**SVKM’s NMIMS**

**Mukesh Patel School of Technology Management & Engineering, Mumbai**

A.Y. 2023 - 24

**Course: Database Management Systems**

**Project Report**

|  |  |  |
| --- | --- | --- |
| Program | MBA Tech Data Science | |
| Semester | IV | |
| Name of the Project: | Airline Reservation System | |
|  | | |
| Details of Project Members |  |  |
| Batch | Roll No. | Name |
| J2 | S040 | Priyanshu Padhi |
| J2 | S046 | Rishi Ponda |
| J2 | S024 | Khushi Parihar |
| J2 | S048 | Saloni Sharma |
| Date of Submission: 01/04/2024 | | |

**Contribution of each project Members:**

|  |  |  |
| --- | --- | --- |
| Roll No. | Name: | Contribution |
| S024 | Khushi Parihar | Normalisation |
| S040 | Priyanshu Padhi | Relational Model |
| S046 | Rishi Ponda | ER Diagram |
| S048 | Saloni Sharma | Theory |

**Github link of your project:**

[PriyanshuPadhi/padhi: Airline Reservation System (github.com)](https://github.com/PriyanshuPadhi/padhi)

**Note:**

1. Create a readme file if you have multiple files
2. All files must be properly named (Example:R004\_DBMSProject)
3. Submit all relevant files of your work ( Report, all SQL files, Any other files)
4. **Plagiarism is highly discouraged (Your report will be checked for plagiarism)**

**Rubrics for the Project evaluation:**

|  |  |
| --- | --- |
| First phase of evaluation:  Innovative Ideas (5 Marks)  Design and Partial implementation (5 Marks) | 10 marks |
| Final phase of evaluation  Implementation, presentation and viva, Self-Learning and Learning Beyond classroom | 10 marks |

**Project Report**

**Airline Reservation System**

**by**

**Khushi Parihar, Roll number: S024**

**Priyanshu Padhi, Roll number: S040**

**Rishi Ponda, Roll number: S046**

**Saloni Sharma, Roll number: S048**

**Course: DBMS**

**AY: 2023-24**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr no.** | **Topic** | **Page no.** |
| **1** | Storyline |  |
| **2** | Components of Database Design |  |
| **3** | Entity Relationship Diagram |  |
| **4** | Relational Model |  |
| **5** | Normalization |  |
| **6** | SQL Queries |  |
| **7** | Learning from the Project |  |
| **8** | Project Demonstration |  |
| **9** | Self-learning beyond classroom |  |
| **10** | Learning from the project |  |
| **11** | Challenges faced |  |
| **12** | Conclusion |  |

**I. Storyline**

Imagine a web-based platform for booking and searching flights by travelers. In order to maintain their profiles and booking history, users can register and establish accounts. Airlines, with their own names, identities, and maybe contact details, run the flights. Particulars about a flight include the airports of departure and arrival (identified by unique numbers), the times of departure and arrival, and the seats that are available. Flights can be found by users using a variety of search parameters, including origin, destination, date, and passenger count. After choosing a preferred flight, users have the option to reserve seats and enter passenger details. The system records passenger information, handles reservations, and streamlines payment processing.

**II. Components of Database Design**

Entities and their attributes:

Users:

User\_id (Primary Key)

User\_name

User\_mobile

User\_email

User\_address

Roles:

Role\_id (Primary Key)

Role\_name

Role\_desc

Employee:

Employee\_id (Primary Key)

Employee\_name

Employee\_age

Employee\_address

Employee\_User:

Employee\_id (Primary Key, Foreign Key referencing Employee(Employee\_id))

User\_id (Foreign Key referencing Users(User\_id))

Permission:

Per\_id (Primary Key)

Per\_name

Per\_address

Roles\_Permission:

Role\_id (Foreign Key referencing Roles(Role\_id))

Per\_id (Foreign Key referencing Permission(Per\_id))

(Composite Primary Key: Role\_id, Per\_id)

Ticket\_booking:

Ticket\_id (Primary Key)

Ticket\_date

Ticket\_describe

Employee\_id (Foreign Key referencing Employee(Employee\_id))

Airlines\_Booking:

AB\_id (Primary Key)

AB\_date

AB\_describe

Employee\_id (Foreign Key referencing Employee(Employee\_id))

Airline\_Enquiry:

AE\_id (Primary Key)

AE\_title

AE\_date

AE\_type

Relationships:

Users - Employee:

One-to-One

Participation: Mandatory on both sides (each user must be an employee and each employee must have a user account)

Employee - Employee\_User:

One-to-Many

Participation: Mandatory on the Employee side (each employee must have at least one user account)

Roles - Roles\_Permission:

Many-to-Many

Participation: Mandatory on both sides (each role must have at least one permission and each permission must be associated with at least one role)

Employee - Ticket\_booking:

One-to-Many

Participation: Optional on the Ticket\_booking side (an employee may or may not make a ticket booking)

Employee - Airlines\_Booking:

One-to-Many

Participation: Optional on the Airlines\_Booking side (an employee may or may not make an airline booking)

Employee - Airline\_Enquiry:

One-to-Many

Participation: Optional on the Airline\_Enquiry side (an employee may or may not make an airline enquiry)

Cardinality:

Users - Employee: 1:1

Employee - Employee\_User: 1:M

Roles - Roles\_Permission: M:M

Employee - Ticket\_booking: 1:M

Employee - Airlines\_Booking: 1:M

Employee - Airline\_Enquiry: 1:M

Participation:

Users - Employee: Mandatory on both sides

Employee - Employee\_User: Mandatory on Employee side

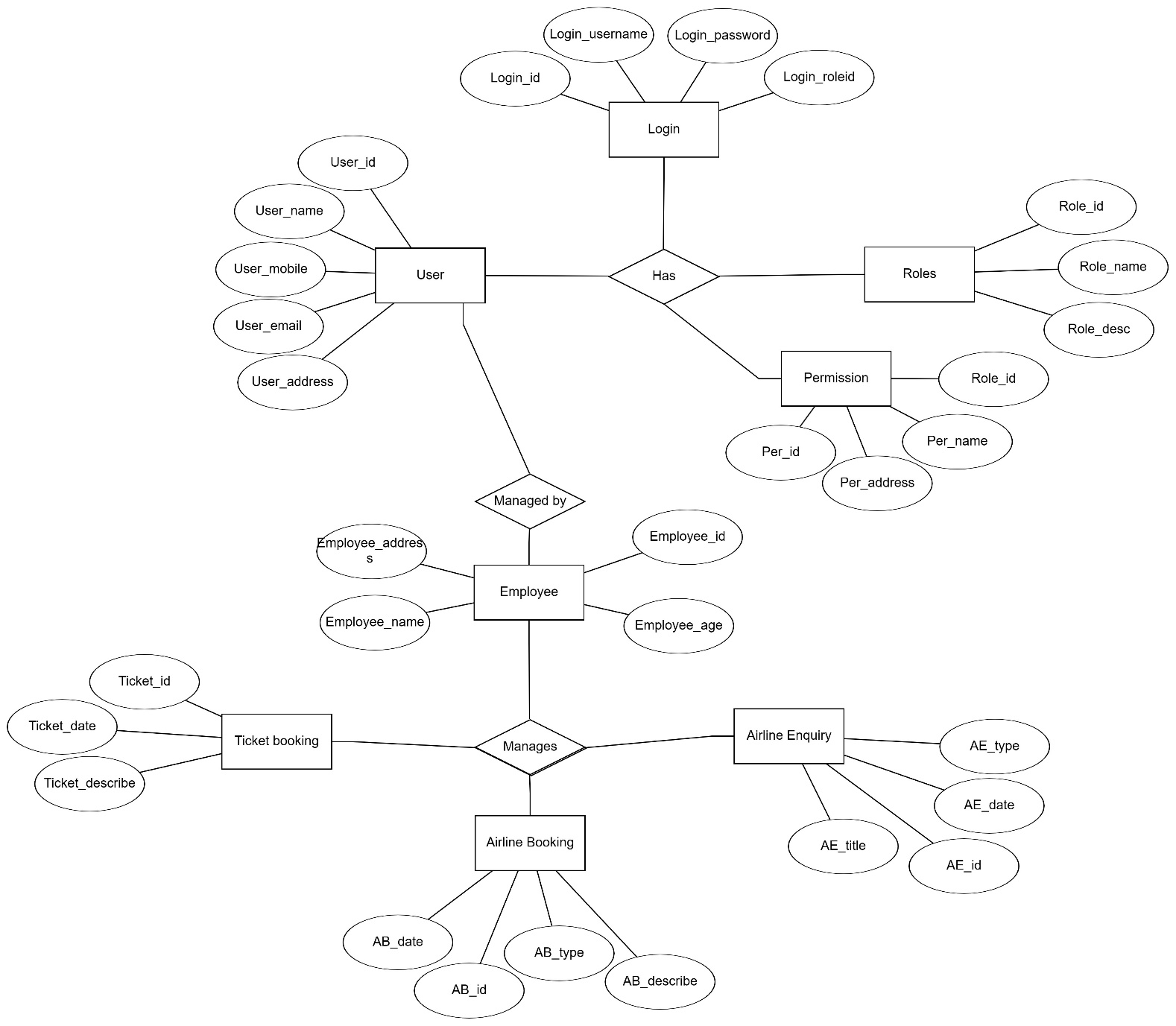
Roles - Roles\_Permission: Mandatory on both sides

Employee - Ticket\_booking: Optional on Ticket\_booking side

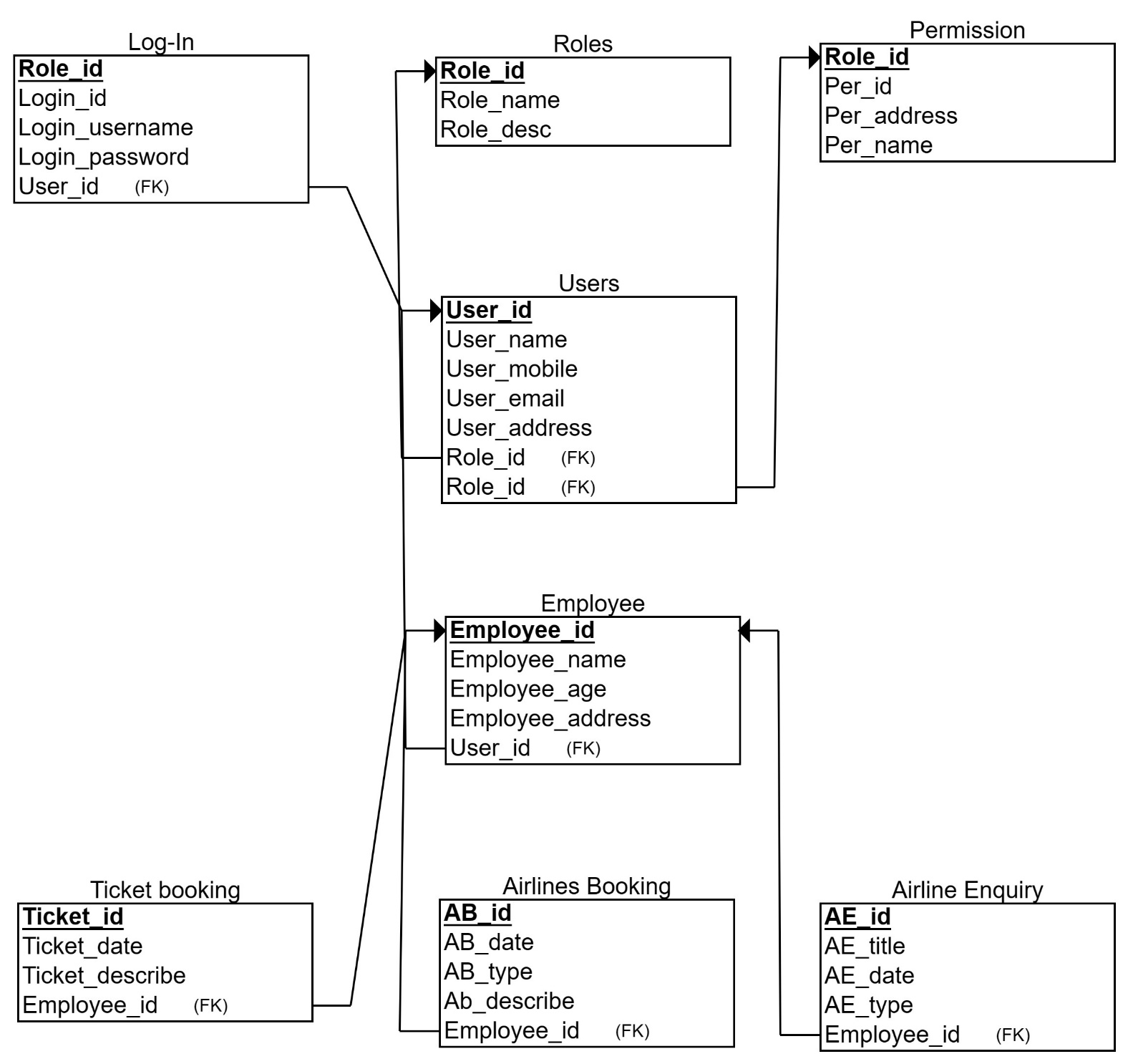
Employee - Airlines\_Booking: Optional on Airlines\_Booking side

Employee - Airline\_Enquiry: Optional on Airline\_Enquiry side

**III. Entity Relationship Diagram**

****

**IV. Relational Model**



**V. Normalization**

**1NF (First Normal Form):**

To achieve 1NF, ensure that each attribute in every table contains only atomic (indivisible) values.

Given the provided schema, the tables already appear to be in 1NF as there are no multivalued attributes.

**2NF (Second Normal Form):**

To achieve 2NF, the table must be in 1NF and all non-key attributes must be fully functionally dependent on the primary key.

Looking at the schema:

Users table:

User\_mobile, User\_email, and User\_address are fully functionally dependent on User\_id.

Roles table:

Role\_desc is fully functionally dependent on Role\_id.

Login table:

Login\_username and Login\_password are fully functionally dependent on Login\_id.

Employee table:

Employee\_name, Employee\_age, and Employee\_address are fully functionally dependent on Employee\_id.

Hence, the tables are already in 2NF.

**3NF (Third Normal Form):**

To achieve 3NF, the table must be in 2NF, and there should be no transitive dependencies.

Users table:

There are no transitive dependencies.

Roles table:

There are no transitive dependencies.

Login table:

There are no transitive dependencies.

Employee table:

There are no transitive dependencies.

Hence, the tables are already in 3NF.

**4NF (Fourth Normal Form):**

To achieve 4NF, the table must be in BCNF (Boyce-Codd Normal Form), and there should be no multi-valued dependencies.

There are no multi-valued dependencies in the provided schema.

**5NF (Fifth Normal Form):**

To achieve 5NF, the table must be in 4NF, and it should not contain any join dependencies.

There are no join dependencies in the provided schema.

Therefore, the provided schema is already in 1NF, 2NF, 3NF, 4NF, and 5NF. No further normalization is required.

**VI. SQL Queries**

SQL for ER diagram and Tables

create database airlines;   
show databases;   
use airlines;  
CREATE TABLE Users (  
  User\_id INT PRIMARY KEY,  
  User\_name VARCHAR(255) NOT NULL,  
  User\_mobile VARCHAR(255) NOT NULL,  
  User\_email VARCHAR(255) NOT NULL,  
  User\_address VARCHAR(255) NOT NULL  
);  
  
CREATE TABLE Roles (  
  Role\_id INT PRIMARY KEY,  
  Role\_name VARCHAR(255) NOT NULL,  
  Role\_desc VARCHAR(255) NOT NULL  
);  
  
CREATE TABLE Login (  
  Login\_id INT PRIMARY KEY,  
  Login\_username VARCHAR(255) NOT NULL,  
  Login\_password VARCHAR(255) NOT NULL,  
  User\_id INT,  
  FOREIGN KEY (User\_id) REFERENCES Users(User\_id)  
);  
  
CREATE TABLE Employee (  
  Employee\_id INT PRIMARY KEY,  
  Employee\_name VARCHAR(255) NOT NULL,  
  Employee\_age INT NOT NULL,  
  Employee\_address VARCHAR(255) NOT NULL,  
  User\_id INT,  
  FOREIGN KEY (User\_id) REFERENCES Users(User\_id)  
);  
  
CREATE TABLE Permission (  
  Per\_id INT PRIMARY KEY,  
  Per\_name VARCHAR(255) NOT NULL,  
  Per\_address VARCHAR(255) NOT NULL  
);  
  
CREATE TABLE Roles\_Permission (  
  Role\_id INT,  
  Per\_id INT,  
  PRIMARY KEY (Role\_id, Per\_id),  
  FOREIGN KEY (Role\_id) REFERENCES Roles(Role\_id),  
  FOREIGN KEY (Per\_id) REFERENCES Permission(Per\_id)  
);  
  
CREATE TABLE Ticket\_booking (  
  Ticket\_id INT PRIMARY KEY,  
  Ticket\_date DATE NOT NULL,  
  Ticket\_describe VARCHAR(255) NOT NULL,  
  Employee\_id INT,  
  FOREIGN KEY (Employee\_id) REFERENCES Employee(Employee\_id)  
);  
  
CREATE TABLE Airlines\_Booking (  
  AB\_id INT PRIMARY KEY,  
  AB\_date DATE NOT NULL,  
  AB\_describe VARCHAR(255) NOT NULL,  
  Employee\_id INT,  
  FOREIGN KEY (Employee\_id) REFERENCES Employee(Employee\_id)  
);  
  
CREATE TABLE Airline\_Enquiry (  
  AE\_id INT PRIMARY KEY,  
  AE\_title VARCHAR(255) NOT NULL,  
  AE\_date DATE NOT NULL,  
  AE\_type VARCHAR(255) NOT NULL  
);

**Sample Data**

INSERT INTO Users (User\_id, User\_name, User\_mobile, User\_email, User\_address) VALUES  
(1, 'John Doe', '1234567890', 'john@example.com', '123 Main Street'),  
(2, 'Jane Smith', '9876543210', 'jane@example.com', '456 Elm Street'),  
(3, 'Alice Johnson', '5551234567', 'alice@example.com', '789 Oak Street'),  
(4, 'Bob Brown', '5559876543', 'bob@example.com', '101 Pine Street'),  
(5, 'Eve Davis', '5557891234', 'eve@example.com', '202 Maple Street'),  
(6, 'Michael Clark', '5554567890', 'michael@example.com', '303 Cedar Street'),  
(7, 'Sarah Lee', '5553219876', 'sarah@example.com', '404 Walnut Street'),  
(8, 'David Wilson', '5556782345', 'david@example.com', '505 Pineapple Street'),  
(9, 'Emily Taylor', '5558901234', 'emily@example.com', '606 Orange Street'),  
(10, 'James Anderson', '5552345678', 'james@example.com', '707 Peach Street');  
  
INSERT INTO Roles (Role\_id, Role\_name, Role\_desc) VALUES  
(1, 'Admin', 'System Administrator'),  
(2, 'User', 'Regular User'),  
(3, 'Manager', 'Team Manager');  
  
INSERT INTO Login (Login\_id, Login\_username, Login\_password, User\_id) VALUES  
(1, 'johndoe', 'password123', 1),  
(2, 'janesmith', 'letmein', 2),  
(3, 'alice', 'alicepassword', 3),  
(4, 'bob', 'bobpassword', 4),  
(5, 'eve', 'evepassword', 5),  
(6, 'michael', 'michaelpassword', 6),  
(7, 'sarah', 'sarahpassword', 7),  
(8, 'david', 'davidpassword', 8),  
(9, 'emily', 'emilypassword', 9),  
(10, 'james', 'jamespassword', 10);  
  
INSERT INTO Employee (Employee\_id, Employee\_name, Employee\_age, Employee\_address, User\_id) VALUES  
(1, 'Mike Johnson', 30, '789 Oak Street', 3),  
(2, 'Emily Brown', 25, '101 Pine Street', 4),  
(3, 'Adam Davis', 35, '202 Maple Street', 5),  
(4, 'Laura Clark', 40, '303 Cedar Street', 6),  
(5, 'Steven Lee', 45, '404 Walnut Street', 7),  
(6, 'Rachel Wilson', 50, '505 Pineapple Street', 8),  
(7, 'Daniel Taylor', 55, '606 Orange Street', 9),  
(8, 'Olivia Anderson', 60, '707 Peach Street', 10),  
(9, 'Sophia Martinez', 28, '808 Lemon Street', 3),  
(10, 'William Thomas', 32, '909 Grape Street', 4);  
  
INSERT INTO Permission (Per\_id, Per\_name, Per\_address) VALUES  
(1, 'Read', '/read'),  
(2, 'Write', '/write'),  
(3, 'Execute', '/execute');  
  
INSERT INTO Roles\_Permission (Role\_id, Per\_id) VALUES  
(1, 1),  
(2, 2),  
(3, 3),  
(1, 2),  
(2, 1),  
(3, 2),  
(1, 3),  
(2, 3),  
(3, 1),  
(3, 2);  
  
INSERT INTO Ticket\_booking (Ticket\_id, Ticket\_date, Ticket\_describe, Employee\_id) VALUES  
(1, '2024-03-30', 'Flight to New York', 1),  
(2, '2024-04-01', 'Train to Chicago', 2),  
(3, '2024-04-03', 'Bus to Los Angeles', 3),  
(4, '2024-04-05', 'Flight to London', 4),  
(5, '2024-04-07', 'Train to Paris', 5),  
(6, '2024-04-09', 'Bus to Rome', 6),  
(7, '2024-04-11', 'Flight to Tokyo', 7),  
(8, '2024-04-13', 'Train to Sydney', 8),  
(9, '2024-04-15', 'Bus to Beijing', 9),  
(10, '2024-04-17', 'Flight to Moscow', 10);  
  
INSERT INTO Airlines\_Booking (AB\_id, AB\_date, AB\_describe, Employee\_id) VALUES  
(1, '2024-04-20', 'Flight to New York', 1),  
(2, '2024-04-22', 'Flight to London', 2),  
(3, '2024-04-24', 'Flight to Tokyo', 3),  
(4, '2024-04-26', 'Flight to Paris', 4),  
(5, '2024-04-28', 'Flight to Sydney', 5),  
(6, '2024-04-30', 'Flight to Beijing', 6),  
(7, '2024-05-02', 'Flight to Rome', 7),  
(8, '2024-05-04', 'Flight to Moscow', 8),  
(9, '2024-05-06', 'Flight to New Delhi', 9),  
(10, '2024-05-08', 'Flight to Dubai', 10);  
  
INSERT INTO Airline\_Enquiry (AE\_id, AE\_title, AE\_date, AE\_type) VALUES  
(1, 'Flight Inquiry', '2024-05-10', 'General'),  
(2, 'Booking Inquiry', '2024-05-12', 'Specific'),  
(3, 'Flight Inquiry', '2024-05-14', 'General'),  
(4, 'Booking Inquiry', '2024-05-16', 'Specific'),  
(5, 'Flight Inquiry', '2024-05-18', 'General'),  
(6, 'Booking Inquiry', '2024-05-20', 'Specific'),  
(7, 'Flight Inquiry', '2024-05-22', 'General'),  
(8, 'Booking Inquiry', '2024-05-24', 'Specific'),  
(9, 'Flight Inquiry', '2024-05-26', 'General'),  
(10, 'Booking Inquiry', '2024-05-28', 'Specific');  
select \* from Users;  
select \* from Roles;  
select \* from Login;  
select \* from Employee;  
select \* from Permission;  
select \* from Roles\_Permission;  
select \* from Ticket\_booking;  
select \* from Airlines\_booking;  
select \* from Airline\_Enquiry;

SELECT \*

FROM Users

INNER JOIN Employee ON Users.User\_id = Employee.User\_id;

SELECT \*

FROM Users

LEFT OUTER JOIN Employee ON Users.User\_id = Employee.User\_id;

SELECT \*

FROM Users

RIGHT OUTER JOIN Employee ON Users.User\_id = Employee.User\_id;

SELECT \*

FROM Users

FULL OUTER JOIN Employee ON Users.User\_id = Employee.User\_id;

1. Retrieve all users who have both a mobile number and an email address.
2. Find the roles of employees who are aged 30 or older.
3. Display the permissions associated with the role named "Admin".
4. List the employees who have made airline bookings.
5. Find the ticket bookings made by employees whose age is less than 30.
6. Retrieve the titles of airline enquiries made on or after 2024-05-10.
7. Display the routes with fares exceeding $100.
8. List the destinations of ticket bookings made on routes with fares less than $50.
9. Find the employee names and their corresponding roles.
10. Display the ages of users who have both a mobile number and an email address.
11. Retrieve the descriptions of roles that have permissions associated with them.
12. List the roles of employees who have made ticket bookings.
13. Find the users who have made both airline bookings and ticket bookings.
14. Display the names of employees who are not associated with any ticket bookings.
15. List the employees who have made ticket bookings and their associated roles.
16. Retrieve the origins and destinations of routes with distances exceeding 500 miles.
17. Display the roles of employees who have made airline enquiries.
18. Find the email addresses of users associated with employees aged 40 or older.
19. What SQL query can be used to retrieve the User\_id and User\_name of users who have either booked a ticket or made an airline booking?
20. Generate a GROUP BY query to find the count of tickets booked by each user.

-- 1. Retrieve all users who have both a mobile number and an email address.

SELECT \*

FROM Users

WHERE User\_mobile IS NOT NULL AND User\_email IS NOT NULL;

-- 2. Find the roles of employees who are aged 30 or older.

SELECT e.Employee\_name, r.Role\_name

FROM Employee e

JOIN Employee\_User eu ON e.Employee\_id = eu.Employee\_id

JOIN Roles r ON eu.Role\_id = r.Role\_id

WHERE e.Employee\_age >= 30;

-- 3. Display the permissions associated with the role named "Admin".

SELECT rp.Role\_id, p.\*

FROM Roles\_Permission rp

JOIN Roles r ON rp.Role\_id = r.Role\_id

JOIN Permission p ON rp.Per\_id = p.Per\_id

WHERE r.Role\_name = 'Admin';

-- 4. List the employees who have made airline bookings.

SELECT e.Employee\_name

FROM Employee e

JOIN Airlines\_Booking ab ON e.Employee\_id = ab.Employee\_id;

-- 5. Find the ticket bookings made by employees whose age is less than 30.

SELECT tb.\*

FROM Ticket\_booking tb

JOIN Employee e ON tb.Employee\_id = e.Employee\_id

WHERE e.Employee\_age < 30;

-- 6. Retrieve the titles of airline enquiries made on or after 2024-05-10.

SELECT AE\_title

FROM Airline\_Enquiry

WHERE AE\_date >= '2024-05-10';

-- 7. Display the routes with fares exceeding $100.

SELECT \*

FROM route\_Header

WHERE Fare > 100;

-- 8. List the destinations of ticket bookings made on routes with fares less than $50.

SELECT rh.Destination

FROM Ticket\_booking tb

JOIN route\_Header rh ON tb.route\_id = rh.route\_id

WHERE rh.Fare < 50;

-- 9. Find the employee names and their corresponding roles.

SELECT e.Employee\_name, r.Role\_name

FROM Employee e

JOIN Employee\_User eu ON e.Employee\_id = eu.Employee\_id

JOIN Roles r ON eu.Role\_id = r.Role\_id;

-- 10. Display the ages of users who have both a mobile number and an email address.

SELECT User\_id, User\_name, User\_mobile, User\_email, User\_address,

CASE WHEN User\_age IS NOT NULL THEN User\_age ELSE 'Unknown'

FROM Users

WHERE User\_mobile IS NOT NULL AND User\_email IS NOT NULL;

-- 11. Retrieve the descriptions of roles that have permissions associated with them.

SELECT r.Role\_name, r.Role\_desc

FROM Roles r

JOIN Roles\_Permission rp ON r.Role\_id = rp.Role\_id;

-- 12. List the roles of employees who have made ticket bookings.

SELECT e.Employee\_name, r.Role\_name

FROM Employee e

JOIN Ticket\_booking tb ON e.Employee\_id = tb.Employee\_id

JOIN Roles r ON e.Role\_id = r.Role\_id;

-- 13. Find the users who have made both airline bookings and ticket bookings.

SELECT u.User\_id, u.User\_name

FROM Users u

JOIN Ticket\_booking tb ON u.User\_id = tb.User\_id

JOIN Airlines\_Booking ab ON u.User\_id = ab.User\_id;

-- 14. Display the names of employees who are not associated with any ticket bookings.

SELECT e.Employee\_name

FROM Employee e

WHERE e.Employee\_id NOT IN (SELECT Employee\_id FROM Ticket\_booking);

-- 15. List the employees who have made ticket bookings and their associated roles.

SELECT e.Employee\_name, r.Role\_name

FROM Employee e

JOIN Ticket\_booking tb ON e.Employee\_id = tb.Employee\_id

JOIN Roles r ON e.Role\_id = r.Role\_id;

-- 16. Retrieve the origins and destinations of routes with distances exceeding 500 miles.

SELECT origin, destination

FROM route\_Header

WHERE distance > 500;

-- 17. Display the roles of employees who have made airline enquiries.

SELECT e.Employee\_name, r.Role\_name

FROM Employee e

JOIN Airline\_Enquiry ae ON e.Employee\_id = ae.Employee\_id

JOIN Roles r ON e.Role\_id = r.Role\_id;

-- 18. Find the email addresses of users associated with employees aged 40 or older.

SELECT u.User\_email

FROM Users u

JOIN Employee e ON u.User\_id = e.User\_id

WHERE e.Employee\_age >= 40;

19

SELECT User\_id, User\_name

FROM Users

WHERE User\_id IN (

SELECT DISTINCT User\_id

FROM (

SELECT User\_id FROM Ticket\_booking

UNION

SELECT User\_id FROM Airlines\_Booking

) AS Bookings

);

20

SELECT Users.User\_id, Users.User\_name, COUNT(\*) AS Ticket\_count

FROM Users

JOIN Ticket\_booking ON Users.User\_id = Ticket\_booking.Employee\_id

GROUP BY Users.User\_id, Users.User\_name;

**VI. Project demonstration**

* ERDPlus software used for constructing the ER diagram and the relational database model.
* [ER Diagram for the Airline Reservation System - javatpoint](https://www.javatpoint.com/er-diagram-for-the-airline-reservation-system) - reference website used as an example for our project.
* MySQL Workbench 8.0 CE used for the query codes of the ER diagram and the normalization.
* ChatGPT and Gemini used for the sample data.

**VII. Self -Learning beyond classroom**

:

Advanced Data Modeling Techniques: In situations when typical relational models might not be the best option, consider modeling approaches like entity-attribute-value (EAV) modeling or complex object modeling.   
Data Warehousing and Business Intelligence: Learn about the principles and methods of data warehousing for the purpose of keeping and evaluating past airline reservation data in order to obtain knowledge about flight performance, consumer behavior, and other related topics.   
SQL Databases: Examine and contrast SQL and relational databases, taking into account which is better suited for particular use cases in airline reservation systems (managing large amounts of real-time booking data, for example).   
Airline sector legislation: Learn about the pertinent legislation governing the airline sector that may affect the way your reservation system collects, stores, and handles data security.

**VIII. Learning from the Project**

Understanding Database Design Principles: This assignment helped you gain a firm grasp of relational tables, entity-relationship modeling, and normalization as database design principles.

Data Modeling for Complex Systems: You gained experience in data modeling for complicated systems, such as airline reservation systems, which have numerous entities and interactions.

Transforming Requirements into a Database Design: Translating actual airline reservation requirements into a tangible database structure was a step in the database design process.

Problem-Solving Ability: You showed problem-solving ability when you designed the database by identifying entities, relationships, and possible problems.

**IX. Challenges Faced**

Data Security Considerations: When handling passenger information and payment details, data security and user convenience must be balanced.   
Scalability and Performance: Organizing the database design to handle future increases in the number of flight options, reservations, and users.   
Integrating with External Systems: Taking into account any potential requirements for the reservation system to communicate with other airline systems or outside providers.

**X. Conclusion**

Important Takeaways (Continued): and using analytical abilities to build a workable database for an airline reservation system.

Prospective Uses: The skills acquired from this project can be used to create databases for different reservation systems (like those used by hotels or rental cars) or other applications where managing reservations and transactions is necessary.