DBMS_LAB_ASSIGNMENT_5

Name: Rohit Sagar Shinde

Roll No: 19BCS099

Team No: 2

Database: 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.

Query:

```
USE T2_HOTEL;

/*ANY*/
SELECT * FROM T2_Customer WHERE Customer_ID <= ANY(SELECT Customer_ID FROM T2_Rooms
WHERE Customer_ID < 3);

SELECT * FROM T2_Rooms WHERE number_of_beds < ANY(SELECT Number_of_guests FROM
T2_Reservation);

SELECT Service_name,Service_cost FROM T2_SERVICES WHERE Service_ID >= ANY(SELECT empid
FROM emp_info WHERE age > 25);

/*ALL*/
SELECT Customer_Name FROM T2_Customer WHERE Customer_ID <= ALL(SELECT Customer_ID FROM
T2_Billing WHERE Room_charge >= 3000);

SELECT* FROM T2_SERVICES WHERE Reservation_number <= ALL(SELECT Reservation_number
FROM T2_Reservation WHERE Reservation_date > '1999-01-01');

SELECT * FROM T2_CUSTOMER_ADDRESS WHERE Customer_ID < ALL(SELECT Customer_ID FROM
T2_Rooms WHERE Room_Type = 'Deluxe');

/*LIKE*/
SELECT * FROM T2_CUSTOMER_ADDRESS WHERE DNO LIKE '7-%';
SELECT empname,Salary FROM emp_info WHERE empname LIKE '%a%';
SELECT * FROM T2_CUSTOMER_ADDRESS WHERE Street LIKE '%nagar';

/*Difference between ALL and ANY*/

SELECT Customer_Name FROM T2_Customer WHERE Customer_ID <= ALL(SELECT Customer_ID FROM
T2_Billing WHERE Room_charge >=3000);

SELECT Customer_Name FROM T2_Customer WHERE Customer_ID <= ANY(SELECT Customer_ID FROM
T2_Billing WHERE Room_charge >=3000);
```

Output:

ANY –

Results | Messages

	Customer_ID	Customer_Name	Phone_number	City	State	Zipcode	Email_ID
1	1	Lofflin	8688543748	Nagpur	MP	534201	loff@gmail.com
2	2	Ram	8688543744	hyderabad	TN	534204	Ram@gmail.com

	Room_number	Room_Type	Room_location	number_of_beds	Customer_ID
1	1	Deluxe	block-2	1	2
2	2	Economic	block-1	3	1
3	3	Deluxe	block-2	1	4

	Service_name	Service_cost
1	Transport	8000
2	Room	4000

Query executed successfully.

ALL –

Results | Messages

	Customer_Name
1	Lofflin
2	Ram

	Service_ID	Service_name	Service_cost	Reservation_number
1	1	Food	3000	1

	Customer_ID	Street	DNO	City	State
1	1	RP NAGAR	7-11	Nagpur	Madhya pradesh

Query executed successfully.

LIKE –

Results | Messages

	Customer_ID	Street	DNO	City	State
1	1	RP NAGAR	7-11	Nagpur	Madhya pradesh
2	3	JS Nagar	7-13	Lucknow	Utter pradesh
3	4	Indira NAGAR	7-8-12	Bengaluru	Karnataka

	empname	Salary
1	Max	6000
2	Jax	10000
3	Nax	5000

	Customer_ID	Street	DNO	City	State
1	1	RP NAGAR	7-11	Nagpur	Madhya pradesh
2	2	Sriram nagar	9-12	hyderabad	Telengana
3	3	JS Nagar	7-13	Lucknow	Utter pradesh
4	4	Indira NAG...	7-8...	Bengaluru	Karnataka

Query executed successfully.

ALL vs ANY –

Results | Messages

	Customer_Name
1	Lofflin
2	Ram

	Customer_Name
1	Lofflin
2	Ram
3	Mahesh
4	Prabha

Query executed successfully.

Q2) One query for each Aggregate Function.

There are 5 Aggregate Functions, MIN (), MAX (), AVG (), SUM (), COUNT ().

Query:

```
USE T2_HOTEL;
SELECT AVG(age) AS average_age FROM emp_info;
SELECT MAX(number_of_beds) AS Max_numofbeds FROM T2_Rooms;
SELECT MIN(Service_cost) AS min_service_charge FROM T2_SERVICES;
SELECT COUNT(Customer_ID) from T2_customer where Customer_Name LIKE '%a%';
SELECT SUM(Room_charge) AS total_charges FROM T2_Billing;
```

Output:

Results | Messages

	average_age
1	37

	Max_numofbeds
1	3

	min_service_charge
1	3000

	(No column name)
1	3

	total_charges
1	16000

Query executed successfully.

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

Query:

```
USE T2_HOTEL;

/*ORDER BY*/
SELECT * FROM T2_Customer ORDER BY Customer_Name ASC;
SELECT * FROM emp_info ORDER BY age DESC;

/*GROUP BY*/
SELECT number_of_beds, COUNT(*) AS number_of_rooms FROM T2_Rooms GROUP BY
number_of_beds;
SELECT Zipcode,COUNT(*) FROM T2_Customer GROUP BY Zipcode;

/*HAVING CLAUSE*/
SELECT COUNT(Room_number),Room_Type FROM T2_Rooms GROUP BY Room_Type HAVING
COUNT(Room_number) >= 1;
SELECT COUNT(Reservation_number), LEFT(Reservation_date,4) FROM T2_Reservation GROUP
BY LEFT(Reservation_date,4) HAVING COUNT(Reservation_number) >= 1;
```

Output:

ORDER BY –

Results | Messages

	Customer_ID	Customer_Name	Phone_number	City	State	Zipcode	Email_ID
1	1	Lofflin	8688543748	Nagpur	MP	534201	loff@gmail.com
2	3	Mahesh	8688543746	Lucknow	UP	534205	mah@gmail.com
3	4	Prabha	8688543766	Bengaluru	Karnataka	534201	prab@gmail.com
4	2	Ram	8688543744	hyderabad	TN	534204	Ram@gmail.com

	empid	empname	dob	age	Salary
1	2	Jax	1959-04-05	62	10000
2	3	Nax	1992-07-07	29	5000
3	1	Max	1999-03-21	22	6000

Query executed successfully.

GROUP BY –

Results | Messages

	number_of_beds	number_of_rooms
1	1	2
2	3	1

	Zipcode	(No column name)
1	534201	2
2	534204	1
3	534205	1

Query executed successfully.

HAVING CLAUSE –

Results | Messages

	(No column name)	Room_Type
1	2	Deluxe
2	1	Economic

	(No column name)	(No column name)
1	3	1999

Query executed successfully.

Q4) Use Aggregate function with group by and having.

Query:

```
USE T2_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;
```

Output:

AVG () –

Results | Messages

	(No column name)
1	3
2	1

Query executed successfully.

COUNT () –

Results | Messages

	(No column name)
1	2
2	1

Query executed successfully.

MIN () –

Results | Messages

	(No column name)
1	5000
2	10000
3	10000

Query executed successfully.

MAX () –

Results | Messages

	(No column name)
1	5000
2	6000
3	3000

Query executed successfully.

SUM () –

Results | Messages

	(No column name)
1	4000

Query executed successfully.

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

Query:

```
USE T2_HOTEL;

SELECT Customer_Name,COUNT(*) FROM T2_Customer
WHERE Customer_ID = ANY(
       SELECT Customer_ID from T2_Reservation
       WHERE Reservation_number = ANY(
           SELECT Reservation_number FROM T2_SERVICES
              WHERE Service_cost >= 4000 ))

GROUP BY Customer_Name HAVING Customer_Name LIKE '%a%'
ORDER BY Customer_Name desc;
```

Output:

Results | Messages

	Customer_Name	(No column name)
1	Prabha	1

Query executed successfully.

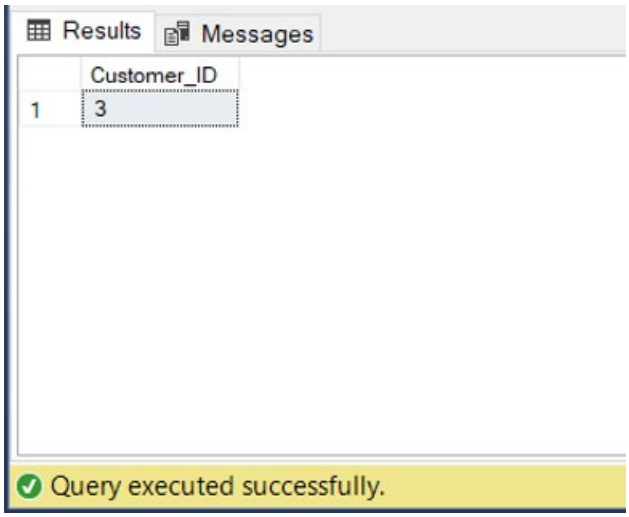
Q6) Illustrate the Usage of Except, Exists, Not Exists, Union, Intersection.

Query:

```
USE T2_HOTEL;

/*EXCEPT*/
SELECT Customer_ID FROM T2_Customer
EXCEPT
SELECT Customer_ID FROM T2_Reservation;

/*EXISTS*/
SELECT Customer_ID FROM T2_Rooms
WHERE EXISTS
(SELECT Customer_ID FROM T2_Billing)
ORDER BY Customer_ID ASC;

/*NOT EXISTS*/
SELECT * FROM T2_Customer
WHERE NOT EXISTS
(SELECT Customer_ID FROM T2_Reservation);

/*UNION*/
SELECT City FROM T2_CUSTOMER_ADDRESS
UNION
SELECT City FROM T2_Customer;

/*INTERSECTION*/
SELECT Room_charge FROM T2_Billing
INTERSECT
SELECT Service_cost FROM T2_SERVICES;
```

Output:

EXCEPT () –



EXISTS () –

Results | Messages

	Customer_ID
1	1
2	2
3	4

Query executed successfully.

NOT EXISTS () –

Results | Messages

Customer_ID	Customer_Name	Phone_number	City	State	Zipcode	Email_ID

Query executed successfully.

UNION –

Results | Messages

	City
1	Bengaluru
2	hyderabad
3	Lucknow
4	Nagpur

Query executed successfully.

INTERSECTION –

Results | Messages

	Room_charge
1	3000

Query executed successfully.

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

Query:

```
USE T2_HOTEL;

/*INNER JOIN*/
SELECT Customer_Name,DNO,Street,T2_Customer.City FROM T2_Customer
INNER JOIN T2_CUSTOMER_ADDRESS
ON T2_Customer.Customer_ID = T2_CUSTOMER_ADDRESS.Customer_ID;

SELECT Customer_Name,Number_of_guests,Check_in_date,Check_out_date FROM T2_Customer
INNER JOIN T2_Reservation
ON T2_Customer.Customer_ID = T2_Reservation.Customer_ID;

SELECT Reservation_number,Reservation_date,Room_Type,Room_location FROM T2_Rooms
INNER JOIN T2_Reservation
ON T2_Rooms.Room_number = T2_Reservation.Room_number;


/*LEFT OUTER JOIN*/
SELECT * FROM T2_Customer
LEFT OUTER JOIN T2_Rooms
ON T2_Customer.Customer_ID = T2_Rooms.Customer_ID;

SELECT * FROM T2_CUSTOMER_ADDRESS
LEFT OUTER JOIN T2_Reservation
ON T2_Reservation.Customer_ID = T2_CUSTOMER_ADDRESS.Customer_ID;

SELECT * FROM emp_info
LEFT OUTER JOIN T2_SERVICES
ON T2_SERVICES.Service_ID = emp_info.empid;


/*RIGHT OUTER JOIN*/
SELECT * FROM T2_Rooms
RIGHT OUTER JOIN T2_Customer
ON T2_Customer.Customer_ID = T2_Rooms.Customer_ID;
```

```sql
SELECT * FROM T2_Reservation
RIGHT OUTER JOIN T2_CUSTOMER_ADDRESS
ON T2_Reservation.Customer_ID = T2_CUSTOMER_ADDRESS.Customer_ID;

SELECT * FROM T2_SERVICES
RIGHT OUTER JOIN emp_info
ON T2_SERVICES.Service_ID = emp_info.empid;
```

Output:

INNER JOIN –

Results | Messages

	Customer_Name	DNO	Street	City
1	Lofflin	7-11	RP NAGAR	Nagpur
2	Ram	9-12	Sriram nagar	hyderabad
3	Mahesh	7-13	JS Nagar	Lucknow
4	Prabha	7-8-12	Indira NAGAR	Bengaluru

	Customer_Name	Number_of_guests	Check_in_date	Check_out_date
1	Ram	5	1999-02-03	1999-02-22
2	Lofflin	4	1999-02-03	1999-02-22
3	Prabha	2	1999-04-03	1999-04-04

	Reservation_number	Reservation_date	Room_Type	Room_location
1	1	1999-02-01	Deluxe	block-2
2	2	1999-02-03	Economic	block-1
3	3	1999-04-01	Deluxe	block-2

Query executed successfully.

LEFT OUTER JOIN –

Results | Messages

	Customer_ID	Customer_Name	Phone_number	City	State	Zipcode	Email_ID	Room_number	Room_Type	Room_location	number_of_beds	Customer_ID
1	1	Lofflin	8688543748	Nagpur	MP	534201	loff@gmail.com	2	Economic	block-1	3	1
2	2	Ram	8688543744	hyderabad	TN	534204	Ram@gmail.com	1	Deluxe	block-2	1	2
3	3	Mahesh	8688543746	Lucknow	UP	534205	mah@gmail.com	NULL	NULL	NULL	NULL	NULL
4	4	Prabha	8688543766	Bengaluru	Karnataka	534201	prab@gmail.com	3	Deluxe	block-2	1	4

	Customer_ID	Street	DNO	City	State	Reservation_number	Check_in_date	Check_out_date	Number_of_guests	Reservation_date	Customer_ID	Room_number
1	1	RP NAGAR	7-11	Nagpur	Madhya pradesh	2	1999-02-03	1999-02-22	4	1999-02-03	1	2
2	2	Sriram nagar	9-12	hyderabad	Telengana	1	1999-02-03	1999-02-22	5	1999-02-01	2	1
3	3	JS Nagar	7-13	Lucknow	Utter pradesh	NULL	NULL	NULL	NULL	NULL	NULL	NULL
4	4	Indira NAGAR	7-8-12	Bengaluru	Karnataka	3	1999-04-03	1999-04-04	2	1999-04-01	4	3

	empid	empname	dob	age	Salary	Service_ID	Service name	Service_cost	Reservation_number
1	1	Max	1999-03-21	22	6000	1	Food	3000	1
2	2	Jax	1959-04-05	62	10000	2	Transport	6000	3
3	3	Nax	1992-07-07	29	5000	3	Room	4000	2
4	4	Armaan	2001-12-11	20	6000	NULL	NULL	NULL	NULL
5	5	Chuck	1976-05-30	45	10000	NULL	NULL	NULL	NULL

RIGHT OUTER JOIN –

Results | Messages

	Room_number	Room_Type	Room_location	number_of_beds	Customer_ID	Customer_ID	Customer_Name	Phone_number	City	State	Zipcode	Email_ID
1	2	Economic	block-1	3	1	1	Lofflin	8688543748	Nagpur	MP	534201	loff@gmail.com
2	1	Deluxe	block-2	1	2	2	Ram	8688543744	hyderabad	TN	534204	Ram@gmail.com
3	NULL	NULL	NULL	NULL	NULL	3	Mahesh	8688543746	Lucknow	UP	534205	mah@gmail.com
4	3	Deluxe	block-2	1	4	4	Prabha	8688543766	Bengaluru	Karnataka	534201	prab@gmail.com

	Reservation_number	Check_in_date	Check_out_date	Number_of_guests	Reservation_date	Customer_ID	Room_number	Customer_ID	Street	DNO	City	State
1	2	1999-02-03	1999-02-22	4	1999-02-03	1	2	1	RP NAGAR	7-11	Nagpur	Madhya pradesh
2	1	1999-02-03	1999-02-22	5	1999-02-01	2	1	2	Sriram nagar	9-12	hyderabad	Telengana
3	NULL	NULL	NULL	NULL	NULL	NULL	NULL	3	JS Nagar	7-13	Lucknow	Utter pradesh
4	3	1999-04-03	1999-04-04	2	1999-04-01	4	3	4	Indira NAGAR	7-8-12	Bengaluru	Karnataka

	Service_ID	Service_name	Service_cost	Reservation_number	empid	empname	dob	age	Salary
1	1	Food	3000	1	1	Max	1999-03-21	22	6000
2	2	Transport	8000	3	2	Jax	1959-04-05	62	10000
3	3	Room	4000	2	3	Nax	1992-07-07	29	5000
4	NULL	NULL	NULL	NULL	4	Armaan	2001-12-11	20	6000
5	NULL	NULL	NULL	NULL	5	Chuck	1976-05-30	45	10000

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

Query:

```
USE T2_HOTEL;

SELECT COUNT(*) ,Room_location FROM T2_Rooms
JOIN T2_Reservation
ON T2_Rooms.Customer_ID = T2_Reservation.Customer_ID
JOIN T2_Customer
ON T2_Rooms.Customer_ID = T2_Customer.Customer_ID
GROUP BY Room_location
HAVING Room_location LIKE 'block%'
ORDER BY Room_location DESC;
```

Output:

Results | Messages

	(No column name)	Room_location
1	2	block-2
2	1	block-1

Query executed successfully.