HOTEL MANAGEMENT SYSTEM
DBMS PROJECT

23CSB0A52-VARDHITHA S

NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL

Table of Contents:

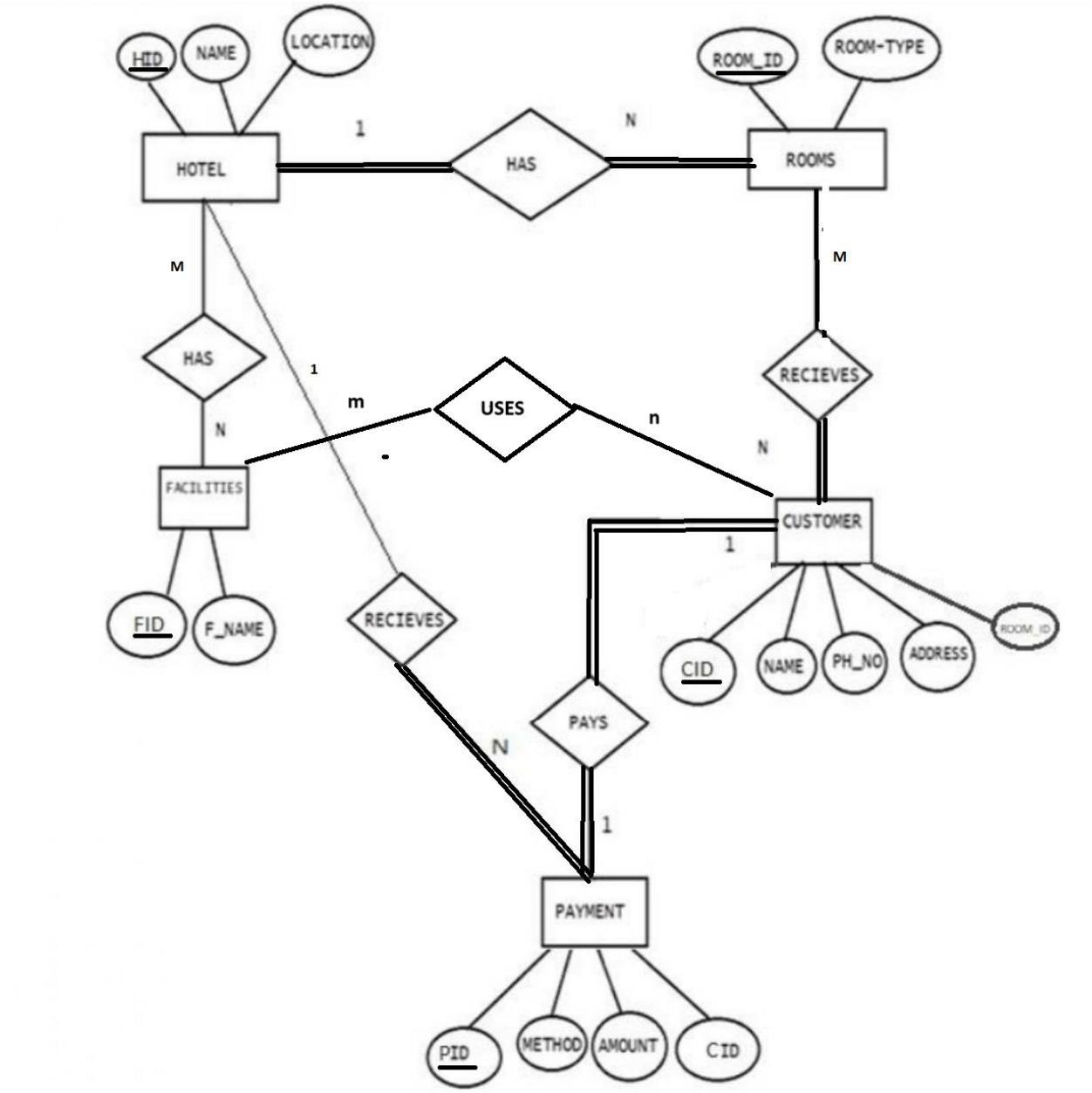
INTRODUCTION OF THE DATABASE SYSTEM:

Hotel management project provides room booking and other hotel management facilities. It enhances our convenience by allowing access to varied resources, all from the comfort of our home.

My project is focused on designing and managing a database system for a hotel , including the handling of rooms, facilities, customers, and payments. The goal is to establish a robust database structure that efficiently stores and retrieves hotel-related information.

Below is the ER Model of my hotel management system project.

ER DIAGRAM


HID
NAME
LOCATION
HOTEL
1
HAS
N
ROOM_ID
ROOM-TYPE
ROOMS
M
M
HAS
1
m
USES
n
N
N
RECIEVES
FACILITIES
CUSTOMER
1
FID
F_NAME
RECIEVES
CID
NAME
PH_NO
ADDRESS
ROOM_ID
PAYS
N
1
PAYMENT
PID
METHOD
AMOUNT
CID

ENTITIES, ATTRIBUTES AND RELATIONSHIPS:

Entities:

An entity is an object that exists. It doesn't have to do anything; it just has to exist. In database administration, an entity can be a single thing, person, place, or object. Data can be stored about such entities. A design tool that allows database administrators to view the relationships between several entities is called the entity relationship diagram.

The following Entities are used in our Database:

1. Hotel
2. Rooms
3. Customer
4. Payment
5. Facilities **Attribute:**

An attribute defines the information about the entity that needs to be stored. If the entity is an employee, attributes could include name, employee ID, health plan enrollment, and work location. An entity will have zero or more attributes, and each of those attributes apply only to that entity.

Relationship:

A relationship, in the context of databases, is a situation that exists between two relational database tables when one table has a foreign key that references the primary key of the other table. Relationships allow relational databases to split and store data in different tables, while linking disparate data items.

Following is a detailed description of every Entity, its Attributes and Relations in between them as employed in our Database:

HOTEL:

This contains information about particular hotel. It contains Hid, Name, Location as its attributes.

One hotel has many rooms and many facilities. Also one facility can be available in many hotels.So hotel maintains one - many relationships with the Rooms and many-many relationships with Facilities. And also one hotel receives multiple payments. So it maintains one-many relationship with payments.

ROOMS:

It contains data about the rooms of the hotel.

It contains RID, Room_type as its attributes.

Here RID is primary key.

It holds many-to-many relationship with Customer entity.

FACILITIES:

This describes about the facilities the hotel contains.

This entities contains attributes as FID,FNAME.

Here FID is primary attribute.

One hotel has many Facilities and one facility can be available in many hotels.So, it maintains many– many relationship.

CUSTOMER:

This gives the information about the customers. This entity contains CID, NAME, ADDRESS, PH_NO, Room_no as attributes.

CID is a primary key and Room_no acts as foreign keys.

As one customer pays one payment. It holds one-to-one relationship.
It holds many-to-many relationship with Rooms entity.

PAYMENT:

It contains data about how the payment is made by the customers.

It has attributes as PID, METHOD, AMOUNT, CID. It maintains one –to-one relationship with customer and many–to-one relationship with hotel.

Here PID is a primary key and CID acts as foreign key.

Process of Normalization:

Normalization:

Normalization is the process of minimizing redundancy from a relation or set of relations. Redundancy in relation may cause insertion, deletion and update-based anomalies. So, it helps to minimize the redundancy in relations.

Normal forms are used to eliminate or reduce redundancy in database tables.

There exist three main types of Normal forms, each being associated with a increasing degree of Normalization:

First Normal Form: A relation is in first normal form if every attribute in that relation is singled valued attribute.

Ex: In our database,

HOTEL Table: **HID, NAME, LOCATION**: Each attribute contains atomic values. Hence are in 1NF

ROOMS Table: **Room_ID, Room_type**: Each attribute contains atomic values

Second Normal Form: A relation is in second normal form if it is in 1NF and has no Partial Dependency, i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

EX: **HOTEL Table**

- **Attributes**: HID (Primary Key), NAME, LOCATION
- **Primary Key**: HID
- **Non-key Attributes**: NAME, LOCATION

There are no partial dependencies because all non-key attributes (NAME, LOCATION) are fully functionally dependent on the primary key (HID).

Ex: CUSTOMERS Table

- **Attributes**: CID (Primary Key), NAME, ADDRESS, PH_NO, Room_ID
- **Primary Key**: CID
- **Non-key Attributes**: NAME, ADDRESS, PH_NO, Room_ID

There are no partial dependencies because all non-key attributes (NAME, ADDRESS, PH_NO, Room_ID) are fully functionally dependent on the primary key (CID).

Third Normal Form: A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form.

In the entire database, there exist no attributes which exhibit any feature of redundancy. Furthermore, each attribute satisfies all the above-mentioned Normal forms, thereby eliminating the need for any further Normalization. **Ex: HOTEL Table**

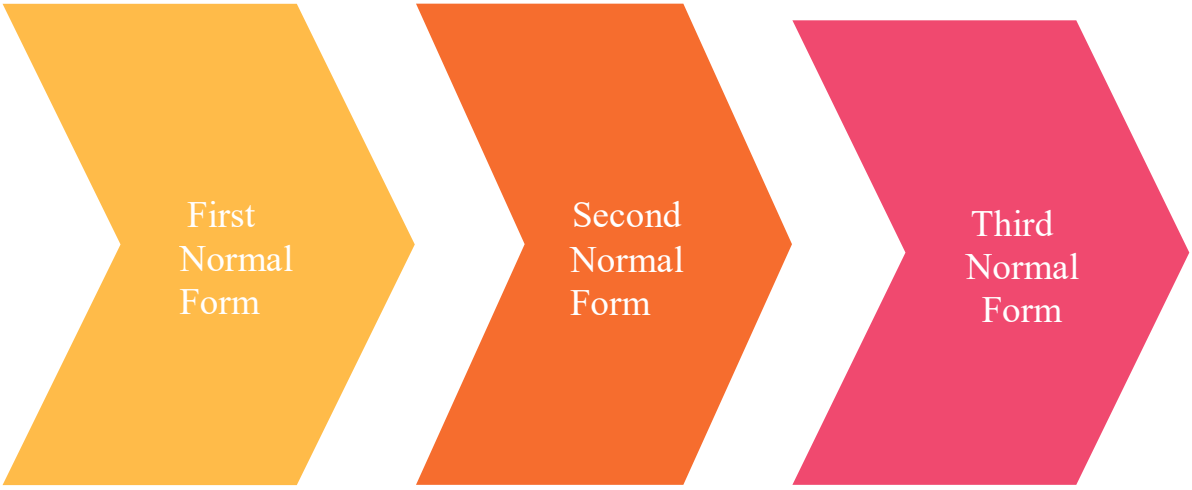
- **Attributes**: HID (Primary Key), NAME, LOCATION
- **Primary Key**: HID
- **Non-key Attributes**: NAME, LOCATION

There are no transitive dependencies because non-key attributes (NAME, LOCATION) depend only on the primary key (HID).

Thereby, we can move on to the actual representation of the Database schema. We achieved the first normal form by keeping the data scalar.

Coming to the second normal form,we made sure that there are no partial dependencies left .

On the third normal form, we made sure that there are no transitive dependencies present.



DATA DEFINITION LANGUAGE:

In total, there exist five tables in our database design. Each table and its associated DDL commands have been listed below:

CREATION:

1)Hotel Table:

```
CREATE TABLE HOTEL
(
HID int not null PRIMARY KEY,
NAME varchar(20),
LOCATION varchar(20)
);
```

2)ROOMS TABLE:

```
CREATE TABLE ROOMS
(
ROOM_ID int not null PRIMARY KEY,
ROOM_TYPE varchar(20)
);
```

3)CUSTOMER TABLE:

```
CREATE TABLE CUSTOMERS
(
CID int not null PRIMARY KEY,
NAME    varchar(20),
ADDRESS varchar(20),
PH_NO int not null, ROOM_ID
int not
null
);
```

Alter table CUSTOMERS Add constraint fk_1 foreign key(ROOM_ID) references ROOMS(ROOM_ID);

4)FACILITIES TABLE:

CREATE TABLE FACILITIES

(

FID int not null PRIMARY KEY,

F_NAME varchar(20)

);

5)PAYMENT TABLE:

CREATE TABLE PAYMENT

(

PID int not null PRIMARY KEY,

METHOD varchar(20),

AMOUNT int

CID int

);

Alter table PAYMENTSS add constraint fk_2 foreign key(CID) references CUSTOMERSS(CID);

INSERTING VALUES:

1) HOTEL TABLE:

INSERT INTO HOTEL VALUES (1, 'TAJ KRISHNA', 'BANJARA HILLS'); **2)**

ROOMS TABLE:

INSERT INTO ROOMS VALUES (1, 'SINGLE BEDROOM');

INSERT INTO ROOMS VALUES (2, 'DOUBLE BEDROOM');

INSERT INTO ROOMS VALUES (3, 'TRIPLE BEDROOM'); **3)**

CUSTOMER TABLE:

INSERT INTO CUSTOMERS VALUES (1,'MARY','KOCHI',12345,1,1);

INSERT INTO CUSTOMERS VALUES (2,'RADHA','HYD',6789,1,3);

INSERT INTO CUSTOMERS VALUES (3,'KRISH','WARANGAL',98989,2,2);

INSERT INTO CUSTOMERS VALUES (4,'SITA','HYD',23232,3,3);

INSERT INTO CUSTOMERS VALUES (5,’ALEX’,'KOCHI',12122,3,3);

INSERT INTO CUSTOMERS VALUES (6,'JOHN','HYD',23122,2,2);

4)FACILITIES TABLE:

INSERT INTO FACILITIES VALUES(1,'SPA');

INSERT INTO FACILITIES VALUES(2,'GYM');

INSERT INTO FACILITIES VALUES(3,'POOL');

5)PAYMENTS TABLE:

INSERT INTO PAYMENTS VALUES(1,'CASH',10000,3);

INSERT INTO PAYMENTS VALUES(2,'UP1',20000,1);

INSERT INTO PAYMENTS VALUES(3,'CASH',15000,4);

INSERT INTO PAYMENTS VALUES(4,'CASH',10000,5); INSERT

INTO PAYMENTS VALUES(5,'CARD',15000,2);

INSERT INTO PAYMENTS VALUES(6,'CARD',10000,6);

SAMPLE TABLE OVERVIEWS:

1. Information in HOTEL table

SELECT * FROM HOTELS;

HID	NAME	LOCATION
1	TAJ KRISHNA	BANJARA HIL

2. **Information in ROOMS table**

SELECT * FROM ROOMS;

ROOM_ID	ROOM_TYPE
1	SINGLE BEDROOM
2	DOUBLE BEDROOM
3	TRIPLE BEDROOM

3.Information in CUSTOMERS TABLE
SELECT * FROM CUSTOMERS;

CID	NAME	ADDRESS	PH_NO	ROOM_NO
1	MARY	KOCHI	12345	1
2	RADHA	HYD	6789	1
3	KRISH	WARANGAL	98989	2
4	SITA	HYD	23232	3
5	ALEX	KOCHI	12122	3

4. Information in FACILITIES table

SELECT * FROM FACILITIES;

FID	F_NAME
1	SPA
2	GYM
3	POOL

5. Information in PAYMENTS table

SELECT * FROM PAYMENTS;

PID	METHOD	AMOUNT	CID
1	CASH	10000	3
2	UPI	20000	1
3	CASH	15000	4
4	CASH	10000	5
5	CARD	15000	2
6	CARD	10000	6

SAMPLE QUERIES:

1. **List customers who paid using 'CASH'**

```
SELECT C.NAME FROM CUSTOMERS C
JOIN PAYMENTS P ON C.CID = P.CID WHERE P.METHOD = 'CASH';
```

NAME
KRISH
SITA
ALEX

2. **List the facility names having the character set 'oo' together**

```
SELECT FNAME FROM FACILITIES WHERE Facility_name LIKE '%OO%';
```

F_NAME
POOL

3. **Apply UNION operation on CUSTOMERS and ROOMS**

```
SELECT ROOM_ID FROM CUSTOMERS UNION SELECT ROOM_ID
FROM ROOMS;
```

ROOM_ID
1
2
3

4. Select customers where address is 'HYD'

SELECT * FROM CUSTOMERS WHERE ADDRESS = 'HYD';

CID	NAME	ADDRESS	PH_NO	ROOM_NO	FID
2	RADHA	HYD	6789	1	3
4	SITA	HYD	23232	3	3
6	JOHN	HYD	23122	2	2

5. Perform an inner join between CUSTOMERS and ROOMS

SELECT * FROM CUSTOMERS C INNER JOIN ROOMS R ON R.ROOM_ID = C.ROOM_ID;

CID	NAME	ADDRESS	PH_NO	ROOM_NO	FID	ROOM_TYPE
1	MARY	KOCHI	12345	1	1	SINGLE BEDROOM
2	RADHA	HYD	6789	1	3	SINGLE BEDROOM
3	KRISH	WARANGAL	98989	2	2	DOUBLE BEDROOM
4	SITA	HYD	23232	3	3	TRIPLE BEDROOM
5	ALEX	KOCHI	12122	3	3	TRIPLE BEDROOM
6	JOHN	HYD	23122	2	2	DOUBLE BEDROOM

6. Get all customer names and the type of room they booked

SELECT C.NAME AS CUSTOMER_NAME, R.ROOM_TYPE
FROM CUSTOMERS C JOIN ROOMS R ON C.ROOM_ID = R.ROOM_ID;

CUSTOMER_NAME	ROOM_TYPE
MARY	SINGLE BEDROOM
RADHA	SINGLE BEDROOM
KRISH	DOUBLE BEDROOM

SITA	TRIPLE BEDROOM
ALEX	TRIPLE BEDROOM
JOHN	DOUBLE BEDROOM

7. Total amount paid by all customers

```
SELECT SUM(AMOUNT) AS TOTAL_REVENUE FROM PAYMENTS;
```

TOTAL_REVENUE
80000

8. Show all room types along with the number of customers booked in each

```
SELECT R.ROOM_TYPE, COUNT(*) AS BOOKED_CUSTOMERS
FROM CUSTOMERS C JOIN ROOMS R ON C.ROOM_ID = R.ROOM_ID
GROUP BY R.ROOM_TYPE;
```

ROOM_TYPE	BOOKED_CUSTOMERS
SINGLE BEDROOM	2
DOUBLE BEDROOM	2
TRIPLE BEDROOM	2

9. List all customers and their payment details

```
SELECT C.NAME, P.METHOD, P.AMOUNT
FROM CUSTOMERS C JOIN PAYMENTS P ON C.CID = P.CID;
```

NAME	METHOD	AMOUNT
MARY	UPI	20000
RADHA	CARD	15000
KRISH	CASH	10000

SITA	CASH	15000
ALEX	CASH	10000
JOHN	CARD	10000

10. Show the most expensive payment

```
SELECT * FROM PAYMENTS
ORDER BY AMOUNT DESC
LIMIT 1;
```

PID	METHOD	AMOUNT	CID
2	UPI	20000	1

11. Show each payment method and amount collected through it

```
SELECT METHOD, SUM(AMOUNT) AS TOTAL
FROM PAYMENTS GROUP BY METHOD;
```

METHOD	TOTAL
CASH	35000
UPI	20000
CARD	25000

12. Display the name and payment amount of customers who paid more than ₹10,000

```
SELECT C.NAME, P.AMOUNT
FROM CUSTOMERS C JOIN PAYMENTS P ON C.CID = P.CID
WHERE P.AMOUNT > 10000;
```

NAME	AMOUNT
MARY	20000
SITA	15000
RADHA	15000

CONCLUSION:

To summarize and conclude, in this project, I created an Online Hotel Management System for users to seamlessly interact with hotels and also to book the rooms, to know further facilities and payment methods.

Throughout the project, I have made use of concepts I learnt , including DDL commands ,Primary and Foreign Keys etc.I built the database from the ground up, starting from the basic ER – Diagram, all the way to acomplete, well defined, well structured database schema. Furthermore, to practically show the usage of the database, number of sample queries for each and every entity based tables have been provided.