A red and white sign

Description automatically generated with low confidence

INSTITUTE OF BUSINESS ADMINISTRATION

KARACHI

**PROJECT REPORT**

(DATABASE SYSTEMS)

[HOTEL MANAGEMENT DATABASE SYSTEMS]

GROUP MEMBERS

|  |  |  |
| --- | --- | --- |
| S.NO | Name | ERP |
| 1 | HUDA HASHIM | 18636 |
| 2 | MUHAMMAD AHSAN HAMEED | 19755 |
| 3 | AYESH REHMAN | 19691 |
| 4 |  |  |

ABSTARCT

The Topic for our project is Hotel Database Management System. It is one of the most commonly used and in-demand application that is used around the world to access, manage and manipulate the data that is gathered as well as manage the tasks that are required to run the operations of a hotel smoothly. It is usually designed to simplify and improve efficiency of the hotel’s operations.

This report is split into multiple chapters that are as follows:

* Problem Statement

This part of the report will deal with all the problems that the past and current hotel management systems face and how our project would help tackle those difficulties.

* Requirements

This part would highlight the main requirements of our project and explain them in detail about how to implement those and what advantage would it provide us

* Data Modeling

It split into 2 sections

1. Conceptual Model

The conceptual model would help to identify the relationship between the different tables using ERD diagram in a simple way.

1. Logical Model

The Logical model will explain the conceptual model in much detail containing the attributes and all the keys alongside the data represented by the conceptual model.

There will be a Power Point Presentation as well that would be presented alongside.

Table of Contents

[1 Problem Statement 4](#_Toc81767773)

[2 Requirements Specifications 6](#_Toc81767774)

[3 Data Modeling 9](#_Toc81767775)

[3.1 Conceptual Model 9](#_Toc81767776)

[3.2 Logical Model 12](#_Toc81767777)

# Problem Statement

[ 2 PAGES]

Hotels are being managed in multiple ways such as manual paperwork, computerized excel filing and as well as a combination of CRM (Customer Relationship Management Systems), PMS (Property Management Systems) and other management systems.

For the manual paperwork, we all know how that is extremely outdated and an obsolete way of keeping records of data in any market. First, it is bad for the environment as it leads to excessive use of paper which means more deforestation. Secondly, It is very difficult to store huge amounts of paperwork as it has to be stored physically in such a way that it does not get damaged and could easily be accessed if required. Searching for records in paperwork is a huge dilemma on its own because even if everything is sorted and kept accordingly, you still have to go through a section to access the data you require, and it is slow as it has to be done by a human.

The Excel filing of data helps to counter the environmental concern and the large space required for physical storage, which is required by manual paperwork, and it is much easier to view and access data from the computerized files as the computer can look up records much faster than humans. But still this has some issues which come up if the data size is huge. First of all it is very difficult to connect excel files to one another and manipulate data. This is due to various reasons such as the old excel files may not work smoothly with the new updated excel files, The queries that you can run are very limited and not complete, the maximum number of records for old excel is 65536 while the new one is around 1 million and there is difficulty in keeping the data secured and it is not designed for collaborative work. There is a high amount of data repetition which results in wasted storage, duplicate records and consequently slower searching and handling of data.

The use of different Management Systems can be done using normal filing which has all the same drawbacks as excel filing, sometimes maybe worse, and it can be done by using database-based systems. The issues with this kind of approach is that you have to buy or create multiple systems for multiple departments of your hotel. For example, you will need PMS for Room Keeping, CMS for Bookings, a separate database for inventory, another one for your employees and so on, all of which are not integrated and will have to be accessed separately and require more resources and technical support to be run. It will require multiple and different staff trainings and it will be having a lot of duplicate records such as a customer’s record would be saved in CMS as well as PMS if he books a room and orders food through room service.

The need for our database system is extreme as hoteliers would like to have as efficient of a system as they can while having minimal run cost. The database system we suggest would be containing all the different attributes of all the different management systems in a single application. This would eradicate the need for multiple trainings as well as multiple technicians to manage as a small group of qualified technicians would be able to handle a single database system efficiently. This would also help to keep the data integrated as it would be saved by a single DBMS, and it could be accessed and manipulated by the same programs given that he has the given rights. You can create users and add security by limiting the access of data viewing, creating, or updating. This would allow only supervisors to have complete control while the employees would have fewer data rights which makes sense as customers provide their personal data on trust which should be kept private from someone with evil intentions. The data duplication would be reduced drastically by normalizing the data and linking all the relational schemas in a good manner. This would help in uniquely identifying our customers or staff and it would be much easier to use the data for anticipation or customer communication.

# Requirements Specifications

[3-4 PAGES]

The main objective of our Hotel Database Management System is to make hotel management much simpler and more effective in order to meet the satisfaction levels of customers as well as make best use of the available resources and make manipulation of the data easier and faster.

Apart from this, our Hotel Database Management System aims to provide the advanced security, thus protecting the confidentiality of all information stored in the system and limiting the access control.

The Major Requirements of our Hotel Database Management System could be divided as follows:

**PAYMENT**

Our Program must be able to calculate the amount that has to be paid by the customers as well as charge the customers according to the rooms they booked, the services they ordered and the number of days they would be staying for. If the customer is a loyal and recurring visitor, they might even get a discount or waive on some sort of service they order.

**STAFF MANAGEMENT (EMPLOYEES AND MANAGERS)**

The DBMS should also be able to manage the staff that the hotel has working for it. It would have their personal details, their department, their salary, their working hours etc. The program must be able to see if an employee is working or free, if not then where are they working and in which rooms have their duties been assigned or in what department are they working. All this will most likely be supervised by a manager who would be supervising each department such as laundry, kitchen, room service and so on. So, the managers should have access to employee data.

**SERVICES**

The services section would include all the extra options that a customer might order. This would have all its details such as what the service is, how much does it cost, what are the timings for it and so on.

For example, some customers might add breakfast option to their billing while some customers might want access to gym facilities or swimming pool facilities. Some customers might require their laundry to be done. So, all the details about the services and whom it is requested by should be managed and linked with billing/payment.

**ACCESS**

One of the very important, if not the most, part of our DBMS is security and control where we must keep unwanted users out as well as protect sensitive information such as customer details, employee details or profits/loss.

This is mainly achieved by just limiting access to the computers themselves and password protecting them, but this is also strengthened by limiting the access of sensitive data of customers/employees to the employees and even managers so that they can not go through that data without interfering with the work that they are supposed to do.

The finances must be kept private from almost everyone and must be accessed by a handful of people so payment records should have limited access too.

.

**Searching**

To view a specific field in the database and to edit we first need to find it. DBMS allows data to be searched through thousands of record simply by the use of some queries. Searching can be done however the user requires, by name, by date, or by any other existing attribute. We can also access different attributes of different entities and view them simultaneously.

**Customer Reservation**

Customers will be able to make reservations with ease through the website where all the provided details would be fetched and stored in our database automatically.

**Relational Schema**

Our DBMS will be very efficient and easier to manage as we have divided our data into multiple entities which are linked with one another using foreign keys. This helps to remove redundant data and make our entities well integrated. As our relations are linked with one another, we can view attributes of one table using the attribute of another table.

# Data Modeling

## Conceptual Model

This Hotel database model consists of 8 related tables:

1. **Bookings:** The first and most important entity of this system is the bookings table. The fields in this table consist of all the bookings made in the hotel with the most relevant details that are required when booking a hotel room. Every other entity in this system is linked to this table either directly or indirectly.

**Relationships:**

Customer= Many to One.

Rooms=Many to one.

Extra-Bookings=one to many.

1. **Customers:** This Table consists of all the relevant details of a customer and keeps them separate from the booking table. By keeping this information separately it is easier to lookup a particular customer or to notify all the old customers when the hotel puts up a deal or a discount.. Moreover, its efficient to keep customer data in record for the future in case of a returning customer or if the same customer makes more than 1 bookings, to avoid repetition in the booking table.

**Relationships:**

Booking= One to Many .

1. **Rooms:** Most big hotels have a range of types of rooms available. This Table keeps details of all the different types of rooms with their description. A single room or room of same type can be booked multiple times, so by putting all these details in the booking table, there will be a lot of repetitive data, to avoid this we create a separate table for rooms and use only the room number to link it with the bookings.

**Relationships:**

Booking= One to Many.

Room Type=

Staff=Many to one.

1. **Extras:** When making a booking many customers have preferences other than what is included in their rooms, every hotel offers several extra facilities which customers can add to their bookings. All these facilities with their details are stored in this table.

**Relationships:**

Extra-Bookings=One to Many.

1. **Extra-Bookings:** When a booking asks for extras, that information is stored in this table. Extras cannot be directly specified in the booking table along with rest of the information because its possible that one booking can have more than one extra which would cause repetition of fields.

**Relationships:**

Booking= Many to One .

Extras=Many to One.

1. **Staff:** We have also created a table for all the staff working in the hotel. It consists of the details of all staff including their job description, every new employee joining the hotel will be added here, this information is important to keep on hand for example in case of any emergency or leave in a department you can easily find out all employees working in that department to deal with the situation effectively.

**Relationships:**

Room Staff= One to Many.

Department-Manager=Many to One.

1. **Department-manager:** This table consists of all the different departments of a hotel and their managers. It is linked with the staff table to easily find out who is the manager of whom simply by their department.

**Relationships:**

Staff= One to Many.

1. **Room Staff**

This table consists of all the staff assigned to the room service. It is linked with the staff table and the rooms table to find the room and the staff.

**Relationships**:

Staff=

Rooms=

## Logical Model

Tables:

1. **Bookings**
2. **Customer**
3. **Rooms**
4. **Room Type**
5. **Room Staff**
6. **Staff**
7. **Department Manager**
8. **Extra**
9. **Extra Booking**

Diagram

Description automatically generated

1. **Bookings**

* BookingID: (primary key) unique id assigned to each booking.
* CustomerID (foreign key) from customers table, to link the details of the customer to the booking. Can occur more than once.
* Roomnum (foreign key) from Rooms table to link the details and price of the room with the booking. Can occur more than once.
* Check-in: the date customer will start their stay
* Checkout: the date customer will end their stay
* Nights: no. of nights between check-in and checkout
* Status: indicates whether the booking is active, completed or going to start.
* Payment\_Status: indicates whether payment has been done or not.
* Amount: Total Amount of the Bill that has been calculated (Room + Extras)
* Method: Method of Payment (Cash/Card)
* Date: Date of Payment

1. **Customer**

* CustomerID: (Primary Key) Unique id assigned to each customer.
* First\_name: first name of the customer.
* Last\_name: last name of the customer.
* Address: address of the customer
* Email: email of the customer.
* Phone: phone number of the customer.

1. **Rooms**

* Roomnum: (Primary Key) unique identifier of each room.
* Roomtype\_id (Foreign Key): unique identifier of the types of rooms.
* MaxPersons: defines the number of guests allowed in the room.
* Availability: whether the room is already booked during the chosen dates or free.
* Price: defines the price of room per night.

1. **Room Type**

* Roomtype\_id (Primary Key): unique identifier for the types of rooms.
* Roomtype\_desc: description of the room whether it’s a suite room or studio etc.

1. **Room Staff**

* Roomnum: (Foriegn Key) unique identifier of each room.
* Cleaning\_Date: consists of date of the room service

1. **Staff**

* StaffID: (Primary Key) unique id assigned to each staff.
* First\_name: first name of the staff member
* Last\_name: last name of the staff member.
* Address: address of the staff member
* Phone: phone number of the staff member
* Job: description of the employee whether it’s a cleaner, chef or manager etc.
* Department\_id: (Foreign Key) From the Department-Manager table. Unique Department Identifer

1. **Department Manager**

* Department\_id: (Primary Key) Unique identifier of the department.
* StaffID: (Foreign Key) Manager of each department. Manager is a staff with job=manager.
* Dept Name: Name/Desc of the Department

1. **Extra**

* ExtrasID: (Primary Key) unique id assigned to each extra facility
* desc: Type of extra for example Breakfast, Extra Bed, cab service Etc
* Price: Price of the facility or add on.

1. **Extra Booking**

* ExtraID: (Primary Key, Foreign Key) from Extras table, can occur more than once as several bookings can ask for the same extra facility, is made a composite key with BookingID
* BookingID: (Primary Key, Foreign Key) from Bookings Table, can occur more than once as one booking can require more than one kind of extra facilities. Made into a composite key with ExtrasID.
* Extra-Booking\_Date: Date of the booking of extra

# Physical Database Design/Updated Data Model

## Constraints

**constraint val\_email CHECK( cust\_email like '%@%') );**

**constraint managerfk foreign key(staff\_id) references Staff(staff\_id)**

**constraint departfk foreign key(Dept\_id) references Departments(Dept\_id)**

**constraint roomtypefk foreign key(roomtype\_id) references Roomtype(roomtype\_id)**

**constraint roomservicefk foreign key(staff\_id) references Staff(staff\_id)**

**constraint roomnumfk foreign key(roomnum) references Rooms(roomnum)**

**constraint roomservicePK primary key(roomnum,cleaning\_date)**

**constraint Bcustidfk foreign key(cust\_id) references Customer(cust\_id)**

**constraint Broomnumfk foreign key(roomnum) references Rooms(roomnum)**

**constraint Ebookingidfk foreign key(booking\_id) references Bookings(booking\_id)**

**constraint extrasfk foreign key(extra\_id) references Extras(extra\_id)**

**constraint extrasck primary key(booking\_id,extra\_id)**

**SEQUENCE customer\_id\_seq**

**SEQUENCE booking\_id\_seq**

## Tables

Table

Description automatically generated Table

Description automatically generated BOOKINGS TABLE CUSTOMER TABLE

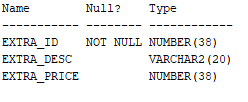
Text

Description automatically generated Text

Description automatically generated

Department-Manager Staff

Text, table

Description automatically generated 

EXTRA BOOKINGS EXTRAS

Table

Description automatically generated Table

Description automatically generated with low confidence

ROOM-STAFF ROOMS

Text

Description automatically generated with medium confidence

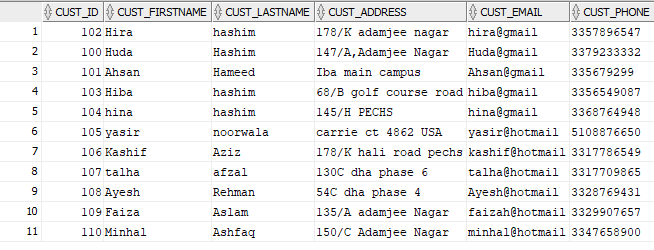
ROOM TYPE

## Sample Data

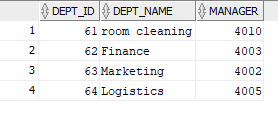
Table

Description automatically generated

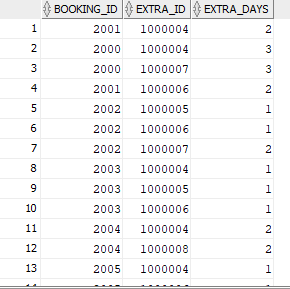
Bookings



CUSTOMER



DEPARTMENTS

 Table

Description automatically generated

EXTRA BOOKINGS EXTRAS

Table

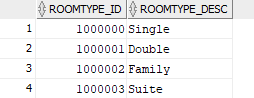
Description automatically generated

ROOMS

Table

Description automatically generated

ROOM STAFF



ROOM TYPE

Graphical user interface, application, table, Excel

Description automatically generated

STAFF

# User Interface Design

So, we made our design using Java as we went for a Desktop Application.

We used Net Beans for that.

Graphical user interface, application

Description automatically generated

**Login Page**

Graphical user interface

Description automatically generated

**Main Menu**

Graphical user interface

Description automatically generated

**Bookings**

Graphical user interface, website

Description automatically generated

**New Bookings/Check In**

Graphical user interface, timeline

Description automatically generated

**Customer Details**

Graphical user interface

Description automatically generated

**Payment/Check Out**

Graphical user interface

Description automatically generated

**Room Management/Room Service**

Graphical user interface

Description automatically generated

**Staff Management**

**Thank You**