**Name : Avilash Bhowmick**

**Week 06 Task :   
Leetcode URL :** <https://leetcode.com/u/AvilashBhowmick12/>  
  
**Resources :**

<https://www.bing.com/ck/a?!&&p=d15884ae11ca0125JmltdHM9MTcwNzA5MTIwMCZpZ3VpZD0zZjliZWFhZC1iNDk1LTYwOWUtMGZiNy1mZWIxYjUyNzYxNjYmaW5zaWQ9NTIwNA&ptn=3&ver=2&hsh=3&fclid=3f9beaad-b495-609e-0fb7-feb1b5276166&psq=leetcode&u=a1aHR0cHM6Ly9sZWV0Y29kZS5jb20v&ntb=1>

Soltuion Queries :

**Select**

1. Recyclable and Low Fat Products

# Write your MySQL query statement below

SELECT product\_id

FROM Products

WHERE low\_fats = 'Y' AND recyclable = 'Y';

2. Find Customer Referee

# Write your MySQL query statement below

select name from customer where referee\_id != 2 OR referee\_id IS NULL;

3. Big Countries

# Write your MySQL query statement below

select name, population , area from World where area >= 3000000 or population >= 25000000;

4. Article Views I

# Write your MySQL query statement below

SELECT DISTINCT author\_id AS id

FROM Views

WHERE author\_id = viewer\_id

ORDER BY id ASC;

5. Invalid Tweets

# Write your MySQL query statement below

SELECT tweet\_id

FROM Tweets

WHERE LENGTH(content) > 15;

**Basic Joins**

6. Replace Employee ID With The Unique Identifier

# Write your MySQL query statement below

SELECT e.name, a.unique\_id

FROM Employees e

LEFT JOIN EmployeeUNI a ON e.id = a.id

ORDER BY e.id;

7. Product Sales Analysis I

# Write your MySQL query statement below

SELECT p.product\_name, s.year, s.price

FROM Sales s

LEFT JOIN Product p ON s.product\_id = p.product\_id

WHERE s.year IS NOT NULL AND s.price IS NOT NULL;

8. Customer Who Visited but Did Not Make Any Transactions

# Write your MySQL query statement below

SELECT V.customer\_id, COUNT(V.visit\_id) AS count\_no\_trans

FROM Visits V

LEFT JOIN Transactions T ON V.visit\_id = T.visit\_id

WHERE T.transaction\_id IS NULL

GROUP BY V.customer\_id;

9. Rising Temperature

# Write your MySQL query statement below

SELECT w1.id

FROM Weather w1

JOIN Weather w2 ON w1.recordDate = DATE\_ADD(w2.recordDate, INTERVAL 1 DAY)

WHERE w1.temperature > w2.temperature;

10. Average Time of Process per Machine

# Write your MySQL query statement below

SELECT machine\_id,

       ROUND(AVG(timestamp\_diff), 3) AS processing\_time

FROM (

    SELECT machine\_id,

           process\_id,

           MAX(CASE WHEN activity\_type = 'end' THEN timestamp END) -

           MIN(CASE WHEN activity\_type = 'start' THEN timestamp END) AS timestamp\_diff

    FROM Activity

    GROUP BY machine\_id, process\_id

) AS process\_times

GROUP BY machine\_id;

11. Employee Bonus

# Write your MySQL query statement below

SELECT e.name, b.bonus

FROM Employee e

LEFT JOIN Bonus b ON e.empId = b.empId

WHERE b.bonus < 1000 OR b.bonus IS NULL;

12. Students and Examinations

# Write your MySQL query statement below

SELECT S.student\_id, S.student\_name, SUB.subject\_name,

       COALESCE(COUNT(E.subject\_name), 0) AS attended\_exams

FROM Students S

CROSS JOIN Subjects SUB

LEFT JOIN Examinations E ON S.student\_id = E.student\_id AND SUB.subject\_name = E.subject\_name

GROUP BY S.student\_id, SUB.subject\_name

ORDER BY S.student\_id, SUB.subject\_name;

13. Managers with at Least 5 Direct Reports

# Write your MySQL query statement below

SELECT Manager.name

FROM Employee

INNER JOIN Employee AS Manager ON (Employee.managerId = Manager.id)

GROUP BY Manager.id

HAVING COUNT(\*) >= 5;

14. Confirmation Rate

# Write your MySQL query statement below

WITH ConfirmationCounts AS (

    SELECT

        s.user\_id,

        SUM(CASE WHEN c.action = 'confirmed' THEN 1 ELSE 0 END) AS confirmed\_count,

        COUNT(\*) AS total\_count

    FROM Signups s

    LEFT JOIN Confirmations c ON s.user\_id = c.user\_id

    GROUP BY s.user\_id

)

SELECT

    c.user\_id,

    ROUND(IFNULL(confirmed\_count / NULLIF(total\_count, 0), 0), 2) AS confirmation\_rate

FROM ConfirmationCounts c;

**Basic Aggregate Functions**

15. Not Boring Movies

# Write your MySQL query statement below

SELECT id, movie, description, rating

FROM Cinema

WHERE id % 2 = 1 AND description != 'boring'

ORDER BY rating DESC;

16. Average Selling Price

# Write your MySQL query statement below

SELECT p.product\_id, IFNULL(ROUND(SUM(units\*price)/SUM(units),2),0) AS average\_price

FROM Prices p LEFT JOIN UnitsSold u

ON p.product\_id = u.product\_id AND

u.purchase\_date BETWEEN start\_date AND end\_date

group by product\_id

17. Project Employees I

# Write your MySQL query statement below

SELECT project\_id, ROUND(AVG(experience\_years), 2) AS average\_years

FROM Project AS p

LEFT JOIN Employee AS e ON p.employee\_id = e.employee\_id

GROUP BY project\_id;

18. Percentage of Users Attended a Contest

select

contest\_id,

round(count(distinct user\_id) \* 100 /(select count(user\_id) from Users) ,2) as percentage

from  Register

group by contest\_id

order by percentage desc,contest\_id

19. Queries Quality and Percentage

# Write your MySQL query statement below

select query\_name , round(avg (rating /position),2) as quality,

round(avg(case when rating < 3 then 1 else 0 end) \* 100 ,2)

as poor\_query\_percentage

from queries

where query\_name is not null

group by query\_name ;

20. Monthly Transactions I

# Write your MySQL query statement below

SELECT  SUBSTR(trans\_date,1,7) as month, country, count(id) as trans\_count, SUM(CASE WHEN state = 'approved' then 1 else 0 END) as approved\_count, SUM(amount) as trans\_total\_amount, SUM(CASE WHEN state = 'approved' then amount else 0 END) as approved\_total\_amount

FROM Transactions

GROUP BY month, country

21. Immediate Food Delivery II

# Write your MySQL query statement below

Select

    round(avg(order\_date = customer\_pref\_delivery\_date)\*100, 2) as immediate\_percentage

from Delivery

where (customer\_id, order\_date) in (

  Select customer\_id, min(order\_date)

  from Delivery

  group by customer\_id

);

22. Game Play Analysis IV

# Write your MySQL query statement below

SELECT

  ROUND(COUNT(DISTINCT player\_id) / (SELECT COUNT(DISTINCT player\_id) FROM Activity), 2) AS fraction

FROM

  Activity

WHERE

  (player\_id, DATE\_SUB(event\_date, INTERVAL 1 DAY))

  IN (

    SELECT player\_id, MIN(event\_date) AS first\_login FROM Activity GROUP BY player\_id

  )

**Sorting and Grouping**

23. Number of Unique Subjects Taught by Each Teacher

# Write your MySQL query statement below

# Write your MySQL query statement below

SELECT teacher\_id, COUNT(DISTINCT subject\_id) AS cnt

FROM Teacher

GROUP BY teacher\_id;

24. User Activity for the Past 30 Days I

# Write your MySQL query statement below

select substr(activity\_date,1,10) as day, count(distinct user\_id) as active\_users

from activity

where   activity\_date between '2019-06-28' and'2019-07-27'

group by activity\_date;

25. Product Sales Analysis III

WITH CTE AS (

    SELECT product\_id, MIN(year) AS minyear FROM Sales

    GROUP BY product\_id

)

SELECT s.product\_id, s.year AS first\_year, s.quantity, s.price

FROM Sales s

INNER JOIN CTE ON cte.product\_id = s.product\_id  AND s.year = cte.minyear;

26. Classes More Than 5 Students

# Write your MySQL query statement below

SELECT class

FROM Courses

GROUP BY class

HAVING COUNT(class) >= 5;

27. Find Followers Count

# Write your MySQL query statement below

SELECT user\_id, COUNT(follower\_id) AS followers\_count

FROM Followers

GROUP BY user\_id

ORDER BY user\_id;

28. Biggest Single Number

# Write your MySQL query statement below

SELECT MAX(num) AS num

FROM (

    SELECT num

    FROM MyNumbers

    GROUP BY num

    HAVING COUNT(num) = 1

) AS single\_numbers;

29. Customers Who Bought All Products

# Write your MySQL query statement below

SELECT customer\_id

FROM Customer

GROUP BY customer\_id

HAVING COUNT(DISTINCT product\_key) = (SELECT COUNT(DISTINCT product\_key) FROM Product);

**Advanced Select and Joins**

30. The Number of Employees Which Report to Each Employee

# Write your MySQL query statement below

SELECT e.employee\_id,

       e.name,

       COUNT(DISTINCT r.employee\_id) AS reports\_count,

       ROUND(AVG(r.age), 0) AS average\_age

FROM Employees e

LEFT JOIN Employees r ON e.employee\_id = r.reports\_to

GROUP BY e.employee\_id, e.name

HAVING COUNT(DISTINCT r.employee\_id) > 0

ORDER BY e.employee\_id;

31. Primary Department for Each Employee

# Write your MySQL query statement below

SELECT employee\_id, department\_id

FROM Employee

GROUP BY employee\_id

HAVING COUNT(employee\_id) < 2

UNION

SELECT employee\_id, department\_id

FROM Employee

WHERE primary\_flag = 'Y';

32. Triangle Judgement

# Write your MySQL query statement below

SELECT x, y, z,

       CASE WHEN x + y > z AND x + z > y AND y + z > x THEN 'Yes' ELSE 'No' END AS triangle

FROM Triangle;

33. Consecutive Numbers

# Write your MySQL query statement below

select distinct l1.num as ConsecutiveNums from logs l1, logs l2, logs l3

where l1.id=l2.id+1 and l1.num=l2.num

and l1.id=l3.id+2 and l1.num= l3.num

34. Product Price at a Given Date

# Write your MySQL query statement below

select

    distinct product\_id,

    10 as price

from

    products

group by

    product\_id

having

    min(change\_date) > "2019-08-16"

union

select

    product\_id,

    new\_price

from

    Products

where

    (product\_id, change\_date) in(

                                select

                                    product\_id,

                                    max(change\_date) as recent\_date

                                from

                                    Products

                                where

                                    change\_date <= "2019-08-16"

                                group by

                                    product\_id

                                )

35. Last Person to Fit in the Bus

# Write your MySQL query statement below

with recursive cte as(

    select person\_name,turn,weight as total

    from queue

    where turn=1

    union all

    select q.person\_name,q.turn,q.weight+cte.total as total

    from queue q

    join cte on cte.turn+1=q.turn

    where cte.total+q.weight<=1000

)

select person\_name from cte order by total desc limit 1

36. Count Salary Categories

# Write your MySQL query statement below

SELECT

    'Low Salary' AS category,

    SUM(income < 20000) AS accounts\_count

FROM

    Accounts

UNION

    SELECT

        'Average Salary' AS category,

        SUM(income BETWEEN 20000 AND 50000 ) AS accounts\_count

    FROM

        Accounts

UNION

    SELECT

        'High Salary' AS category,

        SUM(income > 50000) AS accounts\_count

    FROM

        Accounts;

**Subqueries**

37. Employees Whose Manager Left the Company

# Write your MySQL query statement below

select employee\_id from Employees

where salary < 30000 and manager\_id not in (select employee\_id from Employees) order by employee\_id;

38. Exchange Seats

# Write your MySQL query statement below

select

    case

        when id = (select max(id) from seat) and id % 2 = 1

            then id

        when id % 2 = 1

            then id + 1

        when id % 2 = 0

            then id - 1

    end as id,

    student

from

    seat

order by

    id;

39. Movie Rating

# Write your MySQL query statement below

(select u.name as results

from Users u

join MovieRating m on u.user\_id = m.user\_id

group by u.user\_id

order by count(movie\_id) desc, name asc

limit 1)

union all

(select title as results

from Movies u

join MovieRating m on u.movie\_id = m.movie\_id

where year(created\_at) = '2020' and month(created\_at) = '02'

group by u.movie\_id

order by avg(rating) desc, title asc

limit 1)

40. Restaurant Growth

-- Write your MySQL query statement below

SELECT

    visited\_on,

    (

        SELECT SUM(amount)

        FROM customer

        WHERE visited\_on BETWEEN DATE\_SUB(c.visited\_on, INTERVAL 6 DAY) AND c.visited\_on

    ) AS amount,

    ROUND(

        (

            SELECT SUM(amount) / 7

            FROM customer

            WHERE visited\_on BETWEEN DATE\_SUB(c.visited\_on, INTERVAL 6 DAY) AND c.visited\_on

        ),

        2

    ) AS average\_amount

FROM customer c

WHERE visited\_on >= (

        SELECT DATE\_ADD(MIN(visited\_on), INTERVAL 6 DAY)

        FROM customer

    )

GROUP BY visited\_on;

41. Friend Requests II: Who Has the Most Friends

# Write your MySQL query statement below

with new as (

(select accepter\_id as id, count(\*) as num from RequestAccepted

group by 1)

union all

(select requester\_id as id, count(\*) as num from RequestAccepted

group by 1)

)

select id, sum(num) as num from new

group by id

order by 2 desc limit 1

42. Investments in 2016

# Write your MySQL query statement below

SELECT ROUND(SUM(tiv\_2016), 2) AS tiv\_2016

FROM Insurance

WHERE tiv\_2015 IN (

    SELECT tiv\_2015

    FROM Insurance

    GROUP BY tiv\_2015

    HAVING COUNT(\*) > 1

)

AND (lat, lon) IN (

    SELECT lat, lon

    FROM Insurance

    GROUP BY lat, lon

    HAVING COUNT(\*) = 1

)

43. Department Top Three Salaries

# Write your MySQL query statement below

SELECT

    d.name AS Department,

    e.name AS Employee,

    e.salary AS Salary

FROM

    Employee e

    JOIN Department d ON e.departmentId = d.id

WHERE

    (

        SELECT COUNT(DISTINCT salary)

        FROM Employee e2

        WHERE e2.departmentId = e.departmentId AND e2.salary >= e.salary

    ) <= 3

ORDER BY

    Department, Salary DESC;

**Advanced String Functions / Regex / Clause**

44. Fix Names in a Table

# Write your MySQL query statement below

SELECT Users.user\_id , CONCAT(UPPER(SUBSTR(Users.name,1,1)),LOWER(SUBSTR(Users.name,2))) AS name

FROM Users

ORDER BY

Users.user\_id ASC

45. Patients With a Condition

# Write your MySQL query statement below

select \* from patients

where conditionS like '% DIAB1%' OR conditionS like 'DIAB1%' ;

46. Delete Duplicate Emails

# Write your MySQL query statement below

delete p1 from person p1,person p2

where p1.email=p2.email and p1.id>p2.id;

47. Second Highest Salary

# Write your MySQL query statement below

 Select max(salary)  AS SecondHighestSalary from employee

 where salary<(select max(salary) from employee);

48. Group Sold Products By The Date

# Write your MySQL query statement below

select sell\_date, count( DISTINCT product ) as num\_sold ,

    GROUP\_CONCAT( DISTINCT product order by product ASC separator ',' ) as products

        FROM Activities GROUP BY sell\_date order by sell\_date ASC;

49. List the Products Ordered in a Period

# Write your MySQL query statement below

select product\_name, sum(unit) as unit from products

join orders using(product\_id)

where year(order\_date)=2020 and month(order\_date)=2

group by 1

having sum(unit)>99

50. Find Users With Valid E-Mails

# Write your MySQL query statement below

select

    \*

from

    users

where

    mail REGEXP '^[a-zA-Z][a-zA-Z0-9\_.-]\*@leetcode[.]com$';