### Write the output of the below query ?

Sources:

|  |  |
| --- | --- |
| **ID** | **VALUE\_1** |
| 1 | A |

|  |  |
| --- | --- |
| **ID** | **VALUE\_1** |
| 1 | X |

1. **What is the output of below query?**

select \*

from Left\_table LT

left outer join Right\_Table RT

on LT.ID = RT.ID **AND** LT.VALUE\_1='B'

**Ans:**

The query returns the below results.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **VALUE\_1** | **ID** | **VALUE\_1** |
| 1 | A | NULL | NULL |

**Understanding the Problem**

* A **LEFT OUTER JOIN** keeps all rows from Left\_table and brings matching rows from Right\_Table, filling unmatched rows with NULLs.
* The ON clause defines the join condition—how rows from both tables should match.
* The issue arises because the condition LT.VALUE\_1 = 'B' is applied **before** the join.
  + If LT.VALUE\_1 is not 'B', the join condition fails, and the row is effectively **removed** from the join output.
  + This contradicts the expected behavior of a LEFT OUTER JOIN, which should keep all Left\_table rows even if there is no match.

b)

select \* from Left\_table LT left outer join Right\_Table RT

on LT.ID = RT.ID **WHERE** LT.VALUE\_1='B'

**ANS:**

No Results

1. Write a alternate query to the union statement.

**Source 1:**

|  |
| --- |
| ID |
| 1 |
| 2 |
| 3 |

**Source 2:**

|  |
| --- |
| ID |
| 2 |
| 4 |
| 5 |

select ID from Left\_Table

UNION

select ID from Right\_Table

|  |
| --- |
| ID |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |

Write the alternate query as similar to generate same output without using union.

**ANS:**

select LT.ID,RT.ID ,COALESCE(LT.ID,RT.ID) AS ID FROM LEFT\_TABLE LT

FULL OUTER JOIN RIGHT\_TABLE RT ON LT.ID=RT.ID

3.Write sql query to extract below output.

Sources:

ACCOUNT\_DETAILS

|  |  |  |
| --- | --- | --- |
| **ACCOUNT\_ID** | **OPENED\_ON** | **OPENING\_BALANCE** |
| ACC001 | 05-12-2024 00:00 | 10000 |

Account\_Transaction

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ACCOUNT\_ID** | **TRANSACTION\_ID** | **DATE\_TIME** | **AMOUNT** | **TYPE** |
| ACC001 | T1 | 12-12-2024 01:05 | 2000 | Credit |
| ACC001 | T2 | 12-12-2024 02:05 | 3000 | Credit |
| ACC001 | T3 | 12-12-2024 04:05 | 5000 | Credit |
| ACC001 | T4 | 13-12-2024 04:05 | 10000 | Debit |
| ACC001 | T5 | 13-12-2024 05:05 | 4000 | Debit |
| ACC001 | T6 | 13-12-2024 06:05 | 5000 | Credit |
| ACC001 | T7 | 13-12-2024 17:05 | 6000 | Credit |
| ACC001 | T8 | 17-12-2024 04:05 | 3000 | Credit |
| ACC001 | T9 | 17-12-2024 07:05 | 4000 | Debit |
| ACC001 | T10 | 18-12-2024 09:45 | 2000 | Credit |
| ACC001 | T11 | 19-12-2024 04:05 | 5000 | Credit |
| ACC001 | T12 | 19-12-2024 07:05 | 3000 | Credit |
| ACC001 | T13 | 19-12-2024 23:05 | 2000 | Credit |

Expected Output:

|  |  |  |
| --- | --- | --- |
| **Account\_ID** | **Balance as on** | **Balance** |
| ACC001 | 12-12-2024 | 20000 |
| ACC001 | 13-12-2024 | 17000 |
| ACC001 | 17-12-2024 | 16000 |
| ACC001 | 18-12-2024 | 18000 |
| ACC001 | 19-12-2024 | 28000 |

Ans:

with all\_trns as (

select ACCOUNT\_ID,OPENED\_ON::date as TRNS\_DT,OPENING\_BALANCE AS TRNS\_AMT,'Credit' as Type from account\_details

union

select ACCOUNT\_ID,DATE\_TIME::date as TRNS\_DT,AMOUNT as TRNS\_AMT, Type from account\_transaction

),

NET\_BAL as(

select ACCOUNT\_ID,TRNS\_DT,

sum(case when type='Credit' then TRNS\_AMT ELSE -TRNS\_AMT END) as BALANCE

from all\_trns group by ACCOUNT\_ID,TRNS\_DT

)

select ACCOUNT\_ID,TRNS\_DT,

SUM(BALANCE) OVER(partition by ACCOUNT\_ID ORDER BY TRNS\_DT) AS BAL

from NET\_BAL order by TRNS\_DT

4.Write a sql query to find moving average of how much the customer paid in a seven days window (ie. Current day + 6days before). The average amount should be rounded to two decimal places.

( <https://www.youtube.com/watch?v=MKZBjfU-IN4> )

**Source:**

|  |  |  |  |
| --- | --- | --- | --- |
| CUSTOMER\_ID | NAME | VISITED\_DATE | AMOUNT |
| 1 | John | 01-12-2024 | 105 |
| 2 | Tom | 02-12-2024 | 110 |
| 3 | Yash | 03-12-2024 | 107 |
| 4 | Kavya | 04-12-2024 | 99 |
| 5 | Amit | 05-12-2024 | 111 |
| 6 | Danial | 06-12-2024 | 107 |
| 7 | Elvis | 07-12-2024 | 108 |
| 8 | Maraia | 08-12-2024 | 79 |
| 9 | Jaze | 09-12-2024 | 87 |
| 1 | John | 10-12-2024 | 91 |
| 2 | Tom | 10-12-2024 | 92 |

**Target:**

|  |  |  |
| --- | --- | --- |
| **Visited\_Date** | **Total\_amount** | **Avg\_Amount** |
| 07-12-2024 | 747 | 106.71 |
| 08-12-2024 | 721 | 103 |
| 09-12-2024 | 698 | 99.71 |
| 10-12-2024 | 774 | 110.57 |

ANS:

with cte as (

select visited\_date,sum(amount) as AMOUNT from customers

group by visited\_date

)

select VISITED\_DATE,TOT\_AMT,AVG\_AMT from (

select visited\_date,AMOUNT,

SUM(AMOUNT) over(order by visited\_date ROWS BETWEEN 6 PRECEDING AND CURRENT ROW) as TOT\_AMT,

ROUND(AVG(AMOUNT) over(order by visited\_date ROWS BETWEEN 6 PRECEDING AND CURRENT ROW),2) as AVG\_AMT,

row\_number() over(order by visited\_date) as RNK

from cte

) where rnk > 6

### 5. In Redbus, when we book a seat adjacent seat should be same gender. How do you write SQL write for the same to test?

WITH adjacent\_seats AS (

SELECT

b1.bus\_id,

b1.seat\_number AS seat1,

b2.seat\_number AS seat2,

b1.gender AS gender1,

b2.gender AS gender2

FROM bus\_bookings b1

JOIN bus\_bookings b2

ON b1.bus\_id = b2.bus\_id -- Same bus

AND ABS(b1.seat\_number - b2.seat\_number) = 1 -- Check adjacent seats (Numeric seat numbers)

AND b1.gender != b2.gender -- Different genders

)

SELECT \* FROM adjacent\_seats;

**6. Write a SQL query to find the people who have the most friends and friend count.**

**( https://www.youtube.com/watch?v=yzxEY\_tsM1s )**

**Source:**

|  |  |  |
| --- | --- | --- |
| SEND\_ID | RECEIVE\_ID | DATE\_ID |
| 1 | 2 | 01-01-2024 |
| 1 | 3 | 02-01-2024 |
| 1 | 4 | 03-01-2024 |
| 2 | 3 | 04-01-2024 |
| 3 | 4 | 05-01-2024 |
| 5 | 1 | 06-01-2024 |

**Target:**

|  |  |
| --- | --- |
| ID | No. of Friends |
| 1 | 4 |

Ans:

with cte as (

select SEND\_ID AS ID from request

union all

select receive\_id as ID from request

),

cte2 as (

select id, count(\*) as friend\_cnt from cte group by id

)

select id ,friend\_cnt from cte2 where friend\_cnt >= ( select max(friend\_cnt) from cte2)

**7. Write a query to find the minimum consecutive 3 rows, where visitors count is greater than 100.**

**(** [**https://www.youtube.com/watch?v=6T9pbFB7w6A**](https://www.youtube.com/watch?v=6T9pbFB7w6A) **)**

|  |  |  |
| --- | --- | --- |
| ID | VISIT\_DATE | VISITORS |
| 1 | 01-01-2025 | 101 |
| 2 | 02-01-2025 | 110 |
| 3 | 03-01-2025 | 70 |
| 4 | 04-01-2025 | 105 |
| 5 | 05-01-2025 | 107 |
| 6 | 06-01-2025 | 70 |
| 7 | 07-01-2025 | 107 |
| 8 | 08-01-2025 | 120 |
| 9 | 09-01-2025 | 109 |
| 10 | 10-01-2025 | 99 |

**Target:**

|  |  |
| --- | --- |
| VISIT\_DATE | VISITORS |
| 07-01-2025 | 107 |
| 08-01-2025 | 120 |
| 09-01-2025 | 109 |

**Ans:**

with cte as (

select \* from visit where visitors>100

),

cte1 as (

select \*,row\_number() over(order by visit\_date) as rno from cte ,

),cte2 as (

select \*,id-rno as grp from cte1

),cte3 as (

select \*,count(\*) over(partition by grp) as cnt from cte2 )

select \* from cte3 where cnt>=3

8. Write a sql query to find category based on income.

Less than 20000 is low salary, 20000 to 40000 is avg salary, more than 40000 is high salary.

Source:

|  |  |
| --- | --- |
| ACCOUNT\_ID | INCOME |
| 1 | 90000 |
| 2 | 17000 |
| 3 | 22000 |
| 4 | 13000 |
| 5 | 50000 |
| 6 | 56000 |
| 7 | 19000 |

Ans:

select \*,

case when income<=20000 then 'Low'

when income between 20000 and 40000 then 'Medium'

when income>40000 then 'High'

END status

from accounts

9. Write a query to find the minimum consecutive 3 rows where visitors count is greater than 100.

Source:

|  |  |  |
| --- | --- | --- |
| ID | VISIT\_DATE | VISITORS |
| 1 | 01-01-2025 | 101 |
| 2 | 02-01-2025 | 110 |
| 3 | 03-01-2025 | 70 |
| 4 | 04-01-2025 | 105 |
| 5 | 05-01-2025 | 107 |
| 6 | 06-01-2025 | 70 |
| 7 | 07-01-2025 | 107 |
| 8 | 08-01-2025 | 120 |
| 9 | 09-01-2025 | 109 |
| 10 | 10-01-2025 | 99 |

Target:

|  |  |  |
| --- | --- | --- |
| ID | VISIT\_DATE | VISITORS |
| 7 | 07-01-2025 | 107 |
| 8 | 08-01-2025 | 120 |
| 9 | 09-01-2025 | 109 |

Ans:

with cte as (

select \* from visit where visitors >=100

)

, cte1 as (

select \*,id-row\_number() over(order by id ) as grp from cte

),cte2 as (

select \*,count(\*) over(partition by grp) as cnt from cte1

)

select \* from cte2 where cnt>=3

10. Given us Student table, find out the total marks of top 2 subjects based on marks.

( <https://www.youtube.com/watch?v=q5AsEPJucMU> )

Source:

|  |  |  |
| --- | --- | --- |
| SNAME | SID | MARKS |
| A | X | 75 |
| A | Y | 75 |
| A | Z | 80 |
| B | X | 90 |
| B | Y | 91 |
| B | Z | 75 |

Target:

|  |  |
| --- | --- |
| Sname | Tot\_Marks |
| A | 155 |
| B | 181 |

Ans:

with cte1 as (

select \*, row\_number() over(partition by SNAME order by marks desc) as rnk from students\_1)

select SNAME,sum(MARKS) as tot\_marks from cte1 where rnk <=2 group by SNAME