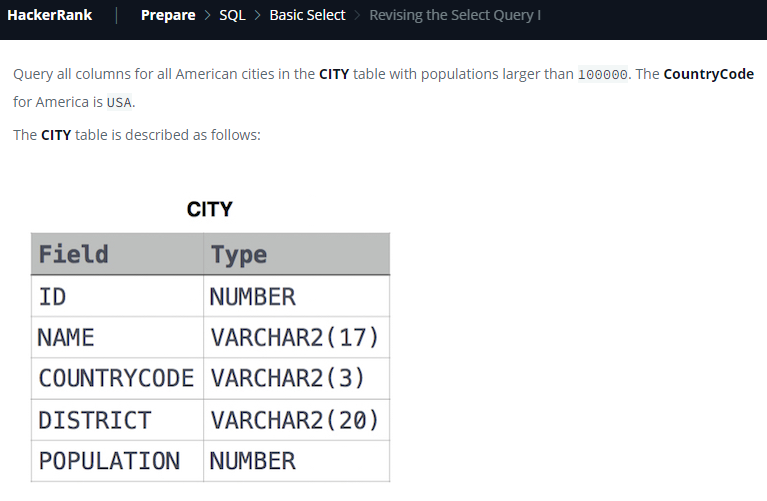
HackerRank – Practice – SQL

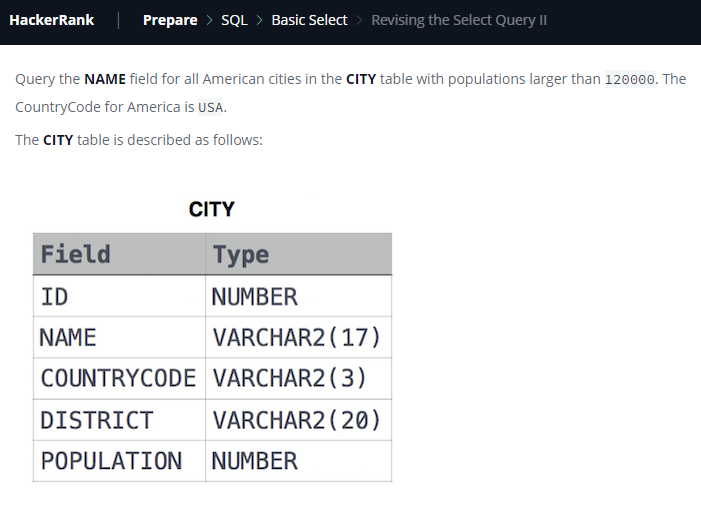
*Author: Preethi Ann Jacob*

*Default Language: MySQL. Otherwise MS SQL server*



select \* from City

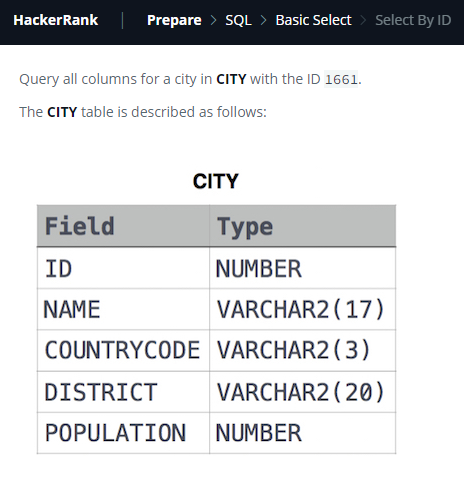
where population>100000 and countrycode='USA'



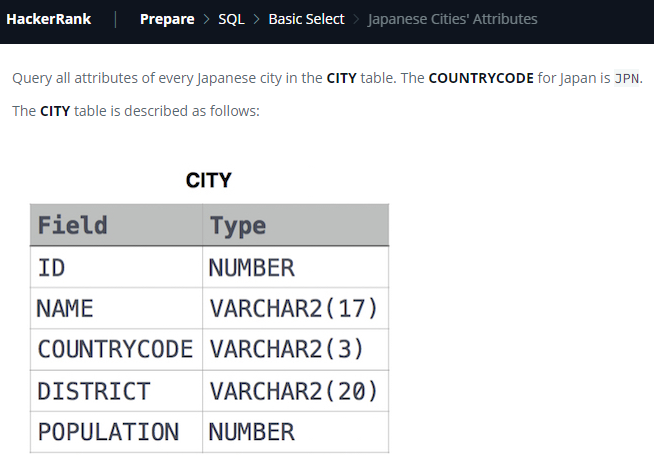
select name from City where population>120000 and CountryCode='USA';



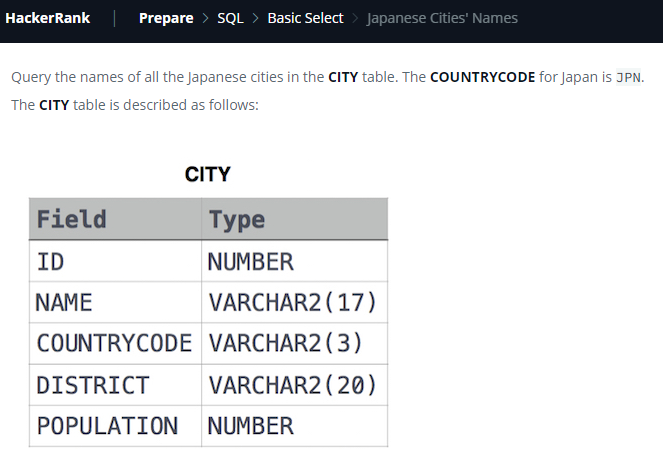
select \* from city



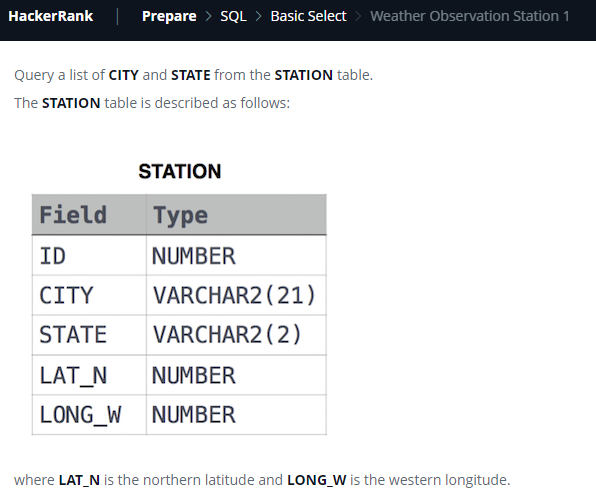
select \* from city where id=1661



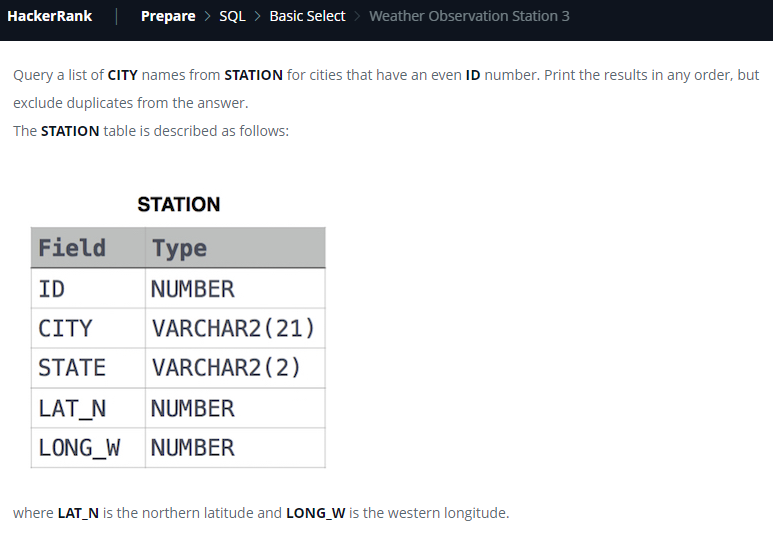
Select \* from City where countrycode='JPN'



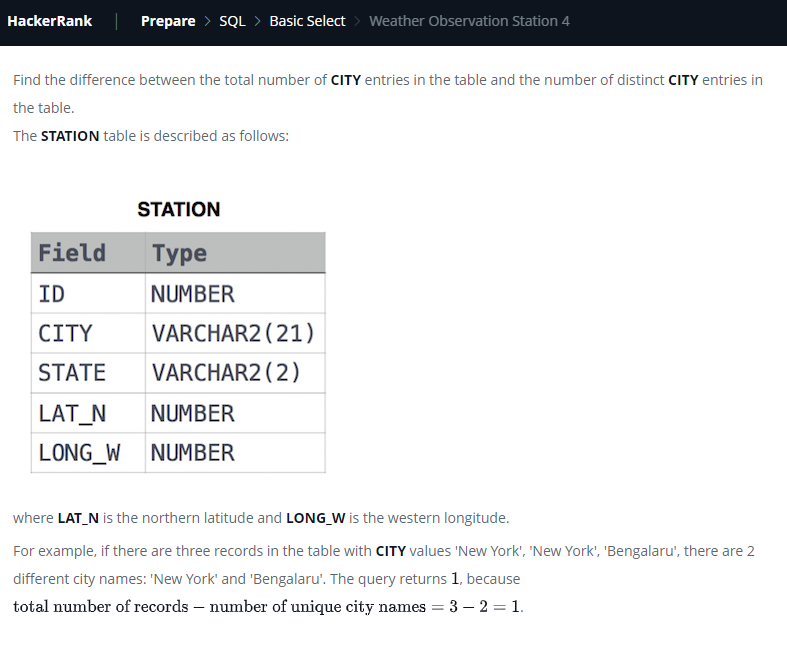
Select name from city where countrycode='JPN'



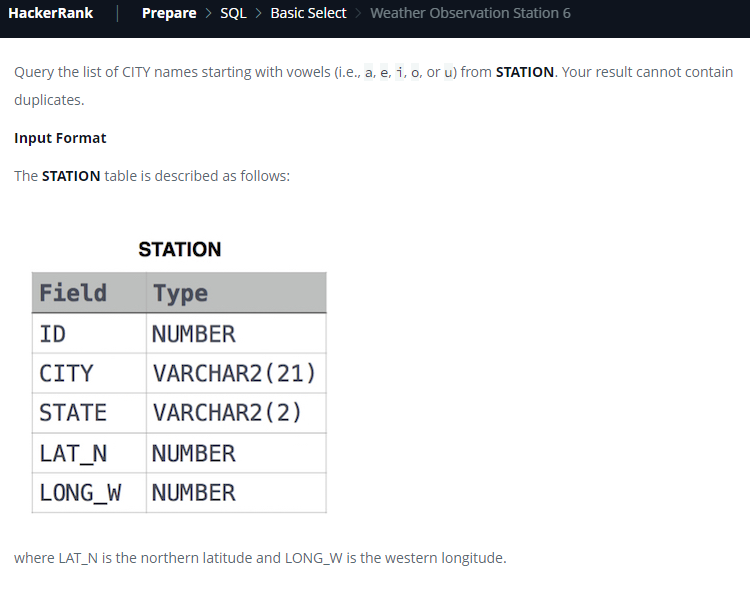
select city,state from station



select distinct city from station where id%2=0



select count(city)-count(distinct city) from station

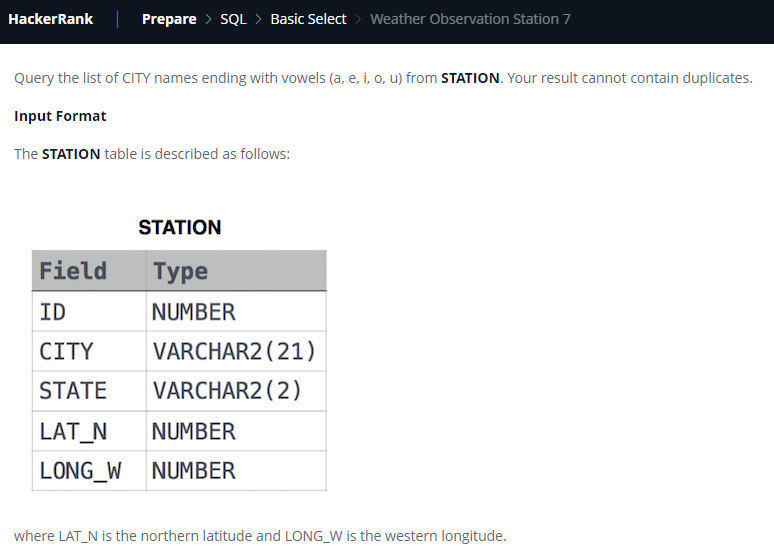


select distinct city from station where left(city,1) in ('a','e','i','o','u');

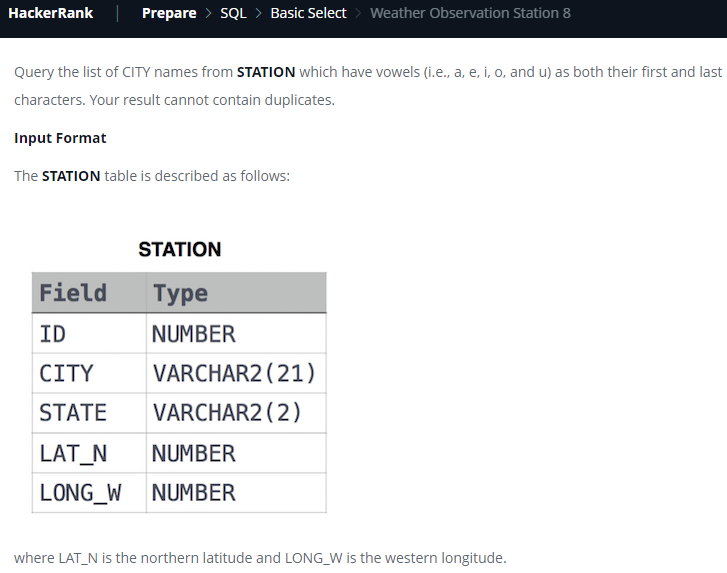
select distinct city from station where substring(city,1,1) in ('a','e','i','o','u');

select distinct city from station where city regexp '^[aeiouAEIOU]';

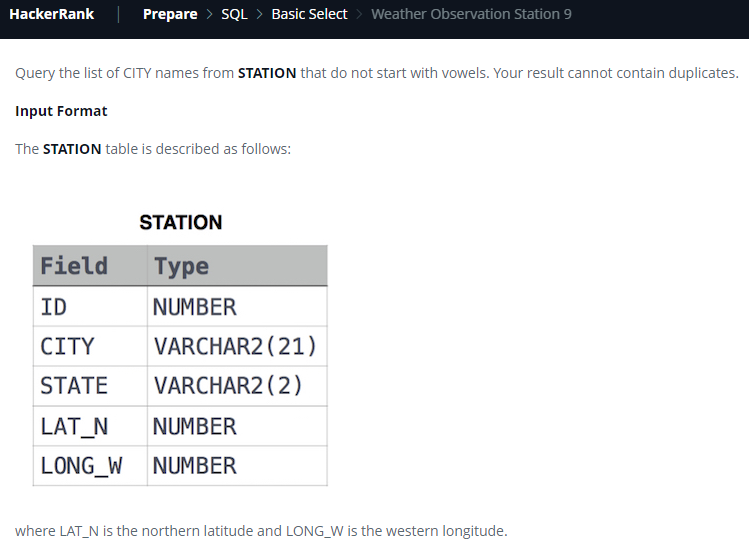
select distinct city from station where city like 'a%' or city like 'e%' or city like 'i%' or city like 'o%' or city like 'u%';



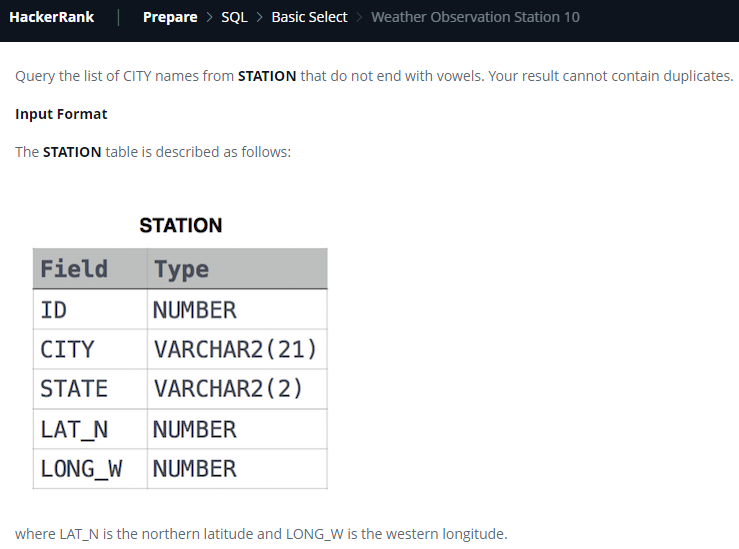
select distinct city from station where city regexp '[aeiouAEIOU]$'



select city from station where city regexp '^[aeiouAEIOU].\*[aeiouAEIOU]$'



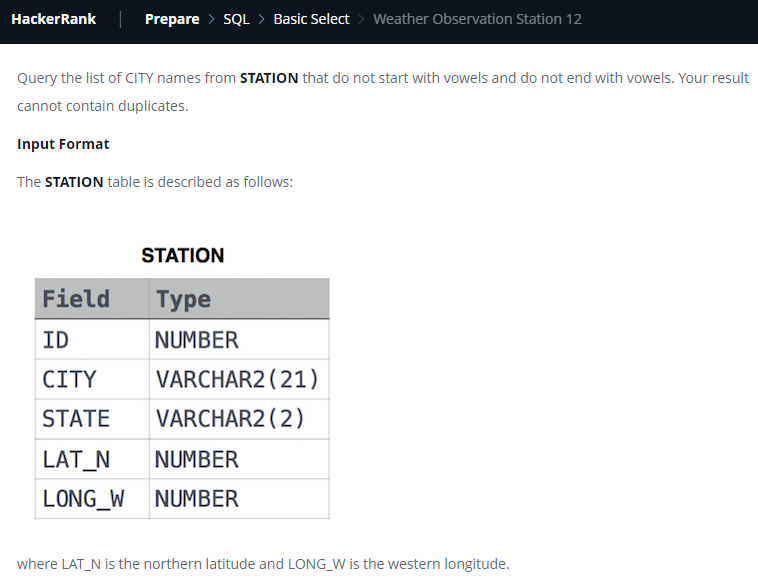
select distinct city from station where city regexp '^[^aeiouAEIOU]';



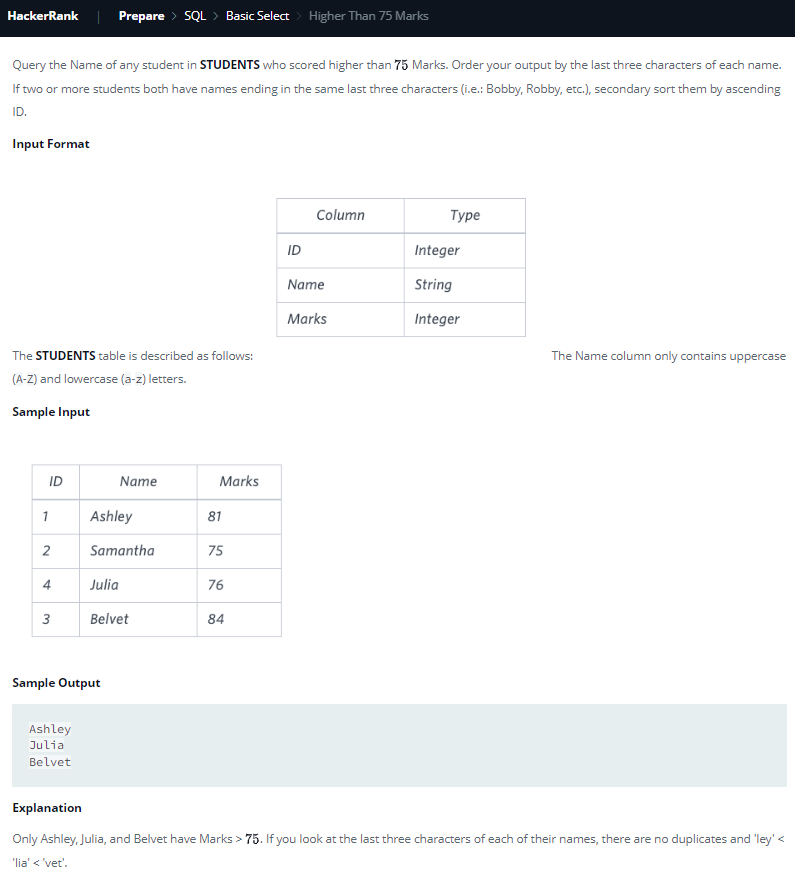
select distinct city from station where city regexp '[^aeiouAEIOU]$';



select distinct city from station where city regexp '^[^aeiouAEIOU]|[^aeiouAEIOU]$'

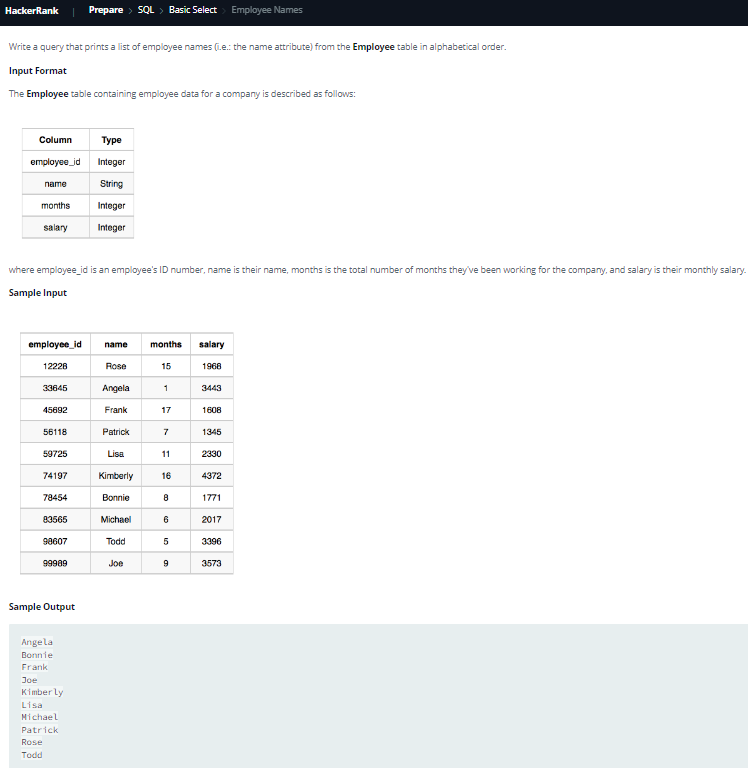


select distinct city from station where city regexp '^[^aeiouAEIOU].\*[^aeiouAEIOU]$'

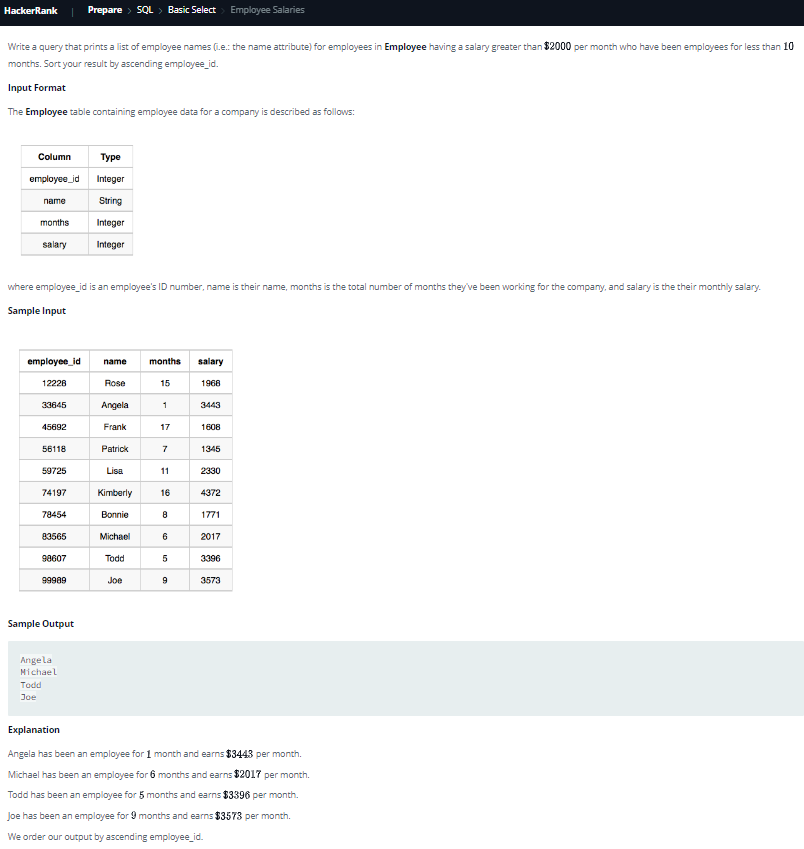


select name from students where marks>75

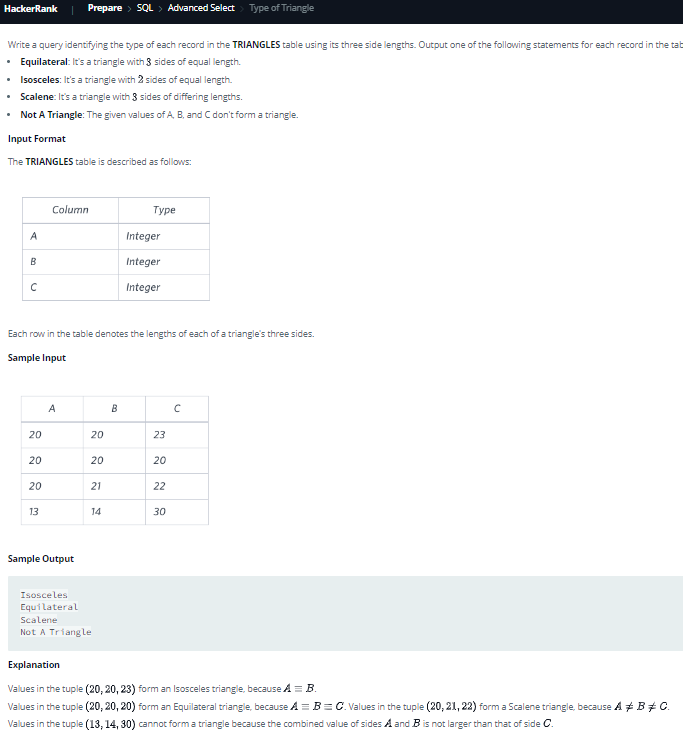
order by substring(name,-3,3), id;



select name from employee order by name;



select name from Employee where salary>2000 and months<10 order by employee\_id;



select

case

when A+B<=C or A+C<=B or B+C<=A then 'Not A Triangle'

when A=B and B=C then 'Equilateral'

when A=B or B=C or A=C then 'Isosceles'

else 'Scalene'

end

from triangles;

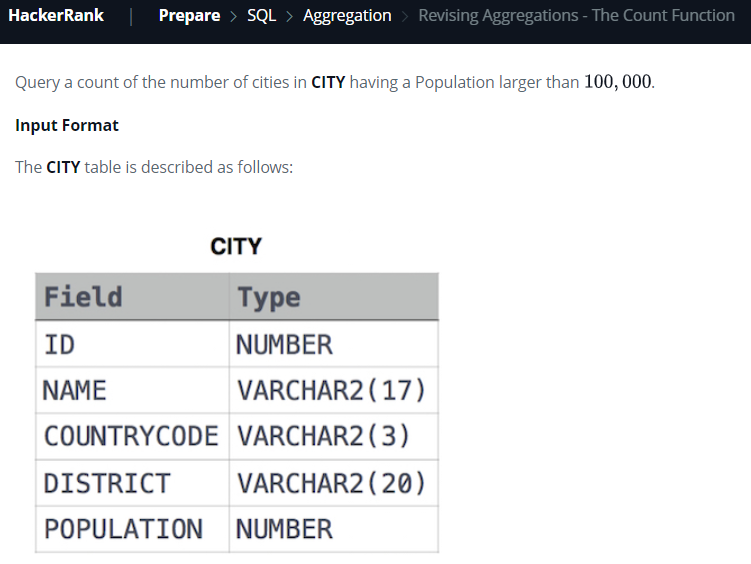
[Reason: Approach: A triangle is valid if sum of its two sides is greater than the third side. If three sides are a, b and c, then three conditions should be met. 

1.a + b > c

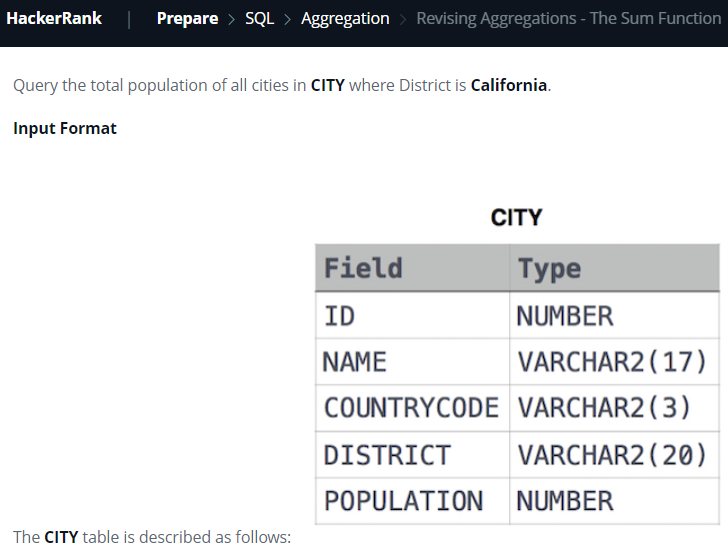
2.a + c > b

3.b + c > a

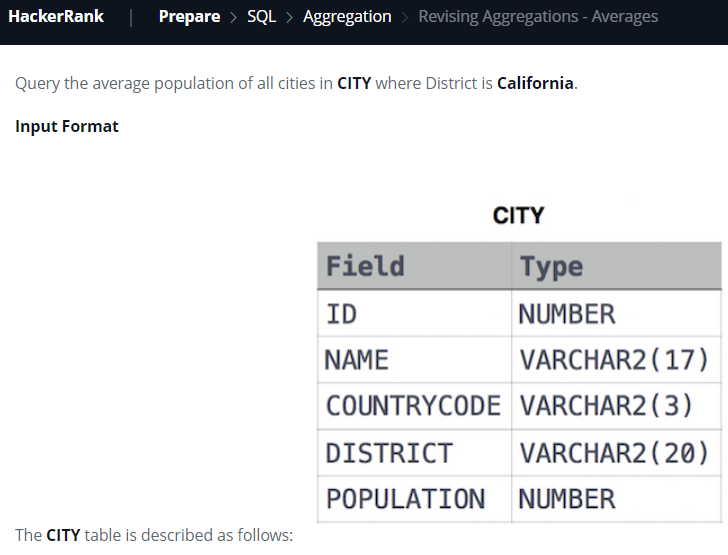
]



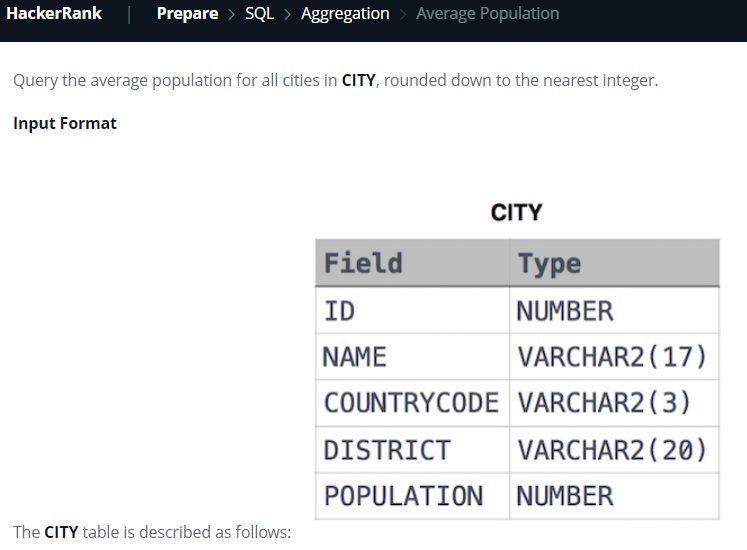
select count(\*) from city where population>100000;



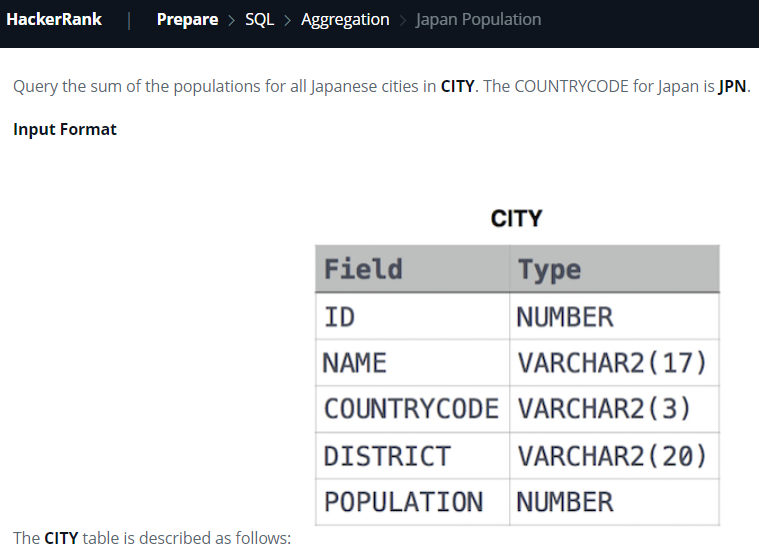
select sum(population) from city where district ='California';



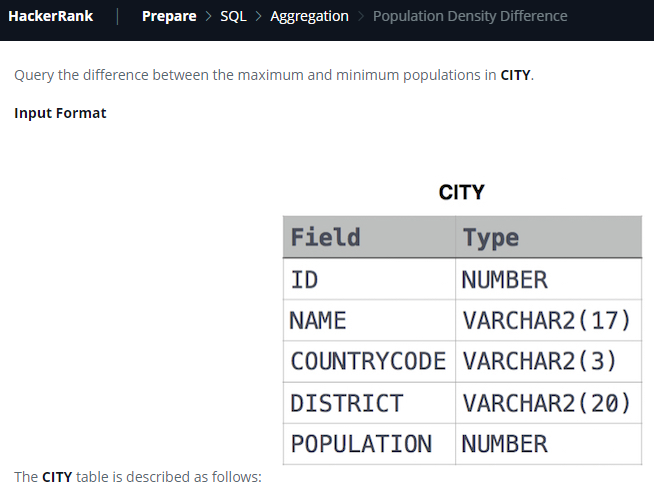
select avg(population) from city where district = 'California';



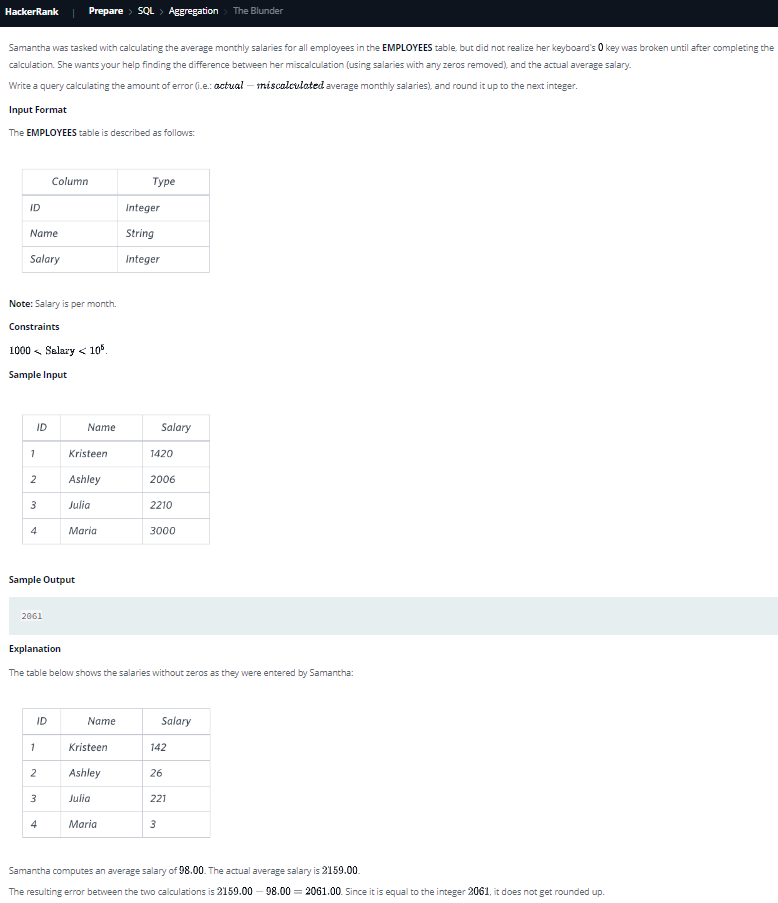
select round(avg(population),0) from city;



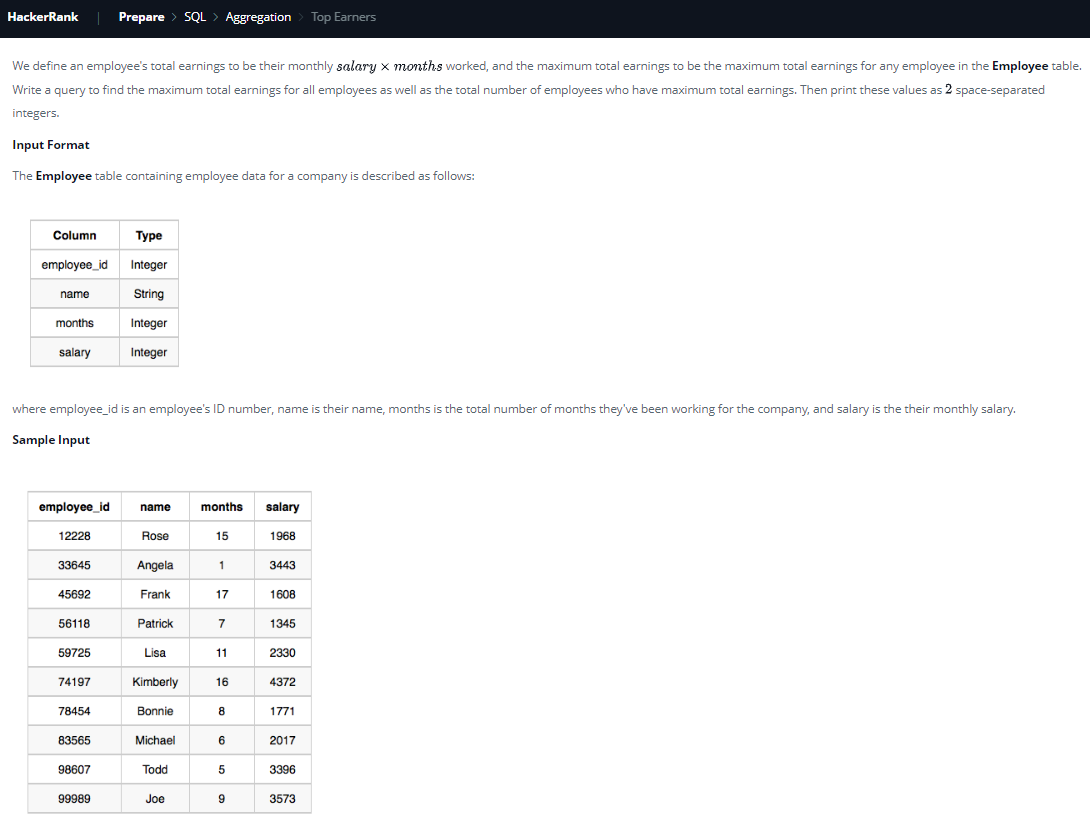
select sum(population) from city where countrycode='JPN';

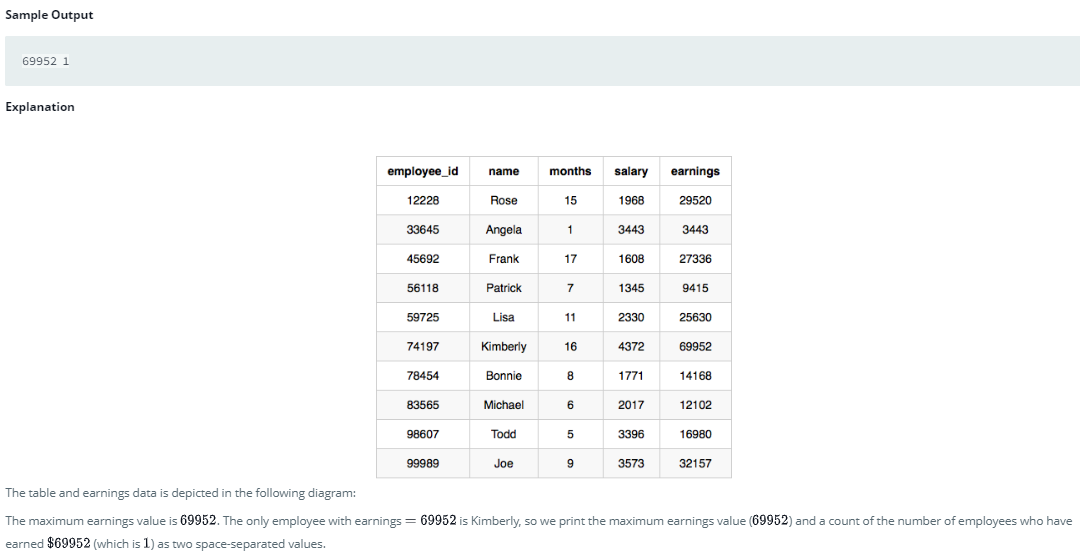


select max(population)-min(population) from city;



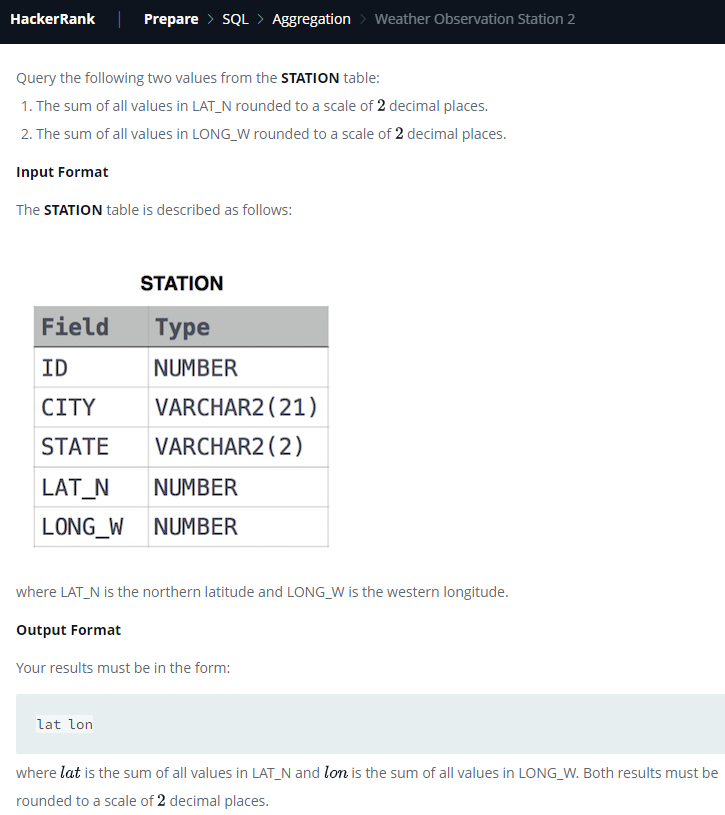
select ceil(avg(salary)-avg(replace(salary,'0',''))) from employees;



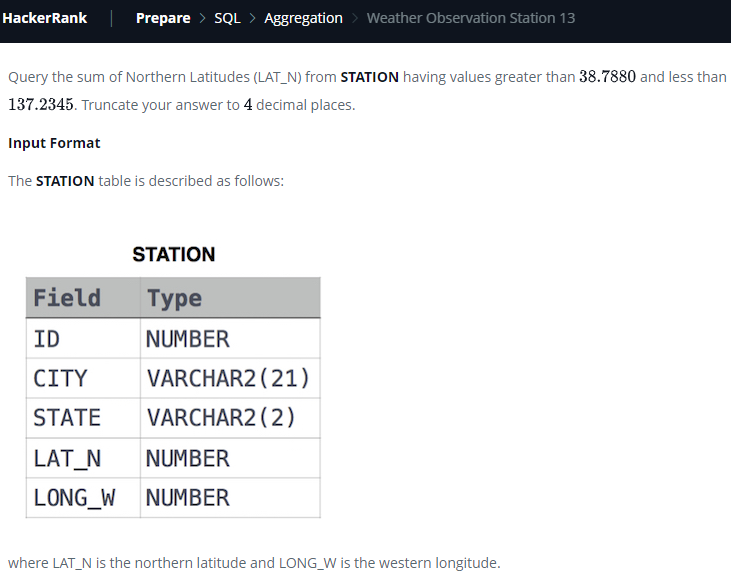


select max(salary\*months),count(\*) from employee

where salary\*months = (select max(salary\*months) from employee);

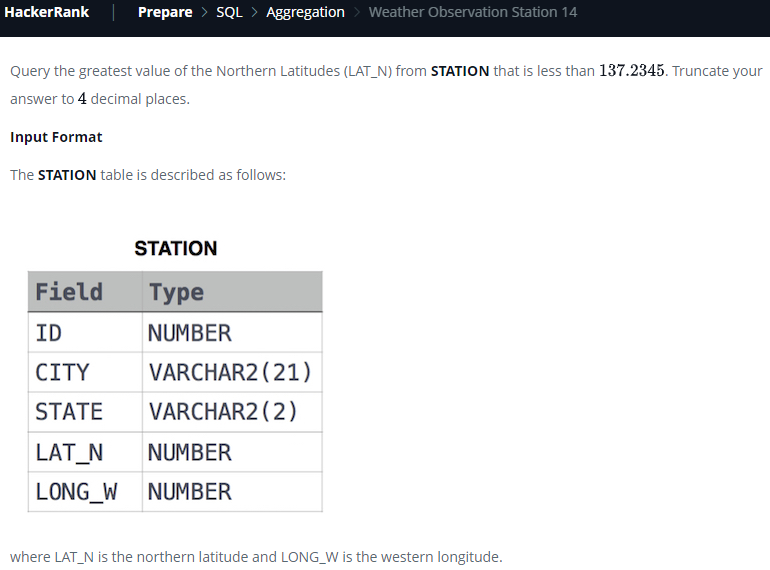


select round(sum(lat\_n),2),round(sum(long\_w),2) from station;

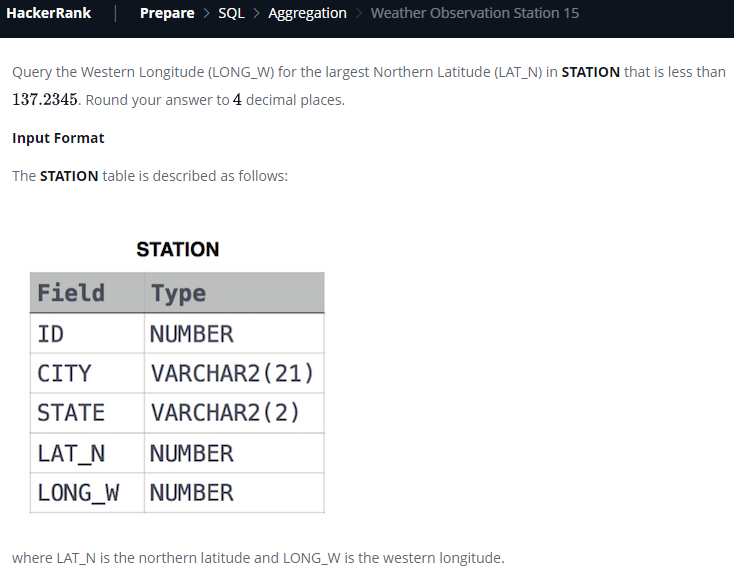


select round(sum(lat\_n),4) from station

where lat\_n between 38.7880 and 137.2345;

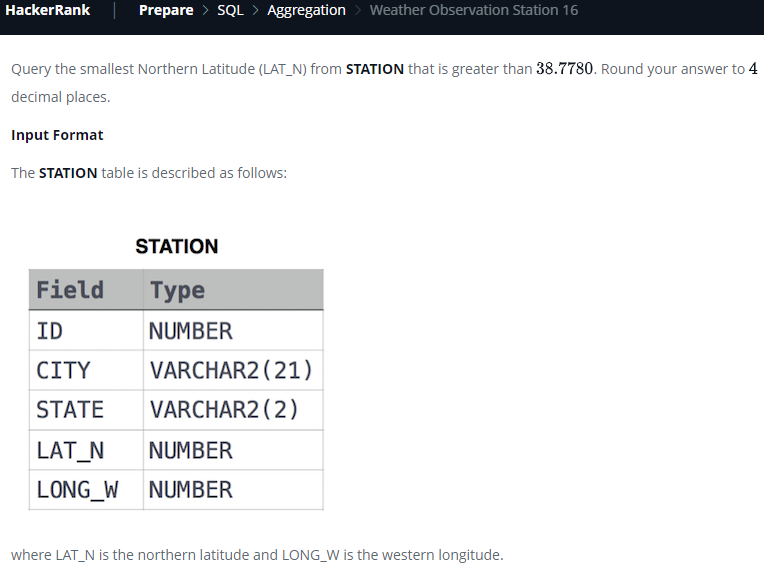


select round(max(lat\_n),4) from station where lat\_n<137.2345

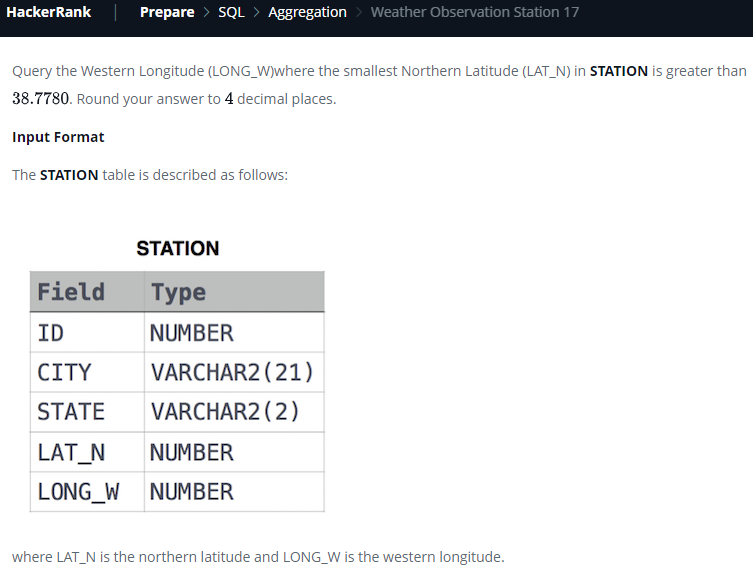


select round(long\_w,4) from station

where lat\_n = (select max(lat\_n) from station where lat\_n<137.2345)

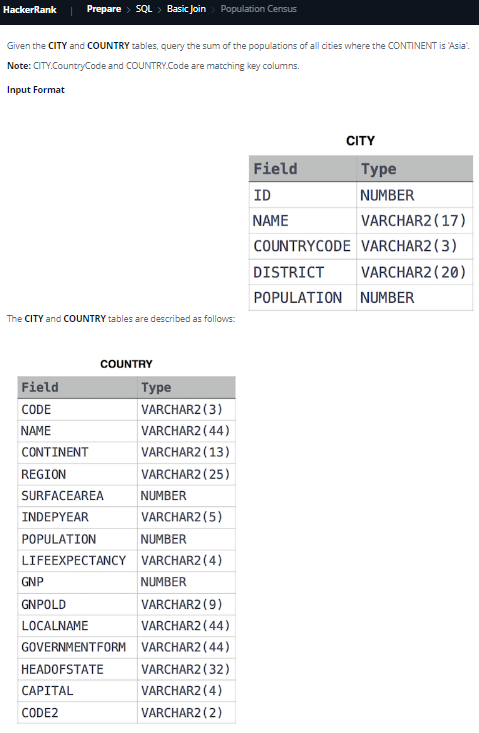


select round(min(lat\_n),4) from station where lat\_n>38.7780



select round(long\_w,4) from station

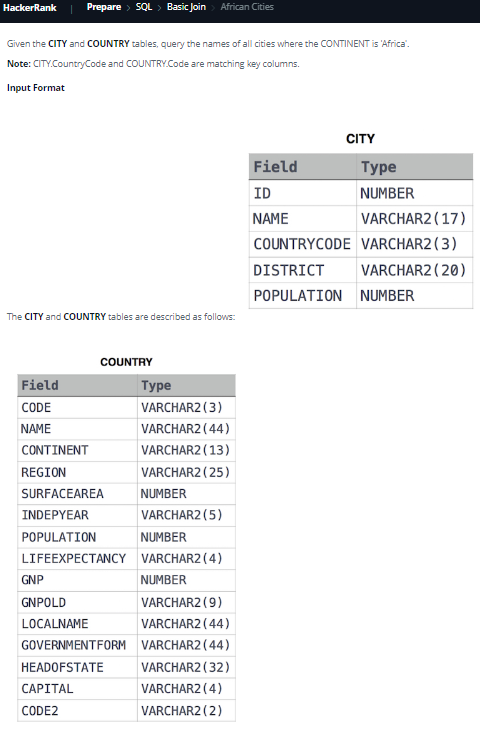
where lat\_n = (select min(lat\_n) from station where lat\_n>38.7780)



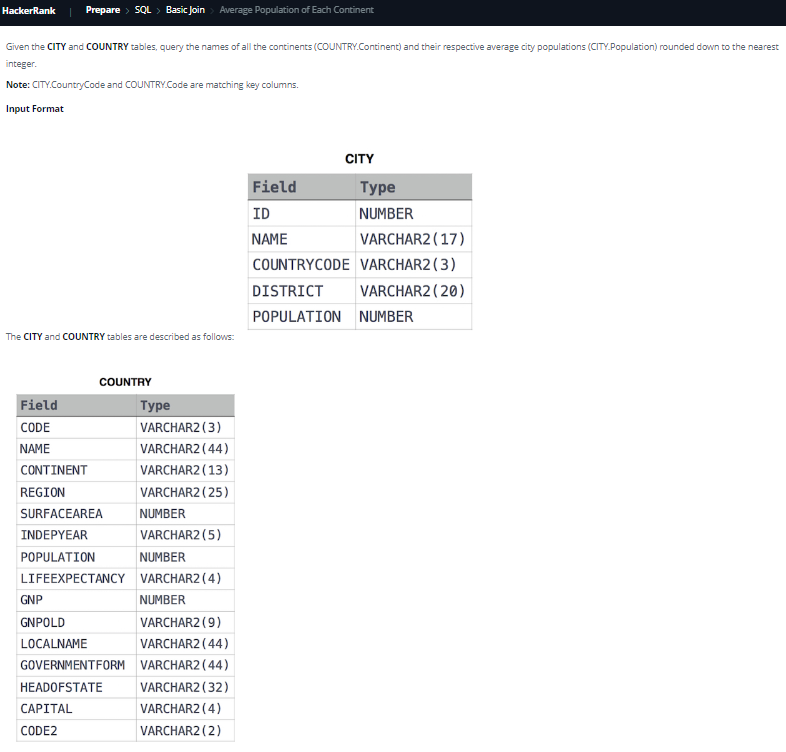
select sum(A.population) from city A

inner join country B on A.countrycode=B.code

where B.continent = 'Asia'



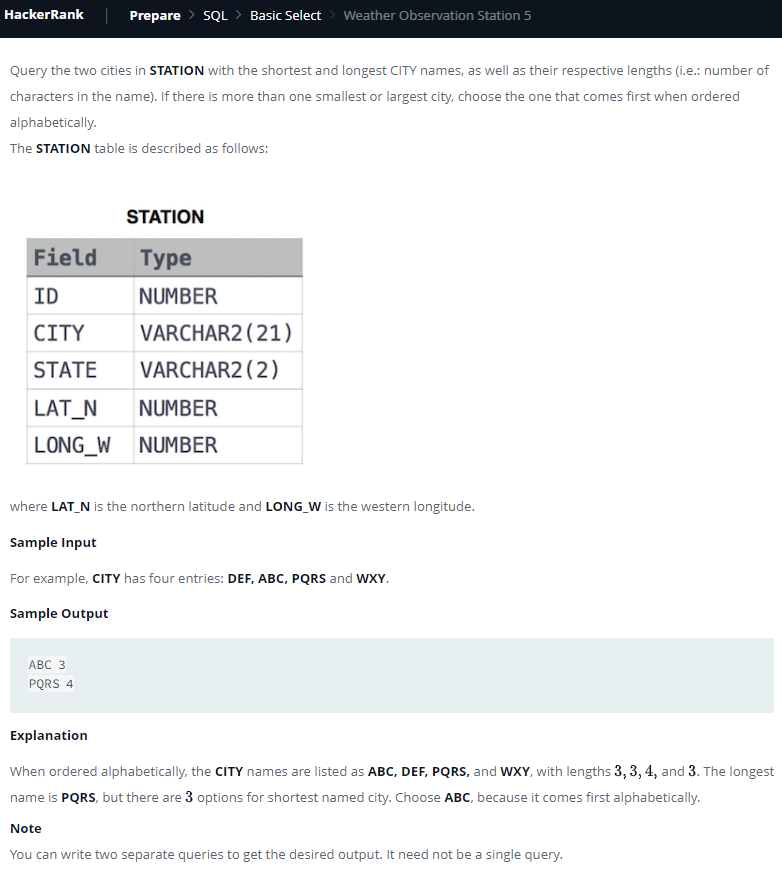
select A.name from City A inner join Country B on A.countrycode=B.code where B.continent='Africa'



select B.continent, floor(avg(A.population))

from Country B inner join City A on A.countrycode=B.code

group by B.continent



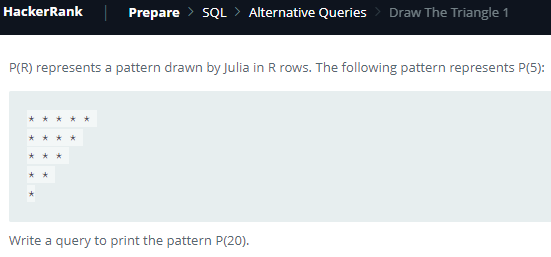
select city,length(city) from station order by length(city),city limit 1;

select city,length(city) from station order by length(city)desc,city limit 1;

OR Another way:

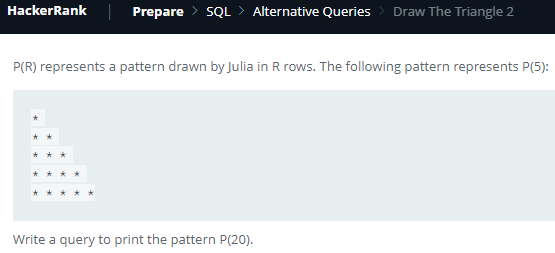
(select city,length(city) from station order by length(city),city limit 1) union

(select city,length(city) from station order by length(city)desc,city limit 1)

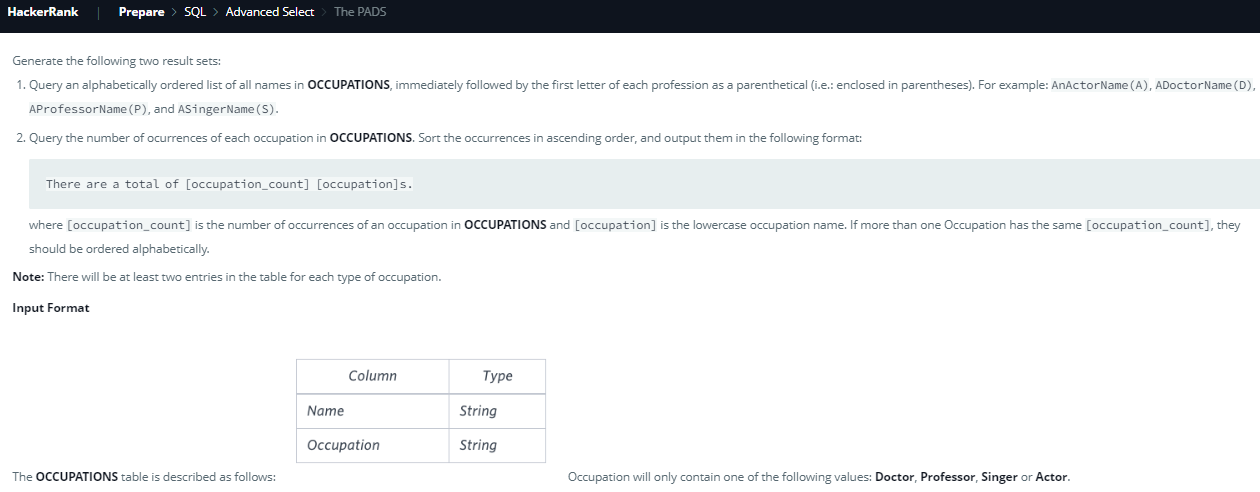


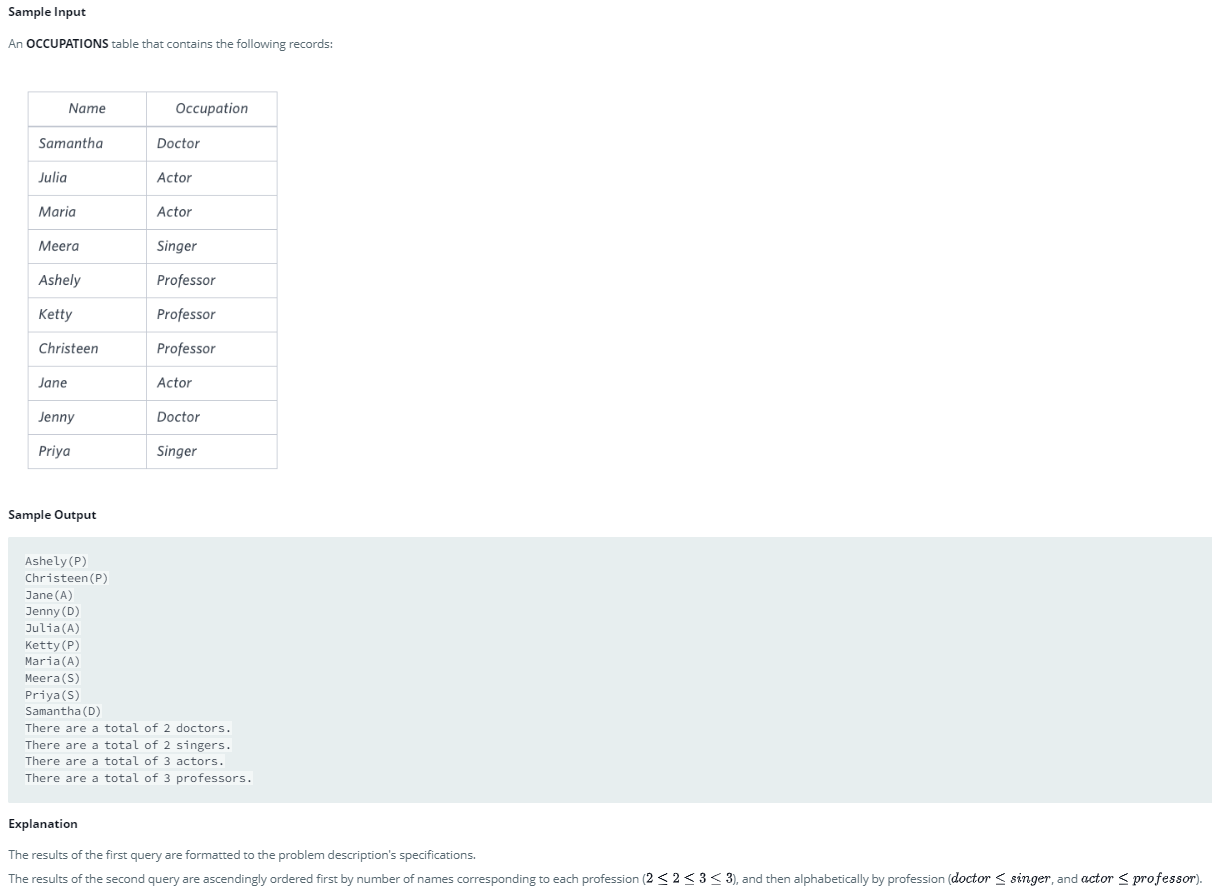
set @num=21;

select repeat('\* ',@num:=@num-1) from information\_schema.tables limit 20;



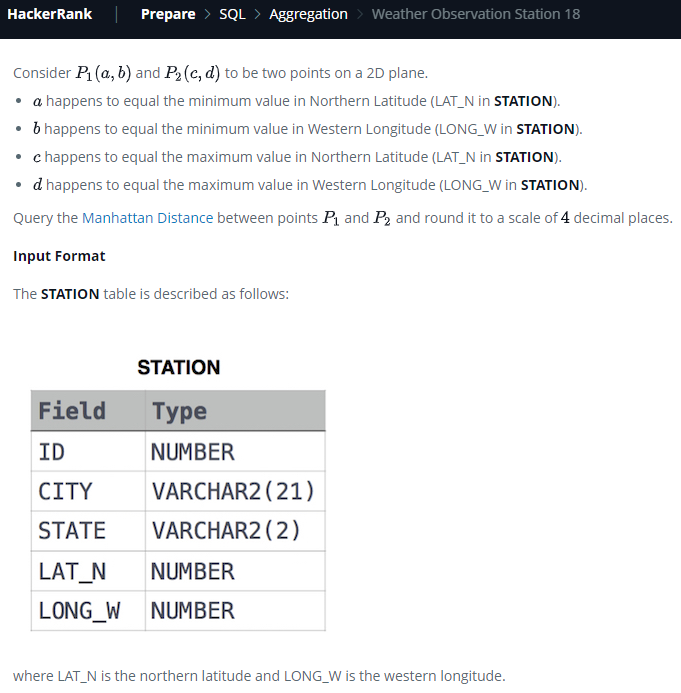
set @num=0;

select repeat('\* ',@num:=@num+1) from information\_schema.tables limit 20; 

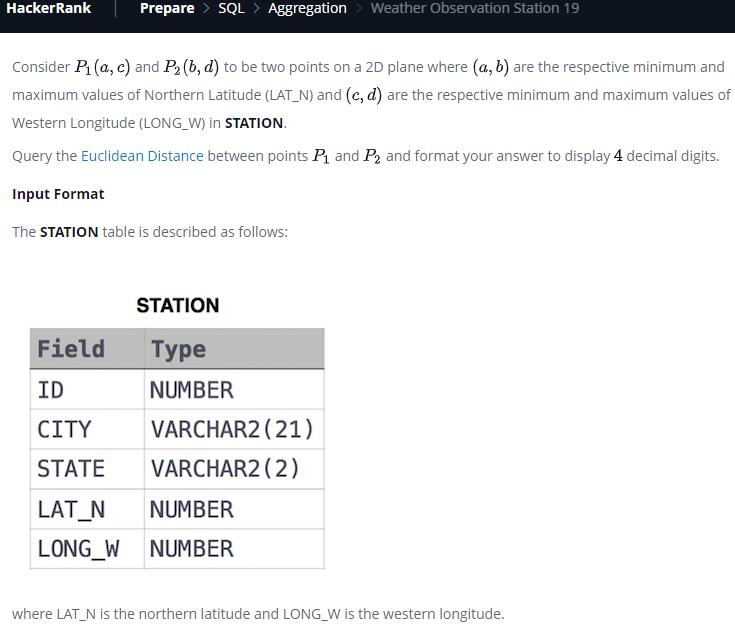


select concat(name, '(', left(occupation,1) ,')') from occupations order by name;

select concat('There are a total of ', count(\*),' ', lower(occupation), 's.') from occupations group by occupation order by count(\*),occupation;

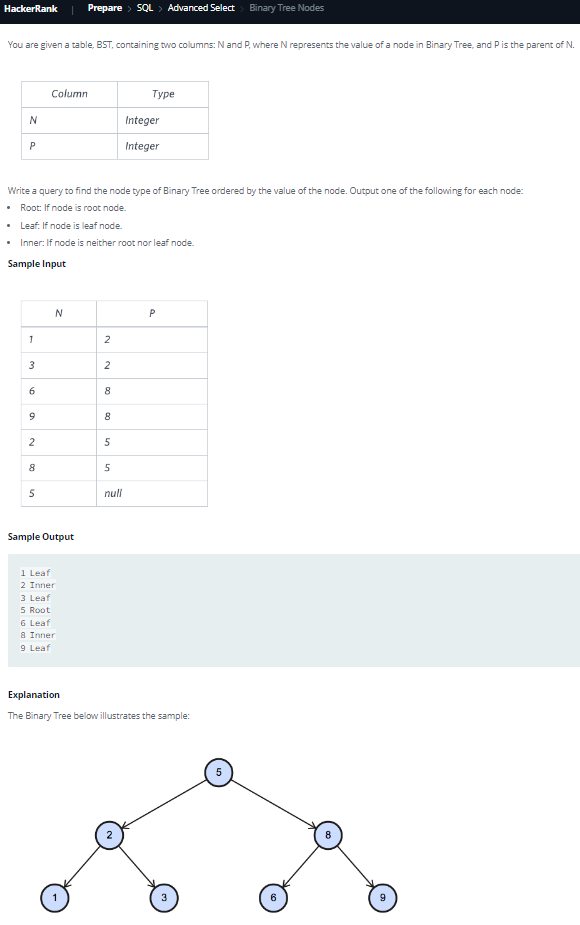


select round(abs(a-c)+abs(b-d),4) from (select min(lat\_n) a, min(long\_w) b, max(lat\_n) c, max(long\_w) d from station) as T;



select round( sqrt( power(b-a,2) + pow(d-c,2) ) , 4)

from (select min(lat\_n) a, max(lat\_n) b, min(long\_w) c, max(long\_w) d from station) as T;



select n, case

when p is null then 'Root'

when n in (select p from bst) then 'Inner'

else 'Leaf'

end

from bst order by n;

OR Another way:

select n, case

when p is null then 'Root'

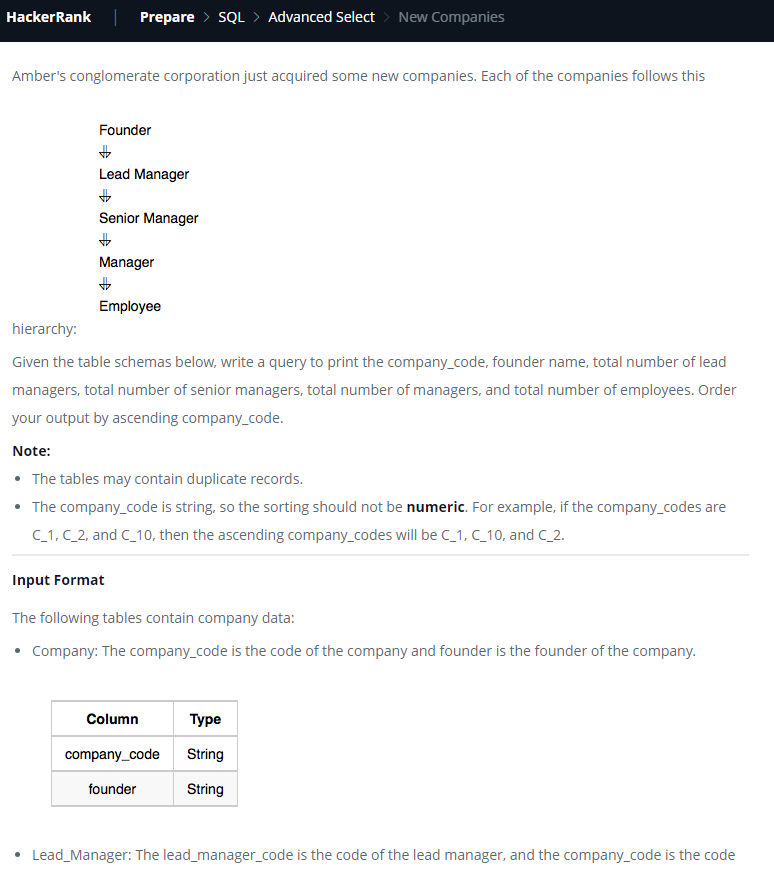
when n not in (select p from bst where p is not null) then 'Leaf'

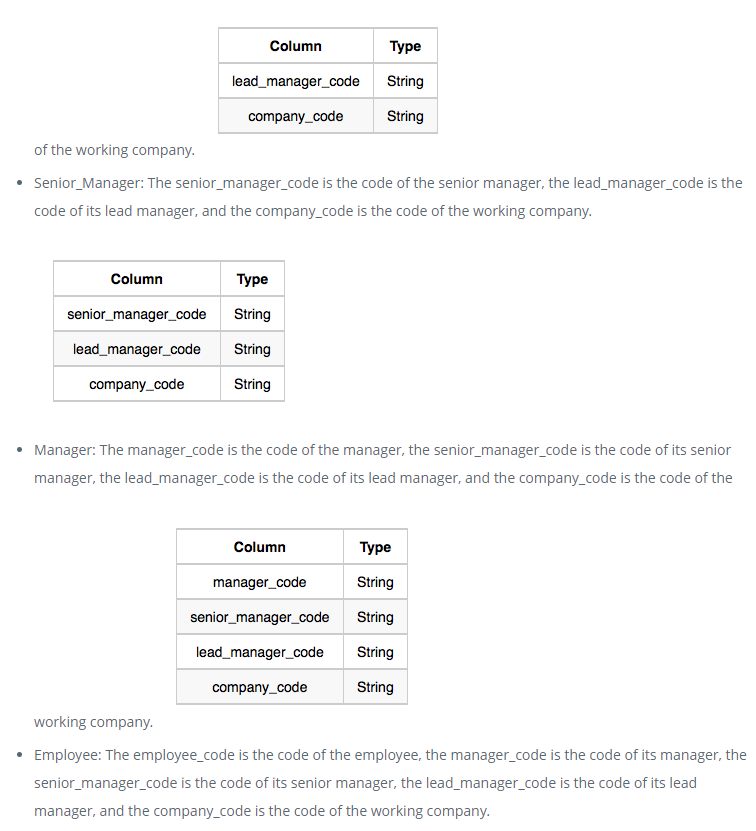
--Remember to use ‘p is not null’ with ‘not in’

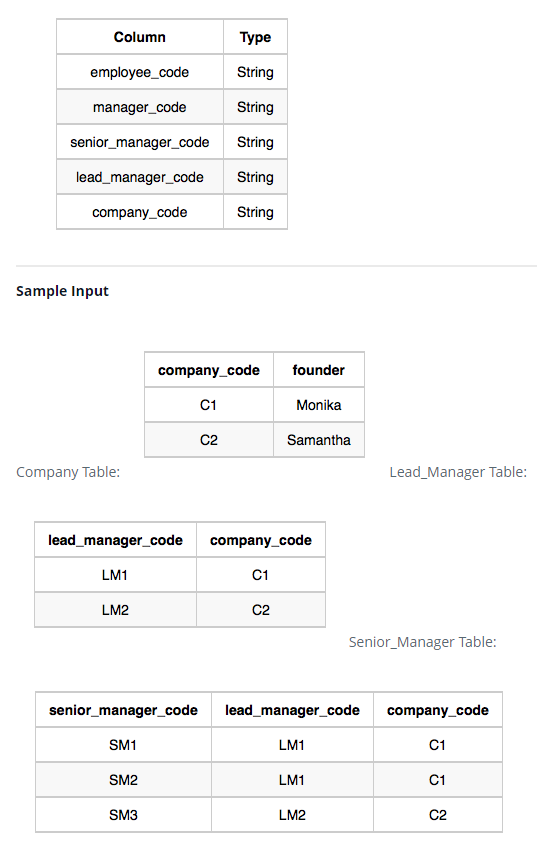
else 'Inner'

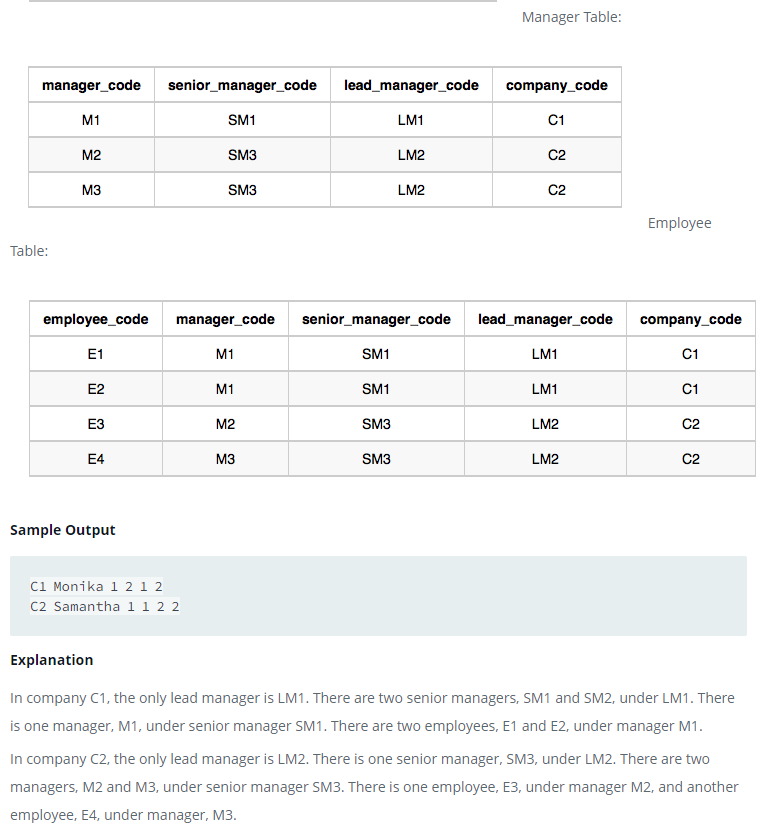
end

from bst order by n;









select C.company\_code, C.founder, count(distinct L.lead\_manager\_code), count(distinct S.senior\_manager\_code), count(distinct M.manager\_code), count(distinct E.employee\_code)

from company C

inner join lead\_manager L on C.company\_code=L.company\_code

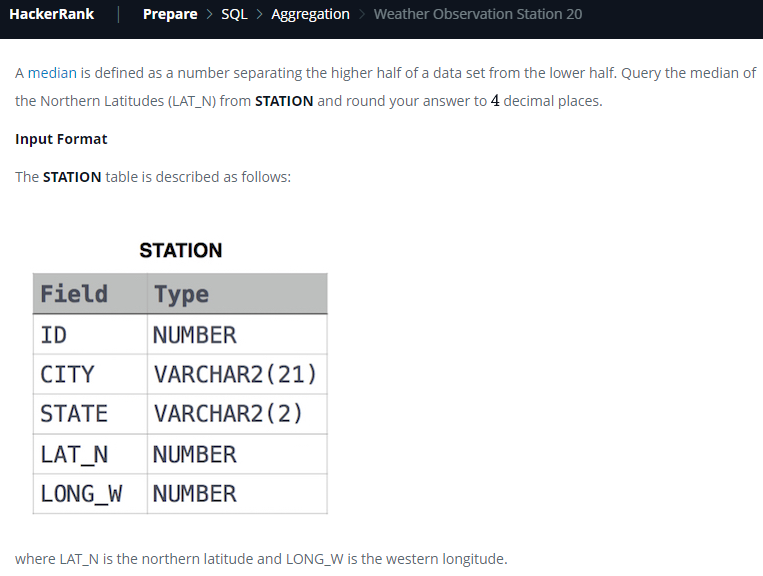
inner join senior\_manager S on S.company\_code=C.company\_code and S.lead\_manager\_code=L.lead\_manager\_code

inner join manager M on M.company\_code=C.company\_code and M.lead\_manager\_code=L.lead\_manager\_code and M.senior\_manager\_code=S.senior\_manager\_code

inner join employee E on E.company\_code=C.company\_code and E.lead\_manager\_code=L.lead\_manager\_code and E.senior\_manager\_code=S.senior\_manager\_code and E.manager\_code=M.manager\_code

group by C.company\_code, C.founder

order by company\_code;

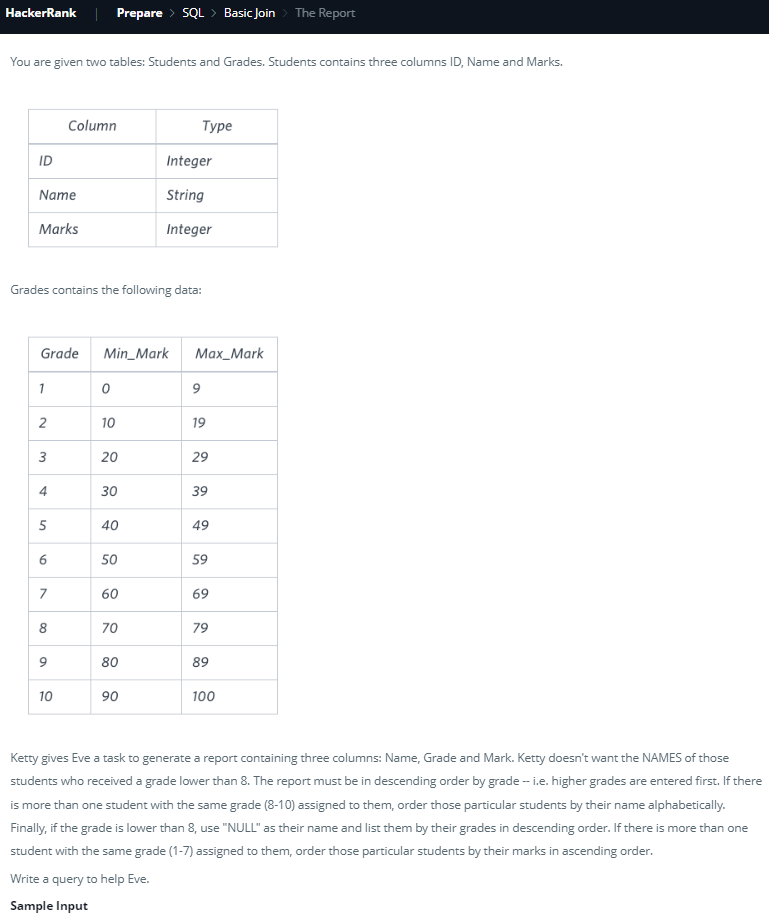


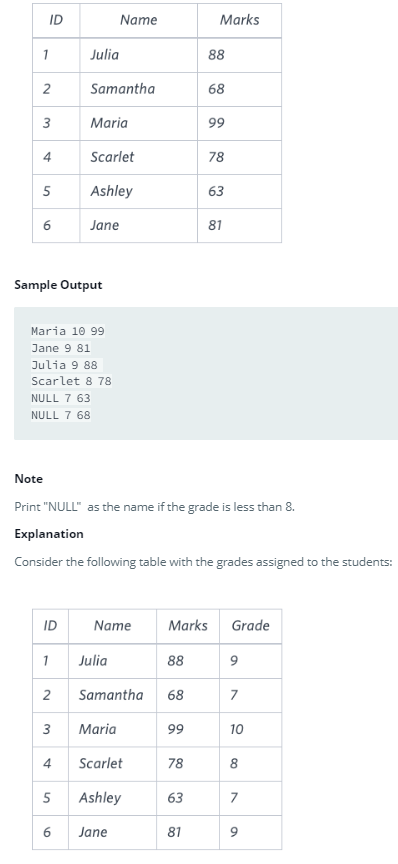
set @rowNo=-1;

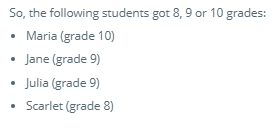
select round(avg(lat\_n),4) from

(select @rowNo:=@rowNo+1 as rowIndex, lat\_n from station order by lat\_n) T

where T.rowIndex in (floor(@rowNo/2),ceil(@rowNo/2));







select

case

when G.grade>=8 then S.name

else NULL

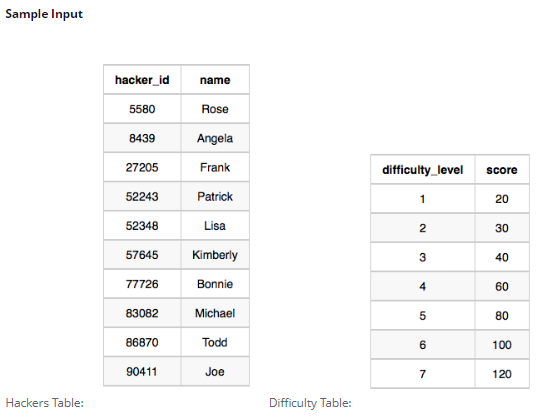
end

,G.grade,S.marks

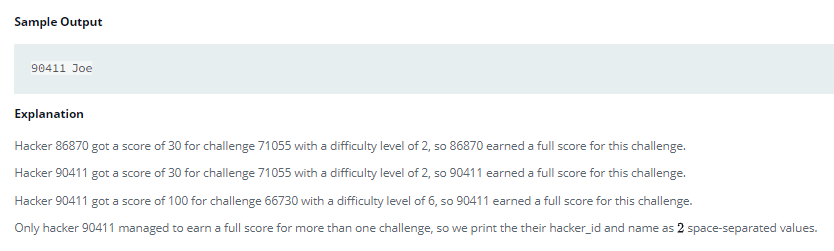
from Students S inner join Grades G on S.marks between G.min\_mark and G.max\_mark

order by G.grade desc, S.name, S.marks;









select T.hacker\_id, T.name

from

(select S.hacker\_id,H.name,count(\*)

from Submissions S

inner join Hackers H on S.hacker\_id=H.hacker\_id

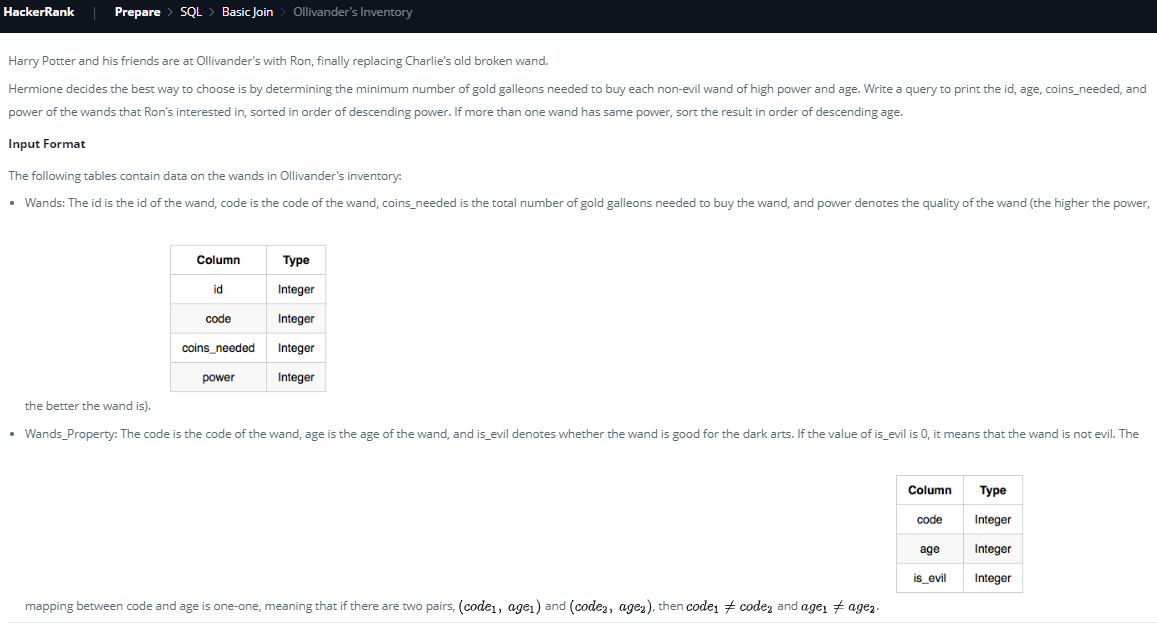
inner join Challenges C on S.challenge\_id = C.challenge\_id

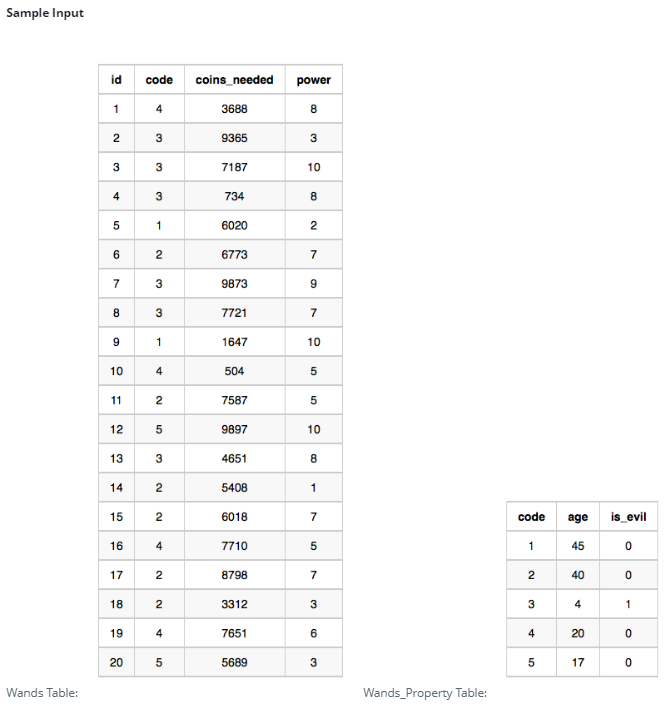
inner join Difficulty D on C.difficulty\_level = D.difficulty\_level and S.score=D.score

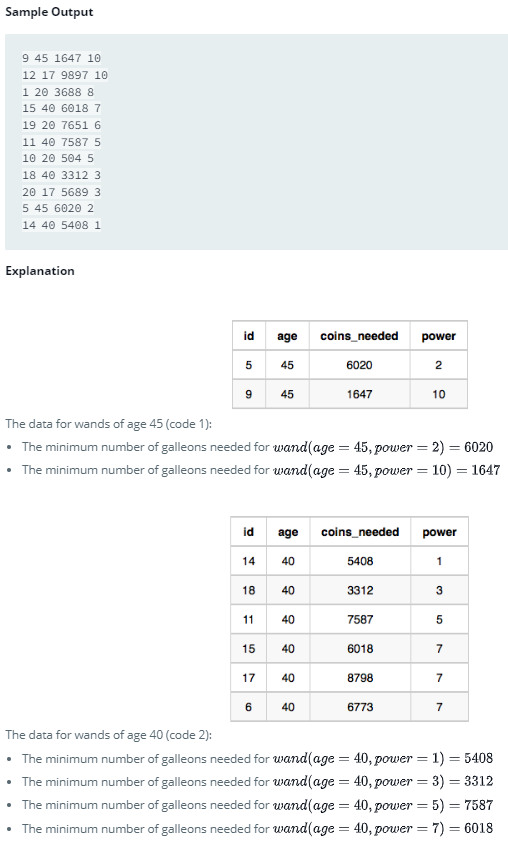
group by S.hacker\_id, H.name

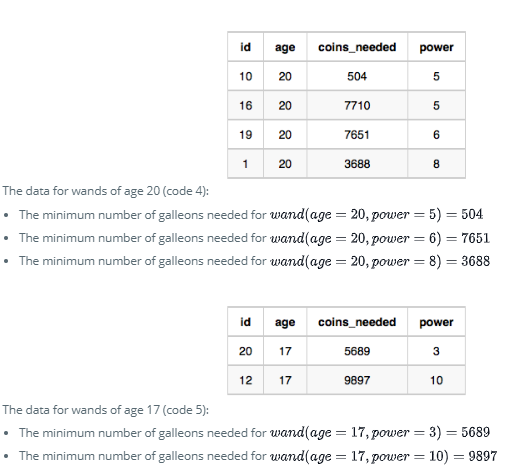
having count(\*)>1

order by count(\*) desc, S.hacker\_id) T









select W.id, T.age, T.minCoins, T.power

from

Wands W

inner join Wands\_property P on W.code=P.code

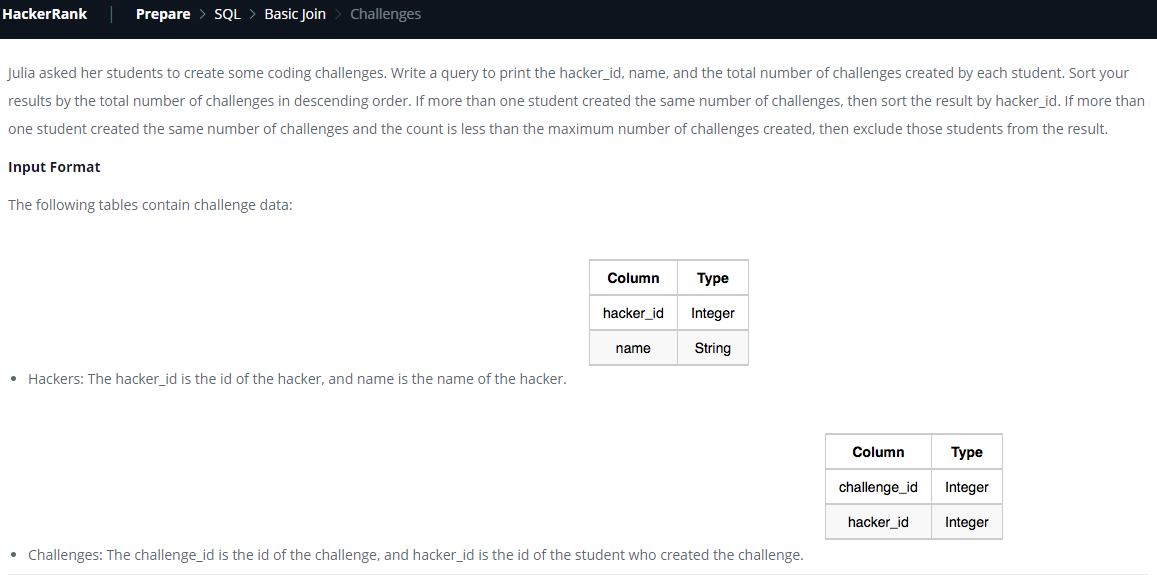
inner join (select W.power, P.age, min(W.coins\_needed) as minCoins

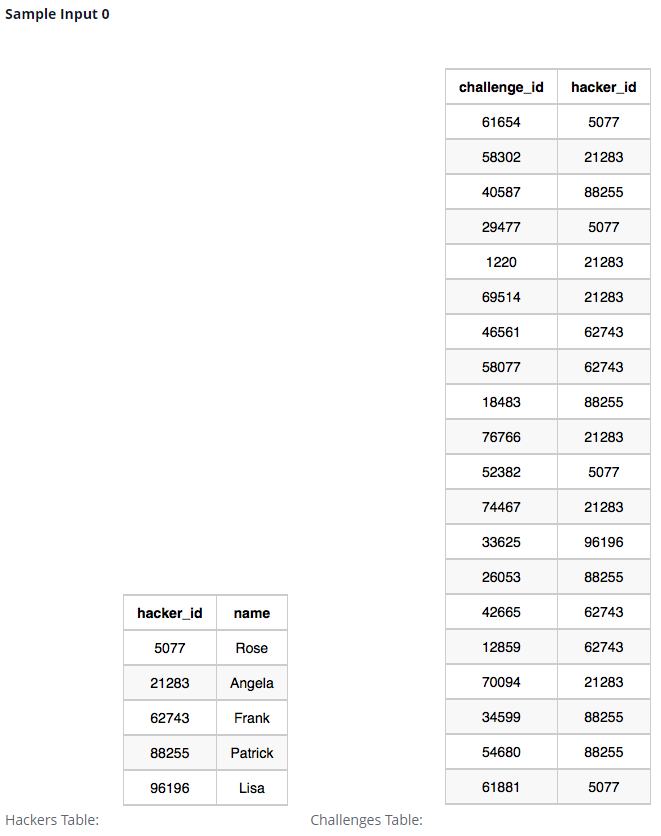
from Wands W inner join Wands\_property P on W.code=P.code and P.is\_evil=0

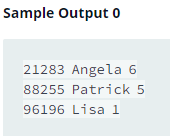
group by W.power, P.age) T

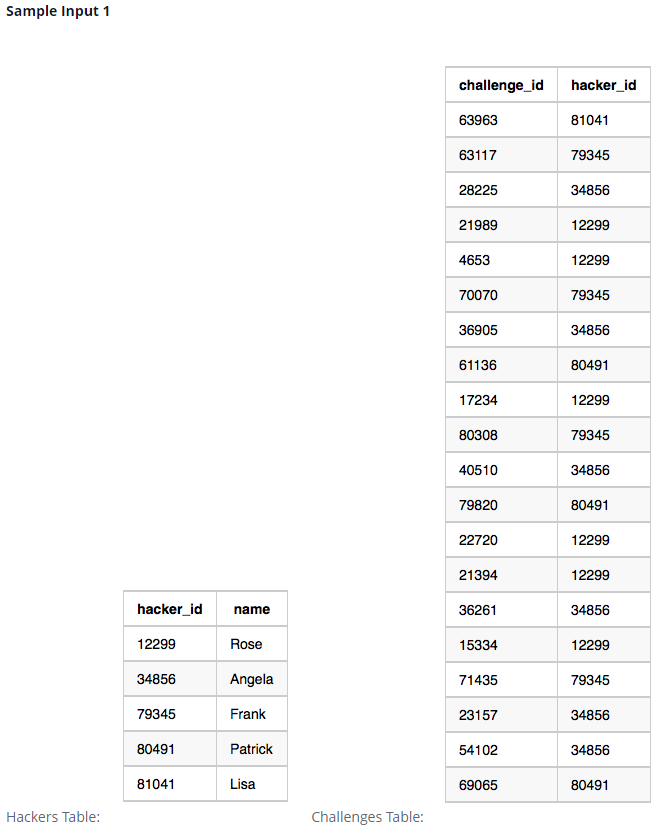
on T.minCoins=W.coins\_needed and T.power=W.power and T.age=P.age

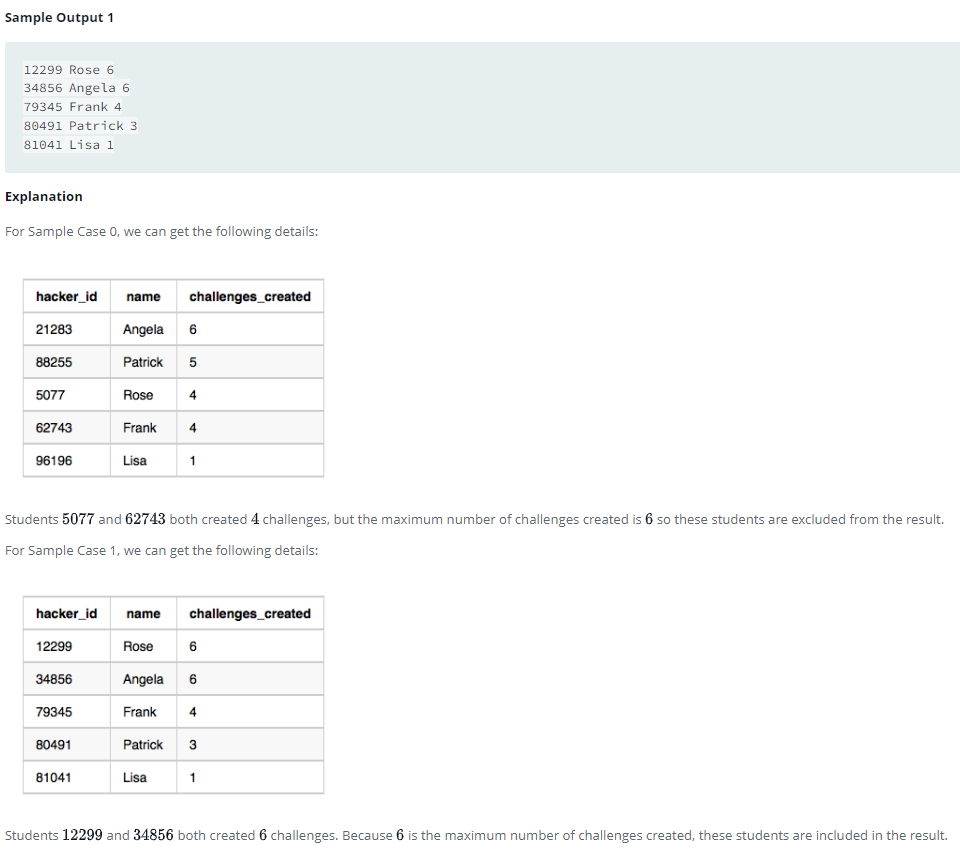
order by T.power desc, T.age desc









With

T1 as (select hacker\_id, count(\*) as cc from Challenges group by hacker\_id),

T2 as (select cc, count(cc) as countcc from T1 group by cc)

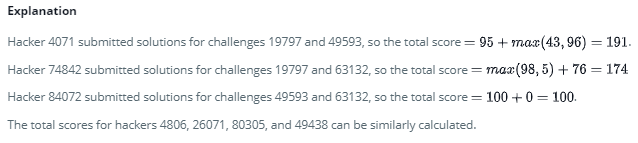
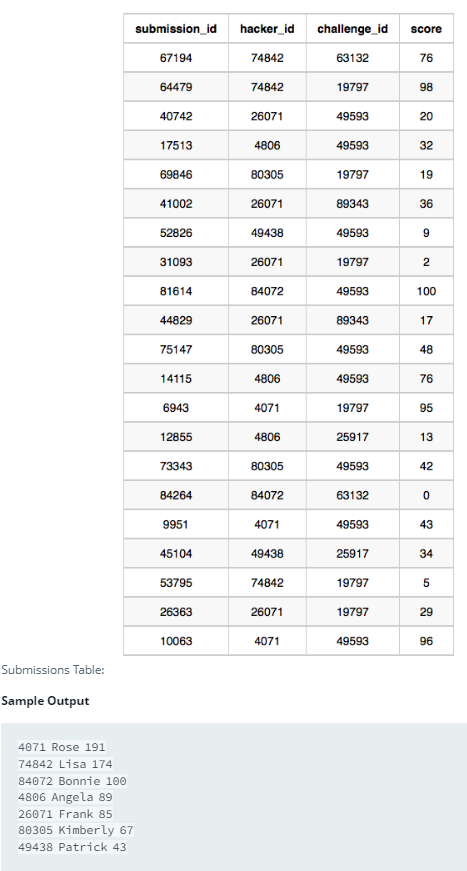
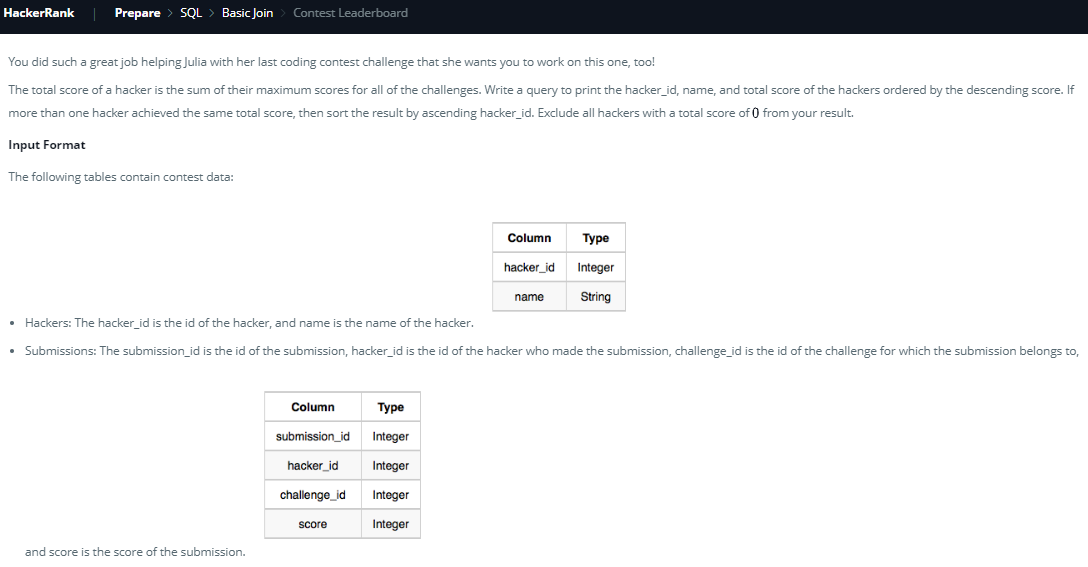
select T1.hacker\_id, H.name, T1.cc

from T1 inner join Hackers H on T1.hacker\_id=H.hacker\_id

inner join T2 on T2.cc=T1.cc

where T2.countcc=1 or (T1.cc in (select max(cc) from T1))

order by T1.cc desc, T1.hacker\_id;



select T.hacker\_id, H.name, sum(T.maxScore) as totalScore

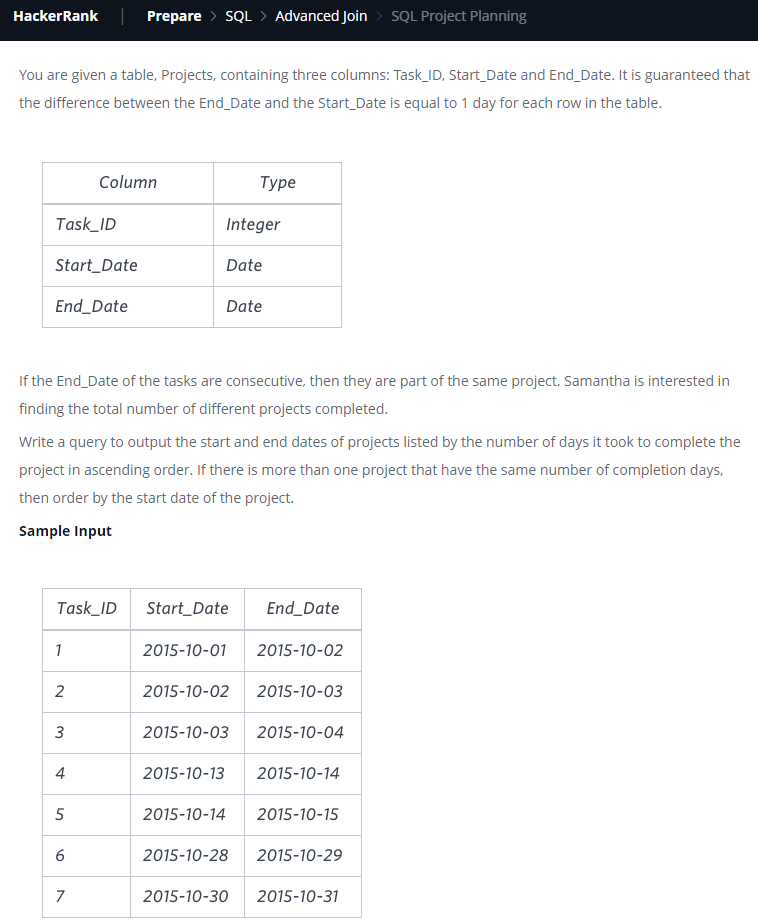
from (select hacker\_id,challenge\_id,max(score) as maxScore from Submissions group by hacker\_id,challenge\_id) T

inner join Hackers H on T.hacker\_id=H.hacker\_id

group by T.hacker\_id, H.name

having totalScore>0

order by totalScore desc, T.hacker\_id

with

T(sd1,ed1,sd2,ed2,sd3,ed3) as

(select

Yesterday.start\_date, Yesterday.end\_date,

Today.start\_date, Today.end\_date,

Tomorrow.start\_date, Tomorrow.end\_date

from

Projects Yesterday

right join Projects Today on Yesterday.end\_date = Today.start\_date

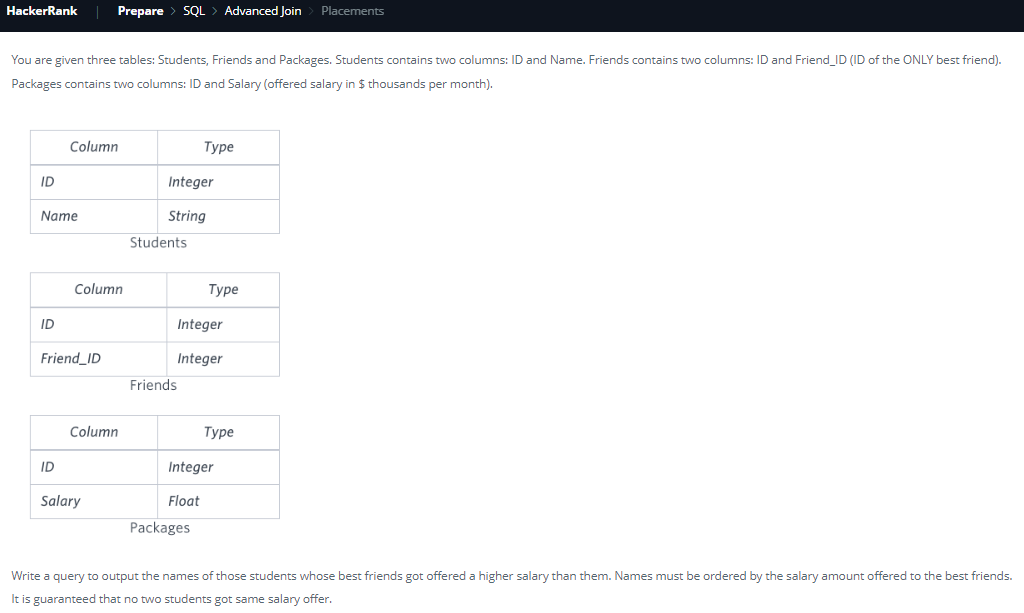
left join Projects Tomorrow on Today.end\_date = Tomorrow.start\_date),

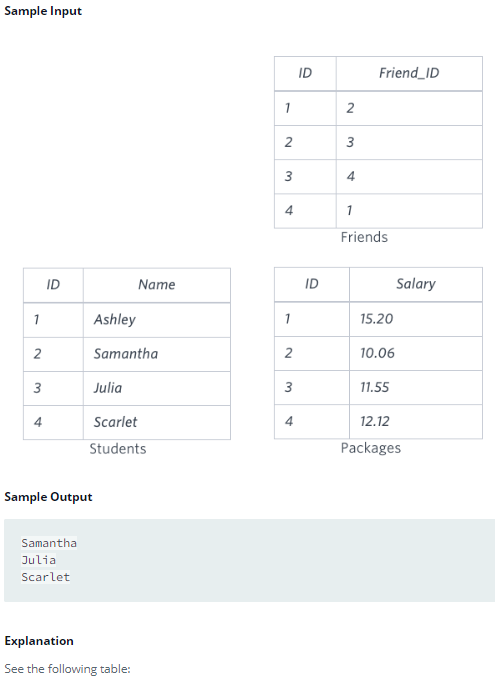
StartDate(projectNo,project\_start\_date) as (select row\_number() over (order by sd2),sd2 from T where sd1 is null),

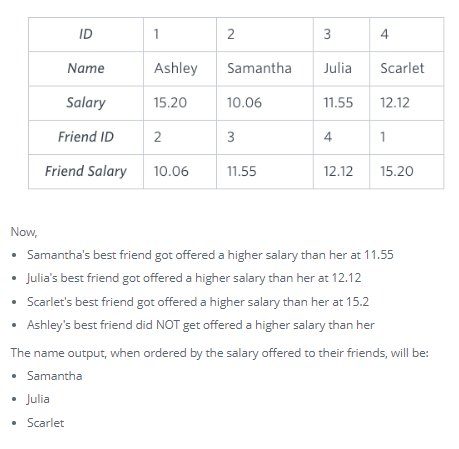
EndDate(projectNo,project\_end\_date) as (select row\_number() over (order by ed2), ed2 from T where sd3 is null)

select sd.project\_start\_date, ed.project\_end\_date from StartDate sd inner join EndDate ed on sd.projectNo=ed.projectNo

order by (ed.project\_end\_date-sd.project\_start\_date), sd.project\_start\_date;







select S\_me.name

from Friends F

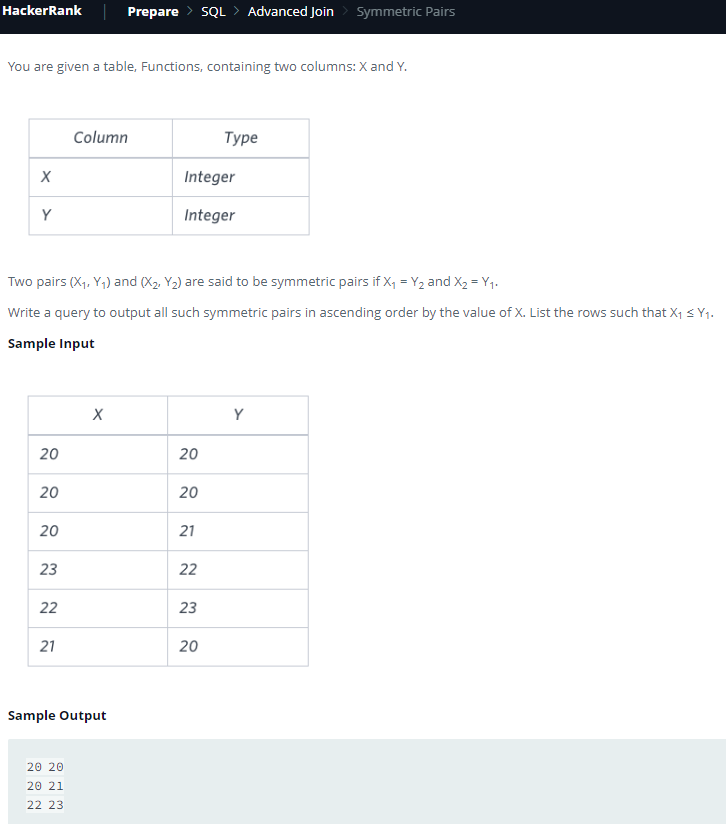
inner join Students S\_me on F.id=S\_me.id

inner join Packages P\_me on P\_me.id=S\_me.id

inner join Packages P\_friend on P\_friend.id=F.friend\_id

where P\_me.salary<P\_friend.salary

order by P\_friend.salary;

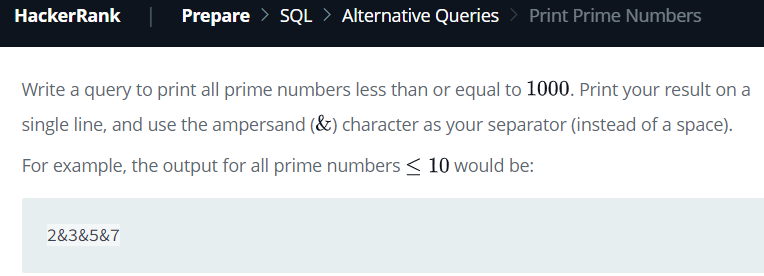


with F(sno,x,y) as (select row\_number() over(order by x),Functions.\* from Functions)

select f1.x,f1.y from F f1 inner join F f2 on f1.sno<f2.sno and f1.x=f2.y and f2.x=f1.y

where f1.x<=f1.y

order by f1.x;

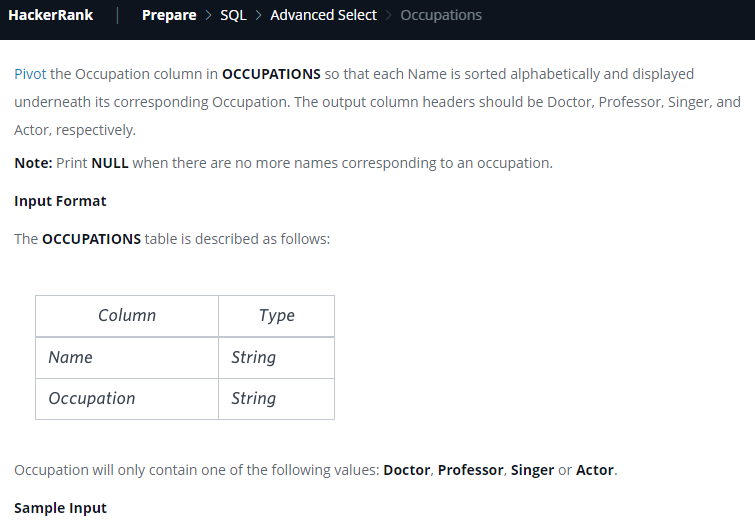


with

recursive AllNumbers(n) as (select 1 union all select n+1 from allNumbers where n<1000)

select group\_concat(Num.n separator '&') from AllNumbers Num

where not exists(select n from AllNumbers FactorsOfNum where FactorsOfNum.n>1 and FactorsOfNum.n<Num.n and Num.n%FactorsOfNum.n=0) and Num.n>1;





select

min(case when occupation='Doctor' then Name end),

min(case when occupation='Professor' then Name end),

min(case when occupation='Singer' then Name end),

min(case when occupation='Actor' then Name end)

from (select name, occupation, row\_number() over (partition by occupation order by name) as Sno from Occupations) T

group by T.Sno;

/\*First partition the Occupation table by Occupation and put Serial No for each Name within the Occupation. Note that this serialNo corresponds to the rowNumber for output.

Now group using SerialNo. For each given occupation and serial no, one single name can be selected. Use any aggregate function(either min/max with name) since we have to use 'group by serialNo'.

Also note: don't leave space after min. e.g. Wrong:"min (case...)". Correct:"min(case...)". Idk the reason but that space causes Error. \*/

It was basically trial and error only. I started trying with small queries growing to the whole answer. I saw the answers posted in the Discussions too. I shall try my best to explain the thought process behind this answer query.

I searched for "Pivot mySql" based on the keyword in question. I saw this site while searching and it was helpful: <https://www.databasestar.com/mysql-pivot/>

In the example given in this site, we can think of product\_name like an index for North, Central, South, West. To create a similar solution for our problem, we need SerialNo as an index for Doctor,Professor,Singer,Actor.

**Try1:** Adding serial number to each occupation. Used row\_number() for the same. For example, showing only for Doctor.

select name, occupation, row\_number() over(order by name) as sno from occupations where occupation = 'Doctor';

**Output1:**

Aamina Doctor 1

Julia Doctor 2

Priya Doctor 3

**Try 2**: Create the same OCCUPATIONS table but with added column Serial\_No. To get that, take UNION (Here UNION or UNION ALL works I guess) to get a combined table for each table from try 1. For example, just showing the result after combining two occupations only.

with

D as (select name, occupation, row\_number() over(order by name) as sno from occupations where occupation = 'Doctor'),

A as (select name, occupation, row\_number() over(order by name) as sno from occupations where occupation = 'Actor'),

DA as ((select \* from D) union (select \* from A))

select \* from DA

**Output 2:**

Aamina Doctor 1

Julia Doctor 2

Priya Doctor 3

Eve Actor 1

Jennifer Actor 2

Ketty Actor 3

Samantha Actor 4

**Try3:** Pivot using CASE. Now we can apply that method used in the example (ProductName,North,Central,South,West)

with

D as (select name, occupation, row\_number() over(order by name) as sno from occupations where occupation = 'Doctor'),

A as (select name, occupation, row\_number() over(order by name) as sno from occupations where occupation = 'Actor'),

DA as ((select \* from D) union (select \* from A))

select

min(case when occupation = 'Doctor' then name else NULL end) as Doctor,

min(case when occupation = 'Actor' then name else NULL end) as Actor

from DA group by sno

**Output 3:**

Aamina Eve

Julia Jennifer

Priya Ketty

NULL Samantha

**Try 4:** Try3 gave answer for Doctor and Actor only. Doing the same for all the four occupations (100% correct solution)

with

D as (select name, occupation, row\_number() over(order by name) as sno from occupations where occupation = 'Doctor'),

P as (select name, occupation, row\_number() over(order by name) as sno from occupations where occupation = 'Professor'),

S as (select name, occupation, row\_number() over(order by name) as sno from occupations where occupation = 'Singer'),

A as (select name, occupation, row\_number() over(order by name) as sno from occupations where occupation = 'Actor'),

DPSA as ((select \* from D) union (select \* from P) union (select \* from S) union (select \* from A))

select

min(case when occupation = 'Doctor' then name else NULL end) as Doctor,

min(case when occupation = 'Professor' then name else NULL end) as Professor,

min(case when occupation = 'Singer' then name else NULL end) as Singer,

min(case when occupation = 'Actor' then name else NULL end) as Actor

from DPSA

group by sno

**Try 5:** After reading discussions, I came to realize that instead of try1 and try2, we can use "Partition By" to get the same DPSA table(Result after try2) in one go. Also noticed that in CASE statement, it was not necessary to mention "ELSE NULL". Hence the final answer.

select

min(case when occupation='Doctor' then Name end),

min(case when occupation='Professor' then Name end),

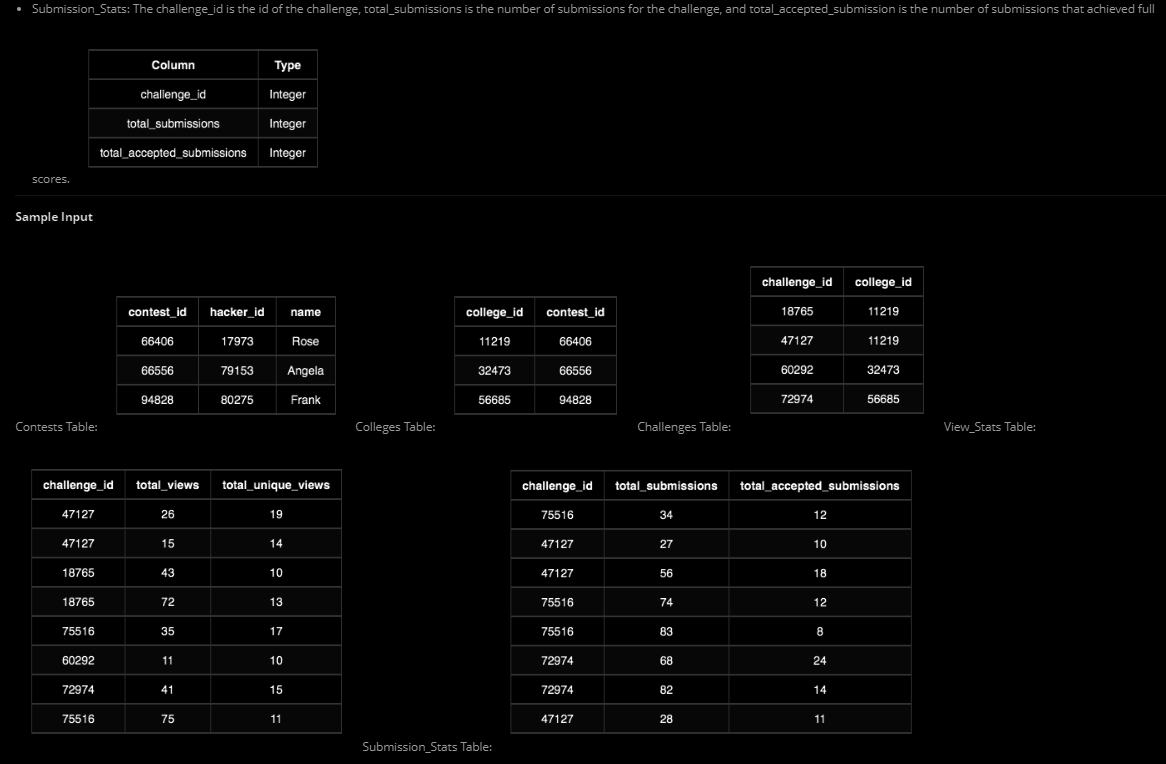
min(case when occupation='Singer' then Name end),

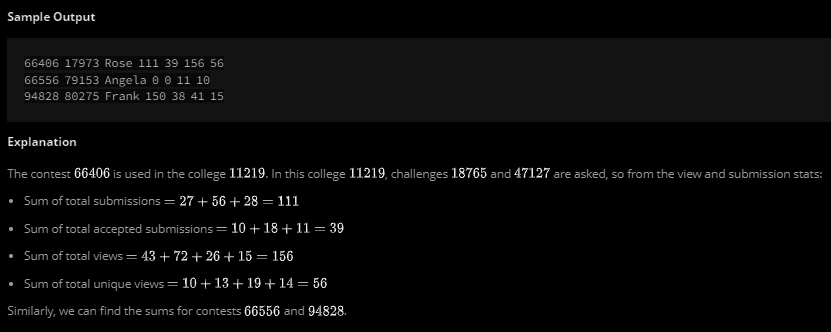
min(case when occupation='Actor' then Name end)

from (select name, occupation, row\_number() over (partition by occupation order by name) as Sno from Occupations) T

group by T.Sno;







Language: MS SQL Server

select

Con.contest\_id, Con.hacker\_id, Con.name, sum(SS.ts), sum(SS.tas), sum(VS.tv), sum(VS.tuv)

from

Contests Con

inner join Colleges Col on Col.contest\_id=Con.contest\_id

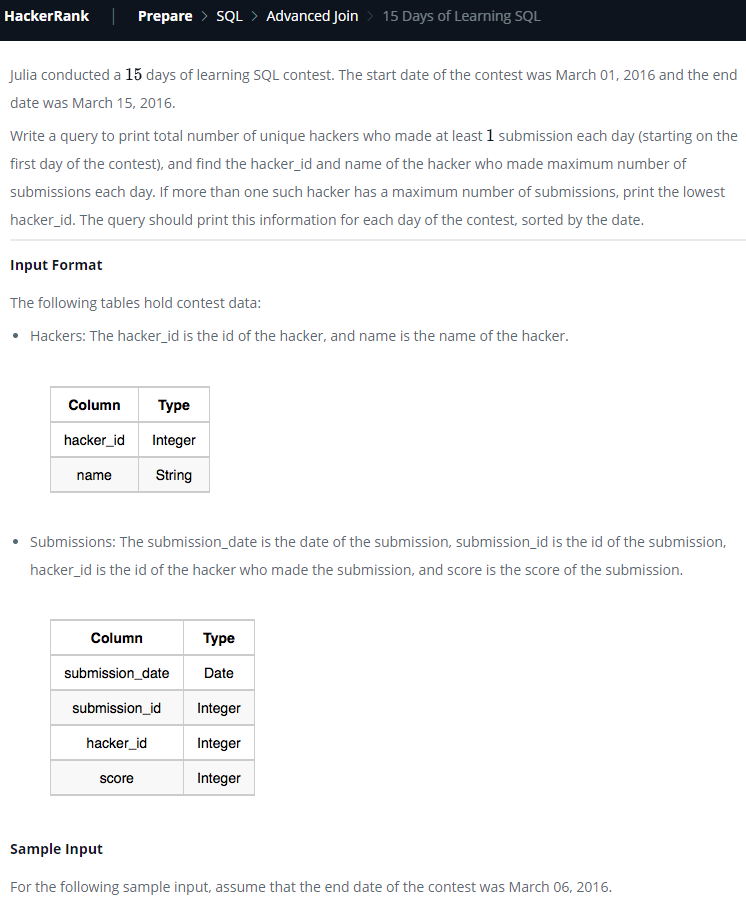
inner join Challenges Cha on Cha.college\_id=Col.college\_id

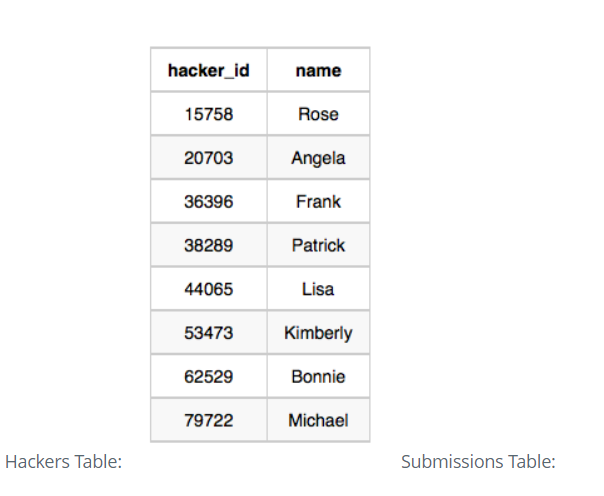
left join (select challenge\_id, sum(total\_submissions) ts, sum(total\_accepted\_submissions) tas from Submission\_Stats group by challenge\_id) SS on SS.challenge\_id=Cha.challenge\_id

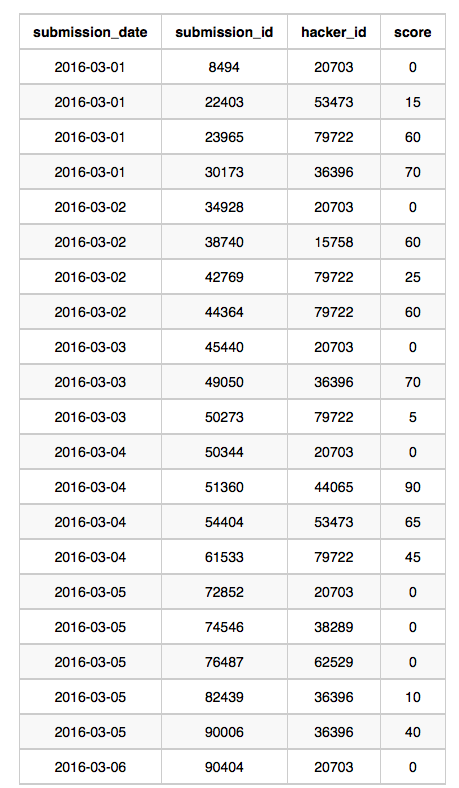
left join (select challenge\_id, sum(total\_views) tv, sum(total\_unique\_views) tuv from View\_Stats group by challenge\_id) VS on VS.challenge\_id=Cha.challenge\_id

group by Con.contest\_id, Con.hacker\_id, Con.name

order by Con.contest\_id;









Language: MS SQL Server

with

-- Query Q1 appends submissions Table with dateRank and hackerDates ranking columns

Q1 as

(select

submission\_date,

submission\_id,

hacker\_id,

dense\_rank() over (order by submission\_date) as dateRank,

-- This column calculates such that Mar 1 corresponds to 1, Mar 2 corresponds to 2 ,... Mar 15 corresponds to 15

dense\_rank() over (partition by hacker\_id order by submission\_date) as hackerDates

-- This column calculates the no of days participated by each hacker\_id starting from 1 Mar,2016 corresponding to each date from Mar 1-15 2016.

--E.g. on Mar 10, for hacker\_id=A, if hackerDates=10 then that means A has been playing on all the 10 days (Mar 1-10)

from submissions),

-- For each submission\_date, the query Q2 calculates the number of unique hacker\_ids who made at least one submission every day from Mar 1 to submission\_date

Q2 as

(select

submission\_date,

count(distinct hacker\_id) as UniqueHackerCount

from Q1

where dateRank=hackerDates

group by submission\_date),

-- For each submission\_date, Query Q3 Calculates the hacker\_id with max number of submission each day. If more than one such hacker\_id, only the lowest hacker\_id is shown.

Q3 as

(select distinct

submission\_date,

first\_value(hacker\_id) over(partition by submission\_date order by count(submission\_id) desc, hacker\_id) as MaxSubmissionHackerId

from

submissions

group by submission\_date, hacker\_id)

select Q2.submission\_date, Q2.UniqueHackerCount, Q3.MaxSubmissionHackerId, H.name as MaxSubmissionHackerName

from

Q2

inner join Q3 on Q2.submission\_date=Q3.submission\_date

inner join Hackers H on H.hacker\_id=Q3.MaxSubmissionHackerId

order by submission\_date