

# SQL queries

Q1

The screenshot shows the HackerRank interface for the problem 'Weather Observation Station 7'. The problem description asks for a query to list city names ending with vowels (a, e, i, o, u) from the STATION table, with no duplicates. The input format specifies the STATION table structure:

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

Below the table, it notes that LAT\_N is the northern latitude and LONG\_W is the western longitude. The right side of the interface shows a code editor with a MySQL dropdown and a 'Submit Code' button.

answer:

```
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";
```

Q2



HackerRank

[Prepare](#)
[SQL](#)
[Basic Select](#)
[Revising the Select Query II](#)

Exit Full Screen View

Problem

Submissions

Leaderboard

Discussions

Editorial

Query the **NAME** field for all American cities in the **CITY** table with populations larger than 120000. The CountryCode for America is USA.

The **CITY** table is described as follows:

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

DB2

1

Line: 1 Col: 1

Upload Code as File

Run Code

Submit Code

Processing...

answer:

```

SELECT NAME
FROM CITY
WHERE COUNTRYCODE= "USA" AND POPULATION >= 120000

```

Leetcode sql challenge:

Q1:

SQL 50 < > Run Submit 0 Premium

Description Editorial Solutions Submissions

### 1757. Recyclable and Low Fat Products

Easy Topics Companies

SQL Schema > Pandas Schema >

Table: Products

Column Name	Type
product_id	int
low_fats	enum
recyclable	enum

product\_id is the primary key (column with unique values) for this table.  
low\_fats is an ENUM (category) of type ('Y', 'N') where 'Y' means this product is low fat and 'N' means it is not.  
recyclable is an ENUM (category) of types ('Y', 'N') where 'Y' means this product is recyclable and 'N' means it is not.

Write a solution to find the ids of products that are both low fat and recyclable.  
Return the result table in **any order**.  
The result format is in the following example.

**Example 1:**

**Input:**  
Products table:

product_id	low_fats	recyclable
0	Y	N
1	Y	Y
2	N	Y
3	Y	Y
4	N	N

Code

MySQL Auto

```
1 # Write your MySQL query statement below
2 select product_id
3 from Products
4 where low_fats='Y' and recyclable='Y';
```

Saved Ln 4, Col 39

Testcase Test Result

**Accepted** Runtime: 1038 ms

**Case 1**

Input

Products =

product_id	low_fats	recyclable
0	Y	N
1	Y	Y
2	N	Y
3	Y	Y
4	N	N

Output

product_id
------------

answer:

```
select product_id
from Products
where low_fats='Y' and recyclable='Y';
```

q2:

SQL 50

RunSubmit

Premium

DescriptionEditorialSolutionsSubmissions

584. Find Customer RefereeSolved

EasyTopicsCompaniesHint

SQL SchemaPandas Schema

Table: Customer

Column Name	Type
id	int
name	varchar
referee_id	int

In SQL, id is the primary key column for this table. Each row of this table indicates the id of a customer, their name, and the id of the customer who referred them.

Find the names of the customer that are **not referred by** the customer with `id = 2`.

Return the result table in **any order**.

The result format is in the following example.

**Example 1:**

**Input:**  
Customer table:

id	name	referee_id
1	Will	null

1.9K149

Code

MySQLAuto

1 # Write your MySQL query statement below

2

SavedLn 2, Col 1

TestcaseTest Result

You must run your code first

answer:

```
SELECT name
FROM Customer
WHERE referee_id IS NULL OR referee_id <> 2;
```

q3:

SQL 50

Run

Submit

0

Premium

Description

Editorial

Solutions

Submissions

### 595. Big Countries

Easy Topics Companies

SQL Schema Pandas Schema

Table: World

Column Name	Type
name	varchar
continent	varchar
area	int
population	int
gdp	bigint

name is the primary key (column with unique values) for this table. Each row of this table gives information about the name of a country, the continent to which it belongs, its area, the population, and its GDP value.

A country is **big** if:

- it has an area of at least three million (i.e., 3000000 km<sup>2</sup>), or
- it has a population of at least twenty-five million (i.e., 25000000).

Write a solution to find the name, population, and area of the **big countries**.

Return the result table in **any order**.

The result format is in the following example.

2.6K

169

Code

MySQL Auto

```

1 select name, population, area
2 from World
3 where area>=3000000 or population>=25000000;

```

Saved

Ln 3, Col 36

Testcase

Test Result

Accepted Runtime: 307 ms

Case 1

Input

World =

name	continent	area	population	gdp
Afghanistan	Asia	652230	25500100	20343000000
Albania	Europe	28748	2831741	12960000000
Algeria	Africa	2381741	37100000	188681000000
Andorra	Europe	468	78115	3712000000
Angola	Africa	1246700	20609294	100990000000

Output

name	population	area
------	------------	------

answer:

```

select name, population, area
from World
where area>=3000000 or population>=25000000;

```

question 4:

The screenshot shows a SQL problem interface. On the left, the problem description for '1148. Article Views I' is displayed. It includes a table schema for 'Views' with columns: article\_id (int), author\_id (int), viewer\_id (int), and view\_date (date). A note states: 'There is no primary key (column with unique values) for this table, the table may have duplicate rows. Each row of this table indicates that some viewer viewed an article (written by some author) on some date. Note that equal author\_id and viewer\_id indicate the same person.' The task is to 'Write a solution to find all the authors that viewed at least one of their own articles. Return the result table sorted by id in ascending order. The result format is in the following example.' An example input table is shown with 7 rows of data.

Column Name	Type
article_id	int
author_id	int
viewer_id	int
view_date	date

There is no primary key (column with unique values) for this table, the table may have duplicate rows.  
Each row of this table indicates that some viewer viewed an article (written by some author) on some date.  
Note that equal author\_id and viewer\_id indicate the same person.

Write a solution to find all the authors that viewed at least one of their own articles.  
Return the result table sorted by `id` in ascending order.  
The result format is in the following example.

**Example 1:**

**Input:**  
Views table:

article_id	author_id	viewer_id	view_date
1	3	5	2019-08-01
1	3	6	2019-08-02
2	7	7	2019-08-01
2	7	6	2019-08-02
4	7	1	2019-07-22
3	4	4	2019-07-21

On the right, the SQL query is entered in a text editor:

```
1 select distinct author_id as id
2 from Views
3 where author_id=viewer_id
4 order by id;
```

Below the query editor, the test result is shown as 'Accepted' with a runtime of 158 ms. The input data is repeated in a table format.

article_id	author_id	viewer_id	view_date
1	3	5	2019-08-01
1	3	6	2019-08-02
2	7	7	2019-08-01
2	7	6	2019-08-02
4	7	1	2019-07-22
3	4	4	2019-07-21

answer:

```
select distinct author_id as id
from Views
where author_id=viewer_id
order by id;
```

question 5:

**1683. Invalid Tweets**

Easy | Topics | Companies

SQL Schema | Pandas Schema

Table: Tweets

Column Name	Type
tweet_id	int
content	varchar

tweet\_id is the primary key (column with unique values) for this table. This table contains all the tweets in a social media app.

Write a solution to find the IDs of the invalid tweets. The tweet is invalid if the number of characters used in the content of the tweet is **strictly greater** than 15.

Return the result table in **any order**.

The result format is in the following example.

**Example 1:**

**Input:**

Tweets table:

tweet_id	content
1	Vote for Biden
2	Let us make America great again!

848 | 108 | ☆ | ✎ | ⌂

MySQL | Auto

```
1 select tweet_id
2 from Tweets
3 where length(content)>15;
```

Saved | Ln 3, Col 27

Testcase | Test Result

tweet_id	content
1	Vote for Biden
2	Let us make America great again!

Output

tweet_id
2

Expected

tweet_id
2

♥ Contribute a testcase

answer:

```
select tweet_id
from Tweets
where length(content)>15;
```

question 6:



SQL 50 < > Run Submit 0 Premium

Description | Editorial | Solutions | Submissions

### 1378. Replace Employee ID With The Unique Identifier Solved

Easy | Topics | Companies

SQL Schema > Pandas Schema >

Table: Employees

Column Name	Type
id	int
name	varchar

id is the primary key (column with unique values) for this table.  
Each row of this table contains the id and the name of an employee in a company.

Table: EmployeeUNI

Column Name	Type
id	int
unique_id	int

(id, unique\_id) is the primary key (combination of columns with unique values) for this table.  
Each row of this table contains the id and the corresponding unique id of an employee in the company.

Write a solution to show the **unique ID** of each user, If a user does not have a unique ID replace just show `null`.

Return the result table in any order.

1.1K 68

Code

```
MySQL Auto
1 select unique_id, name
2 from Employees
3 left join EmployeeUNI on Employees.id=EmployeeUNI.id;
```

Saved Ln 3, Col 52

Testcase Test Result

**Accepted** Runtime: 264 ms

Case 1

Input

Employees =

id	name
1	Alice
7	Bob
11	Meir
90	Winston
3	Jonathan

EmployeeUNI =

id	unique_id
----	-----------

answer:

```
select unique_id, name
from Employees
left join EmployeeUNI on Employees.id=EmployeeUNI.id;
```

q7:

SQL 50 < > ⚙️

Run Submit

Press F11 to exit full screen

### 1068. Product Sales Analysis I

Easy Topics Companies

SQL Schema > Pandas Schema >

Table: Sales

Column Name	Type
sale_id	int
product_id	int
year	int
quantity	int
price	int

(sale\_id, year) is the primary key (combination of columns with unique values) of this table.  
product\_id is a foreign key (reference column) to Product table.  
Each row of this table shows a sale on the product product\_id in a certain year.  
Note that the price is per unit.

Table: Product

Column Name	Type
product_id	int
product_name	varchar

product\_id is the primary key (column with unique values) of this table.  
Each row of this table indicates the product name of each product.

MySQL Auto

```
1 |
```

Saved Ln 1, Col 1

Testcase Test Result

Case 1 +

Sales =

sale_id	product_id	year	quantity	price
1	100	2008	10	5000
2	100	2009	12	5000
7	200	2011	15	9000

Product =

product_id	product_name
100	Nokia
200	Apple
300	Samsung

768 70

answer:

```
select product_name, year, price
from Sales inner join Product on Sales.product_id=Product.product_id
```

q8:

The screenshot shows a SQL editor interface with the following components:

- Table: Visits**

Column Name	Type
visit_id	int
customer_id	int

visit\_id is the column with unique values for this table.  
This table contains information about the customers who visited the mall.
- Table: Transactions**

Column Name	Type
transaction_id	int
visit_id	int
amount	int

transaction\_id is column with unique values for this table.  
This table contains information about the transactions made during the visit\_id.
- SQL Query**

```
1 SELECT v.customer_id, COUNT(v.visit_id) AS count_no_trans
2 FROM Visits v
3 LEFT JOIN Transactions t ON v.visit_id = t.visit_id
4 WHERE t.visit_id IS NULL
5 GROUP BY v.customer_id;
6
```
- Test Result**

Accepted Runtime: 278 ms

Case 1

Input

visit_id	customer_id
1	23
2	9
4	30
5	54
6	96
7	54

View more

answer:

```
SELECT v.customer_id, COUNT(v.visit_id) AS count_no_trans
FROM Visits v
LEFT JOIN Transactions t ON v.visit_id = t.visit_id
WHERE t.visit_id IS NULL
GROUP BY v.customer_id;
```

q9:

SQL 50

Run

Submit

Premium

Description

Editorial

Solutions

Submissions

### 197. Rising Temperature

Easy Topics Companies

SQL Schema Pandas Schema

Table: Weather

Column Name	Type
id	int
recordDate	date
temperature	int

id is the column with unique values for this table.  
There are no different rows with the same recordDate.  
This table contains information about the temperature on a certain day.

Write a solution to find all dates' id with higher temperatures compared to its previous dates (yesterday).

Return the result table in **any order**.

The result format is in the following example.

**Example 1:**

**Input:**  
Weather table:

id	recordDate	temperature
1	2015-01-01	10

Code

MySQL Auto

```

1 SELECT y.id
2 FROM Weather x
3 LEFT JOIN Weather y ON x.Id +1 = y.Id
4 WHERE x.Temperature < y.Temperature

```

Saved Ln 4, Col 36

Testcase

Test Result

**Accepted** Runtime: 178 ms

Case 1

Input

Weather -

id	recordDate	temperature
1	2015-01-01	10
2	2015-01-02	25
3	2015-01-03	20
4	2015-01-04	30

Output

id
2

answer:

```

SELECT w.id
FROM Weather w
JOIN Weather w_prev ON w.recordDate = DATE_ADD(w_prev.recordDate, INTERVAL 1 DAY)
WHERE w.temperature > w_prev.temperature;

```

q10:

SQL 50

Run

Submit

Premium

Description

Editorial

Solutions

Submissions

### 1661. Average Time of Process per Machine

Easy Topics Companies

SQL Schema > Pandas Schema >

Table: Activity

Column Name	Type
machine_id	int
process_id	int
activity_type	enum
timestamp	float

The table shows the user activities for a factory website. (machine\_id, process\_id, activity\_type) is the primary key (combination of columns with unique values) of this table. machine\_id is the ID of a machine. process\_id is the ID of a process running on the machine with ID machine\_id. activity\_type is an ENUM (category) of type ('start', 'end'). timestamp is a float representing the current time in seconds. 'start' means the machine starts the process at the given timestamp and 'end' means the machine ends the process at the given timestamp. The 'start' timestamp will always be before the 'end' timestamp for every (machine\_id, process\_id) pair.

There is a factory website that has several machines each running the **same number of processes**. Write a solution to find the **average time** each machine takes to complete a process.

The time to complete a process is the 'end' timestamp minus the 'start' timestamp. The average time is calculated by the total time to complete every process on the machine divided by the number of processes that were run.

Code

MySQL

Auto

```

9      MAX(CASE WHEN activity_type = 'end' THEN timestamp END) AS end_time
10  FROM
11      Activity
12  GROUP BY
13      machine_id,
14      process_id
15  ) AS process_times
16  GROUP BY
17      machine_id;
18

```

Testcase

Test Result

Accepted Runtime: 156 ms

Case 1

Input

machine_id	process_id	activity_type	timestamp
0	0	start	0.712
0	0	end	1.52
0	1	start	3.14
0	1	end	4.12
1	0	start	0.55
1	0	end	1.55

View more

answer:

```

SELECT
    machine_id,
    ROUND(AVG(end_time - start_time), 3) AS processing_time
FROM (
    SELECT
        machine_id,
        process_id,
        MAX(CASE WHEN activity_type = 'start' THEN timestamp
        MAX(CASE WHEN activity_type = 'end' THEN timestamp EN
    FROM
        Activity
    GROUP BY
        machine_id,
        process_id
) AS process_times

```

```
GROUP BY  
    machine_id;
```

q11:

**577. Employee Bonus** Solved

Easy Topics Companies Hint

SQL Schema Pandas Schema

Table: Employee

Column Name	Type
empId	int
name	varchar
supervisor	int
salary	int

empId is the column with unique values for this table.  
Each row of this table indicates the name and the ID of an employee in addition to their salary and the id of their manager.

Table: Bonus

Column Name	Type
empId	int
bonus	int

```
1  
2  
3 SELECT name, bonus  
4 FROM Employee  
5 LEFT JOIN Bonus ON Employee.empId = Bonus.empId  
6 WHERE bonus < 1000 OR bonus IS NULL;  
7
```

Testcase Test Result

Brad	null	
John	null	
Dan	500	

Expected

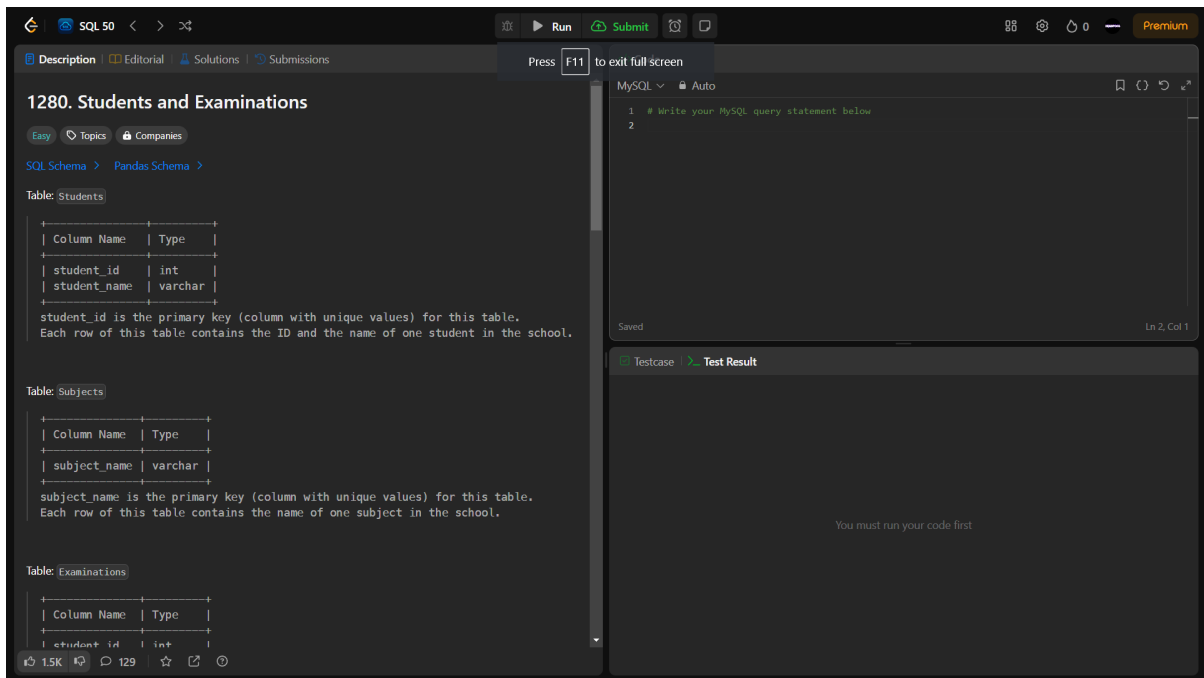
name	bonus
Brad	null
John	null
Dan	500

Contribute a testcase

answer:

```
SELECT name, bonus  
FROM Employee  
LEFT JOIN Bonus ON Employee.empId = Bonus.empId  
WHERE bonus < 1000 OR bonus IS NULL;
```

q12:



answer:

```
SELECT
Students.student_id,
Students.student_name,
Subjects.subject_name,
COUNT(Examinations.subject_name) AS attended_exams
FROM
Students
CROSS JOIN
Subjects
LEFT JOIN
Examinations ON Students.student_id = Examinations.student_id
AND Subjects.subject_name = Examinations.subject_name
GROUP BY
Students.student_id,
Students.student_name,
Subjects.subject_name
ORDER BY
```

Students.student\_id,  
Subjects.subject\_name;

q12:

The screenshot shows a SQL editor interface with a dark theme. On the left, the 'Input' section displays an 'Employee' table with the following data:

id	name	department	managerId
101	John	A	null
102	Dan	A	101
103	James	A	101
104	Amy	A	101
105	Anne	A	101
106	Ron	B	101

Below the table, the 'Output' section shows a single row with the name 'John'. The right side of the editor contains a SQL query:

```
1  
2 SELECT e1.name  
3 FROM Employee e1  
4 JOIN (  
5     SELECT managerId, COUNT(*) AS direct_reports  
6     FROM Employee  
7     GROUP BY managerId  
8 ) e2 ON e1.id = e2.managerId  
9 WHERE e2.direct_reports >= 5;  
10
```

At the bottom, there is a 'Testcase' section with a 'Test Result' table that is currently empty.

answer:

```
SELECT e1.name  
FROM Employee e1  
JOIN (  
    SELECT managerId, COUNT(*) AS direct_reports  
    FROM Employee  
    GROUP BY managerId  
) e2 ON e1.id = e2.managerId  
WHERE e2.direct_reports >= 5;
```