Interviews

HackerRank

Samantha interviews many candidates from different colleges using coding challenges and contests. Write a query to print the *contest_id*, *hacker_id*, *name*, and the sums of *total_submissions*, *total_accepted_submissions*, *total_views*, and *total_unique_views* for each contest sorted by *contest_id*. Exclude the contest from the result if all four sums are 0.

Note: A specific contest can be used to screen candidates at more than one college, but each college only holds 1 screening contest.

Input Format

The following tables hold interview data:

- *Contests:* The *contest_id* is the id of the contest, *hacker_id* is the id of the hacker who created the contest, and *name* is the name of the hacker.

Column	Type
contest_id	Integer
hacker_id	Integer
name	String

- *Colleges:* The *college_id* is the id of the college, and *contest_id* is the id of the contest that Samantha used to screen the candidates.

Column	Type
college_id	Integer
contest_id	Integer

- *Challenges:* The *challenge_id* is the id of the challenge that belongs to one of the contests whose contest_id Samantha forgot, and *college_id* is the id of the college where the challenge was given to candidates.

Column	Type
challenge_id	Integer
college_id	Integer

- *View_Stats:* The *challenge_id* is the id of the challenge, *total_views* is the number of times the challenge was viewed by candidates, and *total_unique_views* is the number of times the challenge was viewed by unique candidates.

Column	Type
challenge_id	Integer
total_views	Integer
total_unique_views	Integer

- *Submission_Stats:* The *challenge_id* is the id of the challenge, *total_submissions* is the number of submissions for the challenge, and *total_accepted_submission* is the number of submissions that achieved full scores.

Column	Type
challenge_id	Integer
total_submissions	Integer
total_accepted_submissions	Integer

Sample Input

Contests Table:

contest_id	hacker_id	name
66406	17973	Rose
66556	79153	Angela
94828	80275	Frank

Colleges Table:

college_id	contest_id
11219	66406
32473	66556
56685	94828

Challenges Table:

challenge_id	college_id
18765	11219
47127	11219
60292	32473
72974	56685

View_Stats Table:

challenge_id	total_views	total_unique_views
47127	26	19
47127	15	14
18765	43	10
18765	72	13
75516	35	17
60292	11	10
72974	41	15
75516	75	11

Submission_Stats Table:

challenge_id	total_submissions	total_accepted_submissions
75516	34	12
47127	27	10
47127	56	18
75516	74	12
75516	83	8
72974	68	24
72974	82	14
47127	28	11

Sample Output

```
66406 17973 Rose 111 39 156 56
66556 79153 Angela 0 0 11 10
94828 80275 Frank 150 38 41 15
```

Explanation

The contest 66406 is used in the college 11219. In this college 11219, challenges 18765 and 47127 are asked, so from the *view* and *submission* stats:

- Sum of total submissions $= 27 + 56 + 28 = 111$
- Sum of total accepted submissions $= 10 + 18 + 11 = 39$
- Sum of total views $= 43 + 72 + 26 + 15 = 156$
- Sum of total unique views $= 10 + 13 + 19 + 14 = 56$

Similarly, we can find the sums for contests 66556 and 94828.

RESPOSTA:

```
SELECT con.contest_id, con.hacker_id, con.name, SUM(sg.total_submissions), SUM(sg.total_accepted_submissions),
SUM(vg.total_views), SUM(vg.total_unique_views)
FROM Contests AS con
JOIN Colleges AS col
ON con.contest_id = col.contest_id
JOIN Challenges AS cha
ON cha.college_id = col.college_id
LEFT JOIN
(SELECT ss.challenge_id, SUM(ss.total_submissions) AS total_submissions, SUM(ss.total_accepted_submissions) AS
total_accepted_submissions FROM
Submission_Stats AS ss GROUP BY ss.challenge_id) AS sg
ON cha.challenge_id = sg.challenge_id
LEFT JOIN
(SELECT vs.challenge_id, SUM(vs.total_views) AS total_views, SUM(total_unique_views) AS total_unique_views FROM
View_Stats AS vs GROUP BY vs.challenge_id) AS vg
ON cha.challenge_id = vg.challenge_id
GROUP BY con.contest_id, con.hacker_id, con.name
HAVING SUM(sg.total_submissions)+
    SUM(sg.total_accepted_submissions)+
    SUM(vg.total_views)+
    SUM(vg.total_unique_views) > 0
ORDER BY con.contest_id;
```