[Basic Select](https://www.hackerrank.com/domains/sql/select)

1.

select \*

from city

where POPULATION > 100000 and CountryCode = "USA"

2.

select name

from city

where POPULATION > 120000 and CountryCode = "USA"

3

select \*

from city

4

select \*

from city

where ID=1661

5

select \*

from city

where Countrycode="JPN"

6

select name

from city

where Countrycode="JPN"

7

select city , state

from station

8

select distinct city

from station

where ID%2=0

9

select (count(city)-count(distinct city) )

from station

10

select top 1 city,len(city)

from station

where len(city)=(select min(len(city) )

from station)

order by city;

select top 1 city,len(city)

from station

where len(city)=(select max(len(city) )

from station)

order by city

11

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%')

12

SELECT distinct City FROM STATION WHERE City LIKE '[A,E,I,O,U]%'

13

SELECT distinct City FROM STATION WHERE City LIKE '%[A,E,I,O,U]'

14

SELECT distinct City FROM STATION WHERE City LIKE '[A,E,I,O,U]%[A,E,I,O,U]'

15

SELECT distinct City FROM STATION WHERE City Not like '[A,E,I,O,U]%'

16

SELECT distinct City FROM STATION WHERE City Not like '[A,E,I,O,U]%[A,E,I,O,U]'

17

SELECT distinct City FROM STATION WHERE City Not like '[A,E,I,O,U]%' and City Not like '%[A,E,I,O,U]'

18

select name

from students

where marks>75

order by right(name,3),id

19

select name

from employee

order by name

20

select name

from employee

where salary>2000 and months<10

order by employee\_id

[Advanced Select](https://www.hackerrank.com/domains/sql/advanced-select)

1

select name

from students

where marks>75

order by right(name,3),id

2

select name

from employee

order by name

3

select name

from employee

where salary>2000 and months<10

order by employee\_id

4

SELECT CASE

WHEN P IS NULL THEN CONCAT(N, ' Root')

WHEN N IN (SELECT DISTINCT P FROM BST) THEN CONCAT(N, ' Inner')

ELSE CONCAT(N, ' Leaf')

END

FROM BST

ORDER BY N ASC

5

select c.company\_code,c.founder,count(distinct e.lead\_manager\_code),count(distinct e.senior\_manager\_code),count(distinct e.manager\_code),count( distinct e.employee\_code)

from employee e, company c

where e.company\_code=c.company\_code

group by c.company\_code, c.founder

order by c.company\_code

Advanced Select Challenges

**1**

select

case

when (a+b<=c) then "Not A Triangle"

when a=b and b=c and c=a then "Equilateral"

when a<>b and b<>c and c<>a then "Scalene"

when a=b or b=c or c=a then "Isosceles"

end

from TRIANGLES

**2**

select name+"("+left(Occupation,1)+")"

from OCCUPATIONS

order by name

select "There are a total of "+str(count(Occupation))+" "+lower(Occupation)+"s."

from OCCUPATIONS

group by Occupation

order by count(Occupation),Occupation

3

select (case

when g.grade<=7 then null

else s.name

end),

g.grade,s.marks

from Students s left join grades g ON s.marks BETWEEN min\_mark AND max\_mark

order by g.grade DESC , s.name ASC;

4

SELECT CASE

WHEN P IS NULL THEN CONCAT(N, ' Root')

WHEN N IN (SELECT DISTINCT P FROM BST) THEN CONCAT(N, ' Inner')

ELSE CONCAT(N, ' Leaf')

END

FROM BST

ORDER BY N ASC

5

select c.company\_code,c.founder,count(distinct e.lead\_manager\_code),count(distinct e.senior\_manager\_code),count(distinct e.manager\_code),count( distinct e.employee\_code)

from employee e, company c

where e.company\_code=c.company\_code

group by c.company\_code, c.founder

order by c.company\_code

Aggregations

1

select count(countrycode)

from city

where Population>100000

2

select sum(population)

from city

where District="California"

3

select sum(population)

from city

where District="California"

4

select avg(population)

from city

where District="California"

5

select round(avg(population),0)

from city

6

select sum(population)

from city

where COUNTRYCODE ="JPN"

7

select max(population)-min(population)

from city

8

SELECT CEIL(AVG(Salary)-AVG(REPLACE(Salary,'0',''))) FROM EMPLOYEES;

9

select top 1 months\*salary as earning,count(\*)

from Employee

group by months\*salary

order by earning DESC

10

select convert(decimal(10, 2),sum(lat\_n)),convert(decimal(10, 2),sum(long\_w))

from STATION

11

select convert(decimal(10,4),sum(lat\_n))

from station

where lat\_n>38.7880 and lat\_n<137.2345

12

select convert(decimal(10,4),max(lat\_n))

from station

where lat\_n<137.2345

13

select convert(decimal(10,4),max(lat\_n))

from station

where lat\_n<137.2345

14

select convert(decimal(10,4),max(long\_w))

from station

where lat\_n = (

select max(lat\_n)

from station

where lat\_n<137.2345

)

15

select convert(decimal(10,4),min(LAT\_N))

from station

where lat\_n > 38.7780

16

select convert(decimal(10,4),min(LONG\_W))

from station

where lat\_n = (

select min(lat\_n)

from station

where lat\_n>38.7780

)

17

select convert(decimal(10,4),max(lat\_n)-min(lat\_n)+max(long\_w)-min(long\_w))

from station

18

select convert(decimal(10,4),sqrt(square(min(lat\_n)-max(lat\_n))+square(min(long\_w)-max(long\_w))))

from station

SELECT top 1 ROW\_NUMBER() OVER (order by lat\_n) as row,id

FROM STATION

19

SELECT CAST(LAT\_N AS DECIMAL (7,4))

FROM

(SELECT LAT\_N, ROW\_NUMBER() OVER (ORDER BY LAT\_N) as ROWNU

FROM STATION

) AS X

WHERE ROWNU = ( SELECT ROUND((COUNT(LAT\_N)+1)/2,0)

FROM STATION

);

20

SELECT CAST(LAT\_N AS DECIMAL (7,4))

FROM

(SELECT LAT\_N, ROW\_NUMBER() OVER (ORDER BY LAT\_N) as ROWNU

FROM STATION

) AS X

WHERE ROWNU = ( SELECT ROUND((COUNT(LAT\_N)+1)/2,0)

FROM STATION

);

Basic Join Challenges

1

select sum(c.population)

from city c, country co

where co.CONTINENT="Asia" and c.CountryCode=co.code

2

select c.name

from city c, country co

where co.CONTINENT="Africa" and c.CountryCode=co.code

3

select co.CONTINENT,avg(c.population)

from city c, country co

where c.CountryCode=co.Code

group by CONTINENT

4

select h.hacker\_id,name

from Hackers h, Challenges c, Submissions s, difficulty d

where h.hacker\_id = s.hacker\_id and c.challenge\_id=s.challenge\_id and d.difficulty\_level=c.difficulty\_level and d.score=s.score

group by h.name,h.hacker\_id

having count(h.hacker\_id)>1

order by count(h.hacker\_id) desc, h.hacker\_id

5

select w.id,wp.age,w.coins\_needed,w.power from wands as w,wands\_property as wp where w.code=wp.code and wp.is\_evil=0 and w.coins\_needed =(select min(wo.coins\_needed) from wands as wo where wo.power=w.power and wo.code=w.code) order by power desc,age desc

6

select w.id,wp.age,w.coins\_needed,w.power

from wands w,wands\_property wp

where w.code=wp.code and wp.is\_evil=0

and w.coins\_needed =(select min(wo.coins\_needed)

from wands wo

where wo.power=w.power

and wo.code=w.code)

order by power desc,age desc

7

WITH COUNTER AS (SELECT HACKER\_ID, CNT=COUNT(\*)

FROM CHALLENGES C

GROUP BY HACKER\_ID),

MAX\_CNT AS (SELECT MAX\_CNT1=MAX(CNT) FROM COUNTER),

ONLY1 AS (SELECT CNT FROM COUNTER

WHERE CNT<(SELECT TOP 1 MAX\_CNT1 FROM MAX\_CNT)

GROUP BY CNT

HAVING COUNT(\*)=1

UNION

SELECT MAX\_CNT1 FROM MAX\_CNT)

SELECT H.HACKER\_ID, H.NAME, C.CNT

FROM COUNTER C , ONLY1 O ,HACKERS H

where C.CNT=O.CNT

and C.HACKER\_ID=H.HACKER\_ID

ORDER BY C.CNT DESC, HACKER\_ID ASC;

8

select h.hacker\_id, name, sum(score) as total\_score

from hackers as h inner join

(select hacker\_id, max(score) as score from submissions group by challenge\_id, hacker\_id) max\_score

on h.hacker\_id=max\_score.hacker\_id

group by h.hacker\_id, name

having total\_score > 0

order by total\_score desc, h.hacker\_id

;

#### Advanced Join Challenges

1

select name

from Students s, Friends f, Packages sp,Packages fp

where s.id=f.id

and s.id=sp.id

and f.Friend\_ID=fp.id

and sp.salary<fp.salary

order by fp.salary

2

select name

from Students s, Friends f, Packages sp,Packages fp

where s.id=f.id

and s.id=sp.id

and f.Friend\_ID=fp.id

and sp.salary<fp.salary

order by fp.salary

3

select f.x,f.y

from functions f join functions s on f.x=s.y and f.y=s.x and f.x<s.x

union

select x,y from functions f where x=y and ((select count(\*) from functions where x=f.x and y=f.y)>1)

order by x

4

WITH SUM\_View\_Stats AS (

SELECT challenge\_id

, total\_views = sum(total\_views)

, total\_unique\_views = sum(total\_unique\_views)

FROM View\_Stats

GROUP BY challenge\_id

)

,SUM\_Submission\_Stats AS (

SELECT challenge\_id

, total\_submissions = sum(total\_submissions)

, total\_accepted\_submissions = sum(total\_accepted\_submissions)

FROM Submission\_Stats

GROUP BY challenge\_id

)

SELECT con.contest\_id

, con.hacker\_id

, con.name

, SUM(total\_submissions)

, sum(total\_accepted\_submissions)

, sum(total\_views)

, sum(total\_unique\_views)

FROM Contests con

INNER JOIN Colleges col

ON con.contest\_id = col.contest\_id

INNER JOIN Challenges cha

ON cha.college\_id = col.college\_id

LEFT JOIN SUM\_View\_Stats vs

ON vs.challenge\_id = cha.challenge\_id

LEFT JOIN SUM\_Submission\_Stats ss

ON ss.challenge\_id = cha.challenge\_id

GROUP BY con.contest\_id,con.hacker\_id,con.name

HAVING (SUM(total\_submissions)

+sum(total\_accepted\_submissions)

+sum(total\_views)

+sum(total\_unique\_views)) <> 0

ORDER BY con.contest\_ID

5

SELECT A.contest\_id, A.hacker\_id, A.Name,

SUM(total\_submissions) As total\_submissions,

SUM(total\_accepted\_submissions) AS total\_accepted\_submissions,

SUM(total\_views) AS total\_views,

SUM(total\_unique\_views) AS total\_unique\_views

FROM Contests AS A

LEFT JOIN Colleges AS B

ON A.contest\_id = B.contest\_id

LEFT JOIN Challenges AS C

ON B.college\_id = C.college\_id

LEFT JOIN (SELECT challenge\_id, SUM(total\_views) AS total\_views,

SUM(total\_unique\_views) AS total\_unique\_views

FROM View\_Stats

GROUP BY challenge\_id) AS D

ON C.challenge\_id = D.challenge\_id

LEFT JOIN (SELECT challenge\_id, SUM(total\_submissions) AS total\_submissions,

SUM(total\_accepted\_submissions) AS total\_accepted\_submissions

FROM Submission\_Stats

GROUP BY challenge\_id) AS E

ON C.challenge\_id = E.challenge\_id

GROUP BY A.contest\_id, A.hacker\_id, A.Name

HAVING (total\_submissions + total\_accepted\_submissions + total\_views + total\_unique\_views) > 0

ORDER BY A.contest\_id

;

Alternative Queries Challenges

1

DECLARE @i INT = 0,@j INT =20;

WHILE @i < @j

BEGIN

print replicate("\* ",@j-@i)

SET @i = @i + 1;

END;

2

Declare @i INT = 1

while @i<=20

begin

print replicate("\* ",@i)

SET @i = @i + 1;

End

3

DECLARE @i Int = 2, @j Int =2, @f Int =0;

declare @s varchar(max)

WHILE(@i<=1000)

BEGIN

set @f=0;

set @j=2;

WHILE(@j<=sqrt(@i))

BEGIN

if (@i%@j=0)

BEGIN

SET @f=1;

break;

END

set @j=@j+1;

END

if(@f=0)

set @s=concat(@s,@i,'&')

set @i=@i+1;

END

print left(@s,len(@s)-1)

SELECT Start\_Date, min(End\_Date)

FROM

(SELECT Start\_Date FROM Projects WHERE Start\_Date NOT IN (SELECT End\_Date FROM Projects)) a,

(SELECT End\_Date FROM Projects WHERE End\_Date NOT IN (SELECT Start\_Date FROM Projects)) b

WHERE Start\_Date < End\_Date

GROUP BY Start\_Date

ORDER BY DATEDIFF(day,Start\_Date,min(End\_Date)), Start\_Date