----------Q1--------------

create table salary(id integer,emp\_id integer,amount integer,pay\_date timestamp )

select \*from salary

insert into salary values(1,1,9000,'2017-03-31'),(2,2,6000,'2017-03-31'),(3,2,10000,'2017-03-31'),

(4,1,7000,'2017-02-28'),(5,2,6000,'2017-02-28'),(6,2,8000,'2017-02-28')

create table dept(

emp\_id integer,

dept\_id integer)

insert into dept values(1,1),(2,2),(3,2)

select \* from dept

create temporary table t1 AS

(

SELECT TO\_CHAR(pay\_date,'YYYY-MM') as pay\_month, dept\_id,

AVG(amount) OVER(PARTITION BY DATE\_PART('month',pay\_date),dept\_id) as dept\_avg,

AVG(amount) OVER(PARTITION BY DATE\_PART('month',pay\_date)) as comp\_avg

FROM salary as s JOIN dept as d

USING (emp\_id)

)

select \* from t1

SELECT DISTINCT pay\_month, dept\_id,

CASE WHEN dept\_avg > comp\_avg THEN 'higher'

WHEN dept\_avg = comp\_avg THEN 'same'

ELSE 'lower'

END AS comparison

FROM t1

ORDER BY pay\_month DESC

--------------Q2-----------------

create table student(student\_id serial primary key, student\_name varchar(100))

select \* from student

insert into student values(1,'Daniel'),(2,'Jade'),(3,'Stella'),(4,'Jonathan'),(5,'William')

create table exam(

exam\_id integer,

student\_id integer references student(student\_id),

score integer)

insert into exam values

(10,1,70),

(10,2,80),

(10,3,90),

(20,1,80),

(30,1,70),

(30,3,80),

(30,4,90),

(40,1,60),

(40,2,70),

(40,4,80)

select \* from Exam

create temporary table t2 AS (

SELECT student\_id

FROM (

SELECT \*,

MIN(score) OVER main\_window as least,

MAX(score) OVER main\_window as most

FROM exam

WINDOW main\_window AS (PARTITION BY exam\_id)

) as a

where least = score or most = score)

select \* from t2

SELECT DISTINCT student\_id,student\_name

FROM exam JOIN student

USING (student\_id)

WHERE student\_id != ALL(SELECT student\_id FROM t2)

ORDER BY student\_id

--------Q3----------------------

create table stadium(id integer primary key,visit\_date timestamp,people integer)

select \* from stadium

insert into stadium values

(1,'2017-01-01',10),

(2,'2017-01-02',109),

(3,'2017-01-03',150),

(4,'2017-01-04',99),

(5,'2017-01-05',145),

(6,'2017-01-06',1455),

(7,'2017-01-07',199),

(8,'2017-01-08',188)

create temporary table t3 AS (

SELECT id,

visit\_date,

people,

id - ROW\_NUMBER() OVER() AS dates

FROM stadium

WHERE people >= 100)

SELECT t3.id,

t3.visit\_date,

t3.people

FROM t3

LEFT JOIN (

SELECT dates,

COUNT(\*) as total

FROM t3

GROUP BY dates) AS b

ON b.dates = t3.dates

WHERE b.total > 2

----------- Q4-----------------------

create table visits(user\_id integer,visit\_date timestamp)

insert into visits values

(1,'2020-01-01'),

(2,'2020-01-02'),

(12,'2020-01-01'),

(19,'2020-01-03'),

(1,'2020-01-01'),

(2,'2020-01-03'),

(1,'2020-01-04'),

(7,'2020-01-11'),

(9,'2020-01-25'),

(8,'2020-01-28')

select \* from visits

create table transactions(

user\_id integer,

transaction\_date timestamp,

amount integer

)

insert into transactions

values

(1,'2020-01-02',120),

(2,'2020-01-03',22),

(7,'2020-01-11',232),

(1,'2020-01-04',7),

(9,'2020-01-25',33),

(9,'2020-01-25',66),

(8,'2020-01-28',1),

(9,'2020-01-25',99)

WITH RECURSIVE t4 AS(

SELECT visit\_date,

COALESCE(num\_visits,0) as num\_visits,

COALESCE(num\_trans,0) as num\_trans

FROM ((

SELECT visit\_date, user\_id, COUNT(\*) as num\_visits

FROM visits

GROUP BY 1, 2) AS a

LEFT JOIN

(

SELECT transaction\_date,

user\_id,

count(\*) as num\_trans

FROM transactions

GROUP BY 1, 2) AS b

ON a.visit\_date = b.transaction\_date and a.user\_id = b.user\_id)

),

t2 AS (

SELECT MAX(num\_trans) as trans

FROM t4

UNION ALL

SELECT trans-1

FROM t2

WHERE trans >= 1)

SELECT trans as transactions\_count,

COALESCE(visits\_count,0) as visits\_count

FROM t2 LEFT JOIN (

SELECT num\_trans as transactions\_count, COALESCE(COUNT(\*),0) as visits\_count

FROM t4

GROUP BY 1

ORDER BY 1) AS a

ON a.transactions\_count = t2.trans

ORDER BY 1

---------------Q5--------------------------

create table failed(

fail\_date timestamp)

insert into failed

values

('2018-12-28'),

('2018-12-29'),

('2019-01-04'),

('2019-01-05');

create table succeeded(

success\_date timestamp

);

insert into succeeded

values

('2018-12-30'),

('2018-12-31'),

('2019-01-01'),

('2019-01-02'),

('2019-01-03'),

('2019-01-06');

WITH t1

AS (

SELECT \*, 'succeeded ' period\_state,

success\_date - ROW\_NUMBER() OVER (ORDER BY success\_date) \* interval '1 day' grp

FROM succeeded where success\_date between '2019-01-01' and '2019-12-31'

union

SELECT \*, 'failed' period\_state,

fail\_date - ROW\_NUMBER() OVER (ORDER BY fail\_date) \* interval '1 day' grp

FROM failed where fail\_date between '2019-01-01' and '2019-12-31'

)

SELECT min(success\_date) AS from\_date

,max(success\_date) AS to\_date

,period\_state

FROM t1

GROUP BY period\_state , grp

ORDER BY from\_date;

select \* from t1