1. Using the given Salary, Income and Deduction tables, first write an SQL query to populate the Emp\_transaction table

and then generate a salary report

create extension tablefunc;

SELECT employee,

basic, allowance,

others,

(basic+allowance+others)

as gross,

insurance, health, house,

(insurance+health+house)

as total\_deductions,

(basic+allowance+others) -

(insurance+health+house) as net\_pay FROM crosstab('SELECT emp\_name, trns\_type, amount FROM

emp\_transaction ORDER BY emp\_name, trns\_type',

'SELECT DISTINCT trns\_type

FROM emp\_transaction ORDER BY trns\_type') as result

(employee varchar, allowance numeric, basic numeric,

health numeric, house numeric, insurance numeric,

others numeric)

2.You are given a table having the marks of one student in every test. You have to output the tests in which the student has

improved his performance. For a student to improve his performance he has to score more than the previous test. Provide 2

solutions, one including the first test score and second excluding it.

-- OUTPUT 1

SELECT test\_id,marks FROM (

SELECT \*, LAG(marks,1,0) OVER(ORDER BY test\_id) as prev\_test\_marks

FROM student\_tests) A

WHERE A.marks > A.prev\_test\_marks

--OUTPUT 2

SELECT test\_id,marks FROM (

SELECT \*, LAG(marks,1,marks) OVER(ORDER BY test\_id) as prev\_test\_marks

FROM student\_tests) A

WHERE A.marks > A.prev\_test\_marks

3. Write a SQL query to merge products as per customer for each day.

SELECT customer\_id,dates, product\_id::varchar FROM orders

GROUP BY customer\_id,dates,product\_id

UNION

SELECT customer\_id, dates, string\_agg(product\_id::varchar,',')FROM orders

GROUP BY customer\_id,dates

ORDER BY dates, customer\_id, product\_id

4. Write a SQL query to split the hierarchy and show the employees corresponding to their team

with recursive cte as (

select c1.employee as manager1, c2.employee, CONCAT('Team',' ',ROW\_NUMBER() OVER(PARTITION BY c1.employee ORDER BY c1.employee)) as rw FROM company c1

JOIN company c2 ON c1.employee = c2.manager

WHERE c1.manager IS NULL

UNION

SELECT company.manager, company.employee, rw FROM cte

JOIN company ON company.manager = cte.employee

),

cte2 as (

SELECT manager1, rw FROM cte

UNION

SELECT employee, rw FROM cte

)

SELECT rw AS Team, string\_agg(manager1,',') as Members FROM cte2

GROUP BY rw

ORDER BY rw

5. Write a SQL query to find number of employees managed by each manager

SELECT manager\_name AS MANAGER, count(emp\_id) AS NO\_OF\_EMPLOYEES FROM(

select e1.id as emp\_id, e1.name as emp\_name, e2.id as manager\_id,

e2.name as manager\_name from employee\_managers e1

join employee\_managers e2 ON e1.manager = e2.id

ORDER BY e2.name) a

GROUP BY manager\_name

ORDER BY count(emp\_id) DESC

6. Given table contains reported covid cases in 2020. Calculate the percentage increase in covid cases each month versus cumulative cases as of the prior month. Return the month number, and the percentage increase rounded to one decimal. Order the result by the month.

with cte as

(select extract(month from dates) as month

, sum(cases\_reported) as monthly\_cases

from covid\_cases

group by extract(month from dates)),

cte\_final as

(select \*

, sum(monthly\_cases) over(order by month) as total\_cases

from cte)

select month

, case when month > 1

then cast(round((monthly\_cases/lag(total\_cases) over(order by month))\*100,1) as varchar)

else '-' end as percentage\_increase

from cte\_final;

7. Write a SQL query find out the employees who attended all the company events

with cte as (

Select emp\_id, COUNT(DISTINCT(event\_name)) AS No\_of\_events FROM events

Group by emp\_id)

select employees.name, No\_of\_events FROM cte

JOIN employees ON cte.emp\_id = employees.id

WHERE No\_of\_events = (SELECT COUNT(Distinct(No\_of\_events)) from cte)

8. Given table showcases details of pizza delivery order for the year of 2023.If an order is delayed then the whole order

is given for free. Any order that takes 30 minutes more than the order time is considered as delayed order. Identify the

percentage of delayed order for each month and also display the total no of free pizzas given each month.

Sort the result in order of month as shown in expected output

select to\_char(order\_time,'Mon-YYYY') as period,

round((cast(sum(CASE WHEN CAST(to\_char(actual\_delivery - order\_time,'MI') AS INT)>30

THEN 1 else 0 END) as decimal)/count(1))\*100,1) delayed\_flag,

SUM(CASE WHEN CAST(to\_char(actual\_delivery - order\_time,'MI') AS INT)>30

THEN no\_of\_pizzas else 0 END) as free\_pizza

FROM pizza\_delivery

where actual\_delivery is not null

group by to\_char(order\_time,'Mon-YYYY')

order by extract(month from to\_date(to\_char(order\_time,'Mon-YYYY'),'Mon-YYYY'))

9. The column 'perc\_viewed' in the table 'post\_views' denotes the percentage of the session duration time the user spent viewing a post. Using it, calculate the total time that each post was viewed by users. Output post ID and the total viewing time in seconds, but only for posts with a total viewing time of over 5 seconds.

with cte as(

select user\_sessions.session\_id,session\_starttime,session\_endtime,post\_views.\*,

extract('epoch'from (session\_endtime - session\_starttime)) as times from user\_sessions

join post\_views on user\_sessions.session\_id = post\_views.session\_id)

select post\_id, sum((perc\_viewed/100)\*times) as total\_viewtime

from cte

group by post\_id

having sum((perc\_viewed/100)\*times) > 5

10. Given table has details of every IPL 2023 matches. Identify the maximum winning streak for each team.

Additional test cases:

1) Update the dataset such that when Chennai Super Kings win match no 17, your query shows the updated streak.

2) Update the dataset such that Royal Challengers Bangalore loose all match and your query should populate the winning streak as 0

with cte as(

select home\_team as teams from ipl\_results

union

select away\_team as teams from ipl\_results),

cte2 as(

select dates, home\_team, away\_team, teams,result from cte

join ipl\_results on ipl\_results.home\_team = cte.teams or

ipl\_results.away\_team = cte.teams

order by teams, dates),

cte3 as (

select \*,

row\_number() over(partition by teams) as rnt from cte2),

cte4 as (

select \*,

row\_number() over(partition by teams) as rnt2 from cte3

where teams = result),

cte5 as(

select \*, count(rnt-rnt2) over(partition by teams, rnt-rnt2) as win from cte4)

select teams, MAx(win) from cte5

group by teams

Order by MAX(win) desc