SQL ASSIGNMENT

1. Given a table of employees, find the number of male and female employees in each department:

EmpID	Name	Gender	Department
1	X	Female	Finance
2	Y	Male	IT
3	Z	Male	HR
4	W	Female	IT

Output:

Department	Num of male	Num of Female
Finance	0	1
HR	1	0
IT	1	1

```
mysql> create database sqlassignment
    -> ;
Query OK, 1 row affected (0.01 sec)

mysql> use database sqlassignment
ERROR 1049 (42000): Unknown database 'database'
mysql> use database sqlassignment;
ERROR 1049 (42000): Unknown database 'database'
mysql> use sqlassignment;
Database changed
mysql> create table employees (
    -> empid integer(4) not null unique,
    -> emp_name varchar(30),
    -> Gender varchar(10),
    -> department varchar(30),
    -> check(Gender in ("Male","Female")));
Query OK, 0 rows affected, 1 warning (0.02 sec)

mysql> Insert into employees values(1,'X','Female','Finance'),(2,'Y','Male','IT'),(3,'Z','Male','HR'),(4,'W','Female','IT');
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from employees;
+-------+----------+--------+------------+
| empid | emp_name | Gender | department |
+-------+----------+--------+------------+
|     1 | X        | Female | Finance    |
|     2 | Y        | Male   | IT         |
|     3 | Z        | Male   | HR         |
|     4 | W        | Female | IT         |
+-------+----------+--------+------------+
4 rows in set (0.00 sec)

mysql> Insert into employees values(5,'V','mal','IT');
ERROR 3819 (HY000): Check constraint 'employees_chk_1' is violated.
mysql> insert into employees values(6,'P','Male',null);
Query OK, 1 row affected (0.01 sec)

mysql> SELECT IFNULL(Department,'Not Assigned') as Department,
    ->     COUNT(CASE WHEN UPPER(Gender)='MALE' THEN 1 END) AS 'Num of Male',
    ->     COUNT(CASE WHEN UPPER(Gender)='FEMALE' THEN 1 END) AS 'Num of Female'
    -> FROM employees GROUP BY Department;
+--------------+-------------+---------------+
| Department   | Num of Male | Num of Female |
+--------------+-------------+---------------+
| Finance      |           0 |             1 |
| IT           |           1 |             1 |
| HR           |           1 |             0 |
| Not Assigned |           1 |             0 |
+--------------+-------------+---------------+
4 rows in set, 1 warning (0.00 sec)

mysql>
```

Commands used:

//to create a database and use it

//to create table employees like above

```
CREATE database `Sqlassignment` ;
USE `Sqlassignment`;
create table employees (
empid integer(4) not null unique,
emp_name varchar(30),
Gender varchar(10),
department varchar(30),
check(Gender in ("Male","Female")));
```

//to insert values into table

```
Insert into employees
values(1,'X','Female','Finance'),(2,'Y','Male','IT'),(3,'Z','
Male','HR'),(4,'W','Female','IT');
```

//trying to insert null values

```
Insert into employees values(5,'X','Female',null);
```

```
No error.
```

//Query to find the number of employees per department at the same time trying to handle the above case where department is null by displaying "not assigned"

```
SELECT IFNULL(Department,'Not Assigned') as Department,
    COUNT(CASE WHEN UPPER(Gender)='MALE' THEN 1 END) AS 'Num
of Male',
    COUNT(CASE WHEN UPPER(Gender)='FEMALE' THEN 1 END) AS
'Num of Female'
FROM employees GROUP BY Department;
```

2. Given a table with salaries of employees for different month, find the max amount from the rows with month name:

Name	Jan	Feb	Mar
X	5200	9093	3832
Y	9023	8942	4000
Z	9834	8197	9903
W	3244	4321	0293

Output:

Name	Value	Month
X	9093	Feb
Y	9023	Jan
Z	9903	Mar
W	4321	Feb

```
mysql> create table employeesalaries (
    -> emp_name varchar(30) not null,
    -> Jan Float(15,2) Not null default 0,
    -> Feb Float(15,2) Not null default 0,
    -> March Float(15,2) Not null default 0,
    -> check(Jan>=0 and Feb >=0 and March >=0));
Query OK, 0 rows affected, 3 warnings (0.01 sec)

mysql> select * from employeesalaries;
Empty set (0.00 sec)

mysql> desc employeesalaries;
+----------+------------+------+-----+---------+-------+
| Field    | Type       | Null | Key | Default | Extra |
+----------+------------+------+-----+---------+-------+
| emp_name | varchar(30) | NO   |     | NULL    |       |
| Jan      | float(15,2) | NO   |     | 0.00    |       |
| Feb      | float(15,2) | NO   |     | 0.00    |       |
| March    | float(15,2) | NO   |     | 0.00    |       |
+----------+------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> Insert into employeesalaries values('X',5200,9093,3832),('Y',9023,8942,4000),('Z',9834,8197,9903),('W',3244,4321,0293);
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from employeesalaries;
+----------+---------+---------+---------+
| emp_name | Jan     | Feb     | March   |
+----------+---------+---------+---------+
| X        | 5200.00 | 9093.00 | 3832.00 |
| Y        | 9023.00 | 8942.00 | 4000.00 |
| Z        | 9834.00 | 8197.00 | 9903.00 |
| W        | 3244.00 | 4321.00 |  293.00 |
+----------+---------+---------+---------+
4 rows in set (0.00 sec)

mysql> Insert into employeesalaries values('Y',100,100,null);
ERROR 1048 (23000): Column 'March' cannot be null
mysql> Insert into employeesalaries values('Y',100,-100,null);
ERROR 1048 (23000): Column 'March' cannot be null
mysql> select emp_name as Name,value,case when idx=1 then 'Jan' when idx=2 then 'Feb' ,when idx=3 then 'Mar' end as Month from (select emp_name,greatest(Jan, Feb, March) as value,field(greatest(Jan, Feb,
March), Jan, Feb, March) as idx from
    ->   employeesalaries
    -> ) emps;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ',when idx=3 then 'Mar' end as Month from (select
emp_name,greatest(Jan, Feb, Mar' at line 1
mysql> select emp_name as Name,value,case when idx=1 then 'Jan' when idx=2 then 'Feb' ,when idx=3 then 'Mar' end as Month from (select emp_name,greatest(Jan, Feb, March) as value,field(greatest(Jan, Feb,
March), Jan, Feb, March) as idx from   employeesalaries ) emps;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ',when idx=3 then 'Mar' end as Month from (select
emp_name,greatest(Jan, Feb, Mar' at line 1
mysql> select emp_name as Name,value,case when idx=1 then 'Jan' when idx=2 then 'Feb' when idx=3 then 'Mar' end as Month from (select emp_name,greatest(Jan, Feb, March) as value,field(greatest(Jan, Feb,
arch), Jan, Feb, March) as idx from   employeesalaries ) emps;
+------+---------+-------+
| Name | value   | Month |
+------+---------+-------+
| X    | 9093.00 | Feb   |
| Y    | 9023.00 | Jan   |
| Z    | 9903.00 | Mar   |
| W    | 4321.00 | Feb   |
+------+---------+-------+
4 rows in set (0.00 sec)
```

Commands used:

//to create table employeesalaries like above

```
create table employeesalaries (
    -> emp_name varchar(30) not null,
    -> Jan Float(15,2) Not null default 0,
    -> Feb Float(15,2) Not null default 0,
    -> March Float(15,2) Not null default 0,
    -> check(Jan>=0 and Feb >=0 and March >=0));
```

//to insert values into table

```
Insert into employeesalaries values('V',100,100,null);

//March column is null hence error is thrown
```

//Query to get salaries of employees of different months , find the max amount from the rows with month name

```
select emp_name as Name,value,case when idx=1
then 'Jan' when idx=2 then 'Feb' when idx=3 then
'Mar' end as Month from (select
emp_name,greatest(Jan, Feb, March) as
value,field(greatest(Jan, Feb, March), Jan, Feb,
March) as idx from   employeesalaries) emps;
```

Name	value	Month
X	9093.00	Feb
Y	9023.00	Jan
Z	9903.00	Mar
W	4321.00	Feb

3. Given the marks obtained by candidates in a test, rank them in proper order.

Candidate_ID	Marks
1	98
2	78
3	87
4	98
5	78

Output:

Marks	Rank	Candidate_ID
98	1	1,4
87	2	3
78	3	2,5

```
mysql> create table test (
    -> candidate_id integer(4) not null unique,
    -> marks float(10,2) default 0);
Query OK, 0 rows affected, 2 warnings (0.01 sec)

mysql> desc test
    -> ;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| candidate_id | int         | NO   | PRI | NULL    |       |
| marks        | float(10,2) | YES  |     | 0.00    |       |
+--------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> Insert into test values (1,98),(2,78),(3,87),(4,98),(5,78);
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> SELECT Marks,GROUP_CONCAT(Candidate_id) as Candidate_id, dense_rank() OVER (order by marks desc ) as 'rank'
    ->    FROM test
    -> GROUP BY marks;
+-------+--------------+------+
| Marks | Candidate_id | rank |
+-------+--------------+------+
| 98.00 | 1,4          |    1 |
| 87.00 | 3            |    2 |
| 78.00 | 2,5          |    3 |
+-------+--------------+------+
3 rows in set (0.00 sec)

mysql>
```

//to create table test like above

```
create table test (
    -> candidate_id integer(4) not null unique,
    -> marks float(10,2) default 0);
```

//insert the above values

```
Insert into test values (1,98),(2,78),(3,87),(4,98),(5,78);
```

//Query to rank students based on the marks and showing student id.

```
 SELECT Marks,GROUP_CONCAT(Candidate_id) as Candidate_id,
dense_rank() OVER (order by marks desc ) as 'rank'
    ->    FROM test
    -> GROUP BY marks;
```

4. If same value is repeated for different id, then keep the value that has smallest id and delete all the other rows having same value:

Candidate_ID	Email
45	abc@gmail.com
23	def@yahoo.com
34	abc@gmail.com
21	bcf@gmail.com
94	def@yahoo.com

Output:

Candidate_ID	Email
34	abc@gmail.com
23	def@yahoo.com
21	bcf@gmail.com

```
mysql> create table mailids (
    -> candidate_id integer(4) not null,
    -> mail varchar(30) not null);
Query OK, 0 rows affected, 1 warning (0.02 sec)

mysql> Insert into mailids values
    -> (45,'abc@gmail.com'),
    -> (23,'def@yahoo.com'),
    -> (34,'abc@gmail.com'),
    -> (21,'bcf@gmail.com'),
    -> (94,'def@yahoo.com');
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> select * from mailids;
+--------------+---------------+
| candidate_id | mail          |
+--------------+---------------+
|           45 | abc@gmail.com |
|           23 | def@yahoo.com |
|           34 | abc@gmail.com |
|           21 | bcf@gmail.com |
|           94 | def@yahoo.com |
+--------------+---------------+
5 rows in set (0.00 sec)

mysql> DELETE FROM mailids WHERE candidate_id in (Select tempcandidate_id from (select Distinct a.candidate_id as tempcandidate_id from mailids a inner join mailids b where a.mail=b.mail and a.candidate_i
d>b.candidate_id) as c) order by candidate_id;
Query OK, 2 rows affected (0.01 sec)

mysql> select * from mailids;
+--------------+---------------+
| candidate_id | mail          |
+--------------+---------------+
|           23 | def@yahoo.com |
|           34 | abc@gmail.com |
|           21 | bcf@gmail.com |
+--------------+---------------+
3 rows in set (0.00 sec)
```

//to create table mailids like above

```
create table mailids (
    -> candidate_id integer(4) not null,
    -> mail varchar(30) not null);
```

//insert the above values

```
Insert into mailids values
    -> (45,'abc@gmail.com'),
    -> (23,'def@yahoo.com'),
    -> (34,'abc@gmail.com'),
    -> (21,'bcf@gmail.com'),
    -> (94,'def@yahoo.com');
```

//Query to delete records having the same id but keeping the ids that have the least id number

```
DELETE FROM mailids WHERE candidate_id in (Select
tempcandidate_id from (select Distinct a.candidate_id as
tempcandidate_id from mailids a inner join mailids b where
a.mail=b.mail and a.candidate_id>b.candidate_id) as c) order
by candidate_id desc;
```