Consecutive Streak - without window function (Approach → NOT EXISTS)

user_id	dt	exists(prev record?)	not exists(streak_start)
A	2025-01-01	no	yes
A	2025-01-02	yes	no
A	2025-01-03	yes	no
B	2025-01-04	no	yes
B	2025-01-05	yes	no

1) user - start_streak
2) user-end_streak
3) Pair them → using this condition: **end_date >= start_date(See detail at the end)**
4) Subtract to calculate length (user → length of the streak)

NOT EXISTS Understanding:

```
SELECT *
FROM table t
WHERE NOT EXISTS (
   SELECT 1
   FROM table t1
   WHERE <some condition comparing t and t1>
)
```

user_id	dt	exists(next record?)	not exists(streak_end)
A	2025-01-01	yes	no
A	2025-01-02	yes	no
A	2025-01-03	no	yes
B	2025-01-04	yes	no
B	2025-01-05	no	yes

Soln:

```
with start_dates AS(
select
user_id,
t.dt as start_streak
from trans_data t
where exists not
( select 1
from trans_data t1
where t.user_id = t1.user_id
and t.dt = t1.dt - INTERVAL '1 DAY' ) ),
end_dates AS(
select
user_id,
```

```
t.dt as end_streak
from trans_data t
where exists not
( select 1
from trans_data t1
where t.user_id = t1.user_id
and t.dt = t1.dt + INTERVAL '1 DAY' ) )
s.user_id,
s.start_date,
e.end_date,
(e.end_date - s.start_date) + 1 streak_length
from start_date s
JOIN end_date e
on s.user_id = e.user_id
AND e.end_streak >= s.start_streak
order by s.user_id, s.start_date
```

Note:

1.) This query works for clean dataset
2.) In case where there are gaps in dates at user level this might not work → It becomes a cross-join for each user

start_streak
2025-01-01
2025-01-06

end_streak
2025-01-03
2025-01-07

Query produces:

start_streak	**end_streak**
1 Jan	3 Jan ✔ correct
1 Jan	7 Jan ❌ wrong pairing
6 Jan	3 Jan ❌ wrong
6 Jan	7 Jan ✔ correct

Revised-Soln →

```
With Start_date AS(
select
t.User_id,
t.dt as start_streak,
row_number() over(partition by user_id order by dt) as r_no
From trans_data t
Where NOT EXISTS (Select 1
From trans_data t1
Where t.user_id = t1.user_id
And t.dt = t1.dt - INTERVAL '1 DAY')),
End_date AS(
select
User_id,
Dt as end_streak
row_number() over(partition by user_id order by dt) as r_no
From trans_data t
Where NOT EXISTS (Select 1
From trans_data t1
Where t.user_id = t1.user_id
And t.dt = t1.dt + INTERVAL '1 DAY'))
)
select
S.user_id,
S.start_date,
E.end_date
From
Start_date s
Join end_date e
on s.user_id = e.user_id
and s.r_no = e.r_no
```

Output → No cross join will happen when the same user has multiple different streaks are there

Start_date (U1)	End_date (U1)	r_no
2025-01-01	2025-01-03	1
2025-01-06	2025-01-07	2

Pair them → end_date >= start_date

Note →

1.) This condition creates candidates between start-dates and end-dates
2.) It does not determine streak by itself (that's why: Count(*) as streak_length will not work)
3.) This condition is simply a **relational join filter** — a way of pairing rows across two sets.
Not a recursive way(keep calling itself until stopping condition is met)

Raw dates

date
2025-01-01
2025-01-02
2025-01-03
2025-01-05
2025-01-06

From this, we detect: →

start_date
2025-01-01
2025-01-05

end_date
2025-01–03
2025-01-06

Step 1 → Perform the JOIN
Start s join end e
On e.end_date >= s.start_date

Check all ends
2025-01-03: → 03(end_date)>=01(start_date) → **Match**
03 >= 05 → No Match
2025-01-06: → 06(end_date)>=01(start_date) → **Match**
06(end_date)>=05(start_date) → **Match**

Resulting joined table

start_date	end_date	Comment
2025-01-01	2025-01-03	
2025-01-01	2025-01-06	Row number eliminates this case
2025-01-05	2025-01-06	