**Classic Questions :**

**Write a SQL query to get the third highest salary of an employee from employee\_table?**

|  |  |
| --- | --- |
|  | SELECT TOP 1 salary  FROM(  SELECT TOP 3 salary  FROM employee\_table  ORDER BY salary DESC) AS emp  ORDER BY salary ASC; |

**175. Combine Two Tables**

Easy

SQL Schema

Table: Person

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

| Column Name | Type |

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

| PersonId | int |

| FirstName | varchar |

| LastName | varchar |

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

PersonId is the primary key column for this table.

Table: Address

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

| Column Name | Type |

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

| AddressId | int |

| PersonId | int |

| City | varchar |

| State | varchar |

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

AddressId is the primary key column for this table.

Write a SQL query for a report that provides the following information for each person in the Person table**, regardless if there is an address for each of those people**:

FirstName, LastName, City, State

**My solution:**

**First try: wrong answer**. I used Inner join. It won’t return any row if the address values are null

**Explanation:**

Simple inner join would ignore rows from person table where it couldn't find any corresponding record/personId in the address table. Choose left outer join, Person as the left table so it would ignore if right table Address is null or not .

SELECT FirstName, LastName, City, State

FROM Person p

LEFT JOIN Address a

ON p.PersonId = a.PersonId;

**176. Second Highest Salary**

Easy

SQL Schema

Write a SQL query to get the second highest salary from the Employee table.

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

| Id | Salary |

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

| 1 | 100 |

| 2 | 200 |

| 3 | 300 |

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

For example, given the above Employee table, the query should return 200 as the second highest salary. If there is no second highest salary, then the query should return null.

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

| SecondHighestSalary |

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

| 200 |

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

**My solution:**

**First Try: Wrong answer,** does not return anything if second rank value is null ( no second ranking value)

SELECT MIN(T1.Salary) AS SecondHighestSalary

FROM(

SELECT Salary

FROM Employee

ORDER BY Salary DESC # order by salary not SecondHighestSalary

LIMIT 2) T1 ;

**Notes:**

# 1. mySQL cannot create a separate table or the syntax in PostgreSQL cannot use here

# 2. You must give an alias for derived table. The outer query would use the derived table name : **T1.Salary**.

Select the max from the smaller ones is better than select the min from the top 2 ( coz latter null won’t be return )

**Correct Solution: consider null**

Using max() will return a NULL if the value doesn't exist

Two steps: one is to create a table with all the salary values less than MAX salalry.

Second is to get the max of the sub table

SELECT MAX(Salary) AS SecondHighestSalary

FROM Employee

WHERE Salary <

(

SELECT MAX(Salary)

FROM Employee

) ;

**Notes:**

1. The inner query returns a value instead of a table, so no alias is needed
2. Use sub query in condition filter

**Using the OFF SET in the LIMIT query**

The **OFF SET** value is also most often used together with the LIMIT keyword. The OFF SET value allows us to specify which row to start from retrieving data , Limit will specify how many records will return

This function takes two arguments. If the first argument is not NULL, the function returns the first argument. Otherwise, the second argument is returned. This function is commonly used to replace NULL value with another value. It is similar to the [**NVL function**](https://www.1keydata.com/sql/sql-nvl.html) in Oracle and the [**ISNULL Function**](https://www.1keydata.com/sql/sql-isnull.html) in SQL Server.

**Method 2: IFNULL() function only available in MySQL**

Select IFNULL((Select Distinct Salary from Employee Order by Salary Desc LIMIT 1,1), NULL) AS SecondHighestSalary

# LIMIT 1,1: start from the SECOND row and return just one record

# SELECT DISTINCT is a must . it would return null if there is null

**Method 3:**

SELECT

CASE WHEN COUNT(DISTINCT(Salary)) <= 1 THEN null

ELSE (select salary from employee order by salary desc limit 1, 1)

END AS SecondHighestSalary

FROM Employee

**181. Employees Earning More Than Their Managers**

Easy

SQL Schema

The Employee table holds all employees including their managers. Every employee has an Id, and there is also a column for the manager Id.

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

| Id | Name | Salary | ManagerId |

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

| 1 | Joe | 70000 | 3 |

| 2 | Henry | 80000 | 4 |

| 3 | Sam | 60000 | NULL |

| 4 | Max | 90000 | NULL |

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

Given the Employee table, write a SQL query that finds out employees who earn more than their managers. For the above table, Joe is the only employee who earns more than his manager.

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

| Employee |

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

| Joe |

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

**My solution: Wrong attempt, my solution gives the highest manager salary instead of the employee salary**

**Notes:**

1. MySQL use key word “ join “ instead of “ self join “ for self join statement
2. Here should have two conditions:E.Id = M.ManagerId and E.Salary > M.Salary , just use latter won’t give the result you want.

SELECT E.Name # here specify the name in which table

FROM Employee E

JOIN Employee M

ON E.Id = M.ManagerId

WHERE E.Id = M.ManagerId AND E.Salary > M.Salary; # consider null

**Correct Answer:**

SELECT E.Name AS Employee # specify E.Name

FROM Employee E

JOIN Employee M

ON M.Id = E.ManagerId

WHERE E.Salary > M.Salary;

**182. Duplicate Emails ( self join )**

Easy

SQL Schema

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

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

| Id | Email |

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

| 1 | a@b.com |

| 2 | c@d.com |

| 3 | a@b.com |

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

For example, your query should return the following for the above table:

+---------+

| Email |

+---------+

| a@b.com |

+---------+

**Note**: All emails are in lowercase.

**My Solution: Wrong attempt**

# First, group the emails

# second find the emails with count greater than 1, ids are different, but emails are same

# use “as” for aggregation alias / derived table

# specify which table the attribute belong to

**Correct Answer:**

**Method 2: brilliant, do not need join**

SELECT Email

FROM Person

GROUP BY Email # group by is very helpful and powerful

HAVING COUNT(Email) > 1 # here must use HAVING for aggregation

**Method 1: inner query group same emails and count them, outer query set condition to filter emails unique , just keep duplicate emails**

SELECT T1.Email

FROM

( SELECT p.Email, COUNT(p.Email) AS email\_count

FROM Person p

GROUP BY p.Email) AS T1

WHERE T1.email\_count > 1;

**Method 3: Self join**

SELECT Email

FROM Person P1, Person P2

WHERE P1.Id >< P2.Id AND P1.Email = P2.Email

**196. Delete Duplicate Emails ( self-join )**

Easy

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 |

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

Id is the primary key column for this table.

For example, after running your query, the above Person table should have the following rows:

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

| Id | Email |

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

| 1 | john@example.com |

| 2 | bob@example.com |

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

**Note:**

Your output is the whole Person table after executing your sql. Use deletestatement.

**DELETE syntax**

**DELETE**

**FROM**

**WHERE**

Analysis:

Step 1 : SELECT all the min id after grouping by emails

Step 2: select all from the result of first step

Step 3: delete all rows if the id not in id list

# alias is used with select statement for table & column & aggregation function not a set of values

**# Method 1 : SELF JOIN**

**limitation:** it won’t order by ID if it is not the default column to order by

DELETE P2.\*

FROM Person AS P1, Person AS P2

WHERE P1.Id < P2.Id AND P1.Email = P2.Email; # it keeps the lowest ID

**# method 2 :**

DELETE # just need delete, it will delete \* meeting the condition

FROM Person

WHERE Id NOT IN ( SELECT \*

FROM (SELECT MIN(Id) AS min\_id

# for aggregation, use alias is better

FROM Person

GROUP BY Email

# get all smallest Ids in all duplicated email groups

) TEMP

# select all min Ids

);

# “in” and “not in “expression do not need an alias here # delete all the rest

**183. Customers Who Never Order**

Easy

SQL Schema

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 |

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

**My solution: Wrong attempt. This one should be easy. Just check those IDs not in CustomerID**

**Notes: do not need a join here. Second table is for building condition filtering purpose.**

SELECT C.Name # column alias

From Customers C

JOIN Orders O

ON C.Id = O.CustomerId

WHERE C.Id NOT IN ( SELECT O.CustomerId

FROM Orders O # donot forget give alias O

) ;

**Correct version:**

**Solution 1:**

SELECT Name AS Customers

# check question, what is the alias name

From Customers

WHERE Id NOT IN ( SELECT CustomerId FROM Orders ) ;

# sub query return a list , use “IN” or “ NOT IN “

**Solution 2:**

SELECT C.Name AS Customers

FROM Customers C

LEFT JOIN Orders O

ON C.Id = O.CustomerId

WHERE O.CustomerId IS NULL;

**# O.CustomerId IS NULL not C.Id IS NULL**

**197. Rising Temperature**

Easy

SQL Schema

Given a Weather table, write a SQL query to find all dates' Ids with higher temperature compared to its previous (yesterday's) dates.

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

| Id(INT) | RecordDate(DATE) | Temperature(INT) |

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

| 1 | 2015-01-01 | 10 |

| 2 | 2015-01-02 | 25 |

| 3 | 2015-01-03 | 20 |

| 4 | 2015-01-04 | 30 |

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

For example, return the following Ids for the above Weather table:

+----+

| Id |

+----+

| 2 |

| 4 |

+----+

**Correct Answer:** TO\_DAYS is a function to return days from year 0

**Analysis: we need a special function TO\_DAYS() to convert date to days**

**Self Join to create a yesterday table and today table**

**Conditions : days and temperature**

SELECT W\_today.Id

FROM Weather W\_yesterday

JOIN Weather W\_today

ON TO\_DAYS(W\_yesterday.RecordDate) + 1 = TO\_DAYS(W\_today.RecordDate)

WHERE W\_yesterday.Temperature < W\_today.Temperature ;

**OR**

SELECT W\_today.Id

FROM Weather W\_yesterday, Weather W\_today

WHERE W\_yesterday.Temperature < W\_today.Temperature AND

TO\_DAYS(W\_yesterday.RecordDate) + 1 = TO\_DAYS(W\_today.RecordDate);

**595. Big Countries**

Easy

SQL Schema

There is a table World

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

| name | continent | area | population | gdp |

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

| Afghanistan | Asia | 652230 | 25500100 | 20343000 |

| Albania | Europe | 28748 | 2831741 | 12960000 |

| Algeria | Africa | 2381741 | 37100000 | 188681000 |

| Andorra | Europe | 468 | 78115 | 3712000 |

| Angola | Africa | 1246700 | 20609294 | 100990000 |

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

A country is big if it has an area of bigger than 3 million square km or a population of more than 25 million.

Write a SQL solution to output big countries' name, population and area.

For example, according to the above table, we should output:

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

| name | population | area |

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

| Afghanistan | 25500100 | 652230 |

| Algeria | 37100000 | 2381741 |

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

**My solution: Accepted**

**Method 1: OR**

SELECT name, population, area

FROM World

WHERE population > 25000000 OR area > 3000000 ;

# do not add comma as separator for these numbers , syntax error

**596. Classes More Than 5 Students**

Easy

SQL Schema

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 |

+---------+

**Note:**  
The students should not be counted duplicate in each course.

# method 2: group by + having

SELECT class

FROM courses

GROUP BY class

HAVING COUNT(**DISTINCT** student) >= 5 ;

# must use DISTINCT to avoid duplicated students( students take a class twice)

**620. Not Boring Movies**

Easy

SQL Schema

X city opened a new cinema, many people would like to go to this cinema. The cinema also gives out a poster indicating the movies’ ratings and descriptions.

Please write a SQL query to output movies with an odd numbered ID and a description that is not 'boring'. Order the result by rating.

For example, table cinema:

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

| id | movie | description | rating |

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

| 1 | War | great 3D | 8.9 |

| 2 | Science | fiction | 8.5 |

| 3 | irish | boring | 6.2 |

| 4 | Ice song | Fantacy | 8.6 |

| 5 | House card| Interesting| 9.1 |

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

For the example above, the output should be:

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

| id | movie | description | rating |

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

| 5 | House card| Interesting| 9.1 |

| 1 | War | great 3D | 8.9 |

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

**My Solution : Can not believe it is wrong. Pay attention to requirement. You need rank them desc**

SELECT \*

FROM cinema

WHERE id % 2 != 0 AND description != 'boring'

**# For remainder, use “ % ”**

**ORDER BY rating DESC;**

**# DESC**

**627. Swap Salary**

Easy

SQL Schema

Given a table salary, such as the one below, that has m=male and f=female values. Swap all f and m values (i.e., change all f values to m and vice versa) with a **single update statement** and no intermediate temp table.

Note that you must **write a single update statement, DO NOT write any select statement for this problem.**

**Example:**

| id | name | sex | salary |

|----|------|-----|--------|

| 1 | A | m | 2500 |

| 2 | B | f | 1500 |

| 3 | C | m | 5500 |

| 4 | D | f | 500 |

After running your **update** statement, the above salary table should have the following rows:

| id | name | sex | salary |

|----|------|-----|--------|

| 1 | A | f | 2500 |

| 2 | B | m | 1500 |

| 3 | C | f | 5500 |

| 4 | D | m | 500 |

**UPDATE syntax:**

**UPDATE table\_name**

**SET**

**WHERE**

**I did not figure out how to do this one**

**method 1:**

UPDATE salary

SET sex = IF (sex = “f” , ” m” , “f”) ) # IF should be outside of the parenthesis

**method 3 :**

UPDATE salary

SET

sex = CASE sex # “set = is” needed

WHEN 'm' THEN 'f'

ELSE 'm'

END ;