Day 27 – Leave Approver

Problem Statement:

Given **vacation_plans** tables shows the vacations applied by each employee during the year 2024.
Leave_balance table has the available leaves for each employee.

Write an SQL query to determine if the vacations applied by each employee can be **approved** or **not** based on the available leave balance.

If an employee has sufficient available leaves then mention the **status** as "**Approved**" else mention "**Insufficient Leave Balance**".

Assume there are no public holidays during 2024. weekends (sat & sun) should be excluded while calculating vacation days.

Input:

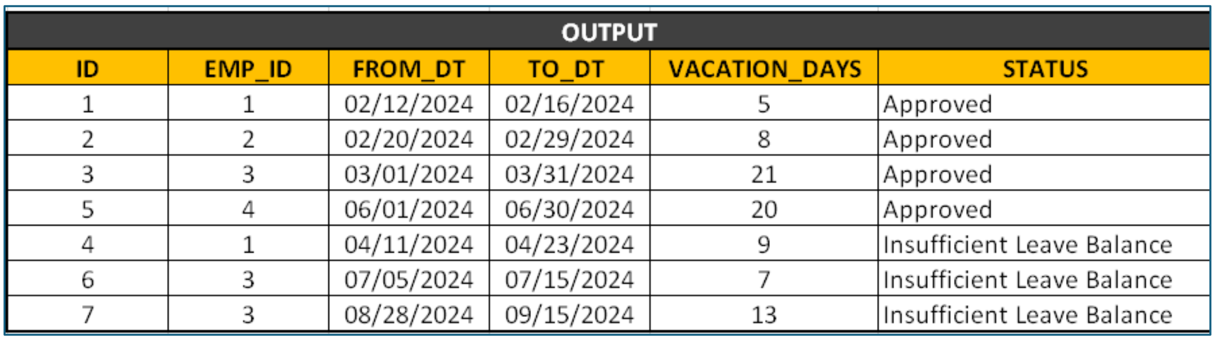
Table: vacation_plans

VACATION_PLANS			
ID	**EMP_ID**	**FROM_DT**	**TO_DT**
1	1	02/12/2024	02/16/2024
2	2	02/20/2024	02/29/2024
3	3	03/01/2024	03/31/2024
4	1	04/11/2024	04/23/2024
5	4	06/01/2024	06/30/2024
6	3	07/05/2024	07/15/2024
7	3	08/28/2024	09/15/2024

Table: leave_balance

LEAVE_BALANCE	
EMP_ID	**BALANCE**
1	12
2	10
3	26
4	20
5	14

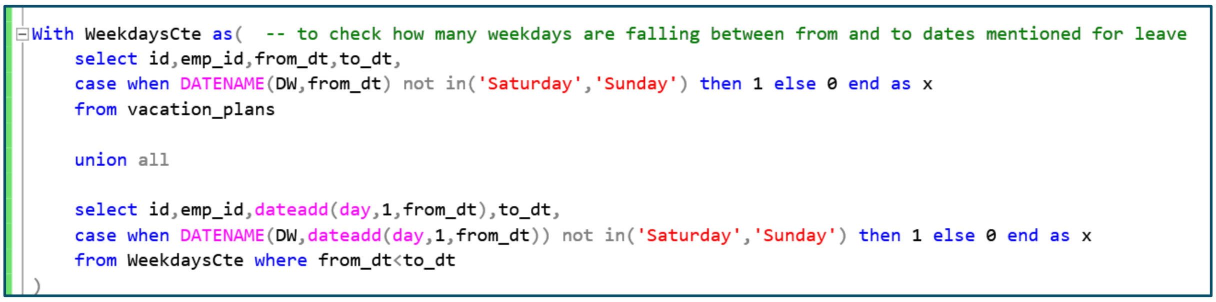
Expected Output



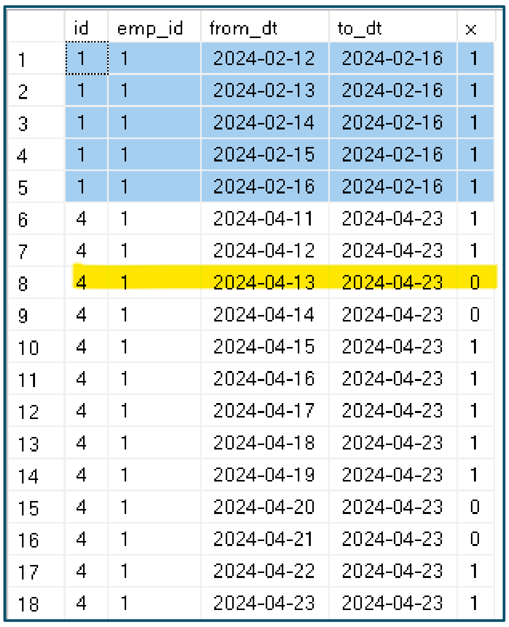
OUTPUT					
ID	EMP_ID	FROM_DT	TO_DT	VACATION_DAYS	STATUS
1	1	02/12/2024	02/16/2024	5	Approved
2	2	02/20/2024	02/29/2024	8	Approved
3	3	03/01/2024	03/31/2024	21	Approved
5	4	06/01/2024	06/30/2024	20	Approved
4	1	04/11/2024	04/23/2024	9	Insufficient Leave Balance
6	3	07/05/2024	07/15/2024	7	Insufficient Leave Balance
7	3	08/28/2024	09/15/2024	13	Insufficient Leave Balance

APPROACH

1. Calculate weekdays falling between from_dt and to_dt for each employee(emp_id) and id(since one employee can apply for multiple leaves)



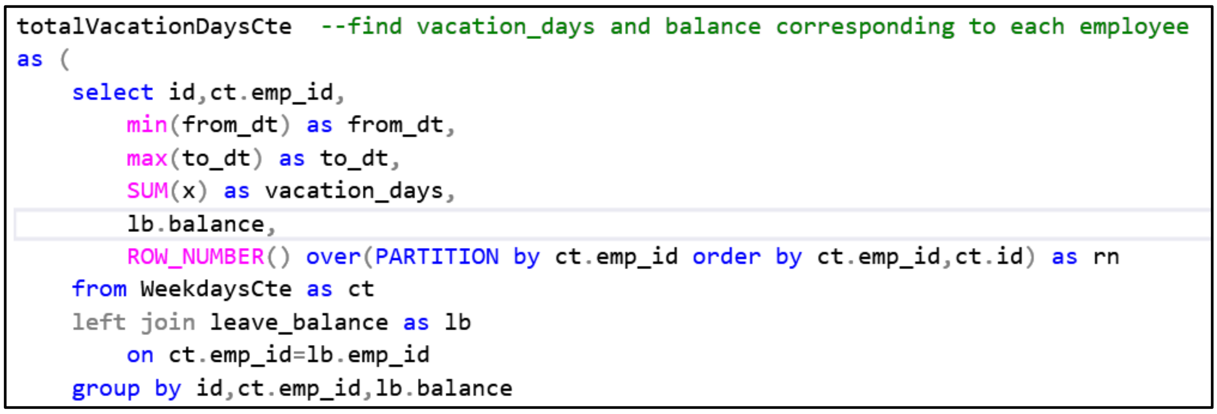
```
With WeekdaysCte as(  -- to check how many weekdays are falling between from and to dates mentioned for leave
    select id,emp_id,from_dt,to_dt,
    case when DATENAME(DW,from_dt) not in('Saturday','Sunday') then 1 else 0 end as x
    from vacation_plans

    union all

    select id,emp_id,dateadd(day,1,from_dt),to_dt,
    case when DATENAME(DW,dateadd(day,1,from_dt)) not in('Saturday','Sunday') then 1 else 0 end as x
    from WeekdaysCte where from_dt<to_dt
)
```



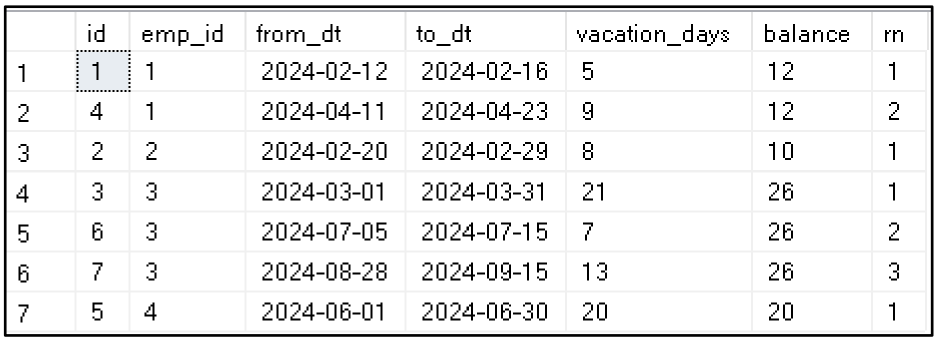
	id	emp_id	from_dt	to_dt	x
1	1	1	2024-02-12	2024-02-16	1
2	1	1	2024-02-13	2024-02-16	1
3	1	1	2024-02-14	2024-02-16	1
4	1	1	2024-02-15	2024-02-16	1
5	1	1	2024-02-16	2024-02-16	1
6	4	1	2024-04-11	2024-04-23	1
7	4	1	2024-04-12	2024-04-23	1
8	4	1	2024-04-13	2024-04-23	0
9	4	1	2024-04-14	2024-04-23	0
10	4	1	2024-04-15	2024-04-23	1
11	4	1	2024-04-16	2024-04-23	1
12	4	1	2024-04-17	2024-04-23	1
13	4	1	2024-04-18	2024-04-23	1
14	4	1	2024-04-19	2024-04-23	1
15	4	1	2024-04-20	2024-04-23	0
16	4	1	2024-04-21	2024-04-23	0
17	4	1	2024-04-22	2024-04-23	1
18	4	1	2024-04-23	2024-04-23	1

To verify: Check the highlighted row with from_dt as 2024-04-13 i.e Saturday, hence we get zero there. For the rows with x = 1, all are weekdays.

2. Sum the weekdays we received from previous CTE --- vacation_days.
   Join the balance table to find leaves that each employee has currently.
   Also, we need a row_number to identify the first record of the applied leaves per employee

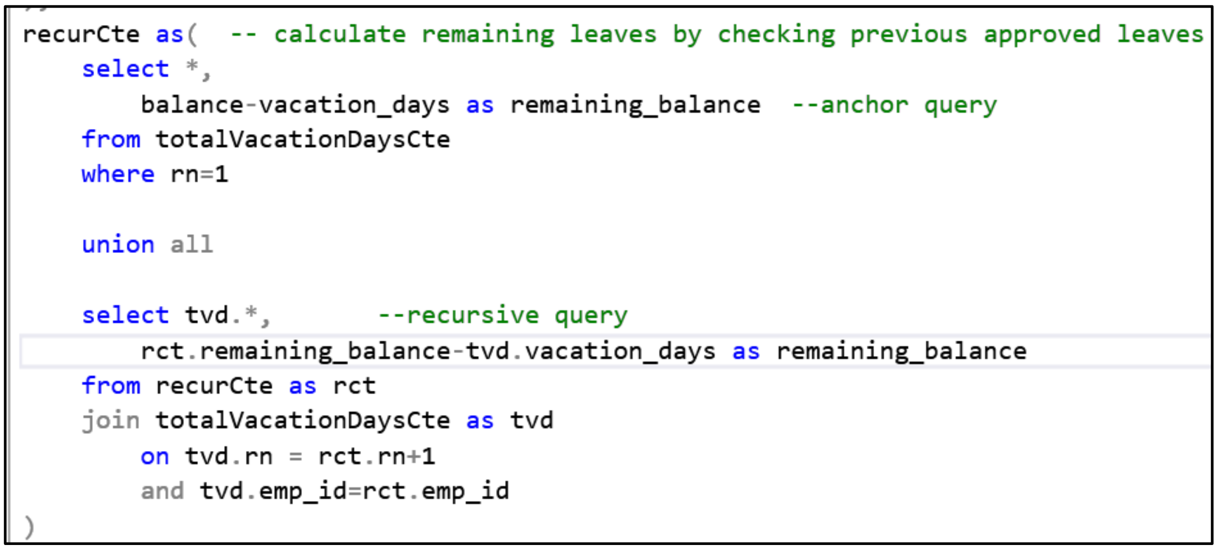


```sql
totalVacationDaysCte  --find vacation_days and balance corresponding to each employee
as (
    select id,ct.emp_id,
        min(from_dt) as from_dt,
        max(to_dt) as to_dt,
        SUM(x) as vacation_days,
        lb.balance,
        ROW_NUMBER() over(PARTITION by ct.emp_id order by ct.emp_id,ct.id) as rn
    from WeekdaysCte as ct
    left join leave_balance as lb
        on ct.emp_id=lb.emp_id
    group by id,ct.emp_id,lb.balance
```

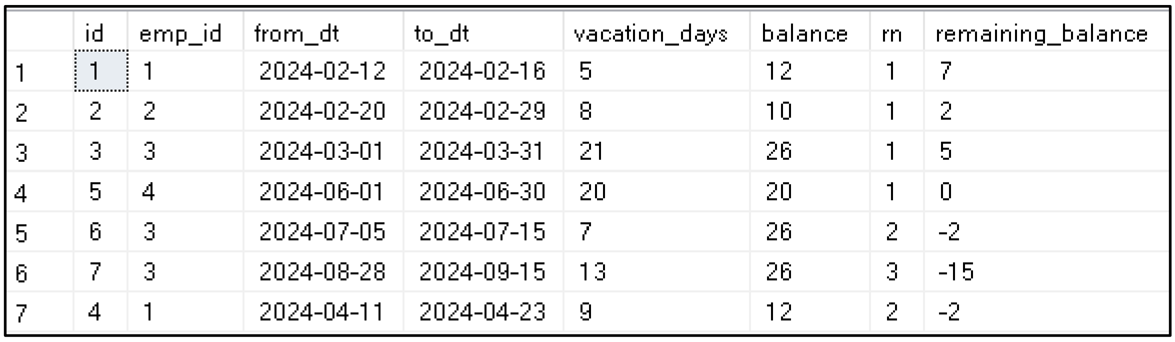


	id	emp_id	from_dt	to_dt	vacation_days	balance	rn
1	1	1	2024-02-12	2024-02-16	5	12	1
2	4	1	2024-04-11	2024-04-23	9	12	2
3	2	2	2024-02-20	2024-02-29	8	10	1
4	3	3	2024-03-01	2024-03-31	21	26	1
5	6	3	2024-07-05	2024-07-15	7	26	2
6	7	3	2024-08-28	2024-09-15	13	26	3
7	5	4	2024-06-01	2024-06-30	20	20	1

3. In this, we are calculating **remaining_balance** for every row where **rownumber** is 1 in **anchor** query.
   In **recursive** query, we will refer to the remaining_balance from anchor query along with the vacation_days has applied for and calculate remaining_balance for other entries of each employee.

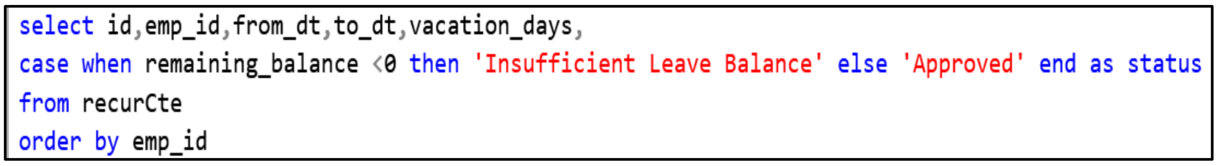


```sql
recurCte as(  -- calculate remaining leaves by checking previous approved leaves
    select *,
        balance-vacation_days as remaining_balance  --anchor query
    from totalVacationDaysCte
    where rn=1

    union all

    select tvd.*,        --recursive query
        rct.remaining_balance-tvd.vacation_days as remaining_balance
    from recurCte as rct
    join totalVacationDaysCte as tvd
        on tvd.rn = rct.rn+1
        and tvd.emp_id=rct.emp_id
)
```



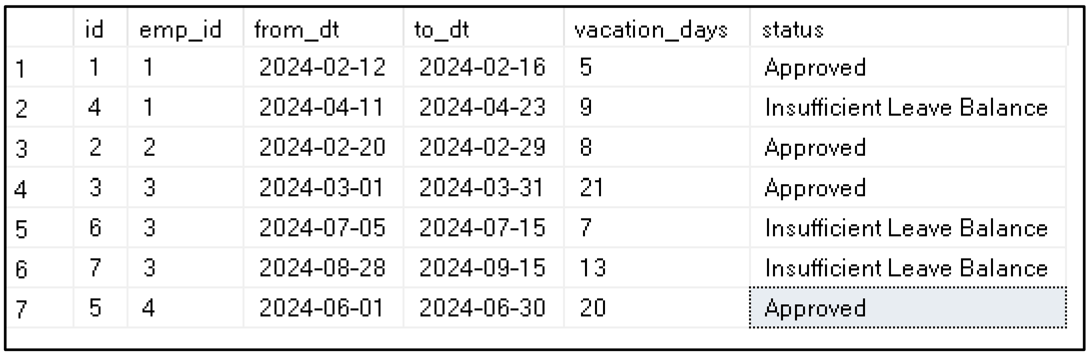
	id	emp_id	from_dt	to_dt	vacation_days	balance	rn	remaining_balance
1	1	1	2024-02-12	2024-02-16	5	12	1	7
2	2	2	2024-02-20	2024-02-29	8	10	1	2
3	3	3	2024-03-01	2024-03-31	21	26	1	5
4	5	4	2024-06-01	2024-06-30	20	20	1	0
5	6	3	2024-07-05	2024-07-15	7	26	2	-2
6	7	3	2024-08-28	2024-09-15	13	26	3	-15
7	4	1	2024-04-11	2024-04-23	9	12	2	-2

4. Finally, to our last step where we just have to **Unapprove** the leaves where remaining_balance is less than zero due to insufficient leaves ans approve the other with remaining_balance more than or equal to zero.



```
select id,emp_id,from_dt,to_dt,vacation_days,
case when remaining_balance <0 then 'Insufficient Leave Balance' else 'Approved' end as status
from recurCte
order by emp_id
```

FINAL OUTPUT

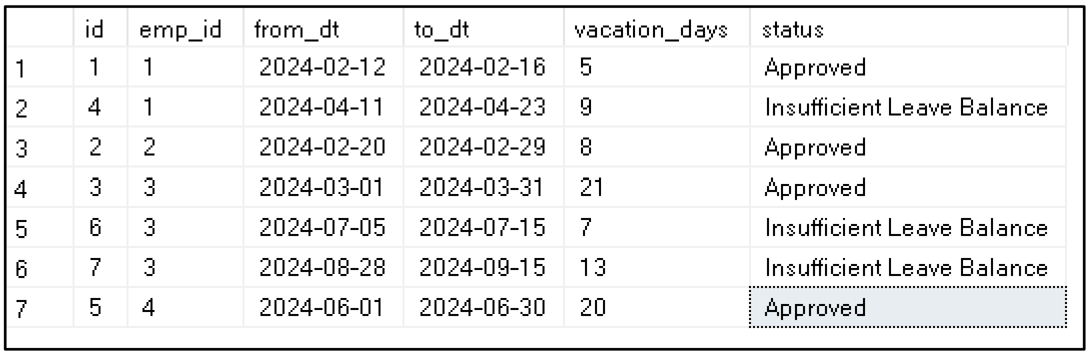


	id	emp_id	from_dt	to_dt	vacation_days	status
1	1	1	2024-02-12	2024-02-16	5	Approved
2	4	1	2024-04-11	2024-04-23	9	Insufficient Leave Balance
3	2	2	2024-02-20	2024-02-29	8	Approved
4	3	3	2024-03-01	2024-03-31	21	Approved
5	6	3	2024-07-05	2024-07-15	7	Insufficient Leave Balance
6	7	3	2024-08-28	2024-09-15	13	Insufficient Leave Balance
7	5	4	2024-06-01	2024-06-30	20	Approved

SOLUTION

```
With WeekdaysCte as(  -- to check how many weekdays are falling
between from and to dates mentioned for leave

      select id,emp_id,from_dt,to_dt,
      case when DATENAME(DW,from_dt) not in('Saturday','Sunday')
      then 1 else 0 end as x
      from vacation_plans

      union all

      select id,emp_id,dateadd(day,1,from_dt),to_dt,
      case when DATENAME(DW,dateadd(day,1,from_dt)) not
      in('Saturday','Sunday') then 1 else 0 end as x
      from WeekdaysCte where from_dt<to_dt
),
totalVacationDaysCte
--find vacation_days and balance corresponding to each employee
as (
      select id,ct.emp_id,min(from_dt) as from_dt,max(to_dt) as
      to_dt, SUM(x) as vacation_days, lb.balance,
      ROW_NUMBER() over(PARTITION by ct.emp_id order by
      ct.emp_id,ct.id) as rn
      from WeekdaysCte as ct
      left join leave_balance as lb
            on ct.emp_id=lb.emp_id
      group by id,ct.emp_id,lb.balance
),
recurCte as(
-- calculate remaining leaves by checking previous approved leaves
      select *, balance-vacation_days as remaining_balance
      from totalVacationDaysCte
      where rn=1

      union all

      select tvd.*,
      rct.remaining_balance-tvd.vacation_days as remaining_balance
      from recurCte as rct
      join totalVacationDaysCte as tvd
            on tvd.rn = rct.rn+1
            and tvd.emp_id=rct.emp_id
)
select id,emp_id,from_dt,to_dt,vacation_days,
case when remaining_balance <0 then 'Insufficient Leave Balance'
else 'Approved' end as status
from recurCte
order by emp_id
```

OUTPUT



	id	emp_id	from_dt	to_dt	vacation_days	status
1	1	1	2024-02-12	2024-02-16	5	Approved
2	4	1	2024-04-11	2024-04-23	9	Insufficient Leave Balance
3	2	2	2024-02-20	2024-02-29	8	Approved
4	3	3	2024-03-01	2024-03-31	21	Approved
5	6	3	2024-07-05	2024-07-15	7	Insufficient Leave Balance
6	7	3	2024-08-28	2024-09-15	13	Insufficient Leave Balance
7	5	4	2024-06-01	2024-06-30	20	Approved