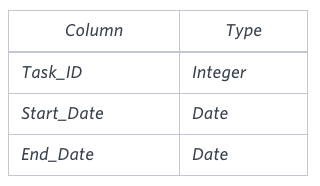
**Project 1.**

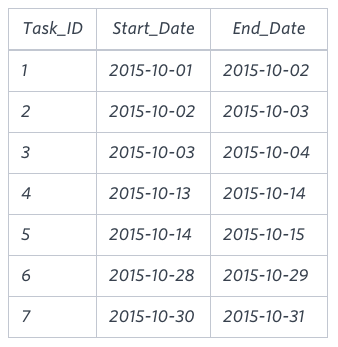
1. You are given a table, *Projects*, containing three columns: *Task\_ID*, *Start\_Date* and *End\_Date*. It is guaranteed that the difference between the *End\_Date* and the *Start\_Date* is equal to *1* day for each row in the table.



If the *End\_Date* of the tasks are consecutive, then they are part of the same project. Samantha is interested in finding the total number of different projects completed.

Write a query to output the start and end dates of projects listed by the number of days it took to complete the project in ascending order. If there is more than one project that have the same number of completion days, then order by the start date of the project.

**Sample Input**



**Sample Output**

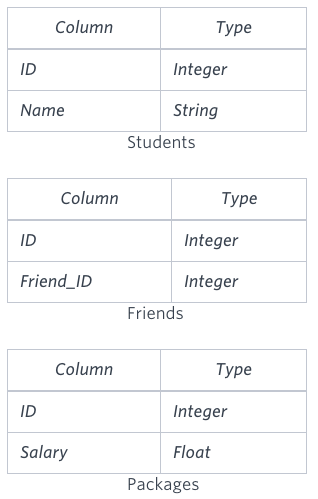
2015-10-28 2015-10-29

2015-10-30 2015-10-31

2015-10-13 2015-10-15

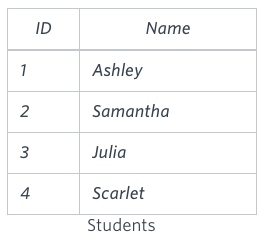
2015-10-01 2015-10-04

**2.** You are given three tables: *Students*,*Friends*and*Packages.* *Students* contains two columns: *ID* and *Name*. *Friends* contains two columns: *ID* and *Friend\_ID* (*ID* of the ONLY best friend). *Packages* contains two columns: *ID* and *Salary* (offered salary in $ thousands per month).



Write a query to output the names of those students whose best friends got offered a higher salary than them. Names must be ordered by the salary amount offered to the best friends. It is guaranteed that no two students got same salary offer.

**Sample Input**

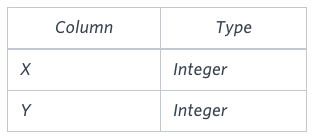
**Sample Output**

Samantha

Julia

Scarlet

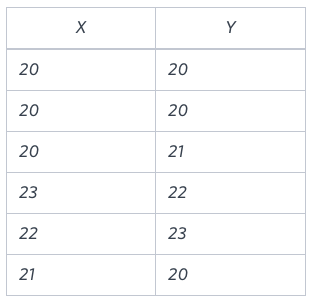
3.You are given a table, *Functions*, containing two columns: *X*and *Y*.



Two pairs *(X1, Y1)* and *(X2, Y2)* are said to be *symmetric* *pairs* if *X1 = Y2* and *X2 = Y1*.

Write a query to output all such *symmetric* *pairs* in ascending order by the value of *X*.

**Sample Input**



**Sample Output**

20 20

20 21

22 23

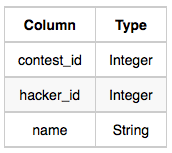
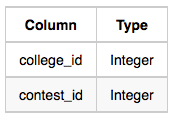
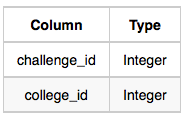
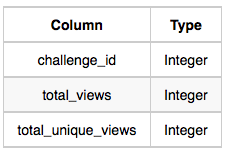
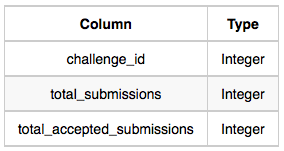
**4.** Samantha interviews many candidates from different colleges using coding challenges and contests. Write a query to print the *contest\_id*, *hacker\_id*, *name*, and the sums of *total\_submissions*, *total\_accepted\_submissions*, *total\_views*, and *total\_unique\_views* for each contest sorted by *contest\_id*. Exclude the contest from the result if all four sums are .

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

Inserting image...

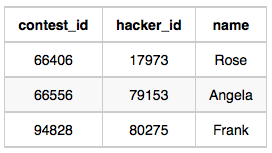
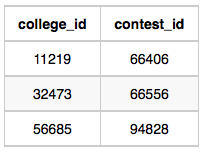
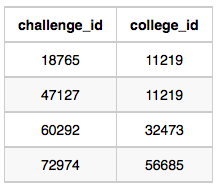
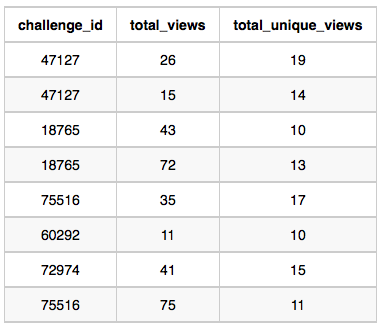
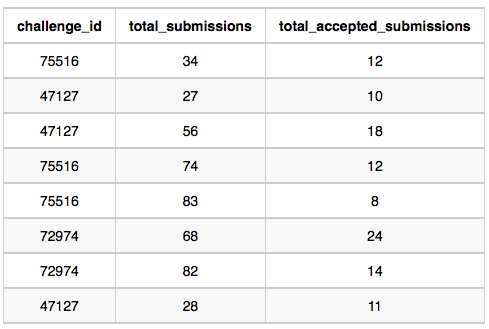
**Input Format**

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. 
* *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. 
* *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. 
* *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. 
* *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. 

Inserting image...

**Sample Input**

*Contests* Table:  *Colleges* Table:  *Challenges* Table:  *View\_Stats* Table: Submission\_Stats Table: 

**Sample Output**

66406 17973 Rose 111 39 156 56

66556 79153 Angela 0 0 11 10

94828 80275 Frank 150 38 41 15