Hotel Reservation System - SQL Queries Report

This document contains the implemented SQL queries for the **Hotel Reservation System** project.  
Each query includes:

* A short description of its purpose
* The actual SQL code
* The result (as text explanation and screenshot)
* Interpretation of the output

🔹 **1. List all rooms that are currently available**

**Description:**  
 This query retrieves all rooms that are available today based on RoomAvailability

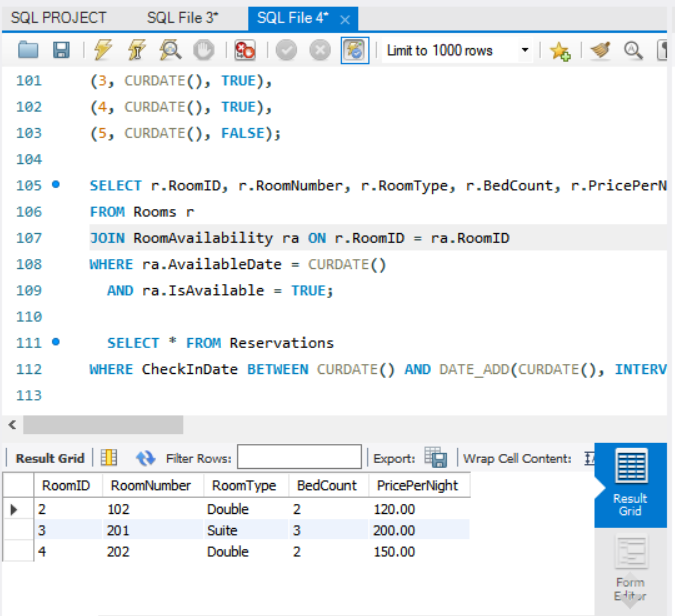
**SQL Query:**

SELECT r.RoomID, r.RoomNumber, r.RoomType, r.BedCount, r.PricePerNight

FROM Rooms r

JOIN RoomAvailability ra ON r.RoomID = ra.RoomID

WHERE ra.AvailableDate = CURDATE() AND ra.IsAvailable = TRUE;

**** **Result:**  
Returned 3 rows — rooms 102, 201, and 202 were available on the current date

**Interpretation of Output:**  
This shows the list of rooms available **today**. If no rows are returned, it means there are no rooms marked as available for the current date in the RoomAvailability table

**🔹 2. Retrieve all reservations made for dates within the next 30 days**

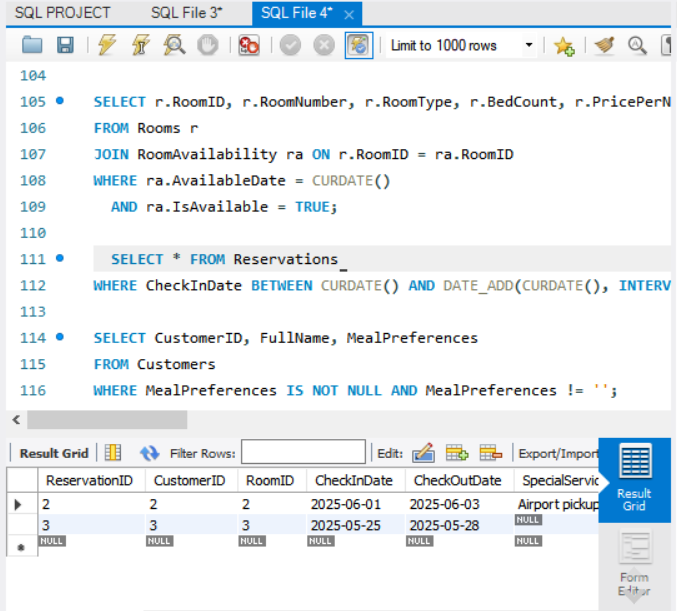
**Description:**

Gets all reservations where the Check-In date is within the next 30 days from today.

**SQL Query:**

SELECT \* FROM Reservations

WHERE CheckInDate BETWEEN CURDATE() AND DATE\_ADD(CURDATE(), INTERVAL 30 DAY);

 **Result:**

2 reservations found starting on 2025-05-25 and 2025-06-01.

**. Interpretation of Output:**  
 This query helps in forecasting occupancy by showing upcoming bookings. If rows are returned, it means those customers are expected to check in within the next month

🔹 **3. Get the list of customers who have special meal preferences**

**Description:**

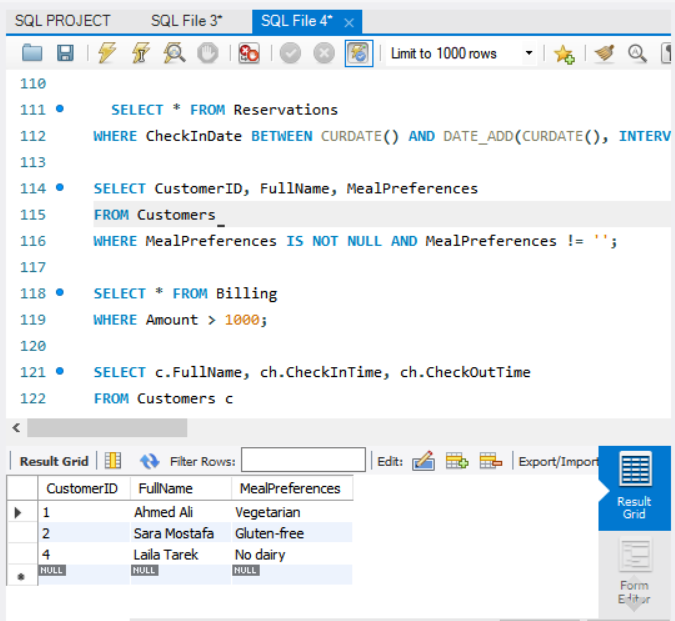
Retrieves customers who entered a value in the MealPreferences field

**SQL Query:**

SELECT CustomerID, FullName, MealPreferences

FROM Customers

WHERE MealPreferences IS NOT NULL AND MealPreferences != '';

 **Result:**

3 customers found: Ahmed Ali, Sara Mostafa, and Laila Tarek

**.** Interpretation of Output:  
Returns customers who have dietary needs. Helps hotel staff prepare meals accordingly

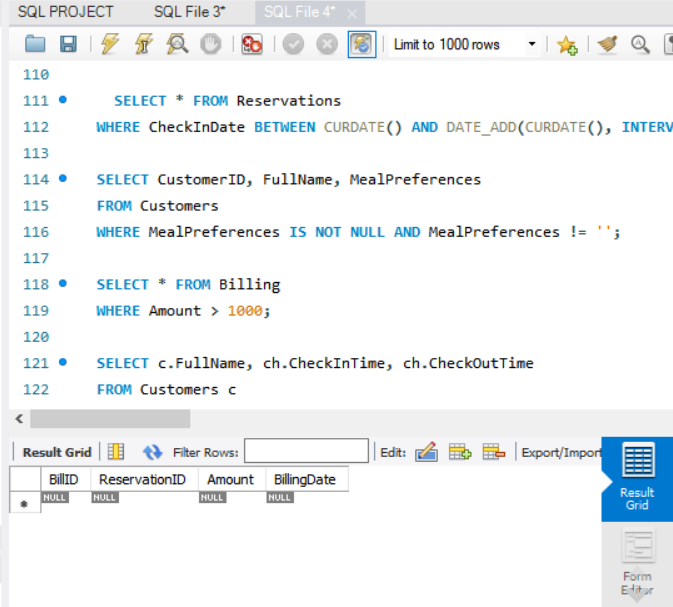
🔹 **4. Find all bills exceeding $1000**

**Description:**

Lists all billing entries where the amount exceeds $1000.

**SQL Query:**

SELECT \* FROM Billing

WHERE Amount > 1000;  **Result:**

No rows returned — all bills were less than $1000

**.** **Interpretation of Output:**  
Identifies high-value customers or long stays. Useful for VIP targeting or loyalty programs

🔹 **5. List customers names along with their check-in and check-out dates**

**Description:**

Combines data from customers, reservations, and check-in/out to show stay history.

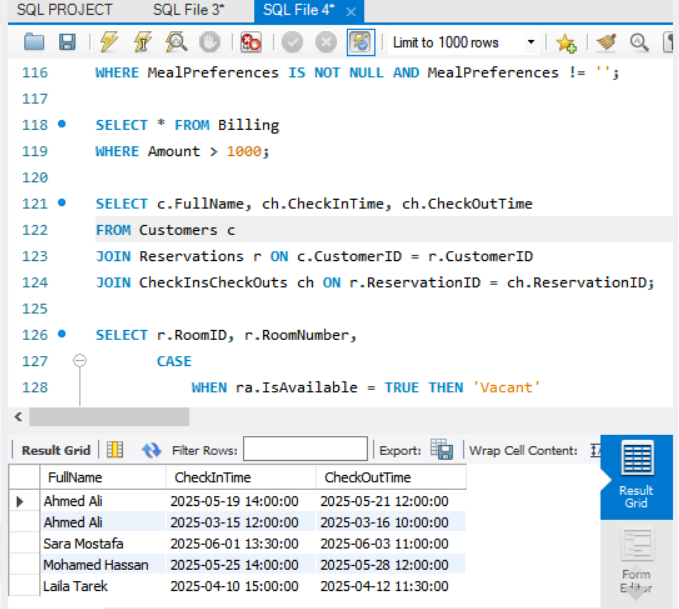
**SQL Query:**

SELECT c.FullName, ch.CheckInTime, ch.CheckOutTime

FROM Customers c

JOIN Reservations r ON c.CustomerID = r.CustomerID

JOIN CheckInsCheckOuts ch ON r.ReservationID = ch.ReservationID;

 **Result:**

5 entries with full check-in and check-out timestamps

**.** **Interpretation of Output:**  
Shows actual stay duration per customer, useful for auditing or tracking peak periods

🔹 **6. Show all rooms along with their occupancy status (occupied or vacant)**

**Description:**

Displays each room and whether it is currently marked as occupied or vacant

**SQL Query:**

**SELECT r.RoomID, r.RoomNumber,**

**CASE**

**WHEN ra.IsAvailable = TRUE THEN 'Vacant'**

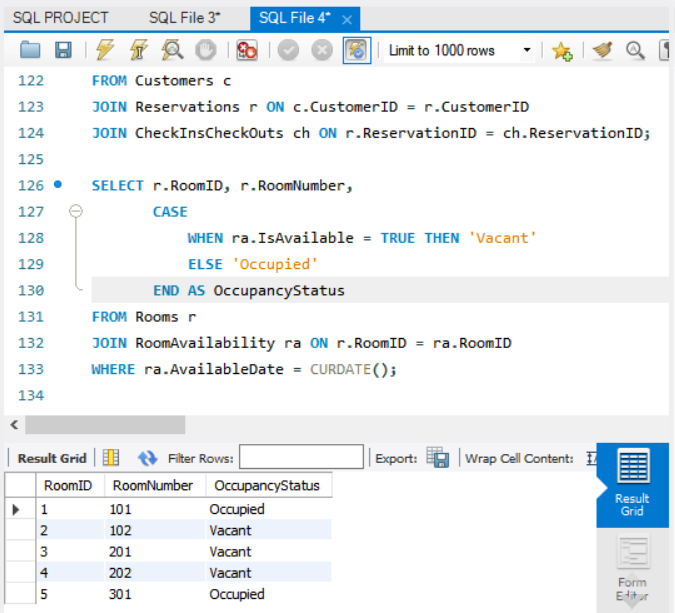
**ELSE 'Occupied'**

**END AS OccupancyStatus**

**FROM Rooms r**

**JOIN RoomAvailability ra ON r.RoomID = ra.RoomID**

**WHERE ra.AvailableDate = CURDATE();**

 **Result:**

rooms shown as Vacant, 2 as Occupied.

**.** **Interpretation of Output:**  
Provides a real-time view of which rooms are currently available (Vacant) or occupied. Essential for front-desk staff

🔹 7. List customers who have stayed more than 5 times

**Description:**

Counts reservations per customer and filters those who booked more than 5 times.

**SQL Query:**

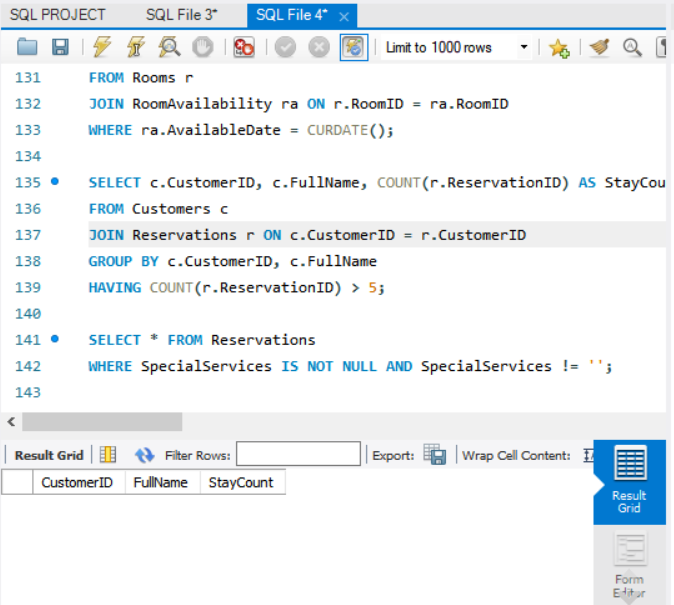
SELECT c.CustomerID, c.FullName, COUNT(r.ReservationID) AS StayCount

FROM Customers c

JOIN Reservations r ON c.CustomerID = r.CustomerID

GROUP BY c.CustomerID, c.FullName

HAVING COUNT(r.ReservationID) > 5;

 **Result:**

No result — no customer booked more than 5 times

**.** **Interpretation of Output:**  
Highlights repeat customers (loyal guests). You can reward or prioritize them

🔹 **8. Find reservations where the customer requested special services**

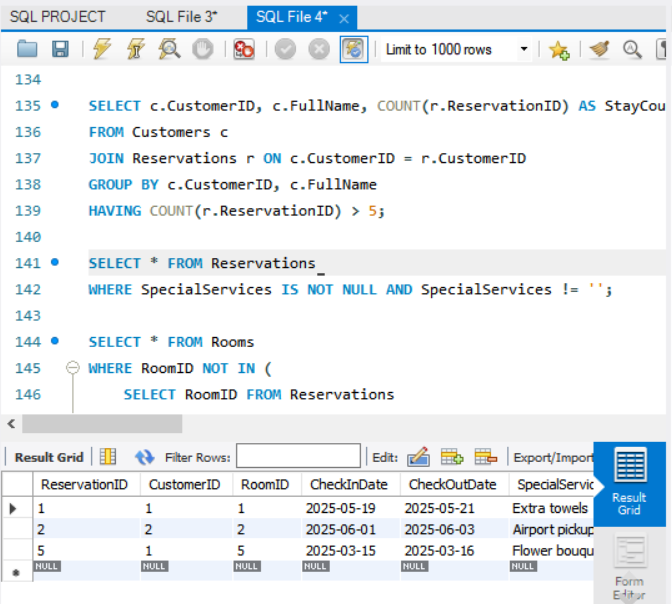
**Description:**

Shows reservations that include additional services requested by customers

**SQL Query:**

SELECT \* FROM Reservations

WHERE SpecialServices IS NOT NULL AND SpecialServices != '';

 **Result:**

3 reservations returned with services like: Extra towels, Airport pickup, Flower bouquet

**.** **Interpretation of Output:**  
Identifies guests who requested additional services like airport pickup, room decoration, etc

🔹 **9. List rooms that have never been booked**

**Description:**

Displays rooms that do not exist in the Reservations table

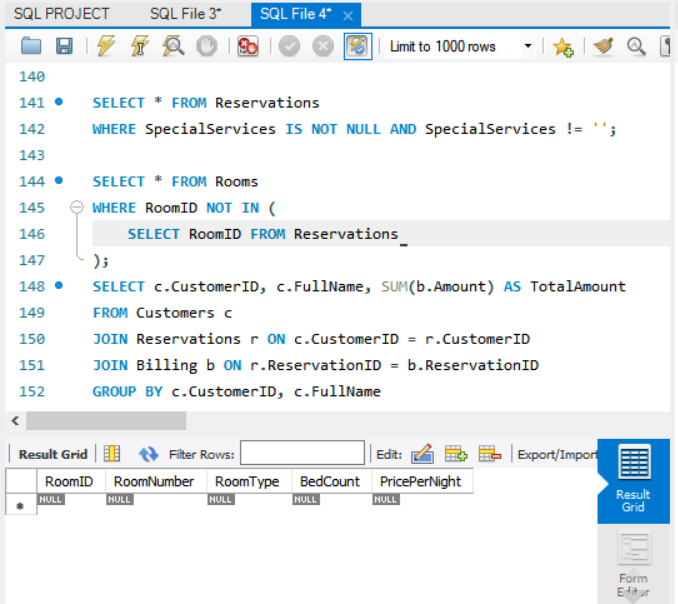
**SQL Query:**

SELECT \* FROM Rooms

WHERE RoomID NOT IN (

SELECT RoomID FROM Reservations

);

 **Result:**

No rooms found — all rooms have at least one booking.

**. Interpretation of Output:**  
Helps detect underutilized rooms — either not shown to guests or not in demand

🔹 **10. Find customers whose total billing amount is higher than the average total billing amount**

**Description:**

Calculates total spending per customer and compares it with the overall average.

**SQL Query:**

SELECT c.CustomerID, c.FullName, SUM(b.Amount) AS TotalAmount

FROM Customers c

JOIN Reservations r ON c.CustomerID = r.CustomerID

JOIN Billing b ON r.ReservationID = b.ReservationID

GROUP BY c.CustomerID, c.FullName

HAVING SUM(b.Amount) > (

SELECT AVG(Total) FROM (

SELECT SUM(b2.Amount) AS Total

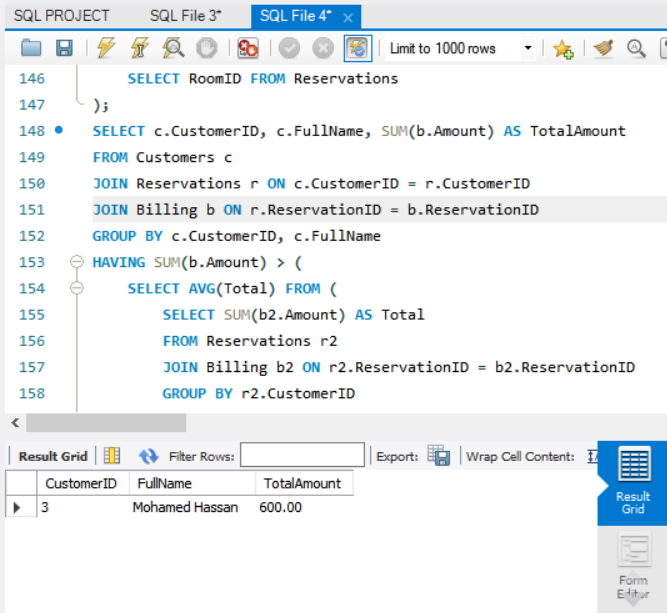
FROM Reservations r2

JOIN Billing b2 ON r2.ReservationID = b2.ReservationID

GROUP BY r2.CustomerID

) AS CustomerTotals

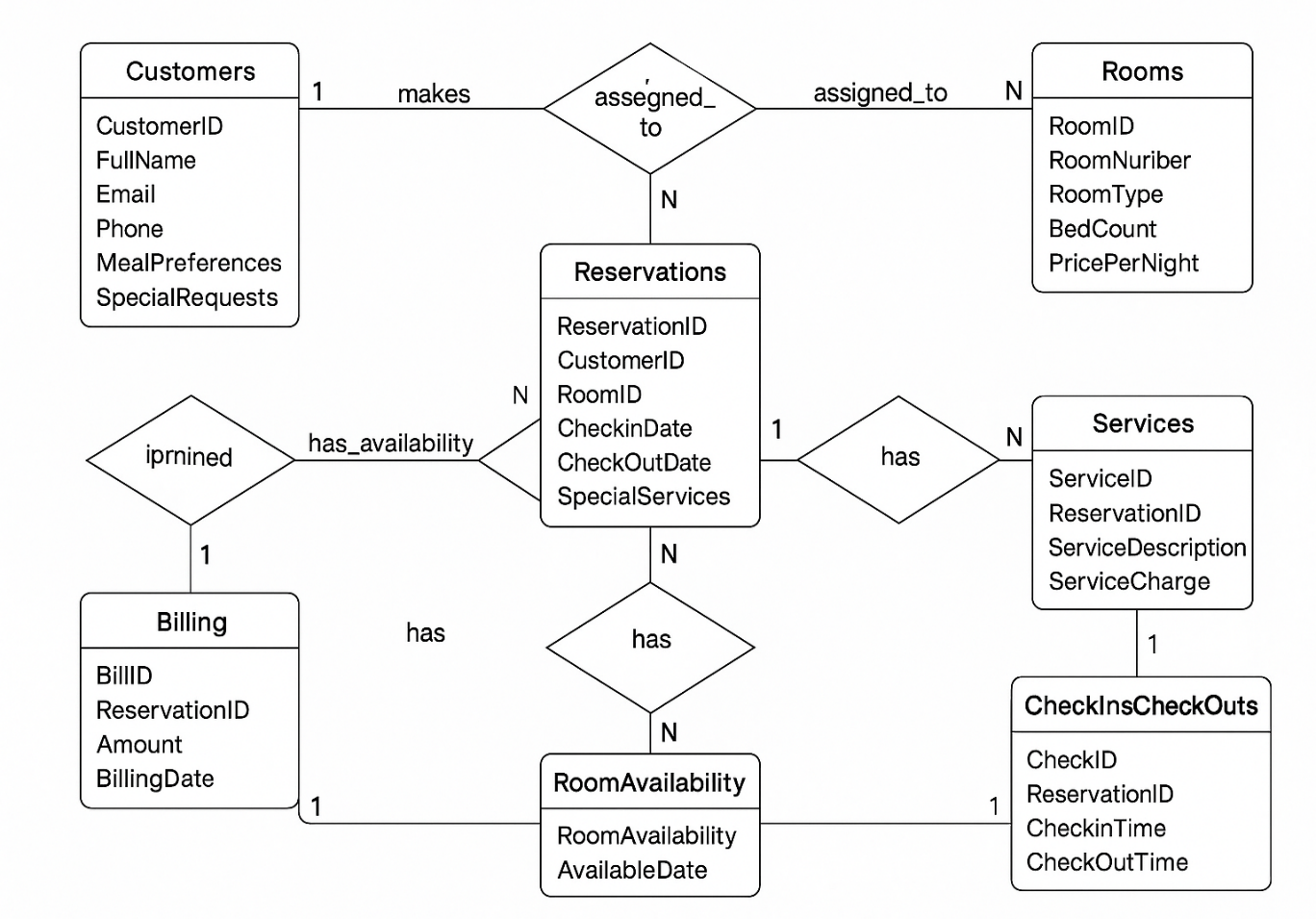
);

** Result:**

**O**nly Mohamed Hassan has a total above the average: $600.

**.** **Interpretation of Output:**  
Returns the high-paying customers — valuable for marketing or providing premium offers

. ERD (Entity-Relationship Diagram):



.key Attributes :

**RoomID(PK), CoustomerID(PK),** **ReservationID(PK),** **BillID(PK),**

**RoomAvailabilityID(PK), ServiceID(PK), CheckID(PK)**