--Classes More Than 5 Students

--There is a table courses with columns: student and class

--Please list out all classes which have more than or equal to 5 students.

--For example, the table:

--

--+---------+------------+

--| student | class |

--+---------+------------+

--| A | Math |

--| B | English |

--| C | Math |

--| D | Biology |

--| E | Math |

--| F | Computer |

--| G | Math |

--| H | Math |

--| I | Math |

--+---------+------------+

--Should output:

--

--+---------+

--| class |

--+---------+

--| Math |

--+---------+

CREATE TABLE Class\_HW (student VARCHAR(30), class VARCHAR(30));

INSERT INTO Class\_HW

VALUES

('A', 'Math'),

('B', 'English'),

('C', 'Math'),

('D', 'Biology'),

('E', 'Math'),

('F', 'Computer'),

('G', 'Math'),

('H', 'Math'),

('I', 'Math');

select class from class\_hw

group by class

having count(class) >=5;

-- Write a SQL query to find all duplicate emails in a table named Person.

-- https://leetcode.com/problems/duplicate-emails/

-- input

--+----+---------+

--| Id | Email |

--+----+---------+

--| 1 | a@b.com |

--| 2 | c@d.com |

--| 3 | a@b.com |

--+----+---------+

-- output

--+---------+

--| Email |

--+---------+

--| a@b.com |

--+---------+

CREATE TABLE Email\_HW1 (Id INT, Email VARCHAR(30));

INSERT INTO Email\_HW1

VALUES

(1, 'a@b.com'),

(2, 'c@d.com'),

(3, 'a@b.com');

select distinct(email) from email\_hw1

group by email

having count(\*) > 1;

-- Customers Who Never Order

-- https://leetcode.com/problems/customers-who-never-order/

--Suppose that a website contains two tables, the Customers table and the Orders table.

--Write a SQL query to find all customers who never order anything.

--Table: Customers.

--+----+-------+

--| Id | Name |

--+----+-------+

--| 1 | Joe |

--| 2 | Henry |

--| 3 | Sam |

--| 4 | Max |

--+----+-------+

--Table: Orders.

--+----+------------+

--| Id | CustomerId |

--+----+------------+

--| 1 | 3 |

--| 2 | 1 |

--+----+------------+

--Using the above tables as example, return the following:

--+-----------+

--| Customers |

--+-----------+

--| Henry |

--| Max |

--+-----------+

CREATE TABLE Customers\_HW (Id INT, Name VARCHAR(30));

INSERT INTO Customers\_HW

VALUES

(1, 'Joe'),

(2, 'Henry'),

(3, 'Sam'),

(4, 'Max');

CREATE TABLE Orders\_HW (Id INT, CustomerId INT);

INSERT INTO Orders\_HW

VALUES

(1, 3),

(2, 1);

select Name as Customers from customers\_hw

where Id not in (select CustomerId from orders\_hw);

--Write a SQL query to delete all duplicate email entries in a table named Person, keeping only unique emails based on its smallest Id.

--+----+------------------+

--| Id | Email |

--+----+------------------+

--| 1 | john@example.com |

--| 2 | bob@example.com |

--| 3 | john@example.com |

--+----+------------------+

-- output

--+----+------------------+

--| Id | Email |

--+----+------------------+

--| 1 | john@example.com |

--| 2 | bob@example.com |

--+----+------------------+

CREATE TABLE Email\_HW (Id INT, Email VARCHAR(30));

INSERT INTO Email\_HW

VALUES

(1, 'john@example.com'),

(2, 'bob@example.com'),

(3, 'john@example.com');

select min(Id) as Id,Email from email\_hw

group by Email;

-- find the Nth highest salary

CREATE TABLE Employee\_HW (Id INT, Salary INT);

INSERT INTO Employee\_HW

VALUES

(1, 1000),

(2, 2000),

(3, 3000),

(4, 4000),

(5, 5000);

-- N = 5

-- N = 2

select Id,Salary from employee\_hw

order by Salary DESC

limit 4,1;

select Id,Salary from employee\_hw

order by Salary DESC

limit 1,1;

-- find who has the most num of transaction records.

CREATE TABLE Transaction\_HW

(

id INT DEFAULT NULL,

num\_trans INT DEFAULT NULL,

dt VARCHAR(30) DEFAULT NULL

);

INSERT INTO Transaction\_HW (id, num)

VALUES

(1, 3, '20190501'),

(1, 2, '20190502'),

(2, 1, '20190501'),

(2, 4, '20190502'),

(3, 2, '20190501')

;

-- should output

--+----+------------------+

--| Id |

--| 1 |

--| 2 |

--+----+------------------+

select id from transaction\_hw

group by id

having count(id) = (select max(s.count) from (

select id,count(\*) as count from transaction\_hw

group by id) s);

--Mary is a teacher in a middle school and she has a table seat storing students' names and their corresponding seat ids.

--The column id is continuous increment.

--Mary wants to change seats for the adjacent students.

--Can you write a SQL query to output the result for Mary?

-- ORIGINAL

--+---------+---------+

--| id | student |

--+---------+---------+

--| 1 | Abbot |

--| 2 | Doris |

--| 3 | Emerson |

--| 4 | Green |

--| 5 | Jeames |

--+---------+---------+

-- NEW

--+---------+---------+

--| id | student |

--+---------+---------+

--| 1 | Doris |

--| 2 | Abbot |

--| 3 | Green |

--| 4 | Emerson |

--| 5 | Jeames |

--+---------+---------+

CREATE TABLE Seat\_HW (Id INT, Student VARCHAR(30));

INSERT INTO Seat\_HW

VALUES

(1, 'Abbot'),

(2, 'Doris'),

(3, 'Emerson'),

(4, 'Green'),

(5, 'Jeames');

SELECT

(CASE

WHEN a.id%2 != 0 AND counts != a.id THEN a.id + 1

WHEN a.id%2 != 0 AND counts = a.id THEN a.id

ELSE a.id - 1

END) AS id,

a.student

FROM

(SELECT \*, COUNT(\*) OVER() AS counts

FROM Seat\_HW) AS a

ORDER BY id ASC;

-- table friend: tb1 = [user\_id , action, target\_id , date].

-- Action in {‘sent’, ‘accept’}. Calculate acceptance rate for all the requests sent in April.

-- Assumption: (a). each user can only send one request to a target user.

-- (b) each user can only accept the request from the same sender once.

-- (c) if A send request to B, B will have to accepted to be considered in our calculation.

-- Groupon interview question:

-- For every user, select the most recent two transection.

-- Table=[user\_id, trans\_id, time]