Airways Management System

Team-06

Samip Shah

1001709873

Anushka Nagpure

1001764114

Samip Shah (1001709873)
Anushka Nagpure (1001764114)

Airways Management System

Abstract

Airways Management Systems is a computerized system used to store and retrieve information and conduct transactions related to air travel. This project is aimed at exposing the importance of Airways Management Systems.

The purpose of this application is to reduce the manual work and get all the details under one roof. It's simple GUI makes it available for all the users and target audience can be increased. The combination of passengers-flight-airport details makes it beneficial for everyone and not for certain audience.

Description

Airways Management System incorporate airline schedules, fare tariffs, passenger personal details & reservations, ticket records, employee details and airport information. This project is a package using which you can confirm passenger reservations, have a look at list of passengers and on list of flights with their schedule, airport information along with details of flight employees. Its user-friendly graphical interface helps users to interact with the system and meet their requirements in just few clicks.

The login page would be having option to register the user for first time and then from next time credentials will be matched and according dashboard will be created. The passenger will be able to view all the flights with details. Employee login will also be similar with list of flights and locations. Both passengers and employees will be able to view airport details and location. Both Passenger and employee may be able to filter and have a look of flights for day or for flight type. They may also be able to filter or sort following data based on price or type of ticket they want.

Data to be captured and displayed

- Passenger's personal details (Name, Number, Address, SSN etc...);
- Passengers flight details (From where and to, dates of travel, price at what ticket is booked, class of seat booked etc…);
- Flight details (Flight number, from where and when it travels, capacity of flight, no of flight attendees etc…);
- Flight attendee's personal details (Name, Number, Address, SSN etc...);
- Airport Details (Airport ID, Location, # of Runways, # of employee)

Queries (This are the output and operations user will be able to perform on dashboard after successful login. They can be modified later)

1. Display Complete Tables.
2. Display Ticket details with Payment details.
3. Display Airport details, Flight details and Schedule.
4. Display Employee details with Airport and flight schedules.
5. Display Passengers travel details.
6. Get total amount received from specific Passenger.
7. Get total Capacity of flight according to its type.
8. GUI to insert Data in table.
9. Delete Record From Table.
10. Filter Flight according to its type.

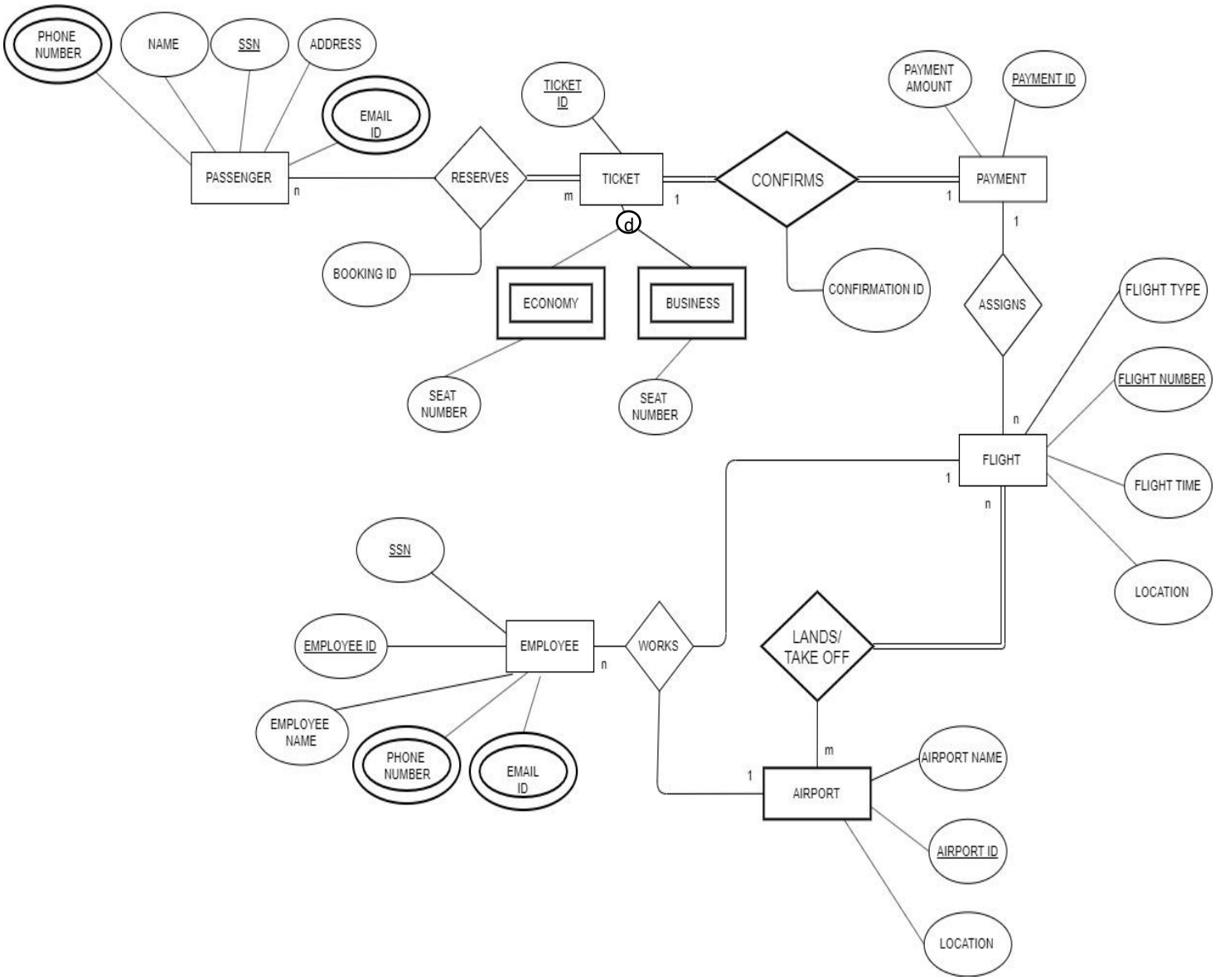
Assumed entities with some of the attributes and relations (It may be modified in next phase according to requirements)

- Passenger personal (Name, *SSN*, Address, Phone number)
- Passenger booking (*Booking ID*, SSN, Flight number, Class, Price, Time)
- Flight attendee (*Employee ID*, Name, SSN, Phone number, Position, Flight number)

- Flight (*Flight number*, Flight time, Flight location)
- Airport (*Airport ID*, Location, # of runways)

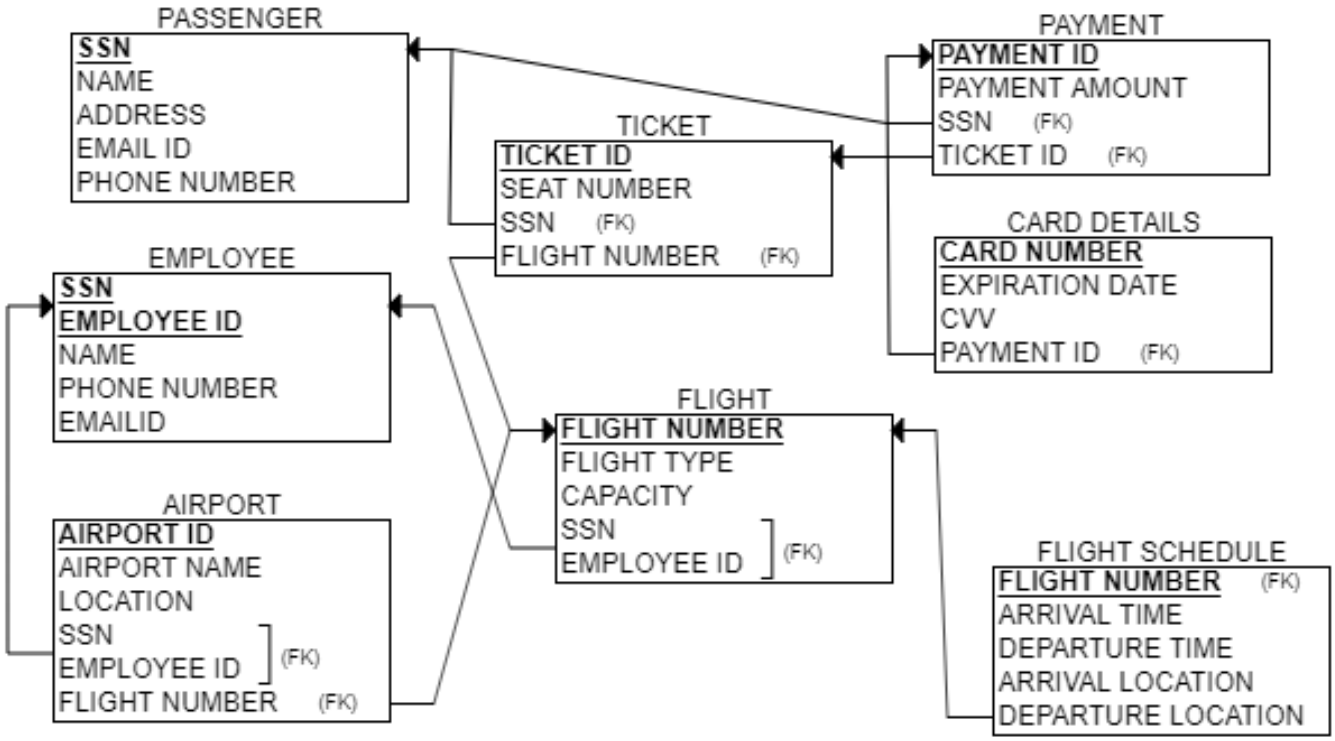
Each SSN may be associated with specific booking ID and each booking ID may relate to specific flight number. There will be many to many relations between airport ID and Flight Number. Employee ID and flight number would have one to one relation. Employee ID may be associated with flight number.

EER DIAGRAM



Samip Shah (1001709873) Team 06
Anushka Nagpure (1001764114)

RELATIONAL MODEL



RELATIONAL ALGEBRA

1. Get Passengers Details;

σ(PASSENGERS)

2. Get flight details of any particular passenger. (Assume get flight details of passenger with SSN: 123456789.)

Π FLIGHT_ID, FLIGHT_TIME, FLIGHT_TYPE, LOCATION (σ *ssn="123456789"*(FLIGHT))

3. Get Complete Flight Details; (All the flights)

σ(FLIGHTS)

4. Sort Flight according to type (Get list of all Domestic Flights)

Π FLIGHT_TYPE (σ*FLIGHT_TYPE="Domestic"*(FLIGHT))

5. Filter flight according to its timings.

Π FLIGHT_TIME (σ*FLIGHT_TIME="Morning"*(FLIGHT))

6. Get flight attendant details

σ(EMPLOYEE)

7. Filtering flight attendants according to their respective flights.
(i. Suppose we find all the employees for flight number 123)
(ii. Suppose we find all the employees for flight time: Morning)

GET_TEMP ← Π SSN (σ (FLIGHT NUMBER = "123")(FLIGHT))
GET_ANSWER ← Π (EMPLOYEE ID, EMPLOYEE NAME, EMAIL ID)(σ (SSN = GET-TEMP(EMPLOYEE))
GET_TEMP ← Π SSN (σ (FLIGHT TIME = "MORNING")FLIGHT)
GET_ANSWER ← Π (EMPLOYEE ID, EMPLOYEE NAME, EMAIL ID)(σ (SSN = GET-TEMP(EMPLOYEE))

8. List of airports with its details
σ(AIRPORT)

9. Filtering/ Sorting flight according to price range (Suppose we search for all the flight whose price is 1000)

GET_TEMP ← Π FLIGHT NUMBER (σ (PAYMENT AMOUNT = "10000")(PAYMENT))
GET_ANSWER ← Π (FLIGHT NUMBER, FLIGHT TYPE, LOCATION)(σ (FLIGHT NUMBER = GET-TEMP(FLIGHT))

10. Filtering/ Sorting passengers according to their tickets (Economy/Business)

GET_TEMP ← Π SSN (σ (FLIGHT TYPE = "ECONOMY")(FLIGHT))
GET_ANSWER ← Π (NAME,PHONE NUMBER,EMAIL ID,ADDRESS)(σ (SSN = GET-TEMP(PASSENGER))

Conclusion

From this project I would be able to learn about entities and what attributes would be possible for each of them. How there would be relation between each of them and for what attribute what could be the primary key and how they can be related to each other.

EER diagram gives the flow of the project and gives details about which entity has what attributes. It also gives details about which entity has what primary key. Relational Model (Relational Schema) gives details about what key is connected where and how the data will be retrieved when search query will be fired. Phase 2 also has relational queries which gives details about how SQL queries will work and what details from what table will be retrieved.

For better output of the project, more entities need to be added and more complex relations need to be formed. GUI can also be made more attractive and user friendly with more complex queries.