BEST SET OF SQL QUERIES I HAVE USED FOR INTERVIEWS

01

SQL QUERY TO FIND CONSECU
APPEARING 3 TIMES (WITH F

```sql
SELECT DISTINCT numbers
FROM (
    SELECT numbers,
           LAG(numbers, 1) OVER (ORDER BY id) AS prev_num,
           LEAD(numbers, 1) OVER (ORDER BY id) AS next_num
    FROM my_table
) t
WHERE numbers = prev_num AND numbers = next_num;

```

02

SQL QUERY TO FIND CONSECU
APPEARING 3 TIMES (WITHOU

```sql
SELECT DISTINCT a.numbers
FROM my_table a, my_table b, my_table c
WHERE a.numbers = b.numbers
AND b.numbers = c.numbers
AND a.id = b.id - 1
AND b.id = c.id - 1;
```

03

FIND THE DAYS WHEN THE TE
HIGHER THAN ITS PREVIOUS
(WITH FUNCTIONS)

```sql
SELECT days, temp
FROM (
    SELECT days, temp, LAG(temp)
    OVER (ORDER BY days) AS prev_temp
    FROM weather
) t
WHERE temp > prev_temp;
```

04

FIND THE DAYS WHEN THE TE
HIGHER THAN ITS PREVIOUS
(WITHOUT FUNCTIONS)

```sql
SELECT w1.days, w1.temp
FROM weather w1, weather w2
WHERE w1.days = w2.days + 1
AND w1.temp > w2.temp;
```

05

SQL QUERY TO DELETE DUPLI
TABLE. (FIRST WAY)

```sql
WITH CTE AS (
    SELECT *,
           ROW_NUMBER() OVER
           (PARTITION BY col1, col2 ORDER BY id) AS rn
    FROM my_table
)
DELETE FROM my_table
WHERE id IN (SELECT id FROM CTE WHERE rn > 1);
```

SQL QUERY TO DELETE DUPLI
TABLE. (SECOND WAY)

```sql
DELETE FROM my_table
WHERE id NOT IN (
    SELECT MIN(id)
    FROM my_table
    GROUP BY col1, col2
);
```

07

FIND THE CUMULATIVE SUM O
EMPLOYEE FROM JAN TO JUL

```sql
SELECT emp_id, month, salary,
       SUM(salary) OVER
       (PARTITION BY emp_id ORDER BY month)
       AS cumulative_salary
FROM employee_salaries
WHERE month BETWEEN 'Jan' AND 'Jul';
```

SQL QUERY FOR YOY GROWTH

YOY GROWTH=

$$\frac{\text{CURRENT YEAR REVENUE} - \text{PREVIOUS YEAR REVENUE}}{\text{PREVIOUS YEAR REVENUE}} \times 100$$

08

```sql
WITH yearly_spend AS (
    SELECT
        product_id,
        EXTRACT(YEAR FROM transaction_date) AS year,
        SUM(spend) AS total_spend,
        LAG(SUM(spend)) OVER
        (PARTITION BY product_id ORDER BY
        EXTRACT(YEAR FROM transaction_date)) AS prev_year_spend
    FROM transactions
    GROUP BY product_id, EXTRACT(YEAR FROM transaction_date)
)
SELECT
    year,
    product_id,
    prev_year_spend,
    (total_spend - prev_year_spend) / prev_year_spend * 100
    AS yoy_growth
FROM yearly_spend;
```

09

SQL QUERY FOR ROLLING AV PER USER

```sql
SELECT
    user_id,
    post_date,
    COUNT(post_id) AS daily_posts,
    AVG(COUNT(post_id)) OVER (
        PARTITION BY user_id
        ORDER BY post_date
        ROWS BETWEEN 6 PRECEDING AND CURRENT ROW
    ) AS rolling_avg_7_days
FROM posts
GROUP BY user_id, post_date
ORDER BY user_id, post_date;
```