AIRLINE RESERVATION SYSTEM

### A Mini Project Report submitted to

**MOHAN BABU UNIVERSITY**

### in Partial Fulfillment of the Requirements for the Award of the degree of

**BACHELOR OF TECHNOLOGY IN**

### COMPUTER SCIENCE AND ENGINEERING (CYBER SECURITY)

*Submitted by*

|  |  |
| --- | --- |
| **J.MANOJ** | **(23102A020014)** |
| **T.VENU SATHVIK REDDY** | **(23102A020016)** |
| **M.V.THEERDHA** | **(23102A020029)** |

*Under the Guidance of*

### Mr. P. Yogendra Prasad

Assistant Professor

Department of Data Science



**Department of Data Science**

**SCHOOL OF COMPUTING**

**MOHAN BABU UNIVERSITY**

Sree Sainath Nagar, Tirupati – 517 102, A.P., INDIA

2024-25

**MOHAN BABU UNIVERSITY**

Sree Sainath Nagar, Tirupati – 517 102, A.P., INDIA

2024-25

**DEPARTMENT OF DATA SCIENCE**

CERTIFICATE

This is to certify that the mini project report entitled

**“AIRLINE RESERVATION SYSTEM”**

is the Bonafide work done by

|  |  |
| --- | --- |
| **J.MANOJ** | **(23102A020014)** |
| **T.VENU SATHVIK REDDY** | **(23102A020016)** |
| **M.V.THEERDHA** | **(23102A020029)** |

in the Department of **Data Science**, and submitted to Mohan Babu University, Tirupati in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering (Cyber Security) during the academic year 2024-2025. This work has been carried out under my supervision. The results of this mini project work have not been submitted to any university for the award of any degree or diploma.

***Guide: Head:***

Mr. P. Yogendra Prasad Dr. P.K. Gupta

Assistant Professor Professor & Head

Dept. of Data Science Dept. of Data Science

**INTERNAL EXAMINER EXTERNALEXAMINER**

**DEPARTMENT OF DATA SCIENCE**

**Vision**

To become a Centre of Excellence in Data Science by imparting high quality education through teaching, training and research

**Mission**

* To impart quality education in Computer Science and Engineering with specializations in Data Science by disseminating knowledge through contemporary curriculum, competent faculty and effective teaching-learning methodologies.
* Nurture research, innovation and entrepreneurial skills among students and faculty to contribute to the needs of industry and society.
* Inculcate professional attitude, ethical and social responsibilities for prospective and promising Engineering profession.
* Encourage students to engage in life-long learning by creating awareness of the contemporary developments in Computer Science and Engineering with specialization in Data Science.

**PROGRAM EDUCATIONAL OBJECTIVES**

After few years of graduation, the graduates of B. Tech. CSE(DS) will:

|  |  |
| --- | --- |
| **PEO1.** | Pursue higher studies in Computer Science, Data science or Management. |
| **PEO2.** | Become successful entrepreneurs or be employed by acquiring required skill sets in the domains of Data Science and allied areas. |
| **PEO3.** | Exhibit progression and effective adaptation to technological developments through life-long learning to address ever changing industrial requirements and follow ethical attitude in professional practice. |

**PROGRAM SPECIFIC OUTCOMES**

On successful completion of the Program, the graduates of B. Tech. CSE(DS) program will be able to:

|  |  |
| --- | --- |
| **PSO1.** | Apply appropriate data analytical techniques for building effective decision-making systems. |
| **PSO2.** | Develop intelligent systems using novel Machine Learning and Artificial Intelligence techniques. |
| **PSO3.** | Design and develop efficient software systems using modern tools, techniques, and platforms to meet societal needs. |
| **PSO4.** | Apply suitable tools and techniques to build secure distributed systems. |

**PROGRAM OUTCOMES**

On successful completion of the Program, the graduates of B.Tech. CSE (DS) Program will be able to:

|  |  |
| --- | --- |
| **PO1.** | **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems. |
| **PO2.** | **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences. |
| **PO3.** | **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations. |
| **PO4.** | **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions. |
| **PO5.** | **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations. |
| **PO6.** | **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice. |
| **PO7.** | **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development. |
| **PO8.** | **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice. |
| **PO9.** | **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings. |
| **PO10.** | **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions. |
| **PO11.** | **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments. |
| **PO12.** | **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change. |

# DECLARATION

We hereby declare that this project report titled **“AIRLINE RESERVATION SYSTEM”** is a genuine work carried out by us, in **B.Tech *(Computer Science and Engineering (Cyber Security)*** degree course of **Mohan Babu University, TIrupati** and has not been submitted to any other course or University for the award of any degree by us.

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in our submission. We understand that any violation of the above will cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature of the students 1.

2.

3.

## ABSTRACT

The DBMS-based Airline Reservation System project is designed to revolutionize the traditional approach to airline booking by addressing the inefficiencies of outdated manual and computerized systems. This project introduces a centralized database that integrates various aspects of airline operations, including flight scheduling, passenger management, and ticket booking, into a single cohesive platform. By transitioning to a DBMS-driven environment, the system is structured to eliminate data redundancy, ensure data integrity, and provide real-time updates. These features are essential in overcoming common issues such as double bookings, scheduling conflicts, and delayed communications, thereby enhancing the overall travel experience for passengers and operational efficiency for airlines.

The system’s architecture is built on robust relational database principles, with an emphasis on automated transaction processing and secure data handling. Advanced features include real-time flight availability updates, dynamic seat inventory management, and a reliable payment gateway that supports seamless financial transactions. Furthermore, the system employs rigorous security measures, such as user authentication and encryption protocols, to protect sensitive customer and operational data. The integration of comprehensive reporting tools allows airline management to monitor performance metrics and gain valuable insights into booking trends and resource utilization, facilitating informed decision-making and proactive service enhancements.

In summary, this project aims to deliver a scalable, user-friendly, and highly efficient airline reservation system that addresses the critical needs of modern air travel. By leveraging the capabilities of a DBMS, the system not only streamlines the booking process and minimizes manual errors but also enhances data security and operational transparency. The expected outcome is a significant improvement in customer satisfaction and operational performance, positioning airlines to better meet the demands of a rapidly evolving travel industry while reducing costs and maximizing revenue opportunities.

Keywords: SQL Server, Microsoft SQL server management studio, HTML, CSS, java script, PHP server.

## TABLE OF CONTENTS

|  |  |
| --- | --- |
| **Title** | **Page No.** |
| **ABSTRACT** | i |
| **CHAPTER 1. INTRODUCTION** |  |
| 1.1 Introduction to the topic | 2-2 |
| 1.2 Problem Statement | 2-3 |
| 1.3 Objectives | 3-3 |
| **CHAPTER 2. DATABASE DESIGN** |  |
| 2.1 List of Attributes, entities and relationship | 4-8 |
| 2.2 E-R Diagram | 8-9 |
| **CHAPTER 3. RELATIONAL MODEL** |  |
| 3.1 Database languages | 10-10 |
| 3.2 Table Description | 11-14 |
| 3.3 Relational Database Scheme | 14-14 |
| 3.4 Relational Queries | 14-40 |
| **CHAPTER 4. CONCLUSION AND FUTUREWORK** |  |
| 4.1 Conclusion | 41-41 |
| 4.2 Future Work | 41-42 |
|  |  |
|  |  |
|  |  |
|  |  |

# CHAPTER 1. INTRODUCTION

**1.1 Introduction to the topic**

The DBMS-based Airline Reservation System represents a modern and efficient solution designed to handle the complexities of airline booking and management. This system integrates a robust database management framework to streamline various functions such as ticket booking, flight scheduling, passenger management, and payment processing, ensuring a seamless user experience. By leveraging relational databases, the system efficiently organizes and retrieves large volumes of data related to flight routes, schedules, passenger records, and reservation details. Its architecture supports high transaction rates, ensuring that booking requests are processed quickly and accurately even during peak travel periods. Additionally, the system incorporates advanced features like real-time availability updates, automated notifications, and secure payment gateways to enhance user satisfaction and operational reliability. The application of normalization techniques and optimized query processing minimizes data redundancy and maximizes data integrity, which are critical in maintaining consistent and reliable information flow. Moreover, the system provides administrators with comprehensive reporting tools and analytics capabilities that enable them to monitor performance, identify trends, and make informed decisions to improve service quality and operational efficiency. The design also emphasizes security measures, including user authentication and encryption protocols, to protect sensitive information from unauthorized access. Overall, the DBMS-based Airline Reservation System not only simplifies the booking process for passengers but also improves resource management and operational coordination for airlines, ultimately contributing to a more efficient, secure, and customer-friendly travel experience.

**1.2 Problem Statement**

The current airline reservation process faces significant challenges that hinder both customer satisfaction and operational efficiency. Manual booking methods and outdated computerized systems often lead to errors, delays, and data inconsistencies, which can result in double bookings, scheduling conflicts, and mismanagement of passenger information. These inefficiencies not only frustrate customers during the booking process but also complicate administrative tasks for airlines, leading to increased operational costs and lost revenue opportunities. Furthermore, the absence of real-time data integration hampers the ability to update flight statuses, manage seat inventories accurately, and provide timely notifications to travelers about schedule changes or cancellations. In an industry where time and precision are critical, these shortcomings affect overall service quality and diminish competitive advantage. Additionally, the lack of robust security measures in legacy systems poses risks of unauthorized data access and potential breaches, undermining customer trust and regulatory compliance. The growing demand for seamless, user-friendly, and secure travel experiences necessitates a comprehensive overhaul of the existing reservation frameworks. A DBMS-based Airline Reservation System is proposed to address these challenges by integrating modern database management techniques, ensuring data integrity, and enabling efficient transaction processing. This system aims to streamline booking procedures, enhance data accuracy, and provide dynamic updates to both customers and staff. By leveraging automation, real-time communication, and secure data practices, the new system will improve the overall travel experience, reduce operational bottlenecks, and support the scalability required to meet future industry demands.

**1.3 Objectives**

The main objectives of the DBMS-based Airline Reservation System project are centered on enhancing efficiency, security, and user satisfaction throughout the entire airline booking process. Firstly, the system aims to develop a centralized and robust database that streamlines the management of flight schedules, passenger details, and booking transactions, thereby ensuring data integrity and reducing redundancy. Additionally, it seeks to implement real-time updates for flight availability and schedule changes, which will allow both customers and airline staff to access the most current information promptly. Another key objective is to automate the booking and ticketing processes, minimizing manual errors and expediting the overall transaction process. The system also emphasizes the importance of incorporating secure user authentication and data encryption measures to safeguard sensitive customer information and maintain compliance with regulatory standards. Furthermore, it aims to integrate a reliable payment processing mechanism, ensuring seamless and secure financial transactions. In addition to these features, the system will provide comprehensive reporting and analytics tools, enabling airline management to monitor operations, identify trends, and make informed decisions to improve service quality and efficiency. Overall, the project is designed to deliver a scalable, user-friendly, and highly efficient reservation system that enhances the travel experience while reducing operational bottlenecks and improving the overall management of airline services.

# CHAPTER 2. DATABASE DESIGN

**2.1 List of Attributes, entities and relationship**

## Entity Name: Airline

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Airline\_ID | int (Primary Key) |
| Name | varchar(200) |
| Country | varchar(50) |
| Headquarters | varchar(100) |

## Entity Name: Airport

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Airport\_ID | int (Primary Key) |
| Name | varchar(100) |
| Country | varchar(50) |
| Code | varchar(10) |
| Location | varchar(100) |

## Entity Name: Aircraft

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Airline\_ID | int (Foreign Key) |
| Aircraft\_ID | int (Primary Key) |
| Capacity | int |
| Model | varchar(50) |
| Manufacturer | varchar(100) |

## Entity Name: Flight

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Flight\_ID | int (Primary Key) |
| Airline\_ID | int (Foreign Key) |
| Aircraft\_ID | int (Foreign Key) |
| Departuretime | datetime |
| Arrivaltime | datetime |

## Entity Name:Schedules

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Schedule\_ID | int (Primary Key) |
| AvailableSeats | int |
| Flight\_ID | int (Foreign Key) |
| Date | date |
| Price | decimal(10,2) |

## Entity Name: Passengers

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Passenger\_ID | int (Primary Key) |
| Passport\_number | varchar(50) (Unique) |
| Email | varchar(100) (Nullable) |
| Nationality | varchar(50) |
| Full\_name | varchar(100) |

## 

## Entity Name: Bookings

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Booking\_ID | int (Primary Key) |
| Passenger\_ID | int (Foreign Key) |
| Bookingstatus | varchar(20) |
| Flight\_ID | int (Foreign Key) |
| Booking\_date | datetime |
| Totalprice | decimal(10,2) |

## 

## Entity Name: Tickets

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Ticket\_ID | int (Primary Key) |
| Booking\_ID | int (Foreign Key) |
| Class | varchar(20) |
| Ticketstatus | varchar(20) |
| Seat\_number | varchar(10) |

## Entity Name: Payments

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Payment\_ID | int (Primary Key) |
| Booking\_ID | int (Foreign Key) |
| Amount | decimal(10,2) |
| Payment\_method | varchar(20) |
| Transaction\_details | varchar(50) |

## Entity Name:Employees

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Employee\_ID | int (Primary Key) |
| Designation | varchar(50) |
| Airline\_ID | int (Foreign Key) |
| Full\_name | varchar(100) |
| Contact\_details | varchar(50) |

## Entity Name: Pilot

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Pilot\_ID | int (Primary Key) |
| Employee\_ID | int (Foreign Key) |
| License\_number | varchar(50) |
| Experience\_in\_hrs | int |
| Type\_of\_pilot | varchar(50) |

## Entity Name: Luggage

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Luggage\_ID | int (Primary Key) |
| Passenger\_ID | int (Foreign Key) |
| Type\_of\_luggage | varchar(50) |
| Charges | decimal(5,2) |
| Weight | decimal(5,2) |

## Entity Name: BoardingPass

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Boardingpass\_ID | int (Primary Key) |
| Passenger\_ID | int (Foreign Key) |
| Flight\_ID | int (Foreign Key) |
| Boarding\_time | datetime |
| Gate\_ID | int (Foreign Key) |

## Entity Name: Gate

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Gate\_ID | int (Primary Key) |
| Bus | varchar(20) |
| Airport\_ID | int (Foreign Key) |
| Gate\_status | varchar(20) |
| Terminal | varchar(10) |

## Entity Name: Checkin

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Checkin\_ID | int (Primary Key) |
| Flight\_ID | int (Foreign Key) |
| Checkin\_status | varchar(20) |
| Checkin\_time | datetime |
| Passenger\_ID | int (Foreign Key) |

## Entity Name: SecurityCheck

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Securitycheck\_ID | int (Primary Key) |
| Passenger\_ID | int (Foreign Key) |
| Flight\_ID | int (Foreign Key) |
| Checked\_time | datetime |
| Status | varchar(20) |

## Entity Name: FlightEngineer

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Employee\_ID | int (Foreign Key) |
| Engineer\_ID | int (Primary Key) |
| Specialization | varchar(50) |
| Assigned\_Aircraft\_ID | int (Foreign Key) |
| Certification | varchar(50) |

## Entity Name: Weather

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Condition | varchar(255) |
| Weather\_ID | int (Primary Key) |
| Airport\_ID | int (Foreign Key) |
| Temperature | float |
| Flight\_ID | int (Foreign Key) |

## Entity Name: Cargo

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Cargo\_ID | int (Primary Key) |
| Flight\_ID | int (Foreign Key) |
| Weight | decimal(10,2) |
| Specialhandlingornot | varchar(4) |
| Type | varchar(50) |

## Entity Name: EmergencyLanding

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| AircraftDamageReport | varchar(200) |
| Dateandtime | datetime |
| Airport\_ID | int (Foreign Key) |
| Flight\_ID | int (Foreign Key) |
| Reason | varchar(200) |
| Landing\_ID | int (Primary Key) |

## Entity Name: MedicalCrew

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Employee\_ID | int (Foreign Key) |
| Medicalcrew\_ID | int (Primary Key) |
| Designation | varchar(100) |
| Experience | int |
| Assignedflight\_ID | int (Foreign Key) |

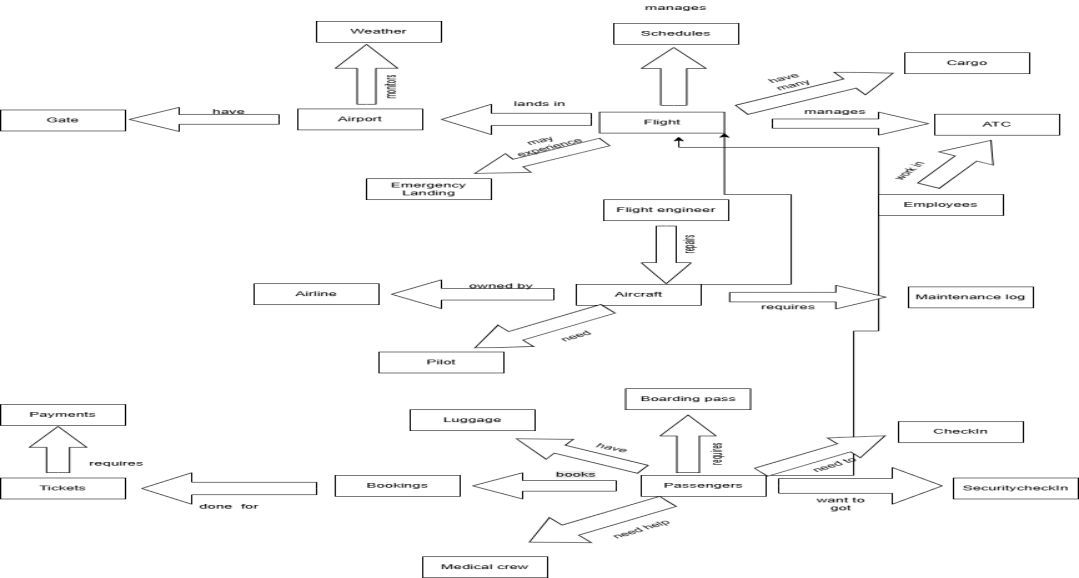
## Entity Name: MaintenanceLog

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Maintenance\_ID | int (Primary Key) |
| Aircraft\_ID | int (Foreign Key) |
| Engineer\_ID | int (Foreign Key) |
| Duration | int |
| Cost | decimal(10,2) |

## Entity Name: ATC

|  |  |
| --- | --- |
| **Attributes** | **Type** |
| Center\_ID | int (Primary Key) |
| Tracking\_ID | varchar(20) |
| Employee\_ID | int (Foreign Key) |
| Controller\_ID | varchar(20) |
| Designation | varchar(30) |

# 2.1.1 Entities and their relationships:

****

The above diagram is a simple representation of entities which shows the connectivity between allthe entities and the relationship between various entities

To know in detail about the types of relationships that exist between all the entities and to know the different attributes that describes about the entity we design ER(entity relation) diagram.

2.2 E-R Diagram

# Airline er

**CHAPTER 3. RELATIONAL MODEL**

# 3.1 Database languages

Four categories of database languages :

**1.** **Data definition language (DDL):**

Data definition language (DDL) creates the framework of the database by specifying the database schema, which is the structure that represents the organization of data. Its common uses include the creation and alteration of tables, files, indexes and columns within the database. This language also allows users to rename or drop the existing database or its components.

Here's a list of DDL statements:

• CREATE: Creates a new database or object, such as a table, index or column.

• ALTER: Changes the structure of the database or object.

• DROP: Deletes the database or existing objects.

• RENAME: Renames the database or existing objects.

**2.** **Data manipulation language (DML):**

Data manipulation language (DML) provides operations that handle user requests, offering a way to access and manipulate the data that users store within a database. Its common functions include inserting, updating and retrieving data from the database.

Here's a list of DML statements:

• INSERT: Adds new data to the existing database table.

• UPDATE: Changes or updates values in the table.

• DELETE: Removes records or rows from the table.

• SELECT: Retrieves data from the table or multiple tables.

**3. Data control language (DCL):**

Data control language (DCL) controls access to the data that users store within a database. Essentially, this language controls the rights and permissions of the database system. It allows users to grant or revoke privileges to the database.

Here's a list of DCL statements:

• GRANT: Gives a user access to the database.

• REVOKE: Removes a user's access to the database.

**4. Transaction control language (TCL):**

Transaction control language (TCL) manages the transactions within a database. Transactions group a set of related tasks into a single, executable task. All the tasks must succeed in order for the transaction to work. Here's a list of TCL statements:

• COMMIT: Carries out a transaction.

• ROLLBACK: Restores a transaction if any tasks fail to execute.

**3.2 Table Description**

Following are the tables along with constraints used in All in one travel booking

database.

1. **Airline:**  
   This table stores information about various airlines, including their names, countries of operation, and headquarters locations.  
   **Constraints:** Airline\_ID is the primary key, ensuring each airline is uniquely identified.
2. **Airport:**  
   This table holds details about airports, such as their names, countries, unique codes, and locations.  
   **Constraints:** Airport\_ID is the primary key.
3. **Aircraft:**  
   This table maintains data on aircraft, including model, capacity, and manufacturer, and it is linked to an airline.  
   **Constraints:** Aircraft\_ID is the primary key, and Airline\_ID serves as a foreign key referencing the Airline table.
4. **Flight:**  
   This table represents scheduled flights with information on departure and arrival times, the airline operating the flight, and the aircraft used.  
   **Constraints:** Flight\_ID is the primary key; both Airline\_ID and Aircraft\_ID are foreign keys.
5. **Schedules:**  
   This table stores scheduling details for flights, such as available seats, flight dates, and prices.  
   **Constraints:** Schedule\_ID is the primary key and Flight\_ID is a foreign key referencing the Flight table.
6. **Passengers:**  
   This table contains passenger information, including passport numbers, full names, nationalities, email addresses, and phone numbers.  
   **Constraints:** Passenger\_ID is the primary key and Passport\_number must be unique.
7. **Bookings:**  
   This table logs passenger bookings with details on booking status, booking dates, total prices, and the flights booked.  
   Constraints: Booking\_ID is the primary key, with Passenger\_ID and Flight\_ID as foreign keys.
8. **Tickets:**  
   This table stores details about tickets issued for bookings, including class, seat number, and ticket status.  
   **Constraints:** Ticket\_ID is the primary key, and Booking\_ID is a foreign key referencing the Bookings table.
9. **Payments:**  
   This table records payment transactions related to bookings, including payment amounts, methods, and transaction details.  
   **Constraints:** Payment\_ID is the primary key; Booking\_ID is a foreign key.
10. **Employees:**  
    This table holds employee data, covering details such as full names, designations, contact information, and the airline they are associated with.  
    **Constraints:** Employee\_ID is the primary key and Airline\_ID is a foreign key referencing the Airline table.
11. **Pilot:**  
    This table contains information on pilots, including their license numbers, experience in hours, and type of pilot.  
    **Constraints:** Pilot\_ID is the primary key and Employee\_ID is a foreign key linked to the Employees table.
12. **Luggage:**  
    This table tracks passenger luggage details such as type, weight, and any additional charges applied.  
    **Constraints:** Luggage\_ID is the primary key; Passenger\_ID is a foreign key referencing the Passengers table.
13. **BoardingPass:**  
    This table stores information about boarding passes issued to passengers, including boarding times and associated gate details.  
    **Constraints:** Boardingpass\_ID is the primary key, with Passenger\_ID, Flight\_ID, and Gate\_ID as foreign keys.
14. **Gate:**  
    This table contains gate information at airports, including bus service details, terminal numbers, and gate status.  
    **Constraints:** Gate\_ID is the primary key and Airport\_ID is a foreign key.
15. **Checkin:**  
    This table logs check-in events for passengers, recording check-in times and statuses for each flight.  
    **Constraints:** Checkin\_ID is the primary key; Passenger\_ID and Flight\_ID are foreign keys.
16. **SecurityCheck:**  
    This table maintains records of security checks conducted for passengers before boarding flights, including time and status.  
    **Constraints:** Securitycheck\_ID is the primary key, with Passenger\_ID and Flight\_ID as foreign keys.
17. **FlightEngineer:**  
    This table holds information about flight engineers, such as their specialization, certifications, and the aircraft they are assigned to maintain.  
    **Constraints:** Engineer\_ID is the primary key; Employee\_ID and Assigned\_Aircraft\_ID are foreign keys.
18. **Weather:**  
    This table stores weather information relevant to flight operations at specific airports, including temperature and conditions.  
    **Constraints:** Weather\_ID is the primary key; Airport\_ID and Flight\_ID are foreign keys.
19. **Cargo:**  
    This table records details about cargo carried on flights, including weight, type, and whether special handling is required.  
    **Constraints:** Cargo\_ID is the primary key, and Flight\_ID is a foreign key.
20. **EmergencyLanding:**  
    This table logs emergency landing events, including reasons, damage reports, and associated flight and airport information.  
    **Constraints:** Landing\_ID is used as the identifier, with Airport\_ID and Flight\_ID as foreign keys.
21. **MedicalCrew:**  
    This table contains details of medical personnel assigned to flights, including their designations and experience levels.  
    **Constraints:** Medicalcrew\_ID is the primary key; Employee\_ID and Assignedflight\_ID are foreign keys.
22. **MaintenanceLog:**  
    This table records logs of aircraft maintenance activities, noting duration, cost, and the engineer responsible.  
    **Constraints:** Maintenance\_ID is the primary key; Aircraft\_ID and Engineer\_ID are foreign keys.
23. **ATC (Air Traffic Control):**  
    This table stores information about air traffic controllers, including their tracking IDs and controller designations.  
    **Constraints:** Center\_ID is the primary key and Employee\_ID is a foreign key.

The above descripted information is the brief detailing about the entities

and relations and their attributes.

**3.3 Relational Database Scheme**

The relational database schema for ***All in one travel booking system***database is as follows:

1. **Airline** (Airline\_ID, Name, Country, Headquarters)
2. **Airport** (Airport\_ID, Name, Country, Code, Location)
3. **Aircraft** (Airline\_ID, Aircraft\_ID, Capacity, Model, Manufacturer)
4. **Flight** (Flight\_ID, Airline\_ID, Aircraft\_ID, Departuretime, Arrivaltime)
5. **Schedules** (Schedule\_ID, AvailableSeats, Flight\_ID, Date, Price)
6. **Passengers** (Passenger\_ID, Passport\_number, Email, Nationality, Full\_name, Phone\_number)
7. **Bookings** (Booking\_ID, Passenger\_ID, Bookingstatus, Flight\_ID, Booking\_date, Totalprice)
8. **Tickets** (Ticket\_ID, Booking\_ID, Class, Ticketstatus, Seat\_number)
9. **Payments** (Payment\_ID, Booking\_ID, Amount, Payment\_method, Transaction\_details)
10. **Employees** (Employee\_ID, Designation, Airline\_ID, Full\_name, Contact\_details)
11. **Pilot** (Pilot\_ID, Employee\_ID, Licensenumber, Experience\_in\_hrs, Type\_of\_pilot)
12. **Luggage** (Luggage\_ID, Passenger\_ID, Type\_of\_luggage, Charges, Weight)
13. **BoardingPass** (Boardingpass\_ID, Passenger\_ID, Flight\_ID, Boarding\_time, Gate\_ID)
14. **Gate** (Gate\_ID, Bus, Airport\_ID, Gatestatus, Terminal)
15. **Checkin** (Checkin\_ID, Flight\_ID, Checkinstatus, Checkintime, Passenger\_ID)
16. **SecurityCheck** (Securitycheck\_ID, Passenger\_ID, Flight\_ID, Checked\_time, Status)
17. **FlightEngineer** (Employee\_ID, Engineer\_ID, Specialization, Assigned\_Aircraft\_ID, Certification, Experience)
18. **Weather** (Condition, Weather\_ID, Airport\_ID, Temperature, Flight\_ID)
19. **Cargo** (Cargo\_ID, Flight\_ID, Weight, Specialhandlingornot, Type)
20. **EmergencyLanding** (AircraftDamageReport, Dateandtime, Airport\_ID, Flight\_ID, Reason, Landing\_ID)
21. **MedicalCrew** (Employee\_ID, Medicalcrew\_ID, Designation, Experience, Assignedflight\_ID)
22. **MaintenanceLog** (Maintenance\_ID, Aircraft\_ID, Engineer\_ID, Duration, Cost, Log\_ID)
23. **ATC** (Center\_ID, Tracking\_ID, Employee\_ID, Controller\_ID, Designation)

**3.4 Relational Queries**

create table Airline(

Airline\_ID int primary key,

Name varchar(200),

Country varchar(50),

Headquarters varchar(100),

);

INSERT INTO Airline (Airline\_ID, Name, Country, Headquarters) VALUES

(1, 'SkyHigh Airways', 'USA', 'New York');

create table Airport(

Airport\_ID int primary key,

Name varchar(100),

Country varchar(50),

Code varchar(10),

Location varchar(100)

);

INSERT INTO Airport (Airport\_ID, Name, Country, Code, Location) VALUES

(1, 'JFK International', 'USA', 'JFK', 'New York'),

create table Aircraft(

Airline\_ID int,

Aircraft\_ID int primary key,

Capacity int,

Model varchar(50),

Manufacturer varchar(100),

foreign key (Airline\_ID) references Airline(Airline\_ID)

);

INSERT INTO Aircraft (Aircraft\_ID, Airline\_ID, Capacity, Model, Manufacturer) VALUES

(1, 1, 180, 'Boeing 737', 'Boeing'),

create table Flight (

Flight\_ID int primary key,

Airline\_ID int,

Aircraft\_ID int,

Departuretime Datetime,

Arrivaltime Datetime,

foreign key (Airline\_ID) references Airline(Airline\_ID),

foreign key (Aircraft\_ID) references Aircraft(Aircraft\_ID)

);

INSERT INTO Flight (Flight\_ID, Airline\_ID, Aircraft\_ID, DepartureTime, ArrivalTime) VALUES

(1, 1, 1, '2025-01-01 08:00:00', '2025-01-01 12:00:00'),

create table Schedules(

Schedule\_ID int primary key,

AvailableSeats int,

Flight\_Id int,

Date date,

Price decimal(10,2),

foreign key (Flight\_ID) references Flight(Flight\_ID)

);

INSERT INTO Schedules (Schedule\_ID, AvailableSeats, Flight\_ID, Date, Price) VALUES

(1, 150, 1, '2025-01-01', 199.99),

create table Passengers (

Passenger\_ID int primary key,

Passport\_number varchar(50) unique,

email varchar(100) null,

Nationality varchar(50),

Full\_name varchar(100),

phone\_number varchar(15)

);

INSERT INTO Passengers (Passenger\_ID, Passport\_number, email, Nationality, Full\_name, Phone\_number) VALUES

(1, 'P10001', 'john.doe@example.com', 'USA', 'John Doe', '555-0101'),

create table bookings (

Booking\_ID int primary key,

Passenger\_ID int,

bookingstatus varchar(20),

Flight\_ID int,

Booking\_date datetime,

Totalprice decimal(10,2),

foreign key (Passenger\_ID) references Passengers(Passenger\_ID),

foreign key (Flight\_ID) references Flight(Flight\_ID)

);

INSERT INTO Bookings (Booking\_ID, Passenger\_ID, bookingstatus, Flight\_ID, Booking\_date, TotalPrice) VALUES

(1, 1, 'Confirmed', 1, '2024-12-20 10:00:00', 199.99),

create table tickets (

Ticket\_ID int primary key,

Booking\_ID int,

Class varchar(20),

Ticketstatus varchar(20),

Seat\_number varchar(10),

foreign key (Booking\_ID) references bookings(Booking\_ID),

);

INSERT INTO Tickets (Ticket\_ID, Booking\_ID, Class, Ticketstatus, Seat\_number) VALUES

(1, 1, 'Economy', 'Active', '12A'),

create table payments (

Payment\_ID int primary key,

Booking\_ID int,

Amount decimal(10,2),

Payment\_method varchar(20),

Transcation\_details varchar(50),

foreign key (Booking\_ID) references bookings(Booking\_ID)

);

INSERT INTO Payments (Payment\_ID, Booking\_ID, Amount, Payment\_method, Transcation\_details) VALUES

(1, 1, 199.99, 'Credit Card', 'TXN1001'),

create table employees(

Employee\_ID int primary key,

Designation varchar(50),

Airline\_ID int,

Full\_name varchar(100),

Contact\_details varchar(50),

foreign key (Airline\_ID) references Airline(Airline\_ID),

);

INSERT INTO Employees (Employee\_ID, Designation, Airline\_ID, Full\_name, Contact\_details) VALUES

(1, 'Pilot', 1, 'Captain John Miller', 'john.miller@email.com'),

create table pilot (

Pilot\_ID int primary key,

Employee\_ID int,

Licensenumber varchar(50),

Experience\_in\_hrs int,

type\_of\_pilot varchar(50),

foreign key (Employee\_ID) references employees(Employee\_ID)

);

INSERT INTO Pilot (Pilot\_ID, Employee\_ID, Licensenumber, Experience\_in\_hrs, type\_of\_pilot) VALUES

(1, 1, 'L1001', 5000, 'Captain'),

create table Luggage (

Luggage\_ID int primary key,

Passenger\_ID int,

Type\_of\_luggage varchar(50),

Charges decimal(5,2),

Weight decimal(5,2),

foreign key (Passenger\_ID) references passengers(Passenger\_ID)

);

INSERT INTO Luggage (Luggage\_ID, Passenger\_ID, Type\_of\_luggage, Charges, Weight) VALUES

(1, 1, 'Suitcase', 25.00, 15.5),

create table boardingpass(

Boardingpass\_ID int primary key,

Passenger\_ID int,

Flight\_ID int,

Boarding\_time datetime,

Gate\_id int,

foreign key (Passenger\_ID) references passengers(Passenger\_ID),

foreign key (Flight\_ID) references Flight(Flight\_ID),

foreign key (Gate\_id) references gate(gate\_ID)

);

INSERT INTO BoardingPass (BoardingPass\_ID, Passenger\_ID, Flight\_ID, Boarding\_time, Gate\_iD) VALUES

(1, 1, 1, '2025-01-01 07:30:00', 1),

create table gate(

gate\_ID int primary key,

Bus varchar(20),

Airport\_ID int,

gatestatus varchar(20),

Terminal varchar(10),

foreign key (Airport\_ID) references Airport(Airport\_ID)

);

INSERT INTO Gate (Gate\_ID, Bus, Airport\_ID, gatestatus, Terminal) VALUES

(1, 'Bus 1', 1, 'Open', 'T1'),

create table checkin (

Checkin\_id int primary key,

Flight\_id int,

checkinstatus varchar(20),

Checkintime datetime,

Passenger\_id int,

foreign key (Flight\_id) references Flight(Flight\_ID),

foreign key (Passenger\_id) references passengers(Passenger\_ID)

);

INSERT INTO CheckIn (Checkin\_id, Flight\_id, checkinstatus, checkintime, Passenger\_id) VALUES

(1, 1, 'CheckedIn', '2025-01-01 07:00:00', 1),

create table securitycheck (

Securitycheck\_id int primary key,

Passenger\_id int,

Flight\_id int,

Checked\_time datetime,

Status varchar(20),

foreign key (Passenger\_id) references passengers(Passenger\_ID),

foreign key (Flight\_id) references Flight(Flight\_ID)

);

INSERT INTO SecurityCheck (SecurityCheck\_id, Passenger\_id, Flight\_id, Checked\_time, Status) VALUES

(1, 1, 1, '2025-01-01 06:30:00', 'Cleared'),

create table flightengineer(

Employee\_id int,

Engineer\_id int primary key,

Specialization varchar(50),

Assigned\_Aircraft\_id int,

Certification varchar(50),

experience int,

foreign key (Employee\_id) references employees(Employee\_ID),

foreign key (Assigned\_Aircraft\_id) references Aircraft(Aircraft\_ID)

);

INSERT INTO FlightEngineer (Engineer\_id, Employee\_id, Specialization, Assigned\_Aircraft\_id, Certification, experience) VALUES

(1, 2, 'Engine', 1, 'Cert-A', 1200),

create table weather (

Condition varchar(255),

Weather\_id int primary key,

Airport\_id int,

Temperature float,

Flight\_id int,

foreign key (Airport\_id) references Airport(Airport\_ID),

foreign key (Flight\_id) references Flight(Flight\_ID)

);

INSERT INTO Weather (Weather\_id, Condition, Airport\_id, Temperature, Flight\_id) VALUES

(1, 'Sunny', 1, 25.0, 1),

create table cargo (

Cargo\_id int primary key,

flight\_id int,

weight decimal(10,2),

Specialhandlingornot varchar(4) ,

Type varchar(50),

foreign key (flight\_id) references Flight(Flight\_ID)

);

INSERT INTO Cargo (Cargo\_id, Flight\_id, Weight, Specialhandlingornot, Type) VALUES

(1, 1, 500.00, 1, 'Perishable'),

create table emergencylanding (

AircraftDamageReport varchar(200),

Dateandtime datetime,

Airport\_ID int,

Flight\_ID int,

Reason varchar(200),

landing\_id int,

foreign key (Airport\_ID) references Airport(Airport\_ID),

foreign key (Flight\_ID) references Flight(Flight\_ID)

);

INSERT INTO EmergencyLanding (Landing\_id, AircraftDamageReport, Dateandtime, Airport\_ID, Flight\_ID, Reason) VALUES

(1, 'Minor engine failure', '2025-01-01 11:30:00', 1, 1, 'Engine malfunction'),

create table medicalcrew (

Employee\_id int,

Medicalcrew\_id int primary key,

designation varchar(100),

Experience int,

Assignedflight\_id int,

foreign key (Assignedflight\_id) references Flight(Flight\_ID),

foreign key (Employee\_id) references employees(Employee\_ID)

);

INSERT INTO MedicalCrew (MedicalCrew\_id, Employee\_id, designation, Experience, Assignedflight\_id) VALUES

(1, 3, 'Doctor', 10, 1),

create table maintenancelog (

Maintenance\_id int primary key,

Aircraft\_id int,

Engineer\_id int,

Duration int,

Cost decimal(10,2),

log\_id int,

foreign key (Aircraft\_id) references Aircraft(Aircraft\_ID)

);

INSERT INTO MaintenanceLog (Maintenance\_id, Aircraft\_id, Engineer\_id, Duration, Cost, log\_id) VALUES

(1, 1, 1, 5, 1500.00, 101),

create table atc(

Center\_ID int primary key,

Tracking\_Id varchar(20),

employee\_id int,

Controller\_id varchar(20),

designation varchar(30),

foreign key (employee\_id) references employees(Employee\_id)

);

INSERT INTO ATC (Center\_ID, Tracking\_Id, Employee\_id, Controller\_id, designation) VALUES

(1, 'TRK1', 4, 'CTRL1001', 'Senior Controller'),

**SQL QUERIES:**

**Query1: Create a new table called DelayedFlights to track flight delays. It should store the Flight\_ID (as primary key), delay duration in minutes, and reason for the delay.**

CREATE TABLE DelayedFlights (

Flight\_ID INT PRIMARY KEY,

DelayMinutes INT,

Reason VARCHAR(200)

);

**Query2:How would you populate a VIP\_Passengers table with all passengers from the United States using data from the existing Passengers table?**

INSERT INTO VIP\_Passengers

SELECT \* FROM Passengers

WHERE Nationality = 'USA';

**Query3:Add a new column called CEO to the Airline table to store the CEO's name as a VARCHAR(100).**

ALTER TABLE Airline ADD CEO VARCHAR(100);

**Query4Remove the Headquarters column from the Airline table.**

ALTER TABLE Airline DROP COLUMN Headquarters;

**Query5:Ensure the Capacity column in the Aircraft table is explicitly defined as an integer type.**

ALTER TABLE Aircraft ALTER COLUMN Capacity INT;

**Query6:Delete all records from the Aircraft table while retaining its structure.**

TRUNCATE TABLE Aircraft;

**Query7:Clear all data from the Flight table without dropping the table itself.**

TRUNCATE TABLE Flight;

**Query8:Remove all entries from the Schedules table in one operation.**

TRUNCATE TABLE Schedules;

**Query9:Delete all employee records from the Employees table immediately.**

TRUNCATE TABLE Employees;

**Query10:Add a DateOfBirth column to the Passengers table to store birthdates.**

ALTER TABLE Passengers ADD DateOfBirth DATE;

**Query11:Add a FoundedYear column to the Airline table to track the year each airline was established.**

ALTER TABLE Airline ADD FoundedYear INT;

**Query12:Reintroduce the Headquarters column (VARCHAR(100)) to the Airline table after accidental removal.**

ALTER TABLE Airline ADD Headquarters VARCHAR(100);

**Query13:Update the headquarters location of the airline with Airline\_ID = 1 to "Los Angeles".**

UPDATE Airline

SET Headquarters = 'Los Angeles'

WHERE Airline\_ID = 1;

**Query14:Change the contact details of the employee with Employee\_ID = 4 to "new.contact@example.com".**

UPDATE Employees

SET Contact\_details = 'new.contact@example.com'

WHERE Employee\_ID = 4;

**Query15:Update the seat number for the ticket with Ticket\_ID = 1 to "14B".**

UPDATE Tickets

SET Seat\_number = '14B'

WHERE Ticket\_ID = 1;

**Query16:Delay the departure time of Flight\_ID = 3 by 1 hour.**

UPDATE Flight

SET DepartureTime = DATEADD(hour, 1, DepartureTime)

WHERE Flight\_ID = 3;

**Query17:Remove all tickets marked with a "Cancelled" status from the Tickets table.**

DELETE FROM Tickets

WHERE Ticketstatus = 'Cancelled';

**Query18:Delete payment records where the amount exceeds $280.**

DELETE FROM Payments

WHERE Amount > 280;

**Query19:Remove all employees with the designation "Maintenance" from the Employees table.**

DELETE FROM Employees

WHERE Designation = 'Maintenance';

**Query20L:ist all unique countries where airlines in the Airline table are based.**

SELECT DISTINCT Country FROM Airline;

**Query21:Retrieve all distinct airport codes from the Airport table.**

SELECT DISTINCT Code FROM Airport;

**Query22:List all distinct travel classes available in the Tickets table.**

SELECT DISTINCT Class FROM Tickets;

**Query23 :Identify all unique aircraft manufacturers from the Aircraft table.**

SELECT DISTINCT Manufacturer FROM Aircraft;

**Query24 :List all unique travel classes from Tickets**

SELECT DISTINCT Class FROM Tickets;

**Query25:Show all distinct booking statuses from Bookings**

SELECT DISTINCT bookingstatus FROM Bookings;

**Query26:Display distinct security check statuses**

SELECT DISTINCT Status FROM SecurityCheck;

**Query27 :Show unique flight engineer specializations**

SELECT DISTINCT Specialization FROM FlightEngineer;

**Query28:List all passenger nationalities**

SELECT DISTINCT Nationality FROM Passengers;

**Query29:Find airlines with names starting with "Sky"**

SELECT \* FROM Airline

WHERE Name LIKE 'Sky%';

**Query30:Get airports with "International" in their name**

SELECT \* FROM Airport

WHERE Name LIKE '%International';

**Query31:Find passengers using @example.com emails**

SELECT \* FROM Passengers

WHERE email LIKE ['%@example.com';](mailto:'%25@example.com';)

**Query32:Locate cargo entries ending with "tronics" (Electronics)**

SELECT \* FROM Cargo

WHERE Type LIKE '%tronics';

**Query33:Show weather conditions ending with "y" (Sunny/Rainy etc.)**

SELECT \* FROM Weather

WHERE Condition LIKE '%y';

**Query34:Find flights departing in first 9 days of Jan 2025**

SELECT \* FROM Flight

WHERE DepartureTime LIKE '2025-01-0%';

**Query35:Get pilots with license numbers starting with "L10"**

SELECT \* FROM Pilot

WHERE Licensenumber LIKE 'L10%';

**Query36:Find employees named with "J..."**

SELECT \* FROM Employees

WHERE Full\_name LIKE 'J%';

**Query37:Show tickets for seats in rows 10-19**

SELECT \* FROM Tickets

WHERE Seat\_number LIKE '1%';

**Query38:Find bookings starting with "Con" (Confirmed)**

SELECT \* FROM Bookings

WHERE bookingstatus LIKE 'Con%';

**Query39:Create various views for simplified data access:Airline names**

CREATE VIEW v\_AirlineNames AS

SELECT Airline\_ID, Name FROM Airline;

**Query40:Create various views for simplified data access:Flight timings**

CREATE VIEW v\_FlightTimes AS

SELECT Flight\_ID, DepartureTime, ArrivalTime FROM Flight;

**Query41:Create various views for simplified data access:Passenger contact info**

CREATE VIEW v\_PassengerContacts AS

SELECT Passenger\_ID, Full\_name, email, phone\_number FROM Passengers;

**Query42:Create various views for simplified data access:Booking summaries**

CREATE VIEW v\_BookingSummary AS

SELECT Booking\_ID, Passenger\_ID, Flight\_ID, bookingstatus, TotalPrice

FROM Bookings;

**Query43:Create various views for simplified data access:Aircraft specs**

CREATE VIEW v\_AircraftDetails AS

SELECT Aircraft\_ID, Model, Manufacturer, Capacity FROM Aircraft;

**Query44:Create various views for simplified data access:Airport codes**

CREATE VIEW v\_AirportCodes AS

SELECT Airport\_ID, Name, Code FROM Airport;

**Query45:Create various views for simplified data access:Employee details**

CREATE VIEW v\_EmployeeInfo AS

SELECT Employee\_ID, Full\_name, Designation, Contact\_details FROM Employees;

**Query46:Create various views for simplified data access:Pilot credentials**

CREATE VIEW v\_PilotDetails AS

SELECT Pilot\_ID, Employee\_ID, Licensenumber, Experience\_in\_hrs FROM Pilot;

**Query47:Create various views for simplified data access:Payment records**

CREATE VIEW v\_PaymentSummary AS

SELECT Payment\_ID, Booking\_ID, Amount, Payment\_method FROM Payments;

**Query48:Create various views for simplified data access:Schedule pricing**

CREATE VIEW v\_SchedulePricing AS

SELECT Schedule\_ID, Flight\_ID, Price FROM Schedules;

**Query49:Count total airlines**

SELECT COUNT(\*) AS TotalAirlines FROM Airline;

**Query50:Calculate average aircraft capacity**

SELECT AVG(Capacity) AS AvgAircraftCapacity FROM Aircraft;

**Query51:Find lowest flight price**

SELECT MIN(Price) AS LowestPrice FROM Schedules;

**Query52:Find highest flight price**

SELECT MAX(Price) AS HighestPrice FROM Schedules;

**Query53:Calculate total payments received**

SELECT SUM(Amount) AS TotalPayments FROM Payments;

**Query54:Count unique airport countries**

SELECT COUNT(DISTINCT Country) AS UniqueCountries FROM Airport;

**Query55:Average pilot flight hours**

SELECT AVG(Experience\_in\_hrs) AS AvgPilotExperience FROM Pilot;

**Query56:Find most expensive booking**

SELECT MAX(TotalPrice) AS HighestBooking FROM Bookings;

**Query57:Find minimum luggage charge**

SELECT MIN(Charges) AS LowestLuggageCharge FROM Luggage;

**Query58:Calculate total cargo weight**

SELECT SUM(Weight) AS TotalCargoWeight FROM Cargo;

**Query59:Total capacity per aircraft manufacturer**

SELECT Manufacturer, SUM(Capacity) AS TotalCapacity

FROM Aircraft

GROUP BY Manufacturer;

**Query60:Flight count per airline**

SELECT Airline\_ID, COUNT(\*) AS TotalFlights

FROM Flight

GROUP BY Airline\_ID;

**Query61:Airline count per country**

SELECT Country, COUNT(\*) AS TotalAirlines

FROM Airline

GROUP BY Country;

**Query62:Flight frequency by departure time**

SELECT DepartureTime, COUNT(\*) AS FlightsPerTime

FROM Flight

GROUP BY DepartureTime

HAVING COUNT(\*) > 0;

**Query63:Average capacity per aircraft model**

SELECT Model, AVG(Capacity) AS AvgCapacity

FROM Aircraft

GROUP BY Model;

**Query64:Booking status distribution**

SELECT bookingstatus, COUNT(\*) AS StatusCount

FROM Bookings

GROUP BY bookingstatus;

**Query65:Passenger count by nationality**

SELECT Nationality, COUNT(\*) AS PassengerCount

FROM Passengers

GROUP BY Nationality;

**Query66:Payment totals by method**

SELECT Payment\_method, SUM(Amount) AS TotalAmount

FROM Payments

GROUP BY Payment\_method;

**Query67:Employee count by role**

SELECT Designation, COUNT(\*) AS EmployeeCount

FROM Employees

GROUP BY Designation;

**Query68:Gate count per airport (with >1 gate)**

SELECT Airport\_ID, COUNT(\*) AS TotalGates

FROM Gate

GROUP BY Airport\_ID

HAVING COUNT(\*) > 1;

**Query69:Find US airlines based in New York**

SELECT \* FROM Airline

WHERE Country = 'USA' AND Headquarters = 'New York';

**Query70:Show airports in UK or USA**

SELECT \* FROM Airport

WHERE Country = 'UK' OR Country = 'USA';

**Query71:Finds Canadian passengers who provided email addresses**

SELECT \* FROM Passengers

WHERE Nationality = 'Canada' AND email IS NOT NULL;

**Query72:Retrieves confirmed bookings over $200.**

SELECT \* FROM Bookings

WHERE bookingstatus = 'Confirmed' AND TotalPrice > 200;

**Query73:Shows active tickets or first-class reservations.**

SELECT \* FROM Tickets

WHERE Ticketstatus = 'Active' OR Class = 'First';

**Query74:Non-Canceled Tickets**

SELECT \* FROM tickets

WHERE Ticketstatus = 'Active' OR Ticketstatus = 'Pending';

**Query75:Finds pilots using formal email addresses.**

SELECT \* FROM Employees

WHERE Designation = 'Pilot' AND Contact\_details LIKE ['%@email.com' ;](mailto:'%25@email.com';)

**Query76:Flights between Jan 5-20, 2025.**

SELECT \* FROM Flight

WHERE DepartureTime > '2025-01-05' AND ArrivalTime < '2025-01-20';

**Query77:Filters credit/debit/online payments.**

SELECT \* FROM Payments

WHERE NOT Payment\_method = 'Cash';

**Query78:Prioritizes availability or budget options.**

SELECT \* FROM Schedules

WHERE AvailableSeats >= 150 OR Price < 220;

**Query79:Identifies standard suitcase pricing**.

SELECT \* FROM Luggage

WHERE Type\_of\_luggage = 'Suitcase' AND Charges = 25.00;

**Query80:Extracts date portion from datetime.**

SELECT Flight\_ID, CONVERT(DATE, DepartureTime) AS DepDate

FROM Flight;

**Query81:Shows time component of departures.**

SELECT Flight\_ID, CONVERT(TIME, DepartureTime) AS DepTime

FROM Flight;

**Query82:Isolates year from datetime (all 2025).**

SELECT Flight\_ID, YEAR(DepartureTime) AS DepYear

FROM Flight;

**Query83:Standardizes date display format.**

SELECT Schedule\_ID, FORMAT(Date, 'yyyy-MM-dd') AS FormattedDate

FROM Schedules;

**Query84:Lists airlines with scheduled flights**

SELECT \* FROM Airline

WHERE Airline\_ID IN (SELECT Airline\_ID FROM Flight);

**Query85:Finds name associated with first booking.**

SELECT Full\_name FROM Passengers

WHERE Passenger\_ID = (SELECT Passenger\_ID FROM Bookings WHERE Booking\_ID = 1);

**Query86:Identifies airline with earliest departure.**

SELECT Airline\_ID, Name FROM Airline

WHERE Airline\_ID = (SELECT TOP 1 Airline\_ID FROM Flight ORDER BY DepartureTime);

**Query87:Finds airports missing gate data.**

SELECT \* FROM Airport

WHERE Airport\_ID NOT IN (SELECT Airport\_ID FROM Gate);

**Query88:Lists flights using Boeing 737 aircraft.**

SELECT \* FROM Flight

WHERE Aircraft\_ID = (SELECT Aircraft\_ID FROM Aircraft WHERE Model = 'Boeing 737');

**Query89:High-cost bookings compared to average.**

SELECT Booking\_ID, TotalPrice FROM Bookings

WHERE TotalPrice > (SELECT AVG(TotalPrice) FROM Bookings);

**Query90:Pilots with 5000+ flight hours.**

SELECT E.\*

FROM Employees E

JOIN Pilot P ON E.Employee\_ID = P.Employee\_ID

WHERE P.Experience\_in\_hrs > 5000;

**Query91:Flight with the earliest departure time**

SELECT Flight\_ID, DepartureTime FROM Flight

WHERE DepartureTime = (SELECT MIN(DepartureTime) FROM Flight);

**Query92:Flights with engine-related emergencies.**

SELECT \* FROM Flight

WHERE Flight\_ID IN (

SELECT Flight\_ID FROM EmergencyLanding WHERE Reason = 'Engine malfunction'

);

**Query93:Matches airlines to their flight times.**

SELECT a.Name, f.DepartureTime

FROM Airline a

INNER JOIN Flight f ON a.Airline\_ID = f.Airline\_ID;

**Query94:Shows passenger names with booking amounts.**

SELECT p.Full\_name, b.TotalPrice

FROM Passengers p

JOIN Bookings b ON p.Passenger\_ID = b.Passenger\_ID;

**Query95: Lists aircraft models used for flights.**

SELECT f.Flight\_ID, a.Model

FROM Flight f

INNER JOIN Aircraft a ON f.Aircraft\_ID = a.Aircraft\_ID;

**Query96:Includes bookings without seats (NULL).**

SELECT b.Booking\_ID, t.Seat\_number

FROM Bookings b

LEFT JOIN Tickets t ON b.Booking\_ID = t.Booking\_ID;

**Query97:Employee names with pilot licenses.**

SELECT e.Full\_name, p.Licensenumber

FROM Employees e

INNER JOIN Pilot p ON e.Employee\_ID = p.Employee\_ID;

**Query98:Combines schedule IDs with flight times.**

SELECT s.Schedule\_ID, f.DepartureTime

FROM Schedules s

INNER JOIN Flight f ON s.Flight\_ID = f.Flight\_ID;

**Query99:Shows terminal locations for gates.**

SELECT ar.Name, g.Terminal

FROM Airport ar

JOIN Gate g ON ar.Airport\_ID = g.Airport\_ID;

**Query100:Retrieves emails associated with bookings.**

SELECT b.Booking\_ID, p.email

FROM Bookings b

JOIN Passengers p ON b.Passenger\_ID = p.Passenger\_ID;

**Query101:Which flights encountered what weather conditions during their journeys?**

SELECT f.Flight\_ID, w.Condition

FROM Flight f

JOIN Weather w ON f.Flight\_ID = w.Flight\_id;

**Query102:Which cargo shipments are associated with specific flights?**

SELECT c.Cargo\_id, f.Flight\_ID

FROM Cargo c

JOIN Flight f ON c.Flight\_id = f.Flight\_ID;

**Query103:Which flights have both scheduled departures and entries in the flight schedule system?**

SELECT Flight\_ID FROM Flight

INTERSECT

SELECT Flight\_ID FROM Schedules;

**Query104:Which airlines have actually operated flights?**

SELECT Airline\_ID FROM Airline

INTERSECT

SELECT Airline\_ID FROM Flight;

**Query105:Which employees are qualified pilots?**

SELECT Employee\_ID FROM employees

INTERSECT

SELECT Employee\_ID FROM pilot;

**Query106:Which bookings have corresponding ticket records?**

SELECT Booking\_ID FROM Bookings

INTERSECT

SELECT Booking\_ID FROM Tickets;

**Query107 :Which passengers have checked luggage?**

SELECT Passenger\_ID FROM Passengers

INTERSECT

SELECT Passenger\_ID FROM Luggage;

**Query108:Which airports have designated gates?**

SELECT Airport\_ID FROM Airport

INTERSECT

SELECT Airport\_ID FROM Gate;

**Query109:Create a unified list of airlines and airports with type identifiers.**

SELECT Airline\_ID AS ID, Name, 'Airline' AS Source

FROM Airline

UNION

SELECT Airport\_ID AS ID, Name, 'Airport' AS Source

FROM Airport;

**Query110:List passengers from either USA or Canada.**

SELECT Passenger\_ID, Full\_name, Nationality

FROM Passengers

WHERE Nationality = 'USA'

UNION

SELECT Passenger\_ID, Full\_name, Nationality

FROM Passengers

WHERE Nationality = 'Canada';

**Query111:Combine flight IDs from both flight records and schedules.**

SELECT Flight\_ID FROM Flight

UNION

SELECT Flight\_ID FROM Schedules;

**Query112:Show top 3 airlines and airports alphabetically in a single list.**

SELECT \* FROM (

SELECT TOP 3 Airline\_ID AS ID, Name, 'Airline' AS Source

FROM Airline

ORDER BY Name

) AS a

UNION

SELECT \* FROM (

SELECT TOP 3 Airport\_ID AS ID, Name, 'Airport' AS Source

FROM Airport

ORDER BY Name

) AS b;

**Query113:What are the 3 highest payment amounts processed?**

SELECT TOP 3 \*

FROM Payments

ORDER BY Amount DESC;

**Query114:List the first 5 airlines alphabetically.**

SELECT TOP 5 \*

FROM Airline

ORDER BY Name ASC;

**Query115:Who are the top 5 passengers by number of bookings?**

SELECT TOP 5 Passenger\_ID, COUNT(\*) AS BookingCount

FROM Bookings

GROUP BY Passenger\_ID

ORDER BY BookingCount DESC;

**Query116:Which 5 flights have the most available seats?**

SELECT TOP 5 f.Flight\_ID, s.AvailableSeats

FROM Flight f

JOIN Schedules s ON f.Flight\_ID = s.Flight\_ID

ORDER BY s.AvailableSeats DESC;

**Query117:Which aircraft are maintained by engineers with 1500+ hours experience?**

SELECT A.\*

FROM Aircraft A

JOIN flightengineer FE ON A.Aircraft\_ID = FE.Assigned\_Aircraft\_id

WHERE FE.experience > 1500;

**Query118:Find flights after Jan 1, 2025 using aircraft with 150+ capacity.**

SELECT f.Flight\_ID, f.DepartureTime, a.Capacity

FROM Flight f

JOIN Aircraft a ON f.Aircraft\_ID = a.Aircraft\_ID

WHERE f.DepartureTime > '2025-01-01' AND a.Capacity > 150;

**Query119:List US-based pilots/flight engineers with their airline.**

SELECT e.Employee\_ID, e.Full\_name, e.Designation, a.Name AS Airline

FROM Employees e

JOIN Airline a ON e.Airline\_ID = a.Airline\_ID

WHERE a.Country = 'USA'

AND e.Designation IN ('Pilot', 'Flight Engineer');

**Query120:How many bookings does each airline have?**

SELECT a.Name AS Airline, COUNT(b.Booking\_ID) AS TotalBookings

FROM Bookings b

JOIN Flight f ON b.Flight\_ID = f.Flight\_ID

JOIN Airline a ON f.Airline\_ID = a.Airline\_ID

GROUP BY a.Name;

**Query121:Show confirmed bookings with passenger, flight, and airline details.**

SELECT p.Full\_name, b.bookingstatus, f.DepartureTime, a.Name AS Airline

FROM Passengers p

JOIN Bookings b ON p.Passenger\_ID = b.Passenger\_ID

JOIN Flight f ON b.Flight\_ID = f.Flight\_ID

JOIN Airline a ON f.Airline\_ID = a.Airline\_ID

WHERE b.bookingstatus = 'Confirmed';

**Query122:List pilots with 5000+ flight hours and their licenses.SELECT p.Pilot\_ID, e.Full\_name,** p.Experience\_in\_hrs, p.Licensenumber

FROM Pilot p

JOIN Employees e ON p.Employee\_ID = e.Employee\_ID

WHERE p.Experience\_in\_hrs > 5000;

**Query123:Show emergency landings with flight departure details.**

SELECT e.landing\_id, e.AircraftDamageReport, e.Dateandtime, e.Reason, f.Flight\_ID, f.DepartureTime

FROM EmergencyLanding e

JOIN Flight f ON e.Flight\_ID = f.Flight\_ID;

**Query124:List all passenger contact information.**

SELECT Passenger\_ID, Full\_name,email

FROM Passengers;

**Query125 :Create formatted passenger details (Name - Nationality).**

SELECT Passenger\_ID, CONCAT(Full\_name, ' - ', Nationality) AS PassengerDetail

FROM Passengers;

**Query126 :Show airports sorted by country and name.**

SELECT Airport\_ID, Name, Country, Code

FROM Airport

ORDER BY Country, Name;

**Query127:Find luggage weighing 10-15 kg.**

SELECT \* FROM Luggage

WHERE Weight BETWEEN 10 AND 15;

**Query128:Identify employees who are pilots using existence check.**

SELECT \* FROM employees e

WHERE EXISTS (

SELECT 1 FROM pilot p

WHERE p.Employee\_ID = e.Employee\_ID

);

**Query129:Find tickets for bookings above average price.**

SELECT \* FROM tickets

WHERE Booking\_ID IN (

SELECT Booking\_ID FROM bookings

WHERE TotalPrice > (SELECT AVG(TotalPrice) FROM bookings)

);

**Query130:How would you remove the passenger contact view?**

DROP VIEW IF EXISTS v\_PassengerContacts;

**Query131:Create a view for flight engineer certifications.**

CREATE VIEW v\_FlightEngineerInfo AS

SELECT Engineer\_id, Employee\_id, Specialization, Certification

FROM flightengineer;

**Query132:Show flights with aircraft models and weather conditions.**

SELECT f.Flight\_ID, a.Model, w.Condition

FROM Flight f

JOIN Aircraft a ON f.Aircraft\_ID = a.Aircraft\_ID

JOIN Weather w ON f.Flight\_ID = w.Flight\_id;

**Query133:How many flights has each airline operated?**

SELECT a.Name, COUNT(f.Flight\_ID) AS FlightCount

FROM Airline a

JOIN Flight f ON a.Airline\_ID = f.Airline\_ID

GROUP BY a.Name;

**Query134:List pilots whose license numbers start with "L10" along with their full name and associated airline name**

SELECT E.Full\_name, P.Licensenumber, A.Name AS AirlineName

FROM Pilot P

JOIN Employees E ON P.Employee\_ID = E.Employee\_ID

JOIN Airline A ON E.Airline\_ID = A.Airline\_ID

WHERE P.Licensenumber LIKE 'L10%';

**Query135:List each airline (whose name starts with "Sky") along with the total number of flights they operate**

SELECT A.Name, COUNT(F.Flight\_ID) AS FlightCount

FROM Airline A

JOIN Flight F ON A.Airline\_ID = F.Airline\_ID

WHERE A.Name LIKE 'Sky%'

GROUP BY A.Name;

**Query136:Retrieve employees who are pilots and whose experience is greater than the average experience of all pilots:**

SELECT E.Employee\_ID, E.Full\_name, P.Experience\_in\_hrs

FROM Employees E

JOIN Pilot P ON E.Employee\_ID = P.Employee\_ID

WHERE P.Experience\_in\_hrs > (SELECT AVG(Experience\_in\_hrs) FROM Pilot);

**Query137:Display passengers with an email domain of "example.com" along with the number of bookings each has made:**

SELECT P.Passenger\_ID, P.Full\_name, COUNT(B.Booking\_ID) AS BookingCount

FROM Passengers P

JOIN bookings B ON P.Passenger\_ID = B.Passenger\_ID

WHERE P.email LIKE '%@example.com'

GROUP BY P.Passenger\_ID, P.Full\_name;

**Query138:Select flights from the Schedules table that are priced above the overall average schedule price**

SELECT Flight\_ID, Price

FROM Schedules

WHERE Price > (SELECT AVG(Price) FROM Schedules);

**Query139:List employees who are not registered as pilots**

SELECT \* FROM Employees

WHERE Employee\_ID NOT IN (SELECT Employee\_ID FROM Pilot);

**Query140:List Passenger\_IDs that are present in both bookings and securitycheck**

SELECT Passenger\_ID FROM bookings

INTERSECT

SELECT Passenger\_ID FROM securitycheck;

**Query141:List all employees along with any pilot information (if available)**

SELECT E.Employee\_ID, E.Full\_name, P.Licensenumber

FROM Employees E

LEFT JOIN Pilot P ON E.Employee\_ID = P.Employee\_ID;

**Query142:Group weather records by the condition (e.g., 'Sunny', 'Cloudy') for cases when the temperature is above 20°C, and show only conditions with at least one record.**

SELECT Condition, COUNT(\*) AS WeatherCount

FROM Weather

WHERE Temperature > 20

GROUP BY Condition

HAVING COUNT(\*) >= 1;

**Query143:Count the number of open gates at each airport.**

SELECT Airport\_ID, COUNT(\*) AS OpenGateCount

FROM gate

WHERE gatestatus = 'Open'

GROUP BY Airport\_ID

HAVING COUNT(\*) >= 1;

**Query144:Count how many passengers belong to the USA, Canada, or the UK.**

SELECT Nationality, COUNT(\*) AS PassengerCount

FROM Passengers

WHERE Nationality IN ('USA', 'Canada', 'UK')

GROUP BY Nationality

HAVING COUNT(\*) >= 1;

**Query145:Sum the total payment amounts for Credit Card and Cash payments, and show only those methods with more than $500 in total.**

SELECT Payment\_method, SUM(Amount) AS TotalPayments

FROM Payments

WHERE Payment\_method IN ('Credit Card', 'Cash')

GROUP BY Payment\_method

HAVING SUM(Amount) > 500;

**Query146:Add a 'FrequentFlyerStatus' column to Passengers**

ALTER TABLE Passengers

ADD FrequentFlyerStatus VARCHAR(20) DEFAULT 'Standard';

**Query147:Change phone\_number to VARCHAR(20) to support international formats**

ALTER TABLE Passengers

ALTER COLUMN phone\_number VARCHAR(20);

**Query148:Airports with codes ending with 'X' (LAX, JFK)**

SELECT \* FROM Airport

WHERE Code LIKE '%X';

**Query149:Passengers with names starting with 'Jo' (John, Joan, etc.)**

SELECT \* FROM Passengers

WHERE Full\_name LIKE 'J%';

**Query150:Search for Gmail users**

SELECT \* FROM Passengers

WHERE email LIKE ['%@gmail.com';](mailto:'%25@gmail.com';)

**Query151:Select schedule records with prices between 200 and 250**

SELECT \* FROM Schedules

WHERE Price BETWEEN 200 AND 250;

**Query152 : Find weather records where the condition starts with 'S'**

SELECT \* FROM Weather

WHERE Condition LIKE 'S%';

**Query153 : Update the name of an airline**

UPDATE Airline

SET Name = 'SkyHigh Airways International'

WHERE Airline\_ID = 1;

**CHAPTER 4. CONCLUSION AND FUTUREWORK**

**4.1 Conclusion**

In conclusion, the DBMS-based Airline Reservation System provides a comprehensive and efficient solution to the challenges faced by traditional airline booking methods. By integrating a centralized database, the system ensures streamlined management of flight schedules, passenger information, and booking transactions, significantly reducing errors, data redundancy, and manual workload. The use of database management principles enhances data integrity, enables real-time updates, and improves overall system reliability and performance.

Through automation and secure transaction processing, the system offers a user-friendly interface for both customers and administrators, making the booking process faster, more accurate, and more secure. The incorporation of features such as seat availability checks, real-time flight information, and a secure payment gateway contributes to a smoother and more satisfactory user experience.

Moreover, the system is designed to be scalable and adaptable, laying the foundation for future enhancements such as mobile application integration, advanced analytics, and personalized services. Overall, this project not only improves operational efficiency for airlines but also enhances convenience and reliability for passengers, ultimately contributing to a more modern, digital, and customer-centric airline service environment.

**4.2 Future Work**

The future work for the DBMS-based Airline Reservation System project centers on enhancing functionality, scalability, and user experience through the integration of advanced technologies and expanded features. One significant area of development is the implementation of a mobile application, which would provide users with a seamless, on-the-go booking experience, integrating push notifications for flight updates and real-time alerts. Additionally, incorporating artificial intelligence (AI) and machine learning (ML) techniques could revolutionize personalized travel recommendations, dynamic pricing models, and fraud detection systems, thereby optimizing operational efficiency and customer satisfaction.

Another promising direction is the expansion of API integrations with external services such as weather forecasting, baggage tracking, and partner airlines, enabling a more comprehensive ecosystem that enhances the overall travel experience. Future enhancements may also include the adoption of cloud-based infrastructure to ensure higher availability, faster data processing, and scalable resource management, which is crucial for handling peak travel periods and expanding user bases. Advanced data analytics and reporting tools can be further developed to provide airlines with deeper insights into travel patterns, customer preferences, and operational bottlenecks, allowing for more informed decision-making and strategic planning.

Security measures will continue to evolve, with the integration of cutting-edge encryption protocols, multi-factor authentication, and continuous monitoring systems to protect sensitive customer data and maintain compliance with evolving regulatory standards. Furthermore, there is potential for integrating a loyalty program management module, which would incentivize repeat customers and foster long-term relationships with passengers. Overall, the future work will focus on transforming the system into a fully integrated, intelligent, and secure platform that not only meets the current needs of the airline industry but also anticipates and adapts to future challenges and innovations.