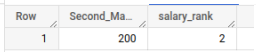
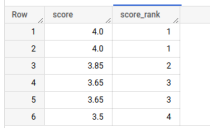
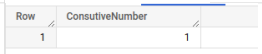
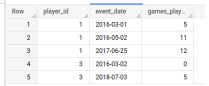
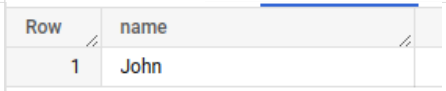
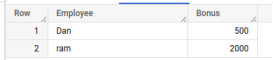
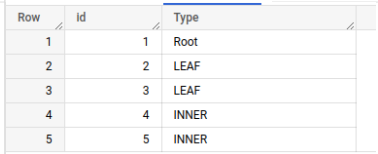
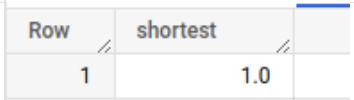
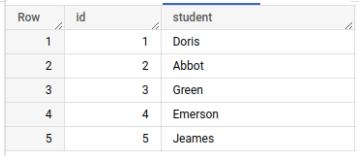
Question	Solution: Query	Solution: Screenshoot
176. Second Highest Salary Write a SQL query to get the second highest salary from the Employee table.	SELECT MAx(salary) AS Second_Maximum_Salary FROM `baburam-shrestha.MedQ.Table_176_Salary` WHERE salary NOT IN (SELECT MAX(Salary) FROM `baburam-shrestha.MedQ.Table_176_Salary`); SELECT salary AS Second_Maximum_Salary FROM `baburam-shrestha.MedQ.Table_176_Salary` ORDER BY salary DESC LIMIT 1 OFFSET 1;	
177. Nth Highest Salary Write a SQL query to get the nth highest salary from the Employee table.	SELECT * FROM (SELECT salary AS Second_Maximum_Salary, RANK() OVER(ORDER BY salary DESC) AS salary_rank FROM `baburam-shrestha.MedQ.Table_177_Salary`) AS X WHERE X.salary_rank=2;	
178. Rank Scores Write a SQL query to rank scores. If there is a tie between two scores, both should have the same ranking. Note that after a tie, the next ranking number should be the next consecutive integer value. In other words, there should be no "holes" between ranks.	SELECT * FROM (SELECT r.score, DENSE_RANK() OVER(ORDER BY score DESC) AS score_rank FROM `baburam-shrestha.MedQ.Table_178_Score` r) AS sc ORDER BY sc.score_rank;	
180. Consecutive Numbers Write a SQL query to find all numbers that appear at least three times consecutively.	SELECT DISTINCT a.number AS ConsutiveNumber FROM`baburam-shrestha.MedQ.Table_180_Number` a JOIN `baburam-shrestha.MedQ.Table_180_Number` b ON a.id = b.id +1 AND a.number = b.number JOIN `baburam-shrestha.MedQ.Table_180_Number` c ON a.id = c.id +2 AND a.number = c.number;	
534. Game Play Analysis III Write an SQL query that reports for each player and date, how many games played so far by the player. That is, the total number of games played by the player until that date.	SELECT player_id, event_date, sum(games_played) OVER(PARTITION BY player_id ORDER BY event_date) AS games_played_so_far FROM `baburam-shrestha.MedQ.Table_534_Player`;	
570. Managers with at Least 5 Direct Reports The Employee table holds all employees including their managers. Every employee has an Id, and there is also a column for the manager Id	SELECT m1.name FROM `baburam-shrestha.MedQ.Table_570_Manager` m1 INNER JOIN `baburam-shrestha.MedQ.Table_570_Manager` m2 ON m1.id = m2.manager_id GROUP BY m1.name HAVING COUNT(DISTINCT m2.id)>=5;	
577. Employee Bonus Select all employee's name and bonus whose bonus is < 1000.	SELECT e1.name AS Employee, e2.bonus as Bonus FROM `baburam-shrestha.MedQ.Table_577_Employee` e1 JOIN `baburam-shrestha.MedQ.Table_577_Bonus` e2 ON e1.emp_id = e2.emp_id;	
608. Tree Node Write a query to print the node id and the type of the node. Sort your output by the node id.	SELECT DISTINCT t1.id, CASE WHEN t1.p_id IS NULL THEN 'Root ' WHEN t1.id IS NOT NULL AND t2.p_id IS NOT NULL THEN 'INNER' ELSE 'LEAF' END AS Type FROM `baburam-shrestha.MedQ.Table_608_Tree` t1 LEFT JOIN `baburam-shrestha.MedQ.Table_608_Tree` t2 ON t1.p_id=t2.id;	
612. Shortest Distance in a Plane Write a query to find the shortest distance between these points rounded to 2 decimals.	SELECT ROUND(SQRT(MIN(POWER(p1.x-p2.x, 2) + POWER(p1.y-p2.y, 2))) , 2) AS shortest FROM `baburam-shrestha.MedQ.Table_612_xy` p1 JOIN `baburam-shrestha.MedQ.Table_612_xy` p2 ON (p1.x, p1.y) != (p2.x, p2.y);	
626. Exchange Seats Can you write a SQL query to output the result for Mary?	SELECT (CASE WHEN MOD((SELECT MAX(id) FROM baburam-shrestha.MedQ.Table_626_Seat),2) = 1 AND id = (SELECT MAX(id) FROM baburam-shrestha.MedQ.Table_626_Seat) THEN id WHEN MOD(id,2) = 1 THEN id+1 ELSE id-1 END) AS id,student FROM baburam-shrestha.MedQ.Table_626_Seat order by id;	

1045. Customers Who Bought All Products

Write an SQL query for a report that provides the customer ids from the Customer table that bought all the products in the Product table.

```
SELECT a.customer_id
FROM (SELECT customer_id, COUNT (product_key) AS num
      FROM baburam-shrestha.MedQ.Table_1045_Customer
      GROUP BY customer_id) AS a
WHERE a.num = (SELECT COUNT(DISTINCT product_key)
               FROM baburam-shrestha.MedQ.Table_1045_Product
               );
```

Row	customer_id
1	1
2	3

1070. Product Sales Analysis III

Write an SQL query that selects the product id, year, quantity, and price for the first year of every product sold.

```
SELECT product_id,year AS first_year,quantity,price
FROM baburam-shrestha.MedQ.Table_1070_Sale
WHERE year IN(
  SELECT MIN(year)
  FROM  baburam-shrestha.MedQ.Table_1070_Product
  GROUP BY product_id);
```

Row	product_id	first_year	quantity	price
1	100	2008	10	5000
2	100	2009	12	5000
3	200	2011	15	9000

1077. Project Employees III

Write an SQL query that reports the most experienced employees in each project. In case of a tie, report all employees with the maximum number of experience years.

```
WITH CTE AS(
SELECT P.project_id,
  P.employee_id, DENSE_RANK()
  OVER(PARTITION BY P.project_id ORDER BY E.experience_years DESC)
  AS experience_rank
FROM  baburam-shrestha.MedQ.Table_1077_Project p
JOIN baburam-shrestha.MedQ.Table_1077_Employee E
  ON P.employee_id = E.employee_id)
SELECT project_id, employee_id
FROM CTE
WHERE experience_rank = 1
ORDER BY project_id;
```

Row	project_id	employee_id
1	1	3
2	2	1

1112. Highest Grade For Each Student

Write a SQL query to find the highest grade with its corresponding course for each student. In case of a tie, you should find the course with the smallest course_id. The output must be sorted by increasing student_id.

```
WITH CTE AS ( SELECT  e.*,
max(grade)  OVER (PARTITION BY student_id ORDER BY student_id) AS max_grade
FROM baburam-shrestha.MedQ.Table_1112_Enrollment e)

SELECT student_id,course_id,grade FROM CTE
WHERE CTE.grade = max_grade;
```

Row	student_id	course_id	grade
1	1	2	99
2	2	2	95
3	2	3	95
4	3	3	82

1126. Active Businesses

Write an SQL query to find all active businesses.

```
SELECT DISTINCT business_id
FROM(
SELECT business_id, event_type,
AVG(occurences) OVER(PARTITION BY event_type) AS avg1, occurences
FROM baburam-shrestha.MedQ.Table_1126_Event) A
WHERE occurences > avg1
GROUP By business_id
HAVING COUNT(DISTINCT event_type) > 1
```

Row	business_id
1	1

1164. Product Price at a Given Date

Write an SQL query to find the prices of all products on 2019-08-16. Assume the price of all products before any change is 10.

```
WITH CTE AS (SELECT *,
  RANK() OVER(PARTITION BY product_id ORDER BY change_date DESC) AS ranking
  FROM baburam-shrestha.MedQ.Table_1164_Product
  WHERE change_date <= '2019-08-16'
  )

SELECT A.product_id, IFNULL(new_price, 10) AS price
FROM (SELECT DISTINCT product_id
  FROM baburam-shrestha.MedQ.Table_1164_Product) A
  LEFT JOIN  (SELECT * FROM CTE WHERE ranking = 1) B
    ON A.product_id = B.product_id
WHERE A.Product_id IS NOT NULL;
```

Row	product_id	price
1	1	35
2	2	50
3	3	10

1174. Immediate Food Delivery II

Write an SQL query to find the percentage of immediate orders in the first orders of all customers, rounded to 2 decimal places.

```
WITH CTE AS (
SELECT *, rank() over(partition by customer_id order by order_date) as ranking
FROM baburam-shrestha.MedQ.Table_1174_Delivery)
SELECT ROUND(AVG(CASE
    WHEN order_date = customer_pref_delivery_date then 1.00 else 0.00 end)*100, 2)
    AS immediate_percentage
FROM CTE
WHERE ranking = 1;
```

Row	immediate_...
1	50.0

1193. Monthly Transactions I

Write an SQL query to find for each month and country, the number of transactions and their total amount, the number of approved transactions and their total amount.

```
SELECT Format_DATE("%Y-%m",trans_date) AS month,country,
    COUNT(id) AS trans_count,
    SUM(case when state='approved' then 1  else 0 end) AS approved_count,
    SUM(amount) as trans_total_amount,
    SUM(case when state='approved' then amount  else 0 end) AS approved_total_amount
FROM baburam-shrestha.MedQ.Table_1193_Transaction
GROUP BY month, country;
```

Row	month	country	trans_count	approved_c...	trans_total_...	approved_to...
1	2018-12	US	2	1	3000	1000
2	2019-01	US	1	1	2000	2000
3	2019-01	DE	1	1	2000	2000

1204. Last Person to Fit in the Elevator

Write an SQL query to find the person_name of the last person who will fit in the elevator without exceeding the weight limit. It is guaranteed that the person who is first in the queue can fit in the elevator.

```
WITH  CTE AS (
SELECT  person_name,weight,
SUM(weight) over(order by turn) As weight_sum
FROM baburam-shrestha.MedQ.Table_1204_Queue)

SELECT person_name FROM CTE
WHERE weight_sum = 1000;
```

Row	person_name
1	Thomas Jefferson

1212. Team Scores in Football Tournament
Write an SQL query that selects the team_id, team_name and num_points of each team in the tournament after all described matches. Result table should be ordered by num_points (decreasing order). In case of a tie, order the records by team_id (increasing order).

```
SELECT team_id, team_name,
  SUM(case when team_id = host_team and host_goals > guest_goals then 3 else 0 end) +
  SUM(case when team_id = guest_team and guest_goals > host_goals then 3 else 0 end) +
  SUM(case when team_id = host_team and host_goals = guest_goals then 1 else 0 end) +
  SUM(case when team_id = guest_team and host_goals = guest_goals then 1 else 0 end)
   AS num_points
FROM baburam-shrestha.MedQ.Table_1212_Team
LEFT JOIN baburam-shrestha.MedQ.Table_1212_Match
  ON team_id = host_team or team_id = guest_team
GROUP BY team_id, team_name
ORDER BY num_points DESC;
```

Row	team_id	team_name	num_points
1	10	Leetcode FC	7
2	20	NewYork FC	3
3	50	Toronto FC	3
4	30	Atlanta FC	1

1264. Page Recommendations
Write an SQL query to recommend pages to the user with user_id = 1 using the pages that your friends liked. It should not recommend pages you already liked.

```
SELECT DISTINCT page_id AS recommended_page
FROM baburam-shrestha.MedQ.Table_1264_Like
WHERE user_id IN (
  SELECT
    CASE
    WHEN user1_id=1 THEN user2_id
    WHEN user2_id=1 THEN user1_id
    END
  FROM baburam-shrestha.MedQ.Table_1264_Friendship
  WHERE user1_id=1 or user2_id=1
  )
  AND page_id NOT IN(
    SELECT page_id FROM baburam-shrestha.MedQ.Table_1264_Like WHERE user_id=1
    );
```

Row	recommend...
1	23
2	77
3	24
4	56
5	33

1270. All People Report to the Given Manager
Write an SQL query to find employee_id of all employees that directly or indirectly report their work to the head of the company.

```
SELECT employee AS employee_id
FROM (
    SELECT e1.employee_id AS employee, e1.manager_id AS manager1,
    e2.manager_id AS manager2, e3.manager_id AS manager3
    from baburam-shrestha.MedQ.Table_1270_Employee e1
    LEFT JOIN baburam-shrestha.MedQ.Table_1270_Employee e2
    ON e1.manager_id = e2.employee_id
    LEFT JOIN baburam-shrestha.MedQ.Table_1270_Employee e3
    on e2.manager_id = e3.employee_id )
WHERE employee <> 1 AND (manager1 = 1 OR manager2 = 1 OR manager3 = 1);
```

Row	employee_id
1	2
2	4
3	7

1285. Find the Start and End Number of Continuous Ranges
Write an SQL query to find the start and end number of continuous ranges in table Logs.

```
WITH CTE AS(
SELECT log_id,log_id - ROW_NUMBER() OVER (ORDER BY log_id) AS roww
FROM baburam-shrestha.MedQ.Table_1285_Log)
SELECT DISTINCT
FIRST_VALUE(log_id) OVER (PARTITION BY roww ORDER BY log_id) start_id,
LAST_VALUE(log_id) OVER (PARTITION BY roww ORDER BY log_id
RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) end_id
FROM CTE;
```

Row	start_id	end_id
1	1	3
2	7	8
3	10	10

1308. Running Total for Different Genders
Write an SQL query to find the total score for each gender on each day.
Order the result table by gender and day

```
SELECT gender, day,
    SUM(score_points) OVER (PARTITION BY gender ORDER BY day) total
FROM baburam-shrestha.MedQ.Table_1308_Score;
```

Row	gender	day	total
1	F	2019-12-30	17
2	F	2019-12-31	40
3	F	2020-01-01	57
4	F	2020-01-07	80
5	M	2019-12-18	2
6	M	2019-12-25	13
7	M	2019-12-30	26

1321. Restaurant Growth
Write an SQL query to compute moving average of how much customer paid in a 7 days window (current day + 6 days before) .

```
select visits.visited_on as visited_on, sum(c.amount) as amount,
  round(sum(c.amount) / 7.0, 2) as average_amount
    from (
        select distinct visited_on
        from baburam-shrestha.MedQ.Table_1321_Customer
            where date_diff(visited_on, (select min(visited_on)
            from baburam-shrestha.MedQ.Table_1321_Customer)) >= 6
  ) visits
  left join baburam-shrestha.MedQ.Table_1321_Customer c
  on date_diff(visits.visited_on, c.visited_on) between 0 and 6
  group by visits.visited_on
  order by visited_on;
```

1225

1336. Number of Transactions per Visit
Write an SQL query to find how many users visited the bank and didn't do any transactions, how many visited the bank and did one transaction and so on.