**Project Report**

**CS 6360.002 - Database Design**



**Hotels.com** Database Design

Table of Contents

[**1.** **Project Description** 2](#_Toc7599837)

[**2.** **Data Requirements:** 2](#_Toc7599838)

[**2.1 Entity Description** 2](#_Toc7599839)

[**2.2 Assumptions** 7](#_Toc7599840)

[**3.** **EER DIAGRAM** 8](#_Toc7599841)

[4. **Mapping of ERD in Relational Diagram** 9](#_Toc7599842)

[**5.** **Normalization** 10](#_Toc7599843)

[**6.** **Final Relational Schema:** 11](#_Toc7599844)

[**7.** **SQL QUERIES** 12](#_Toc7599845)

[**8.1 SQL Statements to CREATE the above Relations:** 12](#_Toc7599846)

[**8.2 SQL statements to INSERT into above relations:** 18](#_Toc7599847)

[**8.** **TRIGGERS:** 22](#_Toc7599848)

[**9.** **Procedures** 24](#_Toc7599849)

# **Project Description**

This project aims at creating a relational database for the hotels.com website.

It involves customers to browse through the different hotels and types of room based on their requirements. It allows customers to book rooms or event halls.

# **Data Requirements:**

## **2.1 Entity Description**

The project has the following entities:

* Accommodation
* Brand
* Room
* Booking
* Hall Reservation
* Deal
* Room Type
* Amenities
* Accommodation Type
* Features
* Event Halls
* Location
* Customer
* Guest
* Registered User
* Review
* Payment

**Accommodation**

This entity represents the accommodations available on the hotels.com website.

**For example:** Double Tree, Embassy Suites, Tru

An Accommodation has the following attributes:

* Accommodation ID
* Number of rooms
* Phone
* Star Rating
* Description

It further has the following represented through relationships:

* Features
* Rooms
* Deals available on it
* An associated Brand
* Associated bookings

**Brand**

A brand may include multiple accommodations associated with it.

**For example:** Hilton, Marriott, Four Seasons

A brand has the following attributes:

* Brand ID
* Brand Name

**Room**

This entity represents the different rooms of different accommodations on the hotels.com website.

A Room has the following attributes:

* Room ID
* Room No.
* Base Price
* Location
* Capacity
* Current Status
* Active To
* Active From

It further has the following represented through relationships:

* Room Type
* Associated Accommodation
* Associated Bookings

**Booking**

This entity represents all the bookings made on the hotels.com website.

A Booking has the following attributes:

* Booking ID
* Booking Date
* Number of Adults
* Number of Children
* Check in date
* Check out date

It further has the following represented through relationships:

* Rooms involved
* Associated Payment
* Customer that initiated the booking

**Event Halls**

This entity represents the event halls that are available for reservation in different hotels on the hotels.com website.

**For example:** Conference halls, Wedding halls

An Event Hall has the following attributes:

* Hall ID
* Hall Name
* Purpose
* Capacity
* Group Type
* Price

It further has the following represented through relationships:

* Associated Hotel
* Associated Hall Reservation

**Hall Reservation**

This entity represents the reservations made on event halls available on the hotels.com website.

A hall reservation has the following attributes:

* Reservation ID
* Reservation Date
* Number of People
* Check in date
* Check out date

It further has the following represented through relationships:

* Involved event hall
* Associated Payment
* Associated Customer

**Room Type**

This entity represents the different room types that categorize the rooms in an accommodation on the hotels.com website.

For example: Deluxe room, Supreme room, Basic room

An Room Type has the following attributes:

* Room Type ID
* Room Type Name
* Meal
* Tax

It further has the following represented through relationships:

* Rooms categorized by the room type
* Included Amenities

**Amenities**

This entity represents the amenities that can be included in a room on the hotels.com website.

For example: Microwave, Refrigerator, Sofa-bed

An amenity has the following attributes:

* Amenity ID
* Amenity Name

It further has the following represented through a relationship:

* The room types they are available in.

**Accommodation Type**

This entity represents the different accommodation types that the hotels.com website offers.

**For example:** Hotel, Resort, Apartment, Motel, Hostel

An Accommodation Type has the following attributes:

* Accommodation ID
* Accommodation Name

It further has the following represented through a relationships

* Hotels categorized under a particular accommodation type

**Features**

This entity represents the different features that the hotels on the hotels.com website offer.

**For example:** Gym, Swimming pool, Wi-fi, Restaurant, Childcare, Airport transfers

A feature has the following attributes:

* Feature ID
* Feature Name

It further has the following represented through a relationship:

* The accommodations they are offered in

**Deal**

This entity represents the deals that are available on accommodations on the hotels.com website.

An Deal has the following attributes:

* Deal ID
* Discount
* Promo Code
* Validity Start Date
* Validity End Date

It further has the following represented through a relationship:

* Accommodations on which the deal is available

**Payment**

This entity represents the all the payments made on the hotels.com website including both Room bookings and Event Hall Bookings.

A Payment has the following attributes:

* Payment ID
* Card Type
* Card Number
* Amount

It further has the following represented through relationships:

* Associated Bookings
* Associated Hall Reservations

**Location**

This entity represents the locations at which different accommodations are present.

A Location has the following attributes:

* Location ID
* Country
* State
* City
* Zip
* Street Name
* Street Number

It further has the following represented through a relationship:

* Accommodation present at that location

**Customer**

This entity is further categorized/specialized into two entities – Registered User and Guest.

A registered user has an account with a profile on hotels.com. A registered user is allowed to write reviews, give ratings and earn reward points.

A customer has the following attributes:

* Customer ID
* First name
* Last name
* Middle name initial
* Phone No.

**Guest:**

* Guest ID

**Registered User:**

* Registered User ID
* Card No.
* Address
* User ID
* Password

It further has the following represented through relationships:

* Bookings made by the customer
* Reviews given by the Registered User
* Hall Reservations made by the customer

**Review**

This entity represents the reviews given by registered users on different accommodations on the hotels.com website.

A Review has the following attributes:

* Review ID
* Feedback
* User Rating

It further has the following represented through relationships:

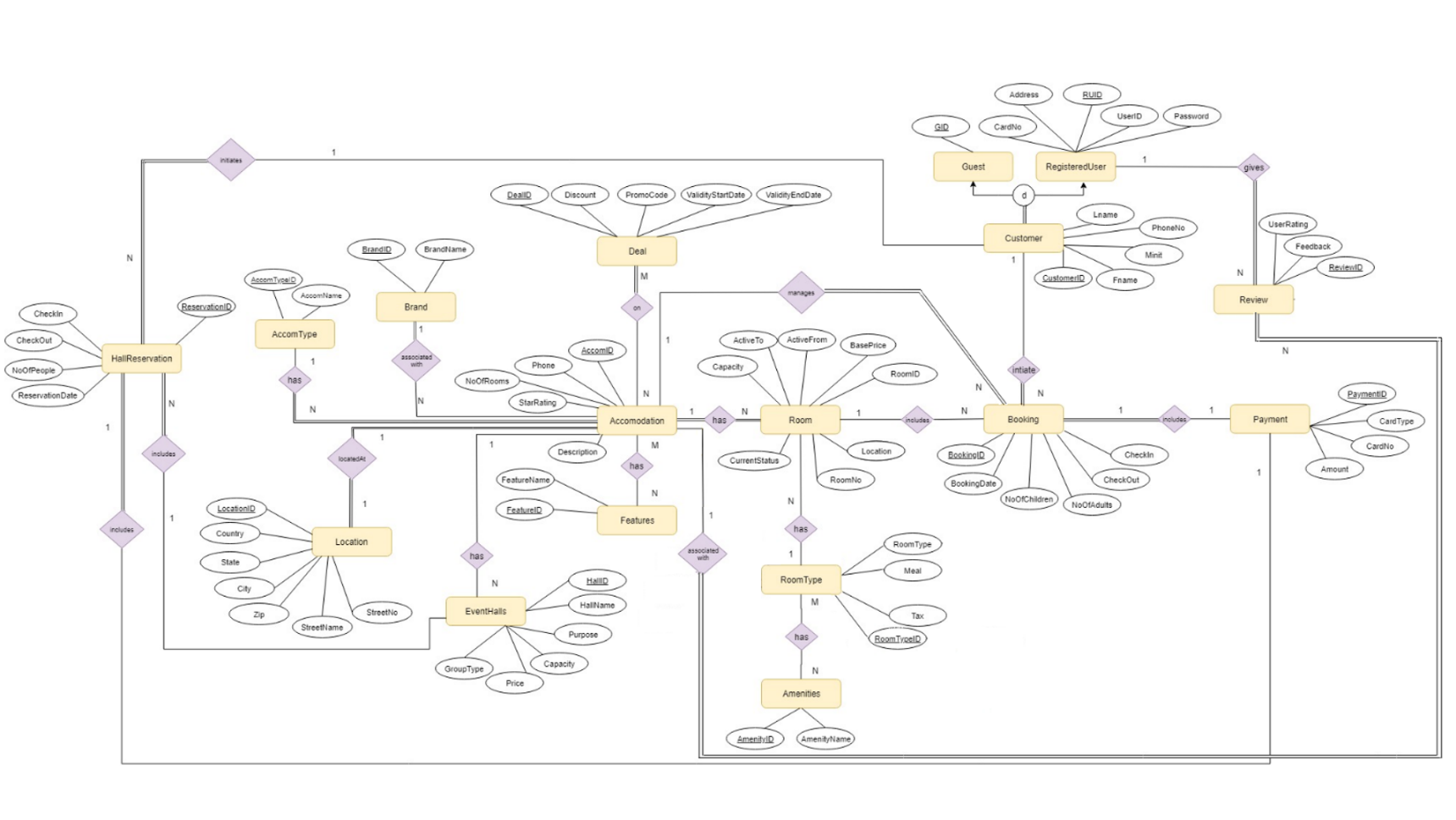
* Associated Registered users
* Accommodation associated with a review

## **2.2 Assumptions**

Our design assumes the following:

* A customer can only book one room at a time
* A customer can either be a guest or a registered user, but not both.
* A booking can be made without registering on the website (guest booking)
* A booking must be initiated by a customer
* Only registered users can give reviews
* An accommodation may not necessarily be associated with a brand but a brand must have some accommodations
* Booking and Hall Reservation are associated with the same Payment entity
* A deal must be associated with an accommodation
* An accommodation must have an accommodation type
* A booking must be managed by a hotel
* An accommodation must have a location and a location must be associated with some accommodation.

# **EER DIAGRAM**



# **Cardinalities**

Few examples:

**One-to-One Relationship**

* Booking -> Payment

(A booking has only one payment associated with it and a payment is only associated with one booking)

**One-to-Many and Many-to-One Relationship**

* Customer -> Booking

(A customer may initiate multiple bookings but a single booking can be initiated by only one customer)

* Brand -> Accommodation

(A brand can have many accommodation under its name but an accommodation can only be associated with one brand)

* Accommodation Type -> Accommodation

(There can be many accommodations under one type category but an accommodation can only be of type)

* Review -> Accommodation

(A review can be associated with only one accommodation whereas an accommodation may have many reviews)

**Many-to-Many Relationship**

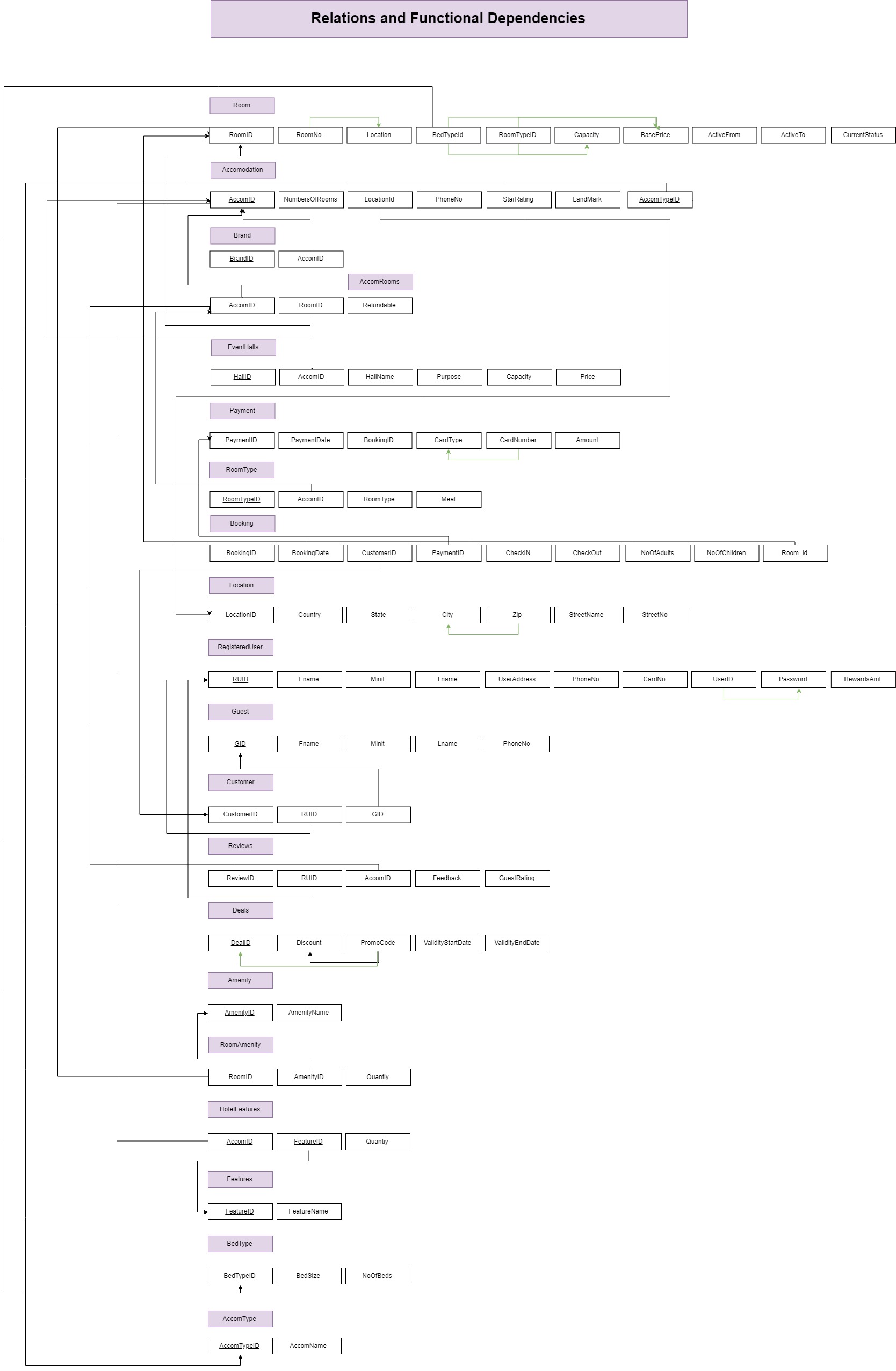
* Accommodation -> Features

(An accommodation can have multiple features and a feature can be available in many accommodations)

* RoomType -> Amenities

(A room type can many amenities available in it and an amenity can be available in multiple room types)

# **Mapping of ER Diagram in Relational Diagram**



# **Normalization (#examples)**

**1NF Definition:** A relation is said to be first normal form if the domain of each attribute contains only atomic values, and the value of each attribute contains only a single value from that domain.

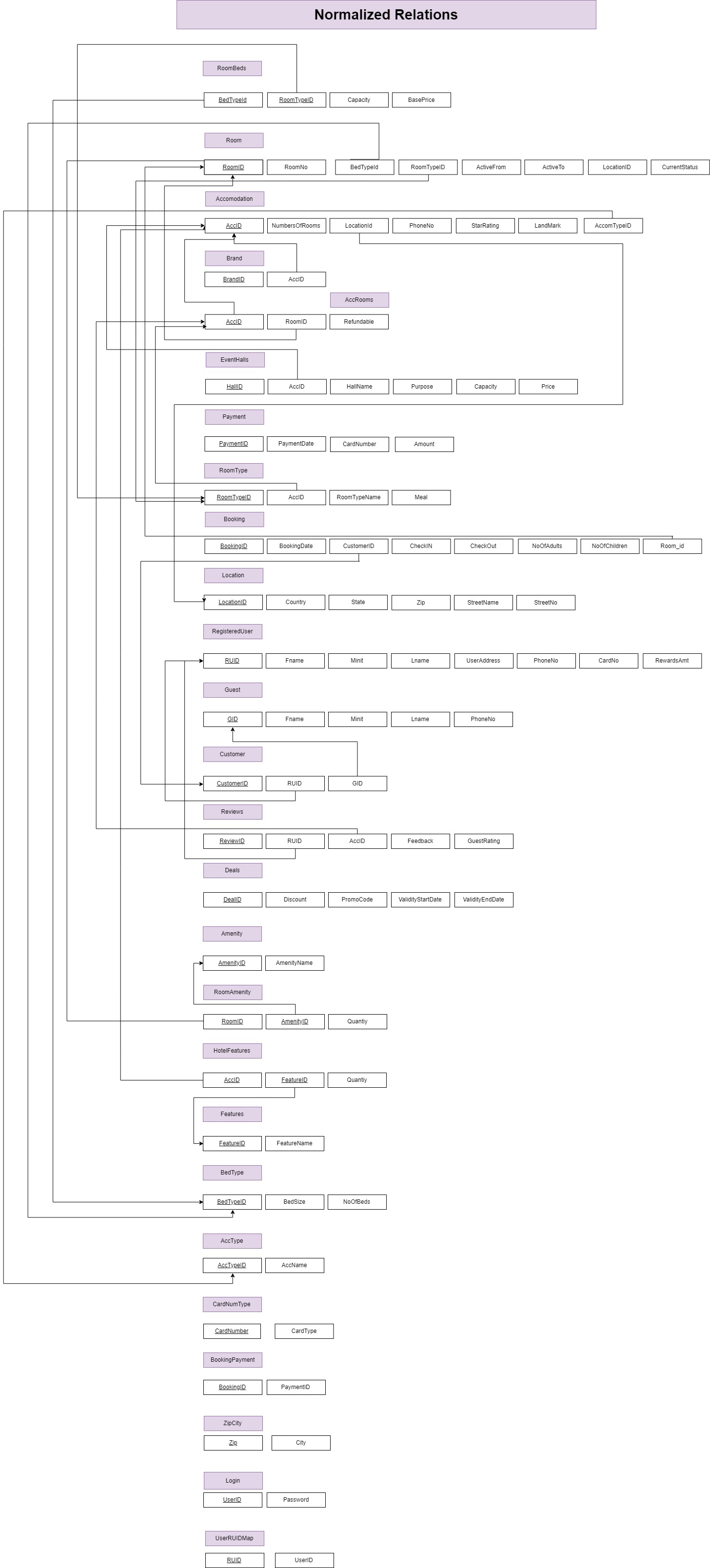
In our database tables, all attributes consist of only atomic values. Hence, all the tables satisfy the 1NF definition.

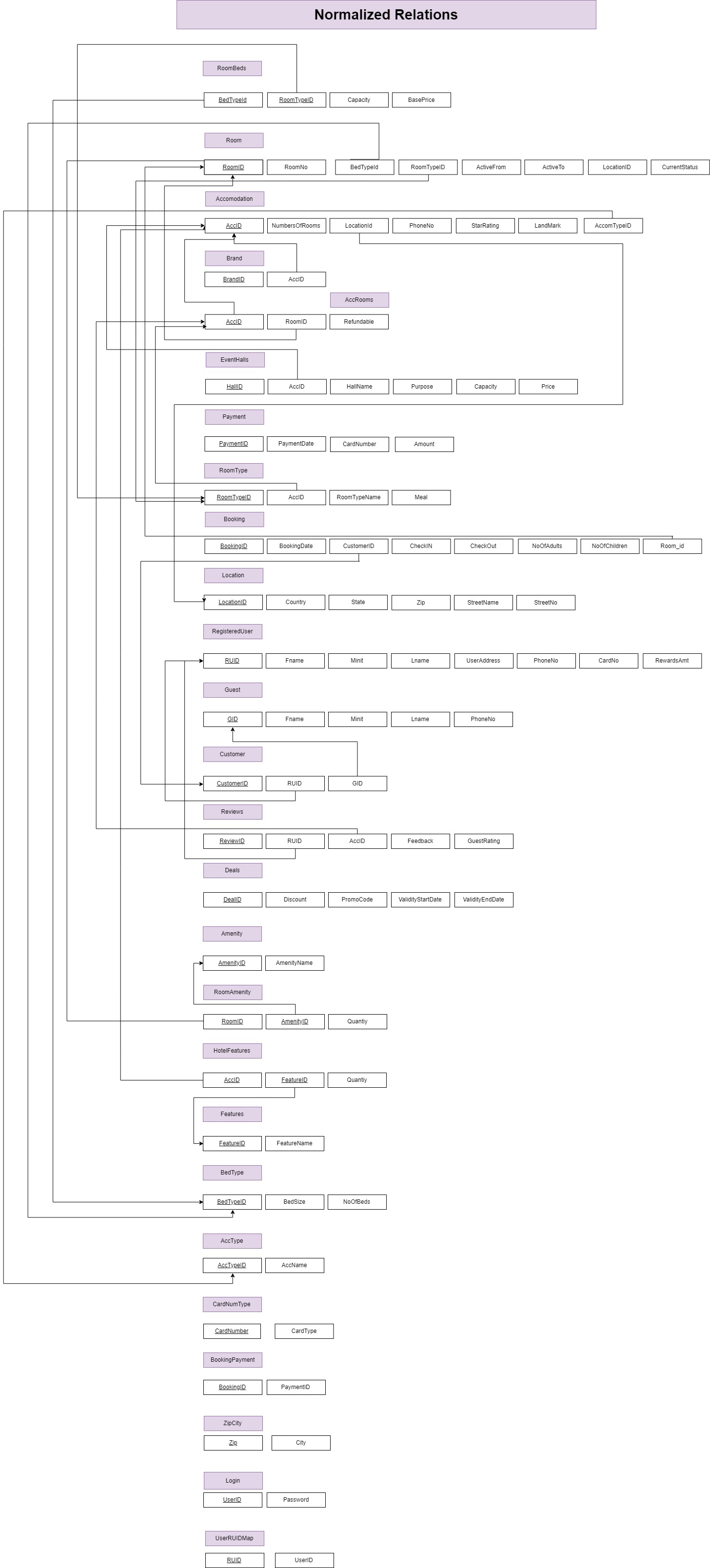
**2NF Definition:** A relation is said to be in second normal form if it is in 1NF and no non-prime attribute is dependent on any proper subset of any candidate key of the table.

All the attributes in our database tables depend on the primary key alone. Hence, they satisfy the 2NF definition.

**3NF Definition:** A relation is said to be in third normal when attributes are determined only by the candidate keys of that table and not by any non-prime attributes. All tables in our database are in 3NF.

# **Final Relational Schema:**





# **SQL QUERIES**

## **8.1 SQL Statements to CREATE the above Relations:**

**Location Table:**

create table Location(

LocationID INT,

Country VARCHAR(20),

State VARCHAR(20),

Zip INT,

StreetName VARCHAR(50),

StreetNo INT,

constraint pk\_Location primary key(LocationID)

);

**Accommodation Type Table:**

create table AccType(

AccTypeID INT,

AccName VARCHAR(50),

constraint pk\_AccomType primary key(AccTypeID)

);

**Accommodation Table:**

create table Accomodation (

AccID INT,

NoOfRooms int,

LocationID INT,

PhoneNumber int,

StarRating int,

LandMark VARCHAR(20),

AccTypeID INT,

Description VARCHAR(250),

constraint pk\_accomodation primary key(AccId),

constraint fk\_Accomodation\_1 FOREIGN KEY (LocationID) REFERENCES Location(LocationID) ON DELETE CASCADE,

constraint fk\_Accomodation\_2 FOREIGN KEY (AccTypeId) REFERENCES AccType(AccTypeId) ON DELETE CASCADE

);

**Bed Type Table:**

create table BedType(

BedTypeID INT,

BedSize VARCHAR(10),

NoOfBeds int,

constraint pk\_BedType primary key(BedTypeID)

);

**Room Type Table:**

create table RoomType (

RoomTypeID INT,

AccID INT,

RoomTypeName VARCHAR(10),

Meal VARCHAR(20),

constraint pk\_RoomType primary key(RoomTypeID),

constraint fk\_RoomType FOREIGN KEY (AccID) REFERENCES Accomodation(AccID)ON DELETE CASCADE

);

**Room Beds Table:**

create table RoomBeds (

BedTypeId INT,

RoomTypeID INT,

Capacity int,

BasePrice FLOAT,

constraint pk\_RoomBeds primary key(BedTypeId,RoomTypeID),

constraint fk\_RoomBeds\_1 FOREIGN KEY (BedTypeId) REFERENCES BedType(BedTypeId) ON DELETE SET NULL,

constraint fk\_RoomBeds\_2 FOREIGN KEY (RoomTypeID) REFERENCES RoomType(RoomTypeID) ON DELETE SET NULL

);

**Amenity Table:**

create table Amenity(

AmenityID INT,

AmenityName VARCHAR(20),

constraint pk\_Amenity primary key(AmenityID)

);

**Room Table:**

create table Room (

RoomID INT,

RoomNo VARCHAR(5),

BedTypeId INT,

RoomTypeID INT,

GREENFrom DATE,

GREENTo DATE,

CurrentStatus VARCHAR2(10) CHECK( CurrentStatus IN ('GREEN','RED','YELLOW')),

constraint pk\_Room primary key(RoomID),

constraint fk\_Room\_1 FOREIGN KEY (BedTypeId) REFERENCES BedType(BedTypeId),

constraint fk\_Room\_2 FOREIGN KEY (RoomTypeId) REFERENCES RoomType(RoomTypeId)

);

**Room Amenity Table:**

create table RoomAmenity(

RoomID INT,

AmenityID INT,

Quantiy int,

constraint pk\_RoomAmenity primary key(RoomID,AmenityID),

constraint fk\_RoomAmenity\_1 FOREIGN KEY (AmenityID) REFERENCES Amenity(AmenityID) ON DELETE CASCADE,

constraint fk\_RoomAmenity\_2 FOREIGN KEY (RoomID) REFERENCES Room(RoomID) ON DELETE CASCADE

);

**Brand Table:**

create table Brand (

BrandID INT,

AccID INT,

constraint pk\_brand primary key(BrandID),

constraint fk\_Brand FOREIGN KEY (AccID) REFERENCES Accomodation(AccID) ON DELETE CASCADE

);

**Accommodation-Rooms Table:**

create table AccRooms (

AccID INT,

RoomID INT,

Refundable int,

constraint pk\_accRooms primary key(AccID),

constraint fk\_AccRooms\_1 FOREIGN KEY (AccID) REFERENCES Accomodation(AccID) ON DELETE CASCADE,

constraint fk\_AccRooms\_2 FOREIGN KEY (RoomID) REFERENCES Room(RoomID) ON DELETE CASCADE

);

**Event Halls Table:**

create table EventHalls (

HallID int,

AccID INT,

HallName VARCHAR(10),

Purpose VARCHAR(20),

Capacity int,

Price FLOAT,

constraint pk\_EventHalls primary key(HallID),

constraint fk\_EventHalls FOREIGN KEY (AccID) REFERENCES Accomodation(AccID) ON DELETE CASCADE

);

**Payment Table:**

create table Payment (

PaymentID INT,

PaymentDate DATE,

CardNumber INT,

Amount NUMERIC,

constraint pk\_Payment primary key(PaymentID)

);

**Card Number-Type Table:**

create table CardNumType (

CardNumber INT,

CardType VARCHAR(5),

constraint pk\_CardNumType primary key(CardNumber)

);

**Registered User Table:**

create table RegisteredUser(

RUID VARCHAR(5),

Fname VARCHAR(10),

Minit VARCHAR(10),

Lname VARCHAR(10),

UserAddress VARCHAR(10),

PhoneNo NUMERIC,

CardNo VARCHAR(15),

RewardsAmt int,

constraint pk\_RegisteredUser primary key(RUID)

);

**Guest Table:**

create table Guest(

GID INT,

Fname VARCHAR(15),

Lname VARCHAR(15),

Minit VARCHAR(5),

PhoneNo INT,

constraint pk\_Guest primary key(GID)

);

**Customer Table:**

create table Customer(

CustomerID VARCHAR(5),

RUID VARCHAR(5),

GID INT,

constraint pk\_Customer primary key(CustomerID),

constraint fk\_Customer\_1 FOREIGN KEY (RUID) REFERENCES RegisteredUser(RUID)ON DELETE CASCADE,

constraint fk\_Customer\_2 FOREIGN KEY (GID) REFERENCES Guest(GID)ON DELETE CASCADE

);

**Booking Table:**

create table Booking (

BookingID VARCHAR(8),

BookingDate DATE,

CustomerID VARCHAR(5),

CheckIN DATE,

CheckOut DATE,

NoOfAdults int,

NoOfChildren int,

RoomId INT,

constraint pk\_Booking primary key(BookingID),

constraint fk\_Booking\_1 FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),

constraint fk\_Booking\_2 FOREIGN KEY (RoomId) REFERENCES Room(RoomId)

);

**Booking Payment Table:**

create table BookingPayment (

BookingID VARCHAR(8),

PaymentID VARCHAR(5),

constraint pk\_BookingPayment primary key(BookingID)

);

**Zip-City Table:**

create table ZipCity(

Zip INT,

City VARCHAR(10),

constraint pk\_ZipCity primary key(Zip)

);

create table Login(

UserID VARCHAR(5),

Password VARCHAR(15),

constraint pk\_Login primary key(UserID)

);

**User to Registered User Mapping Table:**

create table UserRUIDMap(

RUID VARCHAR(5),

UserID VARCHAR(5),

constraint pk\_UserRUIDMap primary key(RUID),

constraint fk\_UserRUIDMap\_1 FOREIGN KEY (UserID) REFERENCES Login(UserID) ON DELETE CASCADE,

constraint fk\_UserRUIDMap\_2 FOREIGN KEY (RUID) REFERENCES RegisteredUser(RUID) ON DELETE CASCADE

);

**Reviews Table:**

create table Reviews(

ReviewID VARCHAR(5),

RUID VARCHAR(5),

AccID INT,

Feedback VARCHAR(50),

GuestRating INT,

constraint pk\_Reviews primary key(ReviewID),

constraint fk\_Reviews\_1 FOREIGN KEY (RUID) REFERENCES RegisteredUser(RUID),

constraint fk\_Reviews\_2 FOREIGN KEY (AccID) REFERENCES AccRooms(AccID) ON DELETE CASCADE

);

**Deals Table:**

create table Deals(

DealID VARCHAR(5),

Discount NUMERIC,

PromoCode VARCHAR(5),

ValidityStartDate DATE,

ValidityEndDate DATE,

constraint pk\_Deals primary key(DealID)

);

**Features Table:**

create table Features(

FeatureID VARCHAR(5),

FeatureName VARCHAR(20),

constraint pk\_Features primary key(FeatureID)

);

**Hotel Features Table:**

create table HotelFeatures(

AccID INT,

FeatureID VARCHAR(5),

Quantiy int,

constraint pk\_HotelFeatures primary key(AccID,FeatureID),

constraint fk\_HotelFeatures\_1 FOREIGN KEY (AccID) REFERENCES Accomodation(AccID) ON DELETE CASCADE,

constraint fk\_HotelFeatures\_2 FOREIGN KEY (FeatureID) REFERENCES Features(FeatureID) ON DELETE CASCADE

);

## **8.2 SQL statements to INSERT into above relations:**

**Location Table:**

INSERT INTO Location VALUES (1, 'India', 'Delhi',110018,'Bates Rd','196');

INSERT INTO Location VALUES (2, 'India', 'Mumbai',400004,'Andheri Versova Road','83');

INSERT INTO Location VALUES (3, 'India', 'Bangalore',560004,'Avenue Street','23');

INSERT INTO Location VALUES (4, 'US', 'Texas',75252,'Barnes','001');

INSERT INTO Location VALUES (5, 'US', 'New York',10005,'Broadway','152');

INSERT INTO Location VALUES (6, 'US', 'California',90002,'Adams Boulevard','382');

INSERT INTO Location VALUES (7, 'China', 'Shanghai',200010,'Dongmen Road','495');

INSERT INTO Location VALUES (8, 'China', 'Tianjin ',300041,'Wukang Road','105');

INSERT INTO Location VALUES (9, 'China', 'Beijing ',100015,'Yan an Road','110');

**Accommodation Type Table:**

INSERT INTO AccType VALUES (0,'Apart-hotel');

INSERT INTO AccType VALUES (1,'Apartment');

INSERT INTO AccType VALUES (2,'Bed and Breakfast');

INSERT INTO AccType VALUES (3,'Guest House');

INSERT INTO AccType VALUES (4,'Hostel');

INSERT INTO AccType VALUES (5,'Hotel');

INSERT INTO AccType VALUES (6,'Inn');

INSERT INTO AccType VALUES (7,'Motel');

INSERT INTO AccType VALUES (8,'Resort');

INSERT INTO AccType VALUES (9,'Vacation home / Condo');

INSERT INTO AccType VALUES (10,'Villa');

**Accommodation Table:**

INSERT INTO Accomodation VALUES (001,2,1,98765354,2,'Spring Valey Road',1,'Vans Apartment');

INSERT INTO Accomodation VALUES (002,4,1,98345354,2,'happ Road',1,'Ny Apartments');

INSERT INTO Accomodation VALUES (003,30,1,54765354,3,'Sp Road',5,'Grea Hotel');

INSERT INTO Accomodation VALUES (004,40,1,676765354,4,'Hemple temple Road',5,'Richardson Hotel');

INSERT INTO Accomodation VALUES (005,80,1,87865354,5,'Wang way',5,'SuperSource Hotel');

INSERT INTO Accomodation VALUES (006,20,1,985655354,4,'Super highway',5,'Britian Hotel');

INSERT INTO Accomodation VALUES (007,200,1,98325354,4,'NH 14 Main',8,'Water Resort');

INSERT INTO Accomodation VALUES (008,300,1,977775354,3,'Leftop Temple',8,'Stay Resort');

INSERT INTO Accomodation VALUES (009,400,1,985575354,4,'Hellsing Road',8,'Calm Resort');

INSERT INTO Accomodation VALUES (010,500,1,987565654,4,'Wayover bridge',8,'Sleeper Resort');

INSERT INTO Accomodation VALUES (011,8,1,98765354,3,'KH52 Subway',3,'Wang Guest Hotel');

INSERT INTO Accomodation VALUES (012,10,1,987656564,4,'MK14 Road',3,'Ulter comfort Guest Hotel');

**BedType Table:**

Insert into Bedtype values(00001,'Single',1);

Insert into Bedtype values(00002,'Single',2);

Insert into Bedtype values(00011,'Single',3);

Insert into Bedtype values(00003,'Twin',1);

Insert into Bedtype values(00004,'Twin',2);

Insert into Bedtype values(00005,'Double',1);

Insert into Bedtype values(00006,'Double',2);

Insert into Bedtype values(00007,'Queen',1);

Insert into Bedtype values(00008,'Queen',2);

Insert into Bedtype values(00009,'King',1);

Insert into Bedtype values(00010,'King',2);

**RoomType Table:**

Insert into RoomType Values(001,5,'Economy','No');

Insert into RoomType Values(002,5,'Premium','Yes');

Insert into RoomType Values(003,5,'Luxury','Yes');

Insert into RoomType Values(004,1,'Economy','No');

Insert into RoomType Values(005,1,'Premium','Yes');

Insert into RoomType Values(006,1,'Luxury','Yes');

Insert into RoomType Values(007,3,'Economy','No');

Insert into RoomType Values(008,3,'Premium','Yes');

Insert into RoomType Values(009,8,'Economy','No');

Insert into RoomType Values(010,8,'Premium','Yes');

**Room Beds Table:**

Insert into RoomBeds values(00001,001,1,100);

Insert into RoomBeds values(00001,002,1,150);

Insert into RoomBeds values(00001,003,1,200);

Insert into RoomBeds values(00003,004,1,200);

Insert into RoomBeds values(00003,005,1,225);

Insert into RoomBeds values(00003,006,1,300);

Insert into RoomBeds values(00005,001,1,300);

Insert into RoomBeds values(00005,002,1,350);

Insert into RoomBeds values(00005,003,1,400);

Insert into RoomBeds values(00007,001,1,600);

Insert into RoomBeds values(00007,002,1,650);

Insert into RoomBeds values(00007,003,1,700);

**Room Table:**

Insert into room values(00001,001,001,001,TO\_DATE('04/27/2019', 'mm/dd/yyyy'),TO\_DATE('02/27/2020', 'mm/dd/yyyy'),'GREEN');

Insert into room values(00002,002,002,002,TO\_DATE('01/27/2019', 'mm/dd/yyyy'),TO\_DATE('02/27/2020', 'mm/dd/yyyy'),'GREEN');

Insert into room values(00003,003,003,002,TO\_DATE('01/27/2019', 'mm/dd/yyyy'),TO\_DATE('02/27/2030', 'mm/dd/yyyy'),'GREEN');

Insert into room values(00004,004,005,002,TO\_DATE('01/27/2019', 'mm/dd/yyyy'),TO\_DATE('02/27/2020', 'mm/dd/yyyy'),'GREEN');

Insert into room values(00005,005,002,003,TO\_DATE('02/27/2019', 'mm/dd/yyyy'),TO\_DATE('01/27/2020', 'mm/dd/yyyy'),'GREEN');

Insert into room values(00006,006,001,003,TO\_DATE('03/27/2019', 'mm/dd/yyyy'),TO\_DATE('01/27/2020', 'mm/dd/yyyy'),'GREEN');

Insert into room values(00007,004,002,003,TO\_DATE('05/27/2019', 'mm/dd/yyyy'),TO\_DATE('08/27/2019', 'mm/dd/yyyy'),'GREEN');

Insert into room values(00008,001,001,003,TO\_DATE('06/27/2019', 'mm/dd/yyyy'),TO\_DATE('06/27/2019', 'mm/dd/yyyy'),'GREEN');

**Registered Users Table:**

INSERT INTO RegisteredUser VALUES (100,'Baljeet','Singh','NA','MacCallum',987654210,4526568452,1000);

INSERT INTO RegisteredUser VALUES (101,'Meghna','Mathur','NA','Frankford',98994210,89668452,5000);

INSERT INTO RegisteredUser VALUES (102,'John','Map','NA','Waterview',88854210,916568452,8000);

**Guest Table:**

INSERT INTO Guest VALUES (200,'Yui','Wan','NA',999985410);

INSERT INTO Guest VALUES (201,'Won','Yang','NA',989655410);

INSERT INTO Guest VALUES (202,'Vedika','Dev','NA',997985410);

INSERT INTO Guest VALUES (203,'Kavya','Mehra','NA',941985410);

**Customer Table:**

INSERT INTO Customer VALUES (001,100,Null);

INSERT INTO Customer VALUES (002,101,Null);

INSERT INTO Customer VALUES (003,102,Null);

INSERT INTO Customer VALUES (004,Null,200);

INSERT INTO Customer VALUES (005,Null,201);

INSERT INTO Customer VALUES (006,Null,202);

INSERT INTO Customer VALUES (007,Null,203);

# **TRIGGERS: (#definitions)**

1. **This trigger is executed after a booking on a room has been made. Once a room is booked, it’s status needs to be changed from available (‘green’) to booked (‘yellow’).**

CREATE OR REPLACE TRIGGER room\_status AFTER

INSERT ON Booking

FOR EACH ROW

DECLARE

before\_status room.currentstatus%type;

after\_status room.currentstatus%type;

v\_Roomid room.roomid%type;

BEGIN

SELECT room.currentstatus, roomid into before\_status, v\_roomid from room where roomid=:new.roomid;

update ROOM

SET

CurrentStatus='YELLOW'

WHERE ROOMID =: new.Roomid;

SELECT ROOM.Currentstatus into after\_status from room where roomid=:new.roomid;

dbms\_output.put\_line('Room Id:'||v\_roomid ||' has been set from '|| before\_status||' to ' ||after\_status);

END;

/

1. **This trigger is executed before a booking on a room can be made. Checks before booking, whether the requested room is available to book.**

CREATE OR REPLACE TRIGGER trg\_before\_booking BEFORE

INSERT ON Booking

FOR EACH ROW

DECLARE

found\_count\_1 INT;

found\_count\_2 INT;

found\_count\_3 INT;

CURSOR checkinout\_cur IS

SELECT CHECKIN,CHECKOUT

FROM Booking

WHERE Roomid =:new.roomid;

l\_booking checkinout\_cur%ROWTYPE;

INVALID\_CUSTOMER EXCEPTION;

INVALID\_ROOM EXCEPTION;

BOOK\_NOT\_AVAILABLE EXCEPTION;

NOT\_AVAILABLE\_ROOM EXCEPTION;

DATE\_INVALID EXCEPTION;

BEGIN

-- dbms\_output.put\_line('Room Number is not Valid');

SELECT COUNT(\*) INTO found\_count\_1 FROM CUSTOMER WHERE CUSTOMERID=:new.CUSTOMERID;

SELECT COUNT(\*) INTO found\_count\_2 FROM ROOM WHERE ROOMID=:new.ROOMID;

SELECT COUNT(\*) INTO found\_count\_3 FROM ROOM WHERE ROOMID=:new.ROOMID and ROOM.CURRENTSTATUS = 'YELLOW' OR ROOM.CURRENTSTATUS = 'GREEN';

IF(found\_count\_1=0) THEN

RAISE INVALID\_CUSTOMER;

ELSIF (found\_count\_2=0) THEN

RAISE INVALID\_ROOM;

ELSIF (found\_count\_3=0) THEN

RAISE NOT\_AVAILABLE\_ROOM;

ELSIF (:new.checkin>:new.checkout) THEN

RAISE DATE\_INVALID;

ELSE

NULL;

END IF;

OPEN checkinout\_cur;

LOOP

FETCH checkinout\_cur INTO l\_booking;

EXIT WHEN checkinout\_cur%NOTFOUND;

if ((l\_booking.checkin<=:new.checkin and l\_booking.checkout>=:new.checkin) or (l\_booking.checkin<=:new.checkout and l\_booking.checkout>=:new.checkout) ) THEN

RAISE BOOK\_NOT\_AVAILABLE;

END if;

END Loop;

CLOSE checkinout\_cur;

EXCEPTION

WHEN INVALID\_CUSTOMER THEN

Raise\_application\_error(-20319, 'Invalid\_Customer');

WHEN INVALID\_ROOM THEN

Raise\_application\_error(-20320, 'Invalid ROOM');

WHEN BOOK\_NOT\_AVAILABLE THEN

Raise\_application\_error(-20321, 'Booking Date not available');

WHEN NOT\_AVAILABLE\_ROOM THEN

Raise\_application\_error(-20322, 'Room is not available for booking');

WHEN DATE\_INVALID THEN

Raise\_application\_error(-20323, 'Checkin Date can not be more than checkout date');

END;

/

# **Procedures (#definitions )**

1. **Procedure to create a new booking for a particular accommodation room. The procedure takes in as inputs the bookingID, bookingDate, customerID, CheckIn and CheckOut dates, No of adults and children, and the room id of the room to be booked. These data values are then inserted into the Booking table.**

CREATE OR REPLACE PROCEDURE booking\_proc (

bookingID VARCHAR,

bookingDate DATE,

customerID VARCHAR,

checkIN DATE,

checkOut DATE,

noOfAdults INT,

noOfChildren INT,

roomId INT

) IS

INVALID\_CUSTOMER EXCEPTION;

BEGIN

INSERT into Booking values (bookingID, bookingDate, customerID, checkIN, checkOut, noOfAdults, noOfChildren, roomId);

dbms\_output.put\_line('Booking Confirmed');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

dbms\_output.put\_line('Error!');

WHEN OTHERS THEN

IF(SQLCODE=-20319) THEN

dbms\_output.put\_line('Error: Invalid\_Customer');

ELSIF(SQLCODE=-20320) THEN

dbms\_output.put\_line('Error: Invalid ROOM');

ELSIF(SQLCODE=-20321) THEN

dbms\_output.put\_line('Error: Booking Date not available');

ELSIF(SQLCODE=-20322) THEN

dbms\_output.put\_line('Error: Room is not available for booking');

ELSIF(SQLCODE=-20323) THEN

dbms\_output.put\_line('Error: Checkin Date is more than CheckOut Data');

ELSE

NULL;

END IF;

END;

/

1. **Procedure to return the count of available rooms of a given room type of a particular accommodation between the specified check in and check out dates. It takes in the search filters such as accommodation ID, room type, check in and check out dates as inputs. The procedure returns the number of rooms of a particular type available during that period.**

CREATE OR REPLACE PROCEDURE available\_room\_count (

accID IN INT,

roomTypeName IN VARCHAR,

checkIN IN DATE,

checkOut IN DATE,

rtn OUT room.currentstatus%type,

roomCount OUT INT

) IS

INVALID\_CUSTOMER EXCEPTION;

BEGIN

SELECT RoomTypeName, COUNT(\*) INTO rtn, roomCount FROM ROOM NATURAL JOIN RoomType WHERE AccID = accID AND RoomTypeName = roomTypeName AND GREENFrom = checkIN AND GREENTo = checkOut AND CurrentStatus = 'GREEN' GROUP BY RoomTypeName;

dbms\_output.put\_line('No. of Rooms: ' || rtn);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

dbms\_output.put\_line('Error!');

WHEN OTHERS THEN

IF(SQLCODE=-20319) THEN

dbms\_output.put\_line('Error: Invalid\_Customer');

ELSIF(SQLCODE=-20320) THEN

dbms\_output.put\_line('Error: Invalid Room');

ELSIF(SQLCODE=-20321) THEN

dbms\_output.put\_line('Error: Booking Date not available');

ELSIF(SQLCODE=-20322) THEN

dbms\_output.put\_line('Error: No rooms available for booking');

ELSIF(SQLCODE=-20323) THEN

dbms\_output.put\_line('Error: Checkin Date is more than CheckOut Data');

ELSE

NULL;

END IF;

END;

/