1. INTRODUCTION

1.1 PURPOSE

The purpose of this document is to build an online system to manage flights and passengers to ease the flight management. *<<Include the purpose as applicable to your project >>*

1.2 DOCUMENT CONVENTIONS

This document uses the following conventions. *<<Include the conventions as per your application >>*

DB	Database
DDB	Distributed Database
ER	Entity Relationship

1.3 INTENDED AUDIENCE AND READING SUGGESTIONS

This project is a prototype for the flight management system and it is restricted within the college premises. This has been implemented under the guidance of college professors. This project is useful for the flight management team and as well as to the passengers.

1.4 PROJECT SCOPE

The purpose of the online flight management system is to ease flight management and to create a convenient and easy-to-use application for passengers, trying to buy airline tickets. The system is based on a relational database with its flight management and reservation functions. We will have a database server supporting hundreds of major cities around the world as well as thousands of flights by various airline companies. Above all, we hope to provide a comfortable user experience along with the best pricing available.

1.5 REFERENCES

- https://krazytech.com/projects
- Fundamentals of database systems by ramez elmarsi and shamkant b.navathe

2. OVERALL DESCRIPTION

2.1 PRODUCT PERSPECTIVE

A distributed airline database system stores the following information.

- **Flight details:**
  It includes the originating flight terminal and destination terminal, along with the stops in between, the number of seats booked/available seats between two destinations etc.

- **Customer description:**
  It includes customer code, name, address, and phone number. This information may be used for keeping the records of the customer for any emergency or for any other kind of information.

- **Reservation description:**
  It includes customer details, code number, flight number, date of booking, date of travel.

2.2 PRODUCT FEATURES

The major features of the airline database system as shown in below **entity–relationship model** (**ER model**)
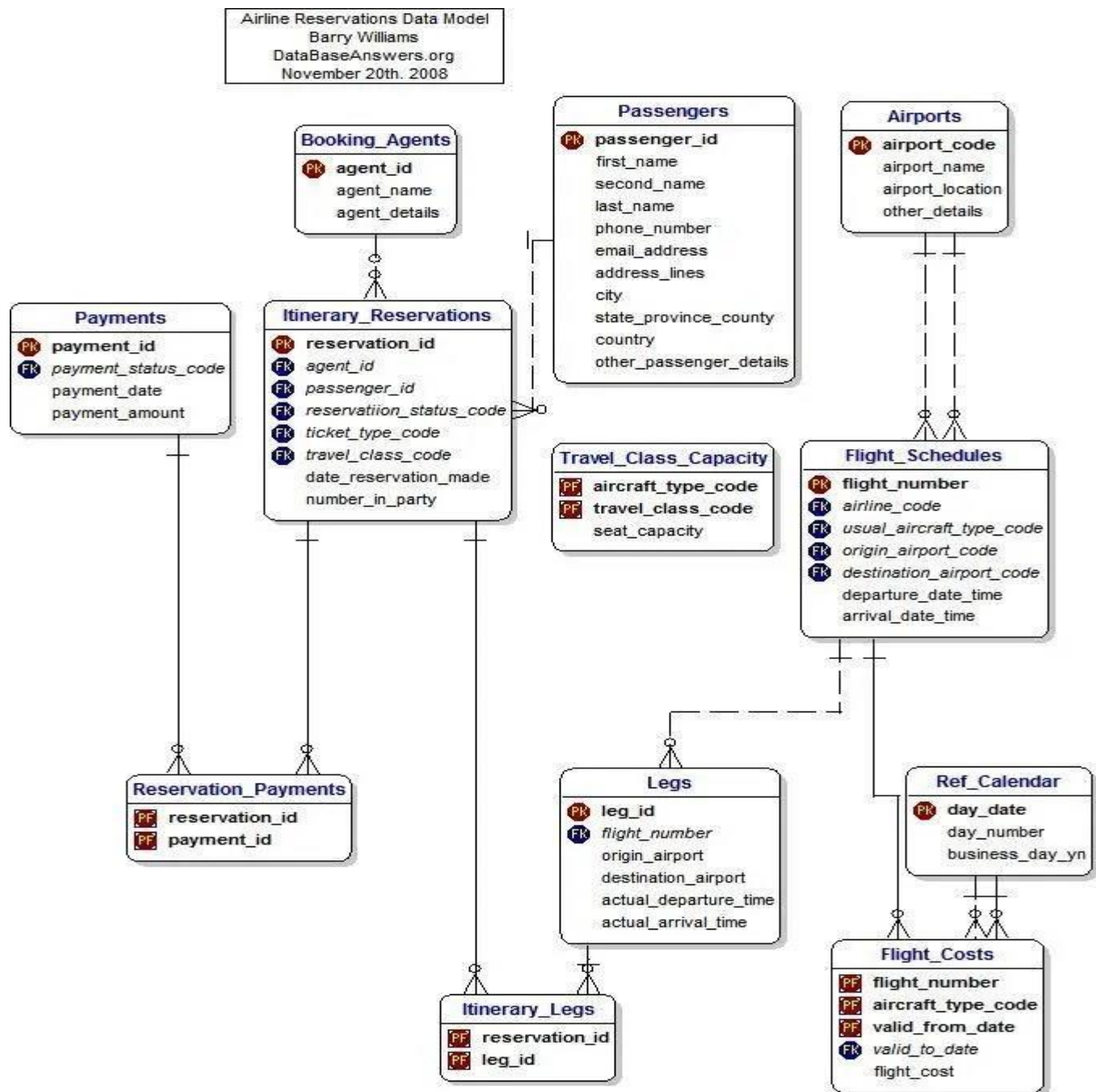
The diagram shows the layout of airline database system – entity–relationship model

2.3 USER CLASS and CHARACTERISTICS

Users of the system should be able to retrieve flight information between two given cities with the given date/time of travel from the database. A route from city A to city B is a sequence of connecting flights from A to B such that: a) there are at most two connecting stops, excluding the starting city and destination city of the trip, b) the connecting time is between one to two hours. The system will support two types of user

privileges, Customer, and Employee. Customers will have access to customer functions, and the employees will have access to both customer and flight management functions. The customer should be able to do the following functions:

- Make a new reservation
  - One-way
  - Round-Trip
  - Multi-city
  - Flexible Date/time
  - Confirmation

- Cancel an existing reservation

- View his itinerary
  The Employee should have the following management functionalities:

- CUSTOMER FUNCTIONS.
  - Get all customers who have seats reserved on a given flight.
  - Get all flights for a given airport.
  - View flight schedule.
  - Get all flights whose arrival and departure times are on time/delayed.
  - Calculate total sales for a given flight.

- ADMINISTRATIVE
  - Add/Delete a flight
  - Add a new airport
  - Update fare for flights.
  - Add a new flight leg instance.
  - Update departure/arrival times for flight leg instances.
  Each flight has a limited number of available seats. There are a number of flights that depart from or arrive at different cities on different dates and times.

2.4 OPERATING ENVIRONMENT

The operating environment for the airline management system is as listed below. *<<Include the details as per your application >>*

- distributed database

- client/server system

- Operating system: Windows.

- database: sql+ database

- platform: vb.net/Java/PHP
  ## 2.5 DESIGN and IMPLEMENTATION CONSTRAINTS

1. The global schema, fragmentation schema, and allocation schema.

2. SQL commands for the above queries/applications

3. How the response for applications 1 and 2 will be generated? Assuming these are global queries. Explain how various fragments will be combined to do so.

4. Implement the database at least using a centralized database management system.

2.6 ASSUMPTION DEPENDENCIES

Let us assume that this is a distributed airline management system and it is used in the following application:

- A request for booking/cancellation of a flight from any source to any destination, giving connected flights in case no direct flight between the specified Source-Destination pair exist.

- Calculation of high fliers (most frequent fliers) and calculating appropriate reward points for these fliers.
 Assuming both the transactions are single transactions, we have designed a distributed database that is geographically dispersed at four cities Delhi, Mumbai, Chennai, and Kolkatta as shown in fig. below.

3. SYSTEM FEATURES

- **DESCRIPTION and PRIORITY**
  The airline reservation system maintains information on flights, classes of seats, personal preferences, prices, and bookings. Of course, this project has a high priority because it is very difficult to travel across countries without prior reservations.

- **STIMULUS/RESPONSE SEQUENCES**
- Search for Airline Flights for two Travel cities
- Displays a detailed list of available flights and makes a "Reservation" or Books a ticket on a particular flight.
- Cancel an existing Reservation.
- **FUNCTIONAL REQUIREMENTS**
  Other system features include:

  DISTRIBUTED DATABASE:

Distributed database implies that a single application should be able to operate transparently on data that is spread across a variety of different databases and connected by a communication network as shown in the below figure.
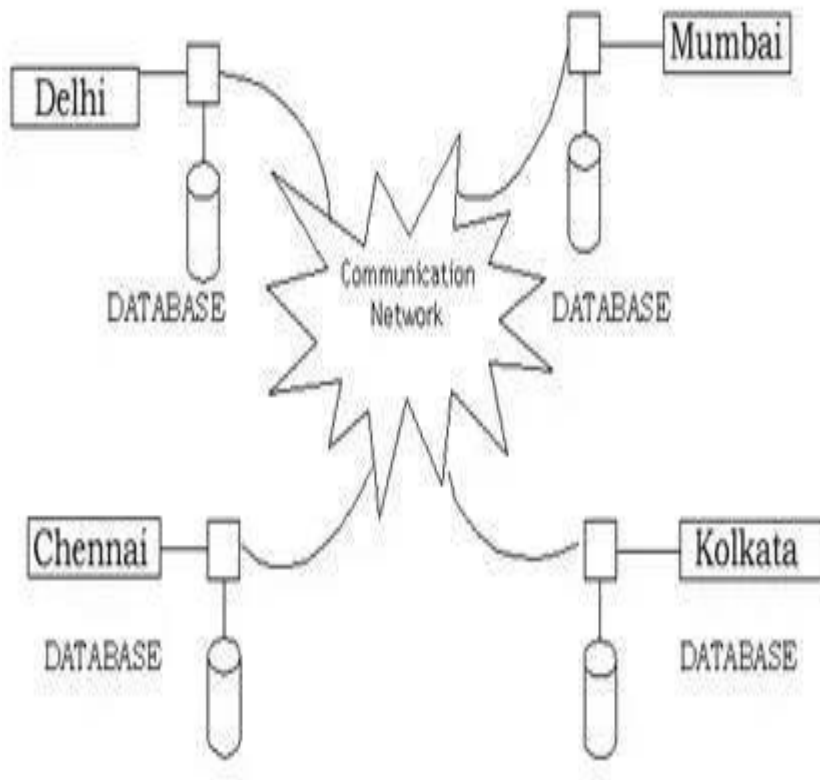
Distributed database located in four different cities

CLIENT/SERVER SYSTEM

The term client/server refers primarily to an architecture or logical division of responsibilities, the client is the application (also known as the front-end), and the server is the DBMS (also known as the back-end).

A client/server system is a distributed system in which,

- Some sites are client sites and others are server sites.
- All the data resides at the server sites.
- All applications execute at the client sites.

4. EXTERNAL INTERFACE REQUIREMENTS

4.1 USER INTERFACES

- Front-end software: Vb.net version

- Back-end software: SQL+
4.2 HARDWARE INTERFACES

- Windows.

- A browser that supports CGI, HTML & Javascript.
4.3 SOFTWARE INTERFACES

Following are the software used for the flight management online application. *<<Include the software details as per your project >>*

Software used	Description
Operating system	We have chosen Windows operating system for its best support and user-friendliness.
Database	To save the flight records, passengers records we have chosen SQL+ database.
VB.Net	To implement the project we have chosen Vb.Net language for its more interactive support.

4.4 COMMUNICATION INTERFACES

This project supports all types of web browsers. We are using simple electronic forms for reservation forms, ticket booking etc.

5. NONFUNCTIONAL REQUIREMENTS

5.1 PERFORMANCE REQUIREMENTS

The steps involved to perform the implementation of the airline database are listed below.

A) E-R DIAGRAM

The E-R Diagram constitutes a technique for representing the logical structure of a database in a pictorial manner. This analysis is then used to organize data as a relation, normalizing relation, and finally obtaining a relation database.

- **ENTITIES:** These specify distinct real-world items in an application.

- **PROPERTIES/ATTRIBUTES:** These specify properties of an entity and relationships.

- **RELATIONSHIPS:** These connect entities and represent meaningful dependencies between them.
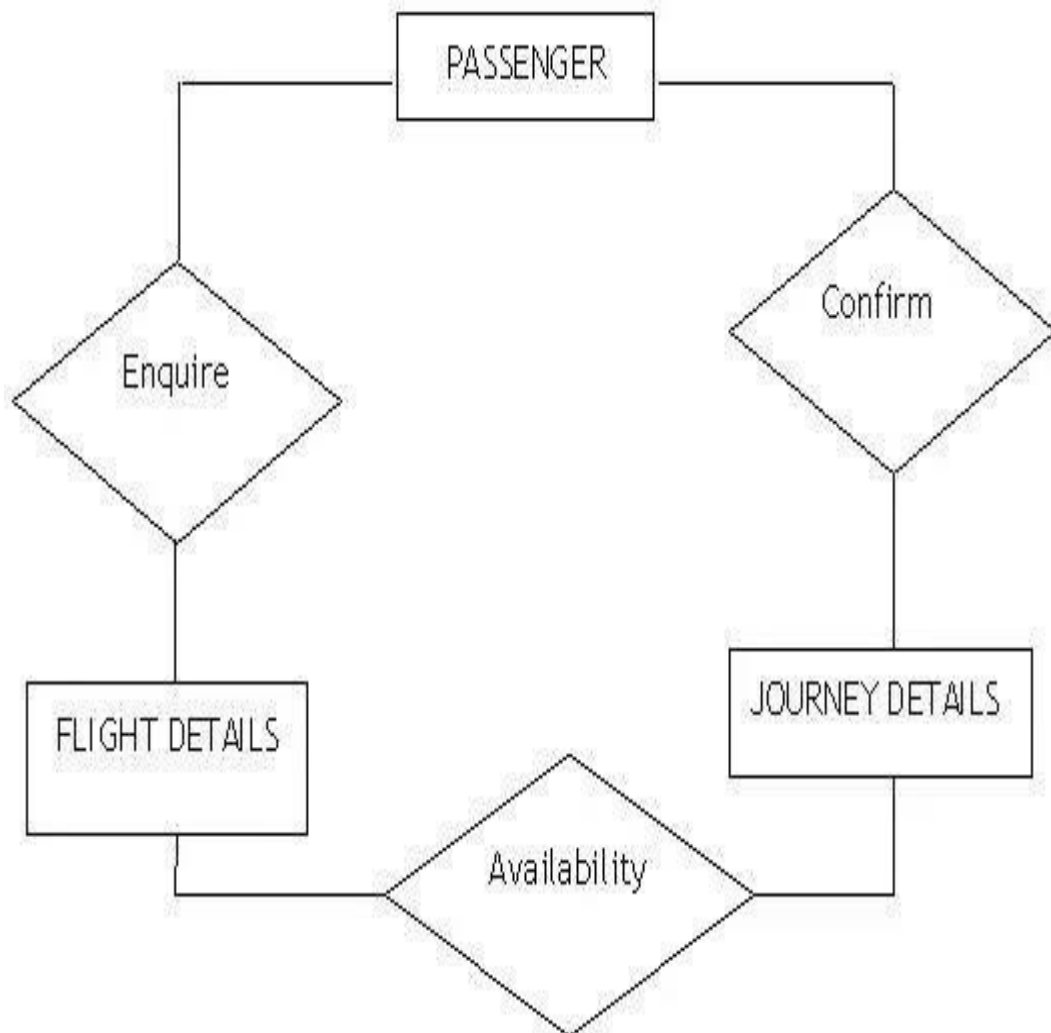
the diagram shows the ER diagram of the airline database

B) NORMALIZATION:

The basic objective of normalization is to reduce redundancy which means that information is to be stored only once. Storing information several times leads to a wastage of storage space and an increase in the total size of the data stored.

If a database is not properly designed it can give rise to modification anomalies. Modification anomalies arise when data is added to, changed, or deleted from a database table. Similarly, in traditional databases as well as improperly designed relational databases, data redundancy can be a problem. These can be eliminated by normalizing a database.

Normalization is the process of breaking down a table into smaller tables. So that each table deals with a single theme. There are three different kinds of modifications of anomalies and formulated the first, second, and third normal forms (3NF) is considered sufficient for most practical purposes. It should be considered only after a thorough analysis and complete understanding of its implications.

5.2 SAFETY REQUIREMENTS

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery method restores a past copy of the database that was backed up to archival storage (typically tape) and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed up log, up to the time of failure.

5.3 SECURITY REQUIREMENTS

Security systems need database storage just like many other applications. However, the special requirements of the security market mean that vendors must choose their database partner carefully.

5.4 SOFTWARE QUALITY ATTRIBUTES

- **AVAILABILITY:** The flight should be available on the specified date and specified time as many customers are doing advance reservations.
- **CORRECTNESS:** The flight should reach start from the correct start terminal and should reach the correct destination.
- **MAINTAINABILITY:** The web application must support regular updates to ensure compatibility with the latest web technologies, Bug fixes, and security patches.
- **USABILITY:** The flight schedules should satisfy a maximum number of customers' needs. The application should be accessible on multiple devices, including desktop and mobile.