Hackerrank SQL 題

題目最好都看本文件中的題目, 因為網頁上的題目有的意思不明(或意思錯), 有的的表故意刷屏. 本文件把這些問題都消除了.

以下代碼都是在 MySQL 中通過的.

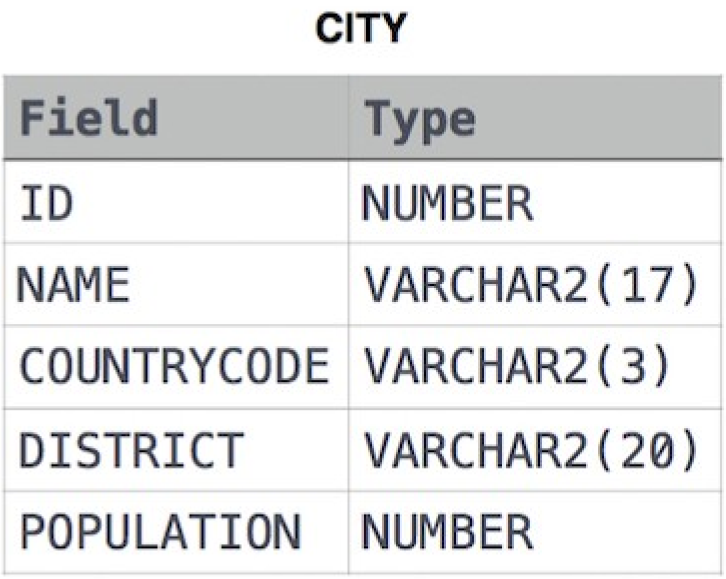
Basic Select

Revising the Select Query I , Easy

Query all columns for all American cities in CITY with populations larger than 100000. The CountryCode for America is USA.

Input Format

The CITY table is described as follows:



CITY

Field	Type
ID	NUMBER
NAME	VARCHAR2(17)
COUNTRYCODE	VARCHAR2(3)
DISTRICT	VARCHAR2(20)
POPULATION	NUMBER

History:
E1 幾分鍾寫好, 改了 POPULATION 的 typo 後即通過. E1 代碼跟 Top Solution 中的一樣. Discussion 中有些代碼也是用的大寫(SELECT, WHERE 等). 這是我人生刷的第一邊 SQL 題, 也是人生第一次自己寫 SQL 代碼.
E2 秒過.

E1 代碼:

```
SELECT *
FROM CITY
WHERE COUNTRYCODE = 'USA' AND POPULATION > 100000;
```

上句不要分號也能通過, Top Solution 中有些人就沒用分號.
大小寫也無所謂, FROM CITY 寫為 fRoM cItY, 也能通過. 後面的題表明, count 寫為 cOUnt 也能通過.

Revising the Select Query II, Easy

Query the names of all American cities in CITY with populations larger than 120000. The CountryCode for America is USA.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:
E1 一分鍾寫好, 一次通過, 代碼跟 Top Solution 中的一樣.
E2 秒過.

E1 的代碼:

```
SELECT NAME
FROM CITY
WHERE COUNTRYCODE = 'USA' AND POPULATION > 120000;
```

Select All , Easy

Query all columns (attributes) for every row in the CITY table.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:
E1 幾秒鍾寫好, 一次通過, 代碼跟 Top Solution 中的一樣.
E2 秒過.

E1 的代碼:

```
SELECT *
FROM CITY;
```

Select By ID, Easy

Query all columns for a city in CITY with the ID 1661.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:

E1 幾秒鍾寫好, 一次通過, 代碼跟 Top Solution 中的一樣. 這些題目太簡單, 略浪費我時間.
E2 秒過.

E1 的代碼:

```
SELECT *
FROM CITY
WHERE ID = 1661; #寫為 where ID = '1661'也能通過
```

Japanese Cities' Attributes, Easy

Query all attributes of every Japanese city in the CITY table. The COUNTRYCODE for Japan is JPN.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).
History:
E1 幾秒鍾寫好, 一次通過, 代碼跟 Top Solution 中的一樣.
E2 秒過.

E1 的代碼:

```
SELECT *
FROM CITY
WHERE COUNTRYCODE = 'JPN'
```

Japanese Cities' Names, Easy

Query the names of all the Japanese cities in the CITY table. The COUNTRYCODE for Japan is JPN.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:
E1 幾秒鍾寫好, 一次通過, 代碼跟 Top Solution 中的一樣.
E2 秒過.

E1 的代碼:

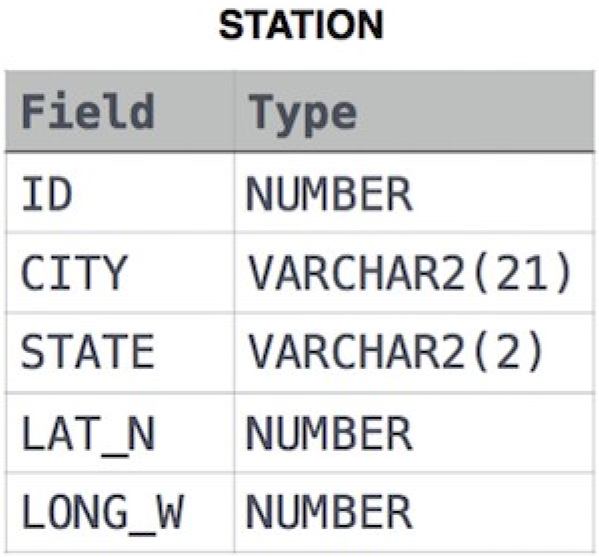
```
SELECT NAME
FROM CITY
WHERE COUNTRYCODE = 'JPN';
```

Weather Observation Station 1, Easy

Query a list of CITY and STATE from the STATION table.

Input Format

The STATION table is described as follows:



STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

History:
E1 幾秒鍾寫好, 一次通過, 代碼跟 Top Solution 中的一樣.
E2 秒過.

E1 的代碼:

```
SELECT CITY, STATE
FROM STATION;
```

Weather Observation Station 3, Easy

Query a list of CITY names from STATION with even ID numbers only. You may print the results in any order, but must exclude duplicates from your answer.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

History:
E1 一分鍾寫好, 一次通過, 代碼跟 Top Solution 中的一樣. Weather Observation Station 2 在 Aggregation 的那類題目中.
E2 秒過.

E1 的代碼:

```
SELECT DISTINCT(CITY)
FROM STATION
WHERE MOD(ID, 2) = 0;
```

上句寫為 WHERE ID % 2 = 0;也能通過, Top Solution 中有的人用的 MOD, 有的用的%.

Weather Observation Station 4, Easy

Let N be the number of CITY entries in STATION, and let N' be the number of distinct CITY names in STATION; query the value of N - N' from STATION. In other words, find the difference between the total number of CITY entries in the table and the number of distinct CITY entries in the table.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

History:
E1 沒做出來, 看的 Discussion. Discussion 中的代碼也通不過, E1 小改後即通過.
E2 秒過.

Top Solution 的代碼(E1 小改後):
SELECT (count(CITY) - count(DISTINCT(CITY)))
FROM STATION;

以上 count 寫為 cOUnt, 也能通過.

Weather Observation Station 5, Easy

Query the two cities in STATION with the shortest and longest CITY names, as well as their respective lengths (i.e.: number of characters in the name). If there is more than one smallest or largest city, choose the one that comes first when ordered alphabetically.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

Sample Input

Let's say that CITY only has four entries: DEF, ABC, PQRS and WXY

Sample Output

ABC 3
PQRS 4
Explanation

When ordered alphabetically, the CITY names are listed as ABC, DEF, PQRS, and WXY, with the respective lengths and . The longest-named city is obviously PQRS, but there are options for shortest-named city; we choose ABC, because it comes first alphabetically.

History:
E1 沒做出來.
E2 沒做出來, E2 方法想錯了, 是按 ORDER BY CITY 寫的.

Top Solution:

```
SELECT MIN(CITY), LENGTH(CITY)
FROM STATION
GROUP BY LENGTH(CITY)
ORDER BY LENGTH(CITY) limit 1; #此分號必須寫, 否則通不過.
```

```
SELECT MIN(CITY), LENGTH(CITY)
FROM STATION
GROUP BY LENGTH(CITY)
ORDER BY LENGTH(CITY) desc limit 1; #此分號可以不寫
```

以上代碼的解釋:
看 SQL-pick-up 這個文件中對 Group by 的講解, 就明白了. 本代碼可以類似理解: 第一段代碼即, 先將所有 CITY 按其長度分 group, 相同長度的為一個 group. 然後在每一個 group 中, 返回 alphabetically 最小的 CITY 及其 長度. 將結果按 CITY 長度 從小到大排列, 並只返回一個(即 limit 1).

The SQL SELECT LIMIT statement is used to retrieve records from one or more tables in a database and limit the number of records returned based on a limit value.

TIP: SELECT LIMIT is not supported in all SQL databases. 但以上代碼可在 MySQL 中通過.

Weather Observation Station 6, Easy

Query the list of CITY names starting with vowels (a, e, i, o, u) from STATION. Your result cannot contain duplicates.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 幾分鍾寫好, 一次通過. E1 是用的 LIKE 'a%'等, 代碼不及 Top Solution 的 REGEXP 簡明. 以下把 E1 和 Top Solution 都帖出來. 由於我對 regular expression 不熟, 所以還是要主要掌握我 E1 的代碼.
E2 沒做出來, 因為 E2 將 REGEXP 寫成了 LIKE.

E1 的代碼:

```
SELECT DISTINCT(CITY)
FROM STATION
WHERE CITY LIKE 'a%' OR CITY LIKE 'e%' OR CITY LIKE 'i%' OR CITY LIKE 'o%' OR CITY
LIKE 'u%';
```

Top Solution:

```
SELECT DISTINCT city
FROM station
WHERE city REGEXP "^[aeiou].*";
```

以上的雙引號改為單引號也能通過.

From Wiki Regular expression (注意沒說是 SQL, 說明是通用的)(以下符號都是後面題目中會用到的):

^ Matches the starting position within the string.

$ Matches the ending position of the string or the position just before a string-ending newline.

[ ] A bracket expression. Matches a single character that is contained within the brackets. For example, [abc] matches "a", "b", or "c". [a-z] specifies a range which matches any lowercase letter from "a" to "z". These forms can be mixed: [abcx-z] matches "a", "b", "c", "x", "y", or "z", as does [a-cx-z].

[^] Matches a single character that is not contained within the brackets. For example, [^abc] matches any character other than "a", "b", or "c". [^a-z] matches any single character that is not a lowercase letter from "a" to "z". Likewise, literal characters and ranges can be mixed.

. Matches any single character (many applications exclude newlines)

* Matches the preceding element zero or more times. For example, ab*c matches "ac", "abc", "abbbc", etc. [xyz]* matches "", "x", "y", "z", "zx", "zyx", "xyzzy", and so on. (ab)* matches "", "ab", "abab", "ababab", and so on.

Weather Observation Station 7, Easy

Query the list of CITY names ending with vowels (a, e, i, o, u) from STATION. Your result cannot contain duplicates.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 最開始用%通過, 然後想寫 Regular expression, 寫成的是"[aeiou].*$", 結果沒通過, 後來看 Discussion 才知道是 "[aeiou]$", 俱體原因不大清楚.
E2 秒過, 按 Top Solution 寫的.

Top Solution:

```
SELECT DISTINCT(CITY)
FROM STATION
WHERE CITY REGEXP "[aeiou]$";
```

E1 的代碼:

```
SELECT DISTINCT(CITY)
FROM STATION
WHERE CITY LIKE '%a' OR CITY LIKE '%e' OR CITY LIKE '%i' OR CITY LIKE '%o' OR CITY
LIKE '%u';
```

Weather Observation Station 8, Easy

Query the list of CITY names from STATION which have vowels (i.e., a, e, i, o, and u) as both their first and last characters. Your result cannot contain duplicates.

Input Format

The STATION table is described as follows(STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 改了幾次後才通過. Discussion 中的代碼都沒我好 E1 好.
E2 改了幾次後才通過.

E1 的代碼:

```
SELECT DISTINCT(CITY)
FROM STATION
WHERE CITY REGEXP "^[aeiou].*[aeiou]$";
```

Weather Observation Station 9, Easy

Query the list of CITY names from STATION that do not start with vowels. Your result cannot contain duplicates.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 用的 NOT IN, 一次通過. E1 代碼沒有 Top Solution 好, 以後用 Top Solution.
E2 秒過, E2 是寫為的"^[^aeiou].*"

Top Solution:

```
SELECT DISTINCT CITY
FROM STATION
WHERE CITY REGEXP '^[^aeiou]'; #寫為"^[^aeiou].*"也能通過.
```

E1 的代碼:

```
SELECT DISTINCT (CITY)
FROM STATION
WHERE CITY NOT IN
   (SELECT DISTINCT(CITY)
    FROM STATION
    WHERE CITY REGEXP "^[aeiou].*");
```

Weather Observation Station 10, Easy

Query the list of CITY names from STATION that do not end with vowels. Your result cannot contain duplicates.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 一分鍾寫好, 一次通過. Discussion 中的代碼都是用的 NOT IN 甚麼的, 沒有我 E1 的好.
E2 秒過.

E1 的代碼:

```
SELECT DISTINCT(CITY)
FROM STATION
WHERE CITY REGEXP "[^aeiou]$";
```

Weather Observation Station 11, Easy

Query the list of CITY names from STATION that either do not start with vowels or do not end with vowels. Your result cannot contain duplicates.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 最開始把題目看錯了, 將 or 看成了 and. 改後即通過. Discussion 中的代碼都沒 E1 的好.
E2 秒過.

E1 的代碼:

```
SELECT DISTINCT(CITY)
FROM STATION
WHERE CITY REGEXP "^[^aeiou].*" OR CITY REGEXP "[^aeiou]$";
```

Weather Observation Station 12, Easy

Query the list of CITY names from STATION that do not start with vowels and do not end with vowels. Your result cannot contain duplicates.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 一分鍾寫好, 一次通過. Discussion 中有人代碼跟我 E1 一樣.
E2 秒過.

E1 的代碼:

```
SELECT DISTINCT (CITY)
FROM STATION
WHERE CITY REGEXP "^[^aeiou].*[^aeiou]$";
```

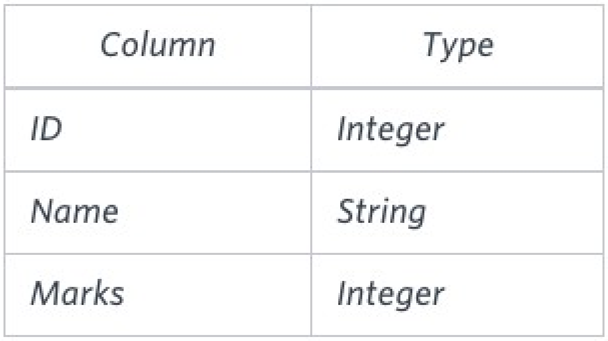
Higher Than 75 Marks, Easy

Query the Name of any student in STUDENTS who scored higher than 75 Marks. Order your output by the last three characters of each name. If two or more students both have names ending in the same last three characters (i.e.: Bobby, Robby, etc.), secondary sort them by ascending ID.

Input Format

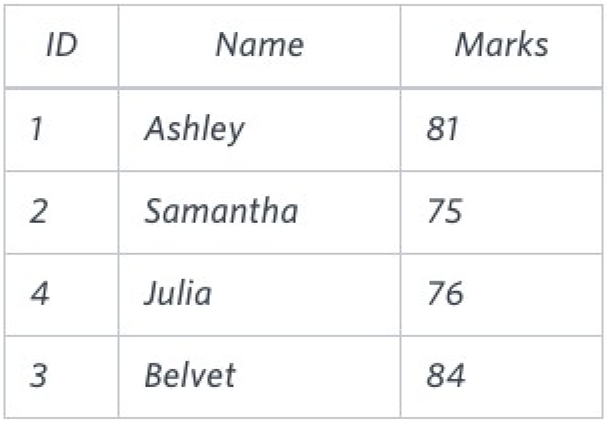
我將以下的表放在了同一行, 好方便看.

The STUDENTS table is described as follows:



Column	Type
ID	Integer
Name	String
Marks	Integer

STUDENTS



ID	Name	Marks
1	Ashley	81
2	Samantha	75
4	Julia	76
3	Belvet	84

Sample Input

The Name column only contains uppercase (A-Z) and lowercase (a-z) letters.

Sample Output

```
Ashley
Julia
Belvet
```

Explanation

Only Ashley, Julia, and Belvet have Marks > 75. If you look at the last three characters of each of their names, there are no duplicates and 'ley' < 'lia' < 'vet'.

History:
E1 沒通過, 原因是寫了兩行 ORDER BY, 看了答案後才知道兩行 ORDER BY 應該寫到一行.
E2 很快通過.

Top Solution:

```
Select Name
from STUDENTS
where Marks > 75
ORDER BY substr(Name,-3,3),ID;
```

以上最後一句寫為 ORDER BY substr(Name,length(Name)-2,3),ID;也能通過, 但 length 不能寫為 len.

SUBSTRING (expression, start, length): 函數意思自明. 要特別注注意的是, SQL 中 string 中的字符指標是從 1 開始的, 而不是從 0 開始, 例如:

```
SELECT SUBSTR(Store_Name,2,4)
FROM Geography
WHERE Store_Name = 'San Diego';
```

結果:

```
'an D'
```

這就是為甚麼本題中應寫為 substr(Name,length(Name)-2,3), 而不是 substr(Name,length(Name)-3,3).

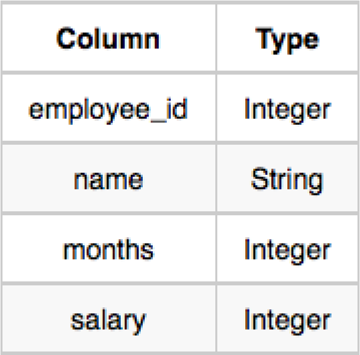
Employee Names, Easy

Write a query that prints a list of employee names (i.e.: the name attribute) from the **Employee** table in alphabetical order.

Input Format

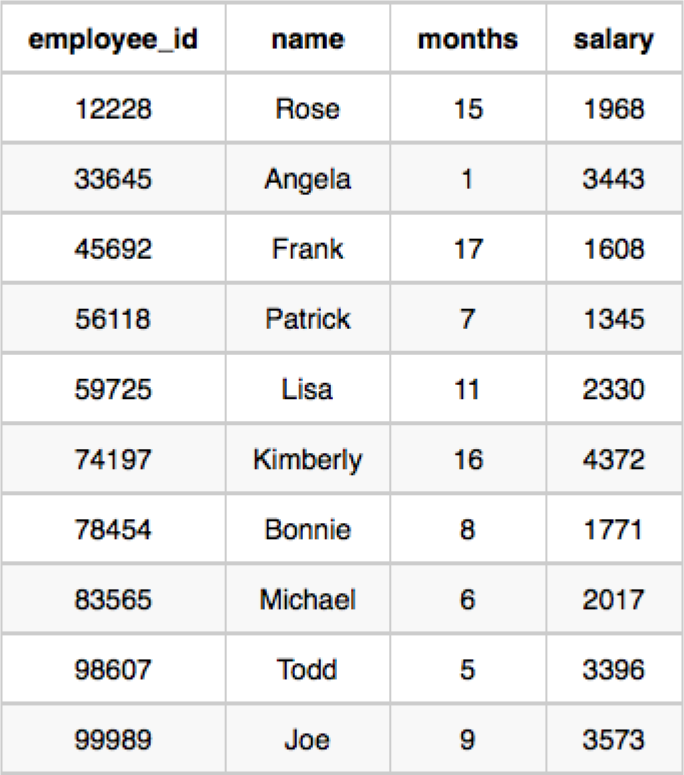
我將以下的表放在了同一行, 好方便看.

The **Employee** table containing employee data for a company is described as follows:



Column	Type
employee_id	Integer
name	String
months	Integer
salary	Integer

Employee



employee_id	name	months	salary
12228	Rose	15	1968
33645	Angela	1	3443
45692	Frank	17	1608
56118	Patrick	7	1345
59725	Lisa	11	2330
74197	Kimberly	16	4372
78454	Bonnie	8	1771
83565	Michael	6	2017
98607	Todd	5	3396
99989	Joe	9	3573

Sample Input

where employee_id is an employee's ID number, name is their name, months is the total number of months they've been working for the company, and salary is the their monthly salary.

Sample Output

Angela
Bonnie
Frank
Joe
Kimberly
Lisa

```
Michael
Patrick
Rose
Todd
```

History:
E1 幾秒鍾寫好, 一次通過. Discussion 中也是 E1 這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT name
FROM Employee
ORDER BY name;
```

Employee Salaries, Easy

Write a query that prints a list of employee names (i.e.: the name attribute) for employees in **Employee** having a salary greater than $2000 per month who have been employees for less than 10 months. Sort your result by ascending employee_id.

Input Format

The **Employee** table containing employee data for a company is described as follows (**Employee** 和 Sample Input 的圖都同上題).

where employee_id is an employee's ID number, name is their name, months is the total number of months they've been working for the company, and salary is the their monthly salary.

Sample Output

```
Angela
Michael
Todd
Joe
```

Explanation

Angela has been an employee for 1 month and earns $3443 per month.
Michael has been an employee for 6 months and earns $2017 per month.
Todd has been an employee for 5 months and earns $3396 per month.
Joe has been an employee for 9 months and earns $3573 per month.
We order our output by ascending employee_id.

History:
E1 一分鍾寫好, 一次通過. Discussion 中也是這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT name
FROM Employee
WHERE salary > 2000 AND months < 10
ORDER BY employee_id;
```

Advanced Select (沒做)

Aggregation

Revising Aggregations - The Count Function, Easy

Query a count of the number of cities in CITY having a Population larger than 100,000.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:
E1 幾秒鍾寫好, 結果把 POPULATION 一詞寫錯了, 改後即通過. Discussion 中也是這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT count(*)
FROM CITY
WHERE POPULATION > 100000;
```

Revising Aggregations - The Sum Function, Easy

Query the total population of all cities in CITY where District is California.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:
E1 幾秒鍾寫好, 結果把 SUM 寫為了 COUNT, 'CALIFORNIA'也忘了加單引號, 改後即通過. Discussion 中也是這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT SUM(POPULATION)
FROM CITY
WHERE DISTRICT = 'CALIFORNIA';
```

Revising Aggregations - Averages, Easy

Query the average population of all cities in CITY where District is California.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:
E1 幾秒鍾寫好, 一次通過. Discussion 中也是這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT AVG(POPULATION)
FROM CITY
WHERE DISTRICT = 'CALIFORNIA';
```

Average Population, Easy

Query the average population for all cities in CITY, rounded down to the nearest integer.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:
E1 最開始把題目看錯了, 以為還要求是 California, 改後即通過. Discussion 中也是這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT FLOOR(AVG(POPULATION))
FROM CITY;
```

From online:
FLOOR(), if you want to round your decimal to the lower integer. Examples:
FLOOR(12345.7344) => 12345

ROUND(), if you want the nearest integer to the argument. Examples:
ROUND(12345.7344) => 12346
ROUND(12345.4311) => 12345

CEIL(), if you want to round your decimal to the upper integer. Examples:
CEIL(12345.4311) => 12346

Japan Population, Easy

Query the sum of the populations for all Japanese cities in CITY. The COUNTRYCODE for Japan is JPN.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:
E1 幾秒鍾寫好, 一次通過. Discussion 中也是這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT SUM(POPULATION)
FROM CITY
WHERE COUNTRYCODE = 'JPN';
```

Population Density Difference, Easy

Query the difference between the maximum and minimum populations in CITY.

Input Format

The CITY table is described as follows (CITY table 圖見本文件夾).

History:
E1 幾秒鍾寫好, 一次通過. Discussion 中也是這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT MAX(POPULATION) - MIN(POPULATION)
FROM CITY;
```

The Blunder, Easy

以下題目廢話太多. 我將題意簡寫於此, 不用再看題目. 稱以下 EMPLOYEES 表(即第一個表)中的平均 Salary 為 actual. EMPLOYEES 表的一個例子即為以下第二個表. 現在要將第二個表中所有 Salary 中的 0 都去掉, 變成以下第三個表那樣, 稱第三個表中的平均 Salary 為 miscalculated. 求(actual - miscalculated), 並將結果 round up to the next integer.

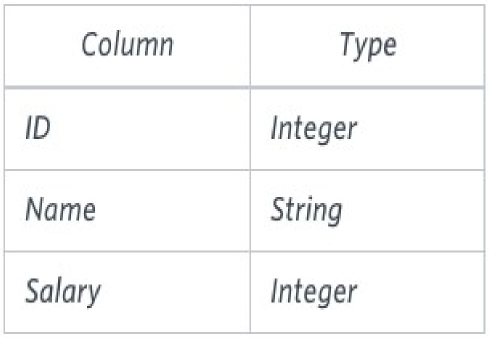
Samantha was tasked with calculating the average monthly salaries for all employees in the EMPLOYEES table, but did not realize her keyboard's 0 key was broken until after completing the calculation. She wants your help finding the difference between her miscalculation (using salaries with any zeroes removed), and the actual average salary.

Write a query calculating the amount of error (i.e.: actual - miscalculated average monthly salaries), and round it up to the next integer.

Input Format

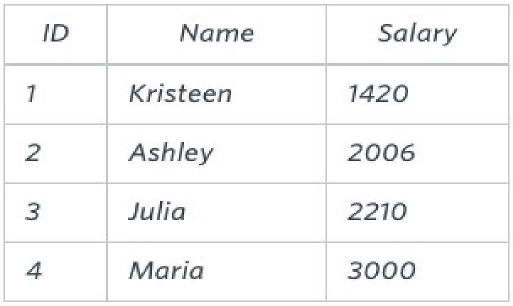
我將以下的表都放在了同一行, 好方便看.

The EMPLOYEES table is described as follows:



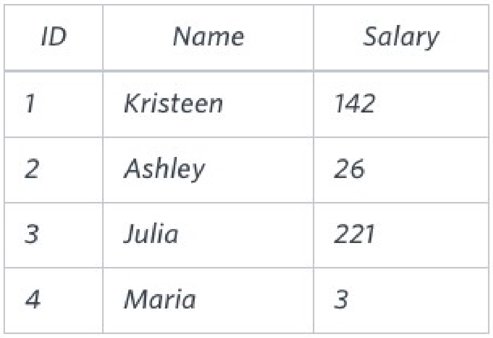
Column	Type
ID	Integer
Name	String
Salary	Integer

EMPLOYEES



ID	Name	Salary
1	Kristeen	1420
2	Ashley	2006
3	Julia	2210
4	Maria	3000

Sample Input



ID	Name	Salary
1	Kristeen	142
2	Ashley	26
3	Julia	221
4	Maria	3

Explanation

Note: Salary is measured in dollars per month and its value is < 10^5.

Sample Output

2061

Explanation

The table below (見上) shows the salaries without zeroes as they were entered by Samantha:

Samantha computes an average salary of 98.00. The actual average salary is 2159.00.

The resulting error between the two calculations is 2159.00 - 98.00 = 2061.00 which, when rounded to the next integer, is 2061.

History:
E1 直接看的答案.
E2 沒做出來, 原因是不小心將 AVG 和 REPLACE 寫反了, 即寫成了 REPLACE(AVG(SALARY, '0', ")))

Top Solution:

```
SELECT CEIL(AVG(SALARY) - AVG(REPLACE(SALARY, '0', ")))
FROM EMPLOYEES;
```

REPLACE(string1, string_to_replace, replacement_string), 作用自明. 例子:

```
SELECT REPLACE('TechOnTheNet.com', 'Tech', '1234');
Result: '1234OnTheNet.com'
```

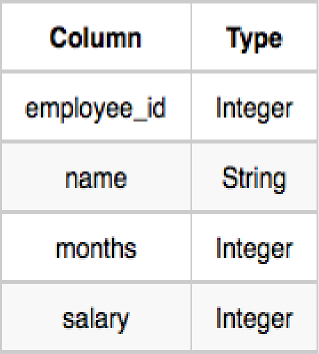
Top Earners, Easy

We define an employee's total earnings to be their monthly salary * months worked, and the maximum total earnings to be the maximum total earnings for any employee in the Employee table. Write a query to find the maximum total earnings for all employees as well as the total number of employees who have maximum total earnings. Then print these values as 2 「space-separated integers」.

Input Format

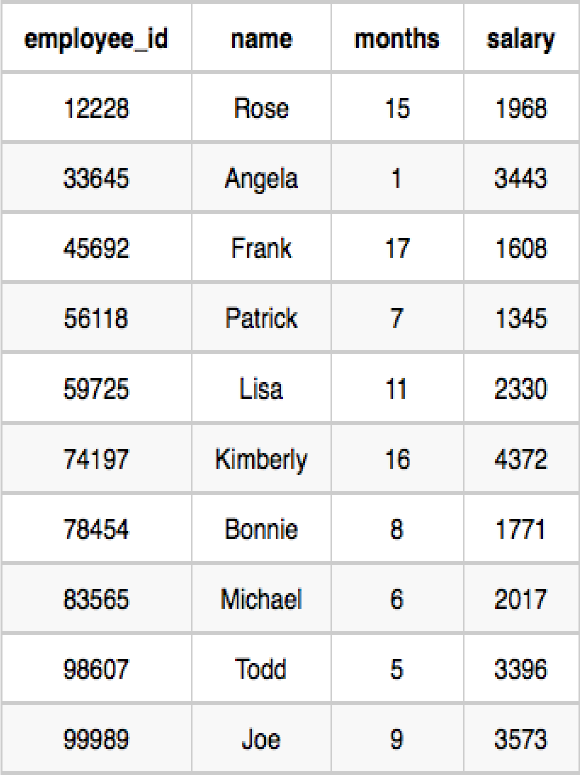
The Employee table containing employee data for a company is described as follows:

Employee



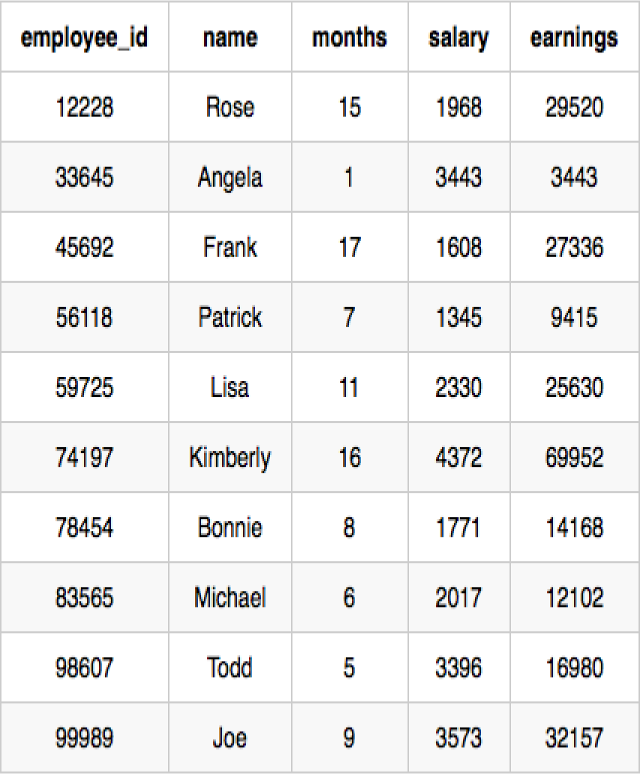
Column	Type
employee_id	Integer
name	String
months	Integer
salary	Integer

Sample Input



employee_id	name	months	salary
12228	Rose	15	1968
33645	Angela	1	3443
45692	Frank	17	1608
56118	Patrick	7	1345
59725	Lisa	11	2330
74197	Kimberly	16	4372
78454	Bonnie	8	1771
83565	Michael	6	2017
98607	Todd	5	3396
99989	Joe	9	3573

Explanation



employee_id	name	months	salary	earnings
12228	Rose	15	1968	29520
33645	Angela	1	3443	3443
45692	Frank	17	1608	27336
56118	Patrick	7	1345	9415
59725	Lisa	11	2330	25630
74197	Kimberly	16	4372	69952
78454	Bonnie	8	1771	14168
83565	Michael	6	2017	12102
98607	Todd	5	3396	16980
99989	Joe	9	3573	32157

where employee_id is an employee's ID number, name is their name, months is the total number of months they've been working for the company, and salary is the their monthly salary.

Sample Output

69952 1

Explanation

The table and earnings data is depicted in the following diagram (見上).

The maximum earnings value is 69952. The only employee with earnings = 69952 is Kimberly, so we print the maximum earnings value (69952) and a count of the number of employees who have earned $69952 (which is 1) as two space-separated values.

History:
E1 沒做出來.
E2 沒做出來, 原因是忘了寫 desc, 甚餘的都寫出來了.

以下代碼的理解(跟 SQL-quick-pick-up 中的那個 Group by 例子一樣的理解方式):
先按 earnings group 好,
再對每個 group, 得到 earnings 和 count, 用一行來表示該 group 的 earnings 和 count,
再將所有行以 earnings 來降序排序
最後只返回最前面的一行.

Top Solution:

```
select (salary * months) as earnings, count(*)
from employee
group by 1 #本句也可以寫為 group by earnings, 也能通過.
order by earnings desc
limit 1;
```

```
SELECT account_id, open_emp_id
         ^^^^        ^^^^
          1           2

FROM account
GROUP BY 1;
```

In above query GROUP BY 1 refers to the first column in select statement which is account_id.

You also can specify in ORDER BY.

Note : The number in ORDER BY and GROUP BY always start with 1 not with 0.

Weather Observation Station 2, Easy

Query the following two values from the STATION table:

1. The sum of all values in LAT_N rounded to a scale of 2 decimal places.
2. The sum of all values in LONG_W rounded to a scale of 2 decimal places.

Input Format

The STATION table is described as follows: (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

Output Format

Your results must be in the form:

lat lon
where lat is the sum of all values in LAT_N and lon is the sum of all values in LONG_W. Both results must be rounded to a scale of 2 decimal places.

History:
E1 一分鍾寫好, 一次通過. Discussion 中也是這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT ROUND(SUM(LAT_N), 2), ROUND(SUM(LONG_W), 2)
FROM STATION;
```

SELECT ROUND(column_name,decimals) FROM table_name;

Parameter	Description
column_name	Required. The field to round.
decimals	Required. Specifies the number of decimals to be returned. 即保留小數點後多少位.

ROUND 函數是按四舍五入弄的，但對於五，到底該往上弄還是往下弄，此文有介紹：http://www.w3schools.com/Sql/sql_func_round.asp

Weather Observation Station 13, Easy

Query the sum of Northern Latitudes (LAT_N) from STATION having values(即LAT_N values) greater than 38.7880 and less than 137.2345. Truncate your answer to 4 decimal places.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 最開始把題目看錯了, 沒看到 sum. 改後即通過. Discussion 中也是這樣寫的.
E2 速過.

E1 的代碼:

```
SELECT TRUNCATE(SUM(LAT_N), 4)
FROM STATION
WHERE LAT_N > 38.7880 AND LAT_N < 137.2345;
```

TRUNCATE() returns a number after truncated to certain decimal places.
TRUNCATE(N, D);
Arguments

Name	Description
N	A number which is to be truncated up to D decimal places.

D A number indicating up to how many decimal places, N is to be truncated. 即保留小數點後多少位.

Weather Observation Station 14, Easy

Query the greatest value of the Northern Latitudes (LAT_N) from STATION that is less than 137.2345. Truncate your answer to 4 decimal places.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 一分鍾寫好, 一次通過. Discussion 中也是這樣寫的.
E2 速過, E2 按 order by lat_n desc 寫的.

E1 的代碼:

```
SELECT TRUNCATE(MAX(LAT_N), 4)
FROM STATION
WHERE LAT_N < 137.2345
```

Weather Observation Station 15, Easy

題目意思不明確. 題目的意思就是: 在 STATION 這個表中找出這麼一些行(稱這些行為 A 類行): A 類行要滿兩個要求: 一是 A 類行的 LAT_N 都為 「LAT_N 在表中的最大值」, 二是 A 類行的 LAT_N 都要小於 137.2345. 最後要返回的是所有 A 類行的 LONG_W 值, 並且這些 LONG_W 值要 round to 4 decimal places.

Query the Western Longitude (LONG_W) for the largest Northern Latitude (LAT_N) in STATION that is less than 137.2345. Round your answer to 4 decimal places.

Input Format

The STATION table is described as follows: (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 幾分鍾寫好, 但把題目中的 Round 看成了 Truncate, 改後即通過. Discussion 中基本上也是這樣寫的.
E2 編譯沒通過, 不知道原因, E2 的代碼帖在最後.

E1 的代碼:

```
SELECT ROUND(LONG_W, 4)
FROM STATION
WHERE LAT_N IN
```

```
    (SELECT MAX(LAT_N)
    FROM STATION
    WHERE LAT_N < 137.2345);
```

E2 的代碼(編譯沒通過):

```
select round(long_w, 4)
from station
where lat_n < 137.2345 and lat_n in #將 and 兩邊去掉任何一個, 編譯都能通過
    (select max(lat_n)
     from station)
```

Weather Observation Station 16, Easy

Query the smallest Northern Latitude (LAT_N) from STATION that is greater than 38.7780. Round your answer to 4 decimal places.

Input Format

The STATION table is described as follows: (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 幾秒鍾寫好, 一次通過. Discussion 中差不多也是這樣寫的.
E2 秒過.

E1 的代碼:

```
SELECT ROUND(MIN(LAT_N), 4)
FROM STATION
WHERE LAT_N > 38.7780
```

Weather Observation Station 17, Easy

本題跟 Weather Observation Station 15 差不多.

Query the Western Longitude (LONG_W) for the smallest Northern Latitude (LAT_N) in STATION that is greater than 38.7780. Round your answer to 4 decimal places.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 一分鍾寫好, 一次通過. Discussion 中也是這樣寫的或類似方法寫的.

E1 的代碼:

```
SELECT ROUND(LONG_W, 4)
FROM STATION
WHERE LAT_N IN
    (SELECT MIN(LAT_N)
     FROM STATION
     WHERE LAT_N > 38.7780);
```

Weather Observation Station 18, Medium

Consider P_1 (a, b) and P_2 (c, d) to be two points on a 2D plane.

- a happens to equal the minimum value in Northern Latitude (LAT_N in STATION).
- b happens to equal the maximum value in Northern Latitude (LAT_N in STATION).
- c happens to equal the minimum value in Western Longitude (LONG_W in STATION).
- d happens to equal the maximum value in Western Longitude (LONG_W in STATION).

Query the Manhattan Distance between points P_1 and P_2 and round it to a scale of 4 decimal places.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 幾分鍾寫好, 一次通過. Discussion 中也是這樣寫的或類似方法寫的.
E2 秒過.

E1 的代碼:

```
SELECT ROUND(ABS(MIN(LAT_N) - MIN(LONG_W)) + ABS(MAX(LAT_N) -
MAX(LONG_W)), 4) #ABS(number) returns the absolute value of a number.
FROM STATION
```

Manhattan Distance:

$$d_1(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\|_1 = \sum_{i=1}^{n} |p_i - q_i|,$$

Weather Observation Station 19, Medium

Consider P_1 (a, b) and P_2 (c, d) to be two points on a 2D plane where (a, b) are the respective minimum and maximum values of Northern Latitude (LAT_N) and (c, d) are the respective minimum and maximum values of Western Longitude (LONG_W) in STATION.

Query the Euclidean Distance between points P_1 and P_2 and format your answer to display 4 decimal digits.

Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 幾分鍾寫好, 一次通過. E1 是用的乘法表示平方, Top Solution 是用的 POWER 函數. E1 那樣可能要多算幾次 MAX 和 MIN, 所以以後用 Top Solution.
E2 秒過.

Top Solution:

```
SELECT ROUND(SQRT(POWER(MAX(LONG_W) - MAX(LAT_N),2) + POWER(MIN(LONG_W) - MIN(LAT_N), 2)), 4)
FROM STATION;
```

E1 的代碼:

```
SELECT ROUND(SQRT((MIN(LAT_N) - MIN(LONG_W)) * (MIN(LAT_N) - MIN(LONG_W)) + (MAX(LAT_N) - MAX(LONG_W)) * (MAX(LAT_N) - MAX(LONG_W))), 4)
FROM STATION
```

SQRT(number) returns the square root of a non-negative number of the argument.
POWER(3, 2) returns the value of 3^2, i.e. 9.

Weather Observation Station 20, Medium

A median is defined as a number separating the higher half of a data set from the lower half. Query the median of the Northern Latitudes (LAT_N) from STATION and round your answer to 4 decimal places.

Note: Oracle solutions are not permitted for this challenge.

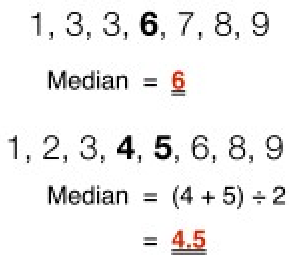
Input Format

The STATION table is described as follows (STATION table 圖見本文件夾).

where LAT_N is the northern latitude and LONG_W is the western longitude.

History:
E1 沒做出來.
E2 直接看的答案.

Wiki 中對 median 的定義:



1, 3, 3, **6**, 7, 8, 9

Median = 6

1, 2, 3, **4**, **5**, 6, 8, 9

Median = (4 + 5) ÷ 2

= 4.5

Top Solution(不用, 只看看, 因為只適用於奇數個數的情況):

看以下代碼時, 注意用我 SQL-quick-pick-up 對如何讀 correlated queries 的解釋. 以下代碼的意思即 找一個數, 使得小於它的數的個數 等於 大於它的數的個數. 這顯然只適用於奇數個數的情況(Discussion 中也有人這樣說了). 以下代碼能通過, 是因為 Hackerrank 上只有一個 test case. 更 General 的方法見下面網上的代碼.

```
Select round(S.LAT_N,4) mediam
from station S
where (select count(Lat_N)
       from station
       where Lat_N < S.LAT_N ) =
      (select count(Lat_N)
       from station
       where Lat_N > S.LAT_N)
```

網上的代碼(大概看看即可, 面試應該不會要求寫這麼複雜的):

Simple way to calculate median with MySQL

要看懂以下代碼, 要补充以下知識:

What does the “@” symbol do in SQL?
The @... means it's a parameter that you will supply a value for later in your code. This is the best way of protecting against SQL injection. Create your query using parameters, rather than concatenating strings and variables. The database engine puts the parameter value into where the placeholder is, and there is zero chance for SQL injection.

What does “:=” do?
It's the assignment operator in Pascal and is often used in proofs and pseudo-code. It's the same thing as = in C-dialect languages.
Historically, computer science papers used = for equality comparisons and ← for assignments. Pascal used := to stand in for the hard-to-type left arrow. C went a different direction and instead decided on the = and == operators.

What does “WHERE 1” mean in SQL?
It means ALWAYS TRUE so it won't have any filtering impact on your query. Query planner will probably ignore that clause.

```
SELECT avg(t1.val) as median_val FROM (
SELECT @rownum:=@rownum+1 as `row_number`, d.val
  FROM data d,  (SELECT @rownum:=0) r
  WHERE 1
  -- put some where clause here
  ORDER BY d.val
) as t1,
(
  SELECT count(*) as total_rows
  FROM data d
  WHERE 1
  -- put same where clause here
) as t2
WHERE 1
AND t1.row_number in ( floor((total_rows+1)/2), floor((total_rows+2)/2) );
```

Basic Join

Asian Population, Easy

Given the CITY and COUNTRY tables, query the sum of the populations of all cities where the CONTINENT is 'Asia'.

Note: CITY.CountryCode and COUNTRY.Code are matching key columns.

Input Format

The CITY and COUNTRY tables are described as follows (CITY 和 COUNTRY 圖見本文件夾).

History:
E1 最開始以為 CONTINENT 是 CITY 表中的, 寫成的 CITY.CONTINENT, 結果怎麼都通不過. 後來 run code 之後才發現 CONTINENT 是 COUNTRY 表中的, 改後即通過. Discussion 中沒人給出代碼.
E2 最開始忘了在 population 前加 city, 改後即通過.

E1 的代碼一:

```
SELECT SUM(CITY.POPULATION) #注意必須寫加 CITY, 因為 COUNTRY 也有 POPULATION
FROM CITY, COUNTRY
WHERE COUNTRYCODE = CODE AND CONTINENT = 'ASIA';
```

E1 的代碼二:

```
SELECT SUM(CITY.POPULATION)
FROM CITY JOIN COUNTRY ON COUNTRYCODE = CODE
WHERE CONTINENT = 'ASIA';
```

E2 的代碼:

```
select sum(city.population)
```

```
from city join country on city.countrycode = country.code and country.continent = 'Asia';
```

African Cities, Easy

Given the CITY and COUNTRY tables, query the names of all cities where the CONTINENT is 'Africa'.

Note: CITY.CountryCode and COUNTRY.Code are matching key columns.

Input Format

The CITY and COUNTRY tables are described as follows (CITY 和 COUNTRY 圖見本文件夾).

History:
E1 一分鍾寫好, 一次通過. Discussion 中也是類似代碼.
E2 秒過.

E1 的代碼:

```
SELECT CITY.NAME
FROM CITY, COUNTRY
WHERE CITY.COUNTRYCODE = COUNTRY.CODE AND COUNTRY.CONTINENT = 'AFRICA';
```

E2 的代碼:

```
select city.name
from city join country on city.countrycode = country.code and country.continent = 'Africa';
```

Average Population of Each Continent, Easy

Given the CITY and COUNTRY tables, query the names of all the continents (COUNTRY.Continent) and their respective average city populations (CITY.Population) rounded down to the nearest integer.

Note: CITY.CountryCode and COUNTRY.Code are matching key columns. Do not include continents without cities in your output.

Input Format

The CITY and COUNTRY tables are described as follows (CITY 和 COUNTRY 圖見本文件夾).

History:
E1 最開始將題目中的 round down 看成了 round, 改後即通過. Discussion 中也差不多是這麼寫的.
E2 速過.

E1 的代碼:

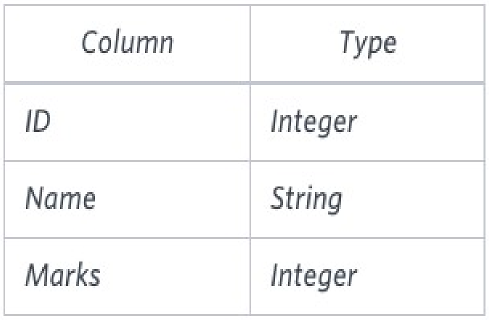
```
SELECT COUNTRY.CONTINENT, FLOOR(AVG(CITY.POPULATION))
FROM CITY, COUNTRY
WHERE CITY.COUNTRYCODE = COUNTRY.CODE
GROUP BY COUNTRY.CONTINENT
```

The Report, Medium

注意, 題目意思有問題, 題目中說:
Finally, if the grade is lower than 8, use "NULL" as their name and list them by their marks in ascending order.
Discussion 中說:
However, the test code is expecting these to be ordered by grade first (descending) then by mark (ascending).
實際上, 由已通過的代碼可以看出, 結果中所有行都是按
order by Grades.Grade desc, Students.Name, Students.Marks asc
這樣的順序來排序的.

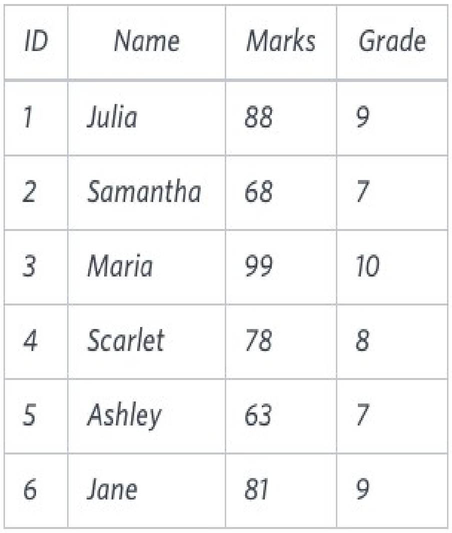
You are given two tables: Students and Grades. Students contains three columns ID, Name and Marks.

Students



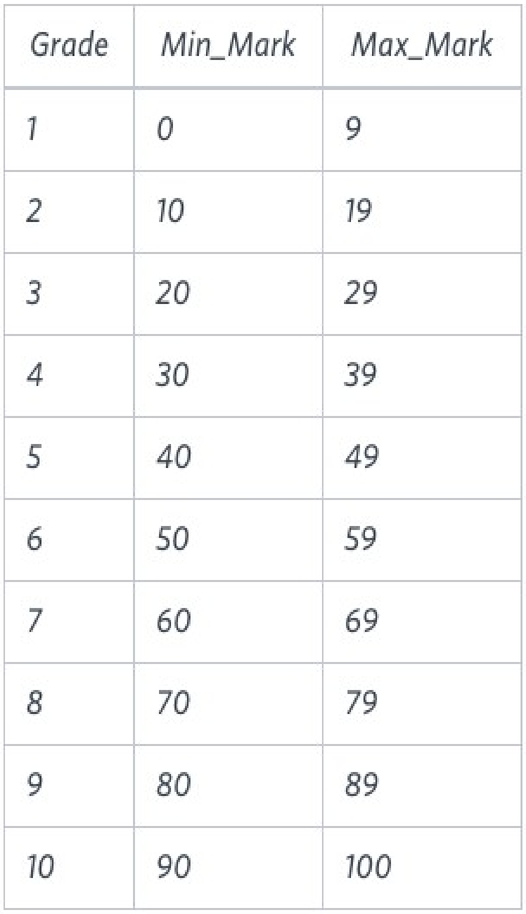
Column	Type
ID	Integer
Name	String
Marks	Integer

Explanation



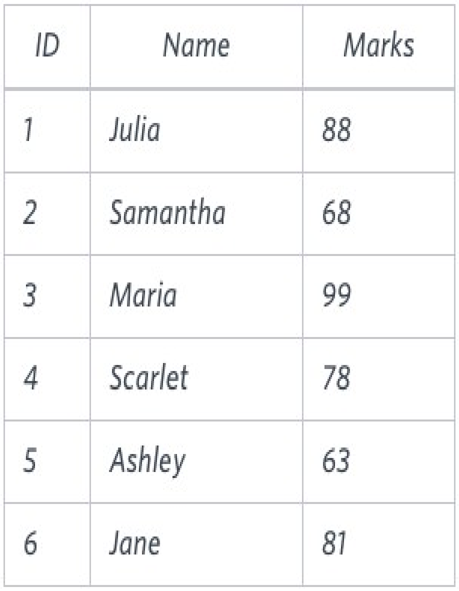
ID	Name	Marks	Grade
1	Julia	88	9
2	Samantha	68	7
3	Maria	99	10
4	Scarlet	78	8
5	Ashley	63	7
6	Jane	81	9

Grades



Grade	Min_Mark	Max_Mark
1	0	9
2	10	19
3	20	29
4	30	39
5	40	49
6	50	59
7	60	69
8	70	79
9	80	89
10	90	100

Sample Input



ID	Name	Marks
1	Julia	88
2	Samantha	68
3	Maria	99
4	Scarlet	78
5	Ashley	63
6	Jane	81

Grades contains the following data (見上)

Ketty gives Eve a task to generate a report containing three columns: Name, Grade and Mark (應為 Marks). Ketty doesn't want the NAMES of those students who received a grade lower than 8. The report must be in descending order by grade -- i.e. higher grades are entered first. If there is more than one student with the same grade (1-10) assigned to them, order those particular students by their name alphabetically. Finally, if the grade is lower than 8, use "NULL" as their name and list them by their marks in ascending order.

Write a query to help Eve.

Sample Output

```
Maria 10 99
Jane 9 81
Julia 9 88
Scarlet 8 78
NULL 7 63
NULL 7 68
```

Note

Print "NULL" as the name if the grade is less than 8.

Explanation

Consider the following table with the grades assigned to the students (見上).

So, the following students got 8, 9 or 10 grades:

Maria (grade 10)
Jane (grade 9)
Julia (grade 9)
Scarlet (grade 8)

History:
E1 沒做出來.
E2 一次通過. E2 的代碼沒用 concat, 比 Top Solution 好.

E2 的代碼(用它):

```
select if(grade < 8, 'NULL', name), grade, marks
from students, grades
where marks >= min_mark and marks <= max_mark
order by grade desc, name, marks asc
```

Top Solution(不用):

```
select if(Grades.Grade < 8, concat('NULL'), Students.Name), Grades.Grade, Students.Marks
from Students inner join Grades # inner join 寫為 join, 也能通過.
    on Students.Marks >= Grades.Min_Mark and Students.Marks <= Grades.Max_Mark
group by Grades.Grade desc, Students.Name, Students.Marks asc
#上句將 group by 改為 order by 也能通過, 建議用 order by, 意思更明確.
```

CONCAT(str1,str2,…):

返回结果为连接参数产生的字符串, 如有任何一个参数为 NULL, 则返回值为 NULL. 例:
select concat('10');
10

select concat('11','22','33');
112233

select concat('11','22',null);
NULL

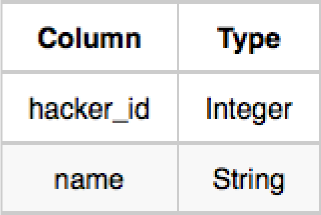
IF(expression, expr_true, expr_false):
if the first expression is true, not zero and not NULL,
it returns the second expression.
Otherwise, it returns the third expression.
Eg: SELECT IF(1>3,'true','false');

Top Competitors, Medium

Julia just finished conducting a coding contest, and she needs your help assembling the leaderboard! Write a query to print the respective hacker_id and name of hackers who achieved full scores for more than one challenge. Order your output in descending order by the total number of challenges in which the hacker earned a full score (Difficulty 中的 score 值即為 full score). If more than one hacker received full scores in same number of challenges, then sort them by ascending hacker_id.

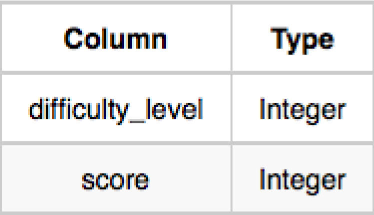
Input Format

The following tables contain contest data (原文不同的表都是竪著排的, 我將它們放在了同一行, 方便看):



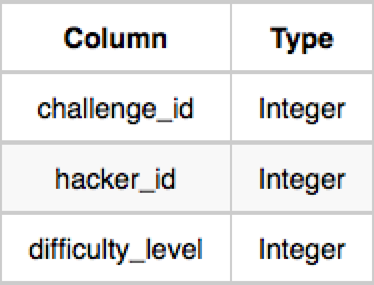
Column	Type
hacker_id	Integer
name	String

Hackers



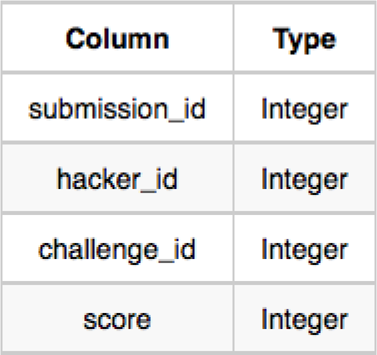
Column	Type
difficulty_level	Integer
score	Integer

Difficulty



Column	Type
challenge_id	Integer
hacker_id	Integer
difficulty_level	Integer

Challenges



Column	Type
submission_id	Integer
hacker_id	Integer
challenge_id	Integer
score	Integer

Submissions

Hackers: The hacker_id is the id of the hacker, and name is the name of the hacker.

Difficulty: The difficult_level is the level of difficulty of the challenge, and score is the (full) score of the challenge for the difficulty level.

Challenges: The challenge_id is the id of the challenge, the hacker_id is the id of the hacker who created the challenge, and difficulty_level is the level of difficulty of the challenge.

Submissions: The submission_id is the id of the submission, hacker_id is the id of the hacker who made the submission, challenge_id is the id of the challenge that the submission belongs to, and score is the score of the submission.

Sample Input

Hackers Table: (竪的表, 很高很窄, 刷屏狗, 已省).

Difficulty Table: (竪的表, 很高窄, 刷屏狗, 已省).

Challenges Table: (竪的表, 很高窄, 刷屏狗, 已省).

Submissions Table (竪的表, 很高窄, 刷屏狗, 已省).

Sample Output

90411 Joe

Explanation

Hacker 86870 got a score of 30 for challenge 71055 with a difficulty level of 2, so 86870 earned a full score for this challenge.

Hacker 90411 got a score of 30 for challenge 71055 with a difficulty level of 2, so 90411 earned a full score for this challenge.

Hacker 90411 got a score of 100 for challenge 66730 with a difficulty level of 6, so 90411 earned a full score for this challenge.

Only hacker 90411 managed to earn a full score for more than one challenge, so we print the their hacker_id and name as 2 space-separated values.

History:
E1 沒做出來, 後來看了 Discussion, 才發現可以用 having, E1 基本上都忘了還有 having 語句存在. 然然根据 Discussion 將我 E1 的代碼改了改, 即通過.
E2 沒做出來, E2 不知道 having 中還可以用 count.

E1 改了後的代碼:

```
select H.hacker_id as HID, H.name
from Hackers as H, Submissions as S, Challenges as C, Difficulty as D
where H.hacker_id = S.hacker_id and S.challenge_id = C.challenge_id and C.difficulty_level = D.difficulty_level and S.score = D.score
group by HID
having count(HID) > 1
order by count(HID) desc, HID;
```

Ollivander's Inventory, Medium

Harry Potter and his friends are at Ollivander's with Ron, finally replacing Charlie's old broken wand.

Hermione decides the best way to choose is by determining the minimum number of gold galleons needed to buy each non-evil wand of high power and age. Write a query to print the id, age, coins_needed, and power of the wands that Ron's interested in, sorted in order of descending power. If more than one wand has same power, sort the result in order of descending age. 日他大爺, 題目中還有個要求沒說, 就是在結果中, 對於那些俱有相同 power 和相同 age 的行, 只選出 coins_needed 最小的一行來, 作為最終結果, 例如對於以下三行:

id	age	coins_needed	power
15	40	6018	7
17	40	8798	7
6	40	6773	7

只選

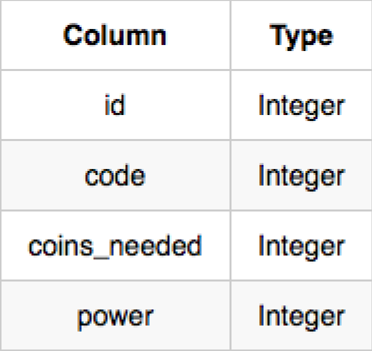
15	40	6018	7

這一行, 因為它 coins_needed 最小.

Input Format

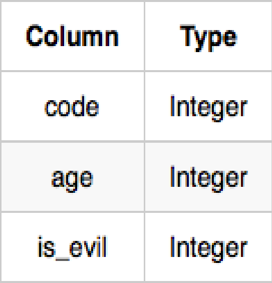
The following tables contain data on the wands in Ollivander's inventory:

Wands



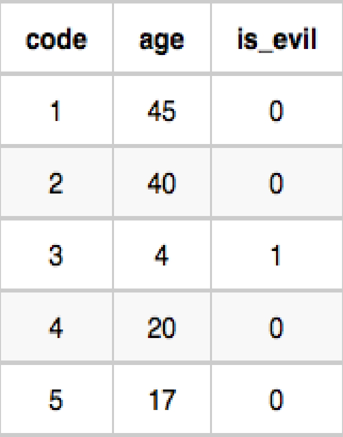
Column	Type
id	Integer
code	Integer
coins_needed	Integer
power	Integer

Wands_Property



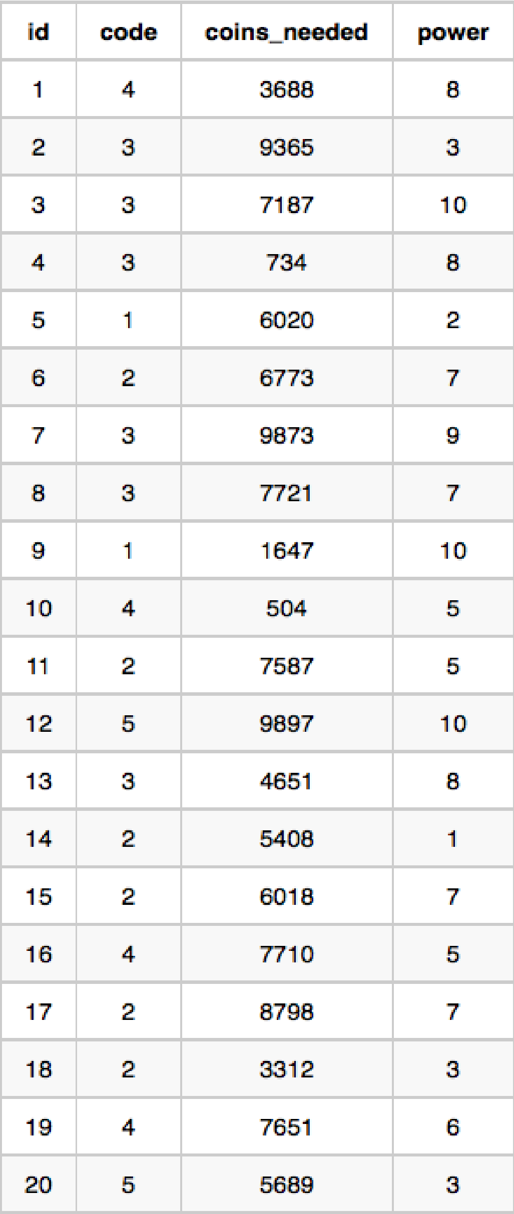
Column	Type
code	Integer
age	Integer
is_evil	Integer

Wands_Property input



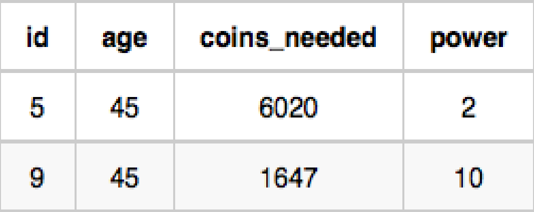
code	age	is_evil
1	45	0
2	40	0
3	4	1
4	20	0
5	17	0

Wands input



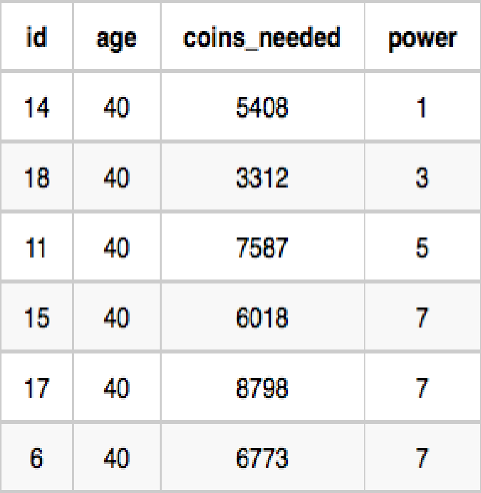
id	code	coins_needed	power
1	4	3688	8
2	3	9365	3
3	3	7187	10
4	3	734	8
5	1	6020	2
6	2	6773	7
7	3	9873	9
8	3	7721	7
9	1	1647	10
10	4	504	5
11	2	7587	5
12	5	9897	10
13	3	4651	8
14	2	5408	1
15	2	6018	7
16	4	7710	5
17	2	8798	7
18	2	3312	3
19	4	7651	6
20	5	5689	3

Data for wands of age 45 (code 1)



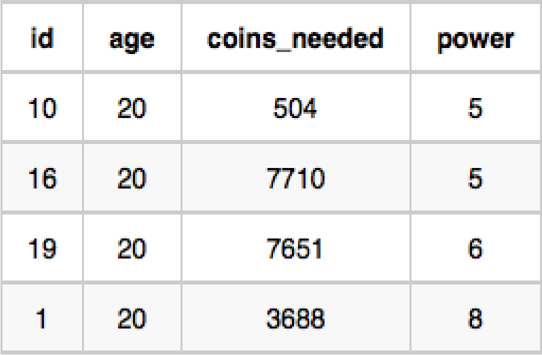
id	age	coins_needed	power
5	45	6020	2
9	45	1647	10

Data for wands of age 40 (code 2)



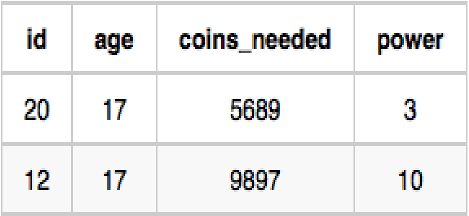
id	age	coins_needed	power
14	40	5408	1
18	40	3312	3
11	40	7587	5
15	40	6018	7
17	40	8798	7
6	40	6773	7

Data for wands of age 20 (code 4)



id	age	coins_needed	power
10	20	504	5
16	20	7710	5
19	20	7651	6
1	20	3688	8

Data for wands of age 17 (code 5)



id	age	coins_needed	power
20	17	5689	3
12	17	9897	10

Wands: The id is the id of the wand, code is the code of the wand, coins_needed is the total number of gold galleons needed to buy the wand, and power denotes the quality of the wand (the higher the power, the better the wand is).

Wands_Property: The code is the code of the wand, age is the age of the wand, and is_evil denotes whether the wand is good for the dark arts. If the value of is_evil is 0, it means that the wand is not evil. The mapping between code and age is one-one, meaning that if there are two pairs, (code_1, age_1) and (code_2, age_2), then code_1 != code_2 and age_1 != age_2.

Sample Input .

Wands Table: (見上). Wands_Property Table: (見上).

Sample Output

```
9 45 1647 10
12 17 9897 10
1 20 3688 8
15 40 6018 7
19 20 7651 6
11 40 7587 5
10 20 504 5
18 40 3312 3
20 17 5689 3
5 45 6020 2
14 40 5408 1
```

Explanation

The data for wands of age 45 (code 1): (見上).

The minimum number of galleons needed for wand(age = 45, power = 2) = 6020
The minimum number of galleons needed for wand(age = 45, power = 10) = 1647
The data for wands of age 40 (code 2): (見上).

The minimum number of galleons needed for wand(age = 40, power = 1) = 5408
The minimum number of galleons needed for wand(age = 40, power = 3) = 3312
The minimum number of galleons needed for wand(age = 40, power = 5) = 7587
The minimum number of galleons needed for wand(age = 40, power = 7) = 6018

The data for wands of age 20 (code 4): (見上).

The minimum number of galleons needed for wand(age = 20, power = 5) = 504
The minimum number of galleons needed for wand(age = 20, power = 6) = 7651
The minimum number of galleons needed for wand(age = 20, power = 8) = 3688

The data for wands of age 17 (code 5): (見上).

The minimum number of galleons needed for wand(age = 17, power = 3) = 5689
The minimum number of galleons needed for wand(age = 17, power = 10) = 9897

History:
E1 根本都不知道題目還有個要求沒說, 所以沒通過, 直接看的 Discussion.
E2 沒做出來.

Top Solution:

```
select w.id, p.age, w.coins_needed, w.power
from Wands as w join Wands_Property as p on (w.code = p.code)
where p.is_evil = 0 and w.coins_needed =
   (select min(coins_needed) #此 coins_needed 可以寫為 w1. coins_needed, 但不能寫為 w. coins_needed
    from Wands as w1 join Wands_Property as p1 on (w1.code = p1.code)
    where w1.power = w.power and p1.age = p.age)
order by w.power desc, p.age desc
```

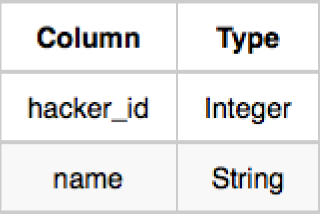
Challenges, Medium

Julia asked her students to create some coding challenges. Write a query to print the hacker_id, name, and the total number of challenges created by each student. Sort your results by the total number of challenges in descending order. If more than one student created the same number of challenges, then sort the result by hacker_id. If more than one student created the same number of challenges and the count is less than the maximum number of challenges created, then exclude those students from the result.

Input Format

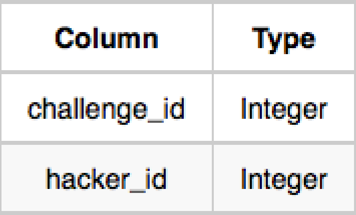
The following tables contain challenge data:

Hackers



Column	Type
hacker_id	Integer
name	String

Challenges



Column	Type
challenge_id	Integer
hacker_id	Integer

Hackers: The hacker_id is the id of the hacker, and name is the name of the hacker.

Challenges: The challenge_id is the id of the challenge, and hacker_id is the id of the student who created the challenge.

Sample Input 0

Hackers Table (刷屏, 且沒用, 故省).

Challenges Table (刷屏, 且沒用, 故省).

Sample Output 0

```
21283 Angela 6
88255 Patrick 5
96196 Lisa 1
```

Sample Input 1

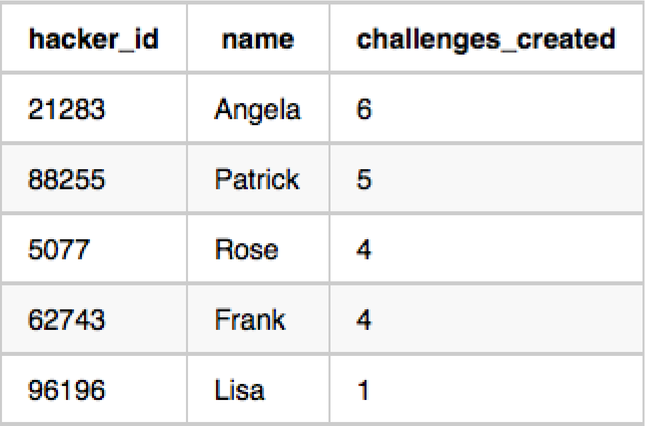
Hackers Table (刷屏, 且沒用, 故省).
Challenges Table (刷屏, 且沒用, 故省).

Sample Output 1

```
12299 Rose 6
34856 Angela 6
79345 Frank 4
80491 Patrick 3
81041 Lisa 1
```

Explanation

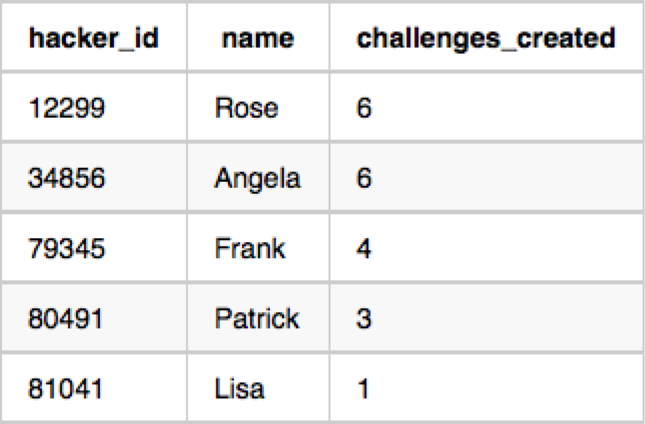
For Sample Case 0, we can get the following details:



hacker_id	name	challenges_created
21283	Angela	6
88255	Patrick	5
5077	Rose	4
62743	Frank	4
96196	Lisa	1

Students 5077 and 62743 both created 4 challenges, but the maximum number of challenges created is 6 so these students are excluded from the result.

For Sample Case 1, we can get the following details:



hacker_id	name	challenges_created
12299	Rose	6
34856	Angela	6
79345	Frank	4
80491	Patrick	3
81041	Lisa	1

Students 12299 and 34856 both created 6 challenges. Because 6 is the maximum number of challenges created, these students are included in the result.

History:
E1 沒做出來.
E2 沒做出來.

Top Solution (簡化後):

-- 以下代碼得票最多. 原代碼在 or 後那段中還要多一層 select, Discussion 中有人指出說沒必要, 我就將其刪除了. 另外 我還做了好幾處簡化. 以下代碼能通過. 盡管做了簡化, 以下仍是我目前見過的最複雜的 SQL 代碼.

```
select c.hacker_id, h.name, count(*) as total
from challenges c inner join hackers h
on h.hacker_id = c.hacker_id
group by c.hacker_id having

-- 以下即要求 total 等於 提出 challenge 最多的那個人 提出的 challenge 個數
total =
    (select count(*) as tot1
     from challenges c1
     group by c1.hacker_id
     ORDER BY tot1 desc limit 1)

or

-- 以下即要求 total 為 這樣一種 challenge 個數: 即只有一個人提出了 這麼多個數的 challenge
-- 以下藍色部分即為一個表, 此表名字叫 t.
total in
    (select t.tot2
     from (select count(*) as tot2, c2.hacker_id
           from challenges c2
           group by c2.hacker_id) as t
     group by t.tot2
     having count(t.hacker_id) = 1)

order by total desc, h.hacker_id;
```

Contest Leaderboard, Medium

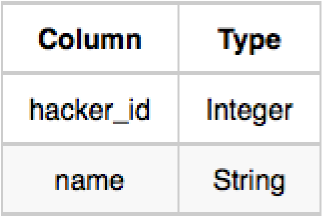
You did such a great job helping Julia with her last coding contest challenge that she wants you to work on this one, too!

The total score of a hacker is the sum of their maximum scores for all of the challenges. Write a query to print the hacker_id, name, and total score of the hackers ordered by the descending score. If more than one hacker achieved the same total score, then sort the result by ascending hacker_id. Exclude all hackers with a total score of 0 from your result. 注意看後面對 Sample Output 的解釋.

Input Format

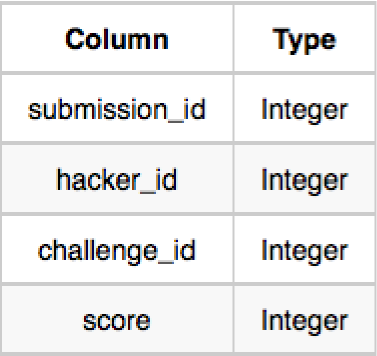
The following tables contain contest data:

Hackers



Column	Type
hacker_id	Integer
name	String

Submissions



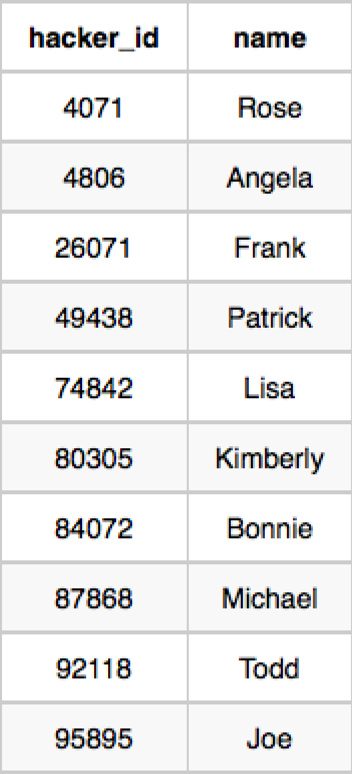
Column	Type
submission_id	Integer
hacker_id	Integer
challenge_id	Integer
score	Integer

Hackers: The hacker_id is the id of the hacker, and name is the name of the hacker.

Submissions: The submission_id is the id of the submission, hacker_id is the id of the hacker who made the submission, challenge_id is the id of the challenge for which the submission belongs to, and score is the score of the submission.

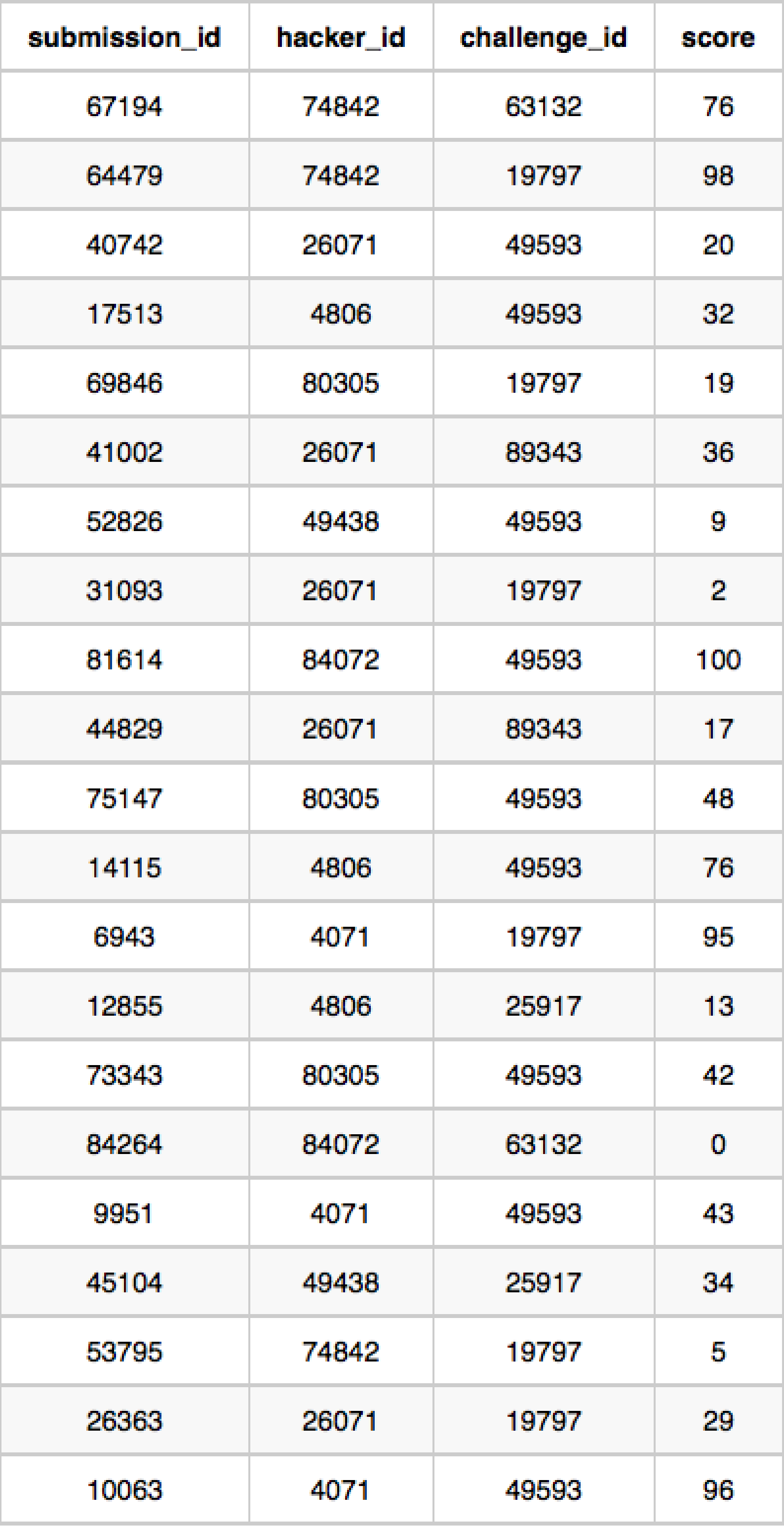
Sample Input

Hackers
Submissions Table:



hacker_id	name
4071	Rose
4806	Angela
26071	Frank
49438	Patrick
74842	Lisa
80305	Kimberly
84072	Bonnie
87868	Michael
92118	Todd
95895	Joe

Table:



submission_id	hacker_id	challenge_id	score
67194	74842	63132	76
64479	74842	19797	98
40742	26071	49593	20
17513	4806	49593	32
69846	80305	19797	19
41002	26071	89343	36
52826	49438	49593	9
31093	26071	19797	2
81614	84072	49593	100
44829	26071	89343	17
75147	80305	49593	48
14115	4806	49593	76
6943	4071	19797	95
12855	4806	25917	13
73343	80305	49593	42
84264	84072	63132	0
9951	4071	49593	43
45104	49438	25917	34
53795	74842	19797	5
26363	26071	19797	29
10063	4071	49593	96

Sample Output

```
4071 Rose 191
74842 Lisa 174
84072 Bonnie 100
4806 Angela 89
26071 Frank 85
80305 Kimberly 67
49438 Patrick 43
```

Explanation

Hacker 4071 submitted solutions for challenges 19797 and 49593, so the total score = 95 + max(43, 96) = 191.

Hacker 74842 submitted solutions for challenges 19797 and 63132, so the total score = max(98, 5) + 76 = 174.

Hacker 84072 submitted solutions for challenges 49593 and 63132, so the total score = 100 + 0 = 100.

The total scors for hackers 4806, 26071, 80305, and 49438 can be similarly calculated.

History:
E1 晚上兩三點做的, 剛做了上題那複雜的, 現在太困了, 就直接看的 Discussion.
E2 直接看的答案.

Top Solution:

```
SELECT h.hacker_id, h.name, sum(scr) as tot
# 以下即 h join chart 1
# 以下藍色部分即為一個表, 此表名字叫 chart1.
FROM Hackers h,
    (SELECT s2.hacker_id as hac_id, s2.challenge_id as ch_id, max(s2.score) as scr
     FROM Submissions s2
     GROUP BY s2.hacker_id, s2.challenge_id
     having scr > 0
    ) AS chart1
WHERE h.hacker_id = hac_id
GROUP BY h.hacker_id #這個 GROUP BY 是為了前面第一行求 sum(src)
ORDER BY tot DESC, h.hacker_id
```

由以上代碼可總結出:
若是

```
SELECT max(...)
FROM ...
```

GROUP BY A, B
則每個 group 是 A 和 B 一起弄成的小 group.
SELECT 中的 max 就是在每個這樣的小 group 中求的.

Advanced Join (沒做)

Alternative Queries (沒做)