---------- LEAVE APPROVAL OF AN EMPLOYEE ----------

Relation Database Management System (RDMS): PostgreSQL.

Prepared by: Michael Henry Bujiku.

PROBLEM STATEMENT:

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

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

VACATION_PLANS				
ID	**EMP_ID**	**Name**	**FROM_DT**	**TO_DT**
1	1	Michael Henry	2/12/2024	2/16/2024
2	2	Francis Mansetus	2/20/2024	2/29/2024
3	3	John Stewart	3/1/2024	3/31/2024
4	1	Michael Henry	4/11/2024	4/23/2024
5	4	Saul McGill	6/1/2024	6/30/2024
6	3	John Stewart	7/5/2024	7/15/2024
7	3	John Stewart	8/28/2024	9/15/2024

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

We have Two table.

- Vacation Plans Table.
- Leave Balance Table.

SQL QUERIES PARTS

```
select * from vacation_plans;
select * from leave_balance;
```

To find the series of days from the Date range (**From DT** to **TO_DT**).

select cast(dates as date) as dates, to_char(dates, 'Day') as day
from generate_series('2024-02-12'::date, '2024-02-16'::date, '1 Day') dates

dates	day
2/12/2024	Monday
2/13/2024	Tuesday
2/14/2024	Wednesday
2/15/2024	Thursday
2/16/2024	Friday

Lateral Keyword: This keyword is used when your subquery in the from clause then you want you those fields which is referred from that current table itself then you need to use lateral keyword.

*select **
from vacation_plans v
cross join lateral (select cast(dates as date) as dates, to_char(dates, 'Day') as day
from generate_series(v.from_dt, v.to_dt, '1 Day') dates) d

id	emp_id	name	from_dt	to_dt	dates	day
1	1	Michael Henry	2/12/2024	2/16/2024	2/12/2024	Monday
1	1	Michael Henry	2/12/2024	2/16/2024	2/13/2024	Tuesday
1	1	Michael Henry	2/12/2024	2/16/2024	2/14/2024	Wednesday
1	1	Michael Henry	2/12/2024	2/16/2024	2/15/2024	Thursday
1	1	Michael Henry	2/12/2024	2/16/2024	2/16/2024	Friday
2	2	Francis Mansetus	2/20/2024	2/29/2024	2/20/2024	Tuesday
2	2	Francis Mansetus	2/20/2024	2/29/2024	2/21/2024	Wednesday
2	2	Francis Mansetus	2/20/2024	2/29/2024	2/22/2024	Thursday
2	2	Francis Mansetus	2/20/2024	2/29/2024	2/23/2024	Friday
2	2	Francis Mansetus	2/20/2024	2/29/2024	2/24/2024	Saturday
2	2	Francis Mansetus	2/20/2024	2/29/2024	2/25/2024	Sunday

Note: In this Every date range from vacation_plans table goes corresponding with new date series which is created by lateral keyword by Cross Join in Every series as shown in a table.

To count the numbers of days planned by Employees for Vacations.
We exclude weekends(Saturday & Sunday).

```
select v.id, v.emp_id, v.name, v.from_dt, v.to_dt, count(d.dates) as vacation_days
from vacation_plans v
cross join lateral (select cast(dates as date) as dates, trim(to_char(dates, 'Day')) as day
                    from generate_series(v.from_dt, v.to_dt, '1 Day') dates) d
where day not in ('Saturday', 'Sunday')
group by v.id, v.emp_id, v.from_dt, v.to_dt;
```

id	emp_id	name	from_dt	to_dt	vacation_days
1	1	Michael Henry	2/12/2024	2/16/2024	5
2	2	Francis Mansetus	2/20/2024	2/29/2024	8
3	3	John Stewart	3/1/2024	3/31/2024	21
4	1	Michael Henry	4/11/2024	4/23/2024	9
5	4	Saul McGill	6/1/2024	6/30/2024	20
6	3	John Stewart	7/5/2024	7/15/2024	7
7	3	John Stewart	8/28/2024	9/15/2024	13

But we will join with the "Leave Balance" Table.

```
select v.id, v.emp_id, v.name, v.from_dt, v.to_dt
, l.balance as leave_balance
, count(d.dates) as vacation_days
from vacation_plans v
cross join lateral (select cast(dates as date) as dates, trim(to_char(dates, 'Day')) as day
                    from generate_series(v.from_dt, v.to_dt, '1 Day') dates) d
join leave_balance l on l.emp_id = v.emp_id
where day not in ('Saturday', 'Sunday')
group by v.id, v.emp_id, v.from_dt, v.to_dt, l.balance
order by v.emp_id, v.id;
```

id	emp_id	name	from_dt	to_dt	leave_balance	vacation_days
1	1	Michael Henry	2/12/2024	2/16/2024	12	5
4	1	Michael Henry	4/11/2024	4/23/2024	12	9
2	2	Francis Mansetus	2/20/2024	2/29/2024	10	8
3	3	John Stewart	3/1/2024	3/31/2024	26	21
6	3	John Stewart	7/5/2024	7/15/2024	26	7
7	3	John Stewart	8/28/2024	9/15/2024	26	13
5	4	Saul McGill	6/1/2024	6/30/2024	20	20

In this part we need to use "Recursion" because we want to check by reducing the number of vacation the Employees will take and to check the balance of days remain if is sufficient or not to take another vacation.
First we need to create a row number syntax in order to help us to identify which is the first application for leave then to minus those days and to get the remaining days by checking if is sufficient when employees want for another leave

, row_number() over(partition by v.emp_id order by v.emp_id, v.id) as rn

Recursion Syntax:

With recursive (CTE_name) as
*(select query (**No-Recursive query** or Base query)*
UNION ALL
*Select query(**Recursive query** using CTE_name [with a termination condition])*
)
*Select * from CTE_name;*

Then we start by base query (To retrieve all data from the first leave application):

This called a Base Query:

```
with cte_data as
    (select v.id, v.emp_id, v.name, v.from_dt, v.to_dt
     , l.balance as leave_balance, count(d.dates) as vacation_days
     , row_number() over(partition by v.emp_id order by v.emp_id, v.id) as rn
     from vacation_plans v
    cross join lateral
        (select cast(dates as date) as dates, trim(to_char(dates, 'Day')) as day
         from generate_series(v.from_dt, v.to_dt, '1 Day') dates) d
  join leave_balance l on l.emp_id = v.emp_id
  where day not in ('Saturday', 'Sunday')
  group by v.id, v.emp_id, v.from_dt, v.to_dt, l.balance
  order by v.emp_id, v.id)

select *, (leave_balance-vacation_days) as remaining_balance
from cte_data
where rn=1
```

Then we will get this one.

id	emp_id	name	from_dt	to_dt	leave_balance	vacation_days	rn	remaining_balance
1	1	Michael Henry	2/12/2024	2/16/2024	12	5	1	7
2	2	Francis Mansetus	2/20/2024	2/29/2024	10	8	1	2
3	3	John Stewart	3/1/2024	3/31/2024	26	21	1	5
5	4	Saul McGill	6/1/2024	6/30/2024	20	20	1	0

Then Recursive query CTE we will get.

select cd., (cte.remaining_balance - cd.vacation_days) as remaining_balance*
from cte
join cte_data cd on cd.rn = cte.rn+1 and cte.emp_id = cd.emp_id

Explanation: To find a new balance of vacation days we need to minus the vacation_days column which is from in cte_data query from the remaining balance which is from cte query

(cte.remaining_balance - cd.vacation_days) as remaining_balance

We need to recall from "cte" table query because that is our new query we created from base query.

The ideas of join is we want to retrive the new row number example a second row (2) from cte query and from cte_data which is the same and then we check by minus the remaining balance from previous calculation to the new vacation days requested by Employees.

Ex. Employee Michael Henry in second row(2 row) it has 9 vacation days request but we check his remaining balance is 7 days which we minus from the previous vacation.

This is why we calculate by putting cte.rn+1. To get the data like that.

on cd.rn = cte.rn+1

Then we put *cte.emp_id = cd.emp_id* in order to make sure the calculation is going proper with the same/corresponding Employees from Each table query which is cte query and cte_data query.

id	emp_id	name	from_dt	to_dt	leave_balance	vacation_days	rn	r_balance
1	1	Michael Henry	2/12/2024	2/16/2024	12	5	1	7
2	2	Francis Mansetus	2/20/2024	2/29/2024	10	8	1	2
3	3	John Stewart	3/1/2024	3/31/2024	26	21	1	5
5	4	Saul McGill	6/1/2024	6/30/2024	20	20	1	0
4	1	Michael Henry	4/11/2024	4/23/2024	12	9	2	-2
6	3	John Stewart	7/5/2024	7/15/2024	26	7	2	-2
7	3	John Stewart	8/28/2024	9/15/2024	26	13	3	-15

FINAL QUERY:

```
with recursive cte as(
        with cte_data as
            (select v.id, v.emp_id, v.name, v.from_dt, v.to_dt
            , l.balance as leave_balance, count(d.dates) as vacation_days
            , row_number() over(partition by v.emp_id order by v.emp_id, v.id) as rn
            from vacation_plans v
            cross join lateral
                (select cast(dates as date) as dates, trim(to_char(dates, 'Day')) as day
                from generate_series(v.from_dt, v.to_dt, '1 Day') dates) d
            join leave_balance l on l.emp_id = v.emp_id
            where day not in ('Saturday', 'Sunday')
            group by v.id, v.emp_id, v.from_dt, v.to_dt, l.balance
            order by v.emp_id, v.id)

        select *, (leave_balance-vacation_days) as remaining_balance
        from cte_data
        where rn=1
        union all
        select cd.*, (cte.remaining_balance - cd.vacation_days) as remaining_balance
        from cte
        join cte_data cd on cd.rn = cte.rn+1 and cte.emp_id = cd.emp_id
)

select id, emp_id, name, from_dt, to_dt, leave_balance, vacation_days
, case when remaining_balance < 0 then 'Insufficient Leave Balance' else 'Approved' end as status
from cte;
```

THE OUTPUT:

Id	Emp_id	Name	From_dt	To_dt	Leave_balance	Vacation_days	Status
1	1	Michael Henry	2/12/2024	2/16/2024	12	5	Approved
2	2	Francis Mansetus	2/20/2024	2/29/2024	10	8	Approved
3	3	John Stewart	3/1/2024	3/31/2024	26	21	Approved
5	4	Saul McGill	6/1/2024	6/30/2024	20	20	Approved
4	1	Michael Henry	4/11/2024	4/23/2024	12	9	Insufficient Leave Balance
6	3	John Stewart	7/5/2024	7/15/2024	26	7	Insufficient Leave Balance
7	3	John Stewart	8/28/2024	9/15/2024	26	13	Insufficient Leave Balance