Project Report

Photo Collage Management System

BUAN 6320.008

KANNAN SRIKANTH

Photo Maker

Group 11

Group members

Rachana Mahapatra
Nikhita Mandava
Yamini Nathani
Vinayak Shanbhag
Kajal Singhal
Shikhar Sinha

Table of Contents

**********PROJECT LOG**********

Meeting date	Time duration	Members present	Items discussed
09-23-2023	5:00pm- 6:00pm	6	• Ideation • Problem statement discussion • Coining business scenario
09-25-2023	10:00pm- 1:00pm	6	• Draft for Proposal
09-28-2023	10:00pm-10:0pm	6	• Edit for Proposal
10-05-2023	4:00pm-5:00pm	6	• Decided on Tables
10-14-2023	4:00pm-5:00pm	6	• Assigned and Created tables
10-19-2023	10:30pm-11:30pm	6	• Worked on Draft report
10-30-2023	8:00pm-9:00pm	6	• Insertion and complex queries discussed.
11-04-2023	4:00pm-5:30 pm	6	• Discussed how to work for the videos, andplanned all complex queries.
11-20-2023	2:00pm– 4:00pm	6	• Complexed queries done
11-30-2023	1:00 pm–3:00 pm	6	• Procedures,Triggers discussed
11-31-2023	12:00pm-2:00pm	6	• Video shoot
12-01-2023	12:00pm-4:00pm	6	• Video shoot
12-04-2023	5:00pm -6:00 pm	6	• Functions discussed
12-09-2023	1:00pm -9:00 pm	6	• Prepared Documentation

INTRODUCTION:

In recent years, the photography landscape has seen an unprecedented surge in self-taught photographers entering the scene, empowered by the democratization of photography through accessible digital tools and platforms. This surge has given rise to a diverse range of talents contributing to the richness of the visual narrative.Each one of them striving to find their niche in the cutthroat field of visual storytelling. The potential and challenges presented by this community continue to expand alongside the community itself.

PROJECT OBJECTIVE AND SCOPE:

The growing presence of amateur photographers aiming to expand their market share underscores a profound desire for acceptance and professional advancement. These emerging talents face challenges such as cultivating a dedicated clientele and managing the complexities of handling multiple assignments. In response to these challenges, the project aims to deliver a robust solution that not only facilitates portfolio promotion but also addresses the intricacies of managing photos, discounts, and customer contacts.

The objective of our Photo Collage Management System is to meet the growing demand in the photography industry. It serves as a vital tool for displaying creativity, efficiently organizing portfolios, and expanding its reach to a broader audience. As photographers increasingly seek to establish distinct identities and undertake diverse projects, our system serves as an essential, unifying platform amid the evolving industry landscape.

Beyond broadening exposure to a larger audience, the system is designed to preserve the integrity of photographers' original works. A tailored solution is essential to stimulate development, innovation, and seamless interactions between photographers and potential clients, aligning with the rising demand for such a comprehensive system.

In the following sections, we provide detailed insights into the Photography Portfolio Management System, outlining its key features, target users, and specific challenges it aims to tackle.

Target Audience:

The primary target audience for the PHOTOMAKER are **amateur photographers** looking to expand their market share and **consumers seeking photographers** for various occasions. The system caters to **photographers of all expertise levels** and genres.

Problems being addressed:

1. **Limited Exposure:** Amateur photographers often face limited exposure. Photomaker addresses this by providing a platform where they can display a selection of their portfolios.

2. **Clientele Building Challenges:** Building a loyal clientele can be challenging for self-taught photographers. PHOTO MAKER facilitates the process by enabling them to reach a wider audience.
3. **Project Acquisition:** Taking on new projects and identifying abilities is a hurdle for many photographers. PHOTO MAKER encourages project acquisition by providing a platform for photographers to showcase their work and attract potential clients.
4. **Picture Abuse and Ownership Concerns:** The system safeguards against picture abuse and grants photographers' ownership of their original creations, ensuring their work is protected.

Key Features:

1. **Photographer Database:**
   - Detailed profiles of photographers including specialties, backgrounds, locations, fees, and availability.
2. **Portfolio Promotion:**
   - Picture collage management system for creating original designs.
   - Customization tools for photographers to personalize their portfolios.
   - Efficient management tools for organizing photographs based on events.
3. **Client Interaction:**
   - Discount management features to attract consumers.
   - Advanced search functionality for consumers based on budget, interests, and occasions.
4. **Subscription Models:**
   - Basic: User registration and access to photographer profiles.
   - Premium: Auction feature for photographers to showcase and auction their images to a larger audience.

UNIQUE SELLING PROPOSITION (USP):

1. An inclusive platform welcoming photographers of all expertise levels and genres.
2. Protection against picture abuse and clear ownership rights for photographers.
3. User-friendly features that enable potential customers to search for and select the photographers they want.

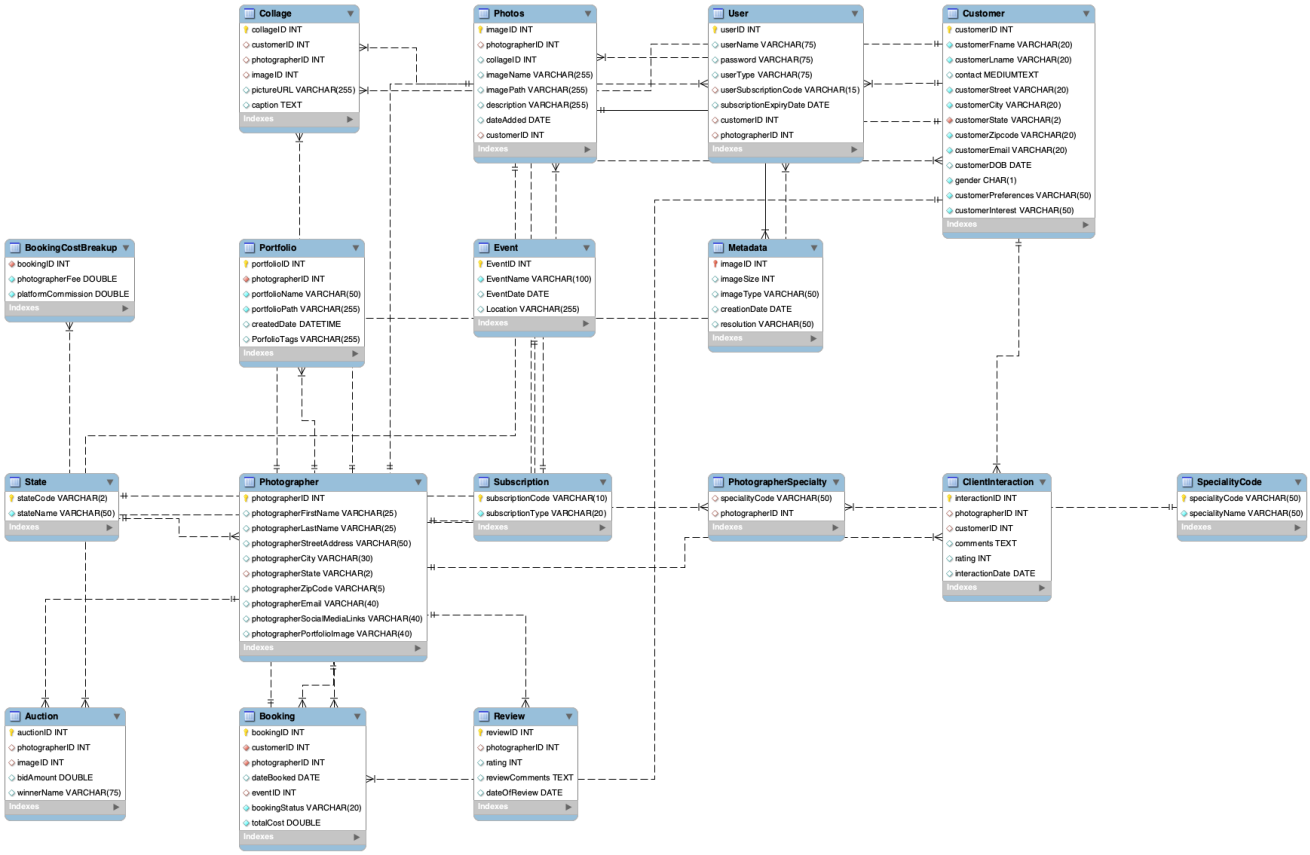
**********DESIGN OF THE DATABASE**********

Table Name	Primary Key	Foreign Key	Non-key attributes	No. of Rows in Table
Customer	customerID	(customerState) REFERENCES State(stateCode)	customerID customerFname customerLname contact customerStreet customerCity customerState customerZipcode customerEmail customer gender customerPreferences customerInterest	**13**
Photographer	photographerID	(photographerState) REFERENCES State(stateCode)	photographerID photographerFirstName photographerLastName photographerStreetAddress photographerCity photographerState photographerZipCode photographerEmail photographerSocialMediaLinks photographerPortfolioImage	**15**
Portfolio	portfolioID	(photographerID) REFERENCES Photographer(photographerID)	portfolioID photographerID portfolioName portfolioPath createdDate PorfolioTags	**17**
Subscription	subscriptionCode		subscriptionCode subscriptionType	**2**

User	userID	customerID PhotographerID userSubscriptionCode	userID userName password userTypeV userSubscriptionCode subscriptionExpiryDate customerID photographerID	**25**
SpecialityCode	specialityCode		specialityCode specialityName	**10**
PhotographerSpecialty		photographerID specialityCode	specialityCode photographerID	**15**
Photos	imageID	CustomerID PhotographerID	imageID photographerID collageID imageName imagePath description dateAdded customerID	**40**
Collage	collageID	(customerID) REFERENCES Customer(customer ID), (photographerID) REFERENCES Photographer(photographerID), (imageID) REFERENCES Photos(imageID)	collageID customerID photographerID imageID pictureURL	**30**
Event	EventID		EventID EventName EventDate Location	**15**
Booking	bookingID	(customerID) REFERENCES Customer(customer	bookingID, customerID photographerID	**15**

		ID) photographerID) REFERENCES Photographer(photographerID) (eventID) REFERENCES Event(eventID)	dateBooked eventID bookingStatustotalCost double,	
BookingCostBreakup	(bookingID) REFERENCES Booking(bookingID)		bookingID photographerFee platformCommission	**15**
ClientInteraction	interactionID	photographerID) REFERENCES Photographer(photographerID), (customerID) REFERENCES Customer(customerID)	interactionID photographerID customerID comments rating interactionDate	**10**
Review	reviewID	(photographerID) REFERENCES Photographer(photographerID)	reviewID photographerID rating reviewComments dateOfReview	**7**
Auction	auctionID	photographerID) REFERENCES Photographer(PhotographerID) (imageID) REFERENCES Photos(imageID)	auctionID photographerID imageID bidAmount winnerName	**20**
Metadata	imageID	(imageID) REFERENCES Photos(imageID)	imageID imageSize imageType creationDate resolution	**40**
State	stateCode		stateCode stateName	**50**

********** ER DIAGRAM**********


Collage
collageID INT
customerID INT
photographerID INT
imageID INT
pictureURL VARCHAR(255)
caption TEXT
Indexes
Photos
imageID INT
photographerID INT
collageID INT
imageName VARCHAR(255)
imagePath VARCHAR(255)
description VARCHAR(255)
dateAdded DATE
customerID INT
Indexes
User
userID INT
userName VARCHAR(75)
password VARCHAR(75)
userType VARCHAR(75)
userSubscriptionCode VARCHAR(15)
subscriptionExpiryDate DATE
customerID INT
photographerID INT
Indexes
Customer
customerID INT
customerFname VARCHAR(20)
customerLname VARCHAR(20)
contact MEDIUMTEXT
customerStreet VARCHAR(20)
customerCity VARCHAR(20)
customerState VARCHAR(2)
customerZipcode VARCHAR(20)
customerEmail VARCHAR(20)
customerDOB DATE
gender CHAR(1)
customerPreferences VARCHAR(50)
customerInterest VARCHAR(50)
Indexes
BookingCostBreakup
bookingID INT
photographerFee DOUBLE
platformCommission DOUBLE
Indexes
Portfolio
portfolioID INT
photographerID INT
portfolioName VARCHAR(50)
portfolioPath VARCHAR(255)
createdDate DATETIME
PorfolioTags VARCHAR(255)
Indexes
Event
EventID INT
EventName VARCHAR(100)
EventDate DATE
Location VARCHAR(255)
Indexes
Metadata
imageID INT
imageSize INT
imageType VARCHAR(50)
creationDate DATE
resolution VARCHAR(50)
Indexes
State
stateCode VARCHAR(2)
stateName VARCHAR(50)
Indexes
Photographer
photographerID INT
photographerFirstName VARCHAR(25)
photographerLastName VARCHAR(25)
photographerStreetAddress VARCHAR(50)
photographerCity VARCHAR(30)
photographerState VARCHAR(2)
photographerZipCode VARCHAR(5)
photographerEmail VARCHAR(40)
photographerSocialMediaLinks VARCHAR(40)
photographerPortfolioImage VARCHAR(40)
Indexes
Subscription
subscriptionCode VARCHAR(10)
subscriptionType VARCHAR(20)
Indexes
PhotographerSpecialty
specialityCode VARCHAR(50)
photographerID INT
Indexes
ClientInteraction
interactionID INT
photographerID INT
customerID INT
comments TEXT
rating INT
interactionDate DATE
Indexes
SpecialityCode
specialityCode VARCHAR(50)
specialityName VARCHAR(50)
Indexes
Auction
auctionID INT
photographerID INT
imageID INT
bidAmount DOUBLE
winnerName VARCHAR(75)
Indexes
Booking
bookingID INT
customerID INT
photographerID INT
dateBooked DATE
eventID INT
bookingStatus VARCHAR(20)
totalCost DOUBLE
Indexes
Review
reviewID INT
photographerID INT
rating INT
reviewComments TEXT
dateOfReview DATE
Indexes

The SQL Code

```
CREATE DATABASE IF NOT EXISTS PhotoMaker;

USE PhotoMaker;


-- State Table

CREATE TABLE IF NOT EXISTS State (

stateCodeVARCHAR(2) PRIMARY KEY,

stateNameVARCHAR(50) NOT NULL

);




-- Insert Data

INSERT INTO State (stateCode, stateName)
VALUES
  ('AL', 'Alabama'),
  ('AK', 'Alaska'),
  ('AZ', 'Arizona'),
  ('AR', 'Arkansas'),
  ('CA', 'California'),
  ('CO', 'Colorado'),
  ('CT', 'Connecticut'),
  ('DE', 'Delaware'),
  ('FL', 'Florida'),
  ('GA', 'Georgia'),
  ('HI', 'Hawaii'),
  ('ID', 'Idaho'),
  ('IL', 'Illinois'),
  ('IN', 'Indiana'),
  ('IA', 'Iowa'),
  ('KS', 'Kansas'),
  ('KY', 'Kentucky'),
```

('LA', 'Louisiana'), ('ME', 'Maine'),
('MD', 'Maryland'),
('MA', 'Massachusetts'),
('MI', 'Michigan'),
('MN', 'Minnesota'),
('MS', 'Mississippi'),
('MO', 'Missouri'),
('MT', 'Montana'),
('NE', 'Nebraska'),
('NV', 'Nevada'),
('NH', 'New Hampshire'),
('NJ', 'New Jersey'),
('NM', 'New Mexico'),
('NY', 'New York'),
('NC', 'North Carolina'),
('ND', 'North Dakota'),
('OH', 'Ohio'),
('OK', 'Oklahoma'),
('OR', 'Oregon'),
('PA', 'Pennsylvania'),
('RI', 'Rhode Island'),
('SC', 'South Carolina'),
('SD', 'South Dakota'),
('TN', 'Tennessee'),
('TX', 'Texas'),
('UT', 'Utah'),
('VT', 'Vermont'),
('VA', 'Virginia'),
('WA', 'Washington'),
('WV', 'West Virginia'),
('WI', 'Wisconsin'),
('WY', 'Wyoming');

-- **Customer Table**

CREATE TABLE IF NOT EXISTS Customer (

customerID INT UNIQUE,

customerFnameVARCHAR(20) NOT NULL,

customerLnameVARCHAR(20) NOT NULL,

contact LONG,

customerStreetVARCHAR(20) NOT NULL,

customerCityVARCHAR(20) NOT NULL ,

```
customerStateVARCHAR(2) NOT NULL,

customerZipcodeVARCHAR(20) NOT NULL ,

customerEmailVARCHAR(20) NOT NULL,

customerDOB DATE,

  gender CHAR(1) NOT NULL ,

customerPreferencesVARCHAR(50) NOT NULL ,

customerInterestVARCHAR(50) NOT NULL ,

  PRIMARY KEY (customerID),

FOREIGN KEY (customerState) REFERENCES State(stateCode)

);
```

-- Insertion INTO Customer

```
INSERT INTO Customer VALUES
(1, 'Nikhita', 'Mandava', 9084948476, '3000 Northside', 'Richardson','TX', '75080', 'abc@gmail.com',
'1976-12-1', 'F', 'Budget save', 'Wedding Photography'),
(2, 'Sanjana', 'Sharma', 4567890989, '2130 Westside', 'Iving', 'CA','56765', 'sanjana@gmail.com',
'1986-09-1', 'F', 'Candid pictures', 'Bridal Photography'),
(3, 'Ranjith', 'Komra', 9012390038, '100 Northside', 'Sadrea', 'TX', '75084', 'ranjith@gmail.com',
'1979-02-09', 'M', 'Budget save', 'Food Photography'),
(4, 'Firoz', 'Rehman', 9081002002, '3023 Bella', 'San Jose', 'CA', '50930', 'rehman@gmail.com', '1992-
11-10', 'M', 'Luxury place- Destination', 'Wedding Photography'),
(5, 'Raha', 'Malhotra', 4090809079, '1000 Benside', 'Montyson', 'FL', '75980', 'raha@gmail.com',
'1986-09-1', 'F', 'Destination clicks candid', 'Wedding Photography'),
(6, 'Doris', 'Hartwig', 4540090989, '4726 - 11th Ave', 'Seattle', 'WA', '98105', 'doris@gmail.com',
'1980-09-09', 'F', 'Luxurious setup', 'Wedding Photography'),
(7, 'William', 'Thompson', 9012390000, '122 Spring Drive', 'Duvall', 'TX', '98019',
'william@gmail.com', '1983-03-09', 'M', 'Budget save', 'Wedding Photography'),
(8, 'Gary', 'Hallmark', 4567890300, 'Route 2 Box 203B', 'Auburn', 'TX', '98006', 'gary@gmail.com',
'1995-10-10', 'M', 'Destination', 'Wedding Photography'),
(9, 'Raj', 'Kumar', 2145670001, '1000 Bethanystreet', 'Fermont', 'CA', '500034',
'rajkumar@gmail.com', '1976-07-01', 'M', 'Destination clicks-candid', 'Wedding Photography'),
(10, 'Andrea', 'Bullock', 5001200032, '312 MountBonnet', 'Austin', 'TX', '80909',
'andreab@gmail.com', '1985-08-03', 'F', 'Destination clicks, candid', 'Wedding Photography');
```

```
INSERT INTO Customer VALUES
(11, 'Rachel', 'Green', 9084348436, '2300 Northside', 'Richardson','TX', '75080', 'rachel@gmail.com',
'1988-11-01', 'F', 'Destination', 'Wedding Photography'),
(12, 'Brown', 'Waine', 9023248436, '2300 Pearl', 'Irwing','TX', '70000', 'brown@gmail.com', '1998-10-
01', 'F', 'Budget save', 'Food Photography');

INSERT INTO Customer VALUES
(13, 'Betsy', 'Stadick', 9000348436, '611 Alpine Drive', 'Palm Springs','CA', '92263',
'BetsySk@gmail.com', '1988-11-11', 'F', 'Destination', 'Wedding Photography');



INSERT INTO ClientInteraction (photographerID, customerID, comments, rating,
interactionDate) VALUES
(1002, 2, 'Good conversation', 3, '2023-01-23'),
(1003, 2, 'Beautiful shots!', 4, '2023-02-20'),
(1001, 5, 'Great experience!', 5, '2023-05-12'),
(1001, 6, 'Beautiful shots!', 4, '2023-04-18');



-- Photographer Table

CREATE TABLE Photographer (

photographerID INT NOT NULL,

photographerFirstNameVARCHAR(25) DEFAULT NULL,

photographerLastNameVARCHAR(25) DEFAULT NULL,

photographerStreetAddressVARCHAR(50) DEFAULT NULL,

photographerCityVARCHAR(30) DEFAULT NULL,

photographerStateVARCHAR(2),

photographerZipCodeVARCHAR(5) DEFAULT NULL,

photographerEmailVARCHAR(40) DEFAULT NULL,

photographerSocialMediaLinksVARCHAR(40) DEFAULT NULL,

photographerPortfolioImageVARCHAR(40) DEFAULT NULL,
```

```
  PRIMARY KEY (photographerID),

  FOREIGN KEY (photographerState) REFERENCES State(stateCode)

);

-- Insert Data

INSERT INTO Photographer VALUES
('1001', 'Kerry', 'Patterson', '9877 Hacienda Drive', 'San Antonio', 'TX', '78284',
'KerryPatterson@gmail.com', 'www.linkedin.com/KerryPatterson', '1234'),
('1002', 'David', 'Hamilton', '908 W.CapitalWay', 'Tacoma', 'WA', '98413',
'DavidHamilton@gmail.com', 'www.linkedin.com/DavidHamilton', '12234'),
('1003', 'Betsy', 'Stadick', '611 Alpine Drive', 'Palm Springs', 'CA', '92263', 'BetsyStadick@gmail.com',
'www.linkedin.com/BetsyStadick', '1234'),
('1004', 'Janice', 'Galvin', '4110 Old Redmond Rd.', 'Redmond', 'WA', '98052',
'JaniceGalvin@gmail.com', 'www.linkedin.com/JaniceGalvin', '1234'),
('1005', 'Doris', 'Hartwig', '4726 - 11th Ave. N.E.', 'Seattle', 'WA', '98105', 'DorisHartwig@gmail.com',
'www.linkedin.com/DorisHartwig', '1234'),
('1006', 'Scott', 'Bishop', '66 Spring Valley Drive', 'Medford', 'OR', '97501', 'ScottBishop@gmail.com',
'www.linkedin.com/ScottBishop', '1234'),
('1007', 'Elizabeth', 'Hallmark', 'Route 2, Box 203B', 'Auburn', 'WA', '98002',
'ElizabethHallmark@gmail.com', 'www.linkedin.com/ElizabethHallmark', '1234'),
('1008', 'Sara', 'Sheskey', '16679 NE 41st Court', 'Portland', 'OR', '97208', 'SaraSheskey@gmail.com',
'www.linkedin.com/SaraSheskey', '1234'),
('1009', 'Karen', 'Smith', '30301 - 166th Ave. N.E.', 'Eugene', 'OR', '97401', 'KarenSmith@gmail.com',
'www.linkedin.com/KarenSmith', '1234'),
('1010', 'Marianne', 'Wier', '908 W. Capital Way', 'Tacoma', 'WA', '98413',
'MarianneWier@gmail.com', 'www.instagram.com/MarianneWier', '1234'),
('1011', 'John', 'Kennedy', '16679 NE 41st Court', 'Portland', 'OR', '97208', 'JohnKennedy@gmail.com',
'www.instagram.com/JohnKennedy', '1234'),
('1012', 'Sarah', 'Thompson', '2222 Springer Road', 'Lubbock', 'TX', '79402',
'SarahThompson@gmail.com', 'www.instagram.com/SarahThompson', '1234'),
('1013', 'Michael', 'Viescas', '15127 NE 24th, #383', 'Redmond', 'WA', '98052',
'MichaelViescas@gmail.com', 'www.instagram.com/MichaelViescas', '1234'),
('1014', 'Kendra', 'Bonnicksen', '12330 Larchlemont Lane', 'Seattle', 'WA', '98105',
'KendraBonnicksen@gmail.com', 'www.instagram.com/KendraBonnicksen', '1234'),
('1015', 'Brannon', 'Jones', '777 Fenexet Blvd', 'Long Beach', 'CA', '90809',
'BrannonJones@gmail.com', 'www.instagram.com/BrannonJones', '1234');


-- Portfolio Table

CREATE TABLE Portfolio (

portfolioID INT AUTO_INCREMENT PRIMARY KEY,

photographerID INT NOT NULL,
```

```
portfolioNameVARCHAR(50) NOT NULL DEFAULT '',

portfolioPathVARCHAR(255) NOT NULL DEFAULT '',

createdDate DATETIME,

PorfolioTagsVARCHAR(255),

  FOREIGN KEY (photographerID) REFERENCES Photographer(photographerID)

);
```

```
 -- Insert Data
INSERT INTO Portfolio (portfolioID, portfolioName, portfolioPath, createdDate, PorfolioTags,
photographerID)
VALUES
(1, 'PF1234', 'http://example.com/portfolio1', '2010-12-10 18:36:20', 'Wedding', 1001),
(2, 'PF1235', 'http://example.com/portfolio2', '2021-11-03 11:36:22', 'Wedding', 1002),
(3, 'PF1236', 'http://example.com/portfolio3', '2021-09-14 12:46:34', 'Wildlife', 1003),
(4, 'PF1237', 'http://example.com/portfolio4', '2022-09-10 1:34:40', 'Celebrity', 1004),
(5, 'PF1238', 'http://example.com/portfolio5', '2021-10-22 08:45:50', 'Food', 1005),
(6, 'PF1239', 'http://example.com/portfolio6', '2022-04-09 10:36:10', 'Wedding', 1006),
(7, 'PF1240', 'http://example.com/portfolio7', '2022-06-15 14:20:45', 'Fashion', 1007),
(8, 'PF1241', 'http://example.com/portfolio8', '2022-08-20 09:55:30', 'Landscape', 1008),
(9, 'PF1242', 'http://example.com/portfolio9', '2022-12-05 16:40:15', 'Wildlife', 1009),
(10, 'PF1243', 'http://example.com/portfolio10', '2023-02-18 22:10:55', 'Wedding', 1010),
(11, 'PF1244', 'http://example.com/portfolio11', '2023-04-30 08:05:10', 'Food', 1011),
(12, 'PF1245', 'http://example.com/portfolio12', '2023-07-12 19:30:25', 'Fashion', 1012),
(13, 'PF1246', 'http://example.com/portfolio13', '2023-09-24 07:15:40', 'Landscape', 1013),
(14, 'PF1247', 'http://example.com/portfolio14', '2023-11-08 13:50:05', 'Wildlife', 1014),
(15, 'PF1248', 'http://example.com/portfolio15', '2024-01-22 18:25:20', 'Wedding', 1015),
(16, 'PF1249', 'http://example.com/portfolio16', '2024-03-05 10:00:35', 'Food', 1001),
(17, 'PF1250', 'http://example.com/portfolio17', '2024-05-17 21:25:50', 'Fashion', 1002);
```

```
-- Subscription Table

CREATE TABLE IF NOT EXISTS Subscription (

subscriptionCodeVARCHAR(10) NOT NULL DEFAULT '',

subscriptionTypeVARCHAR(20) NOT NULL DEFAULT '',

    PRIMARY KEY (subscriptionCode)

);
```

-- **Inserting values into Subscription table**

```
INSERT INTO Subscription (subscriptionCode, subscriptionType) VALUES

('B01', 'Basic'),

('P01', 'Premium');
```

-- **User Table**

```
CREATE TABLE IF NOT EXISTS User (

userID INT,

userNameVARCHAR(75),

password VARCHAR(75),

userTypeVARCHAR(75),

userSubscriptionCode  VARCHAR(15),

subscriptionExpiryDate DATE,

customerID INT,

photographerID INT,

  PRIMARY KEY (userID),

  FOREIGN KEY (customerID) REFERENCES Customer(customerID),

  FOREIGN KEY (PhotographerID) REFERENCES Photographer(PhotographerID),

  FOREIGN KEY (userSubscriptionCode) REFERENCES Subscription(subscriptionCode)

);
```

```
INSERT INTO User (userID, userName, password, userType,
userSubscriptionCode,subscriptionExpiryDate,customerID, photographerID)
VALUES
(1, 'NikhitaMandava', 'password123', 'customer', 'B01', '2024-01-31', 1, NULL),
(2, 'SanjanaSharma', 'securepass', 'customer', 'P01', '2024-03-15', 2, NULL),
```

(3, 'RanjithKomra', 'mypassword', 'customer', NULL, NULL, 3, NULL),
(4, 'FirozRehman', 'wedding123', 'customer', 'B01', '2023-12-01', 4, NULL),
(5, 'RahaMalhotra', 'photography', 'customer', NULL, NULL, 5, NULL),
(6, 'DorisHartwig', 'dorispw', 'customer', 'P01', '2023-11-30', 6, NULL),
(7, 'WilliamThompson', 'will123', 'customer', 'B01', '2023-12-25', 7, NULL),
(8, 'GaryHallmark', 'garypass', 'customer', NULL, NULL, 8, NULL),
(9, 'RajKumar', 'rajpass', 'customer', 'P01', '2024-02-28', 9, NULL),
(10, 'AndreaBullock', 'andreapw', 'customer', NULL, NULL, 10, NULL),
(11, 'KerryPatterson', 'kerrypw', 'photographer', NULL, NULL, NULL, 1001),
(12, 'DavidHamilton', 'davidpass', 'photographer', NULL, NULL, NULL, 1002),
(13, 'BetsyStadick', 'betsypw', 'photographer', NULL, NULL, NULL, 1003),
(14, 'JaniceGalvin', 'janice123', 'photographer', NULL, NULL, NULL, 1004),
(15, 'DorisHartwig', 'dorishart', 'photographer', 'B01', '2023-10-01', NULL, 1005),
(16, 'ScottBishop', 'scottpass', 'photographer', 'P01', '2024-01-15', NULL, 1006),
(17, 'ElizabethHallmark', 'elizabethpw', 'photographer', NULL, NULL, NULL, 1007),
(18, 'SaraSheskey', 'sarapass', 'photographer', 'B01', '2023-08-30', NULL, 1008),
(19, 'KarenSmith', 'karenpw', 'photographer', NULL, NULL, NULL, 1009),
(20, 'MarianneWier', 'marianne123', 'photographer', 'P01', '2024-02-28', NULL, 1010),
(21, 'JohnKennedy', '1234', 'photographer', NULL, NULL, NULL, 1011),
(22, 'SarahThompson', '1234', 'photographer', 'B01', '2023-11-15', NULL, 1012),
(23, 'MichaelViescas', '1234', 'photographer', 'P01', '2024-03-18', NULL, 1013),
(24, 'KendraBonnicksen', '1234', 'photographer', 'B01', '2023-12-05', NULL, 1014),
(25, 'BrannonJones', '1234', 'photographer', 'P01', '2023-12-20', NULL, 1015);

-- **Specialitycode Table**

CREATE TABLE SpecialityCode (

specialityCodeVARCHAR(50) PRIMARY KEY,

specialityNameVARCHAR(50) NOT NULL

);

INSERT INTO SpecialityCode (specialityCode, specialityName) VALUES

('WD', 'Wedding'),

('WL', 'Wildlife'),

('FA', 'Fashion'),

('FD','Food'),

('LA','Landscape'),

('CL','Celebrity');

-- **PhotographerSpeciality Table**

CREATE TABLE PhotographerSpecialty (

specialityCodeVARCHAR(50),

photographerID INT,

FOREIGN KEY (photographerID) REFERENCES Photographer(photographerID),

FOREIGN KEY (specialityCode) REFERENCES SpecialityCode(specialityCode)

);

-- **Insert Data**
INSERT INTO PhotographerSpecialty (specialityCode, photographerID) VALUES
('WD', 1001), ('WD', 1002), ('CL', 1003), ('FA', 1004), ('WD', 1005),
('LA', 1006), ('FA', 1007), ('LA', 1008), ('WL', 1009), ('WD', 1010),
('WL', 1011), ('FD', 1012), ('FA', 1013), ('FD', 1014), ('FA', 1015);

-- **Photos Table**

CREATE TABLE Photos (

imageID INT PRIMARY KEY,

photographerID INT,

collageID INT,

imageNameVARCHAR(255),

imagePathVARCHAR(255),

description VARCHAR(255),

dateAdded DATE,

customerID INT,

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),

```
  FOREIGN KEY (PhotographerID) REFERENCES Photographer(PhotographerID)

);


-- -- Insert Data Photos Table
INSERT INTO Photos (imageID, photographerID, collageID, imageName, imagePath, description, dateAdded, customerID)
VALUES(1,1001,201,'Sunset Pic', 'https://drive.google.com/sunset', 'Beautiful sunset views', '2023-11-01', 1),
  (2, 1001, 202, 'Mountain View', 'https://drive.google.com/mountain', 'Scenic mountain landscape', '2023-10-25', 2),
  (3, 1001, 203, 'City Night', 'https://drive.google.com/city', 'Cityscape at night', '2023-11-05', 2),
  (4, 1002, 204, 'Flower Garden', 'https://drive.google.com/flowers', 'Colorful flower garden', '2023-10-30', 3),
  (5, 1002, 205, 'Wildlife', 'https://drive.google.com/wildlife', 'Various wildlife animals', '2023-11-10', 3),
  (6, 1002, 206, 'Northern Lights', 'https://drive.google.com/northern_lights', 'Aurora Borealis display', '2023-10-28', 3),
  (7, 1003, 207, 'Vintage Cars', 'https://drive.google.com/vintage_cars', 'Exhibition of vintage cars', '2023-11-02', 6),
  (8, 1003, 208, 'Fruit Market', 'https://drive.google.com/fruits', 'Colorful fruit market', '2023-11-08', 5),
  (9, 1003, 209, 'Beach Volleyball', 'https://drive.google.com/beach_volleyball', 'Players in action on the beach', '2023-10-27', 5),
  (10, 1003, 210, 'Landscape Art', 'https://drive.google.com/landscape_art', 'Serene landscape painting', '2023-11-12', 6),
  (11, 1004, 210, '', 'https://drive.google.com/landscape_art', 'Serene landscape painting', '2023-11-12', 6),
(12, 1004, 211, 'Sunrise Bliss', 'https://drive.google.com/sunrise_bliss', 'Picturesque sunrise view', '2023-11-15', 7),
  (13, 1004, 212, 'Autumn Leaves', 'https://drive.google.com/autumn_leaves', 'Colorful autumn foliage', '2023-11-18', 8),
  (14, 1005, 213, 'Bird Watching', 'https://drive.google.com/bird_watching', 'Diverse bird species in their habitat', '2023-11-20', 9),
  (15, 1005, 214, 'Luxury Dining', 'https://drive.google.com/luxury_dining', 'Elegant dining setup', '2023-11-22', 10),
  (16, 1005, 215, 'Snowy Mountains', 'https://drive.google.com/snowy_mountains', 'Snow-covered mountain peaks', '2023-11-25', 10),
  (17, 1006, 216, 'Countryside Charm', 'https://drive.google.com/countryside_charm', 'Scenic countryside landscapes', '2023-11-28', 10),
  (18, 1006, 217, 'Golden Hour', 'https://drive.google.com/golden_hour', 'Warm glow during the golden hour', '2023-11-30', 2),
  (19, 1006, 218, 'Beach Sunset', 'https://drive.google.com/beach_sunset', 'Breathtaking sunset by the beach', '2023-12-02', 3),
  (20, 1007, 219, 'Urban Exploration', 'https://drive.google.com/urban_exploration', 'Exploring cityscapes', '2023-12-05', 4),
  (21, 1007, 220, 'Historical Architecture', 'https://drive.google.com/historical_architecture', 'Capturing historical landmarks', '2023-12-08', 4),
```

(22, 1007, 221, 'Fashion Showcase', 'https://drive.google.com/fashion_showcase', 'Glamorous fashion event', '2023-12-10', 5),
(23, 1008, 222, 'Gourmet Delights', 'https://drive.google.com/gourmet_delights', 'Delicious gourmet food photography', '2023-12-12', 6),
(24, 1008, 223, 'Adventurous Hiking', 'https://drive.google.com/adventurous_hiking', 'Thrilling hiking expedition', '2023-12-15', 7),
(25, 1009, 224, 'Modern Cityscape', 'https://drive.google.com/modern_cityscape', 'Contemporary urban landscapes', '2023-12-18', 8),
(26, 1009, 225, 'Whimsical Wildlife', 'https://drive.google.com/whimsical_wildlife', 'Playful wildlife moments', '2023-12-20', 9),
(27, 1010, 226, 'Epic Landscapes', 'https://drive.google.com/epic_landscapes', 'Spectacular natural landscapes', '2023-12-22', 2),
(28, 1010, 227, 'Cultural Festivities', 'https://drive.google.com/cultural_festivities', 'Celebrating diverse cultural events', '2023-12-25', 2),
(29, 1011, 228, 'Dynamic City Life', 'https://drive.google.com/dynamic_city_life', 'Vibrant city life in motion', '2023-12-28', 1),
(30, 1011, 229, 'Charming Countryside', 'https://drive.google.com/charming_countryside', 'Serene countryside beauty', '2023-12-30', 2),
(31, 1012, 230, 'Cozy Winter Scenes', 'https://drive.google.com/cozy_winter_scenes', 'Warm and inviting winter settings', '2024-01-02', 3),
(32, 1012, 231, 'Artistic Food Display', 'https://drive.google.com/artistic_food_display', 'Creative presentation of gourmet dishes', '2024-01-05', 4),
(33, 1013, 232, 'Sun-kissed Beach Retreat', 'https://drive.google.com/sunkissed_beach_retreat', 'Relaxing beach moments with sunlight', '2024-01-08', 5),
(34, 1013, 233, 'Urban Street Fashion', 'https://drive.google.com/urban_street_fashion', 'Trendy street fashion photography', '2024-01-10', 6),
(35, 1014, 234, 'Dynamic Sports Action', 'https://drive.google.com/dynamic_sports_action', 'High-energy sports moments in action', '2024-01-12', 7),
(36, 1014, 235, 'Ethereal Night Sky', 'https://drive.google.com/ethereal_night_sky', 'Magical night sky with celestial wonders', '2024-01-15', 8),
(37, 1015, 236, 'Cultural Heritage Celebrations', 'https://drive.google.com/cultural_heritage_celebrations', 'Preserving and celebrating cultural heritage', '2024-01-18', 9),
(38, 1015, 237, 'Enchanting Wildlife Moments', 'https://drive.google.com/enchanting_wildlife_moments', 'Capturing enchanting moments in the wild', '2024-01-20', 3),
(39, 1004, 238, 'Spring Blossoms', 'https://drive.google.com/spring_blossoms', 'Beautiful blossoms in the spring season', '2024-01-22', 3),
(40, 1004, 239, 'Epic Adventure Landscapes', 'https://drive.google.com/epic_adventure_landscapes', 'Breathtaking landscapes from adventurous journeys', '2024-01-25', 2);

-- Collage Table

CREATE TABLE Collage (

collageID INT AUTO_INCREMENT PRIMARY KEY,

```
customerID INT,
photographerID INT,
imageID INT,
pictureURLVARCHAR(255),
   caption TEXT,
   FOREIGN KEY (customerID) REFERENCES Customer(customerID),
   FOREIGN KEY (photographerID) REFERENCES Photographer(photographerID),
   FOREIGN KEY (imageID) REFERENCES Photos(imageID)
);
```

-- Insert Data

```
INSERT INTO Collage (collageID, customerID, photographerID, imageID, pictureURL, caption)
VALUES
(101, 1,1001, 1, 'https://example.com/pic1.jpg', 'Beautiful landscape'),
(102,2, 1002, 2, 'https://example.com/pic2.jpg', 'Portrait of a person'),
(103,3, 1003, 3, 'https://example.com/pic3.jpg', 'City skyline at night'),
(104,4, 1015, 4, 'https://example.com/pic4.jpg', 'Nature close-up shot'),
(105,5, 1003, 5, 'https://example.com/pic5.jpg', 'Abstract art'),
(106,6, 1002, 6, 'https://example.com/pic6.jpg', 'Fashion photography'),
(107,7, 1002, 7, 'https://example.com/pic7.jpg', 'Wildlife photography'),
(108,8,1001, 8, 'https://example.com/pic8.jpg', 'Street photography'),
(109,9, 1013, 9, 'https://example.com/pic9.jpg', 'Macro photography'),
(110,10, 1011, 10, 'https://example.com/pic10.jpg', 'Architecture photo'),
(111, 1, 1001, 11, 'https://example.com/pic11.jpg', 'Vintage cars exhibition'),
(112, 2, 1002, 12, 'https://example.com/pic12.jpg', 'Food market scene'),
(113, 3, 1003, 13, 'https://example.com/pic13.jpg', 'Beach volleyball action'),
(114, 4, 1014, 14, 'https://example.com/pic14.jpg', 'Mountain landscape'),
(115, 5, 1003, 15, 'https://example.com/pic15.jpg', 'Northern lights display'),
(116, 6, 1001, 16, 'https://example.com/pic16.jpg', 'Cityscape during sunset'),
(117, 7, 1002, 17, 'https://example.com/pic17.jpg', 'Abstract painting'),
(118, 8, 1001, 18, 'https://example.com/pic18.jpg', 'Ocean view at dawn'),
(119, 9, 1015, 19, 'https://example.com/pic19.jpg', 'Birds in flight'),
(120, 10, 1011, 20, 'https://example.com/pic20.jpg', 'Rural landscape'),
(121, 1, 1001, 21, 'https://example.com/pic21.jpg', 'Sunrise over the city'),
(122, 2, 1002, 22, 'https://example.com/pic22.jpg', 'Culinary delights'),
(123, 3, 1003, 23, 'https://example.com/pic23.jpg', 'Aerial view of a beach'),
(124, 4, 1014, 24, 'https://example.com/pic24.jpg', 'Enchanting forest scene'),
```

(125, 5, 1003, 25, 'https://example.com/pic25.jpg', 'Wildlife in the jungle'),
(126, 6, 1001, 26, 'https://example.com/pic26.jpg', 'Dramatic city lights'),
(127, 7, 1002, 27, 'https://example.com/pic27.jpg', 'Colorful fruit market'),
(128, 8, 1001, 28, 'https://example.com/pic28.jpg', 'Urban exploration'),
(129, 9, 1015, 29, 'https://example.com/pic29.jpg', 'Sunset on the horizon'),
(130, 10, 1011, 30, 'https://example.com/pic30.jpg', 'Epic mountain landscape');

-- Event Table

CREATE TABLE Event (

EventID INT PRIMARY KEY,

EventNameVARCHAR(100) NOT NULL,

EventDate DATE,

Location VARCHAR(255)

);

-- **insert Events**
INSERT INTO Event (EventID, EventName, EventDate, Location)
VALUES
(1, 'Wedding Ceremony', '2018-12-01', '25th Avenue, Hartford Street'),
(2, 'Birthday Celebration', '2018-11-01', '30th Avenue, Oxford Street'),
(3, 'Corporate Event', '2020-01-01', '28th Avenue, Lukcross Street'),
(4, 'Engagement Party', '2019-02-01', '5th Cross, Parker Lane'),
(5, 'Graduation Ceremony', '2012-02-10', '10th Cross, Hart Street'),
(6, 'Fashion Show', '2008-11-19', '2nd Cross ,HemmingWay Lane'),
(7, 'Wildlife Photography Exhibition', '2012-08-06', '8th Lane, NewtonCross Avenue'),
(8, 'Food Festival', '2015-12-04', '25th Lane, St.Frankford Street'),
(9, 'Landscape Art Gallery Opening', '2020-05-10', 'Parker Street, NewCross Lane'),
(10, 'Destination Wedding', '2021-08-24', 'Hartford Street ,ParkwayBoulvevard'),
(11, 'Nature Photography Exhibition', '2016-09-11', '6th Avenue, Lombard Street'),
(12, 'Celebrity Event', '2018-12-17', 'Market Street, Yorkshire Lane'),
(13, 'Beach Volleyball Championship', '2017-08-07', 'Adams Lane, 8th Street'),
(14, 'Fruit Market Grand Opening', '2021-10-03', 'Boulevard, 11th Street'),
(15, 'Northern Lights Viewing Party', '2022-07-08', 'Six Cross, 9th Street');

-- Booking table:

CREATE TABLE IF NOT EXISTS Booking(

```
bookingID INT NOT NULL unique,

customerID INT NOT NULL,

photographerID INT NOT NULL ,

dateBooked date,

eventID int,

bookingStatusVARCHAR(20) NOT NULL,

totalCost double NOT NULL,

PRIMARY KEY (bookingID),

FOREIGN KEY(customerID) REFERENCES Customer(customerID),

FOREIGN KEY(photographerID) REFERENCES Photographer(photographerID) ,

FOREIGN KEY(eventID) REFERENCES Event(eventID)

);
```

-- **Insertion values for booking table :**

```
INSERT INTO Booking (bookingID, customerID, photographerID, dateBooked, eventID,
bookingStatus,totalCost) VALUES
(100, 1, 1001, '2018-12-01', 1, 'booked',23000.00),
(101, 6,1002, '2018-11-01', 2,'pending', 10000.00),
(102, 3, 1003,'2020-01-01',3, 'withdrawn',23000.00),
(103, 4, 1004,'2019-02-01',4, 'booked',22000.00),
(104, 5,1005,'2012-02-10', 5, 'booked',19000.00),
(105, 6,1006,'2008-11-19',6, 'withdrawan',15000.00),
(106, 7,1007,'2012-08-06', 7, 'pending',16000.00),
(107, 8,1008,'2015-12-04', 8, 'pending',25000.00),
(108, 9,1009,'2020-05-10', 9,  'booked',19000.00),
(109, 10,1010,'2021-08-24',10, 'booked',20000.00),
(110,4,1011, '2016-09-11', 11, 'booked',17000.00),
(111,9,1012, '2018-12-17', 12, 'booked',14000.00),
(112,10,1013,'2017-08-07',13, 'pending',29000.00),
(113,1,1014, '2021-10-03', 14, 'pending',26000.00),
(114,9,1015, '2022-07-08', 15, 'withdrawn',28000.00),
(115,5,1005, '2011-06-09', 11, 'pending',30000.00);
```

-- **BookingCostBreakup table**

```
CREATE TABLE IF NOT EXISTS BookingCostBreakup (

bookingID INT UNIQUE NOT NULL,

photographerFee DOUBLE NOT NULL,

platformCommission DOUBLE NOT NULL,

    FOREIGN KEY (bookingID) REFERENCES Booking(bookingID)

);
```

-- **Insertion values for BookingCostBreakup table**

```
INSERT INTO BookingCostBreakup (bookingID, photographerFee, platformCommission)
VALUES
(100, 20000.00, 3000.00),
(101, 8000.00, 2000.00),
(102, 20000.00, 3000.00),
(103, 18000.00, 4000.00),
(104, 15000.00, 4000.00),
(105, 12000.00, 3000.00),
(106, 13000.00, 3000.00),
(107, 20000.00, 5000.00),
(108, 15000.00, 4000.00),
(109, 18000.00, 4000.00),
(110, 15000.00, 2000.00),
(111, 12000.00, 2000.00),
(112, 25000.00, 4000.00),
(113, 22000.00, 4000.00),
(114, 25000.00, 3000.00),
(115, 27000.00, 3000.00);
```

-- Client Interaction Table

```
CREATE TABLE ClientInteraction (

interactionID INT AUTO_INCREMENT PRIMARY KEY,

photographerID INT,

customerID INT,
```

```
    comments TEXT,
    rating INT,
interactionDate DATE,
    FOREIGN KEY (photographerID) REFERENCES Photographer(photographerID),
    FOREIGN KEY (customerID) REFERENCES Customer(customerID)
);
```

```
-- Inserting values into ClientInteraction table
INSERT INTO ClientInteraction (photographerID, customerID, comments, rating,
interactionDate) VALUES
(1001, 1, 'Great experience!', 5, '2023-01-15'),
(1002, 2, 'Beautiful shots!', 4, '2023-02-20'),
(1003, 3, 'Loved the creativity!', 4, '2023-03-10'),
(1004, 4, 'Professional service', 5, '2023-04-05'),
(1005, 5, 'Amazing work!', 5, '2023-05-12'),
(1006, 6, 'Stunning photographs', 4, '2023-06-18'),
(1007, 7, 'Very accommodating', 3, '2023-07-22'),
(1008, 8, 'Captured the essence perfectly', 5, '2023-08-30'),
(1009, 9, 'Exceeded expectations', 4, '2023-09-14'),
(1010, 10, 'Highly recommended', 5, '2023-10-25');
```

-- Review table

```
CREATE TABLE IF NOT EXISTS Review(
reviewID INT PRIMARY KEY,
photographerID INT,
rating INT,
reviewComments TEXT,
dateOfReview DATE,
```

```
FOREIGN KEY (photographerID) REFERENCES Photographer(photographerID));

-- Insertion into Review Table
INSERT INTO Review (reviewID, photographerID, rating, reviewComments, dateOfReview)
VALUES
(1, 1001, 4, 'Great photography skills!', '2023-10-15'),
(2, 1002, 5, 'Very professional and creative.', '2023-11-02'),
(3, 1003, 3, 'Satisfactory service.', '2023-12-05'),
(4, 1004, 4, 'Great photography skills!', '2023-10-15'),
(5, 1005, 5, 'Very professional and creative.', '2023-11-02'),
(6, 1006, 3, 'Not so good ,not satisfactory service.', '2023-12-05'),
(7, 1007, 3, 'Satisfactory service.', '2023-12-05');
```

-- Auction Table

```
CREATE TABLE IF NOT EXISTS Auction (

auctionID INT AUTO_INCREMENT PRIMARY KEY,

photographerID INT,

imageID INT,

bidAmount DOUBLE,

winnerNameVARCHAR(75),

   FOREIGN KEY (photographerID) REFERENCES Photographer(PhotographerID),

   FOREIGN KEY (imageID) REFERENCES Photos(imageID)

);

 -- Insert Data
INSERT INTO Auction (auctionID, photographerID, imageID, bidAmount, winnerName)
VALUES
(1001, 1001, 4, 1800.00, 'Aarav Patel'),
(1002, 1002, 9, 1400.00, 'Advait Sharma'),
(1003, 1003, 10, 1000.00, 'Ishaan Gupta'),
(1004, 1004, 1, 1500.00, 'Jiya Kumar'),
(1005, 1005, 9, 1800.00, 'Kabir Singh'),
(1006, 1006, 5, 1400.00, 'Leela Kapoor'),
(1007, 1007, 7, 1200.00, 'Meera Joshi'),
(1008, 1008, 3, 1500.00, 'Neha Malhotra'),
(1009, 1009, 4, 1000.00, 'Om Sharma'),
```

(1010, 1010, 6, 1200.00, 'Prisha Patel'),
(1011, 1011, 2, 1300.00, 'Aryan Kapoor'),
(1012, 1012, 8, 1100.00, 'Vidhi Malhotra'),
(1013, 1014, 3, 1200.00, 'Rajat Kumar'),
(1014, 1002, 7, 1600.00, 'Anaya Kapoor'),
(1015, 1003, 6, 1400.00, 'Rohan Gupta'),
(1016, 1004, 10, 1800.00, 'Ayesha Sharma'),
(1017, 1015, 1, 2000.00, 'Vivaan Kumar'),
(1018, 1013, 5, 1700.00, 'Arjun Singh'),
(1019, 1007, 9, 1500.00, 'Kiara Malhotra'),
(1020, 1008, 4, 1300.00, 'Zara Malhotra');

-- Metadata Table

CREATE TABLE Metadata (

imageID INT,

imageSize INT,

imageTypeVARCHAR(50),

creationDate DATE,

resolution VARCHAR(50),

PRIMARY KEY (imageID),

FOREIGN KEY (imageID) REFERENCES Photos(imageID)

);

-- -- **Insert Data Metadata Table**
INSERT INTO Metadata (imageID, imageSize, imageType, creationDate, resolution)
VALUES
(1, 1024, 'JPEG', '2023-11-01', '1920x1080'),
(2, 2048, 'PNG', '2023-10-25', '2560x1440'),
(3, 1500, 'JPEG', '2023-11-05', '1920x1080'),
(4, 1200, 'JPEG', '2023-10-30', '1600x900'),
(5, 1800, 'PNG', '2023-11-10', '1920x1080'),
(6, 1600, 'JPEG', '2023-10-28', '1920x1080'),
(7, 2000, 'JPEG', '2023-11-02', '2560x1440'),
(8, 1600, 'PNG', '2023-11-08', '1920x1080'),
(9, 1400, 'JPEG', '2023-10-27', '1600x900'),
(10, 1200, 'JPEG', '2023-11-12', '1600x900'),

(11, 1800, 'PNG', '2023-11-12', '1920x1080'),
(12, 1500, 'JPEG', '2023-11-15', '1920x1080'),
(13, 2000, 'JPEG', '2023-11-18', '2560x1440'),
(14, 1600, 'PNG', '2023-11-20', '1920x1080'),
(15, 1400, 'JPEG', '2023-11-22', '1600x900'),
(16, 1200, 'JPEG', '2023-11-25', '1600x900'),
(17, 1800, 'JPEG', '2023-11-28', '1920x1080'),
(18, 1500, 'JPEG', '2023-11-30', '1920x1080'),
(19, 2000, 'PNG', '2023-12-02', '2560x1440'),
(20, 1600, 'JPEG', '2023-12-05', '1920x1080'),
(21, 1400, 'JPEG', '2023-12-08', '1600x900'),
(22, 1200, 'PNG', '2023-12-10', '1600x900'),
(23, 1800, 'JPEG', '2023-12-12', '1920x1080'),
(24, 1500, 'JPEG', '2023-12-15', '1920x1080'),
(25, 2000, 'JPEG', '2023-12-18', '2560x1440'),
(26, 1600, 'PNG', '2023-12-20', '1920x1080'),
(27, 1400, 'JPEG', '2023-12-22', '1600x900'),
(28, 1200, 'JPEG', '2023-12-25', '1600x900'),
(29, 1800, 'JPEG', '2023-12-28', '1920x1080'),
(30, 1500, 'JPEG', '2023-12-30', '1920x1080'),
(31, 2000, 'PNG', '2024-01-02', '2560x1440'),
(32, 1600, 'JPEG', '2024-01-05', '1920x1080'),
(33, 1400, 'JPEG', '2024-01-08', '1600x900'),
(34, 1200, 'PNG', '2024-01-10', '1600x900'),
(35, 1800, 'JPEG', '2024-01-12', '1920x1080'),
(36, 1500, 'JPEG', '2024-01-15', '1920x1080'),
(37, 2000, 'JPEG', '2024-01-18', '2560x1440'),
(38, 1600, 'PNG', '2024-01-20', '1920x1080'),
(39, 1400, 'JPEG', '2024-01-22', '1600x900'),
(40, 1200, 'JPEG', '2024-01-25', '1600x900');

**********COMPLEX QUERIES**********

use PhotoMaker;

-- **Query 1: Retrieve photographers specializing in 'Wedding' or 'Fashion'.**

```
-- Query 1: Retrieve photographers specializing in 'Wedding' or 'Fashion'.
SELECT p.photographerID, p.photographerFirstName, p.photographerLastName, sc.specialityName FROM Photographer p
JOIN PhotographerSpecialty ps ON p.photographerID = ps.photographerID
join SpecialityCode sc ON ps.specialityCode=sc.specialityCode
WHERE sc.specialityName IN ('Wedding', 'Fashion');
```

photographerID	photographerFirstName	photographerLastName	specialityName
1004	Janice	Galvin	Fashion
1007	Elizabeth	Hallmark	Fashion
1013	Michael	Viescas	Fashion
1015	Brannon	Jones	Fashion
1001	Kerry	Patterson	Wedding
1002	David	Hamilton	Wedding
1005	Doris	Hartwig	Wedding
1010	Marianne	Wier	Wedding

-- **Query 2: Retrieve the latest portfolio added by each photographer**.

```
-- Query 2: Retrieve the latest portfolio added by each photographer.
SELECT p1.photographerFirstName, p2.portfolioName, MAX(p2.createdDate) as LatestPortfolio
FROM Photographer p1
LEFT JOIN Portfolio p2 ON p1.photographerID = p2.photographerID
GROUP BY p1.photographerFirstName, p2.portfolioName;
```

photographerFirstName	portfolioName	LatestPortfolio
Kerry	PF1234	2010-12-10 18:36:20
Kerry	PF1249	2024-03-05 10:00:35
David	PF1235	2021-11-03 11:36:22
David	PF1250	2024-05-17 21:25:50
Betsy	PF1236	2021-09-14 12:46:34
Janice	PF1237	2022-09-10 01:34:40
Doris	PF1238	2021-10-22 08:45:50
Scott	PF1239	2022-04-09 10:36:10
Elizabeth	PF1240	2022-06-15 14:20:45
Sara	PF1241	2022-08-20 09:55:30
Karen	PF1242	2022-12-05 16:40:15
Marianne	PF1243	2023-02-18 22:10:55
John	PF1244	2023-04-30 08:05:10
Sarah	PF1245	2023-07-12 19:30:25
Michael	PF1246	2023-09-24 07:15:40
Kendra	PF1247	2023-11-08 13:50:05
Brannon	PF1248	2024-01-22 18:25:20

-- Query 3: Calculate the total project cost of bookings for each photographer.

```
-- Query 3: Calculate the total project cost of bookings for each photographer.
SELECT p.photographerFirstName,p.photographerLastName, SUM(b.totalCost) AS TotalProjectCost FROM Booking b
JOIN Photographer p ON b.photographerID = p.photographerID
GROUP BY b.photographerID;
```

photographerFirstName	photographerLastName	TotalProjectCost
Kerry	Patterson	23000
David	Hamilton	10000
Betsy	Stadick	23000
Janice	Galvin	22000
Doris	Hartwig	49000
Scott	Bishop	15000
Elizabeth	Hallmark	16000
Sara	Sheskey	25000
Karen	Smith	19000
Marianne	Wier	20000
John	Kennedy	17000
Sarah	Thompson	14000
Michael	Viescas	29000
Kendra	Bonnicksen	26000
Brannon	Jones	28000

-- Query 4: List photographers who have bookings marked as 'withdrawn'.

```sql
-- Query 4: List photographers who have bookings marked as 'withdrawn'.
SELECT p.photographerID, p.photographerFirstName, p.photographerLastName FROM Photographer p
JOIN Booking b ON p.photographerID = b.photographerID WHERE b.bookingStatus = 'withdrawn';
```

photographerID	photographerFirstName	photographerLastName
1003	Betsy	Stadick
1015	Brannon	Jones

-- Query 5 Self Join: Photographers who share the same state

```sql
-- Query 5 Self Join: Photographers who share the same state
SELECT p1.photographerFirstName AS Photographer1,
       p1.photographerLastName AS Photographer1LastName,
       p2.photographerFirstName AS Photographer2,
       p2.photographerLastName AS Photographer2LastName,
       s.stateName AS State
FROM Photographer p1
JOIN Photographer p2 ON p1.photographerID != p2.photographerID AND p1.photographerState = p2.photographerState
JOIN State s ON p1.photographerState = s.stateCode;
```

Photographer1	Photographer1LastNa...	Photographe...	Photographer2LastNa...	State
Kerry	Patterson	Sarah	Thompson	Texas
David	Hamilton	Janice	Galvin	Washington
David	Hamilton	Doris	Hartwig	Washington
David	Hamilton	Elizabeth	Hallmark	Washington
David	Hamilton	Marianne	Wier	Washington
David	Hamilton	Michael	Viescas	Washington
David	Hamilton	Kendra	Bonnicksen	Washington
Betsy	Stadick	Brannon	Jones	California
Janice	Galvin	David	Hamilton	Washington
Janice	Galvin	Doris	Hartwig	Washington
Janice	Galvin	Elizabeth	Hallmark	Washington
Janice	Galvin	Marianne	Wier	Washington
Janice	Galvin	Michael	Viescas	Washington
Janice	Galvin	Kendra	Bonnicksen	Washington
Doris	Hartwig	David	Hamilton	Washington
Doris	Hartwig	Janice	Galvin	Washington
Doris	Hartwig	Elizabeth	Hallmark	Washington
Doris	Hartwig	Marianne	Wier	Washington
Doris	Hartwig	Michael	Viescas	Washington
Doris	Hartwig	Kendra	Bonnicksen	Washington
Scott	Bishop	Sara	Sheskey	Oregon
Scott	Bishop	Karen	Smith	Oregon
Scott	Bishop	John	Kennedy	Oregon
Elizabeth	Hallmark	David	Hamilton	Washington
Elizabeth	Hallmark	Janice	Galvin	Washington
Elizabeth	Hallmark	Doris	Hartwig	Washington
Elizabeth	Hallmark	Marianne	Wier	Washington
Elizabeth	Hallmark	Michael	Viescas	Washington
Elizabeth	Hallmark	Kendra	Bonnicksen	Washington
Sara	Sheskey	Scott	Bishop	Oregon
Sara	Sheskey	Karen	Smith	Oregon
Sara	Sheskey	John	Kennedy	Oregon
Karen	Smith	Scott	Bishop	Oregon
Karen	Smith	Sara	Sheskey	Oregon
Karen	Smith	John	Kennedy	Oregon
Marianne	Wier	David	Hamilton	Washington
Marianne	Wier	Janice	Galvin	Washington
Marianne	Wier	Doris	Hartwig	Washington
Marianne	Wier	Elizabeth	Hallmark	Washington

Result 19

-- Query 6: Retrieve customers who have a 'Basic' subscription and their contact details.

```
-- Query 6: Retrieve customers who have a 'Basic' subscription and their contact details.
SELECT c.customerID, c.customerFname, c.customerLname, c.contact FROM Customer c
JOIN User u ON c.customerID = u.customerID
JOIN Subscription s ON u.userSubscriptionCode = s.subscriptionCode
WHERE s.subscriptionType = 'Basic';
```

customerID	customerFname	customerLname	contact
1	Nikhita	Mandava	9084948476
4	Firoz	Rehman	9081002002
7	William	Thompson	9012390000

-- Query 7: List customers who have not made any bookings.

```sql
--  Query 7: List customers who have not made any bookings.
SELECT c.customerID, c.customerFname, c.customerLname FROM Customer c
LEFT JOIN Booking b ON c.customerID = b.customerID
WHERE b.bookingID IS NULL;
```

customerID	customerFname	customerLname
2	Sanjana	Sharma
11	Rachel	Green
12	Brown	Waine
13	Betsy	Stadick

-- Query 8: Count the number of interactions made by each customer.

```sql
--  Query 8: Count the number of interactions made by each customer.
SELECT c.customerFname,c.customerLname, COUNT(ci.interactionID) AS InteractionCount
FROM ClientInteraction ci
join Customer c ON ci.customerID=c.customerID
GROUP BY ci.customerID;
```

customerFname	customerLname	InteractionCount
Nikhita	Mandava	1
Sanjana	Sharma	5
Ranjith	Komra	1
Firoz	Rehman	1
Raha	Malhotra	3
Doris	Hartwig	3
William	Thompson	1
Gary	Hallmark	1
Raj	Kumar	1
Andrea	Bullock	1

-- Query 9.Retrieve the most booked photographer's name and the number of bookings they have received.

#using joins

```sql
-- Query 9. Retrieve the most booked photographer's name and the number of bookings they have received.

#using joins
SELECT p.photographerFirstName, COUNT(b.bookingID) AS total_bookings
FROM Photographer p
JOIN Booking b ON p.photographerID = b.photographerID
GROUP BY p.photographerID
ORDER BY total_bookings DESC
LIMIT 1;
```

#using subqueries

```sql
#using subqueries
SELECT photographerFirstName,
(SELECT COUNT(bookingID)
FROM Booking
WHERE photographerID = p.photographerID) AS total_bookings
FROM Photographer p
ORDER BY total_bookings DESC
LIMIT 1;
```

photographerFirstName	total_bookin...
Doris	2

-- Query 10: Display metadata details of images uploaded by photographers.

```sql
--  Query 10: Display metadata details of images uploaded by photographers.
SELECT m.*, p.PhotographerID FROM Metadata m
INNER JOIN Photos p ON m.imageID = p.imageID;
```

imageID	imageSize	imageType	creationDate	resolution	PhotographerID
1	1024	JPEG	2023-11-01	1920x1080	1001
2	2048	PNG	2023-10-25	2560x1440	1001
3	1500	JPEG	2023-11-05	1920x1080	1001
4	1200	JPEG	2023-10-30	1600x900	1002
5	1800	PNG	2023-11-10	1920x1080	1002
6	1600	JPEG	2023-10-28	1920x1080	1002
7	2000	JPEG	2023-11-02	2560x1440	1003
8	1600	PNG	2023-11-08	1920x1080	1003
9	1400	JPEG	2023-10-27	1600x900	1003
10	1200	JPEG	2023-11-12	1600x900	1003
11	1800	PNG	2023-11-12	1920x1080	1004
12	1500	JPEG	2023-11-15	1920x1080	1004
13	2000	JPEG	2023-11-18	2560x1440	1004
14	1600	PNG	2023-11-20	1920x1080	1005
15	1400	JPEG	2023-11-22	1600x900	1005
16	1200	JPEG	2023-11-25	1600x900	1005
17	1800	JPEG	2023-11-28	1920x1080	1006
18	1500	JPEG	2023-11-30	1920x1080	1006
19	2000	PNG	2023-12-02	2560x1440	1006
20	1600	JPEG	2023-12-05	1920x1080	1007
21	1400	JPEG	2023-12-08	1600x900	1007
22	1200	PNG	2023-12-10	1600x900	1007
23	1800	JPEG	2023-12-12	1920x1080	1008
24	1500	JPEG	2023-12-15	1920x1080	1008
25	2000	JPEG	2023-12-18	2560x1440	1009
26	1600	PNG	2023-12-20	1920x1080	1009
27	1400	JPEG	2023-12-22	1600x900	1010
28	1200	JPEG	2023-12-25	1600x900	1010
29	1800	JPEG	2023-12-28	1920x1080	1011
30	1500	JPEG	2023-12-30	1920x1080	1011
31	2000	PNG	2024-01-02	2560x1440	1012
32	1600	JPEG	2024-01-05	1920x1080	1012
33	1400	JPEG	2024-01-08	1600x900	1013
34	1200	PNG	2024-01-10	1600x900	1013
35	1800	JPEG	2024-01-12	1920x1080	1014
36	1500	JPEG	2024-01-15	1920x1080	1014
37	2000	JPEG	2024-01-18	2560x1440	1015
38	1600	PNG	2024-01-20	1920x1080	1015
39	1400	JPEG	2024-01-22	1600x900	1004

Result 24

-- Query 11: List all clients who have not booked any photographers.

#using joins

And

#using sub queries

```
-- Query 11.    List all clients who have not booked any photographers.
#using joins
SELECT c.customerID, c.customerFname,c.customerLname
FROM Customer c
LEFT JOIN Booking b ON c.customerID = b.customerID
WHERE b.customerID IS NULL;

#using sub queries
SELECT customerID, customerFname, customerLname
FROM Customer
WHERE customerID NOT IN (
    SELECT DISTINCT customerID
    FROM Booking
    WHERE customerID IS NOT NULL
);
```

customerID	customerFname	customerLname
2	Sanjana	Sharma
11	Rachel	Green
12	Brown	Waine
13	Betsy	Stadick

-- Query 12. Retrieve the names of photographers who have booked more than

one clients in the last month, along with the total number of bookings for each photographer.

```
-- Query 12. Retrieve the names of photographers who have booked more than
-- five clients in the last month, along with the total number of bookings for each photographer.

SELECT p.photographerFirstName, p.photographerLastName, COUNT(b.bookingID) AS totalBookings
FROM Photographer p
JOIN Booking b ON p.photographerID = b.photographerID
WHERE b.dateBooked >= CURDATE() - INTERVAL 1 MONTH
GROUP BY p.photographerID, p.photographerFirstName, p.photographerLastName
HAVING totalBookings > 1;
```

photographerFirstName	photographerLastName	totalBookings
John	Kennedy	5

********** PROCEDURES **********

-- **Procedure to Retrieve Customer Information**

```
-- Procedure to Retrieve Customer Information
DELIMITER //
CREATE PROCEDURE GetCustomerInfo (IN custID INT)
BEGIN
   SELECT * FROM Customer WHERE customerID = custID;
END //
DELIMITER ;
```

```
854 •  call GetCustomerInfo(1);
855
```

100% 25:854

Result Grid | Filter Rows: Search | Export:

customerID	customerFname	customerLna...	contact	customerStre...	customerCi...	customerSta...	customerZipco...	customerEmail	customerDOB	gender	customerPreferenc...	customerInterest
1	Nikhita	Mandava	9084948476	3000 Northside	Richardson	TX	75080	abc@gmail.com	1976-12-01	F	Budget save	Wedding Photography

-- **Procedure to Retrieve Photographer Information**

```
-- Procedure to Retrieve Photographer Information
DELIMITER //
CREATE PROCEDURE GetPhotographerInfo (IN photogID INT)
BEGIN
   SELECT * FROM Photographer WHERE PhotographerID = photogID;
END //
DELIMITER ;
```

```
858 •  call GetPhotographerInfo(1006);
```

100% 25:854

Result Grid Filter Rows: Search Export:

photographer...	photographerFirstName	photographerLastName	photographerStreetAddr...	photographerC...	photographerSt...	photographerZipCo...	photographerEmail	photographerSocialMediaLi...	photographerI
1006	Scott	Bishop	66 Spring Valley Drive	Medford	OR	97501	ScottBishop@gmail.com	www.linkedin.com/ScottBishop	1234

-- **Procedure to Retrieve Booking Information by Customer**

```
-- Procedure to Retrieve Booking Information by Customer
DELIMITER //
CREATE PROCEDURE GetBookingsByCustomer (IN custID INT)
BEGIN
   SELECT * FROM Booking WHERE customerID = custID;
END //
DELIMITER ;
```

```
860 •   call GetBookingsByCustomer(6);
861
```

100% 32:858

Result Grid Filter Rows: Search Export:

bookingID	customerID	photographer...	dateBooked	eventID	bookingStat...	totalCost
101	6	1002	2018-11-01	2	pending	10000
105	6	1006	2008-11-19	6	withdrawan	15000
117	6	1011	2023-11-30	2	pending	18000

This function gets the total revenue of a photographer

```
### This function gets the total revenue of a photographer
DELIMITER //
CREATE FUNCTION CalculatePhotographerRevenue (photogID INT) RETURNS DOUBLE
BEGIN
  DECLARE totalRevenue DOUBLE;
  SELECT COALESCE(SUM(bookingCostBreakup.photographerFee + bookingCostBreakup.platformCommission)) INTO totalRevenue
  FROM Booking
  LEFT JOIN BookingCostBreakup ON Booking.bookingID = BookingCostBreakup.bookingID
  WHERE Booking.photographerID = photogID;
  RETURN totalRevenue;
END //
DELIMITER ;
```

```
701 •   select CalculatePhotographerRevenue(1001) as Revenue;
```

100% | 1:702

Result Grid | Filter Rows: Search | Export:

Revenue
23000

This function tells if the photographer is available for booking on that specified date

```
### This function tells if the photographer is available for booking on that specified date
DELIMITER //
CREATE FUNCTION IsPhotographerAvailable (photogID INT, bookingDate DATE) RETURNS BOOLEAN
BEGIN
  DECLARE available BOOLEAN;
  SELECT COUNT(*) = 0 INTO available
  FROM Booking
  WHERE photographerID = photogID AND dateBooked = bookingDate;
  RETURN available;
END //
DELIMITER ;
```

```
717 •   select IsPhotographerAvailable(11, '2023-12-30') as Availability;
```

100% 1:719

Result Grid Filter Rows: Search Export:

Availability
1

Gets a photographers latest booking date

```
#####  Gets a photographers latest booking date
DELIMITER //
CREATE FUNCTION GetCustomerLatestBookingDate (custID INT) RETURNS DATE
BEGIN
  DECLARE latestBookingDate DATE;
  SELECT MAX(dateBooked) INTO latestBookingDate
  FROM Booking
  WHERE customerID = custID;
  RETURN latestBookingDate;
END //
DELIMITER ;
```

```
734 •   select GetCustomerLatestBookingDate(10) as LatestBooking;
```

100% 1:735

Result Grid Filter Rows: Search Export:

LatestBooking
2021-08-24

Rating for photographers latest work

```
#####  Gets a photographers latest booking date
DELIMITER //
CREATE FUNCTION GetCustomerLatestBookingDate (custID INT) RETURNS DATE
BEGIN
  DECLARE latestBookingDate DATE;
  SELECT MAX(dateBooked) INTO latestBookingDate
  FROM Booking
  WHERE customerID = custID;
  RETURN latestBookingDate;
END //
DELIMITER ;
```

```
734 •   select GetCustomerLatestBookingDate(10) as LatestBooking;
```

100% 1:735

Result Grid Filter Rows: Search Export:

LatestBooking
2021-08-24

Function 5

```
###  Rating for photographers latest work
DELIMITER //
CREATE FUNCTION GetPhotographerLatestWorksRating (photogID INT, reviewCount INT) RETURNS DECIMAL(3,2)
BEGIN
  DECLARE avgRating DECIMAL(3,2);

  SELECT AVG(r.rating) INTO avgRating
  FROM Review r
  JOIN Photographer p ON r.photographerID = p.photographerID
  WHERE r.photographerID = photogID
  ORDER BY r.dateOfReview DESC
  LIMIT reviewCount;
  RETURN IFNULL(avgRating, 0.00);
END //
DELIMITER ;
```

```
752 •   select GetPhotographerLatestWorksRating(1003,3) as LatestRating;
```

100% 1:753

Result Grid Filter Rows: Search Export:

LatestRating
3.00

********** TRIGGERS **********

- Automatically set default values on inserts
- Logging history of changes to critical fields
- Cascading updates/deletes
- Validation constraints and error handling

-- **TRIGGER TO PREVENT DELETION OF CUSTOMER WITH ACTIVE BOOKING**

```
DELIMITER $$

CREATE TRIGGER restrict_customer_delete

BEFORE DELETE ON Customer

FOR EACH ROW

BEGIN

  IF (SELECT COUNT(*) FROM Booking WHERE customerID = OLD.customerID) > 0 THEN

    SIGNAL SQLSTATE '45000'

      SET MESSAGE_TEXT = 'Cannot delete customer with active bookings';

  END IF;

END$$

DELIMITER ;
```

-- **TRIGGER TO SET STATUS AS ACTIVE FOR NEW BOOKING**

```
DELIMITER $$

CREATE TRIGGER booking_status_trigger

AFTER INSERT ON Booking
```

```
FOR EACH ROW

BEGIN

  UPDATE Booking

  SET bookingStatus = 'Active'

  WHERE bookingID = NEW.bookingID;

END$$

DELIMITER ;
```

-- **TRIGGER TO PREVENT CUSTOMER UPDATE**

```
DELIMITER $$

CREATE TRIGGER prevent_customer_update

BEFORE UPDATE ON Customer

FOR EACH ROW

BEGIN

  IF NEW.customerID<>OLD.customerID THEN

    SIGNAL SQLSTATE '45000'

      SET MESSAGE_TEXT = 'Cannot update customerID';

  END IF;

END$$

DELIMITER;

DELIMITER $$

create table booking_audit_log (bookingID integer, time_record date, log varchar(20),foreign
key(bookingID) references booking (bookingID) on delete cascade);
```

```
DELIMITER $$
```

##Auditing the log history of Bookings

```
CREATE TRIGGER IF NOT EXISTS booking_update_log
AFTER UPDATE ON Booking
FOR EACH ROW
BEGIN
   INSERT INTO booking_audit_log (bookingID, time_record, log)
   VALUES (OLD.bookingID, NOW(), CONCAT('RECORD UPDATE : ', NEW.bookingStatus));
END

DELIMITER $$
CREATE TRIGGER IF NOT EXISTS booking_delete_log
AFTER DELETE ON Booking
FOR EACH ROW
BEGIN
   INSERT INTO booking_audit_log (bookingID, time_record, log)
   VALUES (OLD.bookingID, NOW(), 'DELETED');
END
DELIMITER //
```