LEETCODE SQL

Second Highest Salary

```
select(
select distinct salary as SecondHighestSalary from Employee e1 where
2=(select count(distinct salary) from Employee e2 where e1.salary<=e2.salary order
by e1.salary)
) as SecondHighestSalary;
```

Employees Whose Manager Left the company

```
SELECT employee_id FROM Employees
WHERE manager_id NOT IN (SELECT employee_id FROM Employees) AND salary<30000
ORDER BY employee_id;
```

Exchange Seats

```
CASE WHEN id%2=1 AND id != (SELECT COUNT(id) FROM Seat) THEN id+1
WHEN id%2=0 THEN id-1
ELSE id
END AS id, student
FROM Seat
ORDER BY id;
```

Movie Rating

```
# Write your MySQL query statement below
(select name as results from movierating join users on
users.user_id=movierating.user_id
group by users.user_id
order by count(rating) desc, name limit 1)
union all
(select title as results from movierating join movies on
movierating.movie_id=movies.movie_id
where year(created_at)=2020 and month(created_at)=2
group by title
order by avg(rating) desc, title limit 1)
```

Recyclable and Low Fat products

```
SELECT product_id FROM Products WHERE low_fats='Y' AND recyclable='Y';
```

Managers With Atleast 5 Direct Reports

```
select A.name from Employee A join Employee B
ON A.id=B.managerId
GROUP BY A.id
HAVING COUNT(*)>=5;
```

Find Customer Referee

```
SELECT name from Customer WHERE referee_id != 2 OR referee_id IS NULL;
```

Confirmation Rate

```
SELECT Signups.user_id,ROUND(AVG(IF(Confirmations.action="confirmed",1,0)),2) AS
confirmation_rate
FROM Signups LEFT JOIN Confirmations ON Signups.user_id=Confirmations.user_id
GROUP BY signups.user_id;
```

Students And Examinations

```
SELECT
students.student_id,students.student_name,subjects.subject_name,count(examinations
.subject_name) AS attended_exams
from Students JOIN Subjects LEFT JOIN Examinations
ON students.student_id=examinations.student_id and
subjects.subject_name=examinations.subject_name
GROUP BY students.student_id,subjects.subject_name
ORDER BY student_id ASC,subjects.subject_name ASC;
```

Not Boring Movies

```
SELECT * FROM Cinema
WHERE id%2=1 AND description != 'boring'
ORDER BY Rating DESC;
```

Big Countries

```
SELECT name,population,area FROM World
WHERE area>=3000000 OR population>=25000000
ORDER BY name ASC;
```

Employee Bonus

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

Average Time Of Process Per Machine

```
SELECT a1.machine_id,ROUND(AVG(a2.timestamp-a1.timestamp),3) AS processing_time
FROM Activity a1 JOIN Activity a2
ON a1.machine_id=a2.machine_id AND a1.process_id=a2.process_id
AND a1.activity_type='start' AND a2.activity_type='end'
GROUP BY a1.machine_id;
```

Average Selling Price

```
select Prices.product_id,IFNULL(ROUND(SUM(units*price)/SUM(units),2),0) AS
average_price
FROM Prices LEFT JOIN UnitsSold
ON Prices.product_id=UnitsSold.product_id AND UnitsSold.purchase_date BETWEEN
start_date AND end_date
GROUP BY product_id;
```

Project Employees I

```
select project_id,round(avg(experience_years),2) as average_years from
Project JOIN Employee On Project.employee_id=employee.employee_id
GROUP BY project_id;
```

Percentage Of Users Attended A Contest:

```
Select contest_id,
round(count(distinct user_id) * 100 / (Select count(distinct user_id) from Users),
2) as percentage
from Register group by contest_id order by percentage desc, contest_id;
```

Article Views I

```
select distinct author_id as id FROM Views
WHERE author_id=viewer_id
ORDER BY 1;
```

Invalid Tweets

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

Replace Employee ID with the unique identifier

```
SELECT CASE WHEN unique_id IS NULL THEN NULL ELSE unique_id END as unique_id,name
FROM Employees LEFT JOIN EmployeeUNI
ON Employees.id=EmployeeUNI.id
```

Product Sales Analysis I

```
SELECT product_name,year,price
FROM Sales JOIN Product
ON Sales.product_id=Product.product_id;
```

Customer Who Visited But Did Not Make Any Transactions

```
SELECT v.customer_id,COUNT(v.visit_id) AS count_no_trans
FROM Visits v LEFT JOIN Transactions t
```

```sql
ON v.visit_id=t.visit_id
WHERE t.transaction_id IS NULL
GROUP BY v.customer_id;
```

Rising Temperature

```sql
SELECT B.id FROM Weather A JOIN Weather B On DATEDIFF(B.recordDate,A.recordDate)=1
WHERE B.temperature>A.temperature;
```

Queries Quality And Percentages

```sql
select query_name,
    round(avg(rating/position), 2) as quality,
    round(sum(if(rating < 3,1,0)) * 100 / count(*), 2) as poor_query_percentage
from Queries
where query_name is not null
group by query_name;
```