***DBMS mini project***

* ***Problem Statement -*** Design an application to handle the database of hotel consist of all customer data, and room data. Provide functionality for handling that data.
* ***Project Idea -*** Many small Hotels do not have their own websites as maintaining websites need external agency. Nowadays everyone wants simple and easy to use systems to manage their business. So, I decided to design a web-based application to handle the data of hotel and help in their business management. As handling data of all customers , room is tedious work so create a system that can handle all data at one place itself in a structured way by applying the concepts of Database management systems.
* ***Project Scope*** - The system will be designed to provide an electronic version of the Hotel management system. This system will be for administrative purpose.

The main objective of system is to provide –

• 24/7 available system to handle and analyse data of hotel.

• All data at one place system.

The “***HOTEL ROOM BOOKING SYSTEM***” project has been designed keeping in mind both employee and guest. This a system where all functions are available at one place which is not in traditional system. Traditional system has only registration facility.

The application allows the guests to:

(1) Check-in

(2) Checkout

(3)Book Room.

These features have been represented by functions in the program.

***Functional requirements:***

• Booking of the customers.

• Automatic billing of the rooms based on their types.

• Adding, Updating, Deleting the customer/guest’s data.

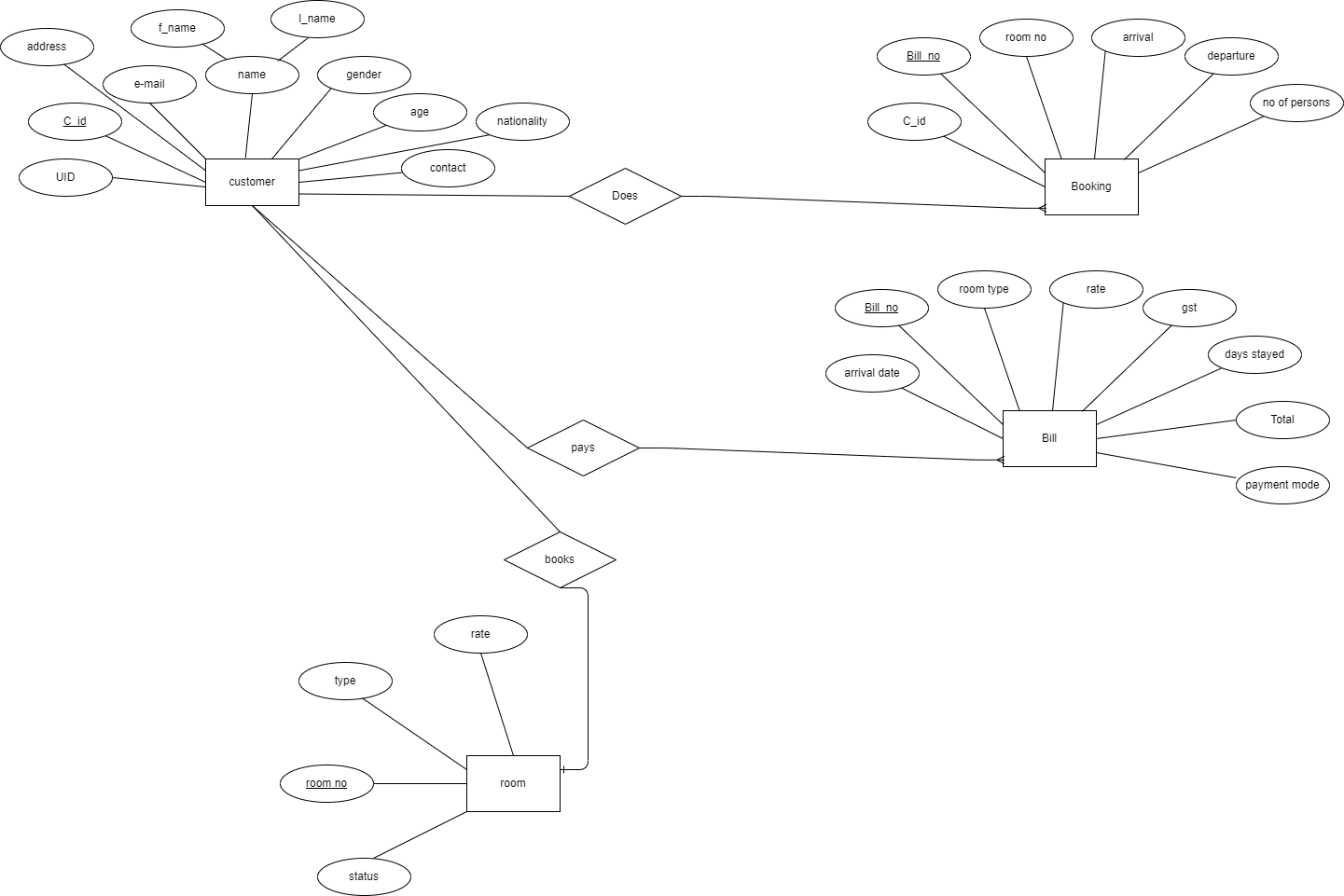
• Adding, updating data about rooms.

***Non-Functional requirements:***

• Fast Performance of the System.

• Reliability, usability, maintainability, availability, flexibility of software.

***ER Diagram –***



***Schema of all the tables –***

1. *Customer=(C\_id, f\_name, l\_name, gender, age, nationality, UID, address, email, contact\_no)*
2. *Booking=(Bill\_no ,C\_id, room\_no, arrival, departure, no\_of\_persons)*
3. *Billing = (Bill\_no,arrival\_date, room\_type, rate, gst, days\_stayed, total, payment\_mode)*
4. *Room = (room\_no, room\_type, rent)*

***Normalization applied –***

*• First Normal Form(1NF):- As all the attributes in a relation above have atomic domains also values in an atomic domain are indivisible units. Hence the database design is in the first normal form.*

*• Second Normal Form(2NF):-Above database is in first normal form , if we have customer and booking table together then every non-key attributes are not identified by the use of primary key . So we removed this by splitting the customer table into 2 table i.e customer and booking table.*

*• Third Normal Form(3NF):- Above database is in first and second normal form , No non-prime attribute is transitively dependent on prime key attribute. There is no case that a non-prime attribute is determined by another non-prime attribute Similarly we have converted it into 3NF*

*• Table customer and booking we divided so that we can exclude the anomalies :*

*Example:*

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *C\_id* | *Name* | *Sex* | *Address* | *Nation-*  *ality* | *Email* | *Contact* | *B*  *\_no* | *Room*  *\_no* | *Arr*  *ival* | *Depat*  *ure* | *Per-*  *sons* |
| *C101* | *Adi* | *M* | *pune* | *Indian* | *A@gmail* | *97644-*  *00859* | *1* | *1* | *29-*  *10-*  *22* | *1-11-*  *22* | *2* |
| *C101* | *Adi* | *M* | *pune* | *Indian* | *A@gmail* | *97644-*  *00859* | *2* | *2* | *2-11-*  *22* | *5-11-*  *22* | *1* |

*• In hotel one customer can came for booking any number of times but if we keep registration and booking together it causes redundancy of data fields like name, sex, address, contact, email, nationality and also if we want to delete booking data of person its personal data will also get lost and we have to insert it again next time so this causes insert and delete anomalies.*