

PHASE 1:

1. Introduction:

The Hostel Management System database aims to streamline and automate the management of hostels within our organization. It will provide a centralized platform for efficient administration, improved data management, and enhanced user experience. This system will replace the existing manual processes, which have proven to be inadequate in meeting the growing demands and complexities of hostel management.



1. Goals/Objectives

- ❖ Easily manage the customer details,
- ❖ room/room status,
- ❖ room booking,
- ❖ hostel branch,
- ❖ employees,
- ❖ Payments and transactions.
- ❖ With Complete Management System, a lot of repetition can be easily evaded which has reduced the data redundancy.

2. Features:

Database Design for Hostel Management System:

- ❖ Creation of building block information
- ❖ Provision of various room types
 - (single/twin/single with air-con room/twin with air-con room/romantic with air-con room/group room for many/luxury with air-con room and family room/family with air-con room)
- ❖ Room allotment to the traveler
- ❖ Traveler check in and check out
- ❖ Monitoring visitors and guest register
- ❖ Transfers of room

3. Entities and Attributes

Customer

customer_id
customer_role_id
customer_username

Roles

role_id
role_name
role_decs

Permission

per_id
per_name
per_role_id
per_module

Allottees

alot_name
alot_id
alot_address
alot_mobile
alot_email
alot_password

Hostel

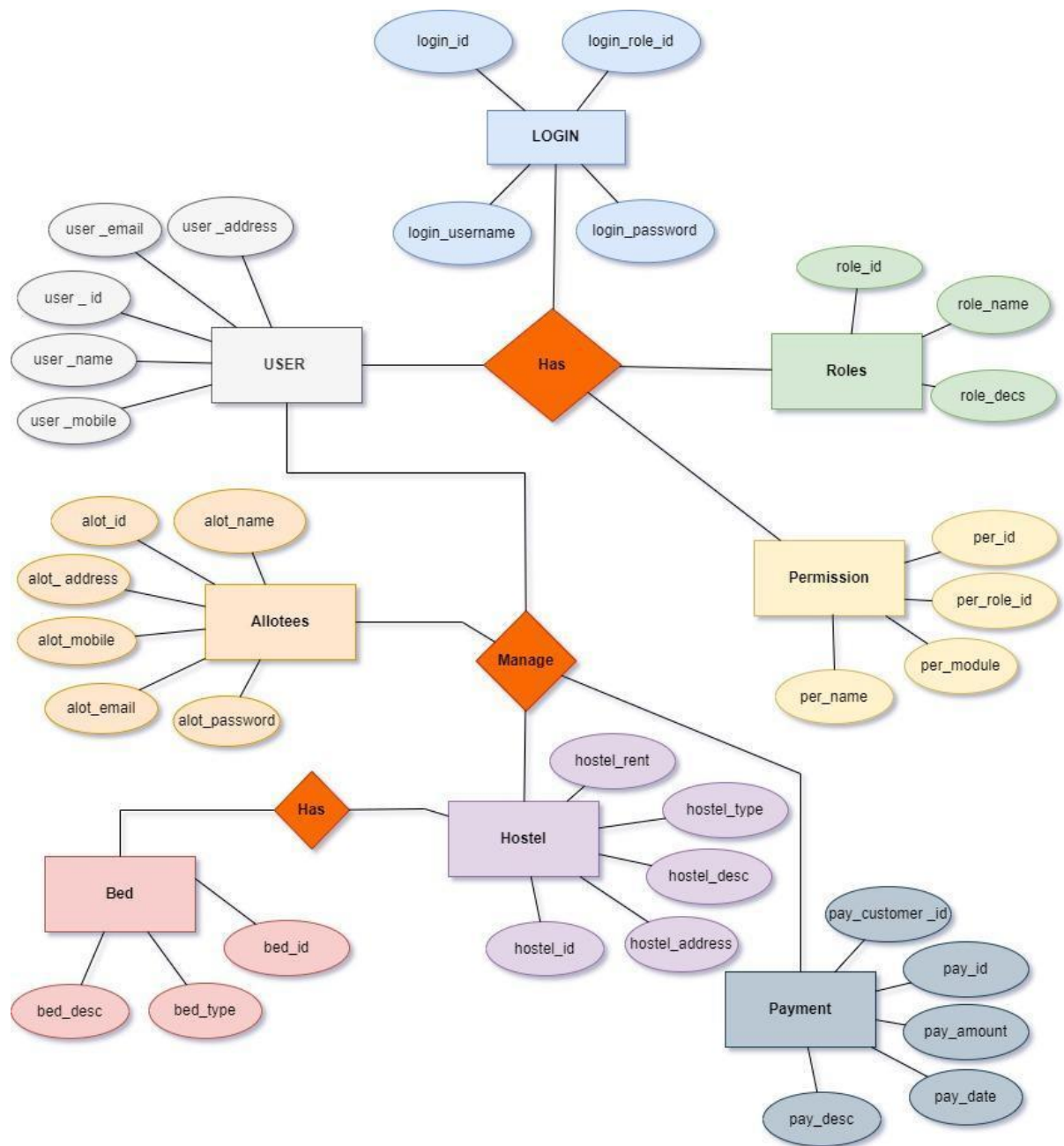
hostel_type
hostel_desc
hostel_address
hostel_id
hostel_rent

Payment

pay_customer_id
pay_id
pay_amount
pay_date
pay_desc

PHASE 2:

5. ER DIAGRAM



6. RELATIONAL TABLE

The screenshot shows a database management interface with a tabbed window. The active tab is 'booking'. Below the tabs is a toolbar with various icons. The SQL editor contains the query: `SELECT * FROM hosteldb.booking;`. Below the editor is a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The result grid displays a table with 8 columns: Booking_ID, customer_ID, Room_ID, Booking_date, check_in_date, check_out_date, and no_of_nights. There are 6 rows of data.

Booking_ID	customer_ID	Room_ID	Booking_date	check_in_date	check_out_date	no_of_nights
1001	1	101	20/09/09	12/10/09	10/02/11	45
1002	2	102	20/09/09	12/10/09	10/02/11	45
1003	2	102	20/09/09	12/10/09	10/02/11	45
1004	3	103	20/09/09	12/10/09	10/02/11	45
1005	2	102	20/09/09	12/10/09	10/02/11	45
1006	1	101	20/09/09	12/10/09	10/02/11	45

CREATE TABLE BOOKING

(Booking_ID int,

customer_ID int,

Room_ID int,

Booking_date varchar(50),

check_in_date varchar(10),

check_out_date varchar(10),

no_of_nights int

);

For creating table(Booking) query !

INSERT INTO Booking

(Booking_ID,customer_ID,Room_ID,Booking_date,check_in_date,check_out_date,no_of_nights) VALUES

(1001,1,101,'20/09/09','12/10/09','10/02/11',45);

INSERT INTO BOOKING

(Booking_ID,customer_ID,Room_ID,Booking_date,check_in_date,check_out_date,no_of_nights) VALUES

```
(1002,2,102,'20/09/09','12/10/09','10/02/11',45);
```

```
INSERT INTO Booking
```

```
(Booking_ID,customer_ID,Room_ID,Booking_date,check_in_date,check_out_date,no_of_nights) VALUES
```

```
(1003,2,102,'20/09/09','12/10/09','10/02/11',45);
```

```
INSERT INTO Booking
```

```
(Booking_ID,customer_ID,Room_ID,Booking_date,check_in_date,check_out_date,no_of_nights) VALUES
```

```
(1004,3,103,'20/09/09','12/10/09','10/02/11',45);
```

```
INSERT INTO Booking
```

```
(Booking_ID,customer_ID,Room_ID,Booking_date,check_in_date,check_out_date,no_of_nights) VALUES
```

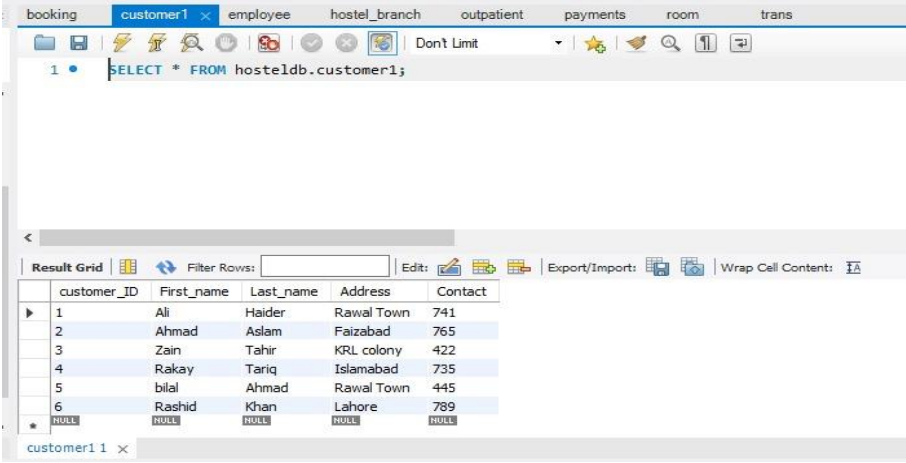
```
(1005,2,102,'20/09/09','12/10/09','10/02/11',45);
```

```
INSERT INTO Booking
```

```
(Booking_ID,customer_ID,Room_ID,Booking_date,check_in_date,check_out_date,no_of_nights) VALUES
```

```
(1006,1,101,'20/09/09','12/10/09','10/02/11',45);
```

Inserting Values in Booking table



	customer_ID	First_name	Last_name	Address	Contact
1	1	Ali	Haider	Rawal Town	741
2	2	Ahmad	Aslam	Faizabad	765
3	3	Zain	Tahir	KRL colony	422
4	4	Rakay	Tariq	Islamabad	735
5	5	bilal	Ahmad	Rawal Town	445
6	6	Rashid	Khan	Lahore	789
*	NULL	NULL	NULL	NULL	NULL

```
CREATE DATABASE Hostel_Management_System_project;
```

```
CREATE TABLE customer1
```

```
( customer_ID int,
```

```
First_name varchar(15),
```

```
Last_name varchar(15),
```

```
Address varchar(25),
```

```
Contact int
```

```
);
```

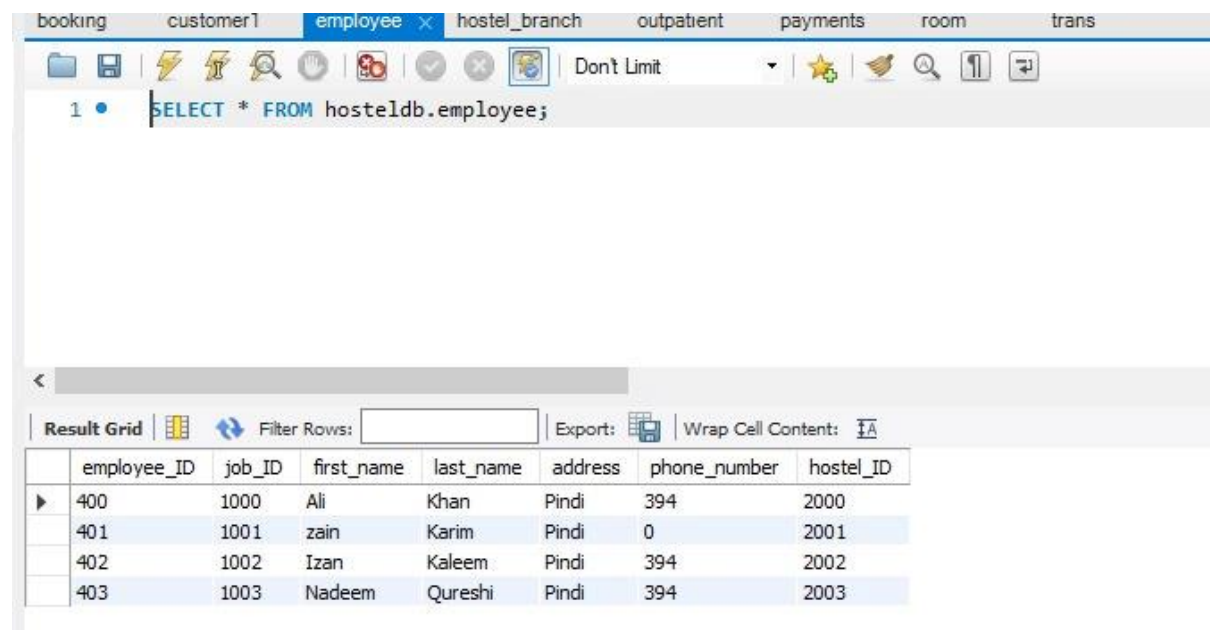
```
INSERT INTO customer1 (customer_ID,First_name,Last_name,Address,Contact) VALUES
```

```
(1,'Ali','Haider','Rawal Town',83479281741);
```

```

INSERT INTO customer1 (customer_ID,First_name,Last_name,Address,Contact) VALUES
(2,'Ahmad','Aslam','Faizabad',0937464765);
INSERT INTO customer1 (customer_ID,First_name,Last_name,Address,Contact) VALUES
(3,'Zain','Tahir','KRL colony',426475967422);
INSERT INTO customer1 (customer_ID,First_name,Last_name,Address,Contact) VALUES
(4,'Rakay','Tariq','Islamabad',57595735);
INSERT INTO customer1 (customer_ID,First_name,Last_name,Address,Contact) VALUES
(5,'bilal','Ahmad','Rawal Town',346666543445);
INSERT INTO customer1 (customer_ID,First_name,Last_name,Address,Contact) VALUES
(6,'Rashid','Khan','Lahore',00998767789);
select*from customer1;

```



The screenshot shows a database management interface with a query editor and a results grid. The query editor contains the SQL statement: `SELECT * FROM hosteldb.employee;`. The results grid displays the following data:

employee_ID	job_ID	first_name	last_name	address	phone_number	hostel_ID
400	1000	Ali	Khan	Pindi	394	2000
401	1001	zain	Karim	Pindi	0	2001
402	1002	Izan	Kaleem	Pindi	394	2002
403	1003	Nadeem	Qureshi	Pindi	394	2003

create table employee

```

(employee_ID int,
job_ID int,
first_name varchar(15),
last_name varchar(15),
address varchar(20),
phone_number int,
hostel_ID int
);
INSERT INTO employee(employee_ID, job_ID, first_name, last_name, address,
phone_number, hostel_ID)values
(400,1000,'Ali','Khan','Pindi',039485699488, 2000);
INSERT INTO employee(employee_ID, job_ID, first_name, last_name, address,
phone_number, hostel_ID)values
(401,1001,'zain','Karim','Pindi',000005699488, 2001);
INSERT INTO employee(employee_ID, job_ID, first_name, last_name, address,
phone_number, hostel_ID)values
(402,1002,'Izan','Kaleem','Pindi',0394856900938, 2002);
INSERT INTO employee(employee_ID, job_ID, first_name, last_name, address,
phone_number, hostel_ID)values
(403,1003,'Nadeem','Qureshi','Pindi',039482199488, 2003);

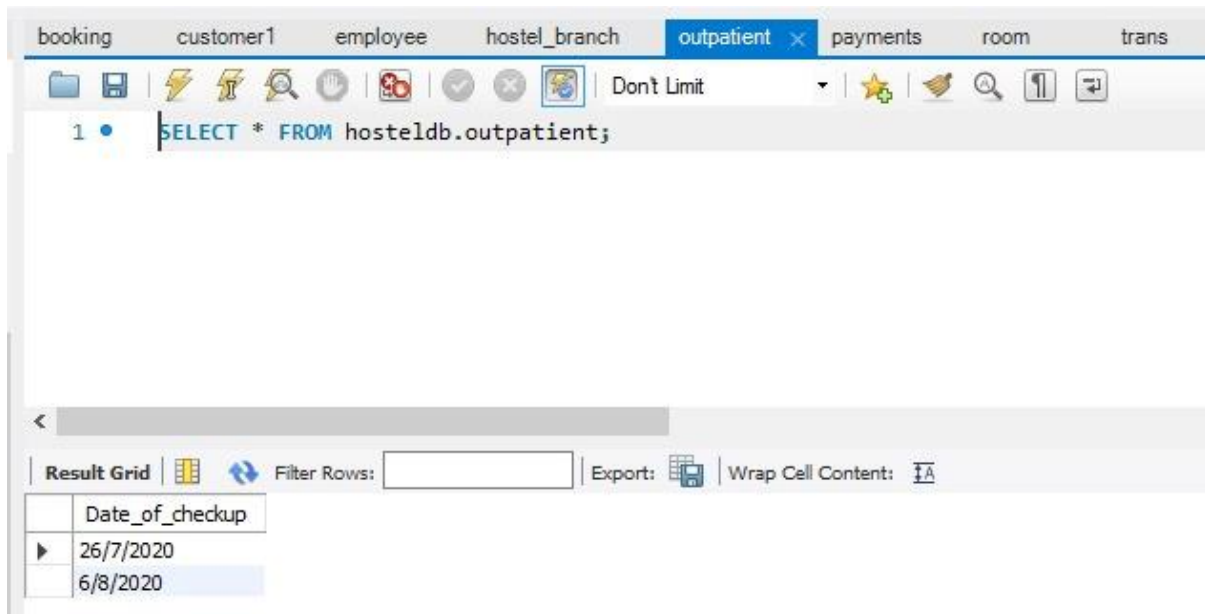
```

The screenshot shows a database management interface with a tab labeled 'hostel_branch' selected. The query editor contains the SQL statement: `SELECT * FROM hosteldb.hostel_branch;`. Below the query editor, the 'Result Grid' displays the following data:

hostel_branch_ID	hostel_name	address	ph_no	manager_ID
900	Shalimar	Pindi	97657899	500
902	pindora	Pindi	976543399	502
903	scheme 3	Pindi	97657000	503

CREATE TABLE Hostel Branch

```
( hostel_branch_ID int,
hostel_name varchar(15),
address varchar(25),
ph_no int,
manager_ID int
);
INSERT INTO Hostel_Branch (hostel_branch_ID,hostel_name, address,ph_no,manager_ID)
VALUES
(900,'Shalimar',"Pindi", 097657899, 500 );
INSERT INTO Hostel_Branch (hostel_branch_ID,hostel_name, address,ph_no,manager_ID)
VALUES
(901,'katarian',"Pindi", 09374895743, 501 );
INSERT INTO Hostel_Branch (hostel_branch_ID,hostel_name, address,ph_no,manager_ID)
VALUES
(902,'pindora',"Pindi", 0976543399, 502 );
INSERT INTO Hostel_Branch (hostel_branch_ID,hostel_name, address,ph_no,manager_ID)
VALUES
(903,'scheme 3',"Pindi", 097657000, 503 );
```

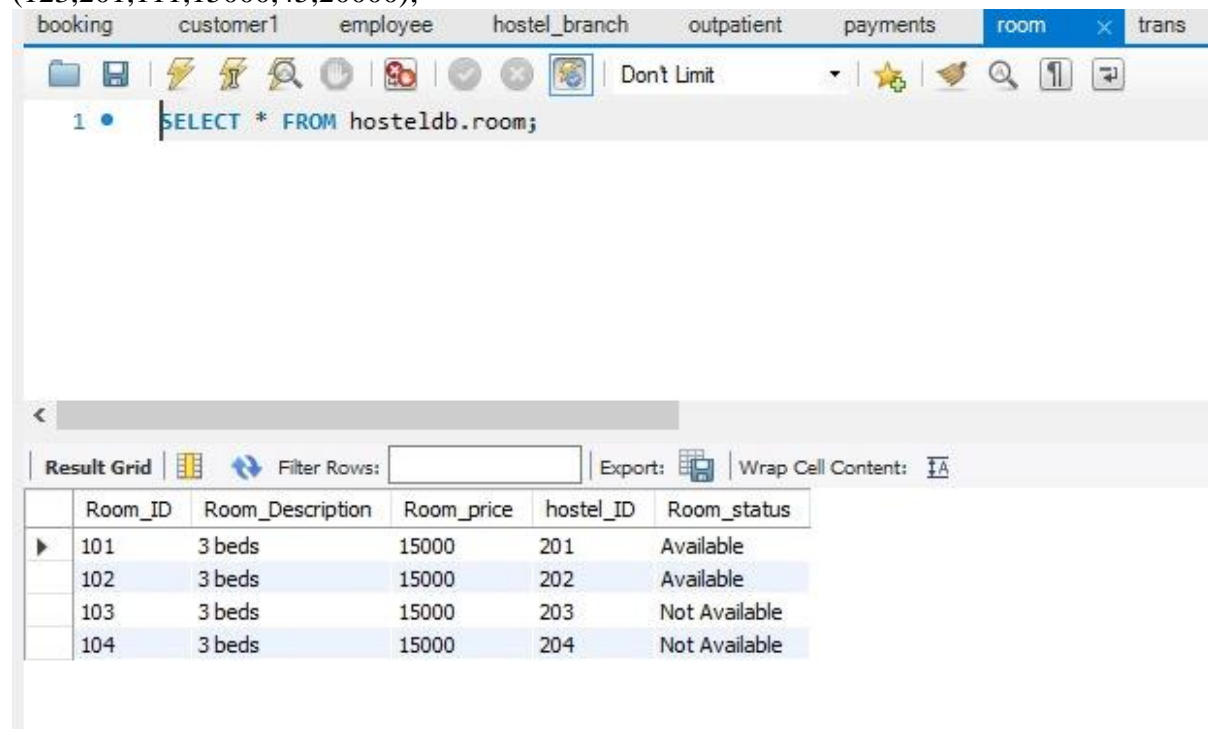
CREATE TABLE Outpatient

```
(Date_of_checkup varchar(45)
);
INSERT INTO Outpatient (Date_of_checkup) VALUES
("26/7/2020");
INSERT INTO Outpatient (Date_of_checkup) VALUES
("6/8/2020");
```

payment_ID	customer_ID	Booking_ID	room_price	no_of_nights	total_payments
123	201	111	15000	45	20000
123	201	111	15000	45	20000
123	201	111	15000	45	20000

```
create table payments
( payment_ID int,
customer_ID int,
Booking_ID int,
room_price int,
no_of_nights int,
total_payments int
);
INSERT INTO payments(payment_ID, customer_ID, Booking_ID,
room_price,no_of_nights,total_payments)VALUES
(123,201,111,15000,45,20000);
INSERT INTO payments(payment_ID, customer_ID, Booking_ID,
room_price,no_of_nights,total_payments)VALUES
```

```
(123,201,111,15000,45,20000);
INSERT INTO payments(payment_ID, customer_ID, Booking_ID,
room_price,no_of_nights,total_payments)VALUES
(123,201,111,15000,45,20000);
```



The screenshot shows a database management interface with a tab labeled 'room' selected. The SQL query editor contains the command: `SELECT * FROM hosteldb.room;`. Below the editor, the 'Result Grid' displays the following data:

Room_ID	Room_Description	Room_price	hostel_ID	Room_status
101	3 beds	15000	201	Available
102	3 beds	15000	202	Available
103	3 beds	15000	203	Not Available
104	3 beds	15000	204	Not Available

CREATE TABLE Room

```
( Room_ID int,
Room_Description varchar(50),
Room_price int,
hostel_ID int,
Room_status varchar(50)
);
INSERT INTO Room (Room_ID,Room_Description,Room_price,hostel_ID,Room_status)
VALUES
(101,'3 beds',15000,201,'Available');
INSERT INTO Room (Room_ID,Room_Description,Room_price,hostel_ID,Room_status)
VALUES
(102,'3 beds',15000,202,'Available');
INSERT INTO Room (Room_ID,Room_Description,Room_price,hostel_ID,Room_status)
VALUES
(103,'3 beds',15000,203,'Not Available');
INSERT INTO Room (Room_ID,Room_Description,Room_price,hostel_ID,Room_status)
VALUES
(104,'3 beds',15000,204,'Not Available');
```

booking	customer1	employee	hostel_branch	outpatient	payments	room	trans
---------	-----------	----------	---------------	------------	----------	------	-------


```
1 • SELECT * FROM hosteldb.trans;
```


	Trans_ID	customer_ID	Booking_ID	payment_ID	Employee_ID
▶	300	1	1005	1	1110
	301	2	1002	2	1111
	302	3	1003	3	1112

CREATE TABLE Trans

```
(Trans_ID int,  
customer_ID int,  
Booking_ID int,  
payment_ID int,  
Employee_ID int  
);
```

```
INSERT INTO Trans (Trans_ID, customer_ID, Booking_ID, payment_ID, Employee_ID)  
VALUES
```

```
(300,1, 1005, 0001, 1110);
```

```
INSERT INTO Trans (Trans_ID, customer_ID, Booking_ID, payment_ID, Employee_ID)  
VALUES
```

```
(301,2, 1002, 0002, 1111);
```

```
INSERT INTO Trans (Trans_ID, customer_ID, Booking_ID, payment_ID, Employee_ID)  
VALUES
```

```
(302,3, 1003, 0003, 1112);
```

```
SELECT *  
FROM hosteldb.customer1;  
select First_name, Contact from customer1;
```

Query 1 x booking booking customer1 employee hostel_branch outp

1 • SELECT *

2 FROM hosteldb.customer1;

3 • select First_name, Contact from customer1;

4

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	First_name	Contact
▶	Ali	741
	Ahmad	765
	Zain	422
	Rakay	735
	bilal	445
	Rashid	789

SELECT * FROM hosteldb.employee;
select employee_ID, job_ID, hostel_ID from employee;

Query 1 booking booking customer1 employee x hostel_branch

1 • SELECT * FROM hosteldb.employee;

2 • select employee_ID, job_ID, hostel_ID from employee;

3

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	employee_ID	job_ID	hostel_ID
▶	400	1000	2000
	401	1001	2001
	402	1002	2002
	403	1003	2003

SELECT * FROM hosteldb.room;
select room_price, total_payments from payments

Query 1 booking booking customer1 employee hostel_bra

1 • SELECT * FROM hosteldb.room;

2 • select room_price, total_payments from payments

Result Grid | Filter Rows: | Export: | Wrap Cell Cont

	room_price	total_payments
▶	15000	20000
	15000	20000
	15000	20000

```
SELECT * FROM hosteldb.customer1;
use hosteldb;
SELECT * FROM customer1
WHERE Address='Faizabad'
or
Address='Islamabad';
```

Query 1 booking booking customer1 x employee

1 • SELECT * FROM hosteldb.customer1;

2 • use hosteldb;

3 • SELECT * FROM customer1

4 WHERE Address='Faizabad'

5 or

6 Address='Islamabad';

7

Result Grid | Filter Rows: | Edit: |

	customer_ID	First_name	Last_name	Address	Contact
▶	2	Ahmad	Aslam	Faizabad	765
	4	Rakay	Tariq	Islamabad	735
*	NULL	NULL	NULL	NULL	NULL

```
SELECT * FROM hosteldb.employee;
select employee_ID, job_ID, hostel_ID from employee;
```

```
SELECT * FROM employee
WHERE first_name LIKE 'N%';
SELECT * FROM employee
WHERE first_name LIKE '%m';
```

Query 1 booking booking customer1 **employee** hostel_branch outpatient

3

```

4 • use hosteldb;
5 • SELECT * FROM employee
6   WHERE first_name LIKE 'N%';
7 • SELECT * FROM employee
8   WHERE first_name LIKE '%m';
9

```

Result Grid

employee_ID	job_ID	first_name	last_name	address	phone_number	hostel_ID
403	1003	Nadeem	Qureshi	Pindi	394	2003

```

SELECT * FROM customer1
WHERE Address IN ('KRL colony');

```

```

• SELECT * FROM customer1
WHERE Address IN ('KRL colony');

```

Result Grid

customer_ID	First_name	Last_name	Address	Contact
3	Zain	Tahir	KRL colony	422
NULL	NULL	NULL	NULL	NULL

```

SELECT * FROM hosteldb.room;
select room_price, total_payments from payments

```

```

use hosteldb;
SELECT * FROM room
WHERE Room_price BETWEEN 10000 AND 20000;

```


Query 1 booking booking customer1 employee hostel_branch ot

1 • SELECT * FROM hosteldb.room;

2 • select room_price, total_payments from payments

3

4 ✖ use hosteldb;

5 • SELECT * FROM room

6 WHERE Room_price BETWEEN 10000 AND 20000;

7

Result Grid Filter Rows: Export: Wrap Cell Content: IA

	Room_ID	Room_Description	Room_price	hostel_ID	Room_status
▶	101	3 beds	15000	201	Available
	102	3 beds	15000	202	Available
	103	3 beds	15000	203	Not Available
	104	3 beds	15000	204	Not Available

SELECT * FROM room
WHERE Room_price NOT BETWEEN 10000 AND 20000;

5 • SELECT * FROM room

6 WHERE Room_price NOT BETWEEN 10000 AND 20000;

7

Result Grid Filter Rows: Export: Wrap Cell Content: IA

	Room_ID	Room_Description	Room_price	hostel_ID	Room_status
--	---------	------------------	------------	-----------	-------------

SELECT customer1.customer_ID, room.Room_Description
FROM customer1
INNER JOIN room
ON customer1.customer_ID=room.Room_Description
ORDER BY customer1.First_name;

12 • SELECT customer1.customer_ID, room.Room_Description

13 FROM customer1

14 INNER JOIN room

15 ON customer1.customer_ID=room.Room_Description

16 ORDER BY customer1.First_name;

17

Result Grid Filter Rows: Export: Wrap Cell Content: IA

	customer_ID	Room_Description
▶	3	3 beds
	3	3 beds
	3	3 beds
	3	3 beds

FRONT END IMPLEMENTATION CODE:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HostelDatabase
{
    12 references
    internal class Booking
    {
        2 references
        public int ID { get; set; }
        4 references
        public String NAME { get; set; }
        4 references
        public String CHECKIN { get; set; }
        4 references
        public String CHECKOUT { get; set; }
    }
}
```

Database Declaration

Database Access

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography.X509Certificates;
using System.Text;
using System.Threading.Tasks;
using static System.Windows.Forms.VisualStyles.VisualStyleElement.TrackBar;

namespace HostelDatabase
{
    0 references
    internal class BookingDAO
    {
        string connectionString = "datasource=localhost;port=3306;username=root;password=root;database=hostel;";

        0 references
        public List<Booking> getAllBooking()
        {
            List<Booking> returnThese = new List<Booking>();

            MySqlConnection connection = new MySqlConnection(connectionString);

            0 references
            public List<Booking> searchTitles(String searchTerm)
            {
                List<Booking> returnThese = new List<Booking>();

                MySqlConnection connection = new MySqlConnection(connectionString);
                connection.Open();

                String searchWildPhrase = "%" + searchTerm + "%";
                MySqlCommand command = new MySqlCommand();
                command.CommandText = "SELECT ID, NAME, CHECKIN, CHECKOUT FROM BOOKING WHERE NAME LIKE @search";
                command.Parameters.AddWithValue("@search", searchWildPhrase);
                command.Connection = connection;

                using (MySqlDataReader reader = command.ExecuteReader())
                {
                    while (reader.Read())
                    {
                        Booking a = new Booking
                        {
                            ID = reader.GetInt32(0),
                            NAME = reader.GetString(1),
                            CHECKIN = reader.GetString(2),
                            CHECKOUT = reader.GetString(3),
                        };
                        returnThese.Add(a);
                    }
                }
                connection.Close();
                return returnThese;
            }

            0 references
            internal int addOneBooking(Booking booking)
            {
                MySqlConnection connection = new MySqlConnection(connectionString);
                connection.Open();

                MySqlCommand command = new MySqlCommand("INSERT INTO `booking` ('NAME', 'CHECKIN', 'CHECKOUT') VALUES (@NAME, @CHECKIN, @CHECKOUT)", connection);
            }
        }
    }
}
```



```

        command.Parameters.AddWithValue("@NAME", booking.NAME);
        command.Parameters.AddWithValue("@CHECKIN", booking.CHECKIN);
        command.Parameters.AddWithValue("@CHECKOUT", booking.CHECKOUT);

        int newRows = command.ExecuteNonQuery();

        connection.Close();

        return newRows;
    }

    0 references
    internal int deleteTrack(int trackID)
    {
        MySqlConnection connection = new MySqlConnection(connectionString);
        connection.Open();

        MySqlCommand command = new MySqlCommand("DELETE FROM 'booking' WHERE 'booking'.ID = @ID;", connection);

        command.Parameters.AddWithValue("@ID", trackID);

        int result = command.ExecuteNonQuery();

        connection.Close();

        return result;
    }

    0 references
    internal int UpdateBooking(Booking booking)
    {
        MySqlConnection connection = new MySqlConnection(connectionString);
        connection.Open();

        MySqlCommand command = new MySqlCommand("UPDATE 'booking' SET 'ID' = @ID, 'NAME' = @NAME, 'CHECKIN' = @CHECKIN, 'CHECKOUT' = @CHECKOUT;", connection);

        command.Parameters.AddWithValue("@NAME", booking.NAME);
        command.Parameters.AddWithValue("@CHECKIN", booking.CHECKIN);
        command.Parameters.AddWithValue("@CHECKOUT", booking.CHECKOUT);

        int update = command.ExecuteNonQuery();
    }
}

```

```

1  using System.Net;
2
3  namespace HostelDatabase
4  {
5      3 references
6      public partial class Form1 : Form
7      {
8          BindingSource bookingBindingSource = new BindingSource();
9
10         1 reference
11         internal List<Booking> Booking { get; private set; }
12
13         1 reference
14         public Form1()
15         {
16             InitializeComponent();
17
18         1 reference
19         private void button1_Click(object sender, EventArgs e)
20         {
21             BookingDAO bookingDAO = new BookingDAO();
22
23             bookingBindingSource.DataSource = bookingDAO.getAllBooking();
24             dataGridView1.DataSource = bookingBindingSource;
25         }
26
27         1 reference
28         private void button2_Click(object sender, EventArgs e)
29         {
30             BookingDAO bookingDAO = new BookingDAO();
31
32             bookingBindingSource.DataSource = bookingDAO.searchTitles(textBox1.Text);
33             dataGridView1.DataSource = bookingBindingSource;
34         }
35
36         1 reference
37         private void button3_Click(object sender, EventArgs e)
38         {
39             Booking booking = new Booking
40             {
41                 NAME = txt_NAME.Text,
42                 CHECKIN = txt_CHECKIN.Text,
43                 CHECKOUT = txt_CHECKOUT.Text,
44             };
45             BookingDAO bookingDAO = new BookingDAO();

```

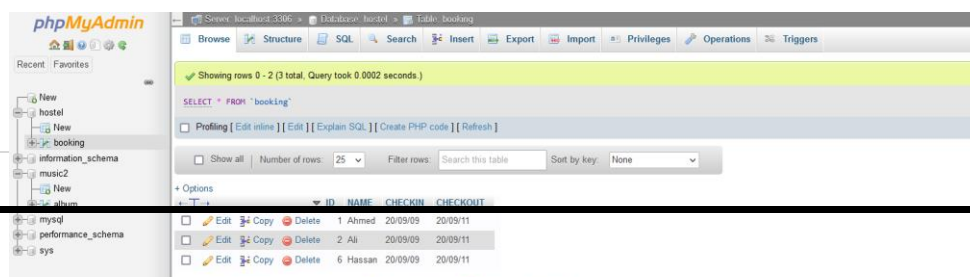
Database Integration with UI

```

39     };
40     BookingDAO bookingDAO = new BookingDAO();
41     int result = bookingDAO.addOneBooking(booking);
42     MessageBox.Show(result + "new rows inserted");
43 }
44
45 1 reference
46 private void label2_Click(object sender, EventArgs e)
47 {
48 }
49
50 1 reference
51 private void button4_Click(object sender, EventArgs e)
52 {
53     int rowClicked = dataGridView1.CurrentRow.Index;
54     int trackID = (int)dataGridView1.Rows[rowClicked].Cells[0].Value;
55     MessageBox.Show("ID of Booking = " + trackID);
56
57     BookingDAO bookingDao = new BookingDAO();
58     int result = bookingDao.deleteTrack(trackID);
59     MessageBox.Show("Result " + result);
60
61     dataGridView1.DataSource = null;
62     Booking = bookingDao.getAllBooking();
63 }
64
65 1 reference
66 private void button5_Click(object sender, EventArgs e)
67 {
68     int rowClicked = dataGridView1.CurrentRow.Index;
69     int trackID = (int)dataGridView1.Rows[rowClicked].Cells[0].Value;
70
71     BookingDAO bookingDao = new BookingDAO();
72     int update = bookingDao.UpdateBooking(booking);
73     MessageBox.Show("Update " + update);
74
75     dataGridView1.DataSource = null;
76     Booking = bookingDao.getAllBooking();
77 }

```

FRONT END



Form1

Load Albums

Edit Album

Add Album

NAME

CHECKIN

CHECKOUT

Add

Delete

Search

Interface

Form1

Load Albums

Edit Album

Add Album

NAME

CHECKIN

CHECKOUT

Add

Delete

ID	NAME	CHECKIN	CHECKOUT
1	Ahmed	20/09/09	20/09/11
2	Ali	20/09/09	20/09/11
4	Haider	20/09/09	20/09/11
*			

Search

Load Database

Form1

Load Albums

Edit Album

Add Album

NAME

CHECKIN

CHECKOUT

Add

Delete

ID	NAME	CHECKIN	CHECKOUT
1	Ahmed	20/09/09	20/09/11
*			

Search

Search Data

Form1

Load Albums

Edit Album

ID	NAME	CHECKIN	CHECKOUT
1	Ahmed	20/09/09	20/09/11
2	Ali	20/09/09	20/09/11
6	Hassan	20/09/09	20/09/11

Add Album

NAME

Hassan

CHECKIN

20/09/09

CHECKOUT

20/09/11

Add

Search

Delete

Delete Data

Form1

Load Albums

Edit Album

ID	NAME	CHECKIN	CHECKOUT
1	Ahmed	20/09/09	20/09/11
2	Ali	20/09/09	20/09/11

Add Album

NAME

Hassan

CHECKIN

20/09/09

CHECKOUT

20/09/11

Add

Search

Delete

Insert Data

Form1

Load Albums

Edit Album

ID	NAME	CHECKIN	CHECKOUT
1	Ahmed	20/09/09	20/09/11
2	Ali	20/09/09	20/09/11
4	Usman	20/09/09	20/09/11

Add Album

NAME

CHECKIN

CHECKOUT

Add

Search

Delete

Update Database

Thank YOU