

## سوال یک

	min	max
Inner join	100	100
Left outer join	100	199
Right outer join	100	100
Full outer join	100	199

## :Inner join

در ابتدا در نظر میگیریم در جدول phones ، همه ی c\_id ها با هم متفاوت است پس ۱۰۰ تا c\_id متفاوت داریم زمانی که این جدول را با جدول customers ، inner join میکنیم به ازای هر c\_id در جدول phone یک c\_id در جدول customers وجود دارد پس در نهایت ۱۰۰ رکورد داریم.

حال فرض میکنیم در جدول phones شخصی داریم که ۱۰۰ شماره تلفن دارد باز هم زمانی که inner join اتفاق می افتد ۱۰۰ تا c\_id با هم مچ میشوند که تنها تفاوتی که اینجا وجود دارد این است که c\_id ها همه یکی است. پس هم در هر دو حالت ۱۰۰ رکورد داریم.

## :Left outer join

باز هم در ابتدا فرض میکنیم در جدول phones همه ی c\_id ها با هم متفاوت هستند زمانی که left outer join میخواهد انجام شود به ازای هر رکورد در جدول customers یک رکورد در جدول phones داریم پس در نهایت ۱۰۰ رکورد میشود.

حال باز فرض میکنیم در جدول phones یک شخص داریم که ۱۰۰ شماره تلفن دارد در این صورت زمانی که left outer join انجام میشود به ازای این شخصی که ۱۰۰ شماره تلفن دارد ۱۰۰ رکورد داریم حال ۹۹ تا c\_id دیگر داریم که به ازای آن ها رکوردی در جدول phones نداریم اما از آنجایی که در این جوبین لازم نیست حتما در جدول دوم رکوردی وجود داشته باشد پس ۹۹ رکورد دیگر داریم که شماره تلفن آن ها null است و در مجموع ۱۹۹ رکورد داریم.

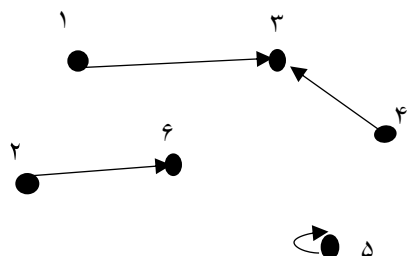
## :Right outer join

حالتی که در جدول phones همه ی c\_id ها با هم متفاوت هستند مشابه دو مورد قبل است اما زمانی که یک شخص داریم که ۱۰۰ شماره تلفن دارد به ازای آن ۱۰۰ رکورد به وجود می آید.

## :Full outer join

دقیقا وضعیت مشابه left outer join پیش می آید.

## سوال دوم

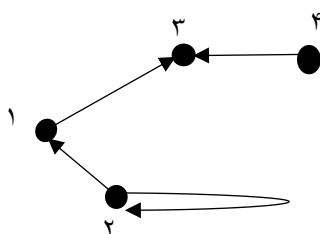


۱	۳	۰
۴	۳	۰
۲	۶	۰
۵	۵	۰

۱	یک	قرمز
۲	دو	سبز
۳	سه	قرمز
۴	چهار	آبی
۵	پنج	سبز
۶	شش	آبی

پرسوجوی اول : اگر یالی وجود داشته باشد که از یک راس خارج شود و به همان راس وارد شود در این صورت چیزی نشان نمیدهد و قسمت not exist غلط میشود اما اگر چنین راسی با چنین مشخصاتی وجود نداشته باشد در این صورت راس هایی که به آن ها یالی وارد شده است را نشان میدهد.

برای پرسوجوی دوم حالت زیر را در نظر میگیریم:



جدول نود	
۱	...
۲	...
۳	...
۴	...

جدول یال	
۱	۳
۴	۳
۲	۱
۲	۲

T1	
۱	۱
۳	۰
۲	۲
۴	۱

T2	
۱	۱
۲	۱
۳	۲
۴	۰

پرسوجوی دوم راس هایی را نشان میدهد که تعداد یال هایی که به آن ها وارد شده اند از تعداد یال هایی که از آن ها خارج شده اند بیشتر است.

## سوال سوم

A

```
-- 3
-- a
create VIEW information_student_instructor AS
select instructor.id,instructor.name,'INS' as type,
case
    when instructor.dept_name like '%Eng.' then 'Engineer'
    ELSE 'Scientist'
end job
from instructor
UNION
select student.id,student.name, 'STU' as type,
case
    when student.dept_name like '%Eng.' then 'Engineer'
    ELSE 'Scientist'
end job
from student;
```

	id character varying (5)	name character varying (20)	type text	job text
1	35935	ODono	STU	Scientist
2	44258	Steinmetz	STU	Scientist
3	22004	OBrien	STU	Scientist
4	27919	Hubr	STU	Scientist
5	11455	Peyse	STU	Scientist
6	28538	Mathur	STU	Scientist
7	95099	Chien	STU	Scientist
8	68010	Blecken	STU	Scientist
9	94522	Pampal	STU	Engineer
10	38476	Rzecz	STU	Scientist

B

```
select information_student_instructor.name,
information_student_instructor.type,
(instructor.salary/department.budget)*100
from information_student_instructor,instructor,department
where information_student_instructor.id = instructor.id
and instructor.dept_name = department.dept_name
and information_student_instructor.type = 'INS'

union

select information_student_instructor.name,
information_student_instructor.type,
(department.budget)/(SELECT count(*) from student where department.dept_name=student.dept_name)
from information_student_instructor,student,department
where information_student_instructor.id = student.id
and student.dept_name = department.dept_name
and information_student_instructor.type = 'STU';
```

	name character varying (20)	type text	?column? numeric
1	Kahs	STU	4647.667529
2	Shuming	INS	11.46424106
3	Jordan	STU	8545.111098
4	Stylian	STU	5052.803361
5	Yoshimoto	STU	9238.854186
6	Gleit	STU	5975.562905
7	Benkov	STU	4463.039595
8	Papakir	STU	5975.562905

a

```

CREATE OR REPLACE FUNCTION check_length()
  RETURNS TRIGGER AS $$
DECLARE
  new_length  BIGINT;
BEGIN

  SELECT INTO new_length film.length
  FROM film
  WHERE film_id = NEW.film_id;

  IF new_length < 50
  THEN
    RAISE EXCEPTION 'error';
  END IF;
  RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER check_length_trigger
AFTER INSERT OR UPDATE ON film
FOR EACH ROW EXECUTE PROCEDURE check_length();

```

```

insert into film(film_id,title,length,language_id)
values (100001,'X',5,1);

```

```

ERROR:  error
CONTEXT:  PL/pgSQL function check_length() line 12 at RAISE
SQL state: P0001

```

b

```

--b
ALTER TABLE payment
add column PAY_TYPE varchar(50)
CHECK (PAY_TYPE in ('credit_card','cash','online'))

```

	payment_id [PK] integer	customer_id smallint	staff_id smallint	rental_id integer	amount numeric (5,2)	payment_date timestamp without time zone	pay_type character varying (50)
1450	18952	116	2	1332	0.99	2007-02-15 10:04:27.996577	[null]
1451	18953	116	2	1533	0.99	2007-02-15 23:14:28.996577	[null]
1452	18954	116	2	1762	4.99	2007-02-16 16:18:45.996577	[null]

```
-- 5
--a

BEGIN;
INSERT INTO department(dept_name,building,budget)
VALUES('medical','Pasteur',700000);
INSERT INTO department(dept_name,building,budget)
VALUES('dental','Pasteur',800000);
COMMIT;

select * from department
where building='Pasteur';
```

	dept_name [PK] character varying (20)	building character varying (15)	budget numeric (12,2)
1	medical	Pasteur	700000.00
2	dental	Pasteur	800000.00

```
--b
BEGIN;
DO $$
DECLARE s INTEGER ;
BEGIN
select budget into s
from department
where department.dept_name='medical';
UPDATE department SET budget = budget + (s*0.1)
WHERE dept_name = 'dental';

UPDATE department SET budget = budget - (s*0.1)
WHERE dept_name = 'medical';
END
$$;
COMMIT;
```



	dept_name [PK] character varying (20)	building character varying (15)	budget numeric (12,2)
1	dental	Pasteur	870000.00
2	medical	Pasteur	630000.00

```

CREATE FUNCTION func_1 (input_actor_id integer)
    returns table(title varchar(255),
                  count_f bigint)
AS $$
BEGIN
    return QUERY
        select film.title,count(*)
        from actor,film_actor,inventory,rental,film
        where actor.actor_id=film_actor.actor_id
        and film_actor.film_id = film.film_id
        and film.film_id = inventory.film_id
        and inventory.inventory_id = rental.inventory_id
        and actor.actor_id=input_actor_id
        group by film.title;

END;|
$$
LANGUAGE 'plpgsql';
select * from func_1(5);

```

	title character varying 	count_f bigint 
1	Enough Raging	18
2	Amadeus Holy	21
3	Heavenly Gun	7
4	Goodfellas Salute	31
5	Daisy Menagerie	16
6	Sunrise League	24
7	Escape Metropolis	25
8	Chitty Lock	8
9	Soldiers Evolution	8
10	Banger Pinocchio	22
11	Coneheads Smoo...	25

```
-- 7
CREATE PROCEDURE proc_6(film_a varchar(255), film_b varchar(255))
LANGUAGE plpgsql AS
$$
DECLARE _r_cost NUMERIC(5,2);
begin
SELECT replacement_cost INTO _r_cost
from film
where title = film_a;
update film
    set replacement_cost = replacement_cost + (0.05*_r_cost)
    where title = film_b;
update film
    set replacement_cost = replacement_cost - (0.05*_r_cost)
    where title = film_a;
end;
$$;

call proc_6 ('Airport Pollock','Bright Encounters');
```

	film_id [PK] integer	title character varying (255)	description text	release_year integer	language_id smallint	rental_duration smallint	rental_rate numeric (4,2)	length smallint	replacement_cost numeric (5,2)	rating mpaa_rating	last_update timestamp without time z
1	8	Airport Pollock	A Epic Tale ...	2006	1	6	4.99	54	15.99	R	2013-05-26 14:50:58.951
2	98	Bright Encounters	A Fateful Y...	2006	1	4	4.99	73	12.99	PG-13	2013-05-26 14:50:58.951

	title character varying (255)	description text	release_year integer	language_id smallint	rental_duration smallint	rental_rate numeric (4,2)	length smallint	replacement_cost numeric (5,2)	rating mpaa_rating
1	Airport Pollock	A Epic Tale of a Moose And a G...	2006	1	6	4.99	54	15.19	R
2	Bright Encounters	A Fateful Yarn of a Lumberjack ...	2006	1	4	4.99	73	13.79	PG-13

```

-- 8
-- a
CREATE OR REPLACE FUNCTION func_r()
    returns trigger
    LANGUAGE PLPGSQL
    as $$
    DECLARE temp INT;
    begin
    select check_count into temp
    from customer
    where customer_id = new.customer_id;
    if temp < 2 THEN
    update customer
    set check_count = check_count+1
    where customer_id = new.customer_id;
    elseif temp=2 THEN

    update rental
    set rental_date = rental_date + interval'7 day'
    where rental_id = new.rental_id;
    update customer
    set check_count = 0
    where customer_id = new.customer_id;
    end if;
    return new;
    end;
    $$

create TRIGGER tri_r
after INSERT
on rental
for each row
when (pg_trigger_depth()<1)
execute procedure func_r();

insert into rental(rental_date,inventory_id,customer_id,return_date,staff_id)
VALUES (now(),5,1,now()+interval'10 day',1)

insert into rental(rental_date,inventory_id,customer_id,return_date,staff_id)
VALUES (now(),6,1,now()+interval'10 day',1)

insert into rental(rental_date,inventory_id,customer_id,return_date,staff_id)
VALUES (now(),7,1,now()+interval'10 day',1)

```



-- 9

```

select film.title, film.rating, rank () over (ORDER BY sum(payment.amount) DESC) as rank_in_all,
rank () over (PARTITION BY film.rating ORDER BY sum(payment.amount) DESC) as rank_in_rating,
sum(payment.amount) as sum_amount,
(CASE
    WHEN(ntile(4) over (order by sum(amount) DESC))=1 THEN 'YES'
    else 'NO'
end
) as is_in_first_quartile
from film, inventory, payment, rental
where film.film_id = inventory.film_id
and inventory.inventory_id = rental.inventory_id
and rental.rental_id = payment.rental_id
group by film.film_id
order by film.title;

```

	title character varying (255)	rating mpaa_rating	rank_in_all bigint	rank_in_rating bigint	sum_amount numeric	is_in_first_quartile text
1	Academy Dinosaur	PG	718	147	33.79	NO
2	Ace Goldfinger	G	508	89	52.93	NO
3	Adaptation Holes	NC-17	699	137	34.89	NO
4	Affair Prejudice	G	261	47	83.79	NO
5	African Egg	G	557	99	47.89	NO
6	Agent Truman	PG	118	24	111.81	YES
7	Airplane Sierra	PG-13	263	64	82.85	NO
8	Airport Pollock	R	239	46	86.85	YES
9	Alabama Devil	PG-13	354	80	71.88	NO
10	Aladdin Calendar	NC-17	66	18	131.77	YES
Total rows: 958 of 958		Query complete 00:00:00.061				

```
--10
```

```
select EXTRACT (month from payment.payment_date) as m , film.rating,
sum(payment.amount) as sum,
lead(sum(payment.amount),-1) OVER (PARTITION by film.rating ORDER by
EXTRACT (month from payment.payment_date)) p_m,
lead(sum(payment.amount),1) OVER (PARTITION by film.rating ORDER by
EXTRACT (month from payment.payment_date)) n_m
from film,inventory,payment,rental
where film.film_id = inventory.film_id
and inventory.inventory_id = rental.inventory_id
and rental.rental_id = payment.rental_id
group by m,film.rating
order by m;
```

	m numeric	rating mpaa_rating	sum numeric	p_m numeric	n_m numeric
1	2	R	1745.78	[null]	4782.76
2	2	PG	1658.99	[null]	4757.52
3	2	NC-17	1667.89	[null]	5085.03
4	2	PG-13	1856.58	[null]	5316.63
5	2	G	1422.60	[null]	3944.62
6	3	PG-13	5316.63	1856.58	6563.76
7	3	R	4782.76	1745.78	5461.78