

SQL - Intermediate - Part 3

Challenges

<https://www.hackerrank.com/challenges/challenges/problem?isFullScreen=true>

hacker-id, name and total no of challenges created by
each student

sort it by descending order.

if more than one

sort it by hacker-id

if count is less than max. no of challenges exclude them
if more than one student created same no of challenges.

Hackers:-

hacker_id
name

Challenges

challenge-id
hacker-id.

Challenge to write max-challenge count
maximum of the challenges = 50

```
SELECT MAX (challenges-count) FROM  
(SELECT count (challenge-id) as challenges-count  
FROM challenges GROUP BY hacker-id)
```

if it is
maximum
duplicates
are allowed
from question

gives count of no. of challenges created by each hacker

Now the next part is how to find the number
of challenges which are duplicate which means
that there count > 1.

```

SELECT challenges - count FROM
(
  SELECT challenges - count FROM
    (SELECT count (challenge - id) as challenge - count
     FROM challenges
     GROUP BY hacker - id) as Subquery
  GROUP BY challenges - count
  HAVING count (*) > 1)

```

group by all along challenges to check there count

list of challenges which are duplicates

```

SELECT h.hacker_id, h.name, COUNT(c.challenge_id) as challenges_count
FROM Hackers h
JOIN Challenges c ON h.hacker_id = c.hacker_id
GROUP BY h.hacker_id, h.name
HAVING COUNT(c.challenge_id) = (SELECT MAX(challenges_count) FROM
                                (SELECT COUNT(challenge_id) as challenges_count
                                 FROM Challenges
                                 GROUP BY hacker_id) as SubQuery)
OR (COUNT(c.challenge_id) NOT IN (SELECT challenges_count FROM
                                   (SELECT COUNT(challenge_id) as challenges_count
                                    FROM Challenges
                                    GROUP BY hacker_id) as SubQuery
                                   GROUP BY challenges_count
                                   HAVING COUNT(*) > 1))
ORDER BY challenges_count DESC, h.hacker_id;

```

join on the table

maximum duplicates allowed

check for duplicates and don't consider them

challenges - count in descending order

hacker - id in ascending order.

Symmetric Pairs

<https://www.hackerrank.com/challenges/symmetric-pairs/problem?isFullScreen=true>

Use of CTE in SQL \rightarrow named result set in query

Syntax :- WITH cte as ()
select statement query.

for example in this case

with a as (select x,y, row-number() over()
from functions)

Condⁿ in this question:-

Symmetric pair $x_1 = x_2$ and $x_2 = x_1$

Query by ascending order of x_i and order s.t. $x_i \leq y_i$

```
with a as ( select x, y, row_number() over() r from functions ),
b as ( select x, y, row_number() over() r from functions )

select distinct a.x,a.y
from a,b → self join
where a.x=b.y and a.y=b.x and a.r!=b.r and a.x<=a.y
order by a.x;

order by x1
→ no repetition
according to question
```

Interviews

<https://www.hackerrank.com/challenges/interviews/problem?isFullScreen=true>

contest_id, hacker_id, name

sums of total_submissions, total_accepted_submissions
total_views and total_unique_views
for each contest by contest_id

Exclude contest if all sum = 0

Specific contest → many college

1 college - 1 contest

Column	Type
contest_id	Integer
hacker_id	Integer
name	String

Contests

Column	Type
challenge_id	Integer
college_id	Integer

Colleges

Column	Type
challenge_id	Integer
total_views	Integer
total_unique_views	Integer

Challenges

Column	Type
challenge_id	Integer
total_views	Integer
total_unique_views	Integer

View - Stats

Column	Type
challenge_id	Integer
total_views	Integer
total_unique_views	Integer

Submission - Stats

SELECT

```
cc.contest_id,  
cc.hacker_id,  
cc.name,  
COALESCE(SUM(sa.total_submissions_ch), 0) AS total_submission_contest,  
COALESCE(SUM(sa.total_accepted_submissions_ch), 0) AS  
total_accepted_submission_contest,  
COALESCE(SUM(va.total_views_ch), 0) AS total_views_contest,  
COALESCE(SUM(va.total_unique_views_ch), 0) AS total_unique_views_contest
```

FROM → table of contest and

```
(SELECT c.contest_id, c.hacker_id, c.name, ch.challenge_id  
FROM colleges col
```

```
JOIN contests c ON col.contest_id = c.contest_id
```

```
JOIN challenges ch ON ch.college_id = col.college_id) AS cc
```

challenge joined
with college

Since she forgot
about which challenges
are there in contest

LEFT JOIN

```
(SELECT challenge_id,
```

↓ Sum over challenges

```
SUM(total_submissions) AS total_submissions_ch,
```

```
SUM(total_accepted_submissions) AS total_accepted_submissions_ch
```

```
FROM submission_stats
```

```
GROUP BY challenge_id) AS sa ON sa.challenge_id = cc.challenge_id
```

LEFT JOIN

```
(SELECT challenge_id,
```

↓ Sum over challenges

```
SUM(total_views) AS total_views_ch,
```

```
SUM(total_unique_views) AS total_unique_views_ch
```

```
FROM view_stats
```

```
GROUP BY challenge_id) AS va ON va.challenge_id = cc.challenge_id
```

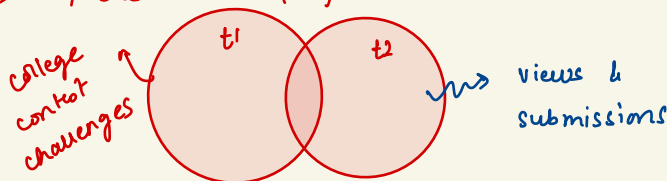
```
GROUP BY cc.contest_id, cc.hacker_id, cc.name
```

```
HAVING (total_submission_contest + total_accepted_submission_contest +  
total_views_contest + total_unique_views_contest) > 0
```

```
ORDER BY cc.contest_id;
```

why left join? → because there might or might not be data which is useful in the tables of views or submissions

But all the data of challenges, contest and colleges must be there



SQL- Advanced - Part 1

Draw the Triangle 1

<https://www.hackerrank.com/challenges/draw-the-triangle-1/problem?isFullScreen=true>

```

* * * * *
* * * *
* * *
* *
*

```

Query to print (20)

Python code :-

```

count = n q no of stars in first row
while count > 1: q till it becomes 1
    print ('* ' * count)
    count = count - 1

```

My SQL

```

SET @count = 21
SELECT REPEAT('*', @count) =
        @count - 1)
FROM information - schema . tables
where @count > 0

```

MS SQL Server

```

DECLARE @count INT = 20
WHILE (@count > 0)
BEGIN
    PRINT REPLICATE('*', @count)
    SET @count = @count - 1
END

```

Draw the Triangle 2

<https://www.hackerrank.com/challenges/draw-the-triangle-2/problem?isFullScreen=true>

My SQL

```

SET @count = 0
SELECT REPEAT('*', @count) =
        @count + 1)
FROM information - schema . tables
WHERE @count < 20;

```

MS SQL Server

```

DECLARE @count INT = 1
WHILE (@count < 21)
BEGIN
    PRINT REPLICATE('*', @count)
    SET @count = @count + 1
END

```