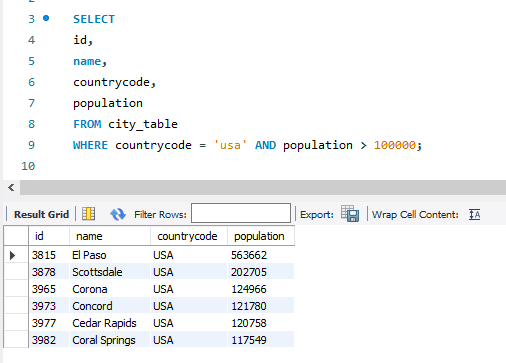
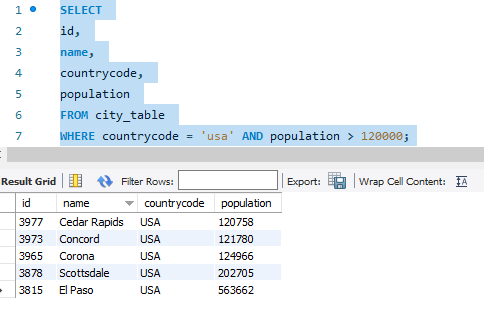
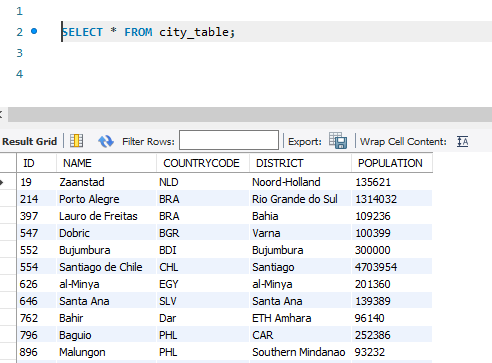
**Q1**. Query all columns for all American cities in the CITY table with populations larger than 100000.



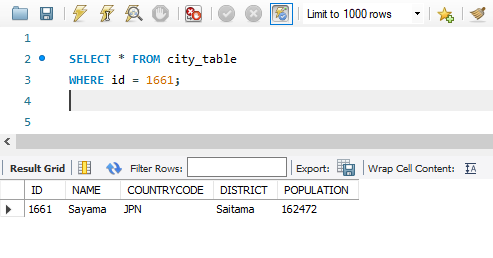
**Q2**. Query the NAME field for all American cities in the CITY table with populations larger than 120000.



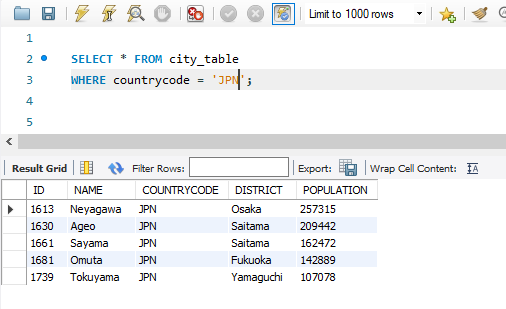
**Q3.** Query all columns (attributes) for every row in the CITY table.



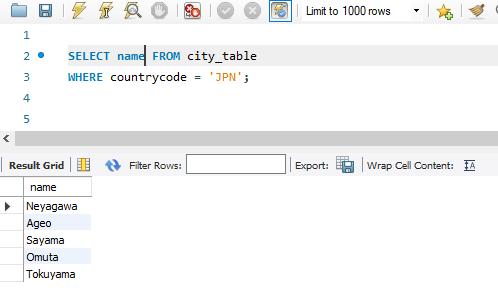
**Q4**. Query all columns for a city in CITY with the ID 1661.



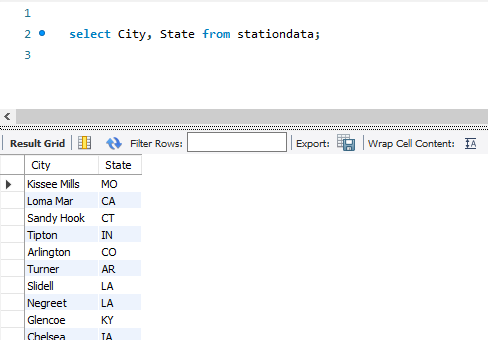
**Q5**. Query all attributes of every Japanese city in the CITY table. The COUNTRYCODE for Japan is



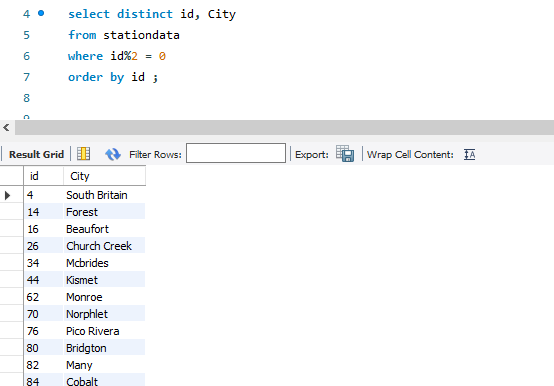
**Q6.** Query the names of all the Japanese cities in the CITY table.



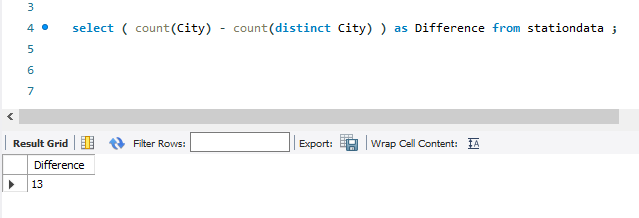
**Q7.** Query a list of CITY and STATE from the STATION table.



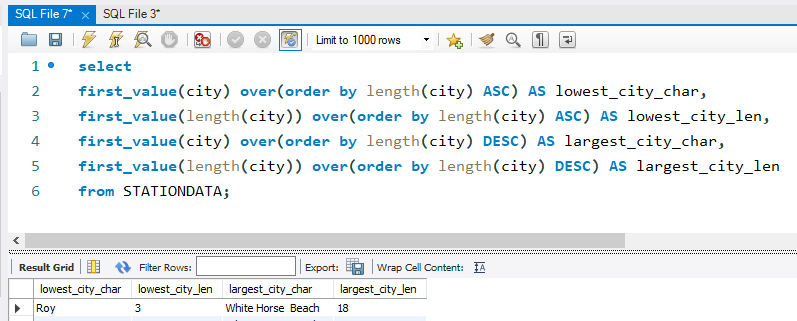
**Q8.** Query a list of CITY names from STATION for cities that have an even ID number. Print the results in any order, but exclude duplicates from the answer.



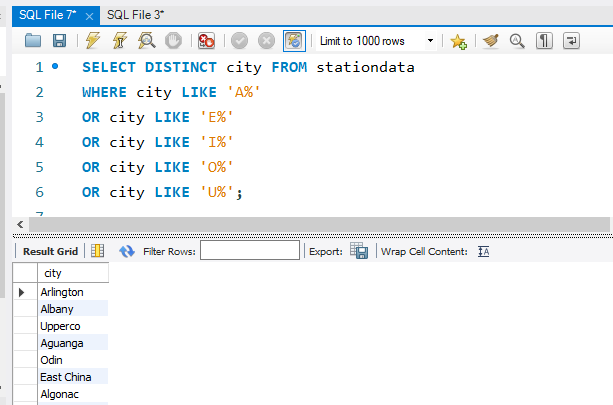
**Q9**. Find the difference between the total number of CITY entries in the table and the number of distinct CITY entries in the table.



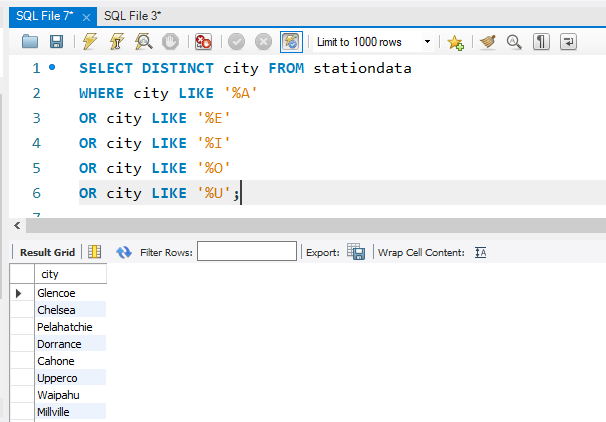
**Q10.** Query the two cities in STATION with the shortest and longest CITY names, as well as their respective lengths (i.e.: number of characters in the name). If there is more than one smallest or largest city, choose the one that comes first when ordered alphabetically.



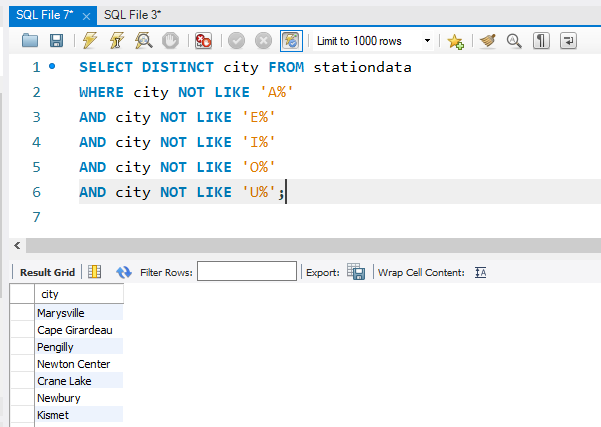
**Q11**. Query the list of CITY names starting with vowels (i.e., a, e, i, o, or u) from STATION.



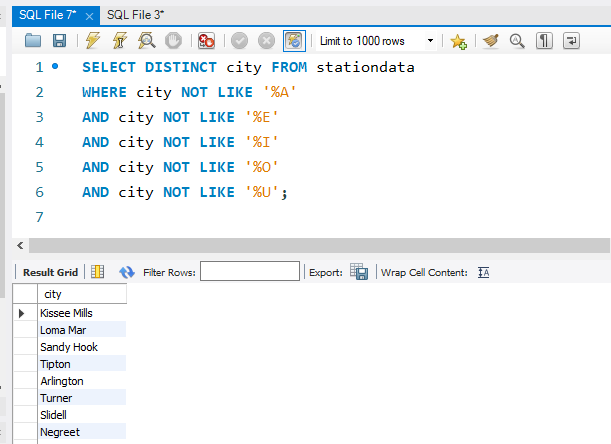
**Q12.** Query the list of CITY names ending with vowels (a, e, i, o, u) from STATION. Your result cannot contain duplicates.



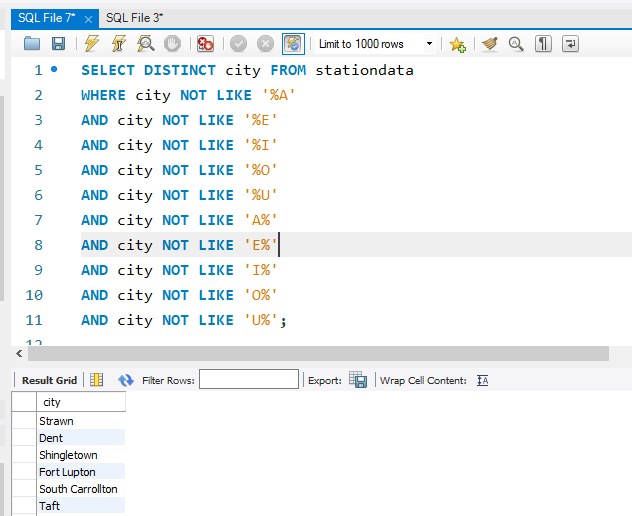
**Q13.** Query the list of CITY names from STATION that do not start with vowels. Your result cannot contain duplicates.



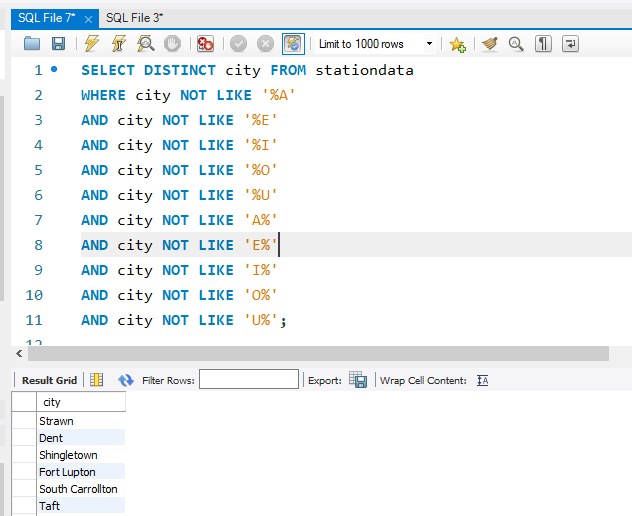
**Q14.** Query the list of CITY names from STATION that do not end with vowels. Your result cannot contain duplicates.



**Q15.** 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.



**Q16.** 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.



**Q.17**

Create table If Not Exists Product (product\_id int, product\_name varchar(10), unit\_price int)

Create table If Not Exists Sales (seller\_id int, product\_id int, buyer\_id int, sale\_date date, quantity int, price int)

Truncate table Product

insert into Product (product\_id, product\_name, unit\_price) values ('1', 'S8', '1000')

insert into Product (product\_id, product\_name, unit\_price) values ('2', 'G4', '800')

insert into Product (product\_id, product\_name, unit\_price) values ('3', 'iPhone', '1400')

Truncate table Sales

insert into Sales (seller\_id, product\_id, buyer\_id, sale\_date, quantity, price) values ('1', '1', '1', '2019-01-21', '2', '2000')

insert into Sales (seller\_id, product\_id, buyer\_id, sale\_date, quantity, price) values ('1', '2', '2', '2019-02-17', '1', '800')

insert into Sales (seller\_id, product\_id, buyer\_id, sale\_date, quantity, price) values ('2', '2', '3', '2019-06-02', '1', '800')

insert into Sales (seller\_id, product\_id, buyer\_id, sale\_date, quantity, price) values ('3', '3', '4', '2019-05-13', '2', '2800');

**Q.18**

Create table If Not Exists Views (article\_id int, author\_id int, viewer\_id int, view\_date date)

Truncate table Views

insert into Views (article\_id, author\_id, viewer\_id, view\_date) values ('1', '3', '5', '2019-08-01')

insert into Views (article\_id, author\_id, viewer\_id, view\_date) values ('1', '3', '6', '2019-08-02')

insert into Views (article\_id, author\_id, viewer\_id, view\_date) values ('2', '7', '7', '2019-08-01')

insert into Views (article\_id, author\_id, viewer\_id, view\_date) values ('2', '7', '6', '2019-08-02')

insert into Views (article\_id, author\_id, viewer\_id, view\_date) values ('4', '7', '1', '2019-07-22')

insert into Views (article\_id, author\_id, viewer\_id, view\_date) values ('3', '4', '4', '2019-07-21')

insert into Views (article\_id, author\_id, viewer\_id, view\_date) values ('3', '4', '4', '2019-07-21')

**Q.19**

select round(100\*d2.immediate\_orders/count(d1.delivery\_id), 2) as immediate\_percentage

from Delivery d1,

(select count(order\_date) as immediate\_orders

from Delivery

where (order\_date = customer\_pref\_delivery\_date)) d2

**Q.20**

select ad\_id,

ifnull(

round(

avg(

case

when action = "Clicked" then 1

when action = "Viewed" then 0

else null

end

) \* 100,

2),

0)

as ctr

from Ads

group by ad\_id

order by ctr desc, ad\_id asc;

**Q.21**

select e.employee\_id, (select count(team\_id) from Employee where e.team\_id = team\_id) as team\_size

from Employee e

OR

SELECT employee\_id, COUNT(team\_id) OVER (PARTITION BY team\_id) team\_size

FROM Employee

**Q.22**

select c.country\_name,

(case when (sum(w.weather\_state) / count(w.day) )<= 15 then "Cold"

when (sum(w.weather\_state) / count(w.day)) >=25 then "Hot"

else "Warm" end ) as weather\_type

from countries c

join weather w

on c.country\_id = w.country\_id

where w.day between '2019-11-01' and '2019-11-30'

group by c.country\_id

**Q.23**

select a.product\_id, round(

sum(a.price \* b.units)/sum(b.units),

2) as average\_price

from Prices as a

join UnitsSold as b

on a.product\_id = b.product\_id and (b.purchase\_date between a.start\_date and a.end\_date)

group by a.product\_id;

**Q.24**

create table Activity(player\_id int, device\_id int, event\_date date, games\_played int);

insert into Activity values(1,2,'2016-03-01', 5);

insert into Activity values(1,2,'2016-05-02', 6);

insert into Activity values(2,3,'2017-06-25', 1);

insert into Activity values(3,1,'2016-03-02', 0);

insert into Activity values(3,4,'2018-07-03', 5);

create table Result as

SELECT

player\_id,

min(event\_date) AS first\_login

FROM

Activity

GROUP BY player\_id;

Select \* from Result

**Q.25**

create table Activity(player\_id int, device\_id int, event\_date date, games\_played int);

insert into Activity values(1,2,'2016-03-01', 5);

insert into Activity values(1,2,'2016-05-02', 6);

insert into Activity values(2,3,'2017-06-25', 1);

insert into Activity values(3,1,'2016-03-02', 0);

insert into Activity values(3,4,'2018-07-03', 5);

create table Result as

SELECT

player\_id,

min(event\_date) AS first\_login

FROM

Activity

GROUP BY player\_id;

**Q.26**

select prod.product\_name, prod.unit

from

(select p.product\_name,

sum(o.unit) as unit

from Products p

left join Orders o

on p.product\_id = o.product\_id and

datediff(order\_date, '2020=02-01') >= 0 and

datediff('2020-03-01', order\_date) > 0

group by p.product\_id) prod

where prod.unit >= 100

**Q.27**

select \*

from Users

where mail regexp '^[A-Za-z][A-Za-z0-9\\_\.\-][\*@leetcode\.com$](mailto:*@leetcode\.com$)'

**Q.28**

select customer\_id, name

from

(

select o.customer\_id, c.name,

sum(case when left(o.order\_date,7) = '2020-06' then p.price \* o.quantity end) as JuneSpend,

sum(case when left(o.order\_date,7) = '2020-07' then p.price \* o.quantity end) as JulySpend

from Orders o

left join Customers c on o.customer\_id = c.customer\_id

left join Product p on o.product\_id = p.product\_id

group by o.customer\_id

having JuneSpend >= 100 and JulySpend >= 100

) as temp

**Q.29**

select distinct title  
from  
(  
 select title  
 from Content c  
 left join TVProgram t  
 on c.content\_id = t.content\_id  
 where Kids\_content = 'Y' and content\_type = 'Movies'  
 and date\_format(program\_date, '%Y-%m') = '2020-06') x

Q.30

select q.id, q.year, ifnull(n.npv,0) as npv

from queries as q

left join npv as n

on (q.id, q.year) = (n.id, n.year)

**Q.31**

select q.id, q.year, ifnull(n.npv,0) as npv

from queries as q

left join npv as n

on (q.id, q.year) = (n.id, n.year)

**Q.32**

SELECT

b.unique\_id AS unique\_id,

a.name AS name

FROM

Employees a

LEFT JOIN

EmployeeUNI b

ON

a.id = b.id;

**Q.33**

select name, sum(ifnull(distance, 0)) as travelled\_distance

from rides r

right join users u

on r.user\_id = u.id

group by name

order by 2 desc,1 asc;

**Q.34**

select a.product\_name, sum(unit) as unit

from Products a

left join Orders b

on a.product\_id = b.product\_id

where b.order\_date between '2020-02-01' and '2020-02-29'

group by a.product\_id

having sum(unit) >= 100

**Q.35**

SELECT user\_name AS results FROM

(

SELECT a.name AS user\_name, COUNT(\*) AS counts FROM Movie\_Rating AS b

JOIN Users AS a

on a.user\_id = b.user\_id

GROUP BY b.user\_id

ORDER BY counts DESC, user\_name ASC LIMIT 1

) first\_query

UNION

SELECT movie\_name AS results FROM

(

SELECT c.title AS movie\_name, AVG(d.rating) AS rate FROM Movie\_Rating AS d

JOIN Movies AS c

on c.movie\_id = d.movie\_id

WHERE substr(d.created\_at, 1, 7) = '2020-02'

GROUP BY d.movie\_id

ORDER BY rate DESC, movie\_name ASC LIMIT 1

) second\_query;

**Q.36**

select name, sum(ifnull(distance, 0)) as travelled\_distance

from rides r

right join users u

on r.user\_id = u.id

group by name

order by 2 desc,1 asc;

**Q.37**

SELECT

b.unique\_id AS unique\_id,

a.name AS name

FROM

Employees a

LEFT JOIN

EmployeeUNI b

ON

a.id = b.id;

**Q.38**

select id, name

from Students

where department\_id not in (select id from Departments);

**Q.39**

select from\_id as person1,to\_id as person2,

count(duration) as call\_count, sum(duration) as total\_duration

from (select \*

from Calls

union all

select to\_id, from\_id, duration

from Calls) t1

where from\_id < to\_id

group by person1, person2

**Q.40**

select p.product\_id,

round(cast(sum(p.price\*us.units) as decimal)/sum(us.units),2)

from prices as p, UnitsSold as us

where us.purchase\_date between p.start\_date and p.end\_date

and p.product\_id = us.product\_id

group by p.product\_id

**Q.41**

select name as warehouse\_name, sum(units \* vol) as volume

from Warehouse w

join (select product\_id, Width\*Length\*Height as vol

from Products) p

on w.product\_id = p.product\_id

group by name;

**Q.42**

select s.sale\_date,

sum(if(s.fruit = 'apples', s.sold\_num, -s.sold\_num)) as diff

from Sales s

group by s.sale\_date;

**Q.43**

create table Activity(player\_id int, device\_id int, event\_date date, games\_played int);

insert into Activity values(1,2,'2016-03-01', 5);

insert into Activity values(1,2,'2016-05-02', 6);

insert into Activity values(2,3,'2017-06-25', 1);

insert into Activity values(3,1,'2016-03-02', 0);

insert into Activity values(3,4,'2018-07-03', 5);

create table Result as

SELECT

player\_id,

min(event\_date) AS first\_login

FROM

Activity

GROUP BY player\_id;

Select \* from Result

**Q.44**

ABLE students

id INTEGER PRIMARY KEY,

firstName VARCHAR(30) NOT NULL,

lastName VARCHAR(30) NOT NULL\*/

select count(\*)

from students

where firstName = 'John'

**Q.45**

SELECT dept\_name, COUNT(student\_id) student\_number

FROM Department d LEFT JOIN Student s

ON s.dept\_id = d.dept\_id

GROUP BY d.dept\_id

ORDER BY student\_number DESC, dept\_name;

**Q.46**

create table Customer(customer\_id int, product\_key int);

insert into Customer(customer\_id,product\_key)

values

(1,5),

(2,6),

(3,5),

(3,6),

(1,6);

create table Product(product\_key int);

insert into Product(product\_key)

values

(5),

(6);

select a.customer\_id from

(select customer\_id, count(distinct product\_key) as num

from Customer

group by customer\_id) a

where a.num = (select count(distinct product\_key) from Product);

**Q.47**

select p.project\_id

from Project p

group by p.project\_id

having count(p.employee\_id) =

(select max(p1.employee\_cnt)

from (select project\_id, count(employee\_id) as employee\_cnt

from Project

group by project\_id) p1

);

**Q.48**

select book\_id, name

from books

where book\_id not in (

select book\_id

from orders

where (dispatch\_date between date\_sub('2019-06-23',interval 1 year) and '2019-06-23')

group by (book\_id)

having sum(quantity) >= 10)

and

available\_from < date\_sub('2019-06-23', interval 1 month);

**Q.49**

select e1.student\_id, min(e1.course\_id) as course\_id, e1.grade

from Enrollments e1

where e1.grade =

(select max(grade) as max\_grade

from Enrollments e2

where e1.student\_id = e2.student\_id)

group by e1.student\_id

order by e1.student\_id;

**Q.50**

select t.team\_id, t.team\_name,

if null(sum(case when t.team\_id = m.host\_team and m.host\_goals > m.guest\_goals then 3

when t.team\_id = m.host\_team and m.host\_goals = m.guest\_goals then 1

when t.team\_id = m.guest\_team and m.host\_goals < m.guest\_goals then 3

when t.team\_id = m.guest\_team and m.host\_goals = m.guest\_goals then 1

else 0 end), 0) as num\_points

from Matches m

right join Teams t

on m.host\_team = t.team\_id or m.guest\_team = t.team\_id

group by team\_id, team\_name

order by num\_points desc, team\_id;