CODING CHALLENGES - HOTEL MANAGEMENT SYSTEM

Problem Statement:

Create an SQL schema from the **Hotel** and **Guest** classes, using their attributes for table column names.

SQL Schema:

Table: Hotels

- HotelID (**Primary Key, int**): Unique identifier for each hotel.
- Name (**string**): The name of the hotel.
- Location (**string**): The location of the hotel.
- Rating (**decimal**): The average rating of the hotel (1-5).

Table: Rooms

- RoomID (**Primary Key, int**): Unique identifier for each room.
- HotelID (**Foreign Key, int**): References **HotelID** in Hotels table.
- RoomNumber (**string**): Room number or identifier.
- RoomType (**string**): Type of room (e.g., "Single," "Double," "Suite").
- PricePerNight (**decimal**): Cost per night.
- Available (**bit**): Indicates if the room is available for booking (1 for yes, 0 for no).

Table: Guests

- GuestID (**Primary Key, int**): Unique identifier for each guest.
- FullName (**string**): Name of the guest.
- Email (**string**): Guest email (unique).
- PhoneNumber (**string**): Guest phone number (unique).
- CheckInDate (**datetime**): The date the guest checked in.
- CheckOutDate (**datetime**): The date the guest checked out.

Table: Bookings

- BookingID (**Primary Key, int**): Unique identifier for each booking.
- GuestID (**Foreign Key, int**): References **GuestID** in Guests table.
- RoomID (**Foreign Key, int**): References **RoomID** in Rooms table.
- BookingDate (**datetime**): The date of booking.
- TotalAmount (**decimal**): The total price for the stay.
- Status (**string**): Booking status (e.g., "Confirmed," "Cancelled," "Checked Out").

Table: Payments

- PaymentID (**Primary Key, int**): Unique identifier for each payment.
- BookingID (**Foreign Key, int**): References **BookingID** in Bookings table.

- AmountPaid (**decimal**): The amount paid.
- PaymentDate (**datetime**): Date and time of payment.
- PaymentMethod (**string**): Payment method (e.g., "Credit Card," "Cash").

Table: Events

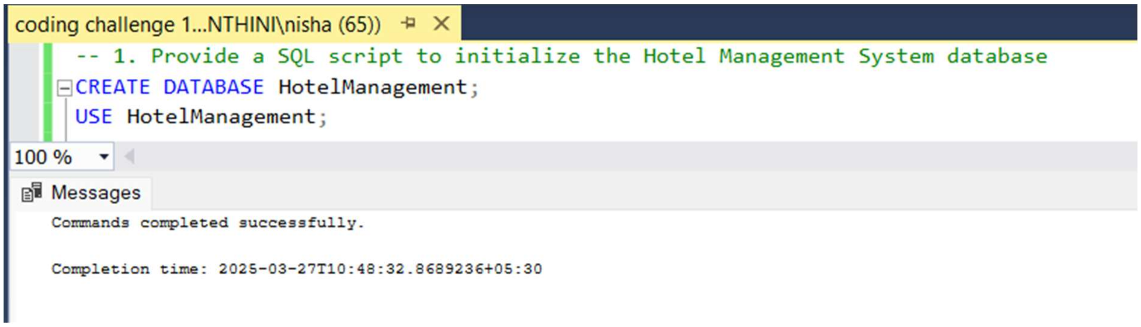
- EventID (**Primary Key, int**): Unique identifier for each event hosted at the hotel.
- HotelID (**Foreign Key, int**): References **HotelID** in Hotels table.
- EventName (**string**): The name or title of the event.
- EventDate (**datetime**): Date and time of the event.
- Venue (**string**): Venue of the event.

Table: EventParticipants

- ParticipantID (**Primary Key, int**): Unique identifier for each participant.
- ParticipantName (**string**): Name of the participant (guest or organization).
- ParticipantType (**string**): Type of participant ("Guest" or "Organization").
- EventID (**Foreign Key, int**): References **EventID** of the associated event.

Tasks:

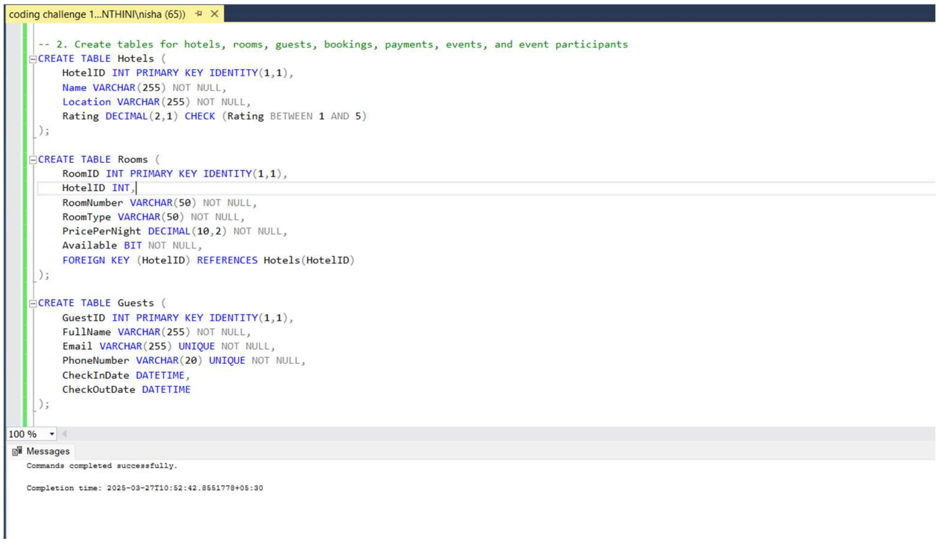
1. **Provide a SQL script** to initialize the **Hotel Management System** database.



```
-- 1. Provide a SQL script to initialize the Hotel Management System database
CREATE DATABASE HotelManagement;
USE HotelManagement;
```

```
Commands completed successfully.

Completion time: 2025-03-27T10:48:32.8689236+05:30
```

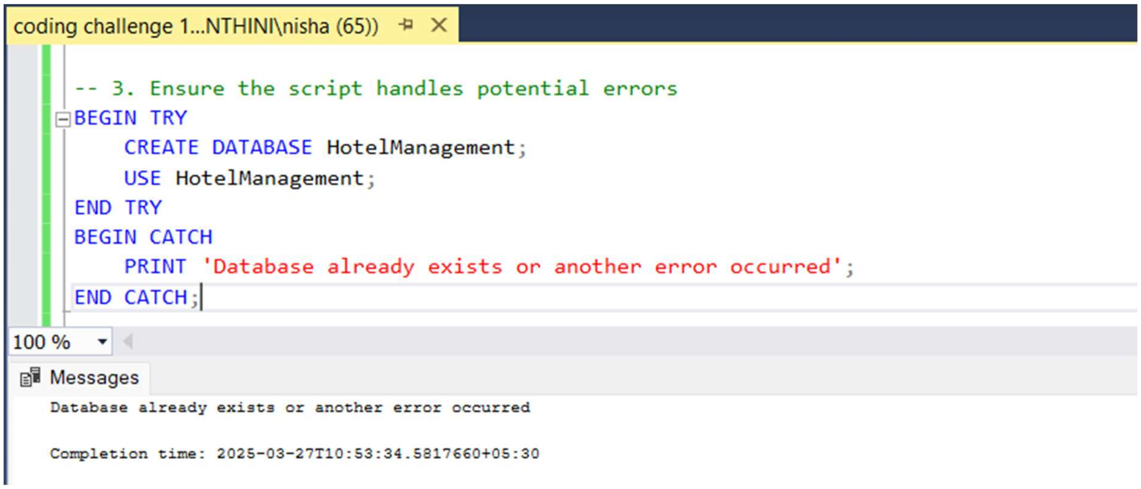
2. **Create tables** for **hotels, rooms, guests, bookings, payments, events, and event participants**, defining appropriate primary and foreign keys.



```
-- 2. Create tables for hotels, rooms, guests, bookings, payments, events, and event participants
CREATE TABLE Hotels (
    HotelID INT PRIMARY KEY IDENTITY(1,1),
    Name VARCHAR(255) NOT NULL,
    Location VARCHAR(255) NOT NULL,
    Rating DECIMAL(2,1) CHECK (Rating BETWEEN 1 AND 5)
);

CREATE TABLE Rooms (
    RoomID INT PRIMARY KEY IDENTITY(1,1),
    HotelID INT,
    RoomNumber VARCHAR(50) NOT NULL,
    RoomType VARCHAR(50) NOT NULL,
    PricePerNight DECIMAL(10,2) NOT NULL,
    Available BIT NOT NULL,
    FOREIGN KEY (HotelID) REFERENCES Hotels(HotelID)
);

CREATE TABLE Guests (
    GuestID INT PRIMARY KEY IDENTITY(1,1),
    FullName VARCHAR(255) NOT NULL,
    Email VARCHAR(255) UNIQUE NOT NULL,
    PhoneNumber VARCHAR(20) UNIQUE NOT NULL,
    CheckInDate DATETIME,
    CheckOutDate DATETIME
);
```

```
Commands completed successfully.

Completion time: 2025-03-27T10:52:42.8551778+05:30
```

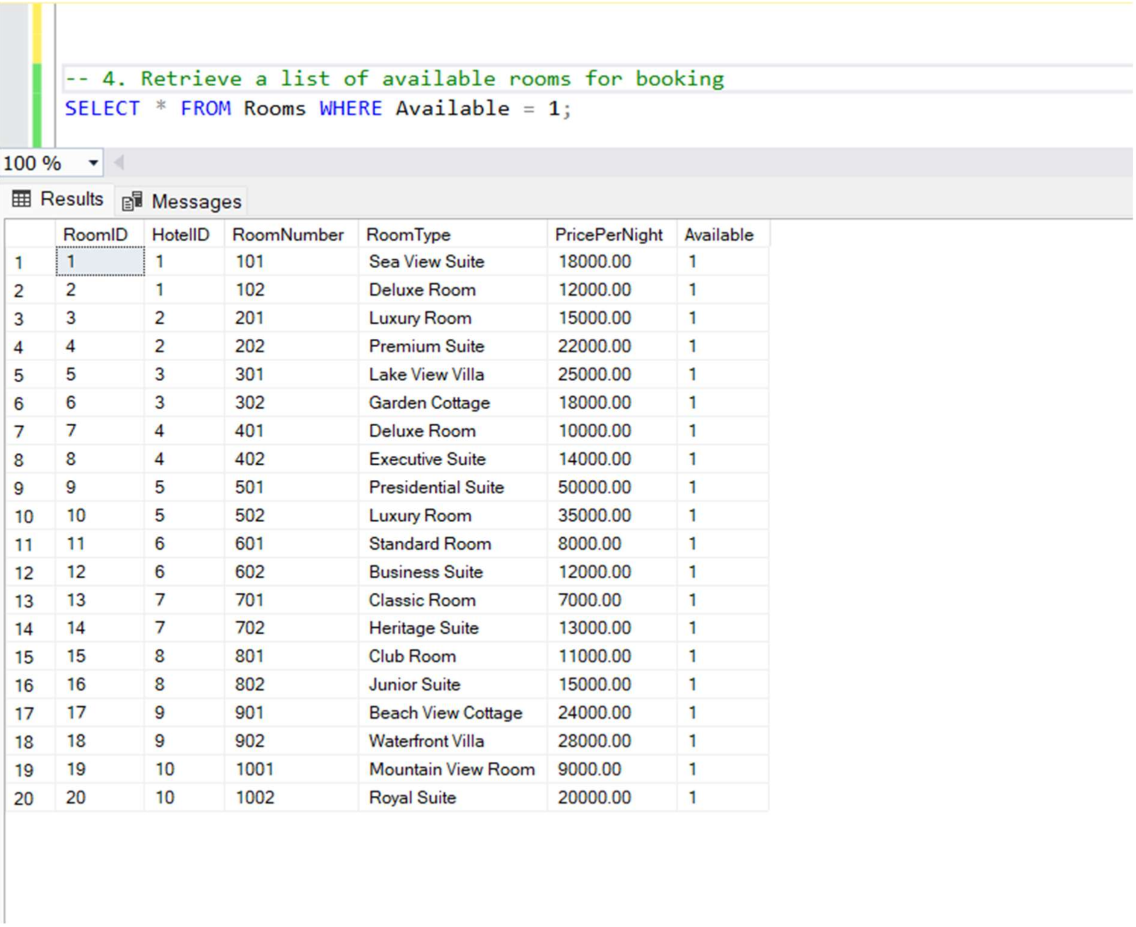
3. **Ensure the script handles potential errors**, such as checking if the database or tables already exist before creating them.



```sql
-- 3. Ensure the script handles potential errors
BEGIN TRY
    CREATE DATABASE HotelManagement;
    USE HotelManagement;
END TRY
BEGIN CATCH
    PRINT 'Database already exists or another error occurred';
END CATCH;
```

```
Database already exists or another error occurred

Completion time: 2025-03-27T10:53:34.5817660+05:30
```

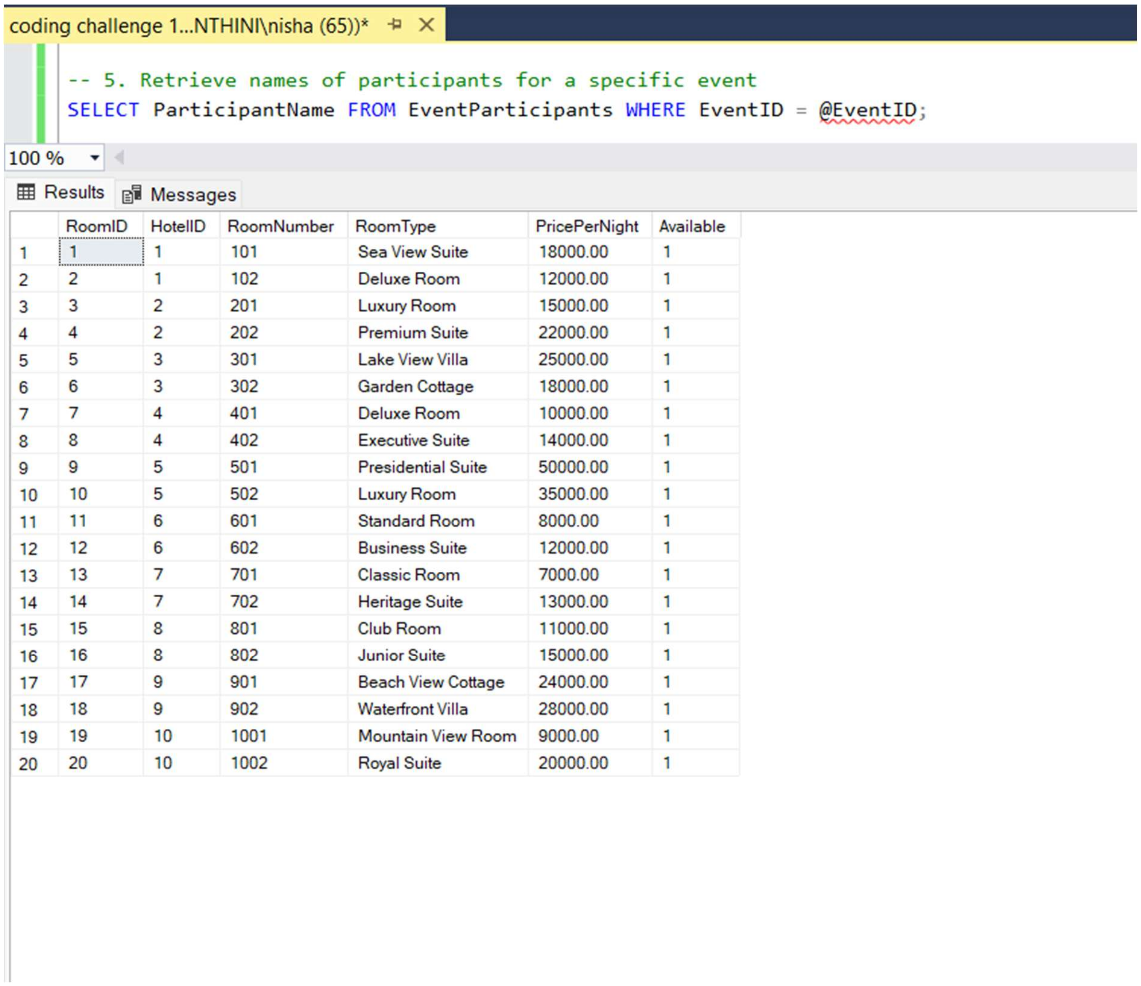
4. **Write an SQL query** to retrieve a list of available rooms for booking (Available = 1).



```sql
-- 4. Retrieve a list of available rooms for booking
SELECT * FROM Rooms WHERE Available = 1;
```

	RoomID	HotelID	RoomNumber	RoomType	PricePerNight	Available
1	1	1	101	Sea View Suite	18000.00	1
2	2	1	102	Deluxe Room	12000.00	1
3	3	2	201	Luxury Room	15000.00	1
4	4	2	202	Premium Suite	22000.00	1
5	5	3	301	Lake View Villa	25000.00	1
6	6	3	302	Garden Cottage	18000.00	1
7	7	4	401	Deluxe Room	10000.00	1
8	8	4	402	Executive Suite	14000.00	1
9	9	5	501	Presidential Suite	50000.00	1
10	10	5	502	Luxury Room	35000.00	1
11	11	6	601	Standard Room	8000.00	1
12	12	6	602	Business Suite	12000.00	1
13	13	7	701	Classic Room	7000.00	1
14	14	7	702	Heritage Suite	13000.00	1
15	15	8	801	Club Room	11000.00	1
16	16	8	802	Junior Suite	15000.00	1
17	17	9	901	Beach View Cottage	24000.00	1
18	18	9	902	Waterfront Villa	28000.00	1
19	19	10	1001	Mountain View Room	9000.00	1
20	20	10	1002	Royal Suite	20000.00	1

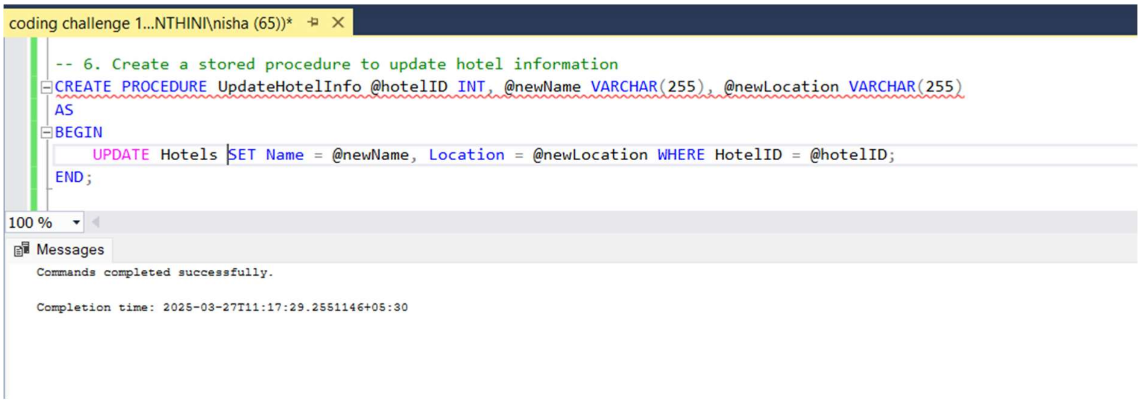
5. **Retrieve names of participants** registered for a specific hotel event using an @EventID parameter.



```sql
-- 5. Retrieve names of participants for a specific event
SELECT ParticipantName FROM EventParticipants WHERE EventID = @EventID;
```

	RoomID	HotelID	RoomNumber	RoomType	PricePerNight	Available
1	1	1	101	Sea View Suite	18000.00	1
2	2	1	102	Deluxe Room	12000.00	1
3	3	2	201	Luxury Room	15000.00	1
4	4	2	202	Premium Suite	22000.00	1
5	5	3	301	Lake View Villa	25000.00	1
6	6	3	302	Garden Cottage	18000.00	1
7	7	4	401	Deluxe Room	10000.00	1
8	8	4	402	Executive Suite	14000.00	1
9	9	5	501	Presidential Suite	50000.00	1
10	10	5	502	Luxury Room	35000.00	1
11	11	6	601	Standard Room	8000.00	1
12	12	6	602	Business Suite	12000.00	1
13	13	7	701	Classic Room	7000.00	1
14	14	7	702	Heritage Suite	13000.00	1
15	15	8	801	Club Room	11000.00	1
16	16	8	802	Junior Suite	15000.00	1
17	17	9	901	Beach View Cottage	24000.00	1
18	18	9	902	Waterfront Villa	28000.00	1
19	19	10	1001	Mountain View Room	9000.00	1
20	20	10	1002	Royal Suite	20000.00	1

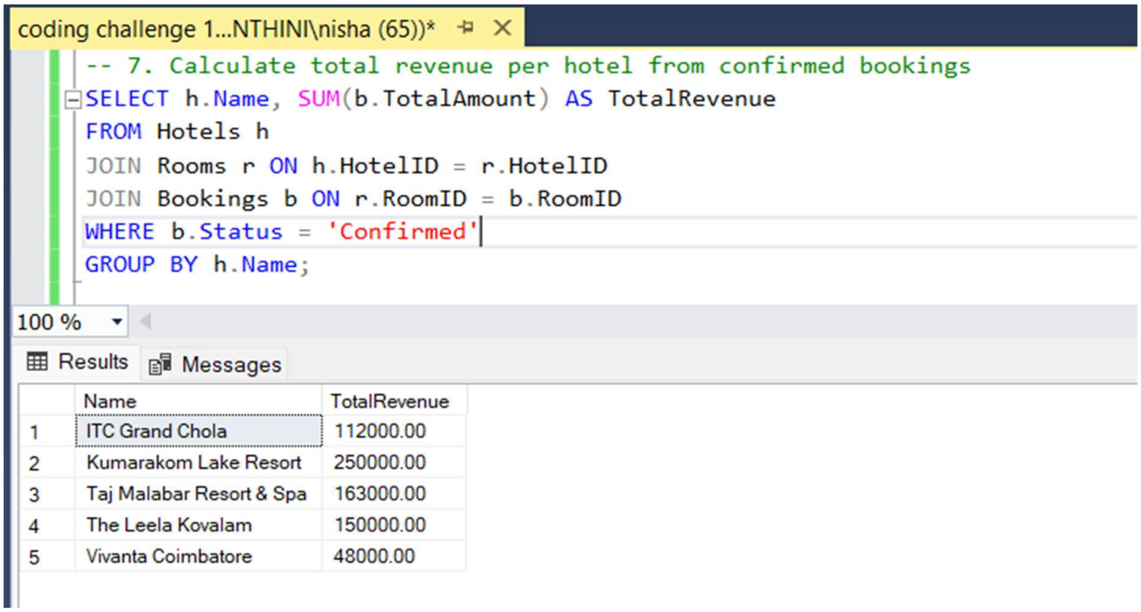
6. **Create a stored procedure** that allows a hotel to update its information (name and location) in the "Hotels" table.



```sql
-- 6. Create a stored procedure to update hotel information
CREATE PROCEDURE UpdateHotelInfo @hotelID INT, @newName VARCHAR(255), @newLocation VARCHAR(255)
AS
BEGIN
    UPDATE Hotels SET Name = @newName, Location = @newLocation WHERE HotelID = @hotelID;
END;
```

```
Commands completed successfully.

Completion time: 2025-03-27T11:17:29.2551146+05:30
```

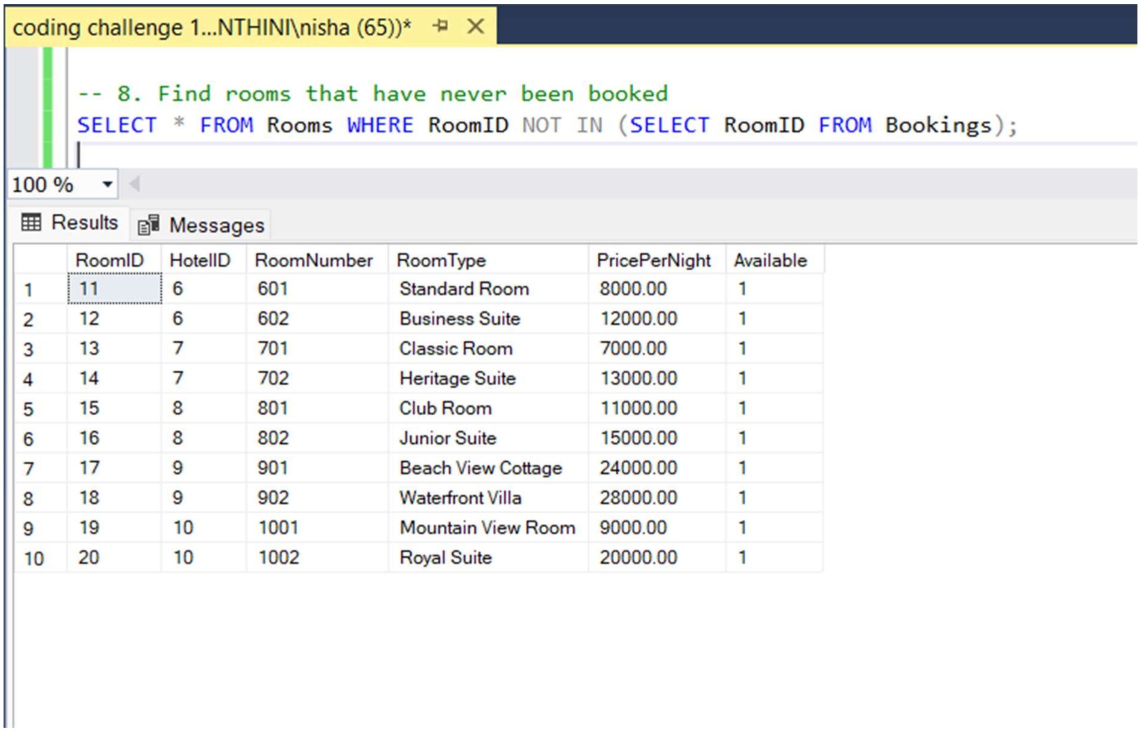
7. **Write an SQL query** to calculate the **total revenue generated** by each hotel from confirmed bookings.



```sql
-- 7. Calculate total revenue per hotel from confirmed bookings
SELECT h.Name, SUM(b.TotalAmount) AS TotalRevenue
FROM Hotels h
JOIN Rooms r ON h.HotelID = r.HotelID
JOIN Bookings b ON r.RoomID = b.RoomID
WHERE b.Status = 'Confirmed'
GROUP BY h.Name;
```

	Name	TotalRevenue
1	ITC Grand Chola	112000.00
2	Kumarakom Lake Resort	250000.00
3	Taj Malabar Resort & Spa	163000.00
4	The Leela Kovalam	150000.00
5	Vivanta Coimbatore	48000.00

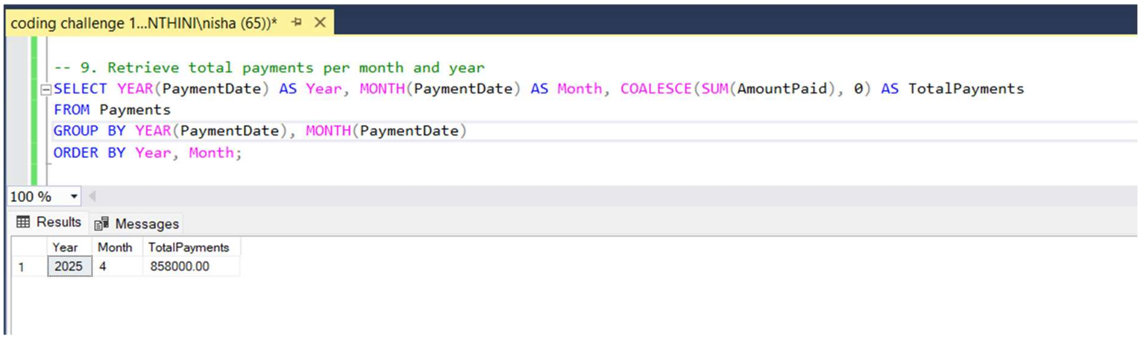
8. **Find rooms that have never been booked** by selecting their details from the Rooms table.



```sql
-- 8. Find rooms that have never been booked
SELECT * FROM Rooms WHERE RoomID NOT IN (SELECT RoomID FROM Bookings);
```

	RoomID	HotelID	RoomNumber	RoomType	PricePerNight	Available
1	11	6	601	Standard Room	8000.00	1
2	12	6	602	Business Suite	12000.00	1
3	13	7	701	Classic Room	7000.00	1
4	14	7	702	Heritage Suite	13000.00	1
5	15	8	801	Club Room	11000.00	1
6	16	8	802	Junior Suite	15000.00	1
7	17	9	901	Beach View Cottage	24000.00	1
8	18	9	902	Waterfront Villa	28000.00	1
9	19	10	1001	Mountain View Room	9000.00	1
10	20	10	1002	Royal Suite	20000.00	1

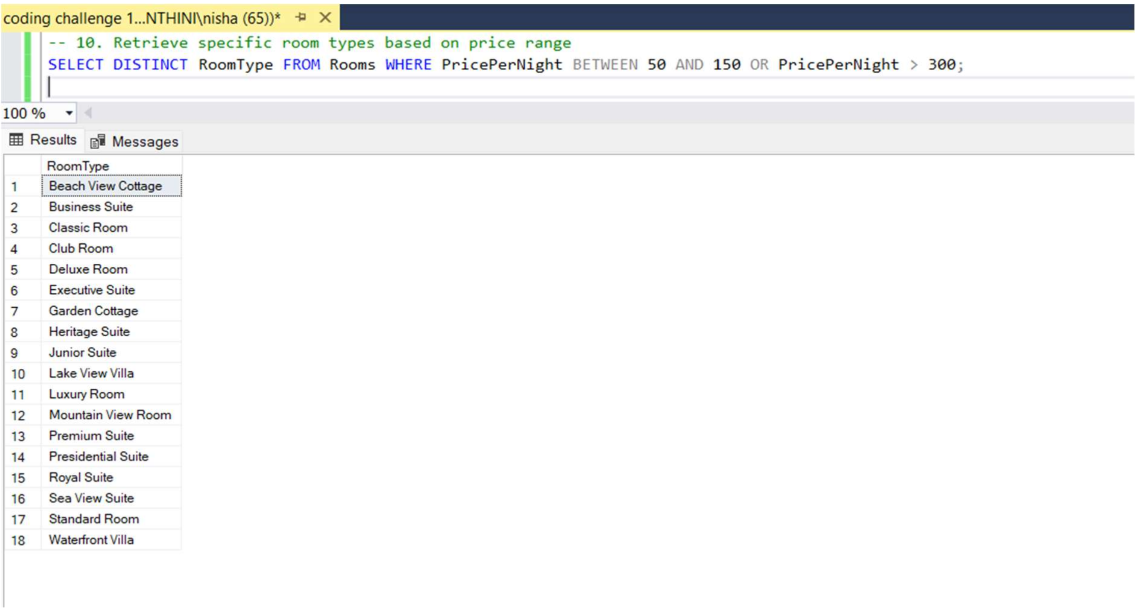
9. **Retrieve total payments per month and year**, ensuring missing months are handled properly.



```sql
-- 9. Retrieve total payments per month and year
SELECT YEAR(PaymentDate) AS Year, MONTH(PaymentDate) AS Month, COALESCE(SUM(AmountPaid), 0) AS TotalPayments
FROM Payments
GROUP BY YEAR(PaymentDate), MONTH(PaymentDate)
ORDER BY Year, Month;
```

	Year	Month	TotalPayments
1	2025	4	858000.00

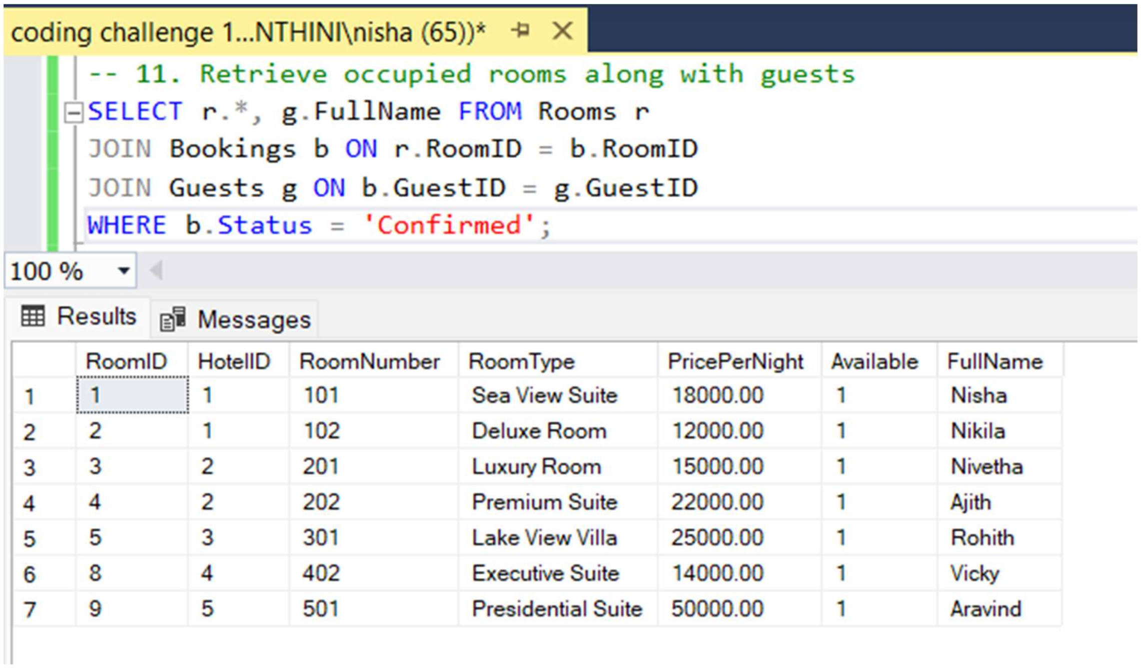
10. **Retrieve a list of room types** that are either **priced between $50 and $150 per night or above $300 per night**.



```
-- 10. Retrieve specific room types based on price range
SELECT DISTINCT RoomType FROM Rooms WHERE PricePerNight BETWEEN 50 AND 150 OR PricePerNight > 300;
```

	RoomType
1	Beach View Cottage
2	Business Suite
3	Classic Room
4	Club Room
5	Deluxe Room
6	Executive Suite
7	Garden Cottage
8	Heritage Suite
9	Junior Suite
10	Lake View Villa
11	Luxury Room
12	Mountain View Room
13	Premium Suite
14	Presidential Suite
15	Royal Suite
16	Sea View Suite
17	Standard Room
18	Waterfront Villa

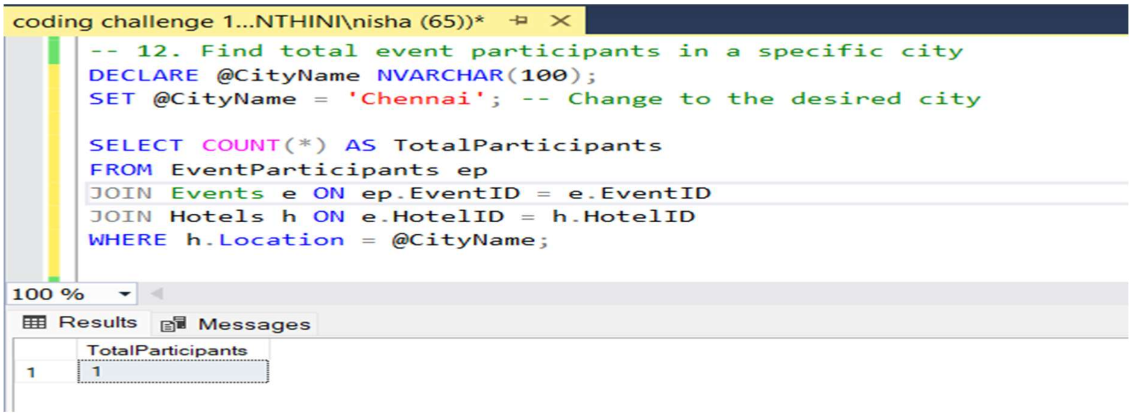
11. **Retrieve rooms along with their guests**, including only rooms that are currently occupied.



```
-- 11. Retrieve occupied rooms along with guests
SELECT r.*, g.FullName FROM Rooms r
JOIN Bookings b ON r.RoomID = b.RoomID
JOIN Guests g ON b.GuestID = g.GuestID
WHERE b.Status = 'Confirmed';
```

	RoomID	HotelID	RoomNumber	RoomType	PricePerNight	Available	FullName
1	1	1	101	Sea View Suite	18000.00	1	Nisha
2	2	1	102	Deluxe Room	12000.00	1	Nikila
3	3	2	201	Luxury Room	15000.00	1	Nivetha
4	4	2	202	Premium Suite	22000.00	1	Ajith
5	5	3	301	Lake View Villa	25000.00	1	Rohith
6	8	4	402	Executive Suite	14000.00	1	Vicky
7	9	5	501	Presidential Suite	50000.00	1	Aravind

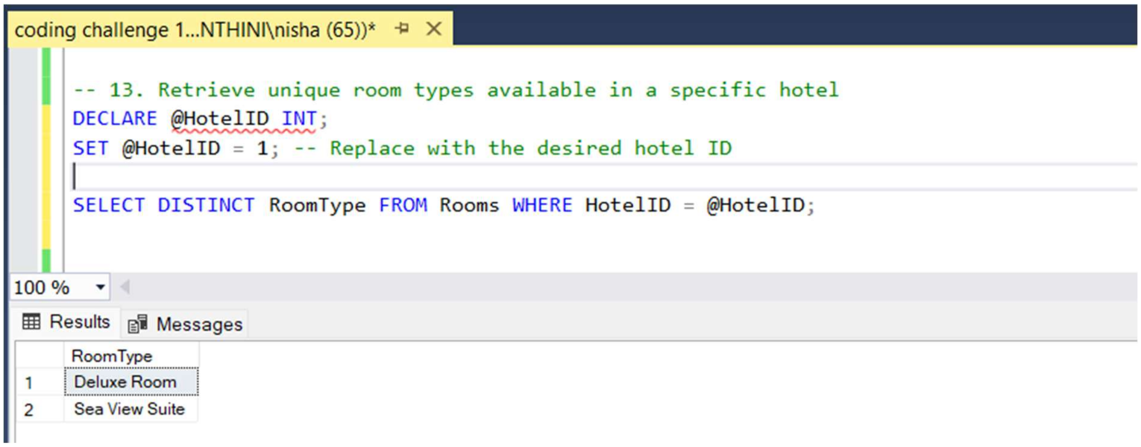
12. **Find the total number of participants** in events held in a specific city (@CityName).



```
-- 12. Find total event participants in a specific city
DECLARE @CityName NVARCHAR(100);
SET @CityName = 'Chennai'; -- Change to the desired city

SELECT COUNT(*) AS TotalParticipants
FROM EventParticipants ep
JOIN Events e ON ep.EventID = e.EventID
JOIN Hotels h ON e.HotelID = h.HotelID
WHERE h.Location = @CityName;
```

	TotalParticipants
1	1

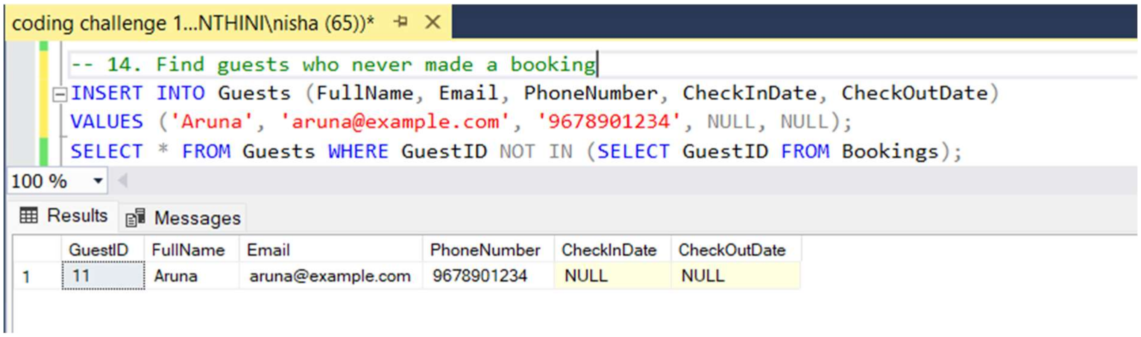
13. **Retrieve a list of unique room types** available in a specific hotel.



```sql
-- 13. Retrieve unique room types available in a specific hotel
DECLARE @HotelID INT;
SET @HotelID = 1; -- Replace with the desired hotel ID

SELECT DISTINCT RoomType FROM Rooms WHERE HotelID = @HotelID;
```

	RoomType
1	Deluxe Room
2	Sea View Suite

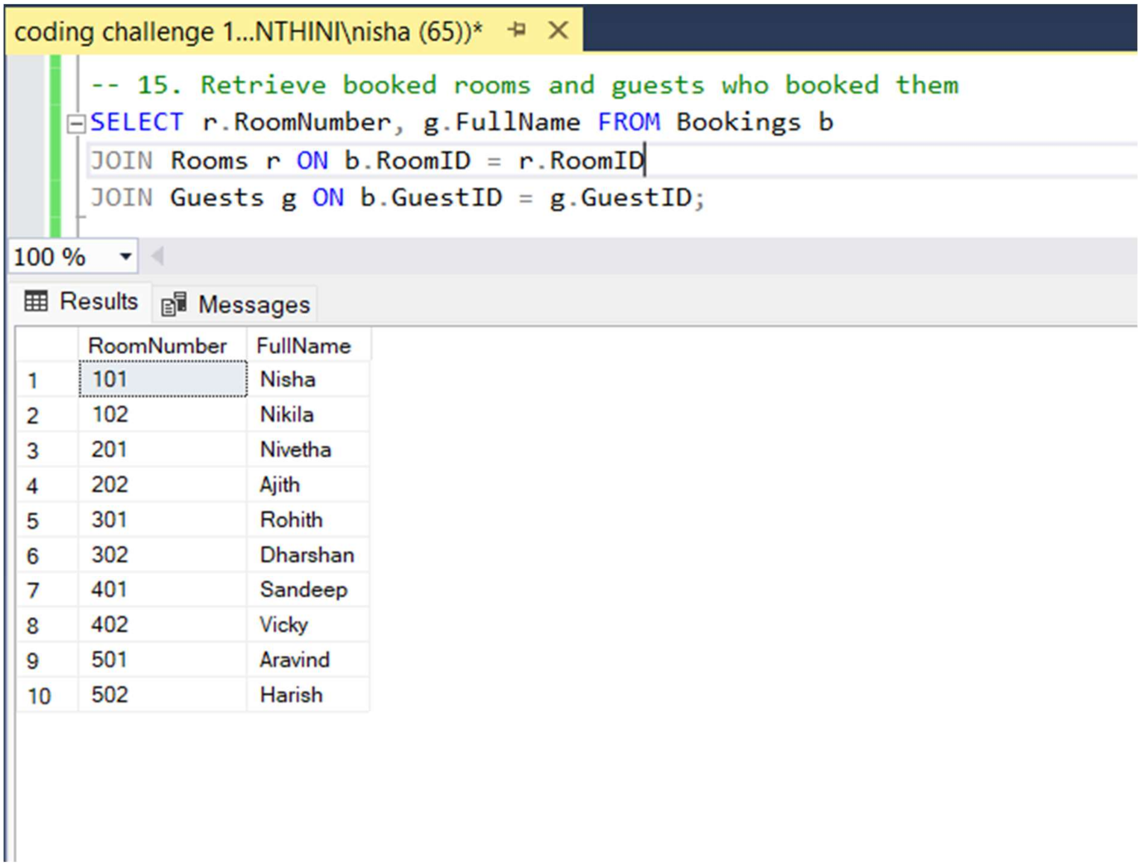
14. **Find guests who have never made a booking** from the hotel management system.



```sql
-- 14. Find guests who never made a booking
INSERT INTO Guests (FullName, Email, PhoneNumber, CheckInDate, CheckOutDate)
VALUES ('Aruna', 'aruna@example.com', '9678901234', NULL, NULL);
SELECT * FROM Guests WHERE GuestID NOT IN (SELECT GuestID FROM Bookings);
```

	GuestID	FullName	Email	PhoneNumber	CheckInDate	CheckOutDate
1	11	Aruna	aruna@example.com	9678901234	NULL	NULL

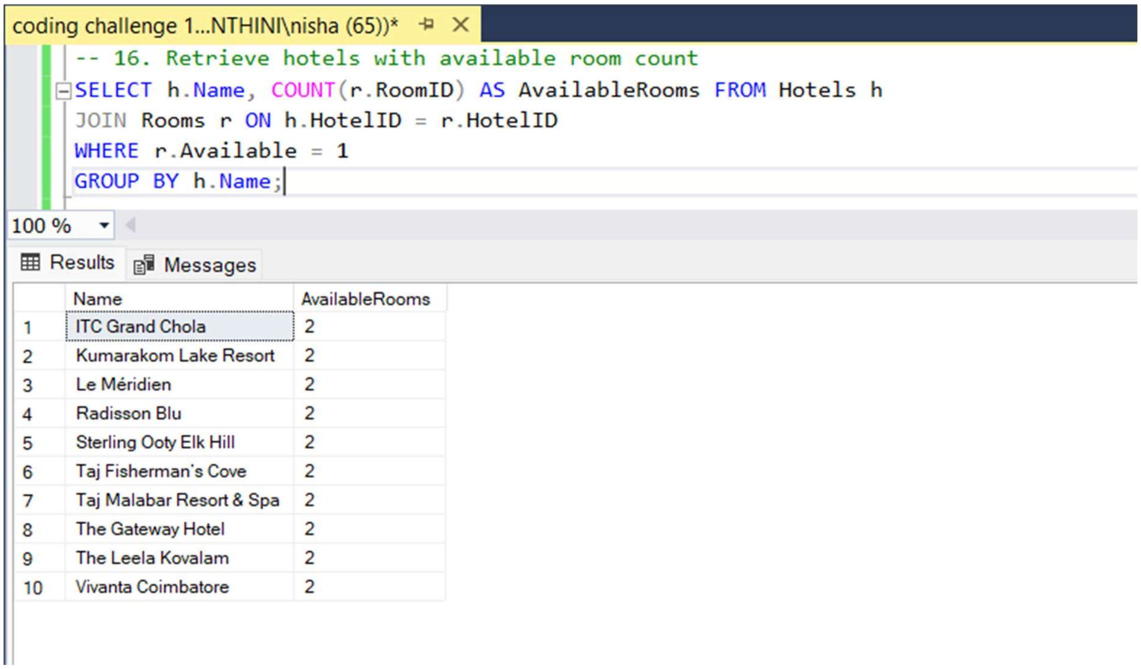
15. **Retrieve names of all booked rooms** along with the guests who booked them.



```sql
-- 15. Retrieve booked rooms and guests who booked them
SELECT r.RoomNumber, g.FullName FROM Bookings b
JOIN Rooms r ON b.RoomID = r.RoomID
JOIN Guests g ON b.GuestID = g.GuestID;
```

	RoomNumber	FullName
1	101	Nisha
2	102	Nikila
3	201	Nivetha
4	202	Ajith
5	301	Rohith
6	302	Dharshan
7	401	Sandeep
8	402	Vicky
9	501	Aravind
10	502	Harish

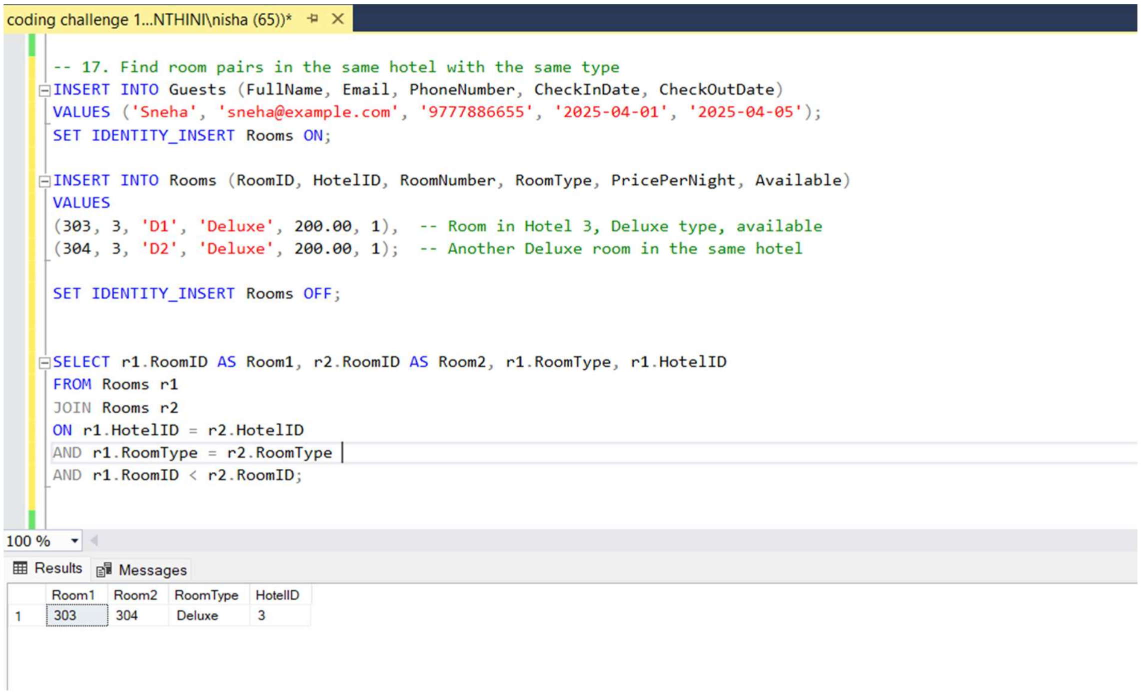
16. **Retrieve all hotels along with the count of available rooms** in each hotel.



```sql
-- 16. Retrieve hotels with available room count
SELECT h.Name, COUNT(r.RoomID) AS AvailableRooms FROM Hotels h
JOIN Rooms r ON h.HotelID = r.HotelID
WHERE r.Available = 1
GROUP BY h.Name;
```

	Name	AvailableRooms
1	ITC Grand Chola	2
2	Kumarakom Lake Resort	2
3	Le Méridien	2
4	Radisson Blu	2
5	Sterling Ooty Elk Hill	2
6	Taj Fisherman's Cove	2
7	Taj Malabar Resort & Spa	2
8	The Gateway Hotel	2
9	The Leela Kovalam	2
10	Vivanta Coimbatore	2

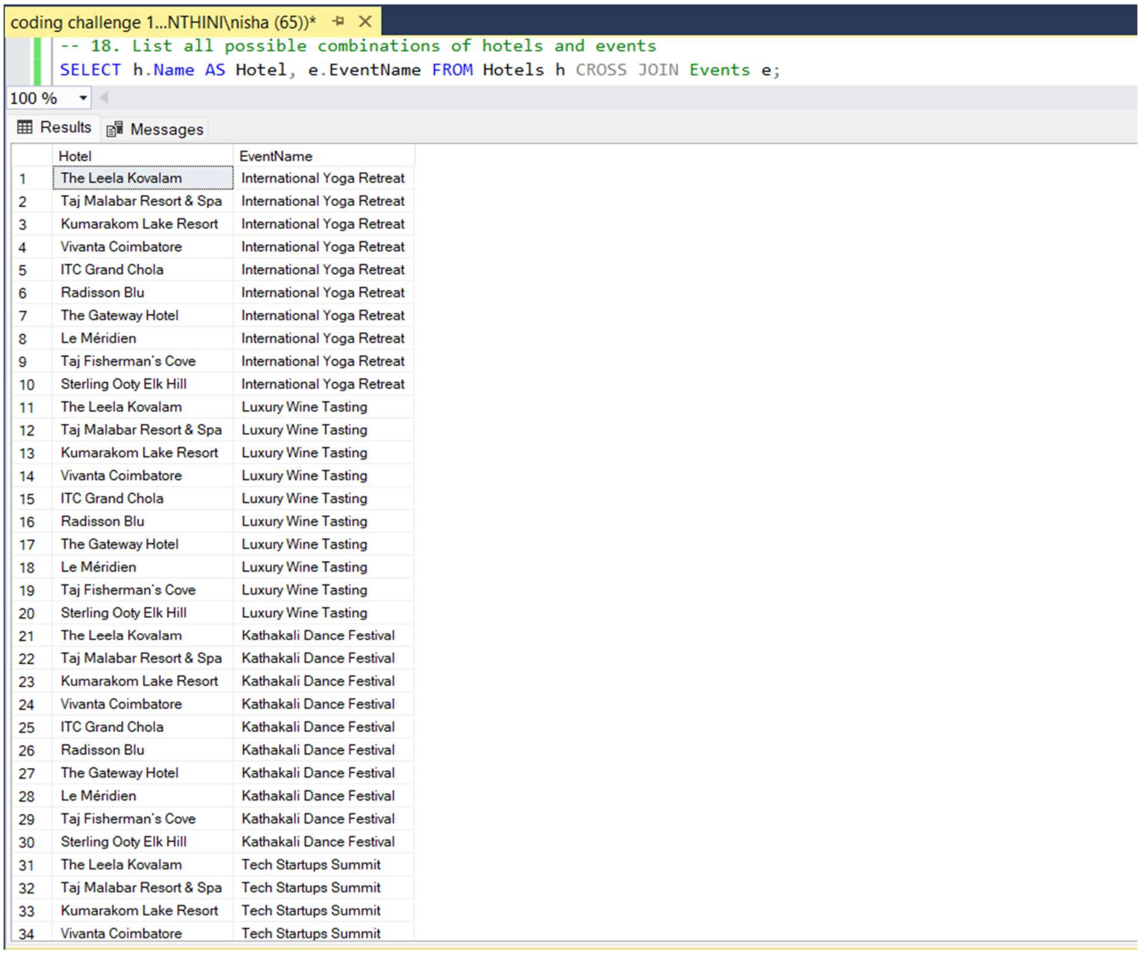
17. **Find pairs of rooms from the same hotel** that belong to the same room type.



```sql
-- 17. Find room pairs in the same hotel with the same type
INSERT INTO Guests (FullName, Email, PhoneNumber, CheckInDate, CheckOutDate)
VALUES ('Sneha', 'sneha@example.com', '9777886655', '2025-04-01', '2025-04-05');
SET IDENTITY_INSERT Rooms ON;

INSERT INTO Rooms (RoomID, HotelID, RoomNumber, RoomType, PricePerNight, Available)
VALUES
(303, 3, 'D1', 'Deluxe', 200.00, 1),  -- Room in Hotel 3, Deluxe type, available
(304, 3, 'D2', 'Deluxe', 200.00, 1);  -- Another Deluxe room in the same hotel

SET IDENTITY_INSERT Rooms OFF;


SELECT r1.RoomID AS Room1, r2.RoomID AS Room2, r1.RoomType, r1.HotelID
FROM Rooms r1
JOIN Rooms r2
ON r1.HotelID = r2.HotelID
AND r1.RoomType = r2.RoomType
AND r1.RoomID < r2.RoomID;
```

	Room1	Room2	RoomType	HotelID
1	303	304	Deluxe	3

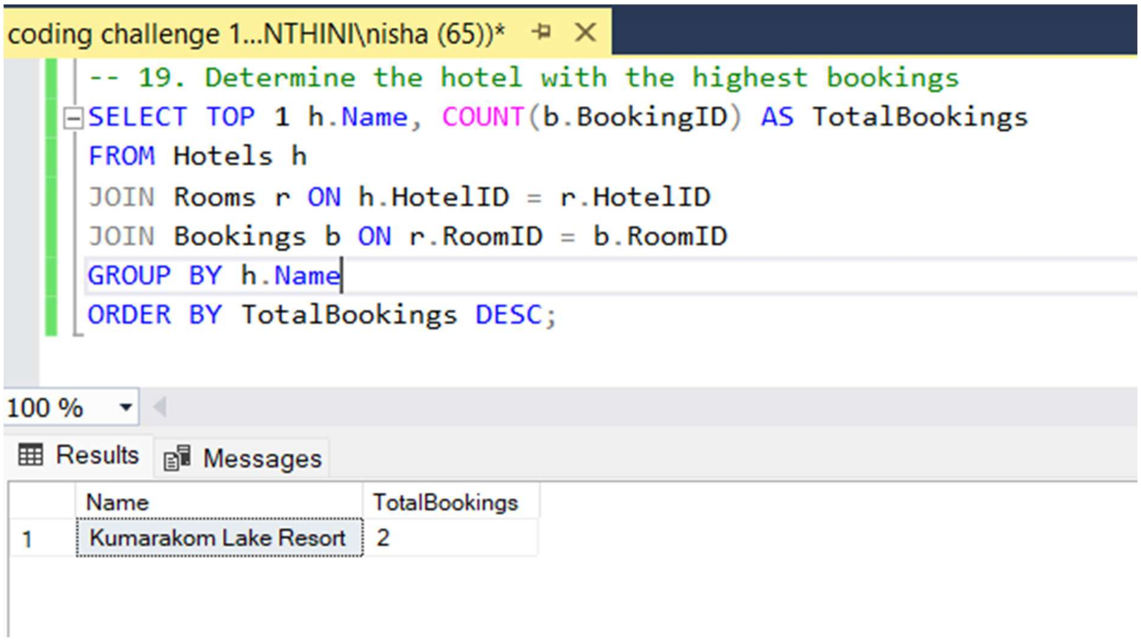
18. **List all possible combinations** of hotels and events.



```
-- 18. List all possible combinations of hotels and events
SELECT h.Name AS Hotel, e.EventName FROM Hotels h CROSS JOIN Events e;
```

	Hotel	EventName
1	The Leela Kovalam	International Yoga Retreat
2	Taj Malabar Resort & Spa	International Yoga Retreat
3	Kumarakom Lake Resort	International Yoga Retreat
4	Vivanta Coimbatore	International Yoga Retreat
5	ITC Grand Chola	International Yoga Retreat
6	Radisson Blu	International Yoga Retreat
7	The Gateway Hotel	International Yoga Retreat
8	Le Méridien	International Yoga Retreat
9	Taj Fisherman's Cove	International Yoga Retreat
10	Sterling Ooty Elk Hill	International Yoga Retreat
11	The Leela Kovalam	Luxury Wine Tasting
12	Taj Malabar Resort & Spa	Luxury Wine Tasting
13	Kumarakom Lake Resort	Luxury Wine Tasting
14	Vivanta Coimbatore	Luxury Wine Tasting
15	ITC Grand Chola	Luxury Wine Tasting
16	Radisson Blu	Luxury Wine Tasting
17	The Gateway Hotel	Luxury Wine Tasting
18	Le Méridien	Luxury Wine Tasting
19	Taj Fisherman's Cove	Luxury Wine Tasting
20	Sterling Ooty Elk Hill	Luxury Wine Tasting
21	The Leela Kovalam	Kathakali Dance Festival
22	Taj Malabar Resort & Spa	Kathakali Dance Festival
23	Kumarakom Lake Resort	Kathakali Dance Festival
24	Vivanta Coimbatore	Kathakali Dance Festival
25	ITC Grand Chola	Kathakali Dance Festival
26	Radisson Blu	Kathakali Dance Festival
27	The Gateway Hotel	Kathakali Dance Festival
28	Le Méridien	Kathakali Dance Festival
29	Taj Fisherman's Cove	Kathakali Dance Festival
30	Sterling Ooty Elk Hill	Kathakali Dance Festival
31	The Leela Kovalam	Tech Startups Summit
32	Taj Malabar Resort & Spa	Tech Startups Summit
33	Kumarakom Lake Resort	Tech Startups Summit
34	Vivanta Coimbatore	Tech Startups Summit

19. **Determine the hotel with the highest number of bookings.**



```
-- 19. Determine the hotel with the highest bookings
SELECT TOP 1 h.Name, COUNT(b.BookingID) AS TotalBookings
FROM Hotels h
JOIN Rooms r ON h.HotelID = r.HotelID
JOIN Bookings b ON r.RoomID = b.RoomID
GROUP BY h.Name
ORDER BY TotalBookings DESC;
```

	Name	TotalBookings
1	Kumarakom Lake Resort	2