**Database Systems Semester Project**

**Spring 2025**

**Project Title:**

**Airline Reservation and Management System**

**Group Members**

**Name:**

**Abeera Shahid (243589)**

**Saleha Khurram (243529)**

**Course Information:**

**Course Name:** Database Systems

**Course Code:** CS130 & CS130L

**Instructor Name:**

**Mr. Muhammad Faisal Idrees**

**Date of Submission:**

 **09-06-2025**

**Contents**

[**1.** **Abstract:** 5](#_Toc200440058)

[**2.** **Introduction:** 5](#_Toc200440059)

[**1.** **Objectives:** 6](#_Toc200440060)

[**2.** **Database Design** 7](#_Toc200440061)

[**ER Diagram:-** 8](#_Toc200440062)

[**3.** **Relational Model:** 8](#_Toc200440063)

[**4.** **Relational Schema:** 10](#_Toc200440064)

[**5.** **Queries:** 11](#_Toc200440065)

[**I.** **Data Definition Language (DDL):** 12](#_Toc200440066)

[**II.** **Data Manipulation Language (DML):** 15](#_Toc200440067)

[ **Insertion queries:** 15](#_Toc200440068)

[ **Update Queries:** 22](#_Toc200440069)

[ **Delete Queries:** 25](#_Toc200440070)

[ **SubQueries:** 26](#_Toc200440071)

[ **Joins:** 30](#_Toc200440072)

[ **Aggregate Functions:** 36](#_Toc200440073)

[Graphical User Interface (GUI) for Airline Reservation System:- 37](#_Toc200440074)

[GUI (Graphical User Interface): 37](#_Toc200440075)

[1. Main Page: 38](#_Toc200440076)

[2. Sign Up Form: 39](#_Toc200440077)

[3. Login Form: 39](#_Toc200440078)

[4. Admin Welcome Page: 40](#_Toc200440079)

[**Manages Airline:** 40](#_Toc200440080)

[**Manages Users:** 42](#_Toc200440081)

[**Manages Staff:** 42](#_Toc200440082)

[**Manages Flights:** 44](#_Toc200440083)

[**Manages Reservation:** 45](#_Toc200440084)

[**Manages Payments:** 46](#_Toc200440085)

[**Manages Aircraft:** 47](#_Toc200440086)

[**Manages Airports:** 48](#_Toc200440087)

[**Manages Routes:** 48](#_Toc200440088)

[**Manages Schedule:** 49](#_Toc200440089)

[**Manages Seat:** 51](#_Toc200440090)

[**Manages Baggage:** 52](#_Toc200440091)

[**Manages BoardingPass:** 53](#_Toc200440092)

[**Search Flights Form:** 58](#_Toc200440093)

[**Update Profile:** 58](#_Toc200440094)

[**Showing Boarding Pass:** 59](#_Toc200440095)

[6. Employee Welcome Page: 59](#_Toc200440096)

[**Update Flight Status:** 60](#_Toc200440097)

[**Passenger Table:** 61](#_Toc200440098)

[**Manages Payments:** 62](#_Toc200440099)

[**Manages Reservations:** 63](#_Toc200440100)

[**Issues Ticket:** 65](#_Toc200440101)

[**Manages Baggage:** 67](#_Toc200440102)

[**BoardingPass** 67](#_Toc200440103)

[**Reservation etc** 67](#_Toc200440104)

[**7.** **Challenges and Solutions in the Airline Reservation System (Database and GUI Combined):** 67](#_Toc200440105)

[**8.** **Conclusion:** 69](#_Toc200440106)

[**9.** **References:** 70](#_Toc200440107)

# **Abstract:**

This project presents the design and implementation of a comprehensive \*Airline Reservation and Management System\* as part of the Database Systems course. The system simulates real-world airline operations, including flight scheduling, passenger reservations, staff assignments, payment processing, seat allocation, and baggage handling. Built using Microsoft SQL Server, the project adheres to relational database principles and employs ER modeling, normalization techniques, and SQL queries (DDL & DML) to ensure data integrity, efficiency, and consistency. Through a robust schema consisting of interrelated tables and realistic sample data, the system enables efficient data retrieval, complex queries using joins and subqueries, and aggregate functions for analytical insights. The project offers a practical demonstration of how databases underpin airline systems, providing a strong foundation in database design, query development, and real-world problem solving.

# **Introduction:**

Booking airline tickets has become a routine part of modern life, especially with the rise of online services. Behind the scenes, there's a complex system handling flight schedules, passenger details, payments, seat availability, and much more. For our Database Systems semester project, we decided to build an “**Air Ticket Reservation System”** that captures all these real-world processes in a digital form.

To get started, we created an **ER diagram** to map out the different entities and how they relate to each other. From there, we moved on to the **relational model** and schema, and finally wrote all the necessary SQL queries for creating tables, inserting data, and retrieving information using joins and subqueries. We also made a simple front-end interface to show how the system could work in a real-world scenario.

This project helped us understand how databases power large-scale systems like those used by airlines, and gave us hands-on experience with designing, building, and querying relational databases.

**Purpose and Motivation:**

The goal of this project is to design a complete system that makes the ticket booking process easier and more efficient. It covers everything from creating reservations and assigning seats, to managing flights, issuing boarding passes, tracking baggage, and scheduling aircraft and staff. We used **MS SQL Server** to build and manage the database, and we followed normalization rules to keep the data clean and avoid redundancy.

**Importance of Databases in Managing Airline Data**:

Databases play a crucial role in modern flight reservation systems. They help in storing, retrieving, and managing data efficiently, ensuring data integrity and security. A well-designed database provides easy access to vital information such as flight schedules, passenger details, seat availability, payment statuses, and reservation updates. Using a database-centric approach enables airlines and travel agencies to offer reliable services to passengers, minimize errors, and enhance customer satisfaction.

**Scope of the Project:**

This Flight Reservation System project covers several critical components:

* Management of airlines and their aircraft
* Storage of airport and route information
* Tracking and managing flights
* Recording passenger details and their reservations
* Seat allocation and boarding pass issuance
* Handling payments and baggage details
* Managing airline staff assignments and schedules

The database ensures data integrity and consistency through well-defined constraints, foreign keys, and normalization. It supports various queries for data retrieval and manipulation, allowing airlines to maintain an efficient, accurate, and secure system.

# **Objectives:**

**State:**

**Objectives of the Flight Reservation System:**

* Efficient storage and retrieval of airline data
* Managing flight schedules and seat availability
* Facilitating passenger bookings and cancellations
* Managing staff and operational data

**Example:**

-- Table to store passenger information

CREATE TABLE Passenger (

PassengerID INT NOT NULL, -- Unique passenger ID

FullName VARCHAR(100) NOT NULL, -- Full name

Gender CHAR(1) NOT NULL, -- Gender (M/F)

DateOfBirth DATE NOT NULL, -- Date of birth

Email VARCHAR(100) NOT NULL, -- Email address

PhoneNum VARCHAR(20) NOT NULL, -- Contact number

Nationality VARCHAR(50) NOT NULL, -- Country of nationality

PassportNum VARCHAR(20) NOT NULL -- Passport number

);

# **Database Design**

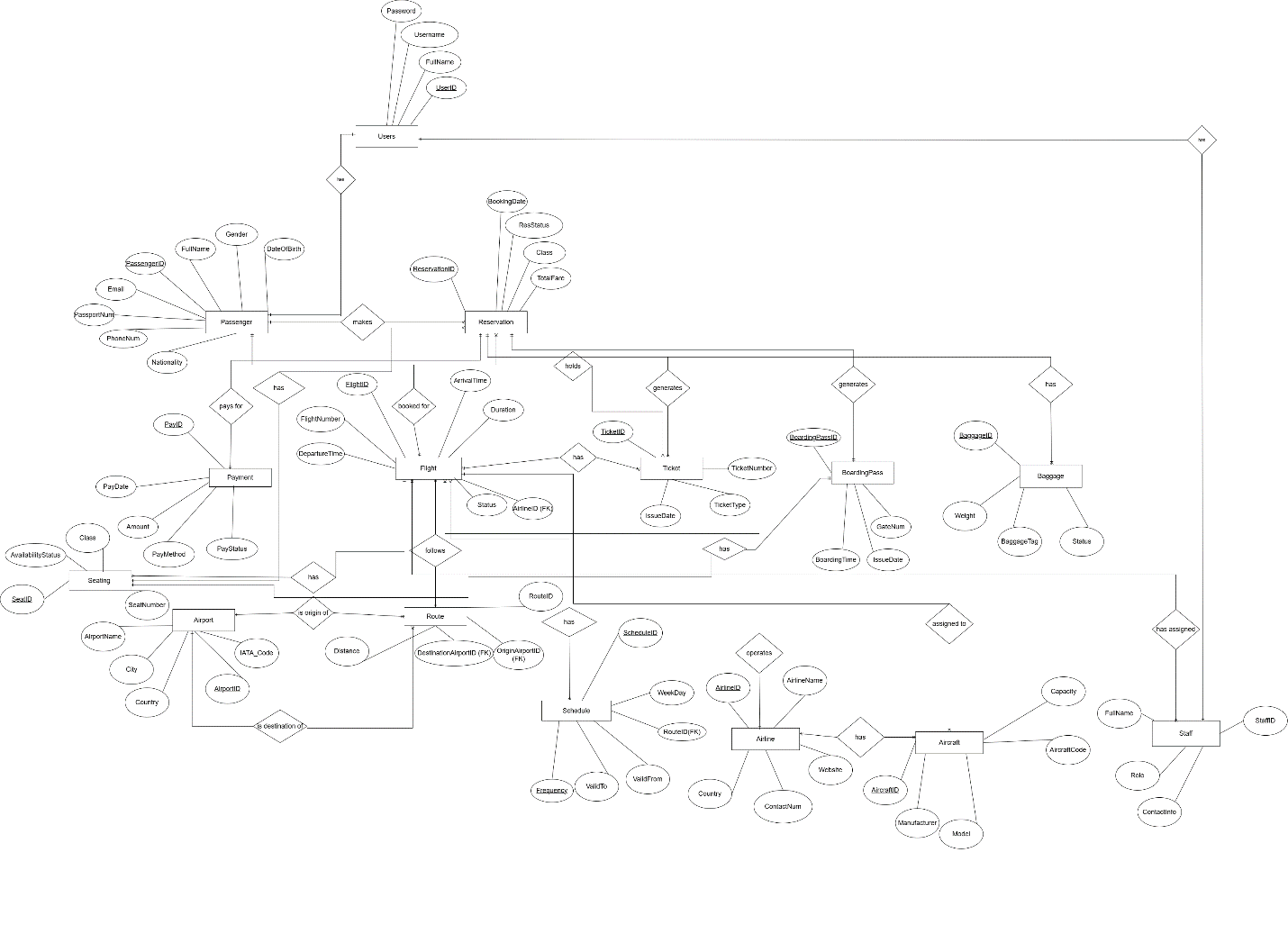
The database design for the Flight Reservation System follows the principles of the relational model to ensure data integrity, consistency, and efficient data retrieval. The design started with a conceptual model represented by an Entity-Relationship Diagram (ERD), identifying all key entities such as Airlines, Airports, Flights, Passengers, Reservations, Staff, and supporting entities like Seats, Tickets, and Baggage.

Each entity was carefully analyzed for its attributes, and relationships between entities were established based on real-world airline operations. Many-to-many relationships (like Flight–Staff assignments or Reservation–Seat assignments) were resolved by introducing junction tables to ensure data normalization and eliminate redundancy.

The logical design phase involved transforming the ERD into a relational schema that identifies primary keys (PKs), foreign keys (FKs), and constraints. The schema ensures that all data remains accurate and supports complex queries needed by the airline system.

The final physical design was implemented in SQL Server using appropriate data types, constraints, and indexing to optimize query performance and ensure data integrity.

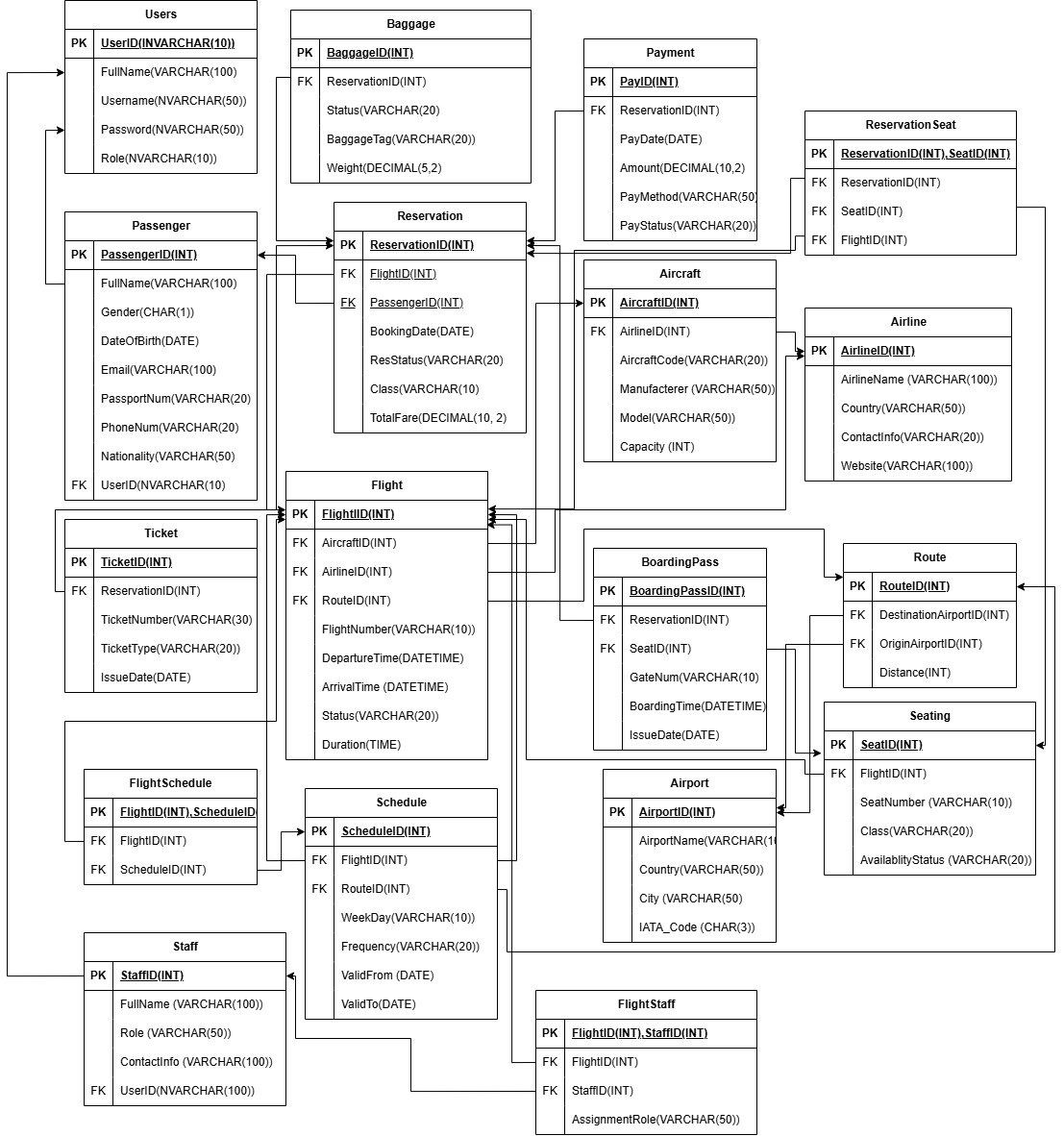
# **ER Diagram:-**



# **Relational Schema:**

* Aircraft (AircraftID, Model, Manufacturer, Capacity, AircraftCode, AirlineID)  
  PK: AircraftID  
  FK: AirlineID REFERENCES Airline(AirlineID)  
  Constraints: Model NOT NULL, Manufacturer NOT NULL, Capacity NOT NULL, AircraftCode NOT NULL, AirlineID NOT NULL
* Airline (AirlineID, AirlineName, Country, ContactNum, Website)  
  PK: AirlineID  
  Constraints: AirlineName NOT NULL, Country NOT NULL, ContactNum NOT NULL, Website NOT NULL
* Airport (AirportID, AirportName, City, Country, IATA\_Code)  
  PK: AirportID  
  Constraints: AirportName NOT NULL, City NOT NULL, Country NOT NULL, IATA\_Code NOT NULL
* Baggage (BaggageID, ReservationID, Weight, BaggageTag, Status)  
  PK: BaggageID  
  FK: ReservationID REFERENCES Reservation(ReservationID)  
  Constraints: ReservationID NOT NULL, Weight NOT NULL, BaggageTag NOT NULL, Status NOT NULL
* BoardingPass (BoardingPassID, ReservationID, IssueDate, GateNum, BoardingTime)  
  PK: BoardingPassID  
  FK: ReservationID REFERENCES Reservation(ReservationID)  
  Constraints: ReservationID NOT NULL, IssueDate NOT NULL, GateNum NOT NULL, BoardingTime NOT NULL
* Flight (FlightID, FlightNumber, Duration, AircraftID, AirlineID, RouteID)  
  PK: FlightID  
  FK: AircraftID REFERENCES Aircraft(AircraftID)  
  FK: AirlineID REFERENCES Airline(AirlineID)  
  FK: RouteID REFERENCES Route(RouteID)  
  Constraints: FlightNumber NOT NULL, Duration NOT NULL, AircraftID NOT NULL, AirlineID NOT NULL, RouteID NOT NULL
* FlightStaff (FlightID, StaffID, AssignmentRole)  
  PK: (FlightID, StaffID) - Composite Primary Key  
  FK: FlightID REFERENCES Flight(FlightID)  
  FK: StaffID REFERENCES Staff(StaffID)  
  Constraints: AssignmentRole NOT NULL
* Passenger (PassengerID, FullName, Gender, DateOfBirth, Email, PhoneNum, Nationality, PassportNum)  
  PK: PassengerID  
  Constraints: FullName NOT NULL, Gender NOT NULL, DateOfBirth NOT NULL, Email NOT NULL (and likely UNIQUE), PhoneNum NOT NULL, Nationality NOT NULL, PassportNum NOT NULL (and likely UNIQUE)
* Payment (PayID, ReservationID, PayDate, Amount, PayMethod, PayStatus)  
  PK: PayID  
  FK: ReservationID REFERENCES Reservation(ReservationID)  
  Constraints: ReservationID NOT NULL, PayDate NOT NULL, Amount NOT NULL, PayMethod NOT NULL, PayStatus NOT NULL
* Reservation (ReservationID, PassengerID, FlightID, BookingDate, ResStatus, Class, TotalFare, SeatID)  
  PK: ReservationID  
  FK: PassengerID REFERENCES Passenger(PassengerID)  
  FK: FlightID REFERENCES Flight(FlightID)  
  FK: SeatID REFERENCES Seat(SeatID)  
  Constraints: PassengerID NOT NULL, FlightID NOT NULL, BookingDate NOT NULL, ResStatus NOT NULL, Class NOT NULL, TotalFare NOT NULL, SeatID NOT NULL
* Staff (StaffID, FullName, Role, ContactInfo)  
  PK: StaffID  
  Constraints: FullName NOT NULL, Role NOT NULL, ContactInfo NOT NULL
* Ticket (TicketID, ReservationID, IssueDate, TicketNumber, TicketType)  
  PK: TicketID  
  FK: ReservationID REFERENCES Reservation(ReservationID)  
  Constraints: ReservationID NOT NULL, IssueDate NOT NULL, TicketNumber NOT NULL (and likely UNIQUE), TicketType NOT NULL
* Users (UserID, FullName, Username, Password, Role)  
  PK: UserID  
  Constraints: FullName NOT NULL, Username NOT NULL (and likely UNIQUE), Password NOT NULL, Role NOT NULL
* Seat (SeatID, FlightID, SeatNumber, Class, AvailabilityStatus)  
  PK: SeatID  
  FK: FlightID REFERENCES Flight(FlightID)  
  Constraints: FlightID NOT NULL, SeatNumber NOT NULL, Class NOT NULL, AvailabilityStatus NOT NULL
* Route (RouteID, OriginAirportID, DestinationAirportID, Distance)  
  PK: RouteID  
  FK: OriginAirportID REFERENCES Airport(AirportID)  
  FK: DestinationAirportID REFERENCES Airport(AirportID)  
  Constraints: OriginAirportID NOT NULL, DestinationAirportID NOT NULL, Distance can be NULL
* ReservationSeat (ReservationID, SeatID)  
  PK: (ReservationID, SeatID) - Composite Primary Key  
  FK: ReservationID REFERENCES Reservation(ReservationID)  
  FK: SeatID REFERENCES Seat(SeatID)
* FlightSchedule (ScheduleID, FlightID)  
  PK: ScheduleID  
  FK: FlightID REFERENCES Flight(FlightID)  
  Constraints: FlightID NOT NULL

# **Relational Model:**



# **Queries:**

## **Data Definition Language (DDL):**

DDL stands for Data Definition Language. It's used to define or modify the structure of a database, including creating, altering, or dropping tables, indexes, and other database objects.

CREATE DATABASE [Flight Reservation System];

USE [Flight Reservation System];



-- Table to store aircraft information

CREATE TABLE Aircraft (

AircraftID INT PRIMARY KEY, -- Unique ID for the aircraft

Model VARCHAR(50) NOT NULL, -- Model name of the aircraft

Manufacturer VARCHAR(50) NOT NULL, -- Manufacturer name

Capacity INT NOT NULL, -- Seating capacity

AircraftCode VARCHAR(20) NOT NULL, -- Unique aircraft code

AirlineID INT NOT NULL -- Associated airline

);

-- Table to store airline information

CREATE TABLE Airline (

AirlineID INT PRIMARY KEY, -- Unique ID for the airline

AirlineName VARCHAR(100) NOT NULL, -- Name of the airline

Country VARCHAR(50) NOT NULL, -- Country where the airline is based

ContactNum VARCHAR(20) NOT NULL, -- Contact phone number

Website VARCHAR(100) NOT NULL -- Official website

);

-- Table to store airport details

CREATE TABLE Airport (

AirportID INT PRIMARY KEY, -- Unique airport ID

AirportName VARCHAR(100) NOT NULL, -- Name of the airport

City VARCHAR(50) NOT NULL, -- City where the airport is located

Country VARCHAR(50) NOT NULL, -- Country where the airport is located

IATA\_Code CHAR(3) NOT NULL -- Standard 3-letter IATA airport code

);

-- Table to store baggage details for reservations

CREATE TABLE Baggage (

BaggageID INT PRIMARY KEY, -- Unique ID for baggage

ReservationID INT NOT NULL, -- Linked reservation

Weight DECIMAL NOT NULL, -- Weight of baggage

BaggageTag VARCHAR(20) NOT NULL, -- Tag identifier

Status VARCHAR(20) NOT NULL -- e.g., Checked, Loaded, Lost

);

-- Table to store boarding pass information

CREATE TABLE BoardingPass (

BoardingPassID INT PRIMARY KEY, -- Unique boarding pass ID

ReservationID INT NOT NULL, -- Linked reservation

IssueDate DATE NOT NULL, -- Date boarding pass was issued

GateNum VARCHAR(10) NOT NULL, -- Assigned gate

BoardingTime DATETIME NOT NULL -- Time for boarding

);

-- Table to store flight information

CREATE TABLE Flight (

FlightID INT PRIMARY KEY, -- Unique flight ID

FlightNumber VARCHAR(10) NOT NULL, -- Public flight number

Duration TIME NOT NULL, -- Duration of the flight

AircraftID INT NOT NULL, -- Assigned aircraft

AirlineID INT NOT NULL, -- Operating airline

RouteID INT NOT NULL -- Flight route

);

-- Table to assign staff members to flights

CREATE TABLE FlightStaff (

FlightID INT NOT NULL, -- Linked flight

StaffID INT NOT NULL, -- Linked staff member

AssignmentRole VARCHAR(50) NOT NULL -- Role on the flight (e.g., Pilot, Attendant)

);

-- Table to store passenger information

CREATE TABLE Passenger (

PassengerID INT NOT NULL, -- Unique passenger ID

FullName VARCHAR(100) NOT NULL, -- Full name

Gender CHAR(1) NOT NULL, -- Gender (M/F)

DateOfBirth DATE NOT NULL, -- Date of birth

Email VARCHAR(100) NOT NULL, -- Email address

PhoneNum VARCHAR(20) NOT NULL, -- Contact number

Nationality VARCHAR(50) NOT NULL, -- Country of nationality

PassportNum VARCHAR(20) NOT NULL -- Passport number

);

-- Table to store payment details

CREATE TABLE Payment (

PayID INT NOT NULL, -- Unique payment ID

ReservationID INT NOT NULL, -- Linked reservation

PayDate DATE NOT NULL, -- Date of payment

Amount DECIMAL NOT NULL, -- Amount paid

PayMethod VARCHAR(50) NOT NULL, -- Payment method (e.g., Card, Cash)

PayStatus VARCHAR(20) NOT NULL -- Status (e.g., Pending, Completed)

);

-- Table to store reservation records

CREATE TABLE Reservation (

ReservationID INT NOT NULL, -- Unique reservation ID

PassengerID INT NOT NULL, -- Passenger who made the reservation

FlightID INT NOT NULL, -- Reserved flight

BookingDate DATE NOT NULL, -- Date of booking

ResStatus VARCHAR(20) NOT NULL, -- Status (e.g., Confirmed, Cancelled)

Class VARCHAR(20) NOT NULL, -- Travel class (e.g., Economy)

TotalFare DECIMAL NOT NULL, -- Total fare for the reservation

SeatID INT NOT NULL -- Assigned seat

);

-- Table to store airline staff details

CREATE TABLE Staff (

StaffID INT NOT NULL, -- Unique staff ID

FullName VARCHAR(100) NOT NULL, -- Name of the staff member

Role VARCHAR(50) NOT NULL, -- Job title or role

ContactInfo VARCHAR(100) NOT NULL, -- Contact details

PRIMARY KEY (StaffID)

);

-- Table to store ticket details linked to reservations

CREATE TABLE Ticket (

TicketID INT NOT NULL, -- Unique ticket ID

ReservationID INT NOT NULL, -- Associated reservation

IssueDate DATE NOT NULL, -- Date ticket was issued

TicketNumber VARCHAR(30) NOT NULL, -- Ticket reference number

TicketType VARCHAR(20) NOT NULL, -- Type (e.g., One-way, Round-trip)

PRIMARY KEY (TicketID),

FOREIGN KEY (ReservationID) REFERENCES Reservation(ReservationID)

);

-- Table for system users (admin, staff, etc.)

CREATE TABLE Users (

UserID NVARCHAR(10) NOT NULL, -- User login ID

FullName NVARCHAR(100) NOT NULL, -- Full name

Username NVARCHAR(50) NOT NULL, -- Login username

Password NVARCHAR(255) NOT NULL, -- Password

Role NVARCHAR(10) NOT NULL, -- Role (e.g., Admin, Employee,Passenger)

PRIMARY KEY (UserID)

);

-- Table to store seat information for flights

CREATE TABLE Seat (

SeatID INT NOT NULL PRIMARY KEY, -- Unique seat ID

FlightID INT NOT NULL, -- Associated flight

SeatNumber VARCHAR(10) NOT NULL, -- e.g., 12A

Class VARCHAR(20) NOT NULL, -- Seat class (Economy, Business)

AvailabilityStatus VARCHAR(20) NOT NULL, -- e.g., Available, Booked

FOREIGN KEY (FlightID) REFERENCES Flight(FlightID)

);

-- Table to store flight routes

CREATE TABLE Route (

RouteID INT NOT NULL, -- Unique route ID

OriginAirportID INT NOT NULL, -- Starting airport

DestinationAirportID INT NOT NULL, -- Destination airport

Distance DECIMAL NULL, -- Distance in km/miles

PRIMARY KEY (RouteID),

FOREIGN KEY (OriginAirportID) REFERENCES Airport(AirportID),

FOREIGN KEY (DestinationAirportID) REFERENCES Airport(AirportID)

);

-- Linking table for many-to-many relationship between Reservation and Seat

CREATE TABLE ReservationSeat (

ReservationID INT NOT NULL, -- Linked reservation

SeatID INT NOT NULL, -- Linked seat

PRIMARY KEY (ReservationID, SeatID),

FOREIGN KEY (ReservationID) REFERENCES Reservation(ReservationID),

FOREIGN KEY (SeatID) REFERENCES Seat(SeatID)

);

-- Linking table for Flights and Schedules

CREATE TABLE FlightSchedule (

ScheduleID INT NOT NULL PRIMARY KEY, -- Unique schedule ID

FlightID INT NOT NULL, -- Linked flight

FOREIGN KEY (FlightID) REFERENCES Flight(FlightID)

);



## **Data Manipulation Language (DML):**

DML commands are used to manipulate data in a database, including:

**INSERT:** Adding new data

**UPDATE:** Modifying existing data

**DELETE:** Deleting data

### **Insertion queries:**

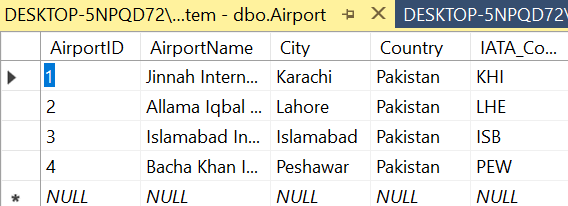
--Insert query for Airport

INSERT INTO Airport (AirportID, AirportName, City, Country, IATA\_Code) VALUES

(1, 'Jinnah International Airport', 'Karachi', 'Pakistan', 'KHI'),

(2, 'Allama Iqbal International Airport', 'Lahore', 'Pakistan', 'LHE'),

(3, 'Islamabad International Airport', 'Islamabad', 'Pakistan', 'ISB');



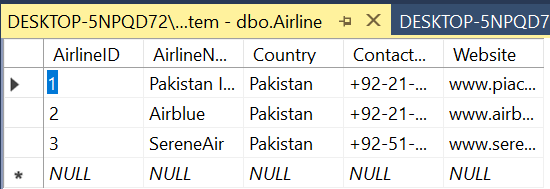
--Insert query for Airline

INSERT INTO Airline (AirlineID, AirlineName, Country, ContactNum, Website) VALUES

(1, 'Pakistan International Airlines', 'Pakistan', '+92-21-111-786-786', 'www.piac.com.pk'),

(2, 'Airblue', 'Pakistan', '+92-21-111-247-258', 'www.airblue.com'),

(3, 'SereneAir', 'Pakistan', '+92-51-111-737-363', 'www.sereneair.com');



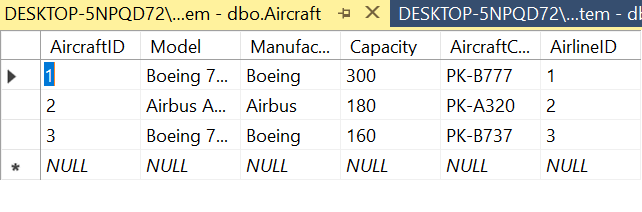
--Insert query for Aircraft

INSERT INTO Aircraft (AircraftID, Model, Manufacturer, Capacity, AircraftCode, AirlineID) VALUES

(1, 'Boeing 777', 'Boeing', 300, 'PK-B777', 1),

(2, 'Airbus A320', 'Airbus', 180, 'PK-A320', 2),

(3, 'Boeing 737', 'Boeing', 160, 'PK-B737', 3),



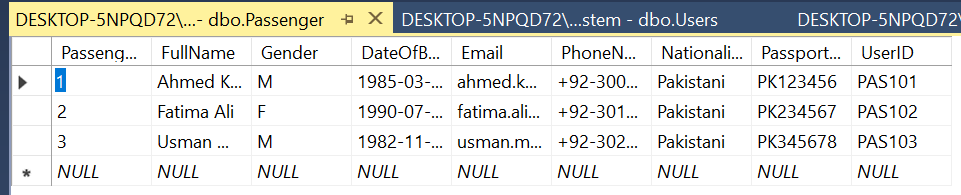
--Insert query for Passenger

INSERT INTO Passenger (PassengerID, FullName, Gender, DateOfBirth, Email, PhoneNum, Nationality, PassportNum, UserID) VALUES

(1, 'Ahmed Khan', 'M', '1985-03-15', 'ahmed.khan@example.com', '+92-300-1234567', 'Pakistani', 'PK123456', 'PAS101'),

(2, 'Fatima Ali', 'F', '1990-07-22', 'fatima.ali@example.com', '+92-301-2345678', 'Pakistani', 'PK234567', 'PAS102'),

(3, 'Usman Malik', 'M', '1982-11-05', 'usman.malik@example.com', '+92-302-3456789', 'Pakistani', 'PK345678', 'PAS103');



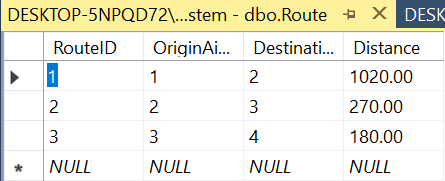
--Insert query for Route

INSERT INTO Route (RouteID, OriginAirportID, DestinationAirportID, Distance) VALUES

(1, 1, 2, 1020),

(2, 2, 3, 270),

(3, 3, 4, 180);



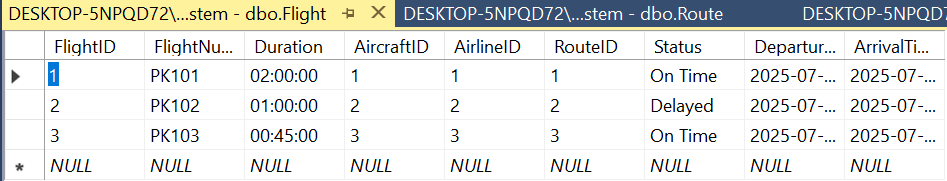
--Insert query for Flight

INSERT INTO Flight (FlightID, FlightNumber, Duration, AircraftID, DepartureTime, ArrivalTime, AirlineID, RouteID, Status) VALUES

(1, 'PK101', '02:00:00', 1, '2025-07-01 08:00:00', '2025-07-01 10:00:00', 1, 1, 'On Time'),

(2, 'PK102', '01:00:00', 2, '2025-07-02 09:00:00', '2025-07-02 10:00:00', 2, 2, 'Delayed'),

(3, 'PK103', '00:45:00', 3, '2025-07-03 10:00:00', '2025-07-03 10:45:00', 3, 3, 'On Time');



--Insert query for Reservation

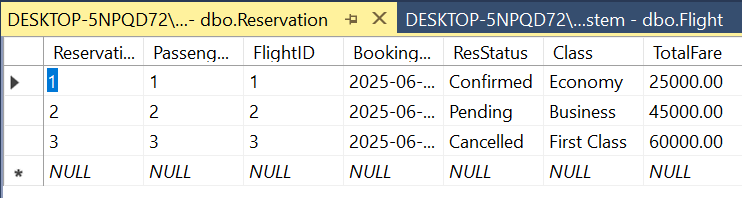
INSERT INTO Reservation (ReservationID, PassengerID, FlightID, BookingDate, ResStatus, Class, TotalFare)

VALUES

(1, 1, 1, '2025-06-01', 'Confirmed', 'Economy', 25000.00),

(2, 2, 2, '2025-06-02', 'Pending', 'Business', 45000.00),

(3, 3, 3, '2025-06-03', 'Cancelled', 'First Class', 60000.00);



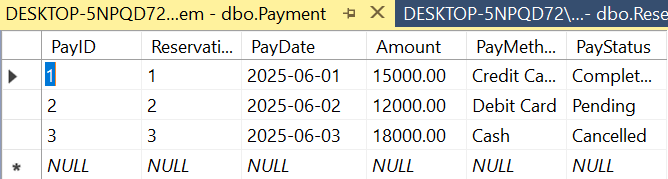
--Insert query for Payment

INSERT INTO Payment (PayID, ReservationID, Amount, PayDate, PayMethod, PayStatus) VALUES

(1, 1, 15000, '2025-06-01', 'Credit Card', 'Completed'),

(2, 2, 12000, '2025-06-02', 'Debit Card', 'Pending'),

(3, 3, 18000, '2025-06-03', 'Cash', 'Cancelled');



--Insert query for Seat

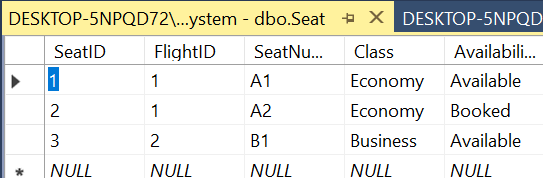
INSERT INTO Seat (SeatID, FlightID, SeatNumber, Class, AvailabilityStatus)

VALUES

(1, 1, 'A1', 'Economy', 'Available'),

(2, 1, 'A2', 'Economy', 'Booked'),

(3, 2, 'B1', 'Business', 'Available');



--Insert query for BoardingPass

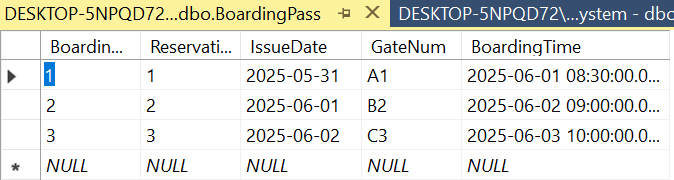
INSERT INTO BoardingPass (BoardingPassID, ReservationID, IssueDate, GateNum, BoardingTime)

VALUES

(1, 1, '2025-05-31', 'A1', '2025-06-01 08:30:00'),

(2, 2, '2025-06-01', 'B2', '2025-06-02 09:00:00'),

(3, 3, '2025-06-02', 'C3', '2025-06-03 10:00:00');

****

--Insert query for Staff

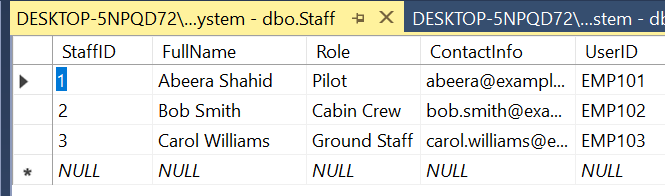
INSERT INTO Staff (StaffID, FullName, Role, ContactInfo,UserID)

VALUES

(1, 'Abeera Shahid', 'Pilot', 'abeera@example.com' ,'EMP101'),

(2, 'Bob Smith', 'Flight Attendant', 'bob.smith@example.com','EMP102'),

(3, 'Carol Williams', 'Ground Staff', 'carol.williams@example.com','EMP103');



--Insert query for FlightStaff (this is a junction table)

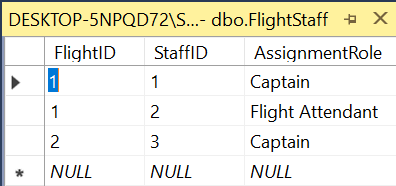
INSERT INTO FlightStaff (FlightID, StaffID, AssignmentRole)

VALUES

(1, 1, 'Captain'),

(1, 2, 'Flight Attendant'),

(2, 3, 'Captain');



--Insert query for Baggage

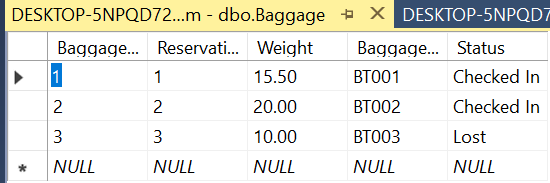
INSERT INTO Baggage (BaggageID, ReservationID, Weight, BaggageTag, Status)

VALUES

(1, 1, 15.5, 'BT001', 'Checked In'),

(2, 2, 20.0, 'BT002', 'Checked In'),

(3, 3, 10.0, 'BT003', 'Lost');



--Insert query for Ticket

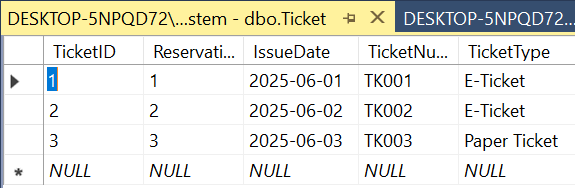
INSERT INTO Ticket (TicketID, ReservationID, IssueDate, TicketNumber, TicketType)

VALUES

(1, 1, '2025-06-01', 'TK001', 'E-Ticket'),

(2, 2, '2025-06-02', 'TK002', 'E-Ticket'),

(3, 3, '2025-06-03', 'TK003', 'Paper Ticket');



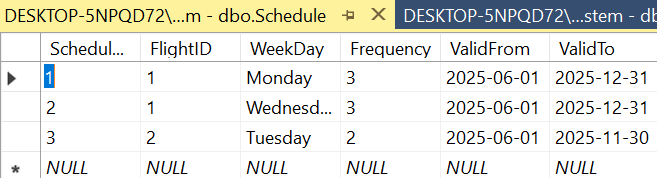
--Insert query for Schedule

INSERT INTO Schedule (ScheduleID, FlightID, Weekday, Frequency, ValidFrom, ValidTo) VALUES

(1, 1, 'Monday', 3, '2025-06-01', '2025-12-31'),

(2, 1, 'Wednesday', 3, '2025-06-01', '2025-12-31'),

(3, 2, 'Tuesday', 2, '2025-06-01', '2025-11-30');

****

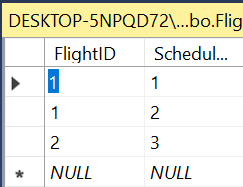
--Insert query for FlightSchedule (this is a junction table)

INSERT INTO FlightSchedule (FlightID, ScheduleID) VALUES

(1, 1),

(1, 2),

(2, 3);



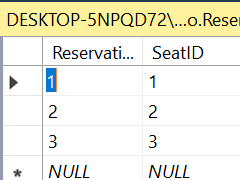
--Insert query for ReservationSeat (this is a junction table)

INSERT INTO ReservationSeat (ReservationID, SeatID) VALUES

(1, 1),

(2, 2),

(3, 3);



--Insert for Users

INSERT INTO Users (UserID, FullName, Username, Password, Role)

VALUES

('ADM101', 'Saleha Khurram', 'saleha', 'salsal123', 'Admin'),

('EMP101', 'Abeera Shahid', 'abeera', 'abeera123', 'Employee'),

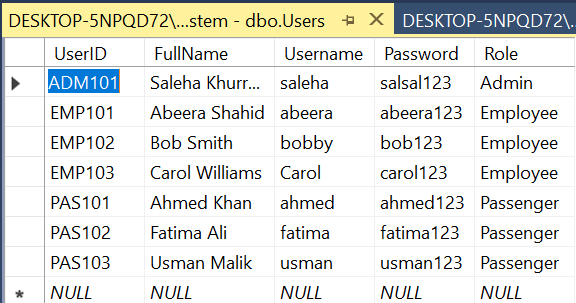
('EMP102', 'Bob Smith', 'bobby', 'bob123', 'Employee'),

('EMP103', 'Carol Williams', 'Carol', 'carol123', 'Employee'),

('PAS102', 'Ahmed Khan', 'ahmed', 'ahmed123', 'Passenger'),

('PAS103', 'Fatima Ali', 'fatima', 'fatima123', 'Passenger'),

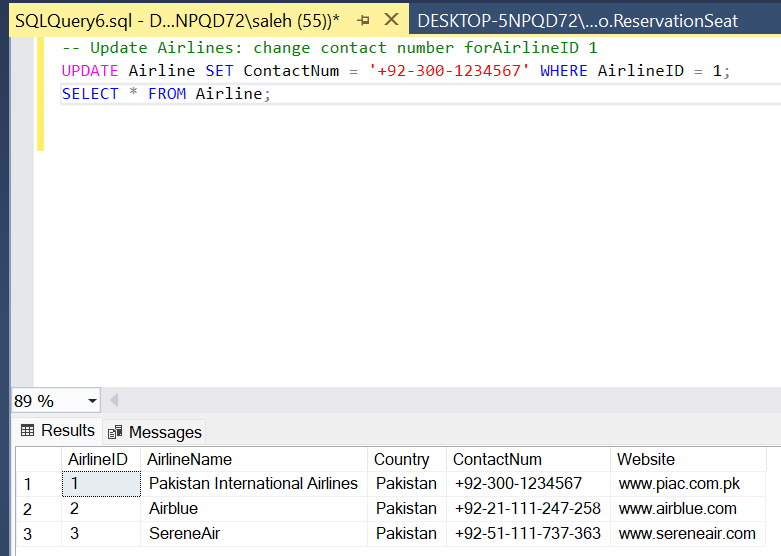
('PAS104', 'Usman Malik', 'usman', 'usman123', 'Passenger');

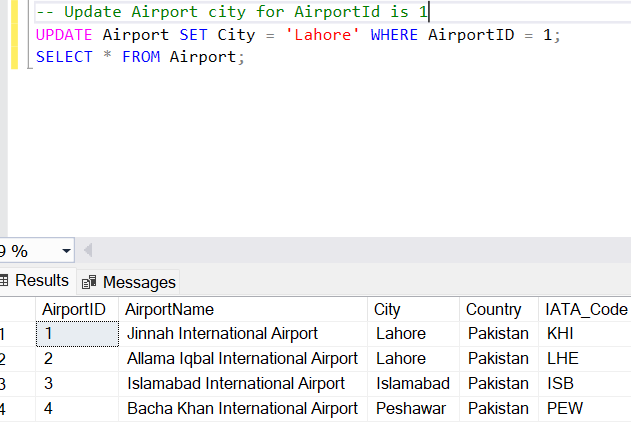


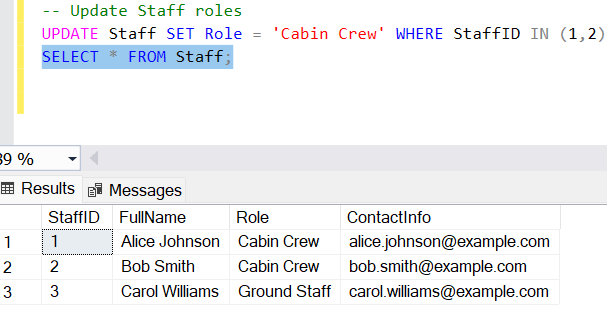
### **Update Queries:**

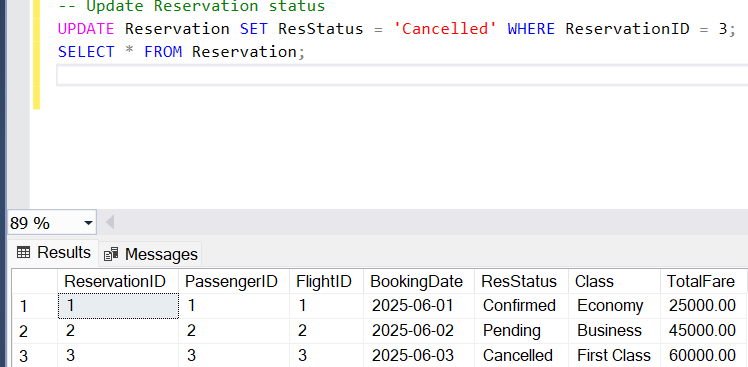
Update queries are essential for maintaining the accuracy of data in the Flight Reservation System database. They allow modification of existing records to reflect real-time changes in operations, passenger details, and reservation status.

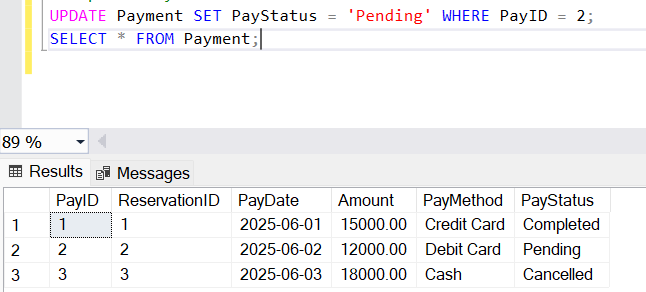
**Examples:**

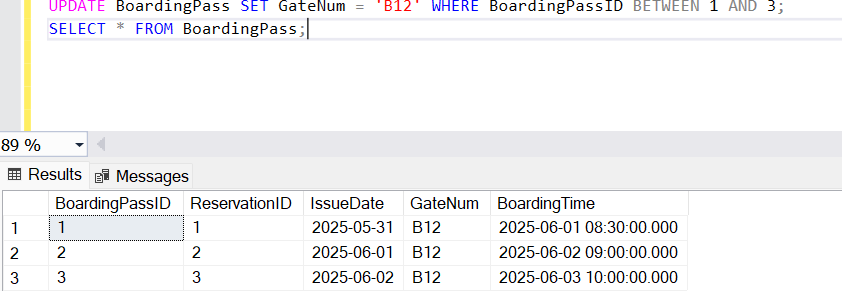
****

****

****

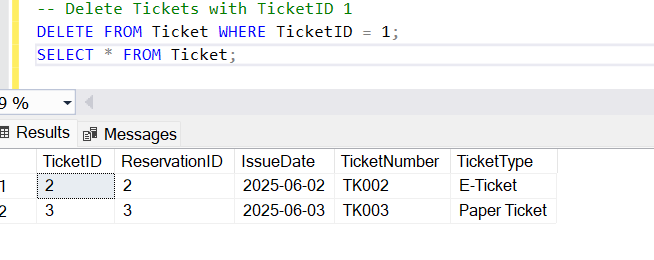
****

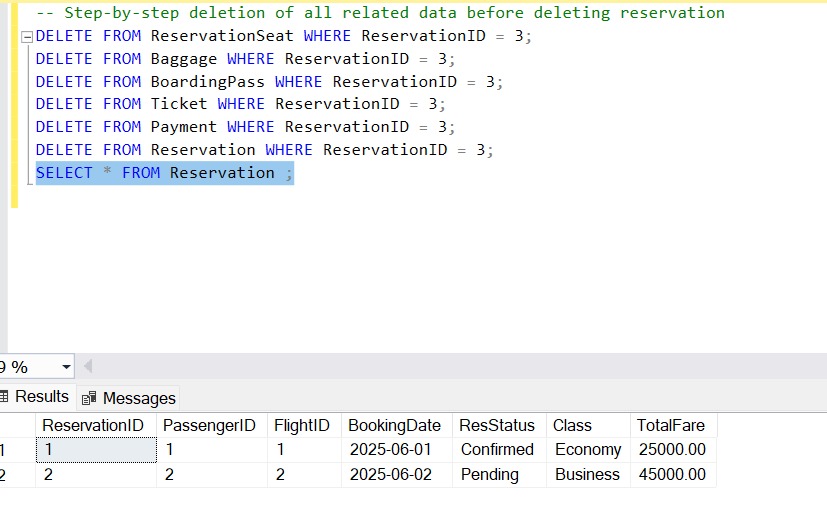
****

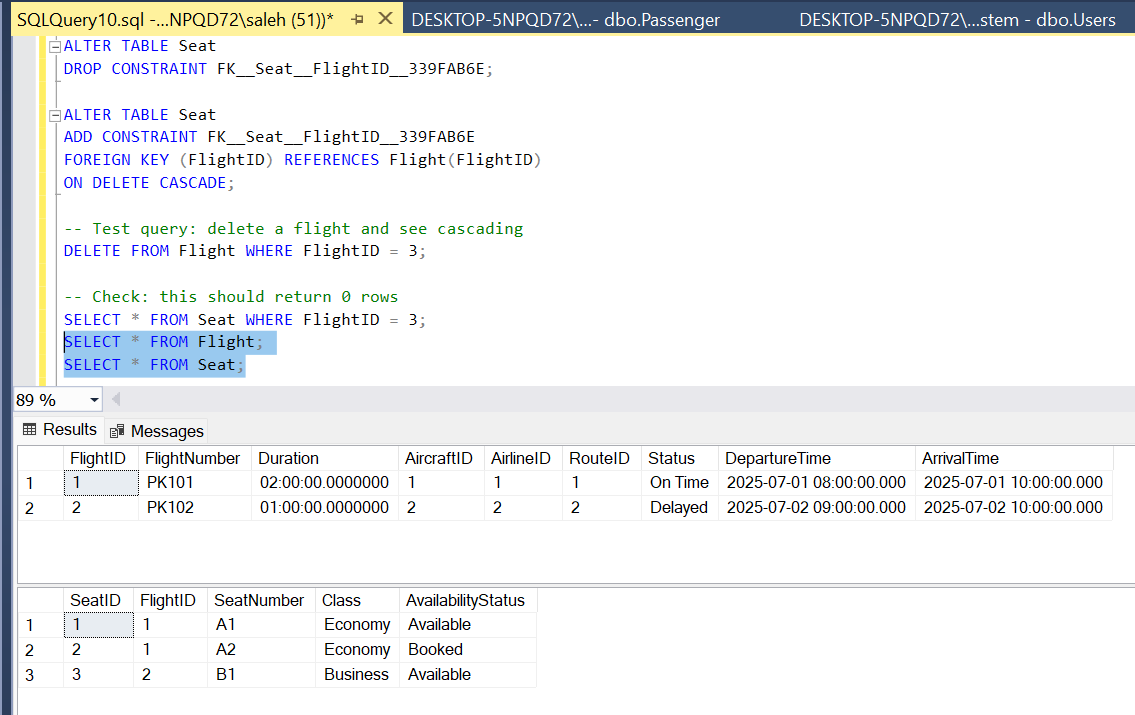
****

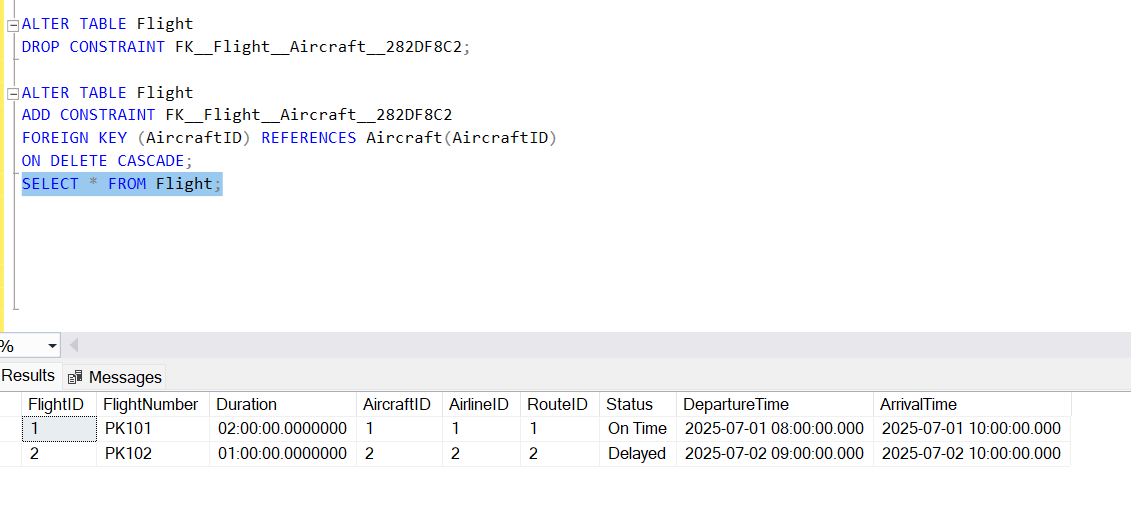
### **Delete Queries:**

Delete queries help remove outdated or incorrect data, maintaining a clean and relevant database.

****

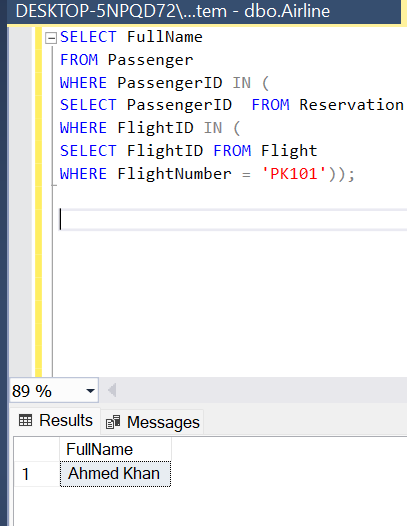


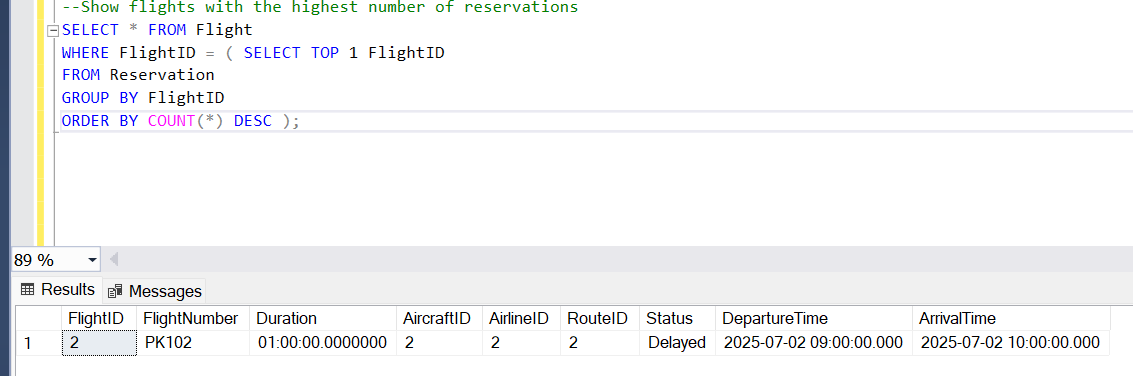
****

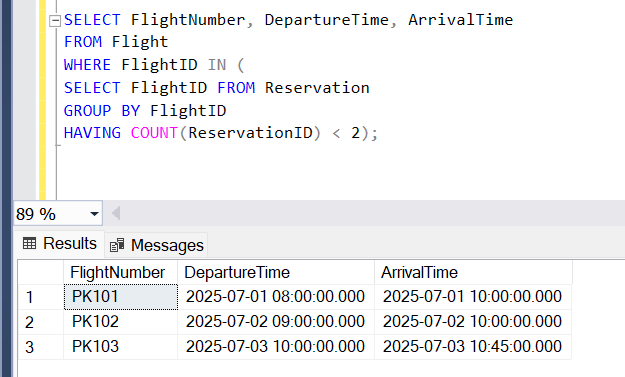
****

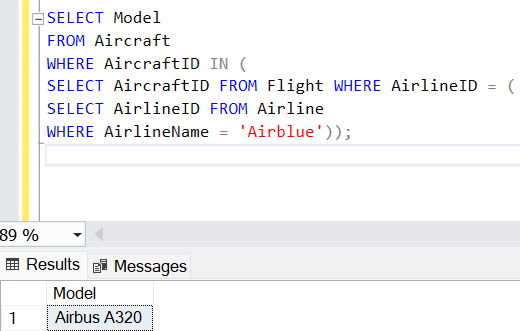
### **SubQueries:**

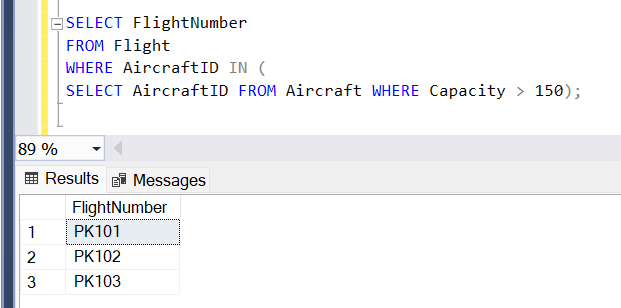
Subqueries allow complex queries by embedding a query inside another. They help extract nested data in one operation.

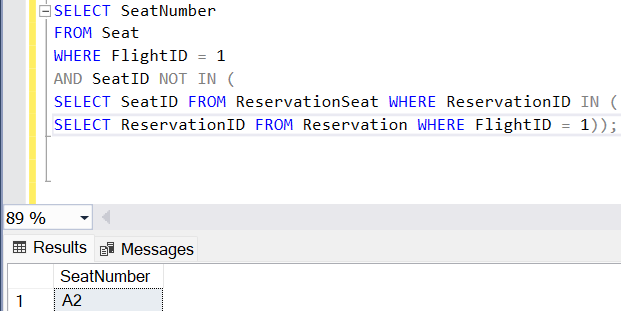
****

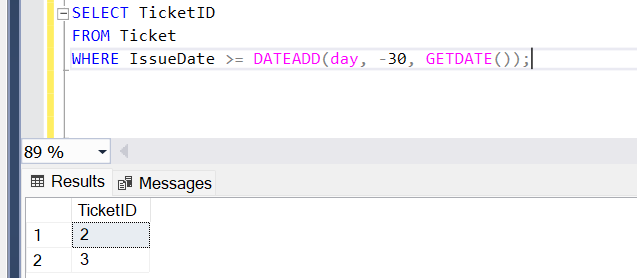
****

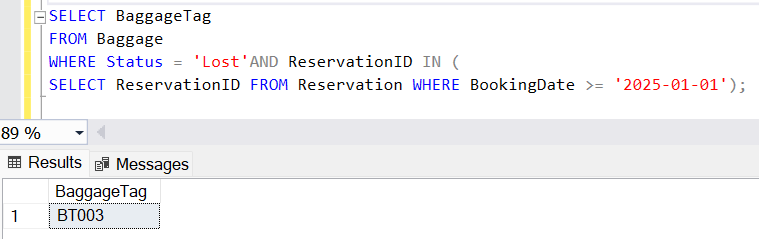
****

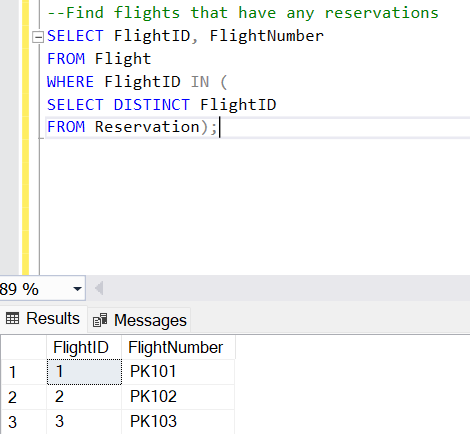
****

****

****

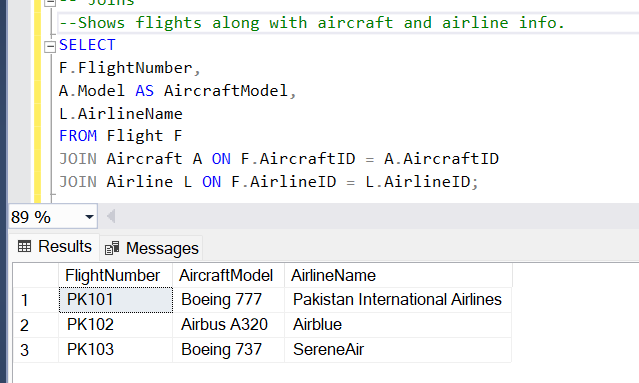
****

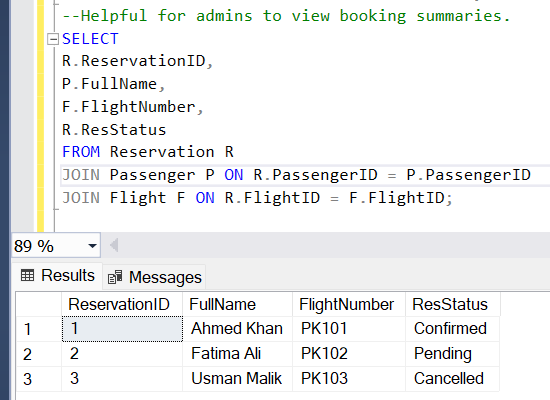
****

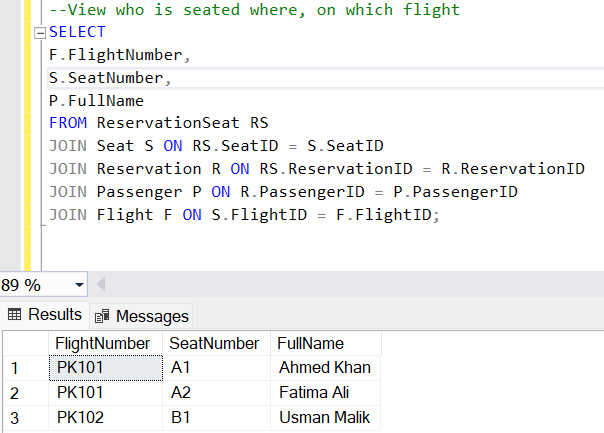
****

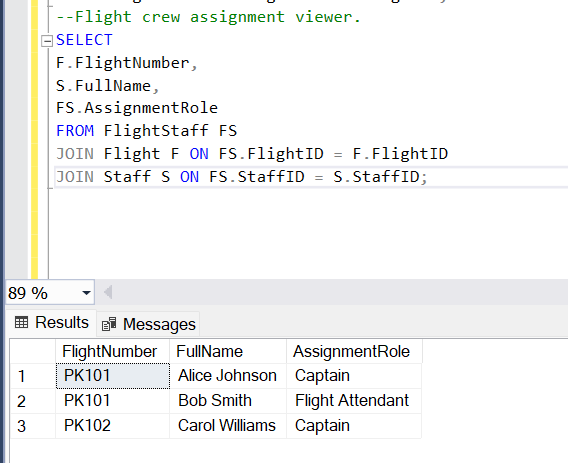
### **Joins:**

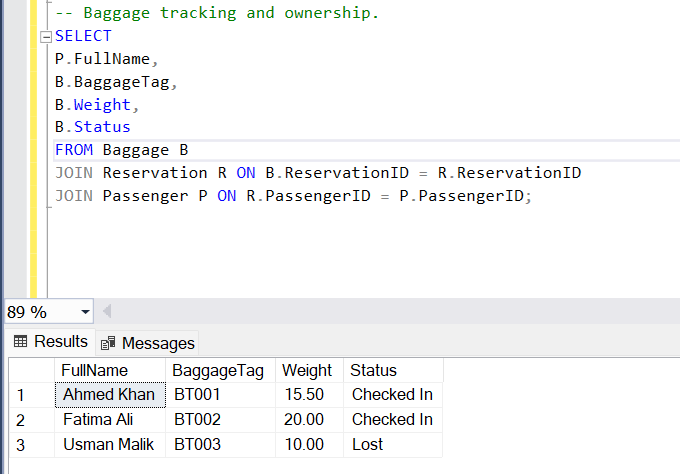
Join queries retrieve data across related tables, combining them based on common keys.

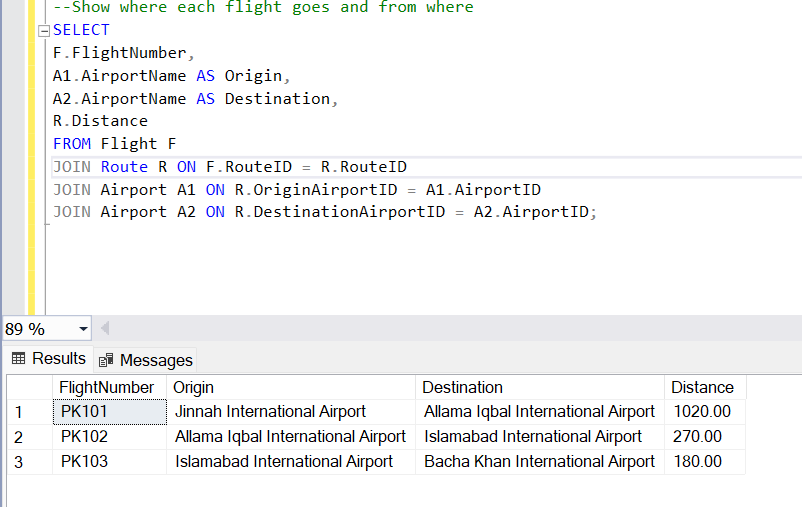
****

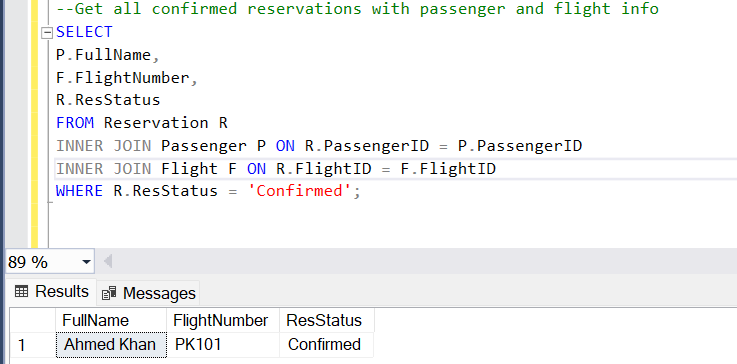
****

****

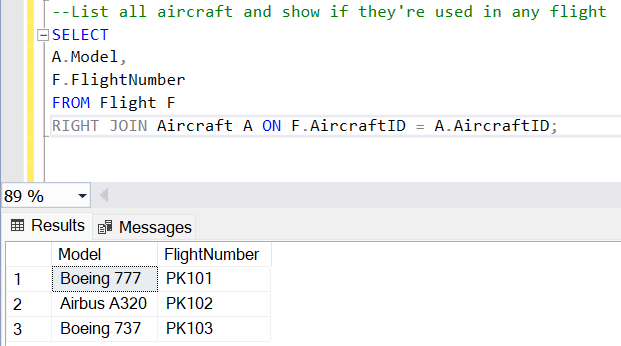
****

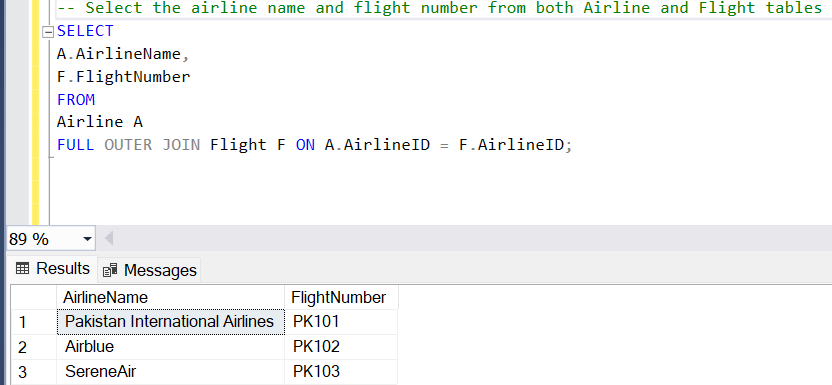
****

****

****

****

****

****

### **Aggregate Functions:**

Aggregate functions are SQL functions that perform calculations on a set of values and return a single value. Common examples include:

**1. SUM:** Calculates the total of a set of values.

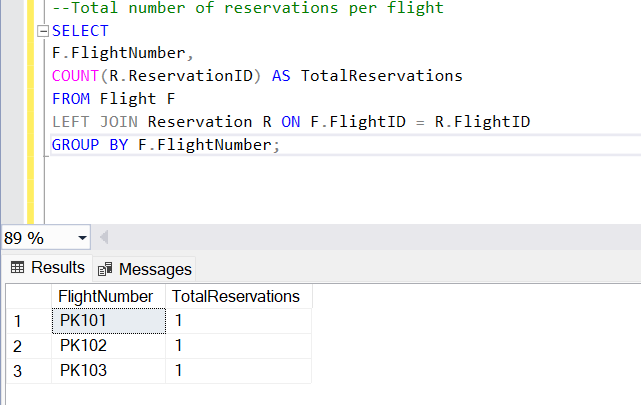
**2. AVG:** Calculates the average of a set of values.

**3. MAX:** Returns the maximum value in a set.

**4. MIN:** Returns the minimum value in a set.

**5. COUNT:** Returns the number of rows in a set.

These functions are often used in combination with GROUP BY clauses to analyze and summarize data. Aggregate functions summarize data for reports and insights.

****

# Graphical User Interface (GUI) for Airline Reservation System:-

## GUI (Graphical User Interface):

A Graphical User Interface (GUI) is a user-friendly visual interface that allows people to interact with a computer system or software using graphical elements like buttons, forms, icons, and windows rather than text-based commands.

It involves:

1. **Visual Components:**

* Buttons for actions (like Save, Delete).
* Forms for data entry.
* Tables or grids for displaying data.
* Menus and navigation bars for easy movement between sections.

1. **Interaction:**

* Users click or tap on these components instead of typing commands in a terminal.
* Behind the scenes:
* When you interact with the GUI (e.g., click “Add”), it sends commands to the system’s backend logic and database.

The database performs the requested actions (e.g., storing data), and the GUI displays updated results.

**Examples in Daily Life:**

* Windows desktop environment
* Mobile apps
* Web applications (like online shopping portals, social media platforms)

In the **flight reservation system**, the GUI includes:

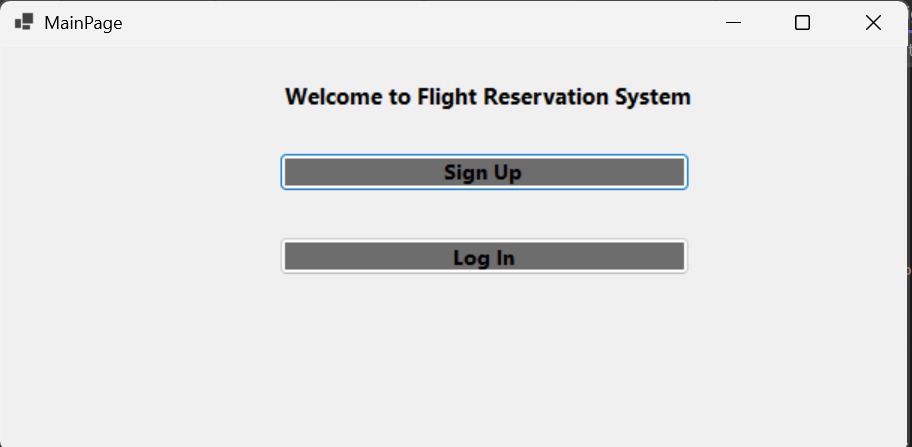
* **Login Screen:** Allows authorized users to enter credentials to access the system.
* **Dashboard:** Serves as the main hub with navigation buttons for managing flights, passengers, and reservations.
* **Booking Forms:** Enable staff members to add new flight bookings by filling out fields such as passenger name, flight number, and seat preferences.
* **Tables:** Display available flights, booked flights, and passenger information in a structured format.
* **Search and Filter Tools:** Help users find flights or reservations quickly by entering criteria like destination, date, or passenger name.
* **Edit and Delete Features:** Allow authorized staff to update or remove bookings and passenger records.
* **Reports Module:** Provides summarized data, such as the number of bookings in a day or the most popular flights.

The **GUI interacts directly with the database**. For example:  
When a staff member books a flight, the GUI sends an **INSERT query** to the reservations table in the database.  
When searching for flights, the GUI sends a **SELECT query** to the flights table and shows the results in a table.  
Editing passenger details sends an **UPDATE query** to the passengers table.  
Deleting a booking triggers a **DELETE query** in the reservations table.

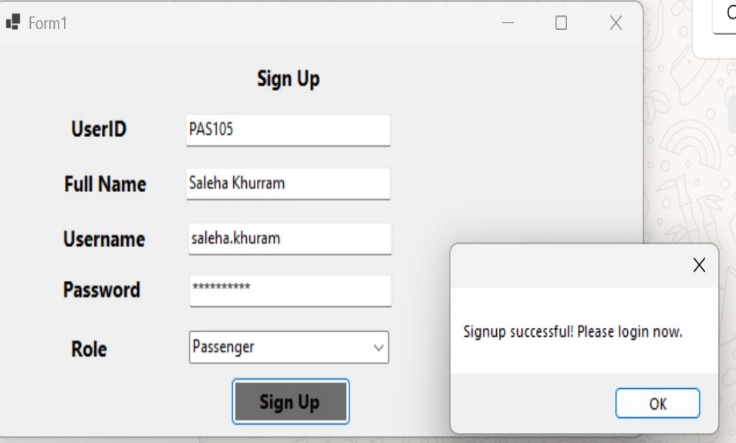
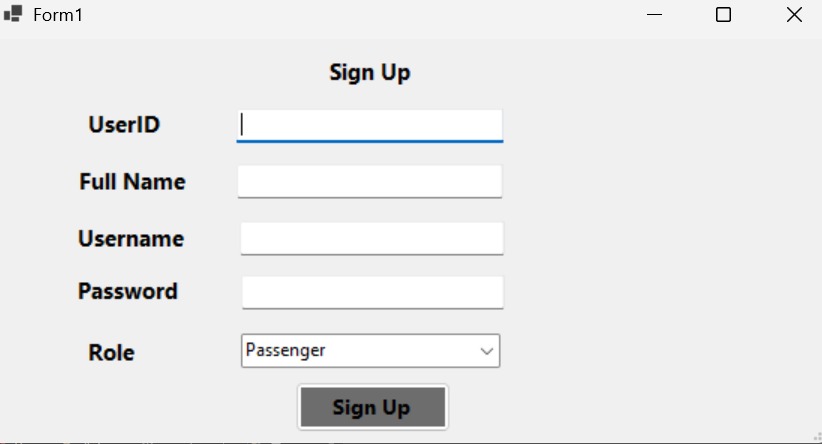
By providing an intuitive visual interface, the GUI makes it easier for staff to manage flight reservations without directly writing SQL commands. This integration of the GUI with the database ensures data consistency, security, and efficient management of flight booking and passenger information

# Main Page:

This image displays the initial "MainPage" of the Flight Reservation System. It presents two primary options to the user: "Sign Up" and "Log In." This page serves as the entry point for new users to register and for existing users to access the system

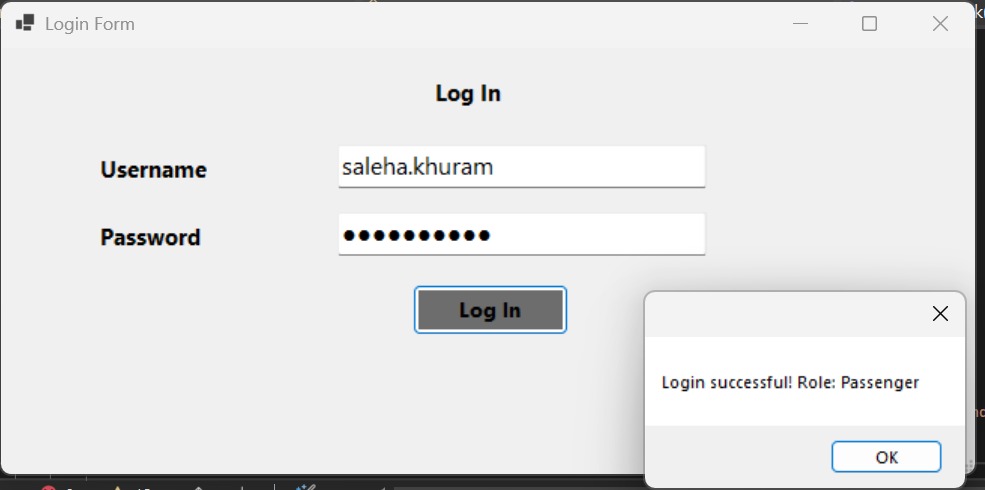


# Sign Up Form:

First time users need to sign up, once registered they can then log in. The image given below shows what it does. The "Sign Up" form (Form1) where a new user, "Saleha Khurram," is registering. The form captures essential user details such as UserID ("PAS105"), Full Name, Username ("saleha.khuram"), Password, and Role, which is set to "Passenger." Upon successful submission, a pop-up message "Signup successful! Please login now." confirms the creation of the user's account, indicating that the data has been successfully stored in the system's database.

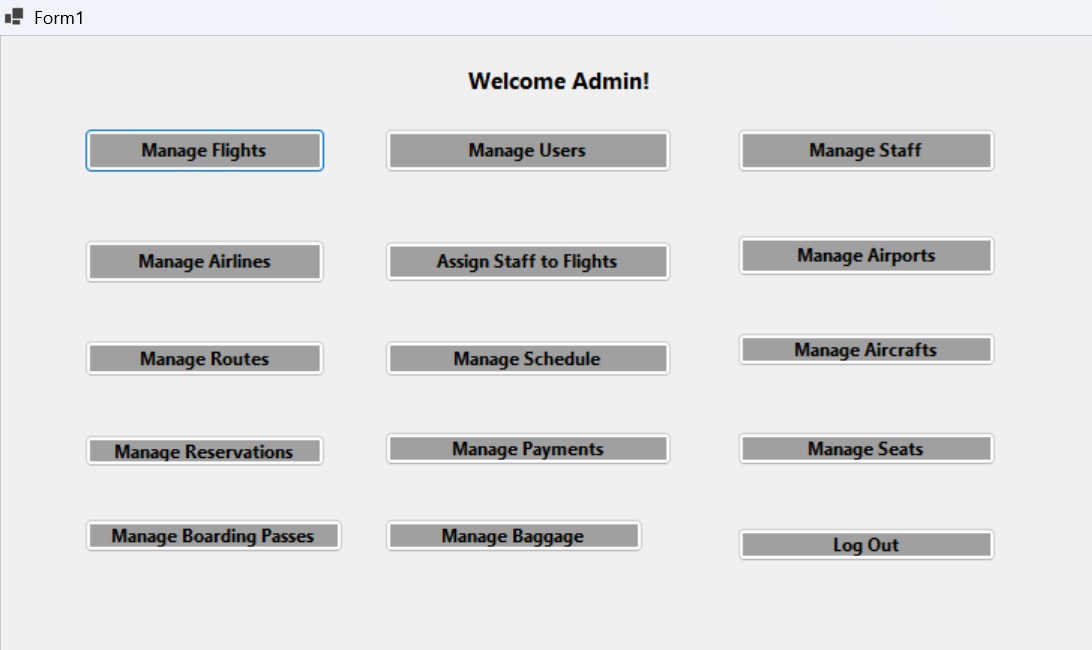
# Login Form:

This image depicts the "Login Form" where the newly registered user, "saleha.khuram," is logging in. After entering the username and password, a "Login successful! Role: Passenger" message appears, confirming successful authentication against the database and indicating the user's assigned role within the system.



# Admin Page:

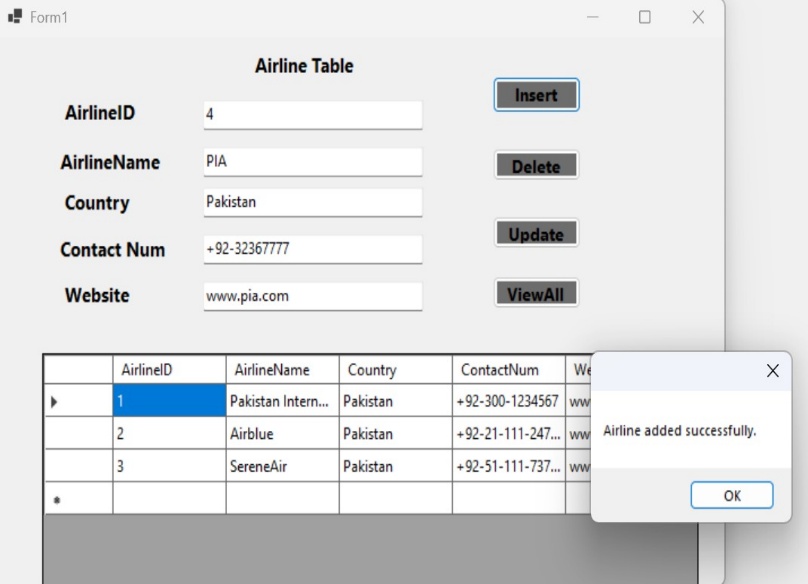
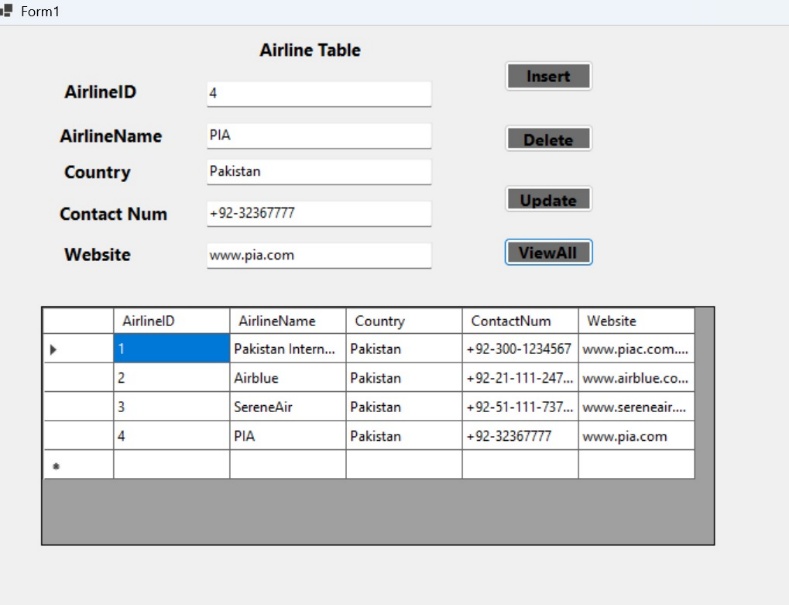
This "Welcome Admin!" page (Form1) suggests a successful login by an administrator. It presents a comprehensive set of management options, including "Manage Flights," "Manage Users," "Manage Staff," "Manage Airlines," "Assign Staff to Flights," "Manage Airports," "Manage Routes," "Manage Schedule," "Manage Aircrafts," "Manage Reservations," "Manage Payments," "Manage Seats," "Manage Boarding Passes," "Manage Baggage," and "Log Out." This page highlights the extensive administrative functionalities available for managing various aspects of the flight reservation system, all of which interact with the underlying database.



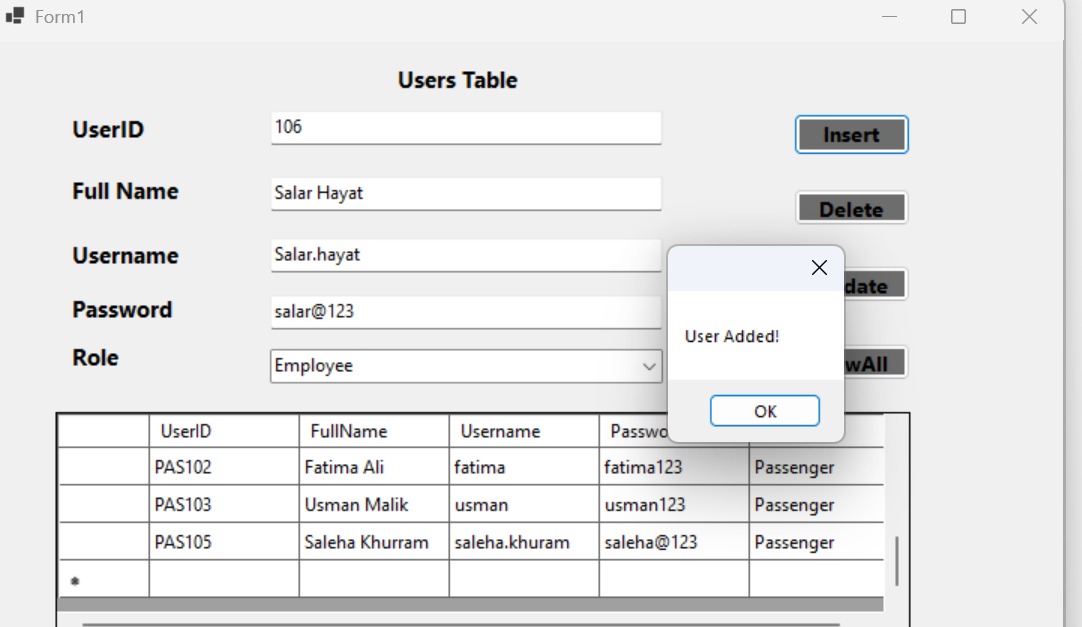
## **Manages Airline:**

Airline Table within the admin interface (Form1) is demonstrating the ability to manage airline information. An entry for "PIA" with AirlineID "4," Country "Pakistan," Contact Number "+92-32367777," and Website "www.pia.com" is being inserted. The table below displays existing airline data, including "Pakistan International Airlines" (AirlineID 1), "Airblue" (AirlineID 2), and "SereneAir" (AirlineID 3). A pop-up confirms "Airline added successfully," indicating a successful database operation.

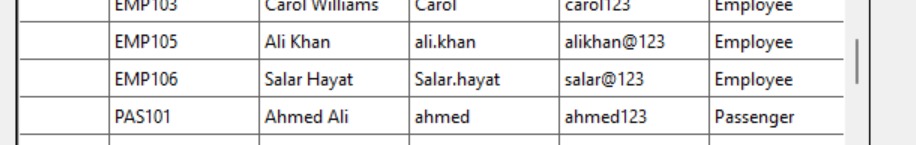
* Like this, insertions in other tables can also be performed.

**For instance**:

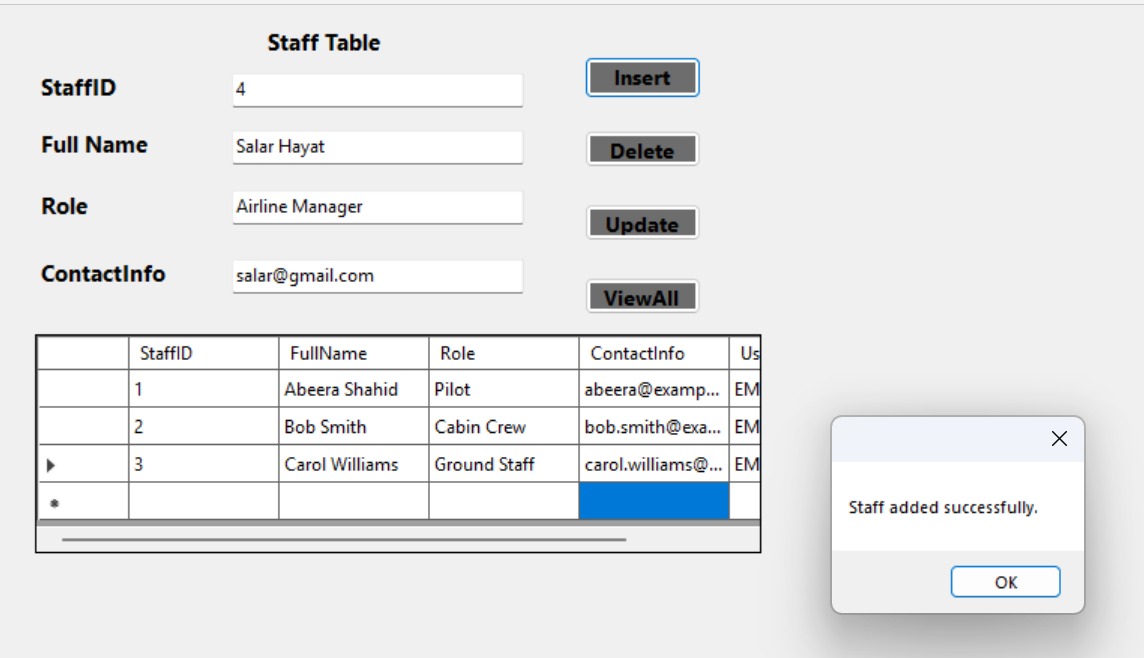
## **Manages Users:**



**Output:**



## **Manages Staff:**

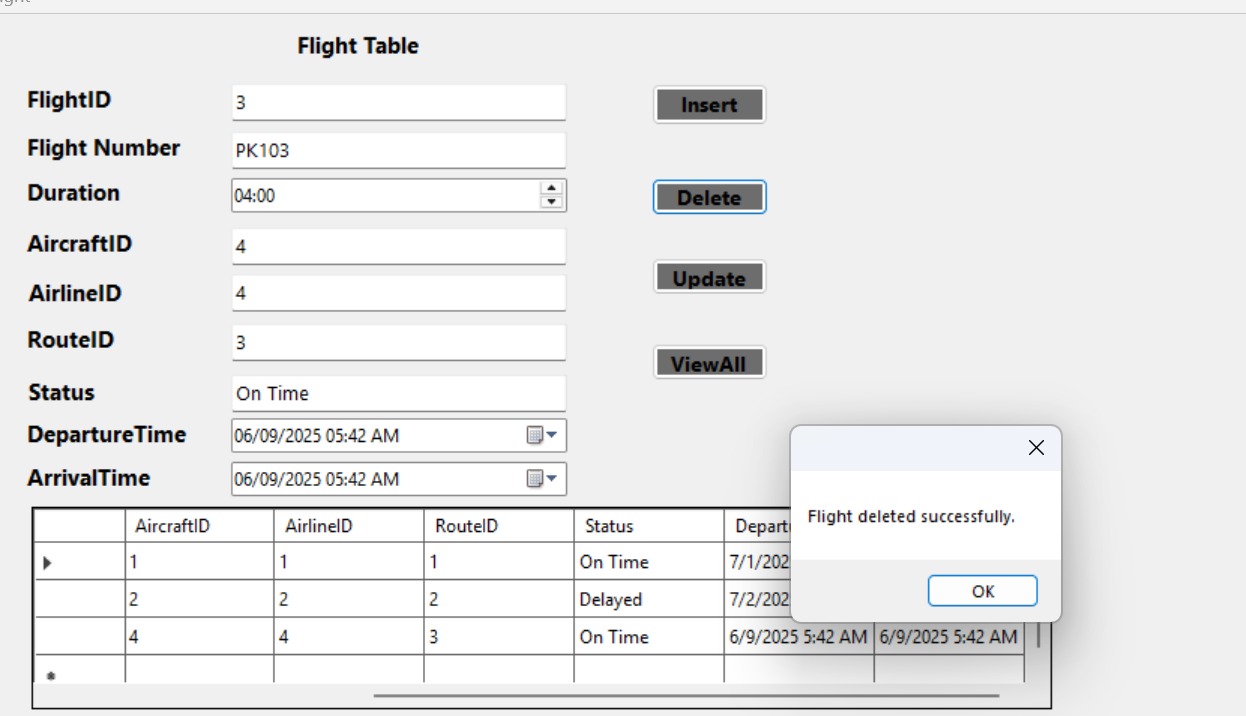


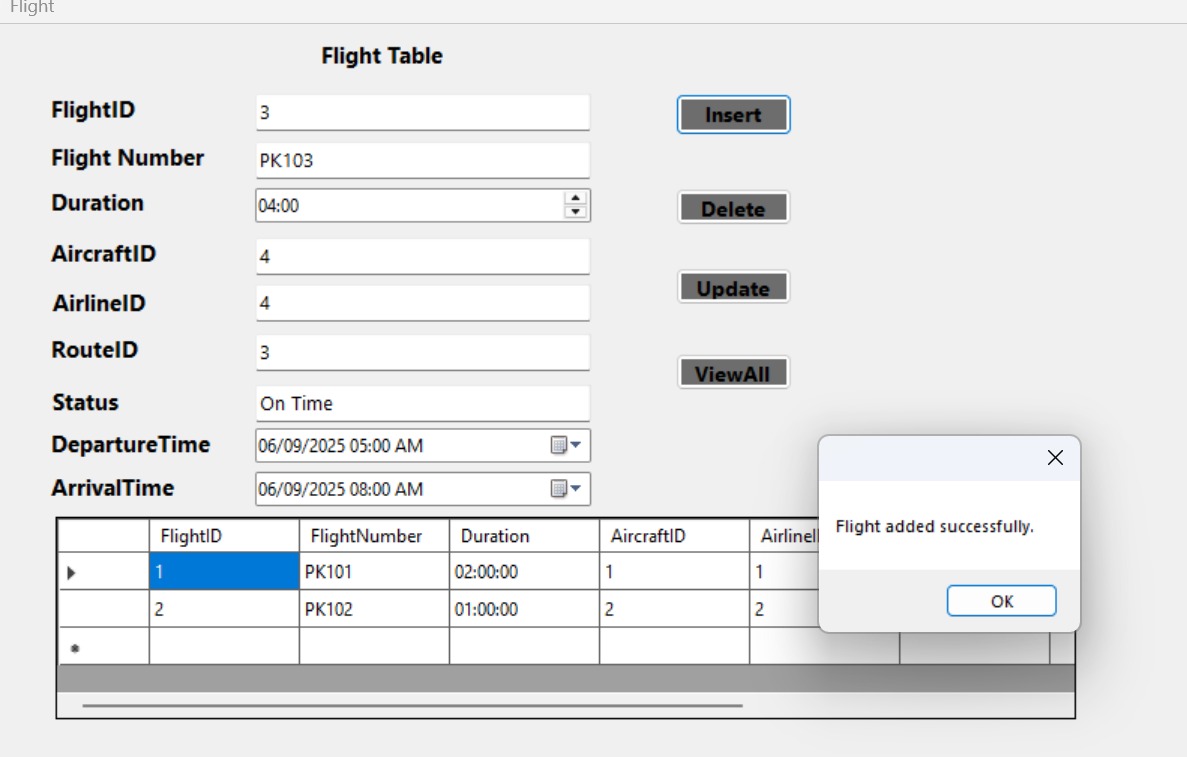
**Output:**



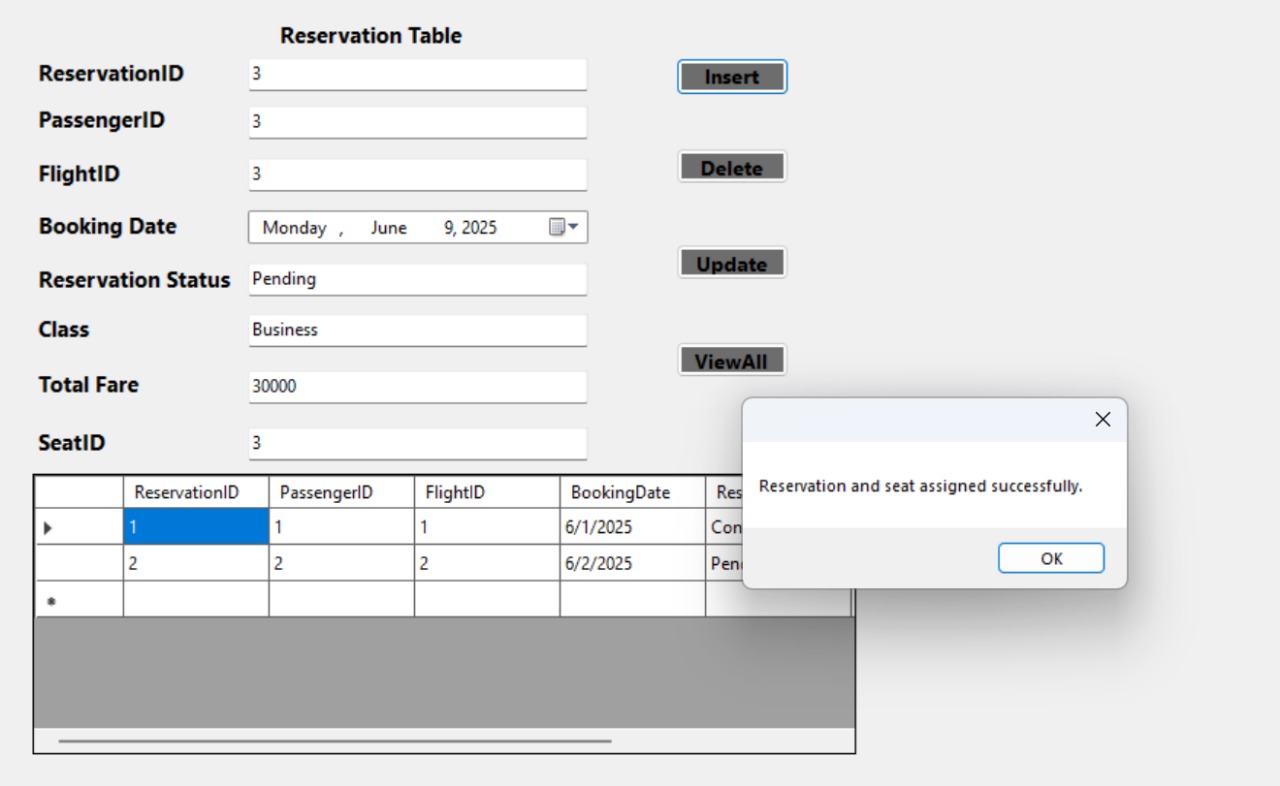
Many other insertions are performed.

## **Manages Flights:**

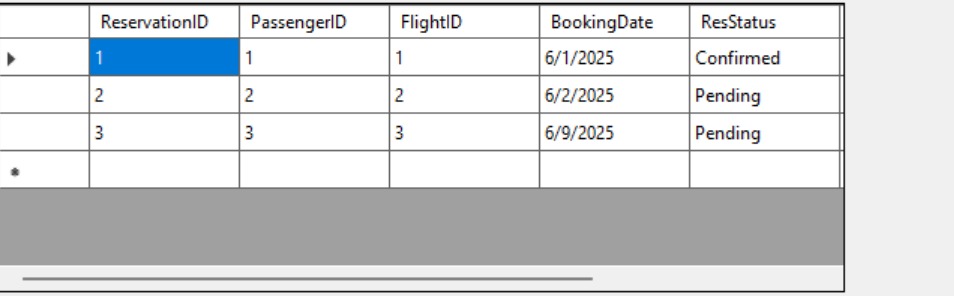




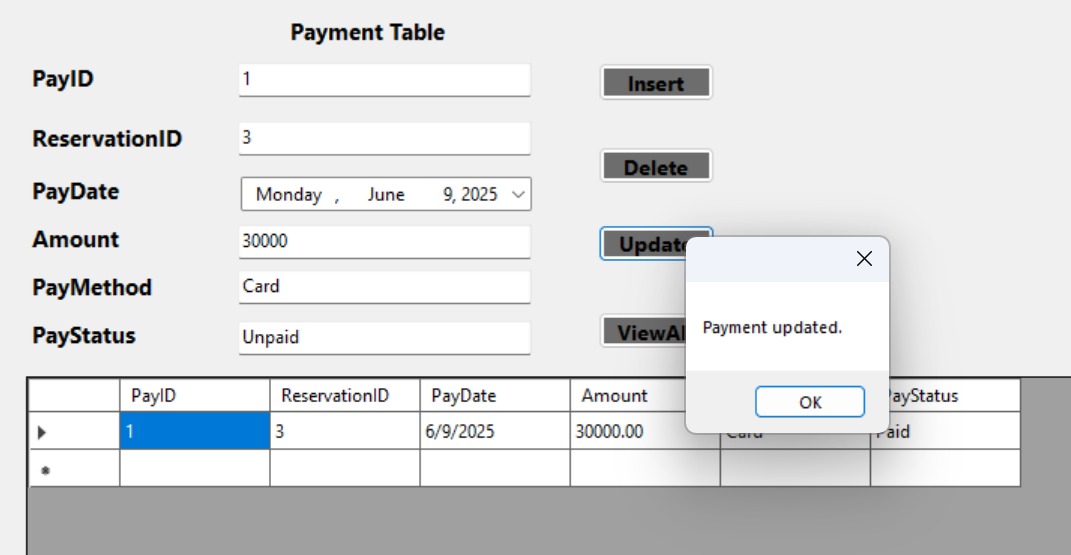
## **Manages Reservation:**



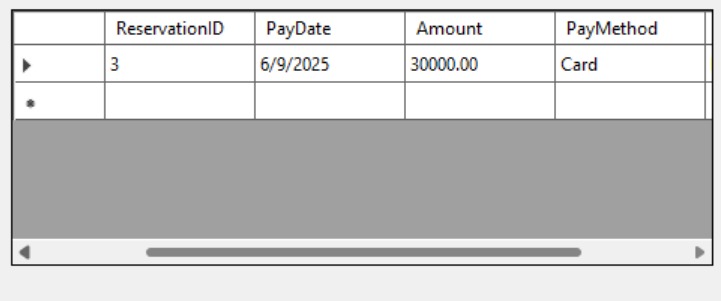
## **Output:**



## **Manages Payments:**

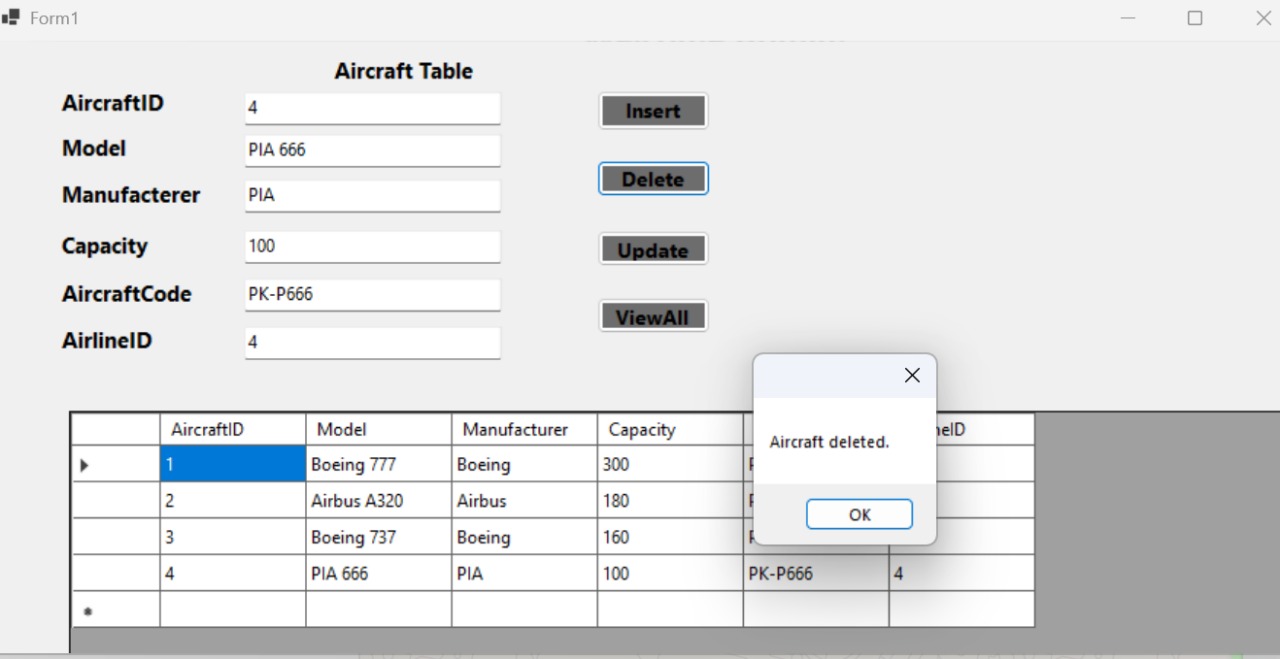


**Output:**

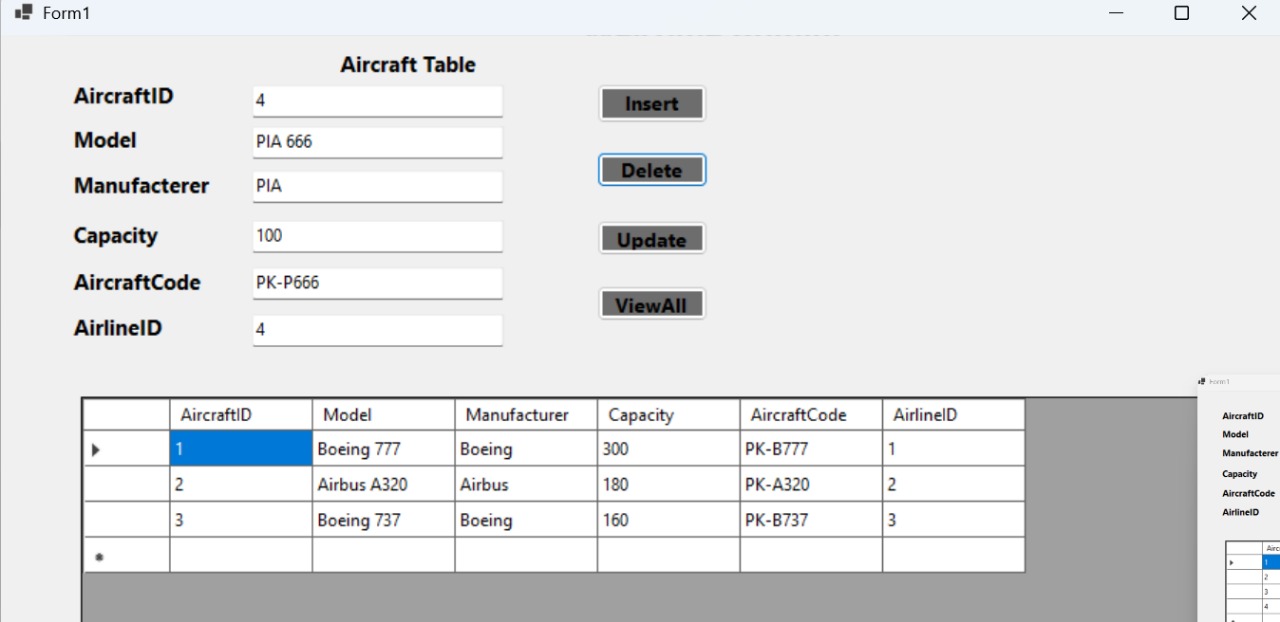


## 

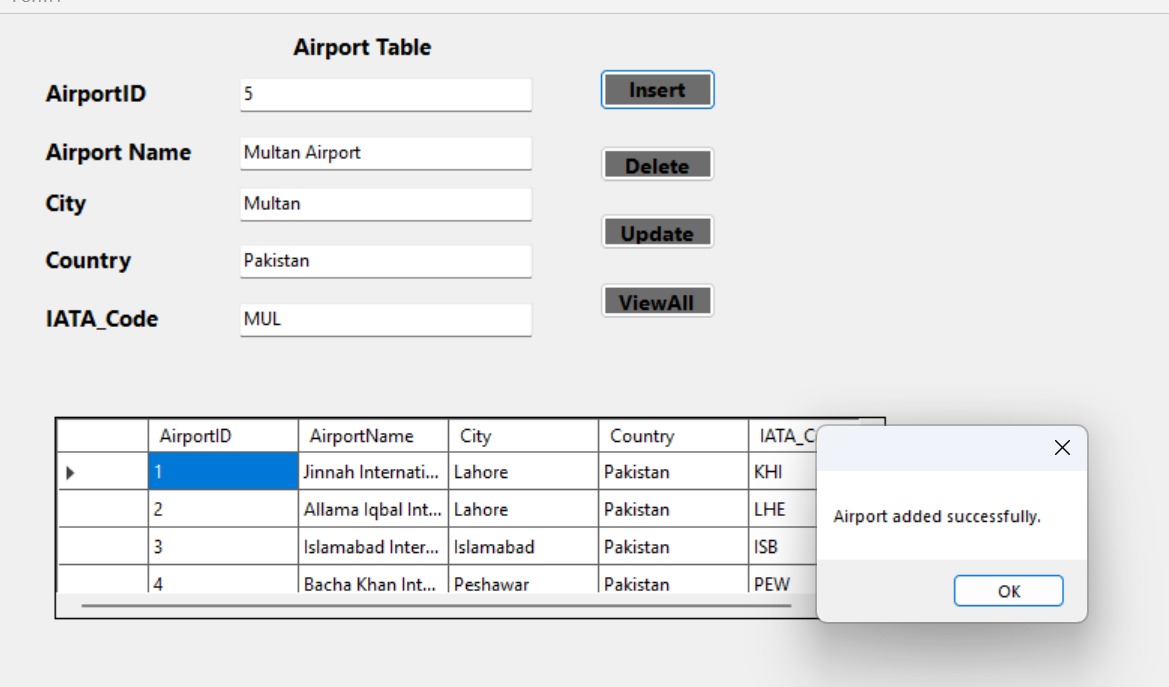
## **Manages Aircraft:**



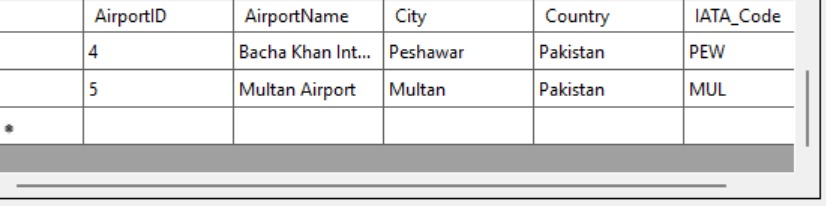
**Output:**



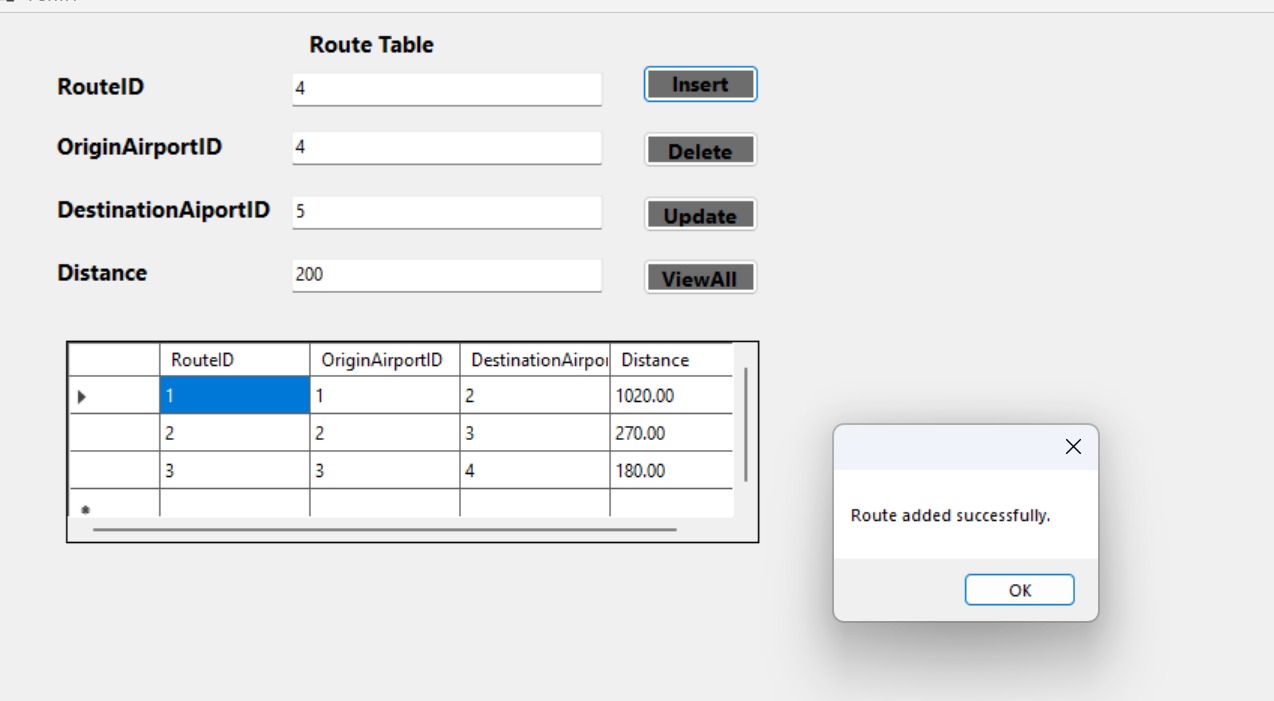
**Manages Airports:**



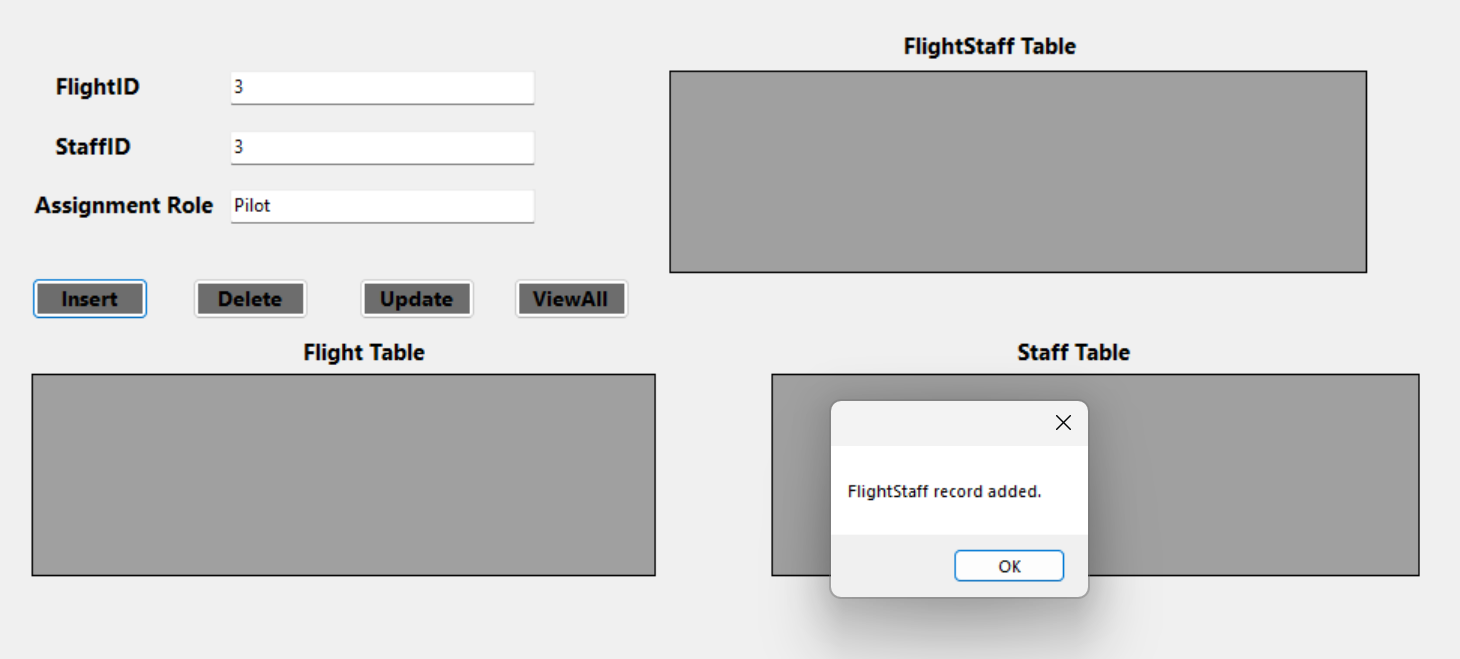
## **Output :**



## **Manages Routes:**



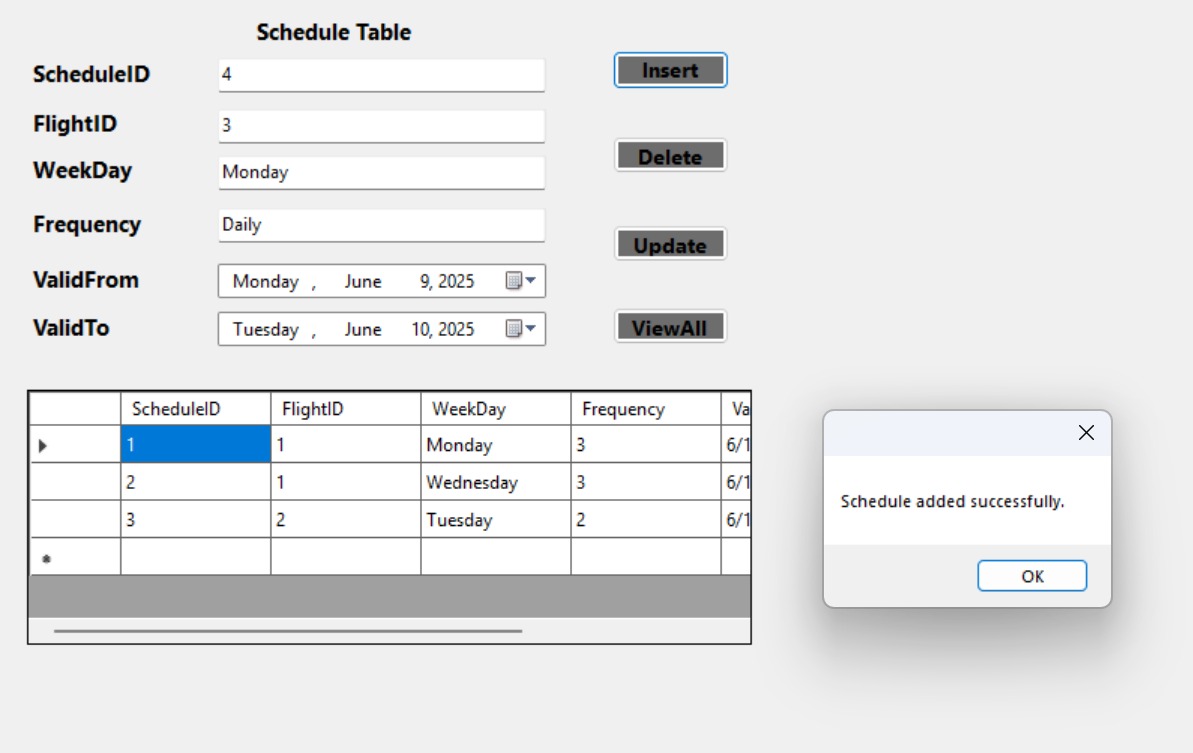
## **Assign Staff To Flight :**



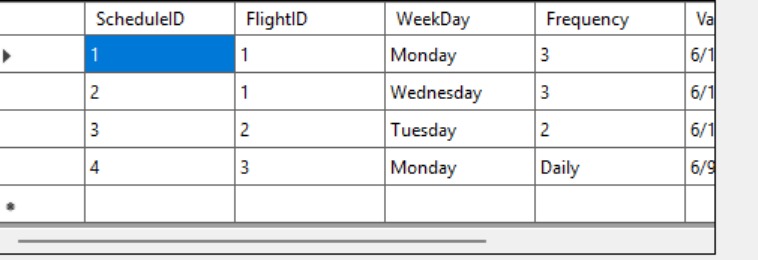
## **Output :**

****

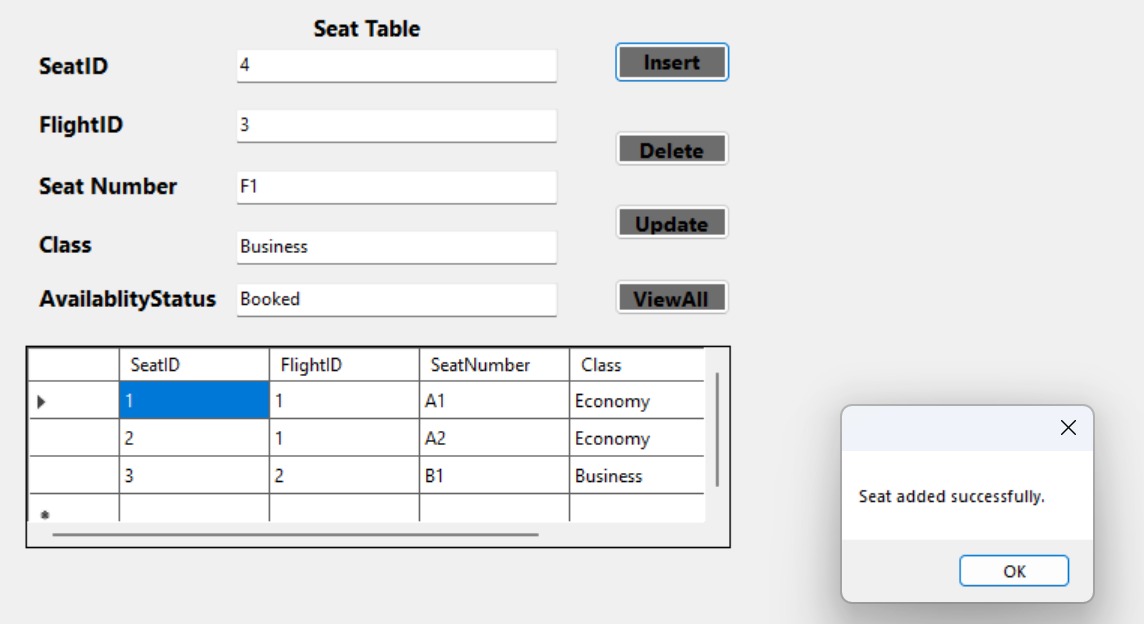
## **Manages Schedule:**

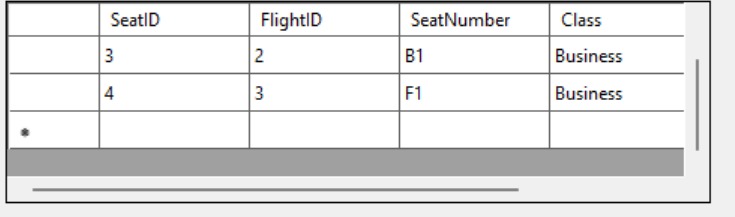


## **Output :**

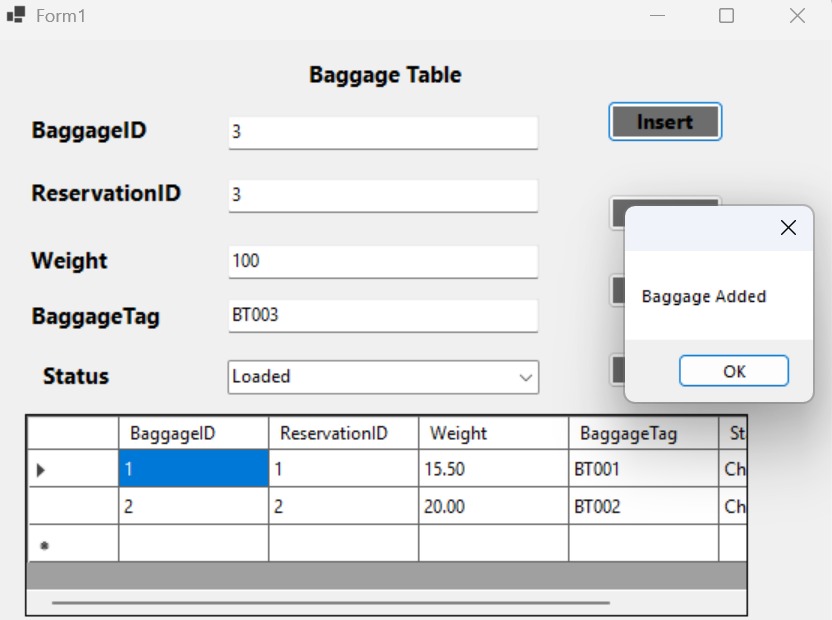


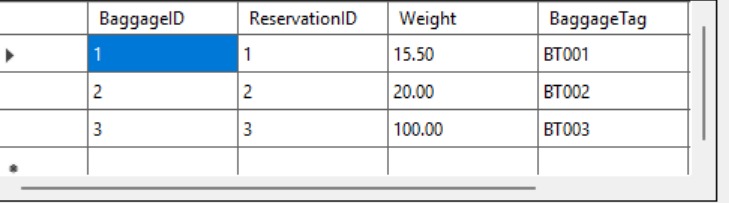
## **Manages Seat:**



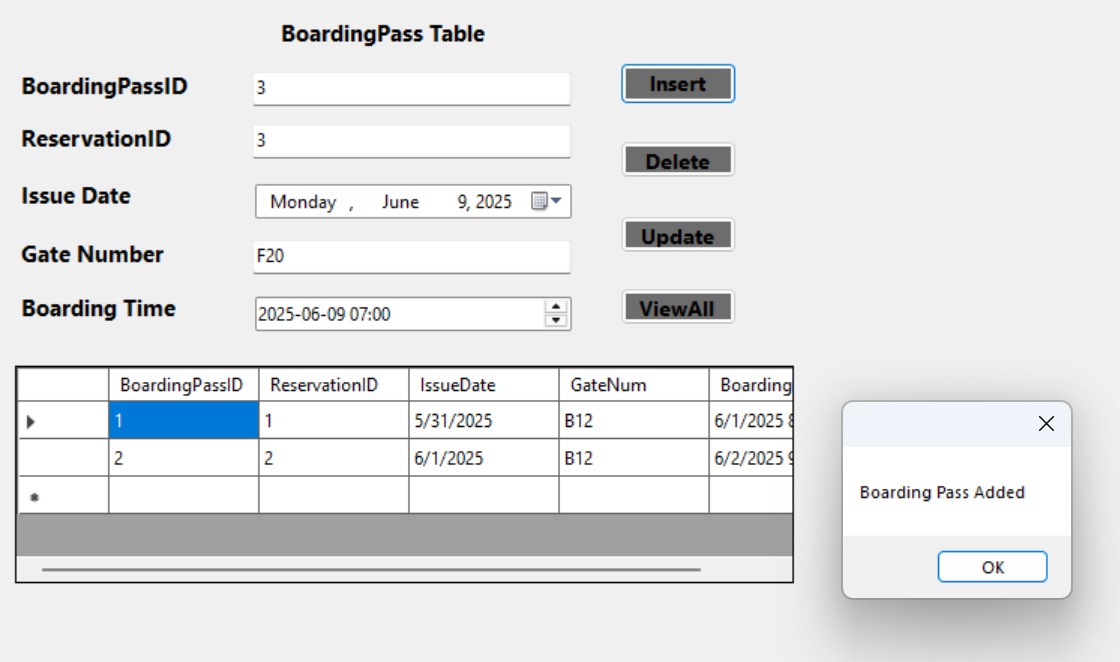


## **Manages Baggage:**





## **Manages BoardingPass:**



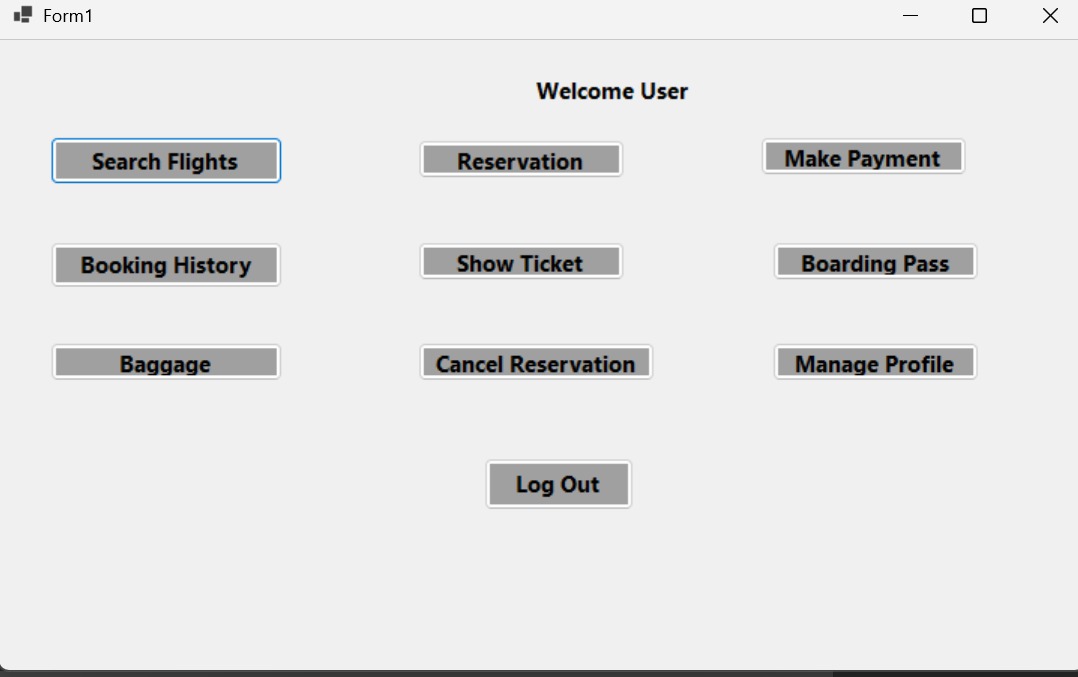
## User Page:

After a successful login by a user, following menu is shown.

The image below displays the "Welcome User" pagewhich is the main interface for a passenger after successfully logging into the Flight Reservation System. This page provides a range of options for users to manage their flight-related activities, including:

* **Search Flights:** Allows the user to look for available flights.
* Book Ticket: Enables the user to reserve a flight ticket.
* Make Payment: Provides an option to process payments for bookings.
* Booking History: Displays a record of past and current bookings.
* Seat Selection: Allows the user to choose their preferred seat for a booked flight.
* Boarding Pass: Enables access to or generation of boarding passes.
* Baggage: Likely provides information or management options related to baggage.
* Cancel: Allows the user to cancel an existing booking.
* Manage Profile: Provides options to update or view user profile details.
* Log Out: Exits the user from the system.

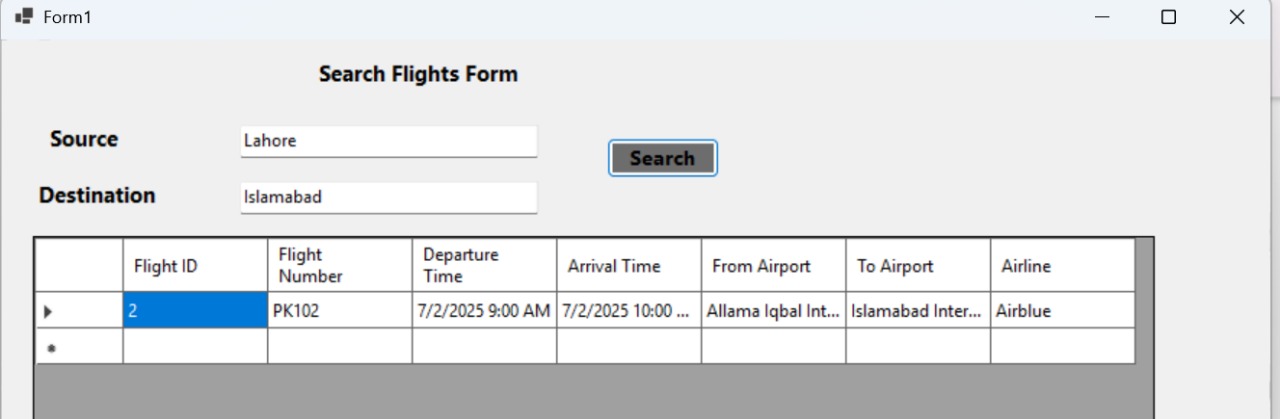
All these functionalities would interact with the underlying flight reservation system database to retrieve, update, or store user-specific and flight-related information.



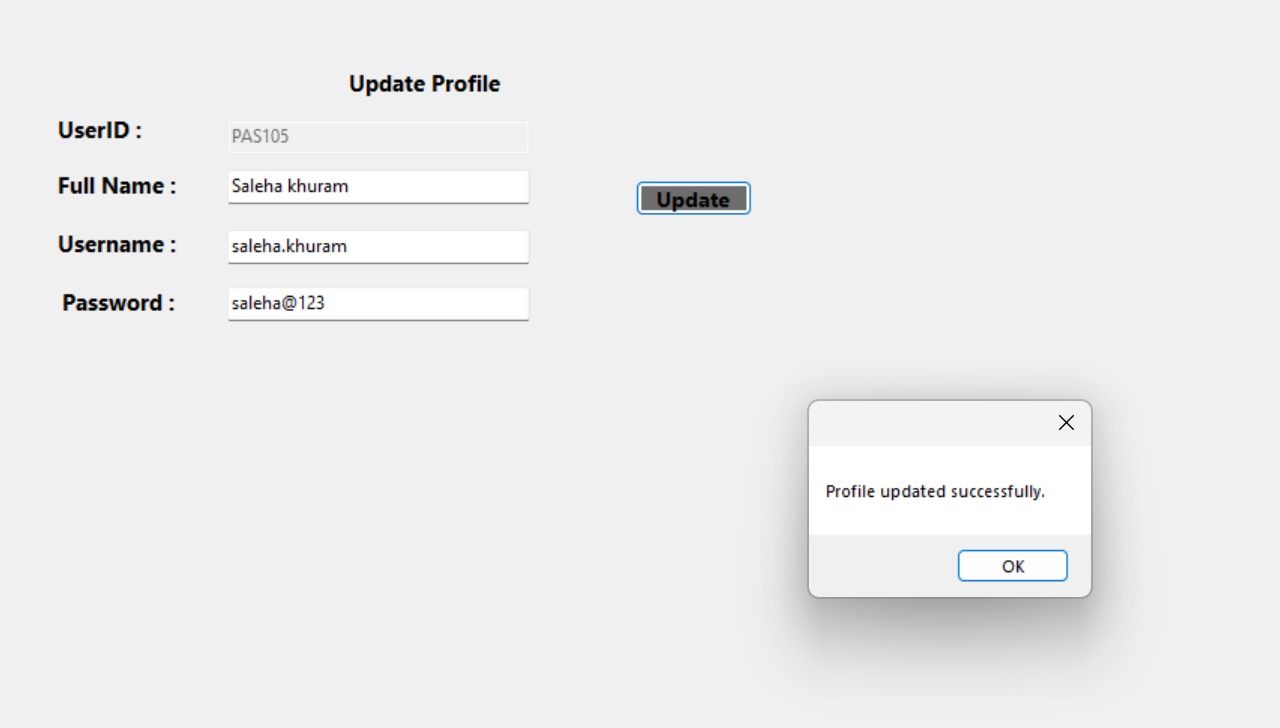
**Examples of User Functionalities:**

## **Search Flights Form:**

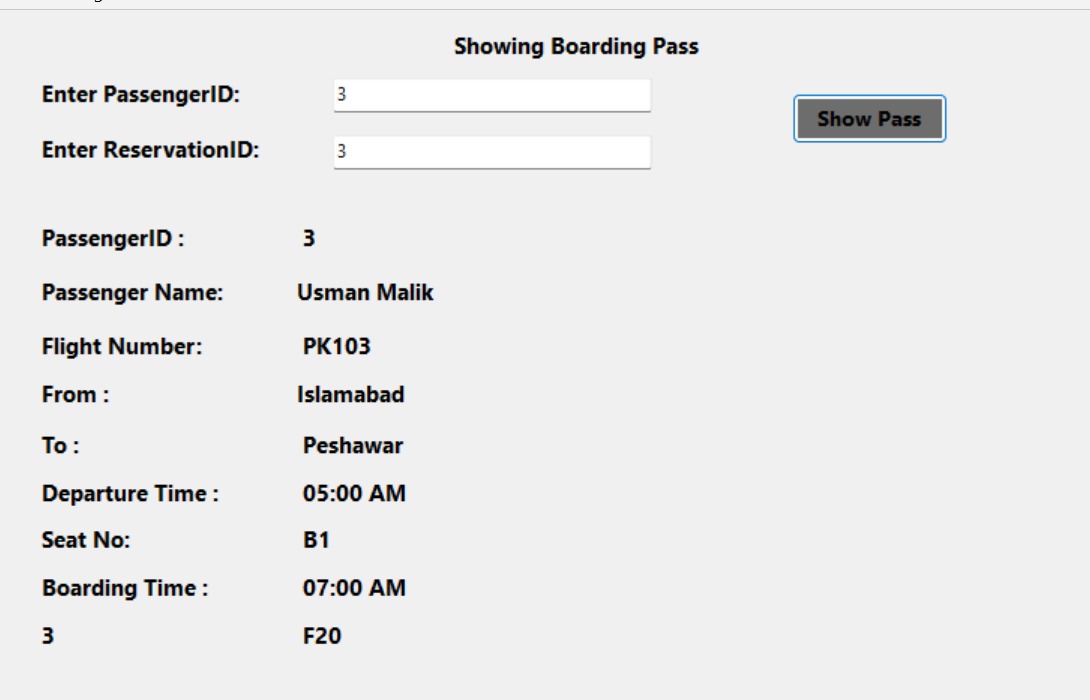
"Search Flights Form" is displayed after a successful user login. It allows the user to search for flights by specifying "Source" (Lahore) and "Destination" (Islamabad). The search results are presented in a table format, showing details for Flight ID "2" (PK102) from Lahore to Islamabad, including departure and arrival times, and the airline (Airblue). This demonstrates the system's ability to retrieve and display flight information from the database based on user queries.



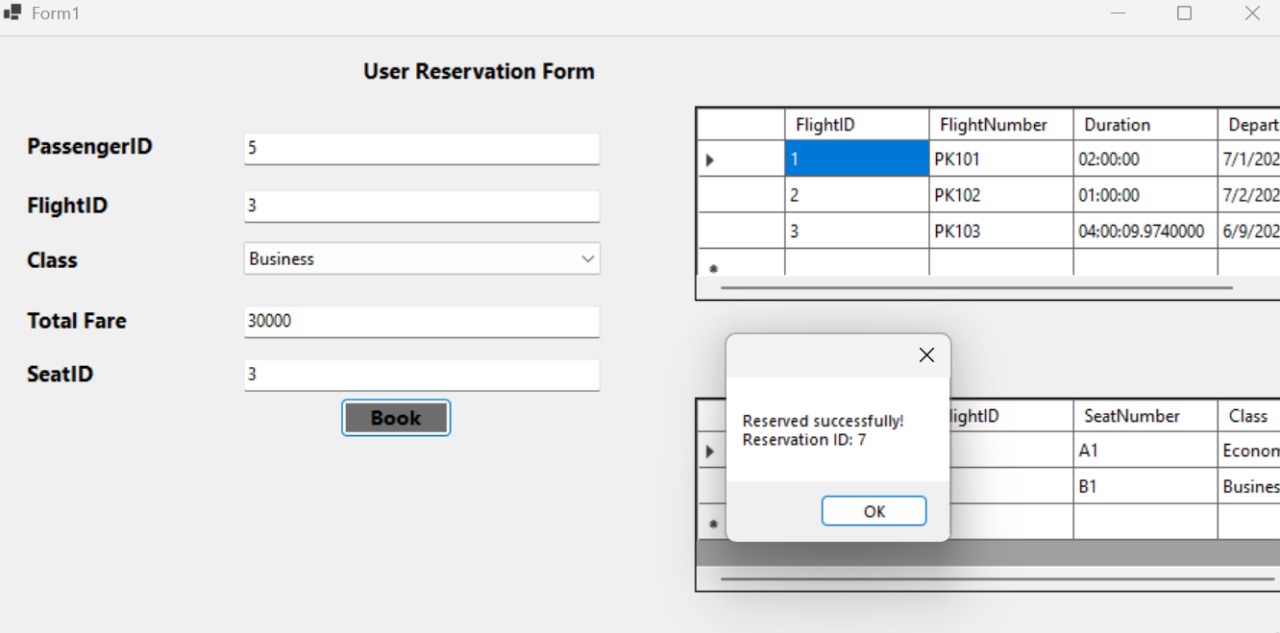
## **Update Profile:**



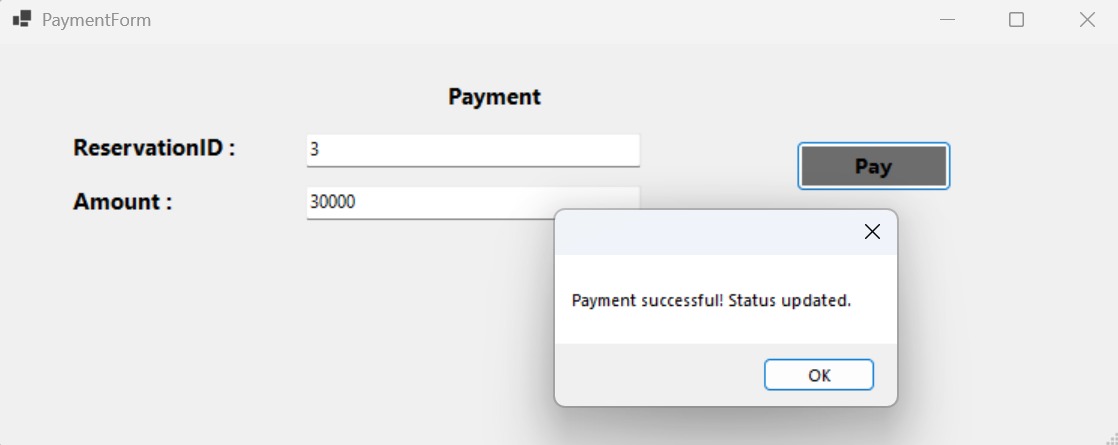
## **Showing Boarding Pass:**



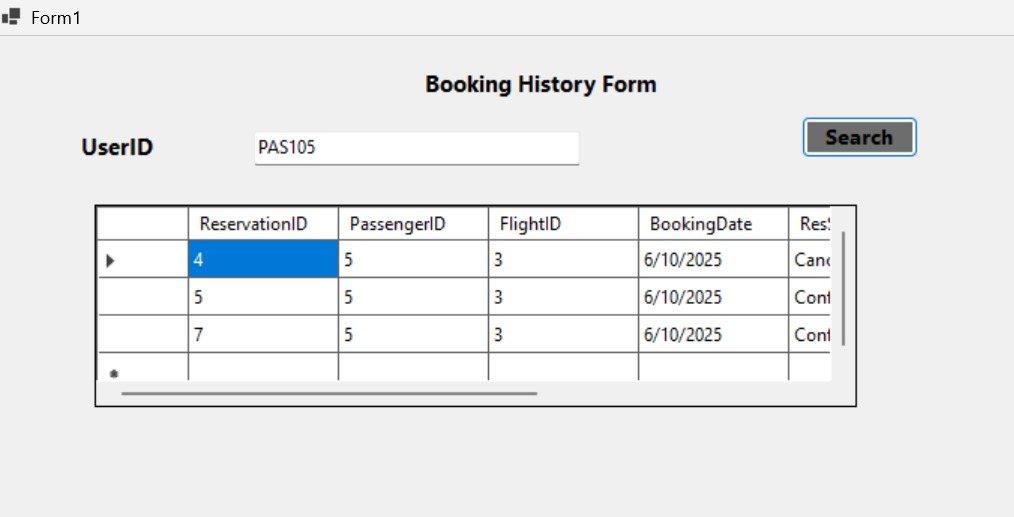
User Reservation :



Make Payment :



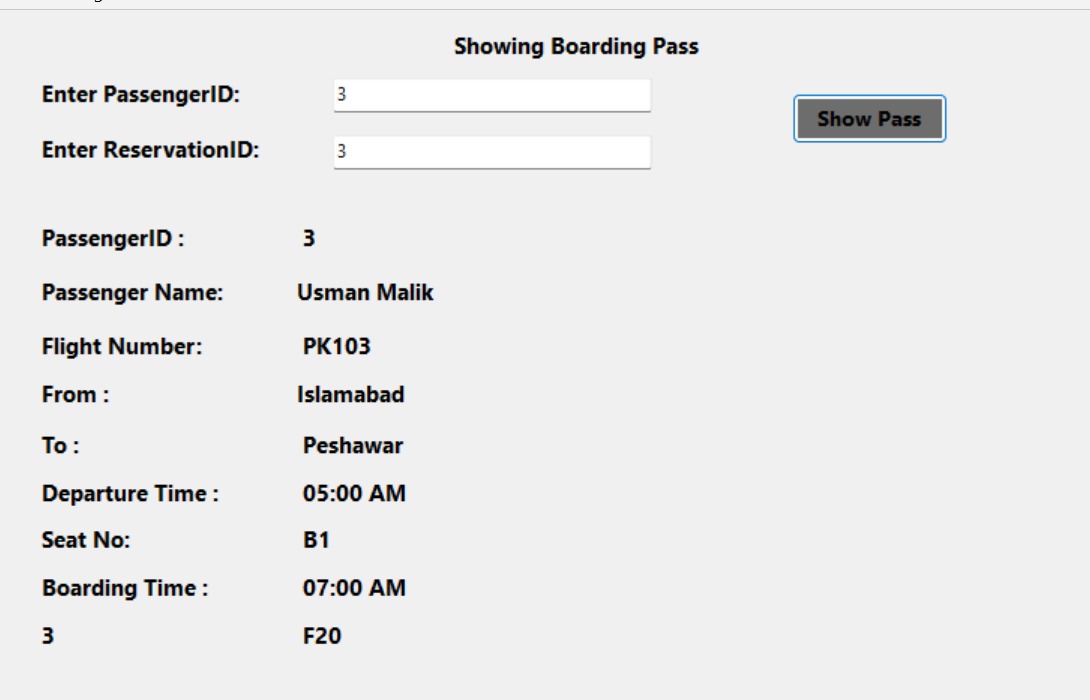
Booking History :



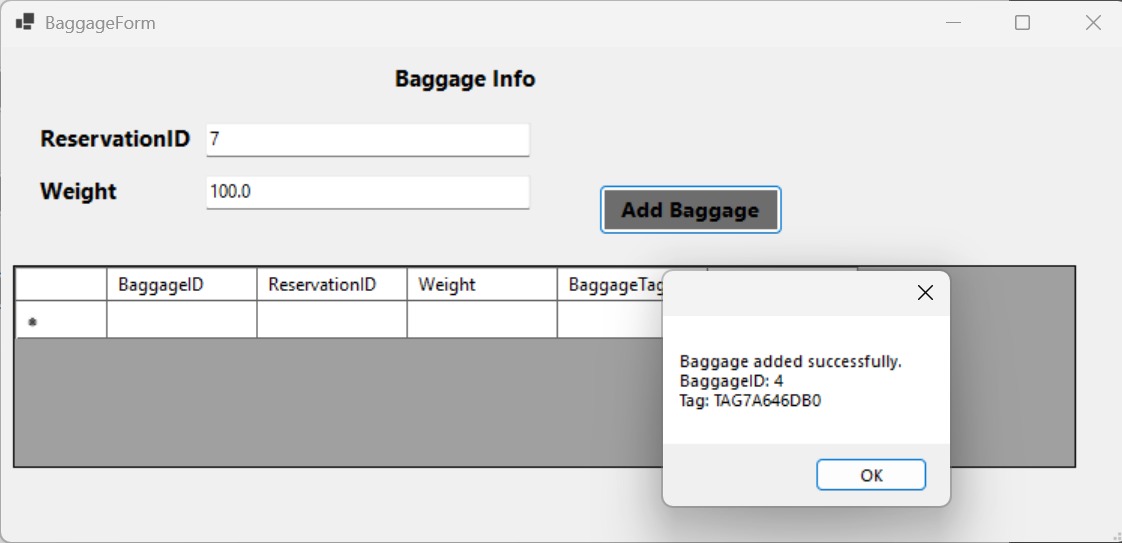
Show Ticket :

# 

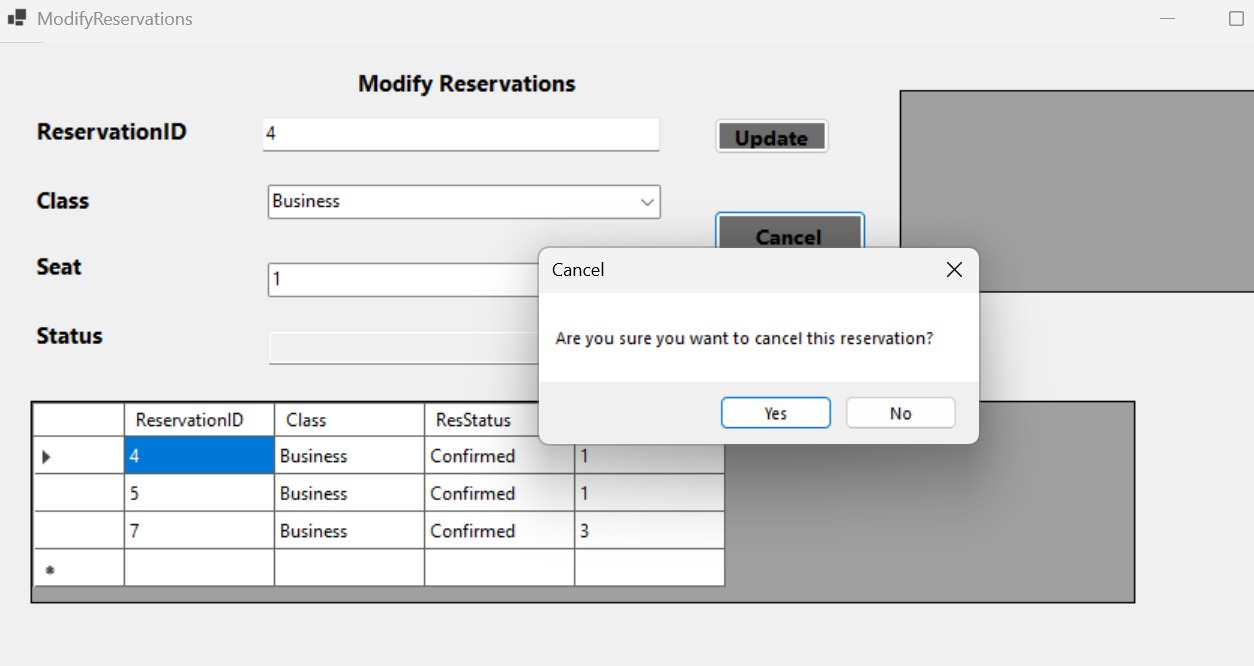
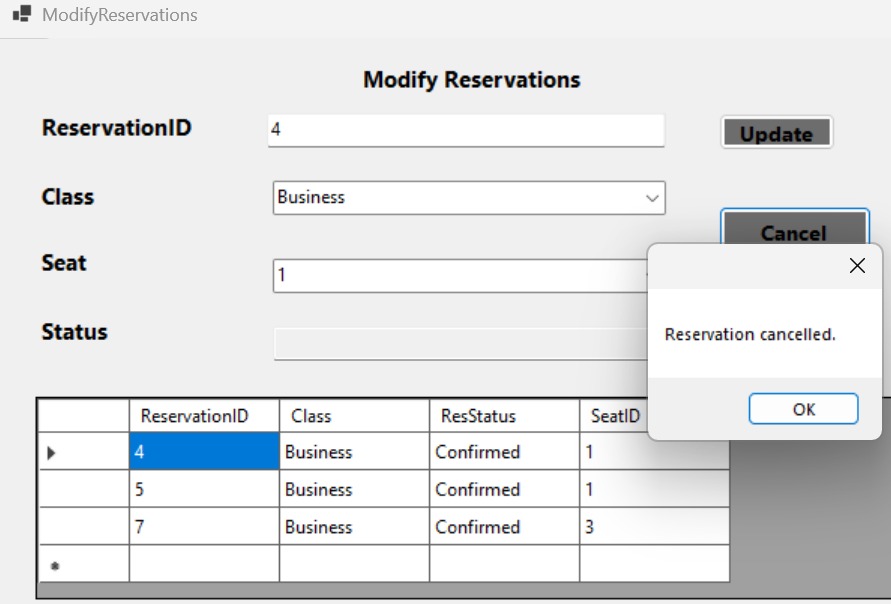
# Boarding Pass :



# Baggage :



# Cancel or Modify Reservation :

# Employee Welcome Page:

After a successful login by an employee, following menu is shown. Menu displays the "Welcome Employee" page (Form1), which serves as the central interface for employees of the Flight Reservation System. This page presents a range of operational tasks that employees can perform to manage various aspects of the system. The available functionalities include: **Manage Flights**: Allows employees to manage flight details.

**Manage Reservations:** Provides options for handling flight reservations.

**Manage Baggage Handling:** Enables management of baggage-related processes.

**Manage Passengers:** Allows employees to manage passenger information.

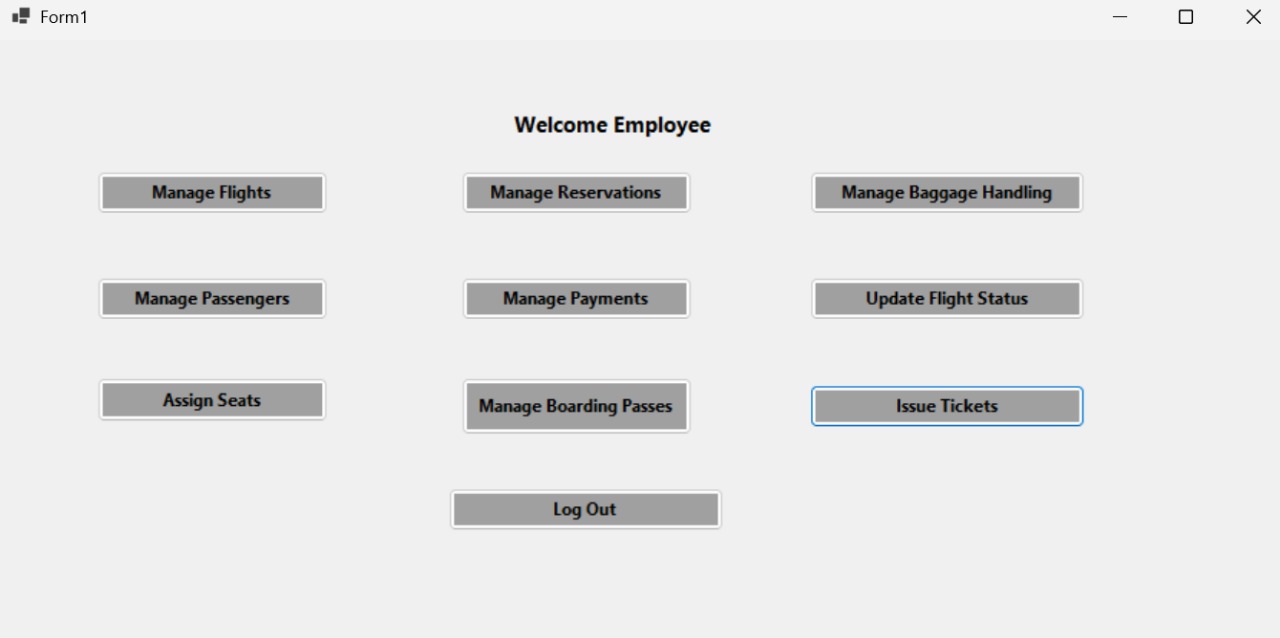
**Manage Payments:** Provides access to payment management functionalities.

**Update Flight Status:** Enables employees to update the status of flights.

**Assign Seats:** Allows for the assignment of seats to passengers.

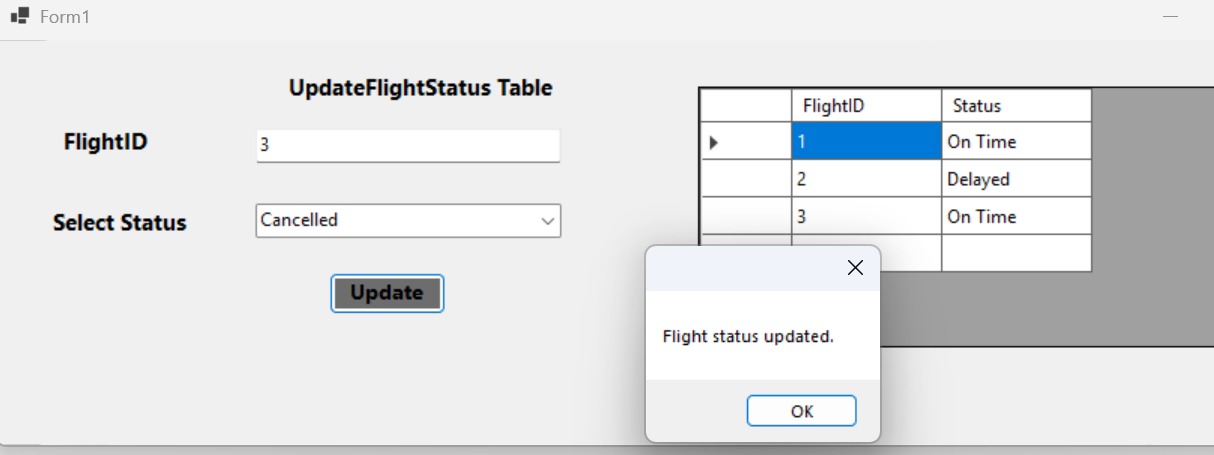
**Manage Boarding Passes:** Provides tools for managing boarding passes.

**Log Out:** Exits the employee from the system.

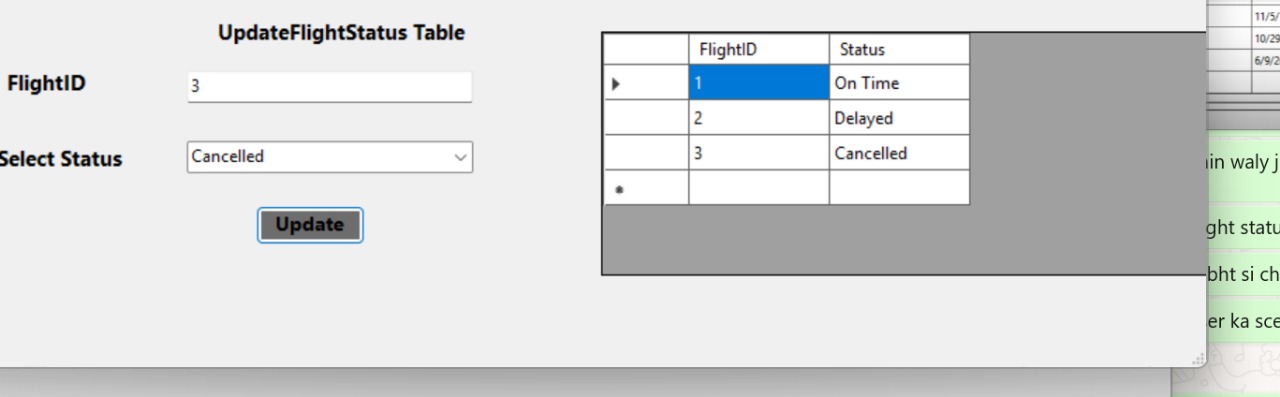


These options demonstrate the employee's interaction with various database modules of the flight reservation system to perform their duties.

## **Update Flight Status:**

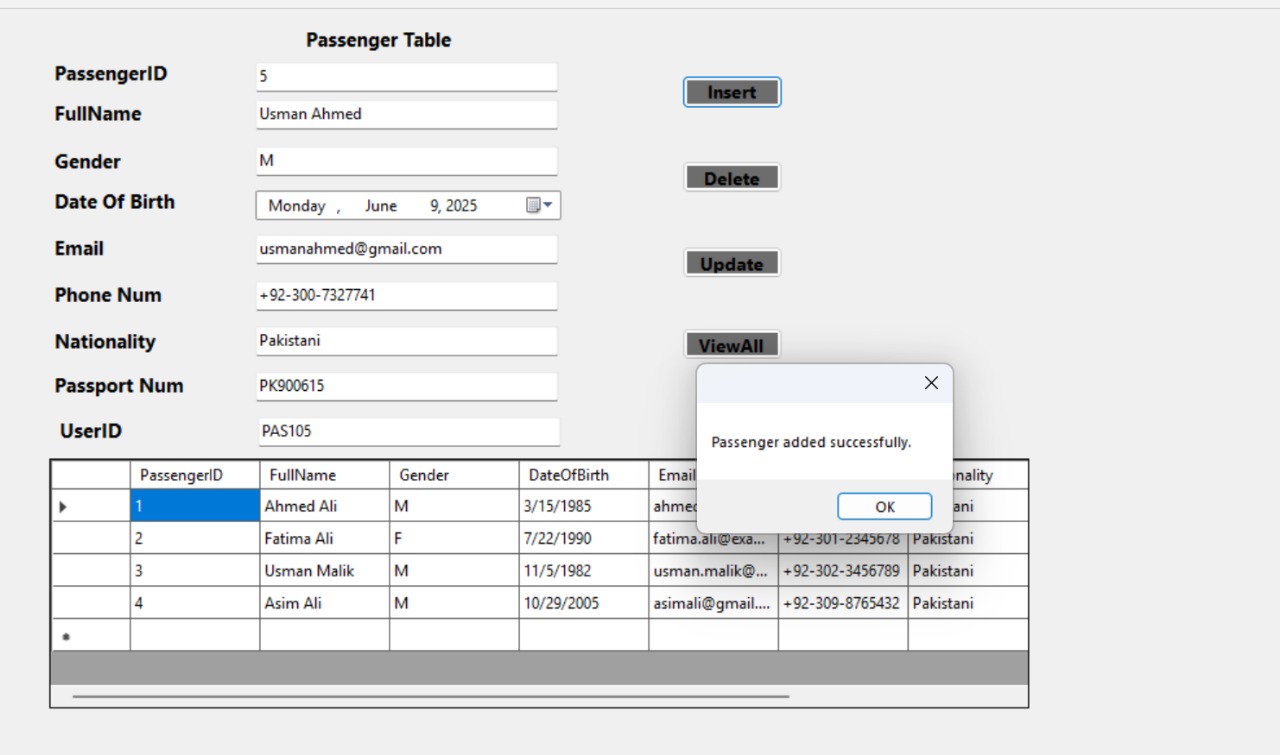


## **Output:**

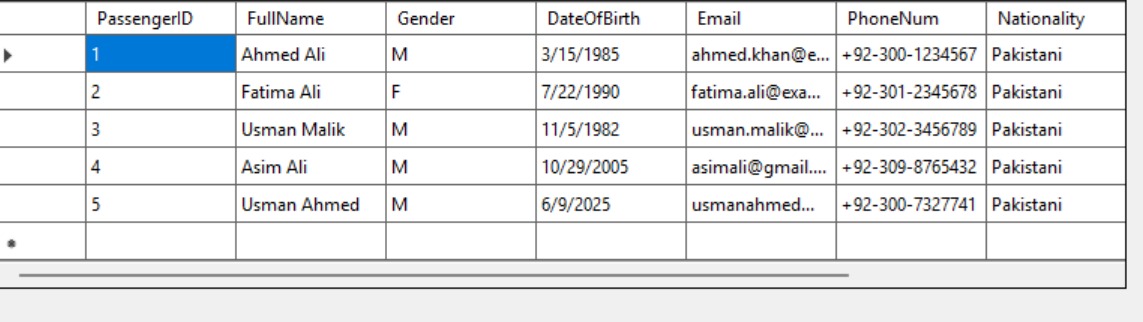


## 

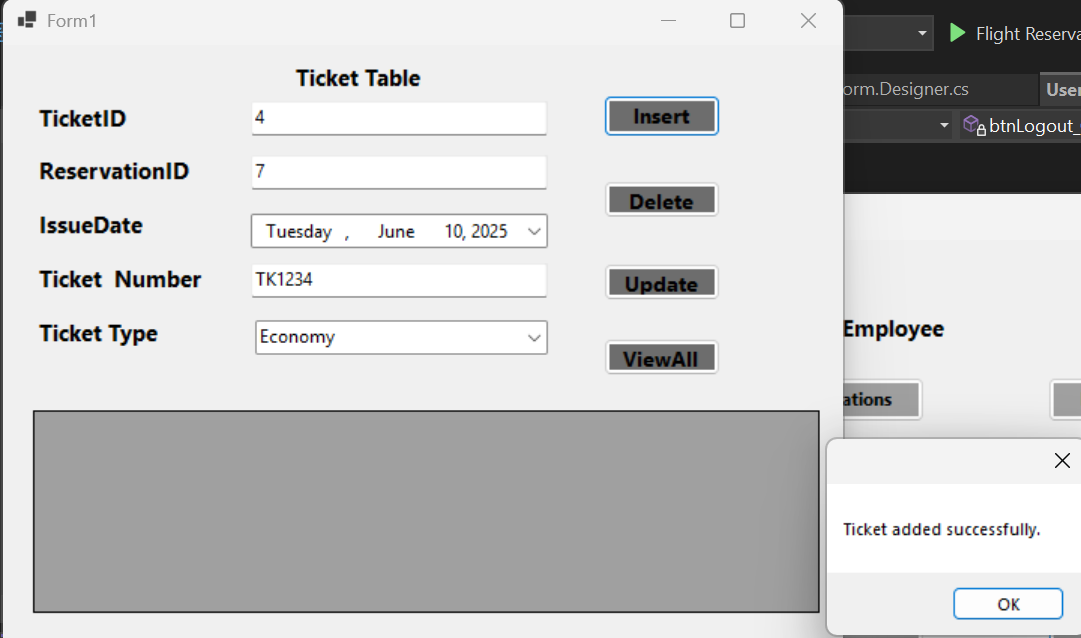
## **Passenger Table:**



## **Output:**



## **Issues Ticket:**



Some functionalities of Employee are same as that of admin as:

Manage BoardingPass,Manage Reservation ,Manage Flights,Manage Baaggage Handling,Assign Seats,Manage payments

# **Challenges and Solutions in the Airline Reservation System (Database and GUI Combined):**

Developing a comprehensive Airline Reservation System involved tackling various challenges, not only in designing a robust database but also in building a user-friendly Graphical User Interface (GUI) that seamlessly interacts with it. Here are the key challenges encountered and their respective solutions:

**Database-Specific Challenges**

**Foreign Key Constraints:**

* **Challenge:** Ensuring data integrity across interconnected tables was crucial. Often, insertion operations failed due to foreign key violations when attempting to add data to a child table without the corresponding parent record existing first (e.g., reserving a seat for a flight that doesn't exist).
* **Solution**: Strict adherence to the database schema design and logical sequencing of data insertion was implemented. This involved inserting records into parent tables (e.g., Flight, Passenger) before inserting related records into child tables (e.g., Reservation, BoardingPass). GUI forms were designed to guide users through the correct data entry flow.

**Data Type Mismatches:**

* **Challenge:** Inconsistent data types between the database schema and the data submitted via the GUI (e.g., trying to insert a string into an integer column) led to errors.
* **Solution:** Careful mapping of GUI input fields to database column types was implemented. Frontend validation was added to ensure that user input conformed to the expected data type before being sent to the database.

**Join Errors and Query Optimization:**

* **Challenge:** Retrieving complex information that spanned multiple tables (e.g., a passenger's full reservation details including flight number, departure airport, and seat) required intricate join operations. Incorrect or inefficient join conditions led to either erroneous results or slow query performance.
* **Solution:** The ER diagram and relational schema were meticulously reviewed to establish correct relationships. SQL queries were carefully crafted using appropriate JOIN clauses (INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN) and WHERE conditions, along with subqueries and aggregate functions, to optimize data retrieval and ensure accuracy. Indexing was considered for frequently queried columns to boost performance.

**GUI-Specific and Integration Challenges**

**User Experience (UX) and Navigation:**

* **Challenge**: Designing an intuitive and easy-to-navigate interface for different user roles (Admin, Passenger, Employee), each with distinct functionalities, was complex. Without clear navigation, users could become lost or struggle to perform tasks.
* **Solution:** The GUI was designed with clear, role-based menus and dedicated welcome pages (e.g., "Welcome Admin!", "Welcome User", "Welcome Employee"). Buttons and forms were logically grouped to guide users through workflows (e.g., "Search Flights" leading to "Book Ticket"). Consistent layout and visual cues were used across different forms.

**Frontend Data Validation and User Feedback:**

* **Challenge**: Preventing invalid or incomplete data from reaching the database, and providing immediate, clear feedback to the user on input errors, was a significant challenge.
* **Solution:** Client-side validation was implemented in the GUI for input fields (e.g., checking for empty fields, correct date formats, valid email patterns). Pop-up messages were used to inform users about successful operations (e.g., "Signup successful!", "Airline added successfully!") or errors (e.g., "Invalid login credentials").

**Real-time Data Synchronization:**

* **Challenge:** Ensuring that the data displayed on the GUI was always up-to-date with the database, especially in scenarios like seat availability or flight status, was critical.
* **Solution:** Mechanisms were implemented to refresh data displayed in tables (e.g., DataGridView in the provided example) after any INSERT, UPDATE, or DELETE operation. This ensured that changes made by one user (or an administrative action) were reflected promptly for others viewing the same information.

**Security and Role-Based Access Control:**

* **Challenge:** Protecting sensitive information (like user passwords) and ensuring that users could only access functionalities relevant to their assigned role (e.g., a passenger cannot manage flights) was paramount.
* **Solution:** Password hashing was implicitly handled by the system's underlying security practices (though not explicitly detailed in the DDL/DML for Users). Role-based access control was implemented, where the GUI presented different menus and functionalities based on the user's Role determined during login (e.g., "Admin," "Employee," "Passenger"), restricting unauthorized actions.

By systematically addressing these challenges, the project delivered an integrated Airline Reservation System where the GUI served as an effective and intuitive interface for managing a robust and reliable database backend.

# **Conclusion:**

This project successfully implemented a comprehensive Airline Reservation System, integrating a robust database backend with an intuitive Graphical User Interface (GUI) to manage all essential aspects of airline operations. The database, built using Microsoft SQL Server, adheres to relational principles through ER modeling, normalization, and a structured schema of interrelated tables. This design ensures data integrity, efficiency, and consistency for flight scheduling, passenger reservations, staff assignments, payment processing, seat allocation, and baggage handling. The effective use of Data Definition Language (DDL) for schema creation and Data Manipulation Language (DML) for data operations, including insertion, updates, deletions, subqueries, joins, and aggregate functions, demonstrates a strong foundation in database management and real-world problem-solving.

The accompanying GUI serves as a user-friendly visual interface, enabling seamless interaction with the underlying database system. Different user roles—Admin, User (Passenger), and Employee—are provided with tailored functionalities through dedicated welcome pages and forms. For instance, the GUI facilitates flight searches, ticket bookings, payment processing, profile management, and viewing boarding passes for users, while administrators and employees can manage flights, users, staff, reservations, and baggage handling directly through visual components. The integration ensures that all user actions on the frontend translate into corresponding database operations, maintaining data consistency and security. This project significantly enhanced the understanding of database design, normalization, data manipulation, and the critical role of a well-designed interface in large-scale systems like airline reservation platforms. Future enhancements could include advanced features like triggers, stored procedures, or views to further automate and strengthen the system.

# **References:**

* **Book Database Systems** - A Practical Approach to Design, Implementation, and Management

SIXTH EDITION

Thomas Connolly & Carolyn Begg.

* **Course Slides and Lectures:**  
  Supported database concepts, ER diagrams, and SQL implementation.
* **Pakistani Airport and Airline Data**:  
   Real-world data was referenced from official Pakistani airline and airport websites for project realism and localization