

## DBMS PRACTICAL-7

## PRE-LAB

- 1) What is the output of following query  
SELECT A.id FROM A WHERE A.age > ALL (SELECT B.age FROM B  
WHERE B.name = "ABC")

DBMS Lab-7

190031249

P.Mohith

prelab

1. SELECT A.id FROM A WHERE A.age > ALL (SELECT  
B.age FROM B WHERE B.name = "ABC")

The output of the above query will be

Id's of person whose age is greater than  
"ABC"

- 2) What is the purpose of WHERE Clause in mysql?

2. WHERE

The WHERE clause is used to filter records.

The WHERE clause is used to extract only  
those records that fulfill a specified  
condition.

- 3) Differentiate between AND, OR and NOT operators in mysql

3. AND :- AND operator is used to set multiple conditions with the WHERE clause, alongside SELECT, UPDATE or DELETE

Ex:- SELECT \* FROM emp WHERE sal < 100000  
AND age > 25

OR :- OR operator is also used to display filter records based on more than one condition. The OR operator displays record if any of the conditions separated by OR is true



Scanned with  
CamScanner

NOT :-

NOT operator displays record if the condition is NOT TRUE

5) Database table by name Loan\_Records is given below.

Borrower	Bank_Manager	Loan_Amount
Ramesh	Sunderajan	10000.00
Suresh	Ramgopal	5000.00
Mahesh	Sunderajan	7000.00

SELECT Count(\*) FROM ( (SELECT Borrower, Bank\_Manager FROM Loan\_Records) AS S  
NATURAL JOIN (SELECT Bank\_Manager, Loan\_Amount FROM Loan\_Records) AS T );

What is the output of the following SQL query?

5. SELECT count (\*) FROM ( (SELECT Borrower,  
Bank\_Manager FROM Loan\_Records) AS S  
NATURAL JOIN (SELECT Bank\_Manager, Loan-  
Amount FROM Loan\_Records) AS T );

output for the above query is

5

- i State the difference between (Null Value Function) nvl() & Coalesce().

NVL and COALESCE are used to achieve the same functionality of providing a default value in case the column returns a null. NVL accepts only 2 Arguments whereas COALESCE can take multiple arguments.

- 4) Consider a database table T containing two columns X and Y each of type integer. After the creation of the table, one record (X=1, Y=1) is inserted in the table. Let MX and MY denote the respective maximum values of X and Y among all records in the table at any point in time. Using MX and MY, new records are inserted in the table 128 times with X and Y values being  $MX+1$ ,  $2*MY+1$  respectively. It may be noted that each time after the insertion, values of MX and MY change. What will be the output of the following SQL query after the steps mentioned above are carried out? [ ] Consider a database table T containing two columns X and Y each of type integer. After the creation of the table, one record (X=1, Y=1) is inserted in the table. Let MX and MY denote the respective maximum values of X and Y among all records in the table at any point in time. Using MX and MY, new records are inserted in the table 128 times with X and Y values being  $MX+1$ ,  $2*MY+1$  respectively. It may be noted that each time after the insertion, values of MX and MY change. What will be the output of the following SQL query after the steps mentioned above are carried out?  
SELECT Y FROM T WHERE X=7;

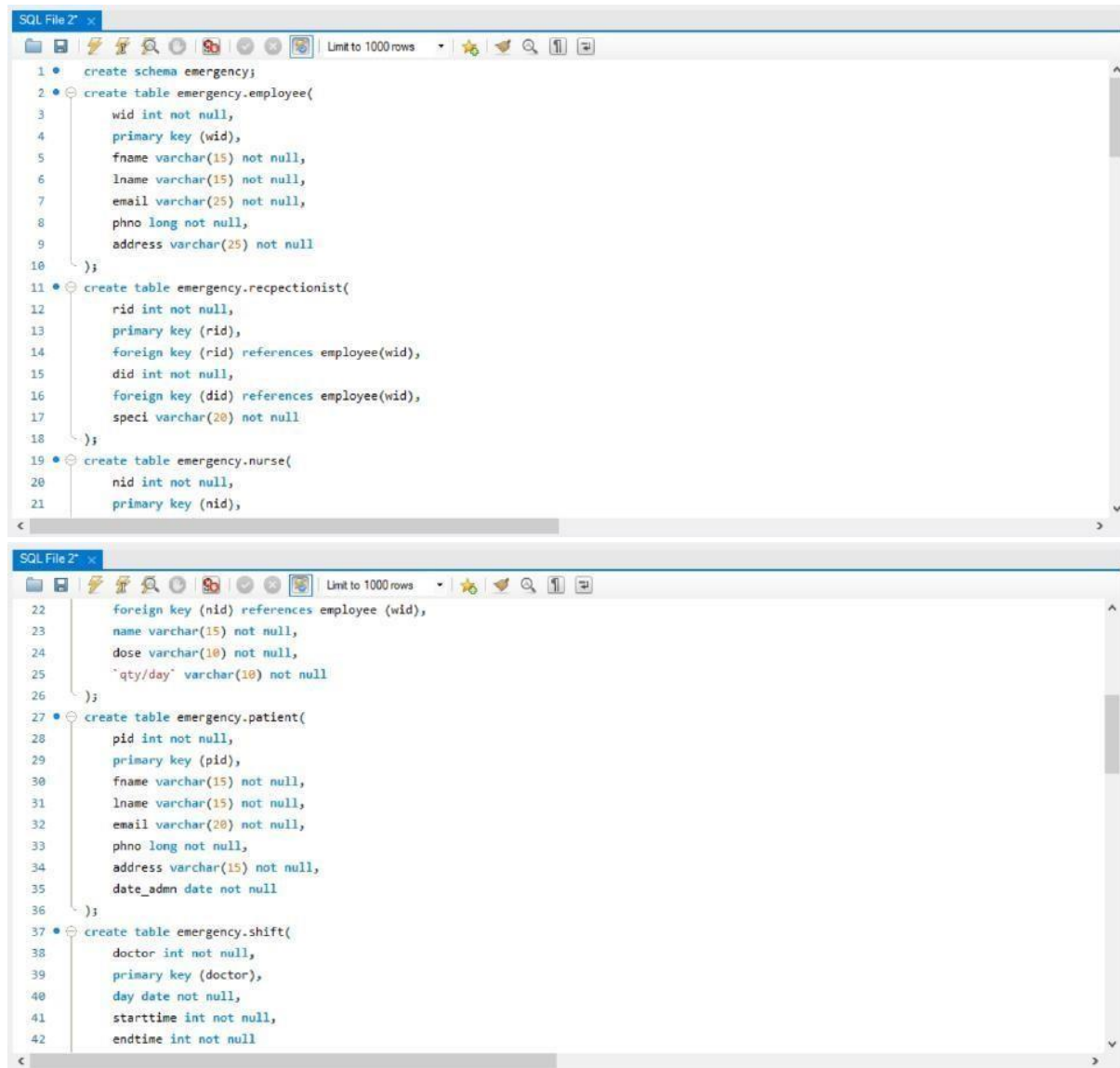
4.

X	Y
1	1
2	3
3	7
4	15
5	31
6	63
7	127

Answer is 127

**Case Study 2 :EMERGENCY ROOM INFORMATION SYSTEM**

1) Create the database in MySQL and create the necessary tables for the given case study using appropriate keys and relationships between the tables



```
1 • create schema emergency;
2 • create table emergency.employee(
3     wid int not null,
4     primary key (wid),
5     fname varchar(15) not null,
6     lname varchar(15) not null,
7     email varchar(25) not null,
8     phno long not null,
9     address varchar(25) not null
10 );
11 • create table emergency.receptionist(
12     rid int not null,
13     primary key (rid),
14     foreign key (rid) references employee(wid),
15     did int not null,
16     foreign key (did) references employee(wid),
17     speci varchar(20) not null
18 );
19 • create table emergency.nurse(
20     nid int not null,
21     primary key (nid),
22     foreign key (nid) references employee (wid),
23     name varchar(15) not null,
24     dose varchar(10) not null,
25     `qty/day` varchar(10) not null
26 );
27 • create table emergency.patient(
28     pid int not null,
29     primary key (pid),
30     fname varchar(15) not null,
31     lname varchar(15) not null,
32     email varchar(20) not null,
33     phno long not null,
34     address varchar(15) not null,
35     date_admn date not null
36 );
37 • create table emergency.shift(
38     doctor int not null,
39     primary key (doctor),
40     day date not null,
41     starttime int not null,
42     endtime int not null
```



```

SQL File 2' x
Limit to 1000 rows
43 }
44 • create TABLE emergency.pcase(
45     bednum int not null,
46     primary key (bednum),
47     pid int not null,
48     foreign key (pid) references patient(pid)
49 );
50 • create table emergency.appointment(
51     pid int not null,
52     foreign key (pid) references patient(pid),
53     amt int not null
54 );
55 • insert into emergency.employee
56 values(100,'Arun','Kumar','ebcd@gmail.com',9988776655,'Hyderabad'),
57        (101,'David','Raju','ghij@gmail.com',9987634567,'Secunderabad'),
58        (102,'Harish','Reddy','klmn@gmail.com',9986492479,'Bowenpally'),
59        (103,'John','Samuel','pqrt@gmail.com',9985350391,'Hyderabad'),
60        (104,'Kira','Kumar','ghjk@gmail.com',9984208303,'Amaravathi'),
61        (105,'Aravind','Babu','ghhjk@gmail.com',9983066215,'Guntur'),
62        (106,'Praveen','Kumar','qwrt@gmail.com',9981924127,'Tullur'),
63        (107,'Ramesh','Kumar','poiugmail.com',9980782039,'Mangalagiri'),

```

2) Insert atleast 10 records into every table that is implemented in the case study

```

SQL File 2' x
Limit to 1000 rows
64 (108,'Jayanth','Kumar','poutu@gmail.com',9979639951,'Vijayawada'),
65 (109,'Eswar','Raju','vghj@gmail.com',9978497863,'Hyderabad');
66 • insert into emergency.receptionist
67 values(104,100,'Anesthician'),
68        (106,101,'Dentist'),
69        (108,102,'Cardiologist'),
70        (109,105,'Neurosurgeon');
71 • insert into emergency.nurse
72 values(103,'bacd','once','2ml'),
73        (107,'xyz','twice','4ml'),
74        (109,'pqr','thrice','100mg');
75 • insert into emergency.patient
76 values(200,'David','Samson','aaa@gmail.com',7654328976,'Mumbai','2019-10-13'),
77        (201,'Bharath','Kumar','bbb@gmail.com',8765438907,'Hyderabad','2020-01-22'),
78        (202,'Eswar','Prasad','ccc@gmail.com',9876548036,'Vijayawada','2020-03-15'),
79        (203,'Jaya','Laxmi','ddd@gmail.com',8765489765,'Delhi','2020-07-06'),
80        (204,'Laxmi','Devi','eee@gmail.com',7654430692,'Mumbai','2020-03-12'),
81        (205,'Pranod','Reddy','fff@gmail.com',6543371619,'Hyderabad','2020-02-05'),
82        (206,'Charan','Kumar','ggg@gmail.com',5432312546,'Vijayawada','2020-04-22'),
83        (207,'Kalyan','Reddy','hhh@gmail.com',4321253473,'Delhi','2020-07-10'),
84        (208,'Rajesh','Yadav','iklm@gmail.com',3210194400,'Chennai','2020-03-03'),

```

```

SQL File 2' x
Limit to 1000 rows
85 (209,'Naveen','Kumar','ighfr@gmail.com',2099135327,'Hyderabad','2020-04-22');
86 • insert into emergency.pcase
87 values(20,200),
88        (21,201),
89        (22,202),
90        (23,203),
91        (24,204),
92        (25,205),
93        (26,206),
94        (27,207),
95        (28,208),
96        (29,209);
97 • insert into emergency.appointment
98 values(200,200),
99        (201,200),
100       (202,300),
101       (203,600),
102       (204,200),
103       (205,300),
104       (206,200),
105       (207,200),

```

```

92      (25,205),
93      (26,206),
94      (27,207),
95      (28,208),
96      (29,209);
97 • insert into emergency.appointment
98   values(200,200),
99      (201,200),
100     (202,300),
101     (203,600),
102     (204,200),
103     (205,300),
104     (206,200),
105     (207,200),
106     (208,200),
107     (209,600);
108 • insert into emergency.shift
109   values(100,'2020-04-12',9,5),
110     (101,'2020-06-20',4,8),
111     (102,'2020-08-01',6,12),
112     (105,'2020-08-05',13,18);

```

3) Write an MYSQL query to add columns actiontaken and personell to the table case?

```

1 • alter table emergency.pcase
2   add column actiontaken varchar(15) not null,
3   add column personell varchar(25) not null;
4 • SELECT * FROM emergency.pcase;

```

Result Grid:

bednum	pid	actiontaken	personell
20	200		
21	201		
22	202		
23	203		
24	204		
25	205		
26	206		
27	207		
28	208		
29	209		

4) Write an MYSQL query to display who are patients required to stay in the hospital?

```

1 • use emergency;
2 • select distinct p.*,bednum from patient p,pcase c where p.pid=c.pid;

```

Result Grid:

pid	fname	lname	email	phno	address	date_admn	bednum
200	David	Samson	aaa@gmail.com	7654328976	Mumbai	2019-10-13	20
201	Bharath	Kumar	bbb@gmail.com	8765438907	Hyderabad	2020-01-22	21
202	Enwar	Prasad	ccc@gmail.com	9876548838	Vijayawada	2020-03-15	22
203	Jaya	Laxmi	ddd@gmail.com	8765489765	Delhi	2020-07-06	23
204	Laxmi	Devi	eee@gmail.com	7654430692	Mumbai	2020-03-12	24
205	Pramod	Reddy	fff@gmail.com	6543371619	Hyderabad	2020-02-05	25
206	Charan	Kumar	ggg@gmail.com	5432312546	Vijayawada	2020-04-22	26
207	Kalyan	Reddy	hhh@gmail.com	4321253473	Delhi	2020-07-10	27
208	Rajesh	Yadav	iknn@gmail.com	3210194400	Chennai	2020-03-03	28
209	Naveen	Kumar	ighfr@gmail.com	2099135327	Hyderabad	2020-04-22	29

- 5) Write an MYSQL query to display who paid amount between 100 and 300 for the appointment

SQL File 6\* SQL File 7 appointment - Table appointment x

```
1 • SELECT p.*,a.amt FROM emergency.appointment a inner join emergency.patient p on a.pid=p.pid where amt between 100 and 300;
```

Result Grid

	pid	fname	lname	email	phno	address	date_admn	amt
200	David	Samson	aaa@gmail.com	7654328976	Mumbai	2019-10-13	200	
201	Bharath	Kumar	bbb@gmail.com	8765438907	Hyderabad	2020-01-22	200	
202	Eswar	Prasad	ccc@gmail.com	9876548838	Vijayawada	2020-03-15	300	
204	Laxmi	Devi	eee@gmail.com	7654430692	Mumbai	2020-03-12	200	
205	Pramod	Reddy	fff@gmail.com	6543371619	Hyderabad	2020-02-05	300	
206	Charan	Kumar	ggg@gmail.com	5432312546	Vijayawada	2020-04-22	200	
207	Kalyan	Reddy	hhh@gmail.com	4321253473	Delhi	2020-07-10	200	
208	Rajesh	Yadav	iklm@gmail.com	3210194400	Chennai	2020-03-03	200	

Result 5 x Read Only

- 6) Write an MYSQLquery to update the value of phno of WID=102 of worker table

SQL File 6\* SQL File 7 appointment - Table appointment employee employee x

```
1 • update emergency.employee set phno = 8885384444 where wid = 102;
```

```
2 • SELECT * FROM emergency.employee;
```

Result Grid

	wid	fname	lname	email	phno	address
100	Arun	Kumar	abcd@gmail.com	9988776655	Hyderabad	
101	David	Raju	ghij@gmail.com	9987634567	Secunderabad	
102	Harish	Reddy	iklm@gmail.com	8885384444	Bowenpally	
103	John	Samuel	pqrt@gmail.com	9985350391	Hyderabad	
104	Kira	Kumar	ghjk@gmail.com	9984208303	Amaravathi	
105	Aravind	Babu	ghjk@gmail.com	9983066215	Guntur	
106	Praveen	Kumar	qwert@gmail.com	9981924127	Tulur	
107	Ramesh	Kumar	poku@gmail.com	9980782039	Mangalagiri	
108	Jayanth	Kumar	poutu@gmail.com	9979639951	Vijayawada	
109	Eswar	Raju	vgjh@gmail.com	9978497863	Hyderabad	

employee 1 x Apply Revert

- 7) Write an MYSQL query to truncate the values of appointment table

SQL File 6\* SQL File 7\* pcase appointment x

```
1 • truncate emergency.appointment;
```

```
2 • SELECT * FROM emergency.appointment;
```

Result Grid

	pid	amt
--	-----	-----

Result Grid Form Editor Field Types

8) Write an MYSQL query to calculate average amount of appointment.

The screenshot shows a SQL IDE window with a query editor and a result grid. The query editor contains the following SQL query:

```
1 • SELECT avg(amt) FROM emergency.appointment;
```

The result grid displays the following data:

avg(amt)
300.0000

The interface includes a toolbar with icons for file operations, a 'Limit to 1000 rows' dropdown, and a 'Filter Rows' input field. The right sidebar contains buttons for 'Result Grid', 'Form Editor', and 'Field Types'.

9) Write an MYSQL query to display patient number in descending order.

The screenshot shows a SQL IDE window with a query editor and a result grid. The query editor contains the following SQL query:

```
1 • SELECT pid FROM emergency.patient order by pid desc;
```

The result grid displays the following data:

pid
209
208
207
206
205
204
203
202
201
200
more

The interface includes a toolbar with icons for file operations, a 'Limit to 1000 rows' dropdown, and a 'Filter Rows' input field. The right sidebar contains buttons for 'Result Grid', 'Form Editor', and 'Field Types'.

10) Write an MYSQL query to display bed number of patient with least patient id.

The screenshot shows a SQL IDE window with a query editor and a result grid. The query editor contains the following SQL query:

```
1 • SELECT bednum,pid FROM emergency.pcase where pid=(Select min(pid) from emergency.pcase);
```

The result grid displays the following data:

bednum	pid
20	200
more	more

The interface includes a toolbar with icons for file operations, a 'Limit to 1000 rows' dropdown, and a 'Filter Rows' input field. The right sidebar contains buttons for 'Result Grid', 'Form Editor', and 'Field Types'.



11) Write an MYSQL query to display bed number of patient with least patient id.

SQL File 6\* patient patienttable

```

1 • use emergency;
2 • create view Patienttable as
3 • select pid,fname,phno
4 • from patient;
5 • SELECT * FROM emergency.patienttable;

```

Result Grid

pid	fname	phno
200	David	7654328976
201	Bharath	8765438907
202	Eswar	9876548838
203	Jaya	8765489765
204	Laxmi	7654430692
205	Pramod	6543371619
206	Charan	5432312546
207	Kalyan	4321253473
208	Rajesh	3210194400
209	Naveen	2099135327

patienttable 1 x Read Only

12) Write a MYSQL query to update view the Patientview where patientid>3 by using replace view statement

SQL File 6\* patient SQL File 7\* patienttable info

```

1 • Create OR Replace VIEW emergency.info AS Select pid,fname,phno from emergency.patient WHERE pid = 203;
2 • SELECT * FROM emergency.info;

```

Result Grid

pid	fname	phno
203	Jaya	8765489765

Form Editor

13) Display the patient date of joining into format “dd month yyyy” for example as “24 March 2020”

SQL File 6\* patient patient

```

1 • select pid,fname,lname,phno,address,date_format(date_admn,'%d %M, %Y') from emergency.patient LIMIT 0, 1000;
2

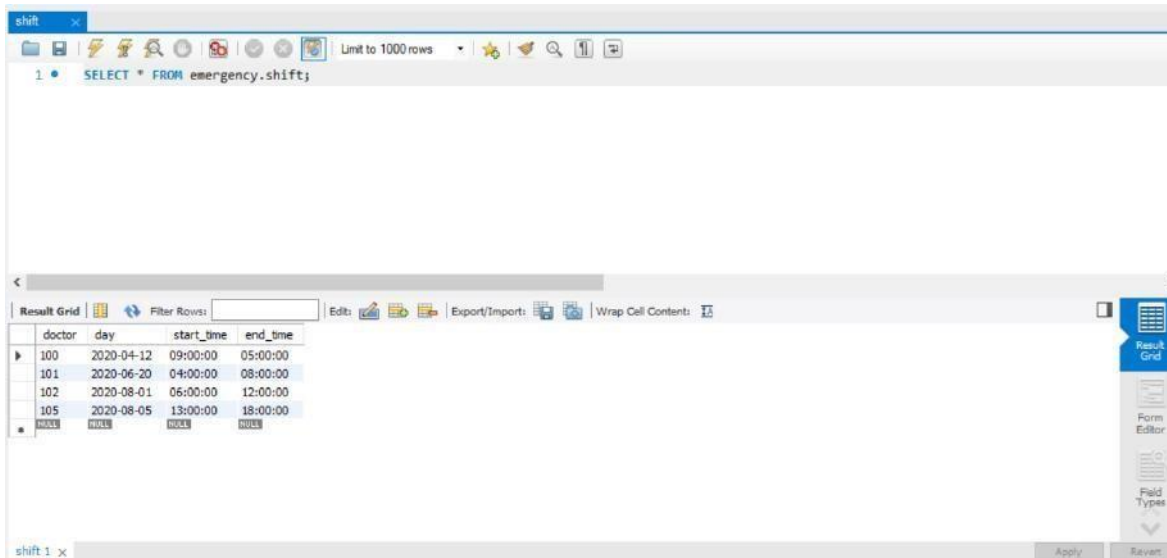
```

Result Grid

pid	fname	lname	phno	address	date_format(date_admn,'%d %M, %Y')
200	David	Samson	7654328976	Mumbai	13 October, 2019
201	Bharath	Kumar	8765438907	Hyderabad	22 January, 2020
202	Eswar	Prasad	9876548838	Vijayawada	15 March, 2020
203	Jaya	Laxmi	8765489765	Delhi	6 July, 2020
204	Laxmi	Devi	7654430692	Mumbai	12 March, 2020
205	Pramod	Reddy	6543371619	Hyderabad	5 February, 2020
206	Charan	Kumar	5432312546	Vijayawada	22 April, 2020
207	Kalyan	Reddy	4321253473	Delhi	10 July, 2020
208	Rajesh	Yadav	3210194400	Chennai	3 March, 2020
209	Naveen	Kumar	2099135327	Mumbai	22 April, 2020

Result 5 x Read Only

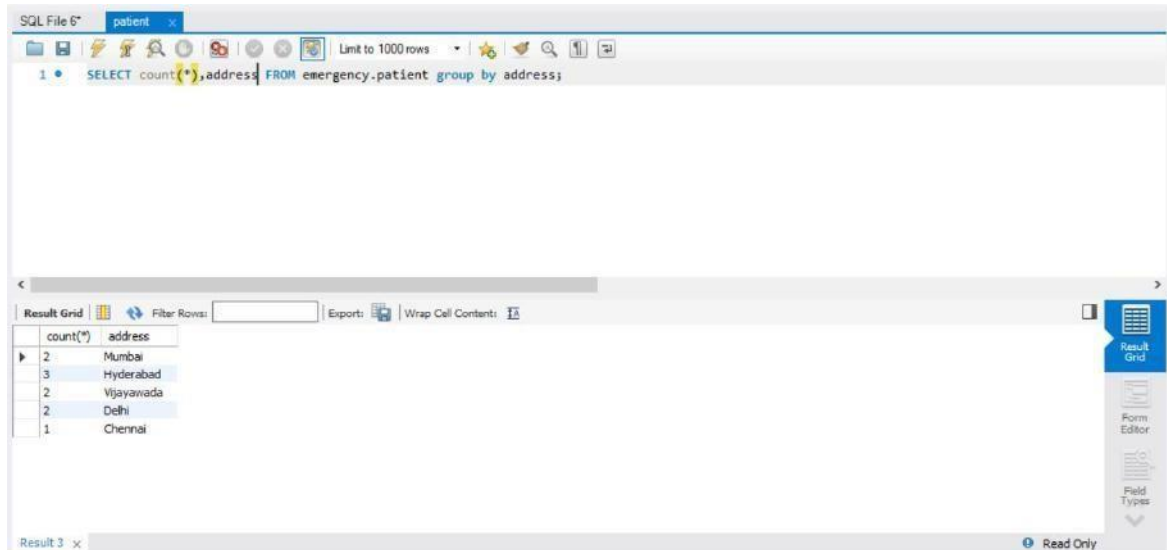
- 14) Display the patient joining time in 24 hour format. For eg: 3:30pm should be displayed as 15:30



The screenshot shows a database application window titled 'shift'. The query bar contains the SQL statement: `SELECT * FROM emergency.shift;`. The result grid displays the following data:

doctor	day	start_time	end_time
100	2020-04-12	09:00:00	05:00:00
101	2020-06-20	04:00:00	08:00:00
102	2020-08-01	06:00:00	12:00:00
105	2020-08-05	13:00:00	18:00:00
NULL	NULL	NULL	NULL

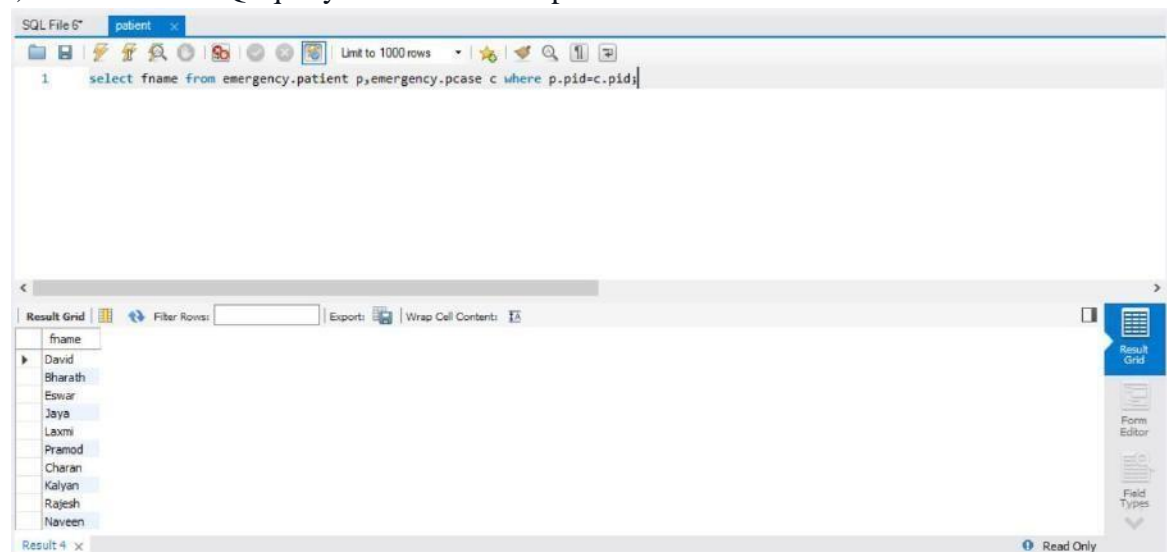
- 15) Write a MYSQL query to display count of patients who come from same city.



The screenshot shows a database application window titled 'patient'. The query bar contains the SQL statement: `SELECT count(*),address FROM emergency.patient group by address;`. The result grid displays the following data:

count(*)	address
2	Mumbai
3	Hyderabad
2	Vijayawada
2	Delhi
1	Chennai

- 16) Write an MYSQLquery to first name of patient who admitted and allotted the bed



The screenshot shows a database application window titled 'patient'. The query bar contains the SQL statement: `select fname from emergency.patient p,emergency.pcasa c where p.pid=c.pid;`. The result grid displays the following data:

fname
David
Bharath
Eswar
Jaya
Laxmi
Pramod
Charan
Kalyan
Rajesh
Navveen

## CASE STUDY 2: TOUR OPERATING SYSTEM

- 1) Create all the tables required with constraints and relationships between them

```

Query 1
1  create Schema Tour_Operating_System;
2  CREATE TABLE Tour_Operating_System.login (
3      log_id INT NOT NULL,
4      PRIMARY KEY (log_id),
5      log_name VARCHAR(20) NOT NULL,
6      log_pwd VARCHAR(15) NOT NULL
7  );
8  CREATE TABLE Tour_Operating_System.Person (
9      id INT NOT NULL,
10     PRIMARY KEY (id),
11     name VARCHAR(20) NOT NULL,
12     city VARCHAR(20) NOT NULL,
13     state VARCHAR(20) NOT NULL,
14     phone LONG NOT NULL,
15     dob DATE NOT NULL,
16     age INT NOT NULL
17 );
18 CREATE TABLE Tour_Operating_System.Customer (
19     cust_id INT NOT NULL,
20     PRIMARY KEY (cust_id),
21     c_name VARCHAR(20) NOT NULL,
22     c_addr VARCHAR(20) NOT NULL,
23     c_mobile LONG NOT NULL
24 );
25 CREATE TABLE Tour_Operating_System.Tour (
26     tr_id INT NOT NULL,
27     PRIMARY KEY (tr_id),
28     tr_start VARCHAR(20) NOT NULL,
29     tr_dest VARCHAR(20),
30     tr_date DATE NOT NULL,
31     tr_status VARCHAR(20) NOT NULL,
32     no_guides INT NOT NULL,
33     no_tourist INT NOT NULL
34 );
35 CREATE TABLE Tour_Operating_System.Hotel (
36     htl_id INT NOT NULL,
37     PRIMARY KEY (htl_id),
38     htl_name VARCHAR(20) NOT NULL,
39     htl_place VARCHAR(20) NOT NULL,
40     htl_rent INT NOT NULL,
41     htl_type INT NOT NULL
42 );

```

- 2) Insert atleast 10 records into the tables

```

Query 1
1  insert into tour_operating_system.login (log_id,log_name,log_pwd)
2  values(100,'Hello','abcd1234'),
3      (101,'welcome','pqrs3456'),
4      (102,'Good','xyzw1234'),
5      (103,'Honesty','qwerty5678');
6  insert into tour_operating_system.person(id,name,city,state,phone,dob,age)
7  values(200,'Kiran','Hyderabad','Telangana',9876543218,'2000-10-12',19),
8      (201,'hari','Vijayawada','Andhra Pradesh',7865409765,'1980-06-25',39),
9      (202,'Raju','Hyderabad','Telangana',8976543352,'1970-01-01',49),
10     (203,'Eswar','Guntur','Andhra Pradesh',8765467800,'1985-04-15',35),
11     (204,'David','Vijayawada','Andhra Pradesh',8554392248,'2000-07-28',20),
12     (205,'Devi','Mangalagiri','Andhra Pradesh',8343316696,'1975-11-10',45),
13     (206,'Rani','Hyderabad','Telangana',8132241144,'1998-02-22',22),
14     (207,'Jaya','Vijayawada','Andhra Pradesh',7921165592,'2001-06-06',19),
15     (208,'Kalyan','Guntur','Andhra Pradesh',7710090040,'1990-09-18',30),
16     (209,'Gopal','Hyderabad','Telangana',7499014488,'1992-12-31',28);
17 insert into tour_operating_system.customer(cust_id,c_name,c_addr,c_mobile)
18 values(300,'Raju','Hyd',8765467800),
19     (301,'Hari','Delhi',8554392248),
20     (302,'Kiran','Mumbai',8343316696),
21     (303,'Giri','Kolkata',8139241144),

```

```

Query 1
Limit to 1000 rows
22 (304,'Devil','Goa',7935165592),
23 (305,'Jaya','Chennai',7731090040),
24 (306,'Kalyan','Mysore',7527014480),
25 (307,'Kiran','Guntur',7322938936);
26 • insert into tour_operating_system.tour(tr_id, tr_start, tr_dest, tr_date, tr_status, no_guides, no_tourist)
27 values(400,'vij','hyd','2020-09-30','Successful',5,25),
28 (401,'hyd','vij','2020-10-01','Cancelled',3,18),
29 (402,'gnt','tnl','2020-10-02','Successful',4,24),
30 (403,'tnl','gnt','2020-10-03','Cancelled',5,26),
31 (404,'vij','Mumbai','2020-10-04','Successful',6,28);
32 • insert into tour_operating_system.hotel(htl_id, htl_name, htl_place, htl_rent, htl_type)
33 values(501,'Taj','Vijayawada',5000,4),
34 (502,'DV Manor','Vijayawada',4000,3),
35 (503,'Minerva','Guntur',3000,3),
36 (504,'Falaknuma palace','Hyderabad',8000,6),
37 (505,'ITC','Mumbai',9500,7);
38 • insert into tour_operating_system.booking(bk_id, bk_date, bk_title, bk_type)
39 values(601,'2020-09-20','Vij-Hyd','Online'),
40 (602,'2020-09-21','Hyd-Vij','Offline'),
41 (603,'2020-09-22','Vij-Mumbai','Online'),
42 (604,'2020-09-23','Gnt-tnl','Offline');

```

3) Write an SQL Query to Update the value of bk\_id=1001 in Booking table.

```

SQL File 3
Limit to 1000 rows
1 • UPDATE tour_operating_system.booking
2 SET
3     bk_id = 1001
4 WHERE
5     bk_date = '2020-09-21';
6
7 • Select * from tour_operating_system.booking;

```

bk_id	bk_date	bk_title	bk_type
601	2020-09-20	Vij-Hyd	Online
603	2020-09-22	Vij-Mumbai	Online
604	2020-09-23	Gnt-tnl	Offline
1001	2020-09-21	Hyd-Vij	Offline

4) Write an SQL Query to Truncate the values of Login table.

```

SQL File 3  SQL File 4  login  login
Limit to 1000 rows
1 • truncate tour_operating_system.login;
2 • SELECT * FROM tour_operating_system.login;

```

log_id	log_name	log_pwd

5) Which hotel rent is the highest amount for accommodations?

SQL File 3\*

```

1 • SELECT
2 • *
3 • FROM
4 •     tour_operating_system.hotel
5 • WHERE
6 •     htl_rent = (SELECT
7 •         MAX(htl_rent)
8 •         FROM
9 •             tour_operating_system.hotel);

```

Result Grid

htl_id	htl_name	htl_place	htl_rent	htl_type
505	ITC	Mumbai	9500	7

6) SQL Query to print the details of Booking information who have booked 5 star rated rooms.

SQL File 3\*

```

1 • SELECT
2 • *
3 • FROM
4 •     tour_operating_system.booking
5 • WHERE
6 •     bk_type = 5;

```

Result Grid

bk_id	bk_date	bk_title	bk_type
601	2020-09-20	Vij-Hyd	5
604	2020-09-23	Gnt-trl	5

7) Insert a column named aadhar\_id of the customer.

SQL File 3\*

```

1 • Alter table tour_operating_system.customer add aadhar_id varchar(12) not null;
2 •
3 • select * from tour_operating_system.customer;

```

Result Grid

cust_id	c_name	c_addr	c_mobile	aadhar_id
300	Raju	Hyd	8765467800	
301	Hari	Delhi	8554392248	
302	Kiran	Mumbai	8343316696	
303	Giri	Kolkata	8139241144	
304	Devi	Goa	7935165592	
305	Jaya	Chennai	7731090040	
306	Kalyan	Mysore	7527014488	
307	Kiran	Guntur	7322938936	

8) Display the persons whose TA\_id and role\_id is matched.

Query 1

```

1 • SELECT distinct * FROM tour_operating_system.travel_agent a inner join tour_operating_system.person p where a.TA_id=p.id;

```

Result Grid

TA_id	TA_name	id	name	city	state	phone	dob	age
-------	---------	----	------	------	-------	-------	-----	-----



9) How many participants are travelling for a particular tour?

SQL File 3

```

1 • SELECT
2   tr_id, no_tourist
3 FROM
4   tour_operating_system.tour;

```

Result Grid

tr_id	no_tourist
400	25
401	18
402	24
403	26
404	28
405	20

10) Search the names of the hotel in ottawa.

SQL File 3 SQL File 4 hotel

```

1 • SELECT * FROM tour_operating_system.hotel where htl_place = 'Ottawa';

```

Result Grid

htl_id	htl_name	htl_place	htl_rent	htl_type
506	Novotel	Ottawa	8700	5

11) Extract the name the customer whose person id and customer id is equal using join.

SQL File 3 SQL File 4 customer person customer

```

1 • SELECT * FROM tour_operating_system.customer c inner join tour_operating_system.person p where c.cust_id=p.id;
2 # No id is same in customer table with person table

```

Result Grid

cust_id	c_name	c_addr	c_mobile	id	name	city	state	phone	dob	age
---------	--------	--------	----------	----	------	------	-------	-------	-----	-----

12) Display the name of the participants whose age is between 35 and 55;

The screenshot shows a SQL query editor with the following query:

```
1 • SELECT
2   name
3 FROM
4   tour_operating_system.person
5 WHERE
6   age >= 35 AND age <= 55;
```

The result grid below the query shows the following data:

name
hari
Raju
Eswar
Devi

13) What is the origin of the travelling for tour id = 403

The screenshot shows a SQL query editor with the following query:

```
1 • SELECT tr_start FROM tour_operating_system.tour where tr_id = 403;
2   #tnl - Tamilnadu
```

The result grid below the query shows the following data:

tr_start
tnl

14) Select a person whose name is available in Customer and TravelAgent using nested query.

The screenshot shows a SQL query editor with the following query:

```
1 • SELECT c_name FROM tour_operating_system.customer where c_name in(select TA_name from tour_operating_system.travel_agent);
```

The result grid below the query shows the following data:

c_name
Raju
Giri

15) Select the past tour details.

SQL File 3\* SQL File 4\* SQL File 4\* tour

Limit to 1000 rows

1 • SELECT \* FROM tour\_operating\_system.tour where tr\_status = 'Completed';

Result Grid

tr_id	tr_start	tr_dest	tr_date	tr_status	no_guides	no_tourist
400	vij	hyd	2020-09-30	Completed	5	25
403	trl	gnt	2020-10-03	Completed	5	26

tour 3 x

Apply Revert

16) Select the future tour details.

SQL File 3\* SQL File 4\* SQL File 4\* tour

Limit to 1000 rows

1 • SELECT \* FROM tour\_operating\_system.tour where tr\_status = 'Future';

Result Grid

tr_id	tr_start	tr_dest	tr_date	tr_status	no_guides	no_tourist
402	gnt	trl	2020-10-02	Future	4	24
404	vij	Mumbai	2020-10-04	Future	6	28

tour 2 x

Apply Revert

## POST-LAB

- 1) Julia asked her students to create some coding challenges. Write a query to print the *hacker\_id*, *name*, and the total number of challenges created by each student. Sort your results by the total number of challenges in descending order. If more than one student created the same number of challenges, then sort the result by *hacker\_id*. If more than one student created the same number of challenges and the count is less than the maximum number of challenges created, then exclude those students from the result.

### Input Format

The following tables contain challenge data:

- *Hackers*: The *hacker\_id* is the id of the hacker, and *name* is the name of the hacker.
- *Challenges*: The *challenge\_id* is the id of the challenge, and *hacker\_id* is the id of the student who created the challenge

Column	Type
<i>hacker_id</i>	Integer
<i>name</i>	String

Column	Type
<i>challenge_id</i>	Integer
<i>hacker_id</i>	Integer

### Explanation

For *Sample Case 1*, we can get the following details:

<i>hacker_id</i>	<i>name</i>	<i>challenges_created</i>
12299	Rose	6
34856	Angela	6
79345	Frank	4
80491	Patrick	3
81041	Lisa	1

Students 12299 and 34856 both created 6 challenges. Because 6 is the maximum number of challenges created, these students are included in the result.

### ANS.

```
SELECT c.hacker_id, h.name, COUNT(c.challenge_id) AS cnt FROM Hackers AS h JOIN
Challenges AS c ON h.hacker_id = c.hacker_id GROUP BY c.hacker_id, h.name HAVING
cnt = (SELECT COUNT(c1.challenge_id) FROM Challenges AS c1 GROUP BY c1.hacker_id
ORDER BY COUNT(*) DESC LIMIT 1) OR cnt NOT IN (SELECT COUNT(c2.challenge_id) FROM
Challenges AS c2 GROUP BY c2.hacker_id HAVING c2.hacker_id <> c.hacker_id) ORDER BY
cnt DESC, c.hacker_id;
```

2) Query all columns for all American cities in the **CITY** table with populations larger than 100000. The **CountryCode** for America is USA. The **CITY** table is described as follows:

CITY	
Field	Type
ID	NUMBER
NAME	VARCHAR2 ( 17 )
COUNTRYCODE	VARCHAR2 ( 3 )
DISTRICT	VARCHAR2 ( 20 )
POPULATION	NUMBER

ANS.

```
SELECT * FROM CITY WHERE COUNTRYCODE = 'USA' AND POPULATION > 100000;
```