DBMS LAB ASSIGNMENT – 5

NAME: DATLA TARUN ANJANEYA VARMA

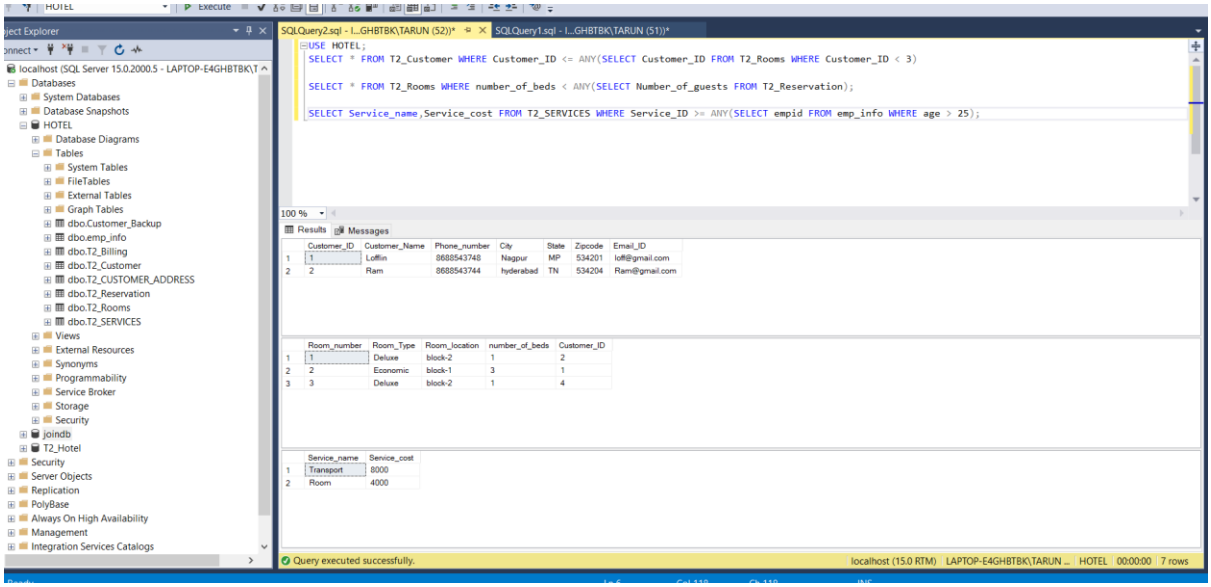
REG NO: 19BCS036

GROUP NO: 2

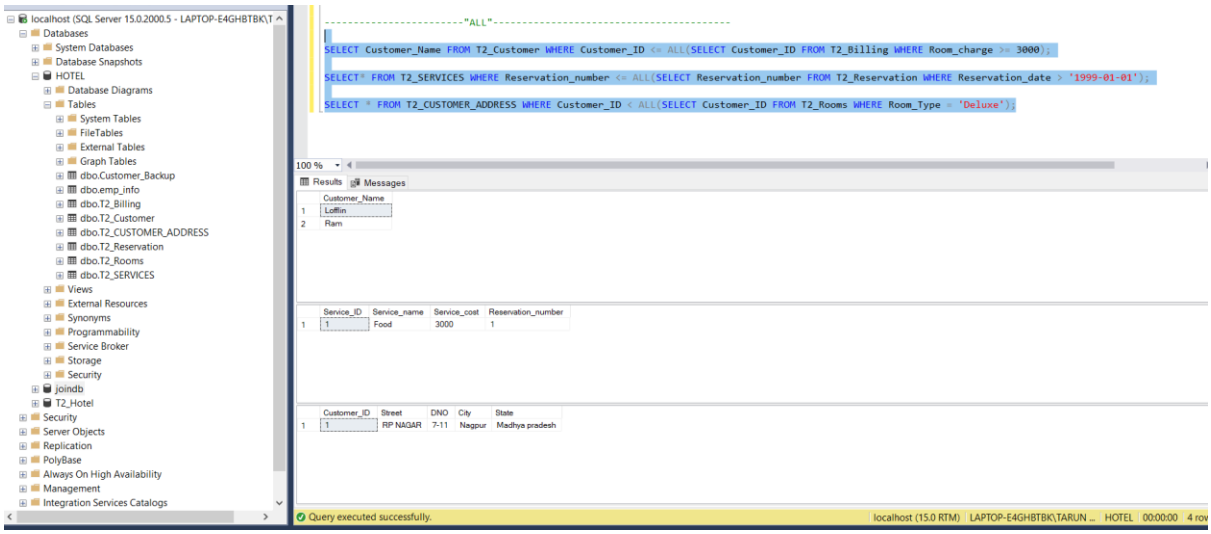
DATABASE NAME: HOTEL

Q1) Illustrate logical ANY, ALL and LIKE operator- the queries should be relevant to your respective databases 3 queries for each operator. One query explaining the difference between ANY and ALL

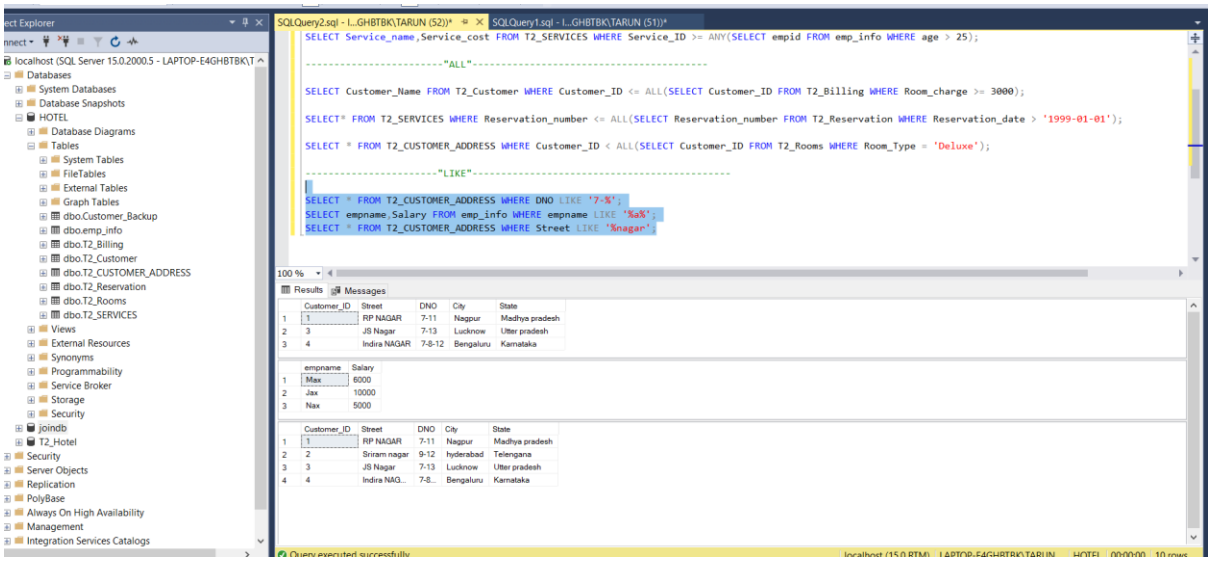
QUERIES FOR "ANY"



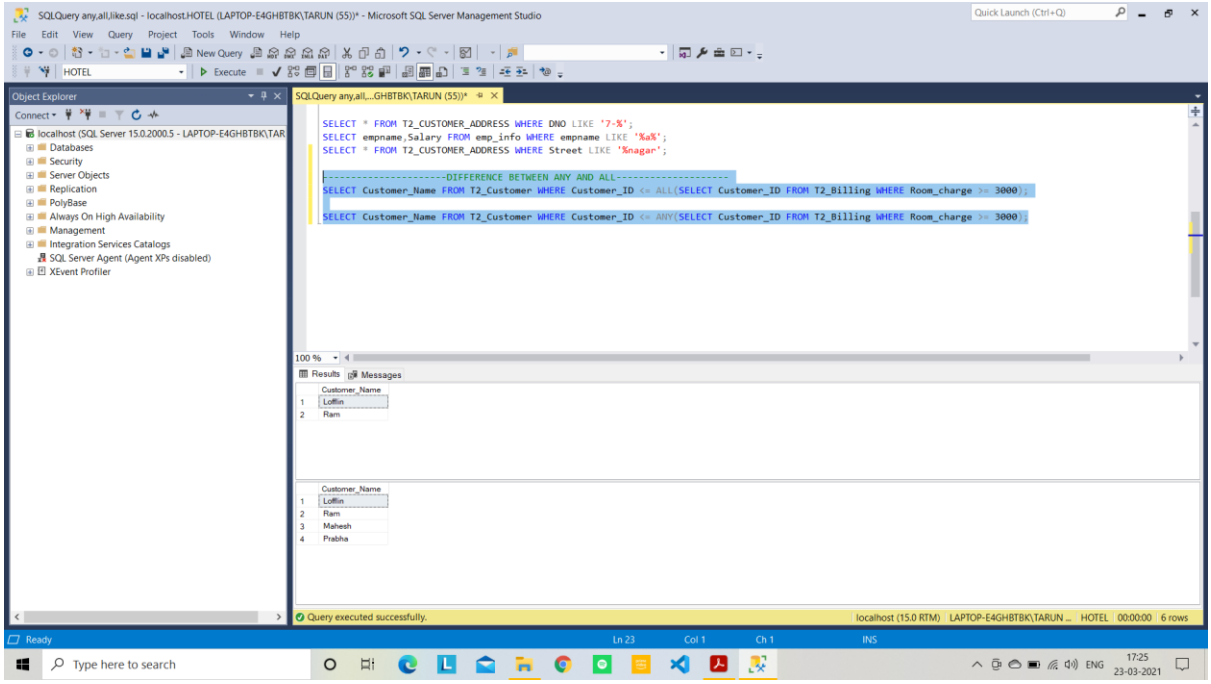
QUERIES FOR “All”



QUERIES FOR “Like”



QUERY FOR DIFFERENCE BETWEEN ANY AND ALL



Q2) One query for each Aggregate function.

The aggregate functions are MIN(), MAX(), COUNT(), AVG(), SUM()

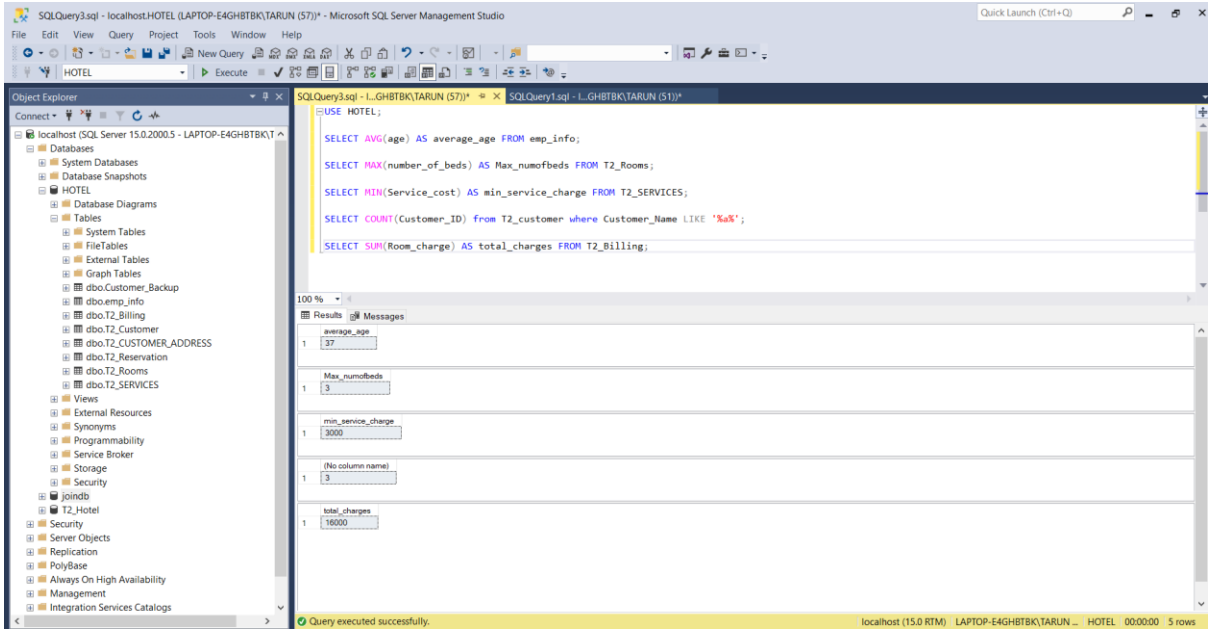
AVG() – return the average of the set

MIN() – returns the minimum value in a set

MAX() – returns the maximum value in set

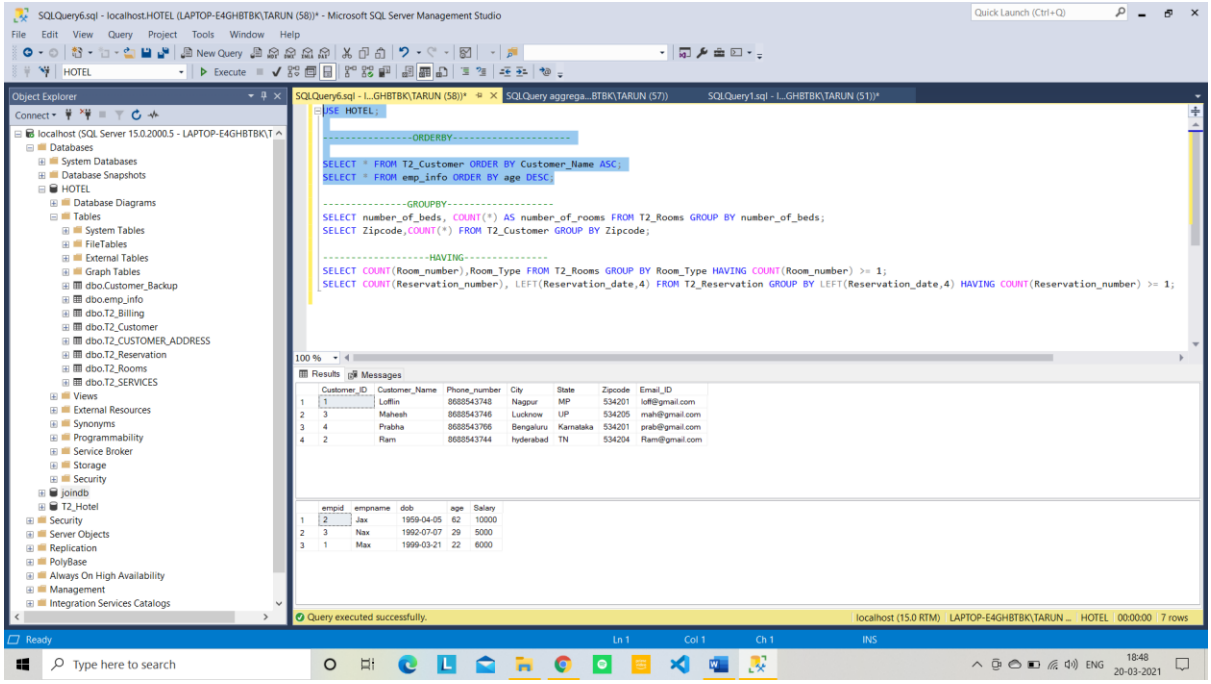
SUM() – returns the sum of all distinct values of a set

COUNT() – returns the number of items in a set

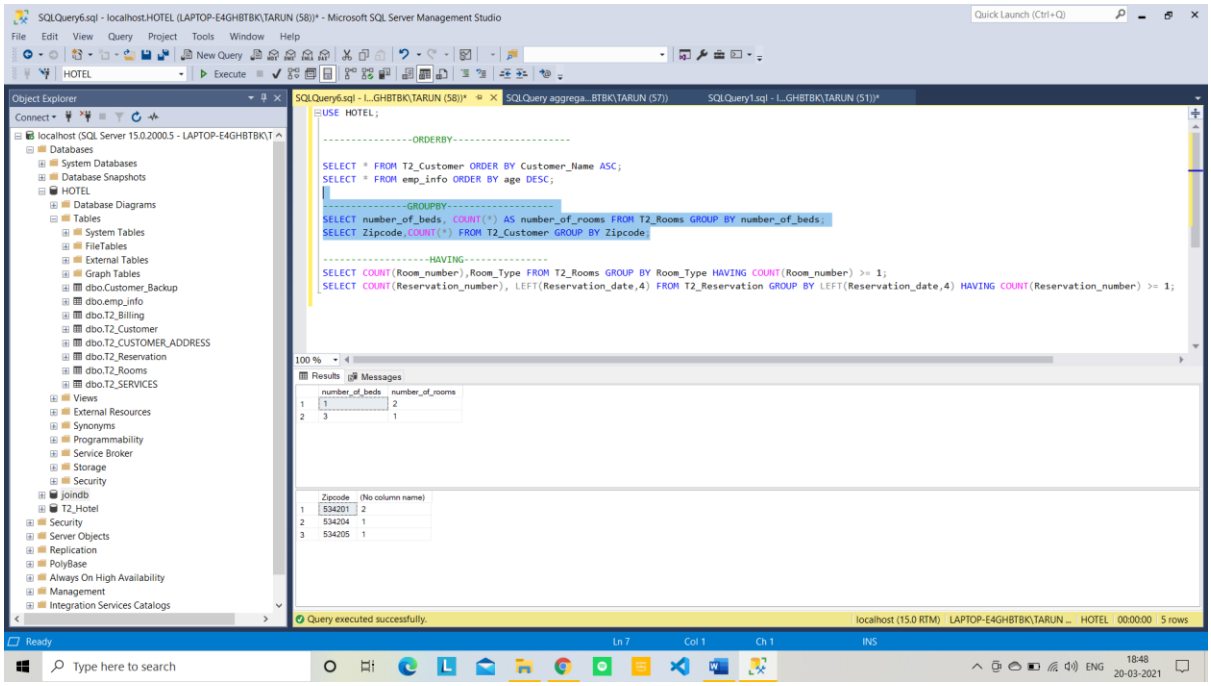


Q3) Illustrate the usage of order by, group by and having clause (2 queries for each case)

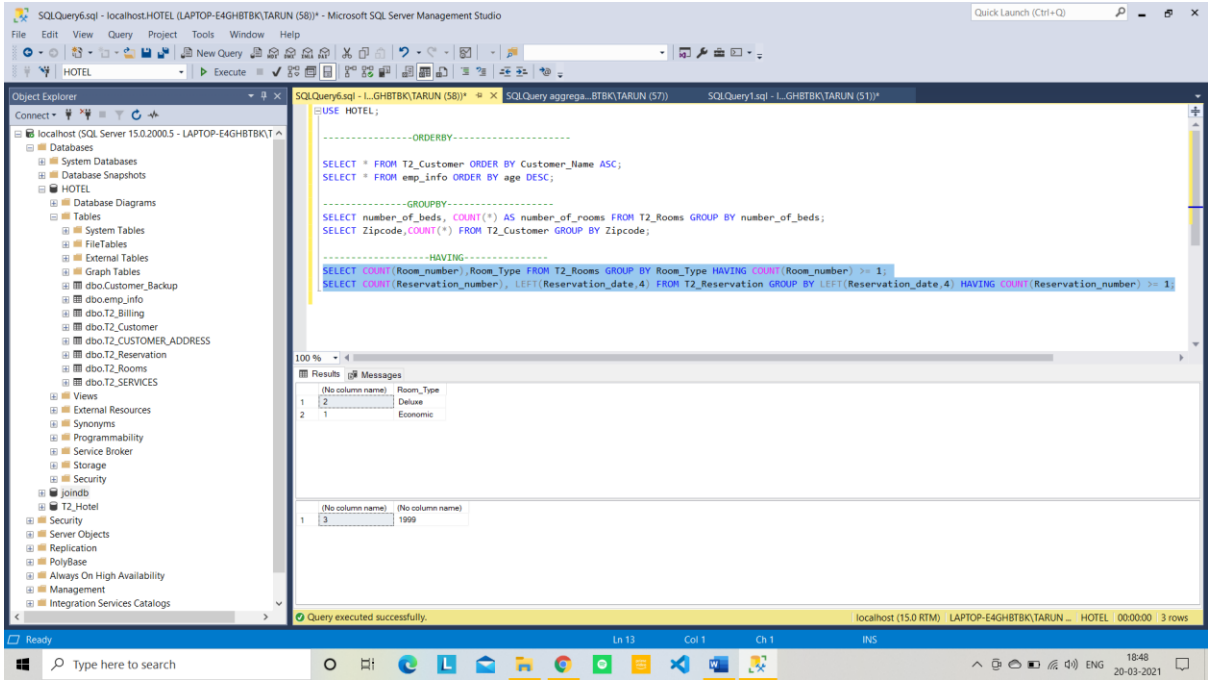
ORDER BY



GROUP BY

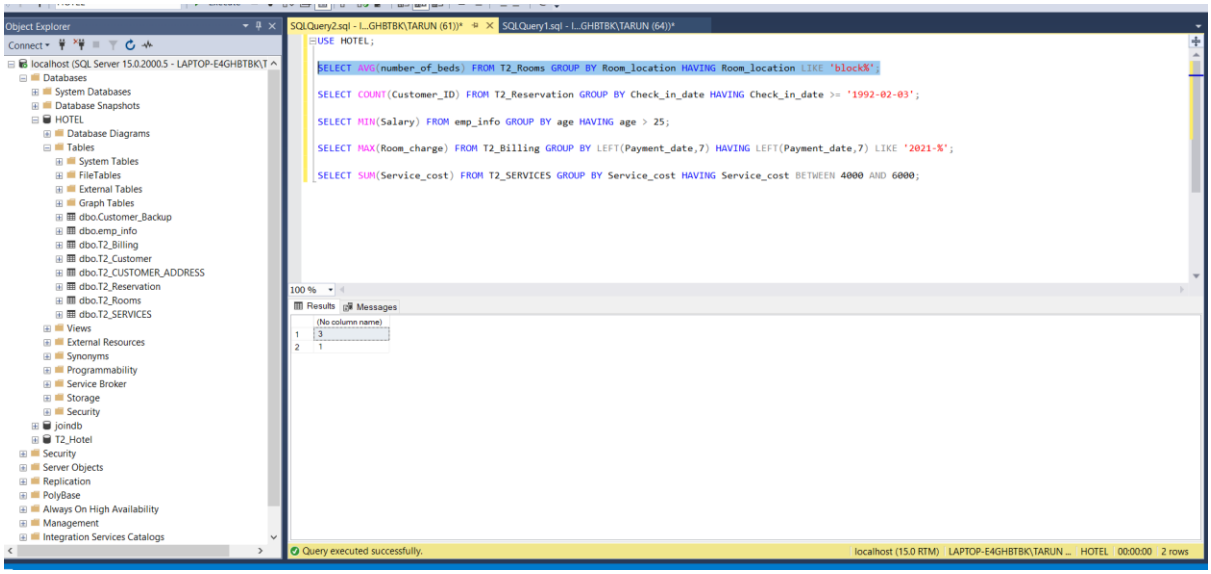


HAVING CLAUSE

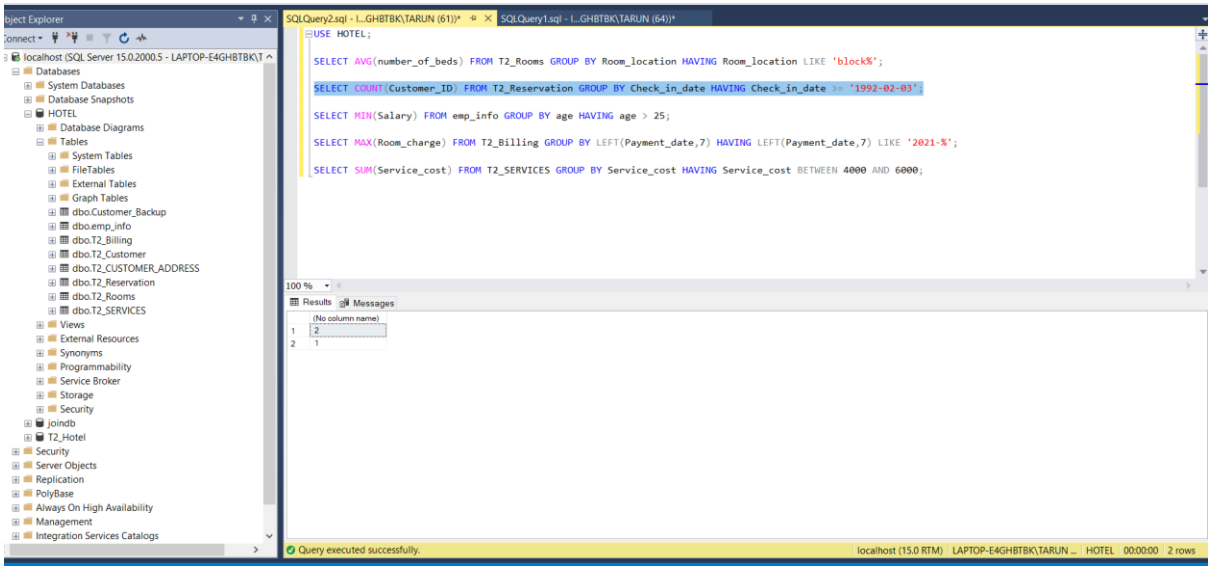


Q4) Use Aggregate function with group by and having

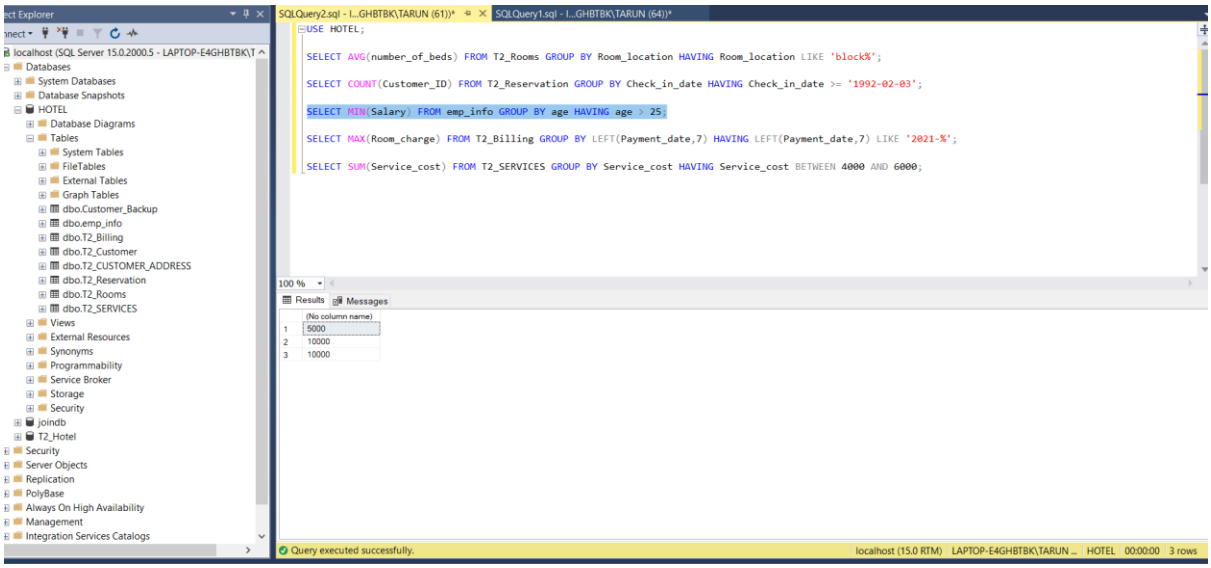
AVG():



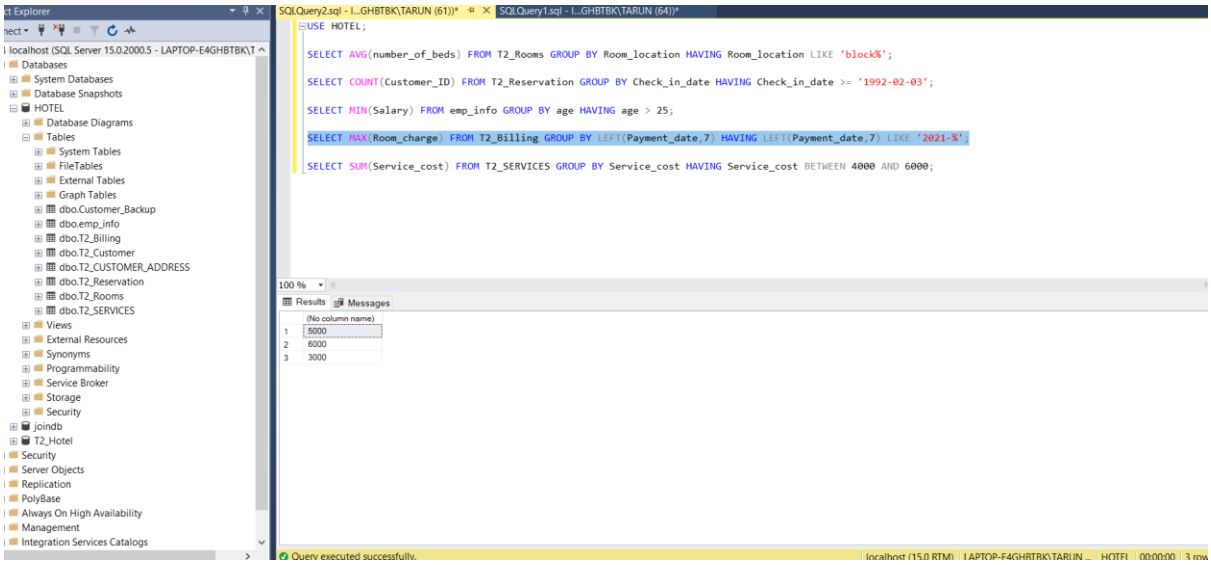
COUNT():


USE HOTEL;
SELECT AVG(number_of_beds) FROM T2_Rooms GROUP BY Room_location HAVING Room_location LIKE 'block%';
SELECT COUNT(Customer_ID) FROM T2_Reservation GROUP BY Check_in_date HAVING Check_in_date >= '1992-02-03';
SELECT MIN(Salary) FROM emp_info GROUP BY age HAVING age > 25;
SELECT MAX(Room_charge) FROM T2_Billing GROUP BY LEFT(Payment_date,7) HAVING LEFT(Payment_date,7) LIKE '2021-%';
SELECT SUM(Service_cost) FROM T2_SERVICES GROUP BY Service_cost HAVING Service_cost BETWEEN 4000 AND 6000;
Results
Messages
(No column name)
1 2
2 1
Query executed successfully.

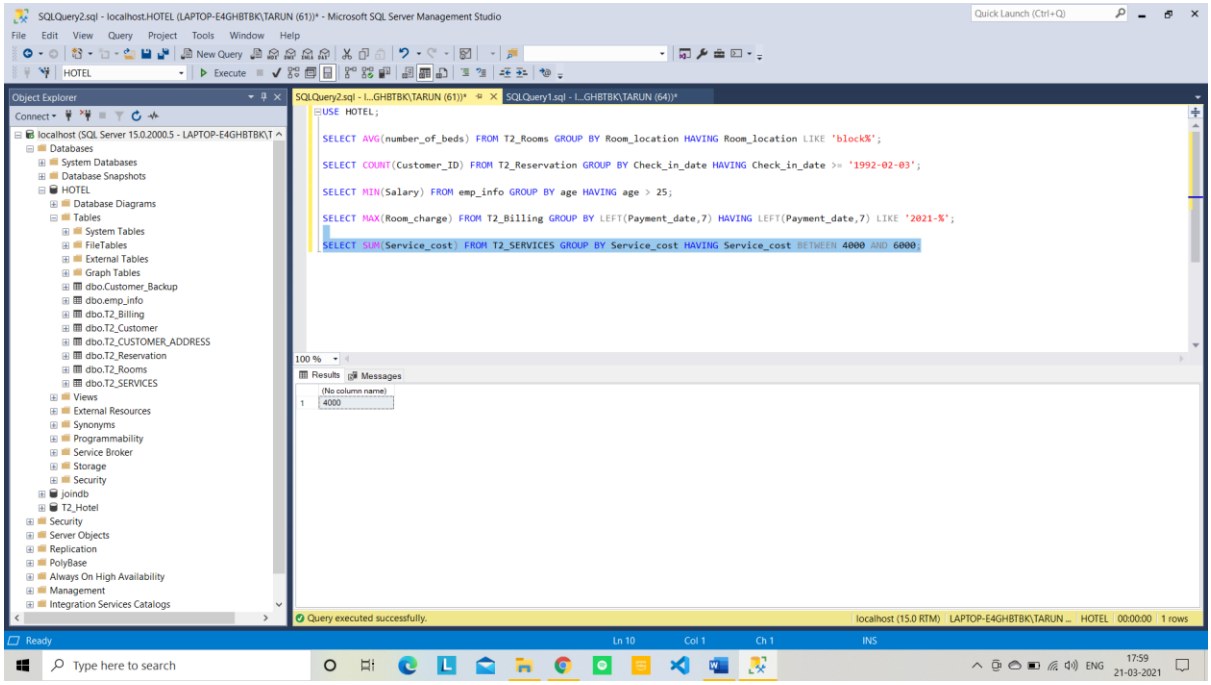
MIN():


USE HOTEL;
SELECT AVG(number_of_beds) FROM T2_Rooms GROUP BY Room_location HAVING Room_location LIKE 'block%';
SELECT COUNT(Customer_ID) FROM T2_Reservation GROUP BY Check_in_date HAVING Check_in_date >= '1992-02-03';
SELECT MIN(Salary) FROM emp_info GROUP BY age HAVING age > 25;
SELECT MAX(Room_charge) FROM T2_Billing GROUP BY LEFT(Payment_date,7) HAVING LEFT(Payment_date,7) LIKE '2021-%';
SELECT SUM(Service_cost) FROM T2_SERVICES GROUP BY Service_cost HAVING Service_cost BETWEEN 4000 AND 6000;
Results
Messages
(No column name)
1 5000
2 10000
3 10000
Query executed successfully.

MAX():

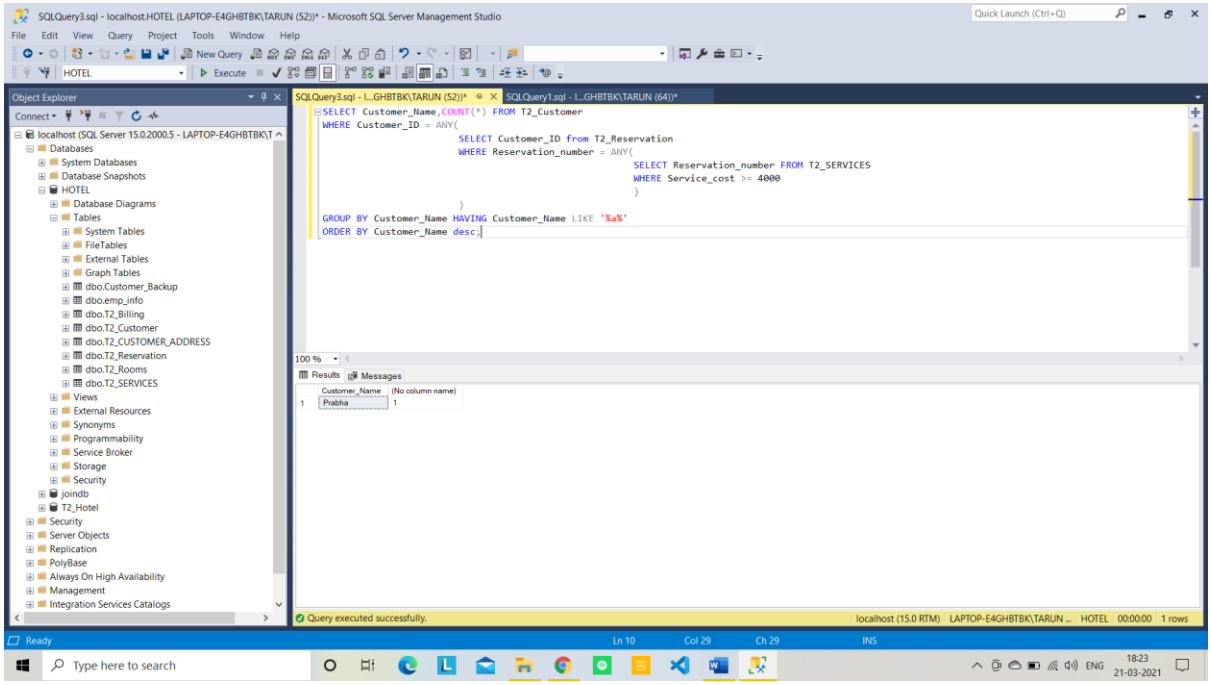

USE HOTEL;
SELECT AVG(number_of_beds) FROM T2_Rooms GROUP BY Room_location HAVING Room_location LIKE 'block%';
SELECT COUNT(Customer_ID) FROM T2_Reservation GROUP BY Check_in_date HAVING Check_in_date >= '1992-02-03';
SELECT MIN(Salary) FROM emp_info GROUP BY age HAVING age > 25;
SELECT MAX(Room_charge) FROM T2_Billing GROUP BY LEFT(Payment_date,7) HAVING LEFT(Payment_date,7) LIKE '2021-%';
SELECT SUM(Service_cost) FROM T2_SERVICES GROUP BY Service_cost HAVING Service_cost BETWEEN 4000 AND 6000;
(No column name)
1 5000
2 6000
3 3000
Query executed successfully.

SUM():


SQLQuery2.sql - localhost.HOTEL (LAPTOP-E4GHBTBK\TARUN (61))* - Microsoft SQL Server Management Studio
USE HOTEL;
SELECT AVG(number_of_beds) FROM T2_Rooms GROUP BY Room_location HAVING Room_location LIKE 'block%';
SELECT COUNT(Customer_ID) FROM T2_Reservation GROUP BY Check_in_date HAVING Check_in_date >= '1992-02-03';
SELECT MIN(Salary) FROM emp_info GROUP BY age HAVING age > 25;
SELECT MAX(Room_charge) FROM T2_Billing GROUP BY LEFT(Payment_date,7) HAVING LEFT(Payment_date,7) LIKE '2021-%';
SELECT SUM(Service_cost) FROM T2_SERVICES GROUP BY Service_cost HAVING Service_cost BETWEEN 4000 AND 6000;
(No column name)
1 4000
Query executed successfully.

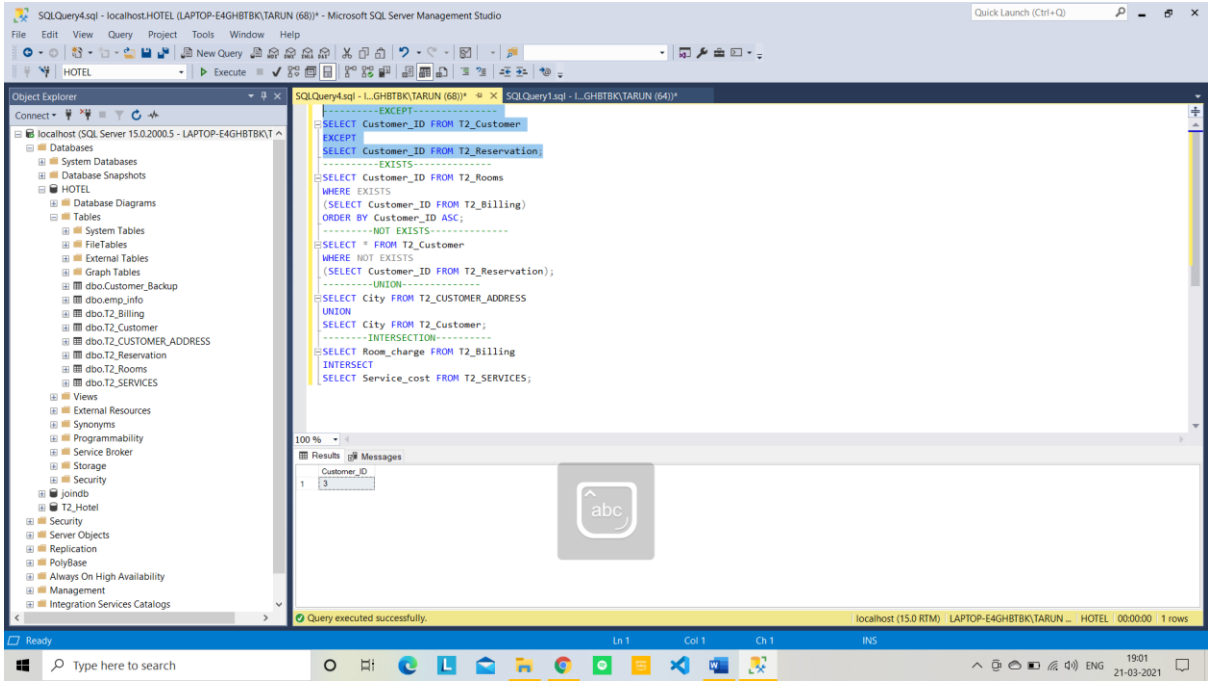
Q5) Write at least 3 nested queries using order by, group by and having clause.

QUERY:

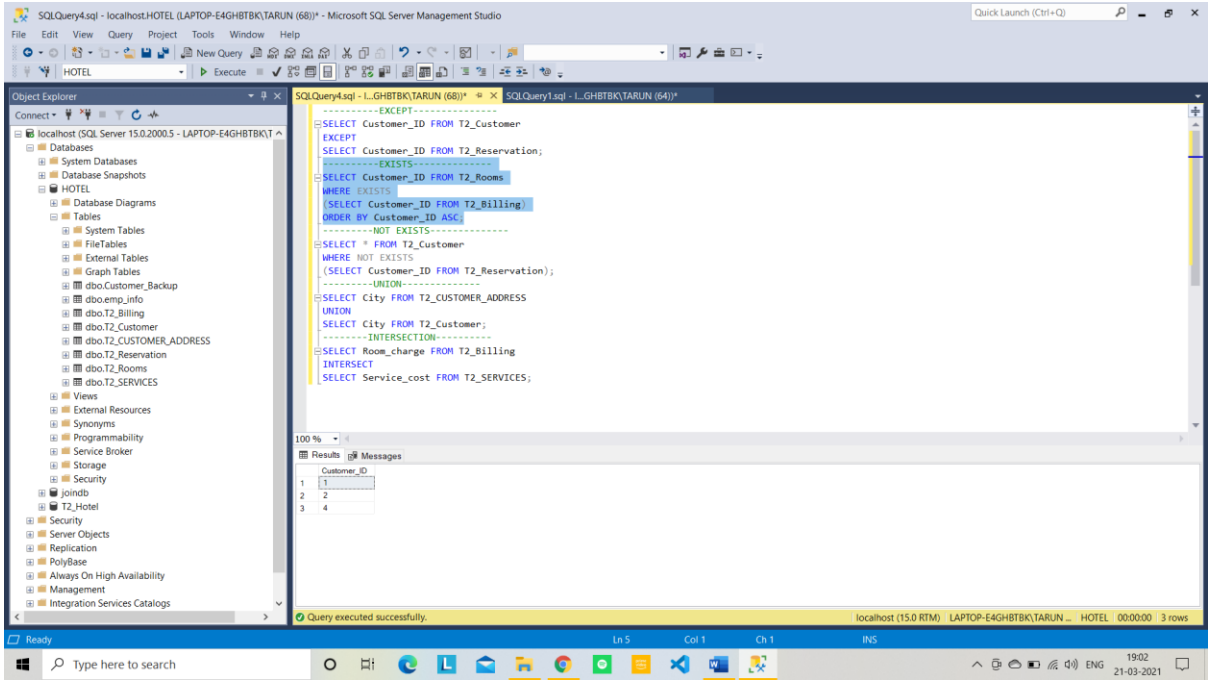


Q6) Illustrate the Usage of Except, Exists, Not Exists, Union, Intersection

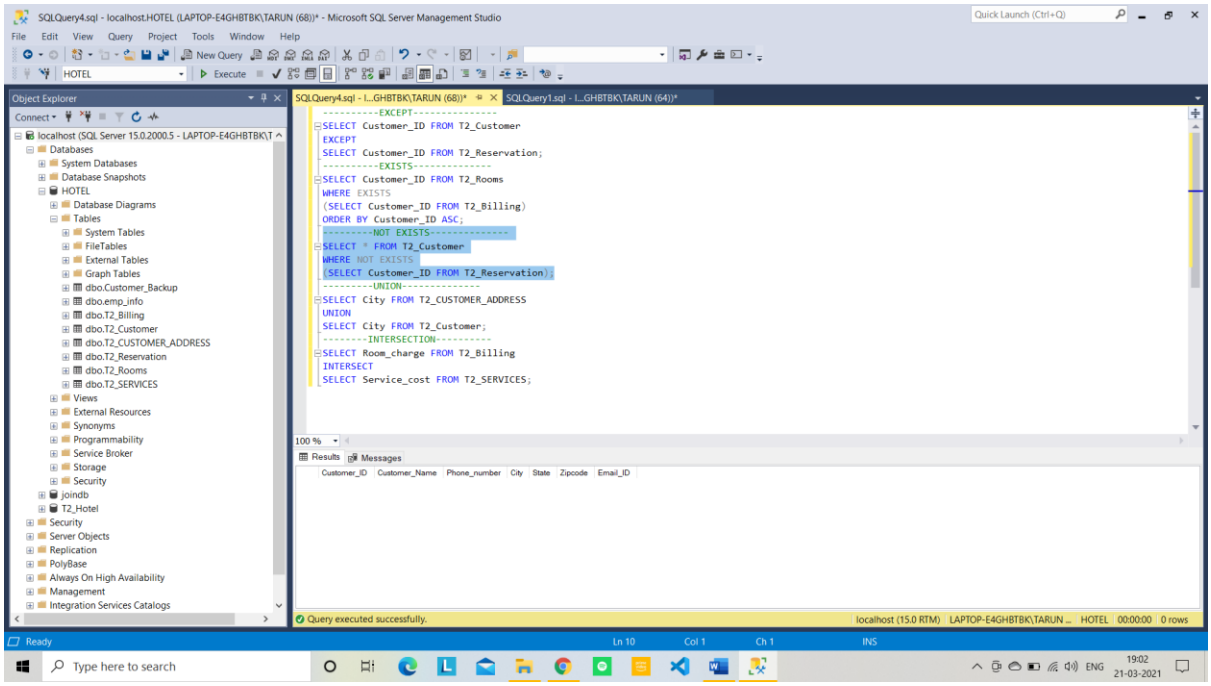
EXCEPT():



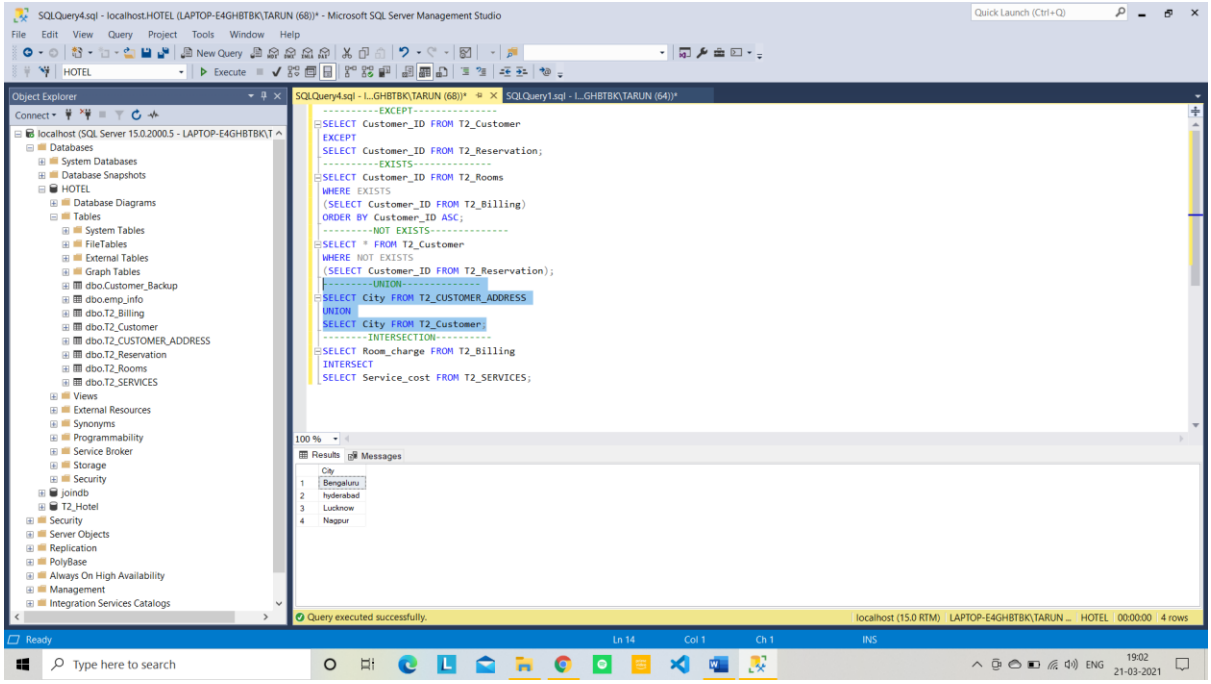
EXISTS():



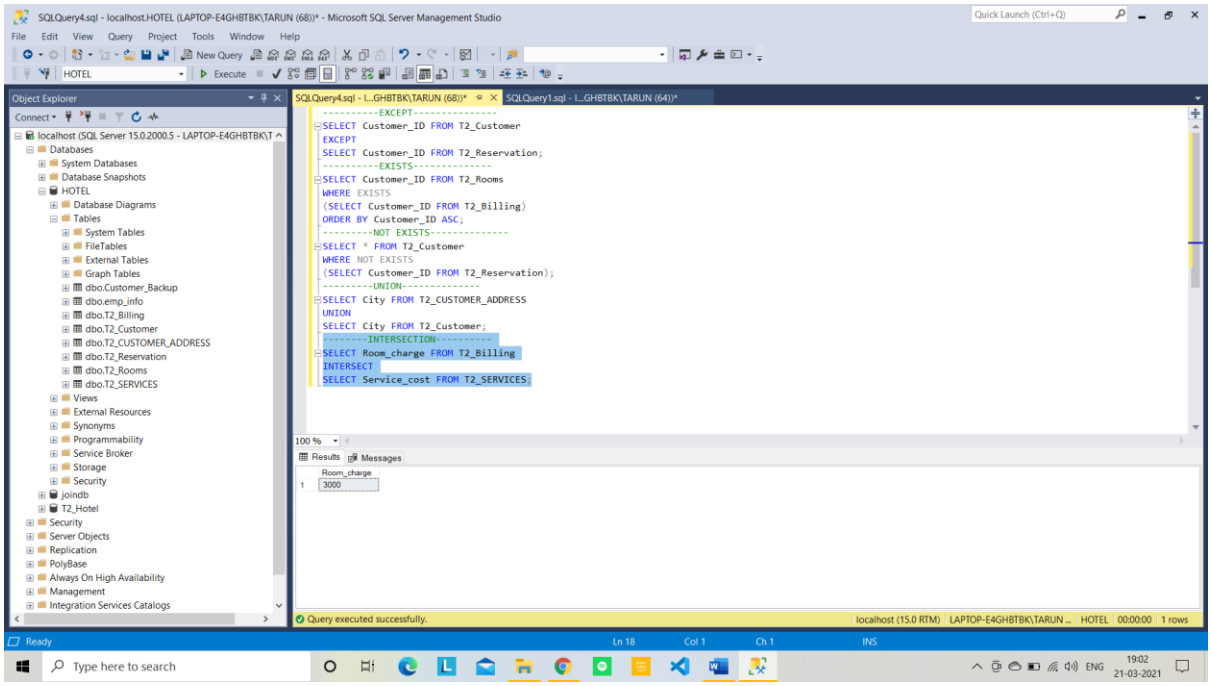
NOT EXISTS():



UNION():

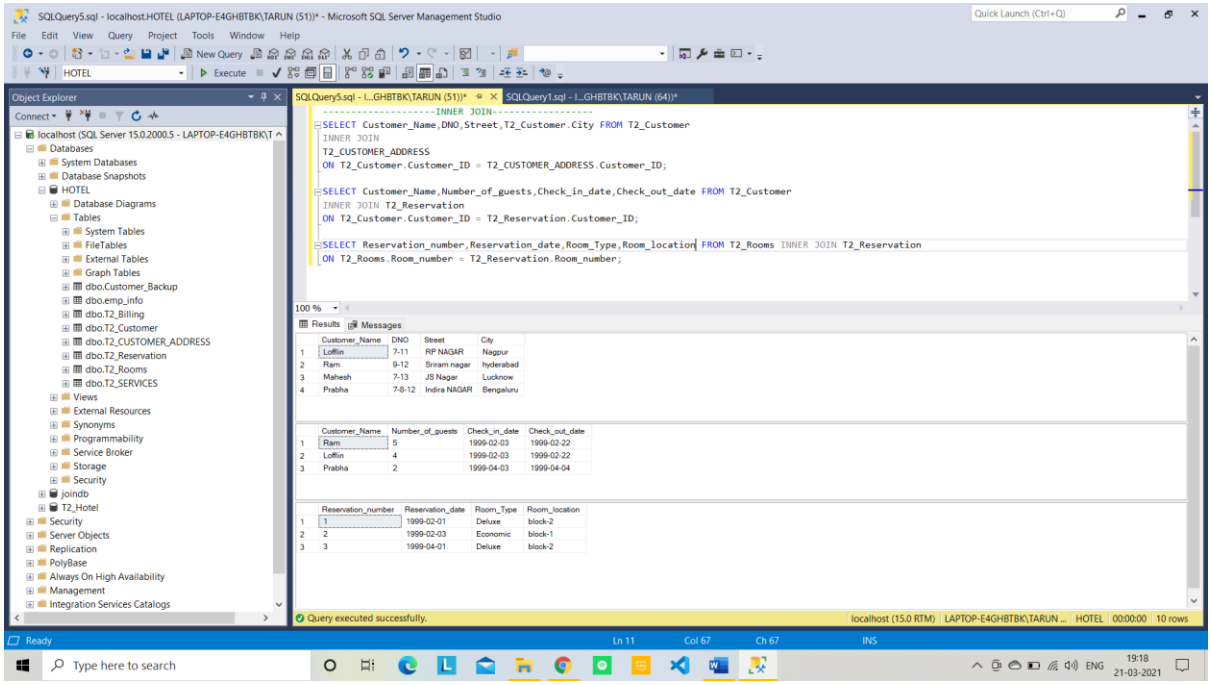


INTERSECT:

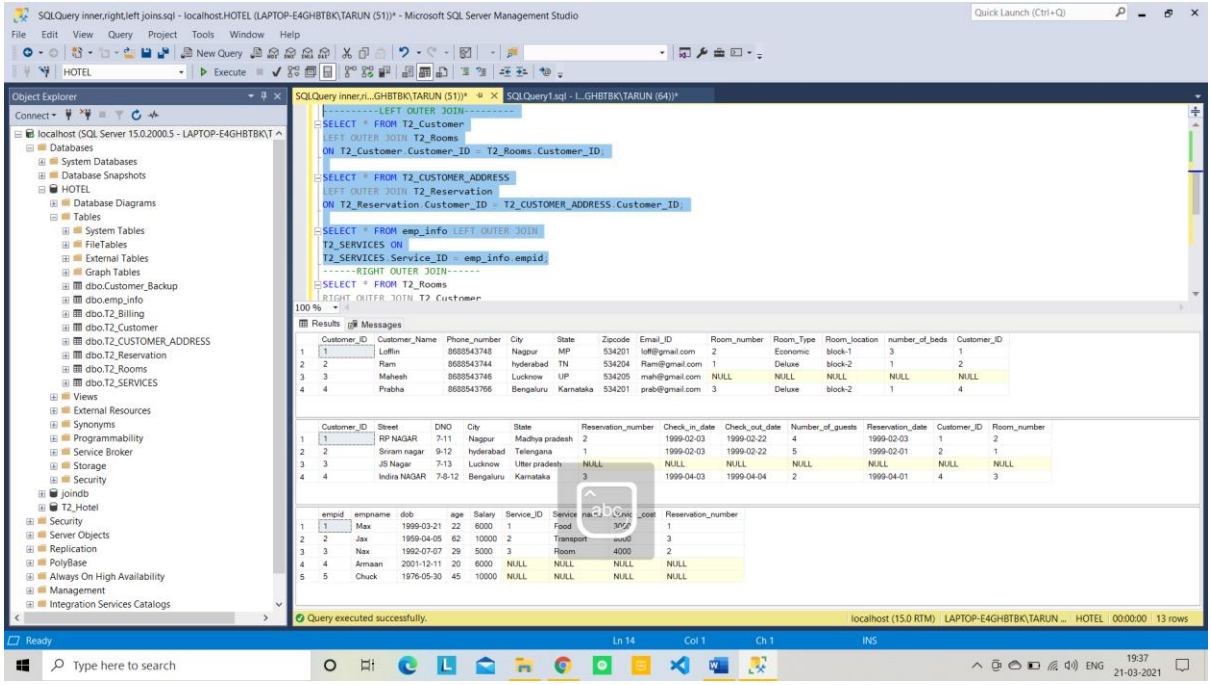


Q7) INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN- 3 queries for each instance

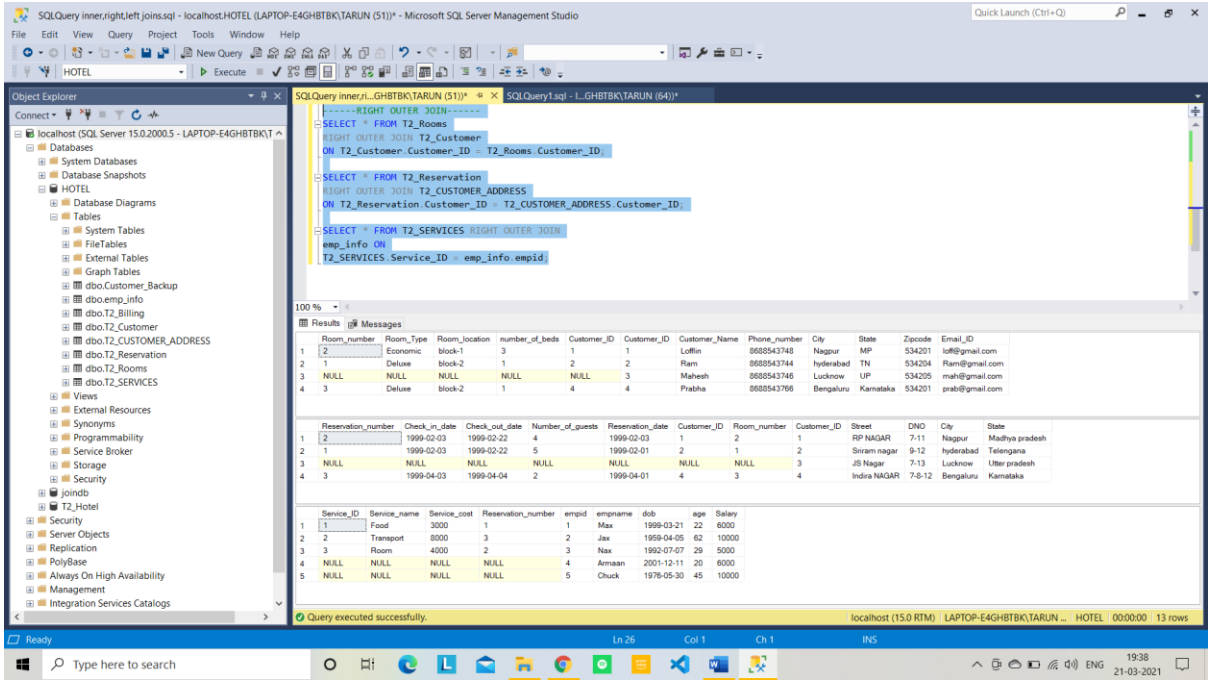
INNER JOIN



LEFT OUTER JOIN



RIGHT OUTER JOIN



Q8) Use all the above condition in JOIN as well.

QUERY: