create database hotel\_reservation\_system;

use hotel\_reservation\_system;

Here are three tables you might consider creating for a Hotel Reservation System, each with various date and datetime columns:

1. **Table: Guests**

This table will store information about the guests who are making reservations.

**CREATE TABLE Guests (**

**guest\_id INT PRIMARY KEY,**

**first\_name VARCHAR(50),**

**last\_name VARCHAR(50),**

**email VARCHAR(100),**

**phone\_number VARCHAR(15),**

**registration\_date DATETIME,**

**date\_of\_birth DATE**

**);**

-- Inserting data into the Guests table

INSERT INTO Guests (guest\_id, first\_name, last\_name, email, phone\_number, registration\_date, date\_of\_birth)

VALUES

(1, 'John', 'Doe', 'john.doe@example.com', '555-123-4567', '2023-09-01 08:00:00', '1990-05-15'),

(2, 'Jane', 'Smith', 'jane.smith@example.com', '555-987-6543', '2023-09-01 09:15:00', '1985-12-10'),

(3, 'Michael', 'Johnson', 'michael.j@example.com', '555-567-8901', '2023-09-01 10:30:00', '1995-03-22'),

(4, 'Emily', 'Brown', 'emily.b@example.com', '555-234-5678', '2023-09-01 11:45:00', '1982-07-04'),

(5, 'William', 'Anderson', 'william.a@example.com', '555-789-1234', '2023-09-01 13:00:00', '1998-09-18'),

(6, 'Olivia', 'Davis', 'olivia.d@example.com', '555-345-6789', '2023-09-01 14:15:00', '1993-11-30'),

(7, 'James', 'Wilson', 'james.w@example.com', '555-876-5432', '2023-09-01 15:30:00', '1989-04-12'),

(8, 'Sophia', 'Lee', 'sophia.l@example.com', '555-432-1098', '2023-09-01 16:45:00', '1997-01-25'),

(9, 'Benjamin', 'Martinez', 'benjamin.m@example.com', '555-210-9876', '2023-09-01 18:00:00', '1991-08-08'),

(10, 'Ava', 'Harris', 'ava.h@example.com', '555-987-6543', '2023-09-01 19:15:00', '1994-06-28');

-- Inserting 10 more rows of data into the Guests table

INSERT INTO Guests (guest\_id, first\_name, last\_name, email, phone\_number, registration\_date, date\_of\_birth)

VALUES

(11, 'Liam', 'Garcia', 'liam.g@example.com', '555-123-7890', '2023-09-02 08:00:00', '1996-10-17'),

(12, 'Charlotte', 'Smith', 'charlotte.s@example.com', '555-555-5555', '2023-09-02 09:15:00', '1987-09-02'),

(13, 'Mason', 'Wang', 'mason.w@example.com', '555-111-2222', '2023-09-02 10:30:00', '1992-02-25'),

(14, 'Emma', 'Brown', 'emma.b@example.com', '555-999-8888', '2023-09-02 11:45:00', '1999-03-12'),

(15, 'Noah', 'Jones', 'noah.j@example.com', '555-777-6666', '2023-09-02 13:00:00', '1994-07-29'),

(16, 'Olivia', 'Martinez', 'olivia.m@example.com', '555-333-4444', '2023-09-02 14:15:00', '1988-12-05'),

(17, 'Ethan', 'Miller', 'ethan.m@example.com', '555-222-3333', '2023-09-02 15:30:00', '1993-04-20'),

(18, 'Ava', 'Johnson', 'ava.j@example.com', '555-888-9999', '2023-09-02 16:45:00', '1990-06-14'),

(19, 'Liam', 'Davis', 'liam.d@example.com', '555-444-5555', '2023-09-02 18:00:00', '1986-08-31'),

(20, 'Sophia', 'Smith', 'sophia.s@example.com', '555-666-7777', '2023-09-02 19:15:00', '1995-01-10');

-- Inserting 10 more rows of data into the Guests table

INSERT INTO Guests (guest\_id, first\_name, last\_name, email, phone\_number, registration\_date, date\_of\_birth)

VALUES

(21, 'William', 'Johnson', 'william.j@example.com', '555-123-4567', '2023-09-03 08:00:00', '1998-09-15'),

(22, 'Olivia', 'Garcia', 'olivia.g@example.com', '555-987-6543', '2023-09-03 09:15:00', '1993-11-20'),

(23, 'James', 'Lee', 'james.l@example.com', '555-567-8901', '2023-09-03 10:30:00', '1989-04-02'),

(24, 'Emma', 'Anderson', 'emma.a@example.com', '555-234-5678', '2023-09-03 11:45:00', '1982-07-15'),

(25, 'Liam', 'Harris', 'liam.h@example.com', '555-789-1234', '2023-09-03 13:00:00', '1997-09-18'),

(26, 'Ava', 'Brown', 'ava.b@example.com', '555-345-6789', '2023-09-03 14:15:00', '1994-11-30'),

(27, 'Noah', 'Wilson', 'noah.w@example.com', '555-876-5432', '2023-09-03 15:30:00', '1986-04-12'),

(28, 'Sophia', 'Davis', 'sophia.d@example.com', '555-432-1098', '2023-09-03 16:45:00', '1990-01-25'),

(29, 'Benjamin', 'Smith', 'benjamin.s@example.com', '555-210-9876', '2023-09-03 18:00:00', '1985-08-08'),

(30, 'Charlotte', 'Martinez', 'charlotte.m@example.com', '555-777-8888', '2023-09-03 19:15:00', '1991-06-28');

-- Inserting 10 more rows of data into the Guests table

INSERT INTO Guests (guest\_id, first\_name, last\_name, email, phone\_number, registration\_date, date\_of\_birth)

VALUES

(31, 'Ethan', 'Thompson', 'ethan.t@example.com', '555-123-4567', '2023-09-04 08:00:00', '1999-10-15'),

(32, 'Mia', 'Jackson', 'mia.j@example.com', '555-987-6543', '2023-09-04 09:15:00', '1992-11-20'),

(33, 'Liam', 'Taylor', 'liam.t@example.com', '555-567-8901', '2023-09-04 10:30:00', '1987-04-02'),

(34, 'Olivia', 'Hernandez', 'olivia.h@example.com', '555-234-5678', '2023-09-04 11:45:00', '1983-07-15'),

(35, 'Noah', 'Gonzalez', 'noah.g@example.com', '555-789-1234', '2023-09-04 13:00:00', '1995-09-18'),

(36, 'Emma', 'Lopez', 'emma.l@example.com', '555-345-6789', '2023-09-04 14:15:00', '1990-11-30'),

(37, 'Liam', 'Wright', 'liam.w@example.com', '555-876-5432', '2023-09-04 15:30:00', '1988-04-12'),

(38, 'Ava', 'Perez', 'ava.p@example.com', '555-432-1098', '2023-09-04 16:45:00', '1993-01-25'),

(39, 'James', 'Rodriguez', 'james.r@example.com', '555-210-9876', '2023-09-04 18:00:00', '1996-08-08'),

(40, 'Sophia', 'Gomez', 'sophia.g@example.com', '555-777-8888', '2023-09-04 19:15:00', '1998-06-28');

-- Inserting 10 more rows of data into the Guests table

INSERT INTO Guests (guest\_id, first\_name, last\_name, email, phone\_number, registration\_date, date\_of\_birth)

VALUES

(41, 'Oliver', 'Harrison', 'oliver.h@example.com', '555-123-4567', '2023-09-05 08:00:00', '1997-03-10'),

(42, 'Mia', 'Morales', 'mia.m@example.com', '555-987-6543', '2023-09-05 09:15:00', '1990-08-22'),

(43, 'Elijah', 'Kim', 'elijah.k@example.com', '555-567-8901', '2023-09-05 10:30:00', '1995-05-18'),

(44, 'Sophia', 'Nguyen', 'sophia.n@example.com', '555-234-5678', '2023-09-05 11:45:00', '1984-12-03'),

(45, 'Noah', 'Patel', 'noah.p@example.com', '555-789-1234', '2023-09-05 13:00:00', '1992-02-25'),

(46, 'Ava', 'Reyes', 'ava.r@example.com', '555-345-6789', '2023-09-05 14:15:00', '1999-07-12'),

(47, 'Liam', 'Taylor', 'liam.t@example.com', '555-876-5432', '2023-09-05 15:30:00', '1991-11-30'),

(48, 'Olivia', 'Lopez', 'olivia.l@example.com', '555-432-1098', '2023-09-05 16:45:00', '1998-06-14'),

(49, 'Jackson', 'Hernandez', 'jackson.h@example.com', '555-210-9876', '2023-09-05 18:00:00', '1993-05-28'),

(50, 'Emma', 'Wright', 'emma.w@example.com', '555-777-8888', '2023-09-05 19:15:00', '1986-09-15');

**Table: Rooms**

This table will hold information about the different rooms available in the hotel.

**CREATE TABLE Rooms (**

**room\_id INT PRIMARY KEY,**

**room\_number VARCHAR(10),**

**room\_type VARCHAR(50),**

**max\_occupancy INT,**

**price\_per\_night DECIMAL(10, 2),**

**last\_cleaned\_date DATE,**

**maintenance\_due\_date DATE**

**);**

-- Inserting data into the Rooms table

INSERT INTO Rooms (room\_id, room\_number, room\_type, max\_occupancy, price\_per\_night, last\_cleaned\_date, maintenance\_due\_date)

VALUES

(1, '101', 'Standard', 2, 100.00, '2023-09-01', '2023-10-01'),

(2, '102', 'Standard', 2, 100.00, '2023-09-02', '2023-10-02'),

(3, '201', 'Deluxe', 3, 150.00, '2023-09-03', '2023-10-03'),

(4, '202', 'Deluxe', 3, 150.00, '2023-09-04', '2023-10-04'),

(5, '301', 'Suite', 4, 200.00, '2023-09-05', '2023-10-05'),

(6, '302', 'Suite', 4, 200.00, '2023-09-06', '2023-10-06'),

(7, '401', 'Standard', 2, 100.00, '2023-09-07', '2023-10-07'),

(8, '402', 'Standard', 2, 100.00, '2023-09-08', '2023-10-08'),

(9, '501', 'Deluxe', 3, 150.00, '2023-09-09', '2023-10-09'),

(10, '502', 'Deluxe', 3, 150.00, '2023-09-10', '2023-10-10');

-- Inserting 30 more rows of data into the Rooms table

INSERT INTO Rooms (room\_id, room\_number, room\_type, max\_occupancy, price\_per\_night, last\_cleaned\_date, maintenance\_due\_date)

VALUES

(11, '601', 'Suite', 4, 200.00, '2023-09-11', '2023-10-11'),

(12, '602', 'Suite', 4, 200.00, '2023-09-12', '2023-10-12'),

(13, '701', 'Standard', 2, 100.00, '2023-09-13', '2023-10-13'),

(14, '702', 'Standard', 2, 100.00, '2023-09-14', '2023-10-14'),

(15, '801', 'Deluxe', 3, 150.00, '2023-09-15', '2023-10-15'),

(16, '802', 'Deluxe', 3, 150.00, '2023-09-16', '2023-10-16'),

(17, '901', 'Suite', 4, 200.00, '2023-09-17', '2023-10-17'),

(18, '902', 'Suite', 4, 200.00, '2023-09-18', '2023-10-18'),

(19, '1001', 'Standard', 2, 100.00, '2023-09-19', '2023-10-19'),

(20, '1002', 'Standard', 2, 100.00, '2023-09-20', '2023-10-20'),

(21, '1101', 'Deluxe', 3, 150.00, '2023-09-21', '2023-10-21'),

(22, '1102', 'Deluxe', 3, 150.00, '2023-09-22', '2023-10-22'),

(23, '1201', 'Suite', 4, 200.00, '2023-09-23', '2023-10-23'),

(24, '1202', 'Suite', 4, 200.00, '2023-09-24', '2023-10-24'),

(25, '1301', 'Standard', 2, 100.00, '2023-09-25', '2023-10-25'),

(26, '1302', 'Standard', 2, 100.00, '2023-09-26', '2023-10-26'),

(27, '1401', 'Deluxe', 3, 150.00, '2023-09-27', '2023-10-27'),

(28, '1402', 'Deluxe', 3, 150.00, '2023-09-28', '2023-10-28'),

(29, '1501', 'Suite', 4, 200.00, '2023-09-29', '2023-10-29'),

(30, '1502', 'Suite', 4, 200.00, '2023-09-30', '2023-10-30');

**Table: Reservations**

This table will store the reservations made by guests.

**CREATE TABLE Reservations (**

**reservation\_id INT PRIMARY KEY,**

**guest\_id INT,**

**room\_id INT,**

**check\_in\_date DATETIME,**

**check\_out\_date DATETIME,**

**reservation\_date DATETIME,**

**num\_guests int,**

**total\_amount DECIMAL(10, 2),**

**FOREIGN KEY (guest\_id) REFERENCES Guests(guest\_id),**

**FOREIGN KEY (room\_id) REFERENCES Rooms(room\_id)**

**);**

-- Inserting data into the Reservations table

INSERT INTO Reservations (reservation\_id, guest\_id, room\_id, check\_in\_date, check\_out\_date, reservation\_date, num\_guests, total\_amount)

VALUES

(1, 1, 5, '2023-09-01 14:00:00', '2023-09-03 12:00:00', '2023-08-15 10:00:00', 2, 250.00),

(2, 2, 7, '2023-09-02 15:00:00', '2023-09-05 11:00:00', '2023-08-16 11:30:00', 3, 375.00),

(3, 3, 10, '2023-09-03 12:00:00', '2023-09-06 10:00:00', '2023-08-17 09:45:00', 2, 300.00),

(4, 4, 15, '2023-09-04 14:00:00', '2023-09-07 12:00:00', '2023-08-18 13:15:00', 4, 600.00),

(5, 5, 21, '2023-09-05 16:00:00', '2023-09-08 14:00:00', '2023-08-19 14:30:00', 2, 400.00),

(6, 6, 28, '2023-09-06 10:00:00', '2023-09-09 08:00:00', '2023-08-20 15:45:00', 3, 450.00),

(7, 7, 22, '2023-09-07 14:00:00', '2023-09-10 12:00:00', '2023-08-21 08:30:00', 2, 350.00),

(8, 8, 28, '2023-09-08 11:00:00', '2023-09-11 09:00:00', '2023-08-22 12:00:00', 4, 500.00),

(9, 9, 22, '2023-09-09 13:00:00', '2023-09-12 11:00:00', '2023-08-23 14:45:00', 2, 275.00),

(10, 10, 25, '2023-09-10 15:00:00', '2023-09-13 13:00:00', '2023-08-24 16:30:00', 3, 425.00);

-- Inserting 20 more rows of data into the Reservations table

INSERT INTO Reservations (reservation\_id, guest\_id, room\_id, check\_in\_date, check\_out\_date, reservation\_date, num\_guests, total\_amount)

VALUES

(11, 11, 12, '2023-09-11 14:00:00', '2023-09-13 12:00:00', '2023-09-01 10:00:00', 2, 220.00),

(12, 12, 14, '2023-09-12 15:00:00', '2023-09-15 11:00:00', '2023-09-02 11:30:00', 3, 345.00),

(13, 13, 17, '2023-09-13 12:00:00', '2023-09-16 10:00:00', '2023-09-03 09:45:00', 2, 280.00),

(14, 14, 19, '2023-09-14 14:00:00', '2023-09-17 12:00:00', '2023-09-04 13:15:00', 4, 520.00),

(15, 15, 22, '2023-09-15 16:00:00', '2023-09-18 14:00:00', '2023-09-05 14:30:00', 2, 380.00),

(16, 16, 25, '2023-09-16 10:00:00', '2023-09-19 08:00:00', '2023-09-06 15:45:00', 3, 435.00),

(17, 17, 29, '2023-09-17 14:00:00', '2023-09-20 12:00:00', '2023-09-07 08:30:00', 2, 360.00),

(18, 18, 30, '2023-09-18 11:00:00', '2023-09-21 09:00:00', '2023-09-08 12:00:00', 4, 520.00),

(19, 19, 30, '2023-09-19 13:00:00', '2023-09-22 11:00:00', '2023-09-09 14:45:00', 2, 275.00),

(20, 20, 22, '2023-09-20 15:00:00', '2023-09-23 13:00:00', '2023-09-10 16:30:00', 3, 425.00),

(21, 21, 26, '2023-09-21 14:00:00', '2023-09-24 12:00:00', '2023-09-11 10:00:00', 2, 230.00),

(22, 22, 30, '2023-09-22 15:00:00', '2023-09-25 11:00:00', '2023-09-12 11:30:00', 3, 350.00),

(23, 23, 3, '2023-09-23 12:00:00', '2023-09-26 10:00:00', '2023-09-13 09:45:00', 2, 210.00),

(24, 24, 6, '2023-09-24 14:00:00', '2023-09-27 12:00:00', '2023-09-14 13:15:00', 4, 590.00),

(25, 25, 9, '2023-09-25 16:00:00', '2023-09-28 14:00:00', '2023-09-15 14:30:00', 2, 370.00),

(26, 26, 12, '2023-09-26 10:00:00', '2023-09-29 08:00:00', '2023-09-16 15:45:00', 3, 425.00),

(27, 27, 15, '2023-09-27 14:00:00', '2023-09-30 12:00:00', '2023-09-17 08:30:00', 2, 340.00),

(28, 28, 18, '2023-09-28 11:00:00', '2023-10-01 09:00:00', '2023-09-18 12:00:00', 4, 510.00),

(29, 29, 21, '2023-09-29 13:00:00', '2023-10-02 11:00:00', '2023-09-19 14:45:00', 2, 290.00),

(30, 30, 24, '2023-09-30 15:00:00', '2023-10-03 13:00:00', '2023-09-20 16:30:00', 3, 440.00);

-- Inserting 20 more rows of data into the Reservations table

INSERT INTO Reservations (reservation\_id, guest\_id, room\_id, check\_in\_date, check\_out\_date, reservation\_date, num\_guests, total\_amount)

VALUES

(31, 31, 27, '2023-10-01 14:00:00', '2023-10-04 12:00:00', '2023-09-21 10:00:00', 2, 260.00),

(32, 32, 30, '2023-10-02 15:00:00', '2023-10-05 11:00:00', '2023-09-22 11:30:00', 3, 375.00),

(33, 33, 24, '2023-10-03 12:00:00', '2023-10-06 10:00:00', '2023-09-23 09:45:00', 2, 290.00),

(34, 34, 17, '2023-10-04 14:00:00', '2023-10-07 12:00:00', '2023-09-24 13:15:00', 4, 540.00),

(35, 35, 10, '2023-10-05 16:00:00', '2023-10-08 14:00:00', '2023-09-25 14:30:00', 2, 390.00),

(36, 36, 13, '2023-10-06 10:00:00', '2023-10-09 08:00:00', '2023-09-26 15:45:00', 3, 455.00),

(37, 37, 16, '2023-10-07 14:00:00', '2023-10-10 12:00:00', '2023-09-27 08:30:00', 2, 370.00),

(38, 38, 19, '2023-10-08 11:00:00', '2023-10-11 09:00:00', '2023-09-28 12:00:00', 4, 520.00),

(39, 39, 9, '2023-10-09 13:00:00', '2023-10-12 11:00:00', '2023-09-29 14:45:00', 2, 285.00),

(40, 40, 14, '2023-10-10 15:00:00', '2023-10-13 13:00:00', '2023-09-30 16:30:00', 3, 435.00),

(41, 41, 19, '2023-10-11 14:00:00', '2023-10-14 12:00:00', '2023-10-01 10:00:00', 2, 240.00),

(42, 42, 24, '2023-10-12 15:00:00', '2023-10-15 11:00:00', '2023-10-02 11:30:00', 3, 360.00),

(43, 43, 29, '2023-10-13 12:00:00', '2023-10-16 10:00:00', '2023-10-03 09:45:00', 2, 270.00),

(44, 44, 24, '2023-10-14 14:00:00', '2023-10-17 12:00:00', '2023-10-04 13:15:00', 4, 520.00),

(45, 45, 29, '2023-10-15 16:00:00', '2023-10-18 14:00:00', '2023-10-05 14:30:00', 2, 385.00),

(46, 46, 24, '2023-10-16 10:00:00', '2023-10-19 08:00:00', '2023-10-06 15:45:00', 3, 440.00),

(47, 47, 29, '2023-10-17 14:00:00', '2023-10-20 12:00:00', '2023-10-07 08:30:00', 2, 365.00),

(48, 48, 7, '2023-10-18 11:00:00', '2023-10-21 09:00:00', '2023-10-08 12:00:00', 4, 510.00),

(49, 49, 12, '2023-10-19 13:00:00', '2023-10-22 11:00:00', '2023-10-09 14:45:00', 2, 280.00),

(50, 50, 17, '2023-10-20 15:00:00', '2023-10-23 13:00:00', '2023-10-10 16:30:00', 3, 430.00);

**Query 1: Count the number of reservations made on or after September 1, 2023.**

SELECT COUNT(\*) AS num\_reservations FROM Reservations WHERE reservation\_date >= '2023-09-01';

**Answer:** This query counts the number of reservations that were made on or after September 1, 2023.

**Query 2: Calculate the average length of stay for all reservations.**

SELECT AVG(DATEDIFF(check\_out\_date, check\_in\_date) + 1) AS avg\_length\_of\_stay FROM Reservations;

**Answer:** This query calculates the average length of stay for all reservations by calculating the difference between check-out and check-in dates.

**Query 3: Find the total revenue earned in the month of October 2023.**

SELECT SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(check\_in\_date) = 2023 AND MONTH(check\_in\_date) = 10;

**Answer:** This query calculates the total revenue earned from reservations with check-in dates in October 2023.

**Query 4: Calculate the average number of guests per reservation.**

SELECT AVG(num\_guests) AS avg\_guests\_per\_reservation FROM Reservations;

**Answer:** This query calculates the average number of guests per reservation.

**Query 5: Find the number of rooms that were cleaned more than 10 days ago.**

SELECT COUNT(\*) AS num\_rooms\_to\_clean FROM Rooms WHERE DATEDIFF(NOW(), last\_cleaned\_date) > 10;

**Answer:** This query counts the number of rooms that were last cleaned more than 10 days ago.

**Query 6: Calculate the average price per night for each room type.**

SELECT room\_type, AVG(price\_per\_night) AS avg\_price\_per\_night FROM Rooms GROUP BY room\_type;

**Answer:** This query calculates the average price per night for each room type.

**Query 7: Calculate the difference in days between the first and last reservation made.**

SELECT DATEDIFF( (SELECT MIN(reservation\_date) FROM Reservations), (SELECT MAX(reservation\_date) FROM Reservations) ) AS days\_between\_first\_and\_last\_reservation;

**Answer:** This query calculates the difference in days between the first and last reservation made.

**Query 8: Find the number of reservations that were checked out on a Friday.**

SELECT COUNT(\*) AS num\_reservations\_on\_friday FROM Reservations WHERE DAYOFWEEK(check\_out\_date) = 6;

**Answer:** This query counts the number of reservations that were checked out on a Friday (where 6 corresponds to Friday in MySQL's DAYOFWEEK function).

**Query 9: Calculate the average number of guests per reservation for each room type**

SELECT room\_type, AVG(num\_guests) AS avg\_guests\_per\_reservation FROM Reservations, Rooms WHERE Reservations.room\_id = Rooms.room\_id GROUP BY room\_type;

**Answer:** This query calculates the average number of guests per reservation for each room type without using explicit joins. It achieves the same result by specifying the relationship between the two tables in the WHERE clause.

**Query 10: Find the room with the highest total revenue earned from reservations.**

SELECT room\_id, SUM(total\_amount) AS total\_revenue FROM Reservations GROUP BY room\_id ORDER BY total\_revenue DESC LIMIT 1;

**Answer:** This query identifies the room with the highest total revenue earned from reservations by grouping and summing reservation totals

**Query 11: Find the guest who has the highest total amount spent on reservations in the last 6 months.**

SELECT CONCAT(first\_name, ' ', last\_name) AS guest\_name, MAX(total\_amount) AS highest\_total\_amount FROM Guests WHERE guest\_id IN ( SELECT guest\_id FROM Reservations WHERE check\_in\_date >= DATE\_SUB(NOW(), INTERVAL 6 MONTH) ) GROUP BY guest\_id;

**Answer:** This query identifies the guest who has spent the highest total amount on reservations in the last 6 months

**Query 12: Find the rooms that haven't been cleaned in the last 7 days and are due for maintenance within the next 5 days.**

SELECT \* FROM Rooms WHERE DATEDIFF(NOW(), last\_cleaned\_date) > 7 AND DATEDIFF(maintenance\_due\_date, NOW()) <= 5;

**Answer:** This query retrieves rooms that haven't been cleaned in the last 7 days and are due for maintenance within the next 5 days.

**Query 13: Calculate the total revenue earned on weekends (Saturday and Sunday).**

SELECT SUM(total\_amount) AS total\_revenue\_on\_weekends FROM Reservations WHERE DAYOFWEEK(check\_in\_date) IN (1, 7);

**Answer:** This query calculates the total revenue earned from reservations with check-in dates on weekends.

**Query 14: Calculate the average length of stay for reservations made by guests who have booked a Penthouse room in the past year.**

SELECT AVG(DATEDIFF(check\_out\_date, check\_in\_date)) AS avg\_length\_of\_stay FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 AND room\_id IN ( SELECT room\_id FROM Rooms WHERE room\_type = 'Penthouse' );

**Query 15: Calculate the total revenue earned in each month of the current year.**

SELECT MONTH(check\_in\_date) AS month, SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) GROUP BY MONTH(check\_in\_date);

**Answer:** This query calculates the total revenue earned in each month of the current year.

**Query 16: Calculate the average stay duration for guests of each age group (in years).**

SELECT CASE WHEN TIMESTAMPDIFF(YEAR, date\_of\_birth, CURDATE()) < 25 THEN 'Under 25' WHEN TIMESTAMPDIFF(YEAR, date\_of\_birth, CURDATE()) BETWEEN 25 AND 45 THEN '25-45' ELSE 'Over 45' END AS age\_group, AVG(DATEDIFF(check\_out\_date, check\_in\_date)) AS avg\_stay\_duration FROM Reservations, Guests WHERE Reservations.guest\_id = Guests.guest\_id GROUP BY age\_group;

**Answer:** This query calculates the average stay duration for guests in different age groups.

**Query 17: Calculate the total revenue earned from reservations made on weekdays (Monday to Friday).**

SELECT SUM(total\_amount) AS total\_revenue\_weekdays FROM Reservations WHERE DAYOFWEEK(check\_in\_date) BETWEEN 2 AND 6;

**Answer:** This query calculates the total revenue earned from reservations made on weekdays.

**Query 18: Count the number of reservations that were made by guests who have a registration date before their date of birth.**

SELECT COUNT(\*) AS num\_invalid\_reservations FROM Reservations WHERE guest\_id IN ( SELECT guest\_id FROM Guests WHERE registration\_date < date\_of\_birth );

**Answer:** This query counts the number of reservations made by guests whose registration date is before their date of birth.

**Query 19: Find the room number and room type of the room that has the highest total revenue generated from reservations in the past year.**

SELECT room\_number, room\_type FROM Rooms WHERE room\_id = ( SELECT room\_id FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 GROUP BY room\_id ORDER BY SUM(total\_amount) DESC LIMIT 1 );

**Answer:** This query finds the room number and room type of the room that has generated the highest total revenue from reservations in the past year.

**Query 20: Calculate the percentage of reservations that were made by guests who registered in the current year.**

SELECT (COUNT(\*) / (SELECT COUNT(\*) FROM Reservations)) \* 100 AS percentage FROM Reservations WHERE guest\_id IN ( SELECT guest\_id FROM Guests WHERE YEAR(registration\_date) = YEAR(NOW()) );

**Answer:** This query calculates the percentage of reservations that were made by guests who registered in the current year.

**Query 21: Calculate the total revenue earned in the current month.**

SELECT SUM(total\_amount) AS total\_revenue FROM Reservations WHERE MONTH(check\_in\_date) = MONTH(NOW());

**Answer:** This query calculates the total revenue earned in the current month.

**Query 22: Calculate the average stay duration for all reservations made in the last year.**

SELECT AVG(DATEDIFF(check\_out\_date, check\_in\_date)) AS avg\_stay\_duration FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1;

**Answer:** This query calculates the average stay duration for all reservations made in the last year.

**Query 23: Calculate the number of reservations made on weekends (Saturday and Sunday) in the past 6 months.**

SELECT COUNT(\*) AS num\_weekend\_reservations FROM Reservations WHERE DAYOFWEEK(check\_in\_date) IN (1, 7) AND DATEDIFF(NOW(), reservation\_date) <= 180;

**Answer:** This query calculates the number of reservations made on weekends in the past 6 months.

**Query 24: Calculate the average lead time for all reservations made in the last 3 months.**

SELECT AVG(DATEDIFF(check\_in\_date, reservation\_date)) AS avg\_lead\_time FROM Reservations WHERE DATEDIFF(NOW(), reservation\_date) <= 90;

**Answer:** This query calculates the average lead time for all reservations made in the last 3 months.

**Query 25: Calculate the total number of guests who stayed during weekdays (Monday to Friday) in the past year.**

SELECT SUM(num\_guests) AS total\_guests\_weekdays FROM Reservations WHERE DAYOFWEEK(check\_in\_date) BETWEEN 2 AND 6 AND YEAR(check\_in\_date) = YEAR(NOW()) - 1;

**Answer:** This query calculates the total number of guests who stayed during weekdays in the past year.

**Query 26: Find the total number of reservations made by each guest in the last 12 months.**

SELECT guest\_id, COUNT(\*) AS num\_reservations FROM Reservations WHERE DATEDIFF(NOW(), reservation\_date) <= 365 GROUP BY guest\_id;

**Answer:** This query calculates the total number of reservations made by each guest in the last 12 months.

**Query 27: Calculate the average number of guests per reservation for the current year.**

SELECT AVG(num\_guests) AS avg\_guests\_per\_reservation FROM Reservations WHERE YEAR(reservation\_date) = YEAR(NOW());

**Answer:** This query calculates the average number of guests per reservation for the current year.

**Query 28: Find the rooms that have been occupied continuously for at least 5 days.**

SELECT DISTINCT room\_id, room\_number FROM Reservations WHERE DATEDIFF(check\_out\_date, check\_in\_date) >= 5;

**Answer:** This query identifies rooms that have been occupied continuously for at least 5 days.

**Query 29: Calculate the average stay duration for each room type in the past year.**

SELECT room\_id, AVG(DATEDIFF(check\_out\_date, check\_in\_date)) AS avg\_stay\_duration FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 GROUP BY room\_id;

**Answer:** This query calculates the average stay duration for each room type in the past year.

**Query 30: Calculate the number of reservations made by guests aged between 25 and 40 in the last 6 months.**

SELECT COUNT(\*) AS num\_reservations FROM Reservations WHERE DATEDIFF(NOW(), reservation\_date) <= 180 AND guest\_id IN ( SELECT guest\_id FROM Guests WHERE TIMESTAMPDIFF(YEAR, date\_of\_birth, NOW()) BETWEEN 25 AND 40 );

**Answer:** This query calculates the number of reservations made by guests aged between 25 and 40 in the last 6 months.

**Query 31: Calculate the average total amount spent by guests for reservations made on weekends (Saturday or Sunday).**

SELECT AVG(total\_amount) AS avg\_weekend\_spending FROM Reservations WHERE DAYOFWEEK(check\_in\_date) = 1 OR DAYOFWEEK(check\_in\_date) = 7;

**Answer:** This query calculates the average total amount spent by guests for reservations made on weekends (Saturday or Sunday).

**Query 32: Find the room with the highest average length of stay for reservations made in the current year.**

SELECT room\_id, AVG(DATEDIFF(check\_out\_date, check\_in\_date)) AS avg\_length\_of\_stay FROM Reservations WHERE YEAR(reservation\_date) = YEAR(NOW()) GROUP BY room\_id HAVING AVG(DATEDIFF(check\_out\_date, check\_in\_date)) = ( SELECT MAX(avg\_length) FROM ( SELECT room\_id, AVG(DATEDIFF(check\_out\_date, check\_in\_date)) AS avg\_length FROM Reservations WHERE YEAR(reservation\_date) = YEAR(NOW()) GROUP BY room\_id ) AS avg\_lengths );

**Answer:** This query identifies the room with the highest average length of stay for reservations made in the current year.

**Query 33: Calculate the total revenue earned from reservations made by guests from a specific city in the past year.**

SELECT SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 AND guest\_id IN ( SELECT guest\_id FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 GROUP BY guest\_id HAVING MAX(city = 'CityName') );

**Answer:** This query calculates the total revenue earned from reservations made by guests from a specific city in the past year.

**Query 34: Find the room with the highest total revenue earned in the current year.**

SELECT room\_id, SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(reservation\_date) = YEAR(NOW()) GROUP BY room\_id HAVING SUM(total\_amount) = ( SELECT MAX(total\_revenue) FROM ( SELECT room\_id, SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(reservation\_date) = YEAR(NOW()) GROUP BY room\_id ) AS total\_revenues );

**Answer:** This query identifies the room with the highest total revenue earned in the current year.

**Query 35: Calculate the average number of days guests book in advance before making a reservation.**

SELECT AVG(DATEDIFF(check\_in\_date, reservation\_date)) AS avg\_days\_booked\_in\_advance FROM Reservations;

**Answer:** This query calculates the average number of days guests book in advance before making a reservation.

**Query 36: Calculate the total revenue earned from reservations for each room type in the current year, sorted by room type.**

SELECT room\_type, SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(reservation\_date) = YEAR(NOW()) GROUP BY room\_type ORDER BY room\_type;

**Answer:** This query calculates the total revenue earned from reservations for each room type in the current year, sorted by room type.

**Query 37: Find the room with the lowest total revenue earned in the past year.**

SELECT room\_id, SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 GROUP BY room\_id HAVING SUM(total\_amount) = ( SELECT MIN(total\_revenue) FROM ( SELECT room\_id, SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 GROUP BY room\_id ) AS total\_revenues );

**Answer:** This query identifies the room with the lowest total revenue earned in the past year.

**Query 38: Calculate the average number of guests per reservation for each room type in the past year, sorted by room type.**

SELECT room\_type, AVG(num\_guests) AS avg\_guests\_per\_reservation FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 GROUP BY room\_type ORDER BY room\_type;

**Answer:** This query calculates the average number of guests per reservation for each room type in the past year, sorted by room type.

**Query 39: Calculate the total revenue earned from reservations made by guests who have stayed more than once in the past year.**

SELECT SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 AND guest\_id IN ( SELECT guest\_id FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 GROUP BY guest\_id HAVING COUNT(\*) > 1 );

**Answer:** This query calculates the total revenue earned from reservations made by guests who have stayed more than once in the past year.

**Query 40: Calculate the total revenue earned from reservations for each room type in the past year, sorted by total revenue in descending order.**

SELECT room\_type, SUM(total\_amount) AS total\_revenue FROM Reservations WHERE YEAR(check\_in\_date) = YEAR(NOW()) - 1 GROUP BY room\_type ORDER BY total\_revenue DESC;

**Answer:** This query calculates the total revenue earned from reservations for each room type in the past year, sorted by total revenue in descending order.