each entity has its own service:

- `CustomerService`
- `PassengerService`
- `FlightService`
- `BookingService`
- `BookingPassengerService`
- `TicketService`
- `ReportService`
- `Database` (connection provider)

This makes the project modular, easy to maintain, and consistent with professional design practices.

3. Database Connection Layer

All SQL operations use a shared connection from the `Database` class:

```
public static MySqlConnection GetConnection()
{
    return new MySqlConnection(
        "server=localhost;user=root;password=;database=airline");
}
```

This ensures reuse, better organization, and centralized configuration.

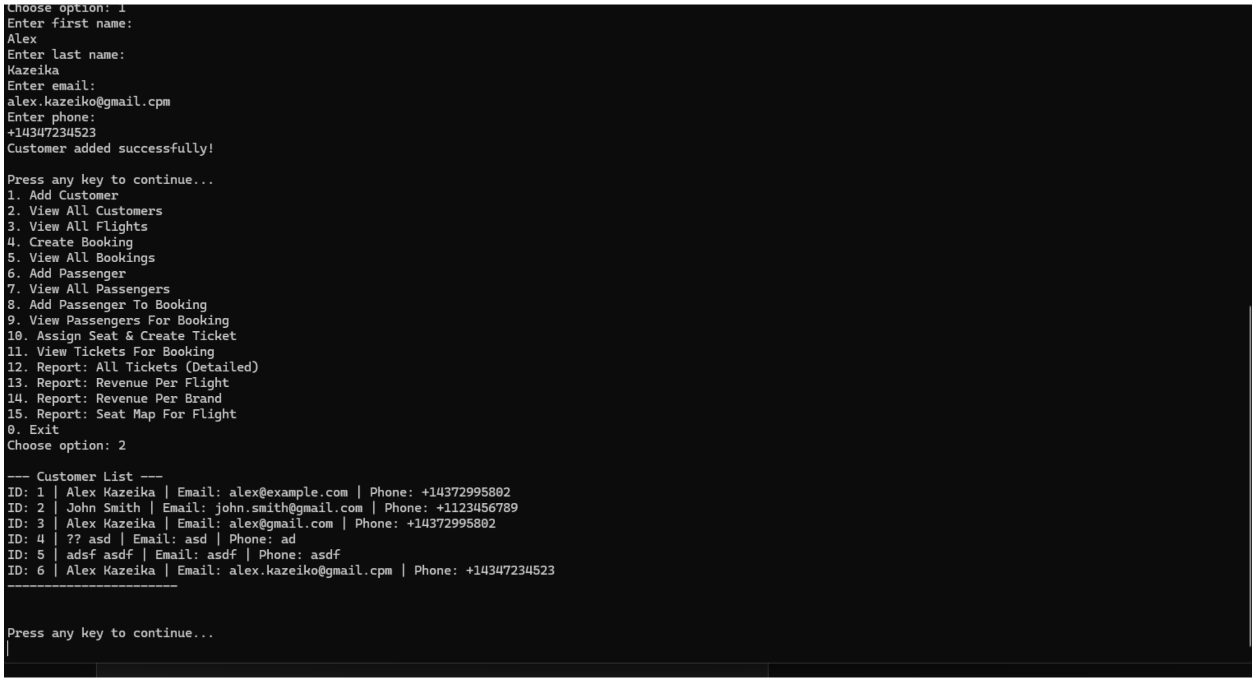
4. Customer Management

Create Customer

The application asks for first name, last name, email, and phone, then inserts into the **Customer** table.

View All Customers

Displays every row from Customer with formatted output.



```
Choose option: 1
Enter first name:
Alex
Enter last name:
Kazeika
Enter email:
alex.kazeiko@gmail.cpm
Enter phone:
+14347234523
Customer added successfully!

Press any key to continue...
1. Add Customer
2. View All Customers
3. View All Flights
4. Create Booking
5. View All Bookings
6. Add Passenger
7. View All Passengers
8. Add Passenger To Booking
9. View Passengers For Booking
10. Assign Seat & Create Ticket
11. View Tickets For Booking
12. Report: All Tickets (Detailed)
13. Report: Revenue Per Flight
14. Report: Revenue Per Brand
15. Report: Seat Map For Flight
0. Exit
Choose option: 2

--- Customer List ---
ID: 1 | Alex Kazeika | Email: alex@example.com | Phone: +14372995802
ID: 2 | John Smith | Email: john.smith@gmail.com | Phone: +1123456789
ID: 3 | Alex Kazeika | Email: alex@gmail.com | Phone: +14372995802
ID: 4 | ?? asd | Email: asd | Phone: ad
ID: 5 | adsf asdf | Email: asdf | Phone: asdf
ID: 6 | Alex Kazeika | Email: alex.kazeiko@gmail.cpm | Phone: +14347234523
----------------------


Press any key to continue...
```

5. Passenger Management

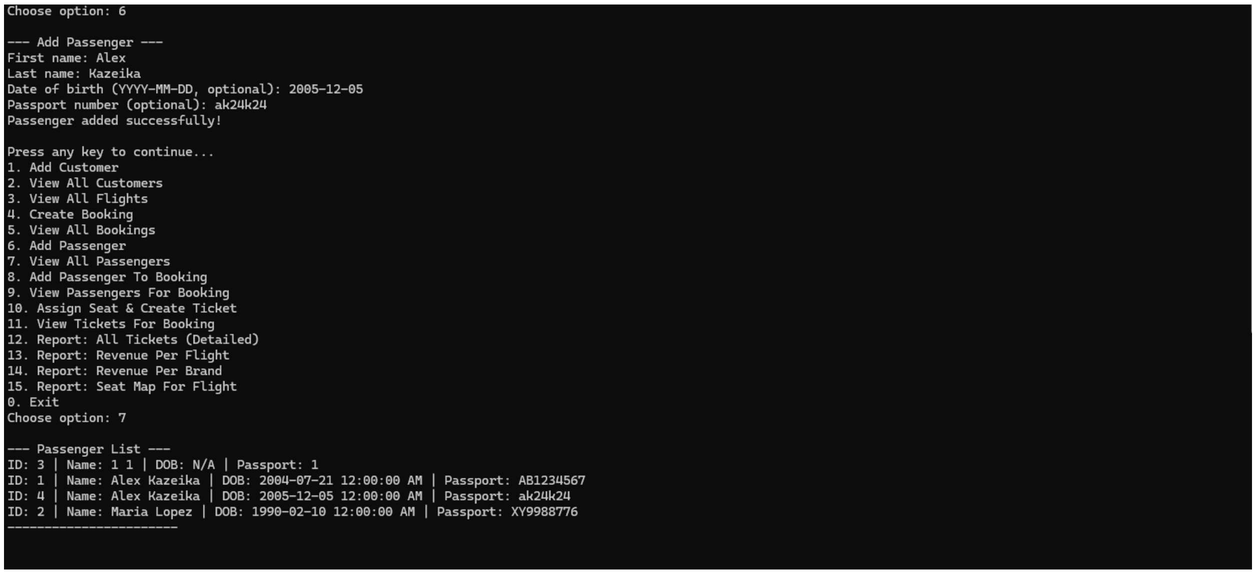
Passengers represent the travelers who board the aircraft—different from customers.

Add Passenger

Includes optional Date of Birth and Passport Number.

View All Passengers

Shows all passengers stored in the database.



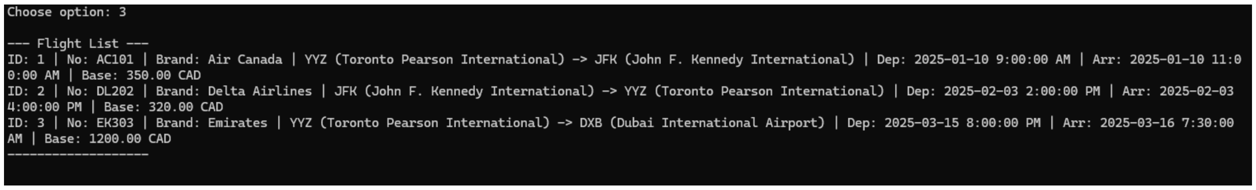
```
Choose option: 6

--- Add Passenger ---
First name: Alex
Last name: Kazeika
Date of birth (YYYY-MM-DD, optional): 2005-12-05
Passport number (optional): ak24k24
Passenger added successfully!

Press any key to continue...
1. Add Customer
2. View All Customers
3. View All Flights
4. Create Booking
5. View All Bookings
6. Add Passenger
7. View All Passengers
8. Add Passenger To Booking
9. View Passengers For Booking
10. Assign Seat & Create Ticket
11. View Tickets For Booking
12. Report: All Tickets (Detailed)
13. Report: Revenue Per Flight
14. Report: Revenue Per Brand
15. Report: Seat Map For Flight
0. Exit
Choose option: 7

--- Passenger List ---
ID: 3 | Name: 1 1 | DOB: N/A | Passport: 1
ID: 1 | Name: Alex Kazeika | DOB: 2004-07-21 12:00:00 AM | Passport: AB1234567
ID: 4 | Name: Alex Kazeika | DOB: 2005-12-05 12:00:00 AM | Passport: ak24k24
ID: 2 | Name: Maria Lopez | DOB: 1990-02-10 12:00:00 AM | Passport: XY9988776
-----------------------
```

6. Flight Browsing

The application displays all scheduled flights using:

```
SELECT FlightID, FlightNumber, BrandID, AircraftID,
DepartureAirportCode, ArrivalAirportCode, DepartureDateTime, BasePrice
```

This allows users to check available flights before creating a booking.



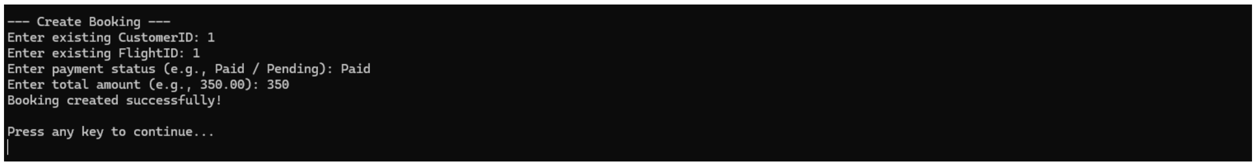
```
Choose option: 3

--- Flight List ---
ID: 1 | No: AC101 | Brand: Air Canada | YYZ (Toronto Pearson International) -> JFK (John F. Kennedy International) | Dep: 2025-01-10 9:00:00 AM | Arr: 2025-01-10 11:0
0:00 AM | Base: 350.00 CAD
ID: 2 | No: DL202 | Brand: Delta Airlines | JFK (John F. Kennedy International) -> YYZ (Toronto Pearson International) | Dep: 2025-02-03 2:00:00 PM | Arr: 2025-02-03
4:00:00 PM | Base: 320.00 CAD
ID: 3 | No: EK303 | Brand: Emirates | YYZ (Toronto Pearson International) -> DXB (Dubai International Airport) | Dep: 2025-03-15 8:00:00 PM | Arr: 2025-03-16 7:30:00
AM | Base: 1200.00 CAD
-------------------
```

7. Booking Creation

A booking links:

- A **Customer**
- A **Flight**
- A payment status
- Total amount

The application inserts a row into **Booking** with the current date and time.



```
--- Create Booking ---
Enter existing CustomerID: 1
Enter existing FlightID: 1
Enter payment status (e.g., Paid / Pending): Paid
Enter total amount (e.g., 350.00): 350
Booking created successfully!

Press any key to continue...
```

8. Adding Passengers to a Booking (BookingPassenger)

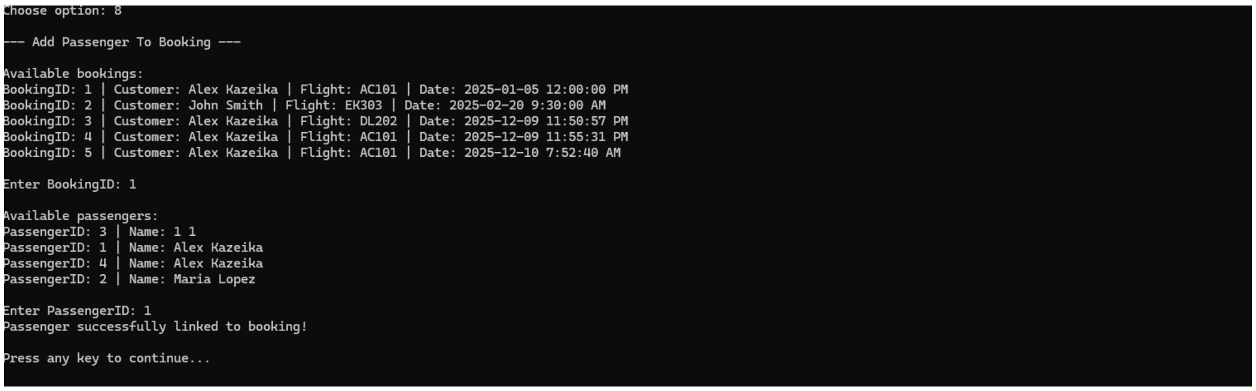
Because a booking can contain multiple passengers, the application implements:

Add Passenger to Booking

Creates a row in the **BookingPassenger** junction table.

View Passengers for a Booking

Joins Passenger + BookingPassenger for a clear view.



```
Choose option: 8

--- Add Passenger To Booking ---

Available bookings:
BookingID: 1 | Customer: Alex Kazeika | Flight: AC101 | Date: 2025-01-05 12:00:00 PM
BookingID: 2 | Customer: John Smith | Flight: EK303 | Date: 2025-02-20 9:30:00 AM
BookingID: 3 | Customer: Alex Kazeika | Flight: DL202 | Date: 2025-12-09 11:50:57 PM
BookingID: 4 | Customer: Alex Kazeika | Flight: AC101 | Date: 2025-12-09 11:55:31 PM
BookingID: 5 | Customer: Alex Kazeika | Flight: AC101 | Date: 2025-12-10 7:52:40 AM

Enter BookingID: 1

Available passengers:
PassengerID: 3 | Name: 1 1
PassengerID: 1 | Name: Alex Kazeika
PassengerID: 4 | Name: Alex Kazeika
PassengerID: 2 | Name: Maria Lopez

Enter PassengerID: 1
Passenger successfully linked to booking!

Press any key to continue...
```

9. Seat Assignment & Ticket Creation

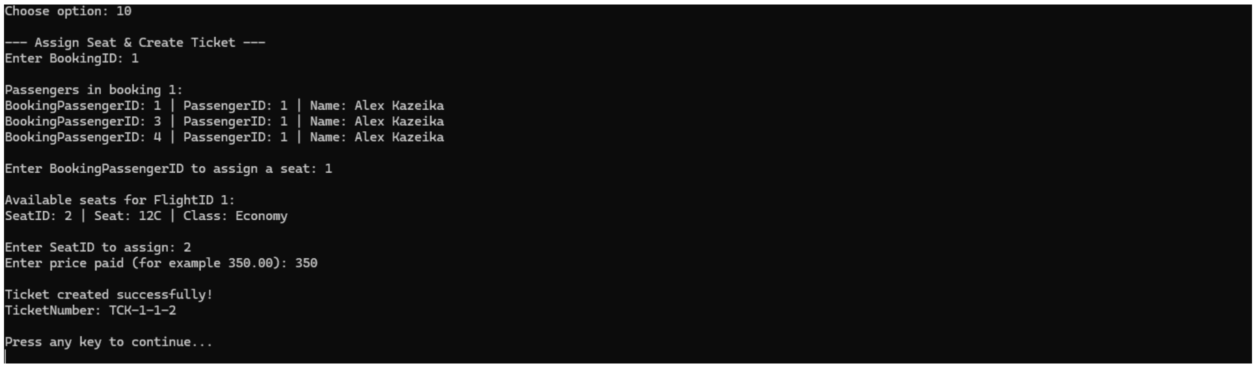
This is the core functionality of an airline system.

Step-by-step process performed by the application:

1. User selects **BookingID**
2. The system lists all passengers linked to that booking
3. It retrieves the **Aircraft** used for the booking's Flight
4. It loads all **available seats**
5. User selects **SeatID**

6. System generates a **Ticket Number** automatically
7. Inserts record into **Ticket** table

This ensures each passenger receives a unique seat and ticket.



```
Choose option: 10

--- Assign Seat & Create Ticket ---
Enter BookingID: 1

Passengers in booking 1:
BookingPassengerID: 1 | PassengerID: 1 | Name: Alex Kazeika
BookingPassengerID: 3 | PassengerID: 1 | Name: Alex Kazeika
BookingPassengerID: 4 | PassengerID: 1 | Name: Alex Kazeika

Enter BookingPassengerID to assign a seat: 1

Available seats for FlightID 1:
SeatID: 2 | Seat: 12C | Class: Economy

Enter SeatID to assign: 2
Enter price paid (for example 350.00): 350

Ticket created successfully!
TicketNumber: TCK-1-1-2

Press any key to continue...
```

10. Reporting and Analytics (ReportService)

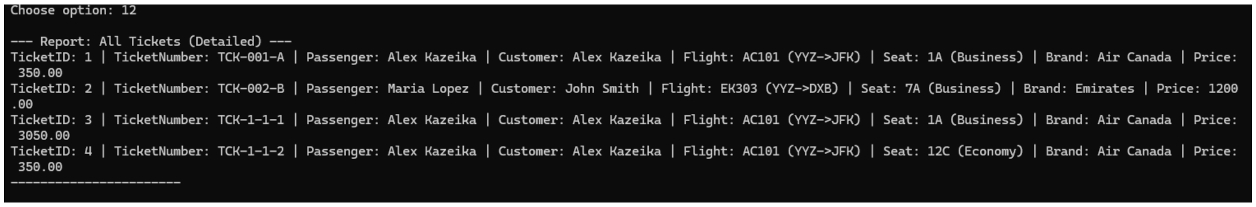
The project includes four advanced SQL reports, demonstrating real-world analytical capabilities.

10.1 Detailed Ticket Report

Shows full ticket information:

- passenger
- customer
- flight
- airports
- airline brand
- seat
- price

This uses multiple JOINs across nearly all tables.

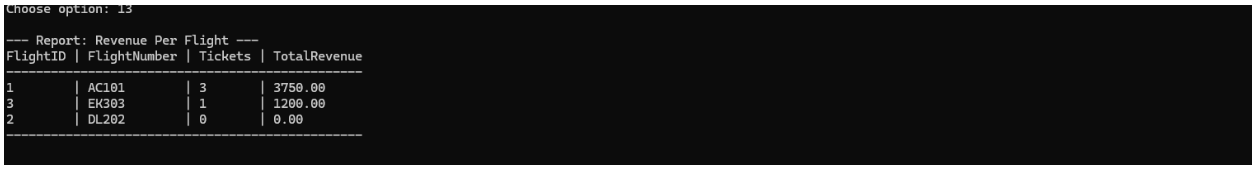

```
Choose option: 12

--- Report: All Tickets (Detailed) ---
TicketID: 1 | TicketNumber: TCK-001-A | Passenger: Alex Kazeika | Customer: Alex Kazeika | Flight: AC101 (YYZ->JFK) | Seat: 1A (Business) | Brand: Air Canada | Price:
 350.00
TicketID: 2 | TicketNumber: TCK-002-B | Passenger: Maria Lopez | Customer: John Smith | Flight: EK303 (YYZ->DXB) | Seat: 7A (Business) | Brand: Emirates | Price: 1200
.00
TicketID: 3 | TicketNumber: TCK-1-1-1 | Passenger: Alex Kazeika | Customer: Alex Kazeika | Flight: AC101 (YYZ->JFK) | Seat: 1A (Business) | Brand: Air Canada | Price:
 3050.00
TicketID: 4 | TicketNumber: TCK-1-1-2 | Passenger: Alex Kazeika | Customer: Alex Kazeika | Flight: AC101 (YYZ->JFK) | Seat: 12C (Economy) | Brand: Air Canada | Price:
 350.00
-----------------------
```

10.2 Revenue per Flight

Calculates:

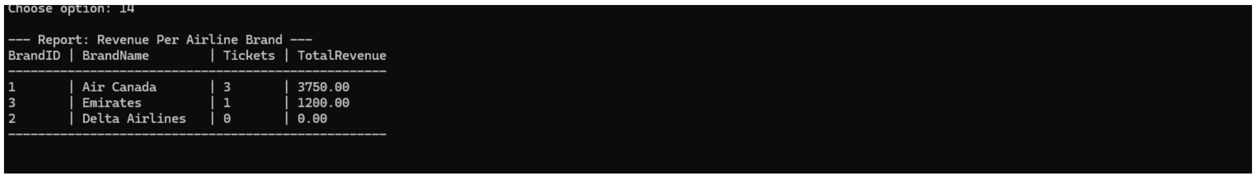
- number of sold tickets
- total revenue per flight

Uses GROUP BY and aggregated values.


```
Choose option: 13

--- Report: Revenue Per Flight ---
FlightID | FlightNumber | Tickets | TotalRevenue
------------------------------------------------
1        | AC101        | 3       | 3750.00
3        | EK303        | 1       | 1200.00
2        | DL202        | 0       | 0.00
------------------------------------------------
```

10.3 Revenue per Airline Brand

Shows the financial contribution of each brand (Air Condor, Air Condor Express, etc.)

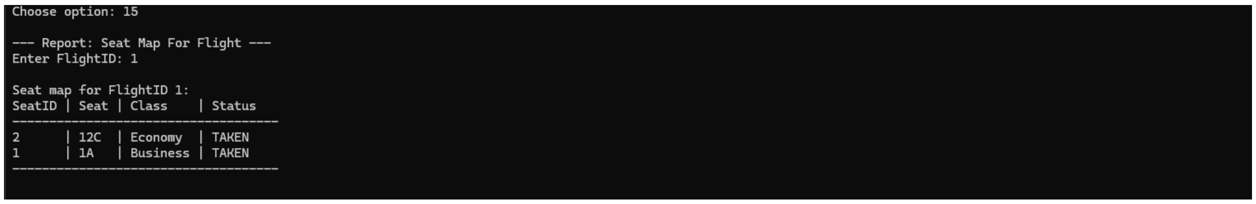

```
Choose option: 14

--- Report: Revenue Per Airline Brand ---
BrandID | BrandName         | Tickets | TotalRevenue
----------------------------------------------------
1       | Air Canada        | 3       | 3750.00
3       | Emirates          | 1       | 1200.00
2       | Delta Airlines    | 0       | 0.00
----------------------------------------------------
```

10.4 Seat Map for a Flight

Displays which seats are:

- **FREE**
- **TAKEN**

Perfect demonstration of subqueries and relational integrity.



```
Choose option: 15

--- Report: Seat Map For Flight ---
Enter FlightID: 1

Seat map for FlightID 1:
SeatID | Seat | Class    | Status
-----------------------------------
2      | 12C  | Economy  | TAKEN
1      | 1A   | Business | TAKEN
-----------------------------------
```

11. Summary

The C# console application successfully demonstrates that:

- The ERD is correct
- The relationships between tables work
- CRUD operations function across all key entities
- The system supports:
    - flight browsing
    - booking
    - passenger handling
    - seat assignment
    - ticket generation
- The database is properly normalized (3NF)
- Advanced reporting shows real analytical value

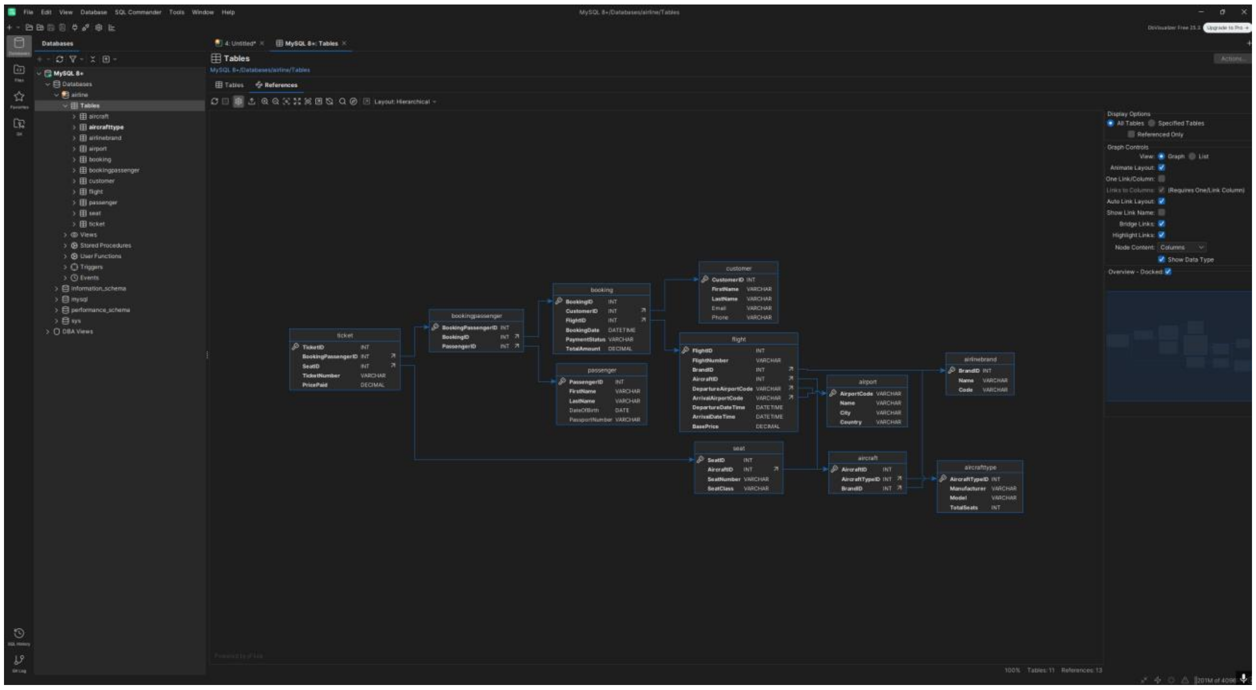
Phase 6: Airline Database Project Report

This project implements a fully functional **airline booking database** using MySQL.
The system includes tables for aircraft, flights, airports, customers, bookings, tickets, and all necessary relationships.

The goal was to design tables, enforce foreign key constraints, and populate the database with sample data.

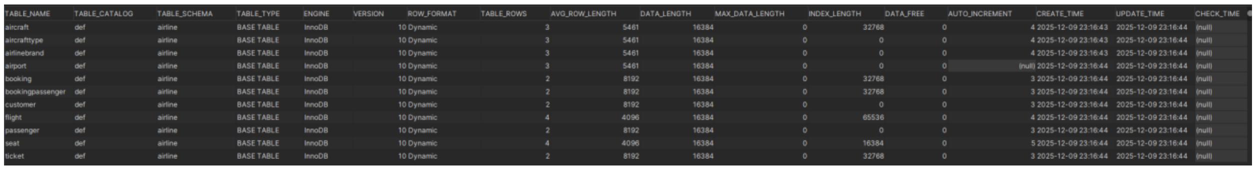
2. ER Diagram (Database Structure)

The diagram below shows the relationships between all tables (primary keys, foreign keys, and dependencies).



3.List of Created Tables

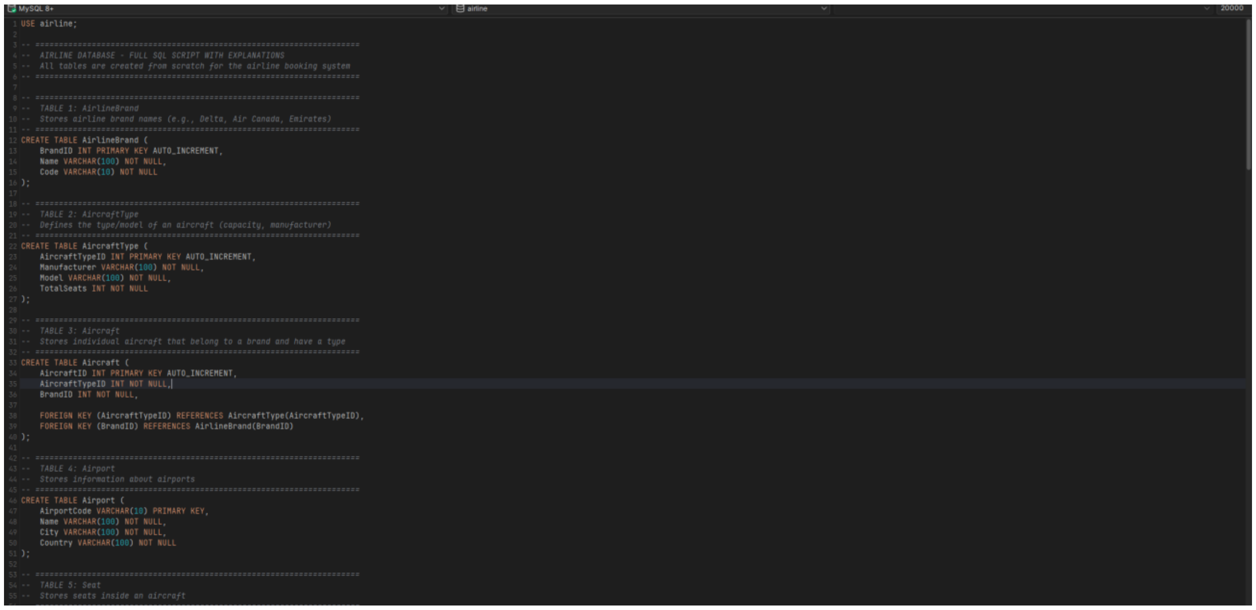
The following screenshot shows all tables created inside the MySQL *airline* schema.



TABLE_NAME	TABLE_CATALOG	TABLE_SCHEMA	TABLE_TYPE	ENGINE	VERSION	ROW_FORMAT	TABLE_ROWS	AVG_ROW_LENGTH	DATA_LENGTH	MAX_DATA_LENGTH	INDEX_LENGTH	DATA_FREE	AUTO_INCREMENT	CREATE_TIME	UPDATE_TIME	CHECK_TIME
aircraft	def	airline	BASE TABLE	InnoDB	10	Dynamic	3	5461	16384	0	32768	0	4	2025-12-09 23:16:43	2025-12-09 23:16:44	(null)
aircrafttype	def	airline	BASE TABLE	InnoDB	10	Dynamic	3	5461	16384	0	0	0	4	2025-12-09 23:16:43	2025-12-09 23:16:44	(null)
airlinebrand	def	airline	BASE TABLE	InnoDB	10	Dynamic	3	5461	16384	0	0	0	4	2025-12-09 23:16:43	2025-12-09 23:16:44	(null)
airport	def	airline	BASE TABLE	InnoDB	10	Dynamic	3	5461	16384	0	0	0	(null)	2025-12-09 23:16:44	2025-12-09 23:16:44	(null)
booking	def	airline	BASE TABLE	InnoDB	10	Dynamic	2	8192	16384	0	32768	0	3	2025-12-09 23:16:44	2025-12-09 23:16:44	(null)
bookingpassenger	def	airline	BASE TABLE	InnoDB	10	Dynamic	2	8192	16384	0	32768	0	3	2025-12-09 23:16:44	2025-12-09 23:16:44	(null)
customer	def	airline	BASE TABLE	InnoDB	10	Dynamic	2	8192	16384	0	0	0	3	2025-12-09 23:16:44	2025-12-09 23:16:44	(null)
flight	def	airline	BASE TABLE	InnoDB	10	Dynamic	4	4096	16384	0	65536	0	4	2025-12-09 23:16:44	2025-12-09 23:16:44	(null)
passenger	def	airline	BASE TABLE	InnoDB	10	Dynamic	2	8192	16384	0	0	0	3	2025-12-09 23:16:44	2025-12-09 23:16:44	(null)
seat	def	airline	BASE TABLE	InnoDB	10	Dynamic	4	4096	16384	0	16384	0	5	2025-12-09 23:16:44	2025-12-09 23:16:44	(null)
ticket	def	airline	BASE TABLE	InnoDB	10	Dynamic	2	8192	16384	0	32768	0	3	2025-12-09 23:16:44	2025-12-09 23:16:44	(null)

4.SQL Script

Below is the complete SQL script used to generate the entire database (tables + sample data).



```sql
USE airline;

-- ====================================================================
--  AIRLINE DATABASE - FULL SQL SCRIPT WITH EXPLANATIONS
--  All tables are created from scratch for the airline booking system
-- ====================================================================

-- ====================================================================
--  TABLE 1: AirlineBrand
--  Stores airline brand names (e.g., Delta, Air Canada, Emirates)
-- ====================================================================
CREATE TABLE AirlineBrand (
    BrandID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(100) NOT NULL,
    Code VARCHAR(10) NOT NULL
);

-- ====================================================================
--  TABLE 2: AircraftType
--  Defines the type/model of an aircraft (capacity, manufacturer)
-- ====================================================================
CREATE TABLE AircraftType (
    AircraftTypeID INT PRIMARY KEY AUTO_INCREMENT,
    Manufacturer VARCHAR(100) NOT NULL,
    Model VARCHAR(100) NOT NULL,
    TotalSeats INT NOT NULL
);

-- ====================================================================
--  TABLE 3: Aircraft
--  Stores individual aircraft that belong to a brand and have a type
-- ====================================================================
CREATE TABLE Aircraft (
    AircraftID INT PRIMARY KEY AUTO_INCREMENT,
    AircraftTypeID INT NOT NULL,
    BrandID INT NOT NULL,

    FOREIGN KEY (AircraftTypeID) REFERENCES AircraftType(AircraftTypeID),
    FOREIGN KEY (BrandID) REFERENCES AirlineBrand(BrandID)
);

-- ====================================================================
--  TABLE 4: Airport
--  Stores information about airports
-- ====================================================================
CREATE TABLE Airport (
    AirportCode VARCHAR(10) PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    City VARCHAR(100) NOT NULL,
    Country VARCHAR(100) NOT NULL
);

-- ====================================================================
--  TABLE 5: Seat
--  Stores seats inside an aircraft
```

5. SQL Code Used

Here you paste the full SQL code.

```sql
USE airline;

--
================================================================
=====
--  AIRLINE DATABASE - FULL SQL SCRIPT WITH EXPLANATIONS
--  All tables are created from scratch for the airline booking
system
--
================================================================
=====


--
```

```
================================================================
=====
--  TABLE 1: AirlineBrand
--  Stores airline brand names (e.g., Delta, Air Canada,
Emirates)
--
================================================================
=====
CREATE TABLE AirlineBrand (
    BrandID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(100) NOT NULL,
    Code VARCHAR(10) NOT NULL
);

--
================================================================
=====
--  TABLE 2: AircraftType
--  Defines the type/model of an aircraft (capacity,
manufacturer)
--
================================================================
=====
CREATE TABLE AircraftType (
    AircraftTypeID INT PRIMARY KEY AUTO_INCREMENT,
    Manufacturer VARCHAR(100) NOT NULL,
    Model VARCHAR(100) NOT NULL,
    TotalSeats INT NOT NULL
);

--
================================================================
=====
--  TABLE 3: Aircraft
--  Stores individual aircraft that belong to a brand and have a
type
--
================================================================
=====
CREATE TABLE Aircraft (
    AircraftID INT PRIMARY KEY AUTO_INCREMENT,
    AircraftTypeID INT NOT NULL,
    BrandID INT NOT NULL,

    FOREIGN KEY (AircraftTypeID) REFERENCES
AircraftType(AircraftTypeID),
    FOREIGN KEY (BrandID) REFERENCES AirlineBrand(BrandID)
);

--
================================================================
=====
--  TABLE 4: Airport
--  Stores information about airports
--
================================================================
=====
CREATE TABLE Airport (
    AirportCode VARCHAR(10) PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    City VARCHAR(100) NOT NULL,
    Country VARCHAR(100) NOT NULL
);

--
================================================================
=====
--  TABLE 5: Seat
--  Stores seats inside an aircraft
--
================================================================
=====
CREATE TABLE Seat (
    SeatID INT PRIMARY KEY AUTO_INCREMENT,
    AircraftID INT NOT NULL,
    SeatNumber VARCHAR(10) NOT NULL,
    SeatClass VARCHAR(20) NOT NULL,   -- e.g., Economy, Business

    FOREIGN KEY (AircraftID) REFERENCES Aircraft(AircraftID)
);
```

```
--
=================================================================
=====
--  TABLE 6: Flight
--  Stores flight information: aircraft, brand, airports, time,
price
--
=================================================================
=====
CREATE TABLE Flight (
    FlightID INT PRIMARY KEY AUTO_INCREMENT,
    FlightNumber VARCHAR(20) NOT NULL,
    BrandID INT NOT NULL,
    AircraftID INT NOT NULL,
    DepartureAirportCode VARCHAR(10) NOT NULL,
    ArrivalAirportCode VARCHAR(10) NOT NULL,
    DepartureDateTime DATETIME NOT NULL,
    ArrivalDateTime DATETIME NOT NULL,
    BasePrice DECIMAL(10,2) NOT NULL,

    FOREIGN KEY (BrandID) REFERENCES AirlineBrand(BrandID),
    FOREIGN KEY (AircraftID) REFERENCES Aircraft(AircraftID),
    FOREIGN KEY (DepartureAirportCode) REFERENCES
Airport(AirportCode),
    FOREIGN KEY (ArrivalAirportCode) REFERENCES
Airport(AirportCode)
);

--
=================================================================
=====
--  TABLE 7: Customer
--  Stores customer (person who purchases bookings)
--
=================================================================
=====
CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY AUTO_INCREMENT,
    FirstName VARCHAR(100) NOT NULL,
    LastName VARCHAR(100) NOT NULL,
    Email VARCHAR(200),
    Phone VARCHAR(20)
);

--
================================================================
=====
--  TABLE 8: Passenger
--  The passenger who actually boards a flight
--
================================================================
=====
CREATE TABLE Passenger (
    PassengerID INT PRIMARY KEY AUTO_INCREMENT,
    FirstName VARCHAR(100) NOT NULL,
    LastName VARCHAR(100) NOT NULL,
    DateOfBirth DATE,
    PassportNumber VARCHAR(50)
);

--
================================================================
=====
--  TABLE 9: Booking
--  A booking made by a customer for a flight
--
================================================================
=====
CREATE TABLE Booking (
    BookingID INT PRIMARY KEY AUTO_INCREMENT,
    CustomerID INT NOT NULL,
    FlightID INT NOT NULL,
    BookingDate DATETIME NOT NULL,
    PaymentStatus VARCHAR(30) NOT NULL,
    TotalAmount DECIMAL(10,2) NOT NULL,

    FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),
    FOREIGN KEY (FlightID) REFERENCES Flight(FlightID)
);
```

```
--
================================================================
=====
--  TABLE 10: BookingPassenger
--  Connects bookings with the passengers traveling in them
--
================================================================
=====
CREATE TABLE BookingPassenger (
    BookingPassengerID INT PRIMARY KEY AUTO_INCREMENT,
    BookingID INT NOT NULL,
    PassengerID INT NOT NULL,

    FOREIGN KEY (BookingID) REFERENCES Booking(BookingID),
    FOREIGN KEY (PassengerID) REFERENCES Passenger(PassengerID)
);

--
================================================================
=====
--  TABLE 11: Ticket
--  Stores tickets issued per passenger seat for a booking
--
================================================================
=====
CREATE TABLE Ticket (
    TicketID INT PRIMARY KEY AUTO_INCREMENT,
    BookingPassengerID INT NOT NULL,
    SeatID INT NOT NULL,
    TicketNumber VARCHAR(30) NOT NULL,
    PricePaid DECIMAL(10,2) NOT NULL,

    FOREIGN KEY (BookingPassengerID) REFERENCES
BookingPassenger(BookingPassengerID),
    FOREIGN KEY (SeatID) REFERENCES Seat(SeatID)
);

--
================================================================
```

```sql
=====
--  SAMPLE DATA INSERTS
--
================================================================
=====

-- ---------------- AirlineBrand ----------------
INSERT INTO AirlineBrand (Name, Code)
VALUES
('Air Canada', 'AC'),
('Delta Airlines', 'DL'),
('Emirates', 'EK');

-- ---------------- AircraftType ----------------
INSERT INTO AircraftType (Manufacturer, Model, TotalSeats)
VALUES
('Boeing', '737 MAX', 180),
('Airbus', 'A320', 170),
('Boeing', '777-300ER', 396);

-- ---------------- Aircraft ----------------
INSERT INTO Aircraft (AircraftTypeID, BrandID)
VALUES
(1, 1), -- Air Canada Boeing 737
(2, 2), -- Delta Airbus A320
(3, 3); -- Emirates Boeing 777

-- ---------------- Airport ----------------
INSERT INTO Airport (AirportCode, Name, City, Country)
VALUES
('YYZ', 'Toronto Pearson International', 'Toronto', 'Canada'),
('JFK', 'John F. Kennedy International', 'New York', 'USA'),
('DXB', 'Dubai International Airport', 'Dubai', 'UAE');

-- ---------------- Seat ----------------
INSERT INTO Seat (AircraftID, SeatNumber, SeatClass)
VALUES
(1, '1A', 'Business'),
(1, '12C', 'Economy'),
(2, '3D', 'Economy'),
(3, '7A', 'Business');

-- ----------------- Flight -----------------
INSERT INTO Flight
(FlightNumber, BrandID, AircraftID, DepartureAirportCode,
ArrivalAirportCode, DepartureDateTime, ArrivalDateTime,
BasePrice)
VALUES
('AC101', 1, 1, 'YYZ', 'JFK', '2025-01-10 09:00:00', '2025-01-10
11:00:00', 350.00),
('DL202', 2, 2, 'JFK', 'YYZ', '2025-02-03 14:00:00', '2025-02-03
16:00:00', 320.00),
('EK303', 3, 3, 'YYZ', 'DXB', '2025-03-15 20:00:00', '2025-03-16
07:30:00', 1200.00);

-- ----------------- Customer -----------------
INSERT INTO Customer (FirstName, LastName, Email, Phone)
VALUES
('Alex', 'Kazeika', 'alex@example.com', '+14372995802'),
('John', 'Smith', 'john.smith@gmail.com', '+1123456789');

-- ----------------- Passenger -----------------
INSERT INTO Passenger (FirstName, LastName, DateOfBirth,
PassportNumber)
VALUES
('Alex', 'Kazeika', '2004-07-21', 'AB1234567'),
('Maria', 'Lopez', '1990-02-10', 'XY9988776');

-- ----------------- Booking -----------------
INSERT INTO Booking (CustomerID, FlightID, BookingDate,
PaymentStatus, TotalAmount)
VALUES
(1, 1, '2025-01-05 12:00:00', 'Paid', 350.00),
(2, 3, '2025-02-20 09:30:00', 'Paid', 1200.00);

-- ----------------- BookingPassenger -----------------
INSERT INTO BookingPassenger (BookingID, PassengerID)
VALUES
(1, 1),
(2, 2);
```

```sql
-- ---------------- Ticket ----------------
INSERT INTO Ticket (BookingPassengerID, SeatID, TicketNumber,
PricePaid)
VALUES
(1, 1, 'TCK-001-A', 350.00),
(2, 4, 'TCK-002-B', 1200.00);
```

6. Verification

After executing the script:

- All 11 tables were successfully created
- All foreign key constraints are valid
- Sample data was inserted without errors
- Queries run correctly inside MySQL

7. Conclusion

The airline booking database was successfully designed and implemented.
All required components (aircraft, airports, customers, bookings, tickets) were integrated using relational modeling and normalization principles.
The project meets the assignment requirements and is ready for use in future application development.