Bases de Données		Fall, 2024
	TD 8	

triangles (sidelength1 integer, sidelength2 integer, sidelength3 integer)

Each record contains lengths of the three sides of a triangle in 2D. Chaque ligne contient les longueurs des trois côtés d'un triangle en 2D.

Write a query to decide, for each triangle, if a triangle with these three side lengths can actually exist. Écrivez une requête pour décider, pour chaque triangle, si un triangle avec ces trois longueurs de côté peut réellement exister.

Example:

sidelength1	sidelength2	sidelength3
1	5	11
10	21	2
7	1	3
2	5	4
3	7	12

(a) triangles

sidelength1	sidelength2	sidelength 3	exists
1	5	11	No
10	21	2	No
7	1	7	Yes
2	5	4	Yes
3	7	12	No

(b) output

```
SELECT sidelength1, sidelength2, sidelength3,
    CASE
    WHEN sidelength1 <= sidelength2 + sidelength3
        AND sidelength2 <= sidelength1 + sidelength3
        AND sidelength3 <= sidelength1 + sidelength2
        THEN 'Yes'
        ELSE 'No'
    END
FROM triangles</pre>
```

```
points (id integer, x integer, y integer)
```

Each record contains the id of a point in 2D, and its x and y coordinates. Chaque ligne contient l'id d'un point en 2D, ainsi que ses coordonnées x et y.

Write a query to output the pair of points with the shortest distance. Écrivez une requête pour calculer la tuple de points avec la distance la plus courte.

Example:

id	x	y
1	5	11
2 3	21	2
3	1	3
4	5	4
5	7	12
-		

id	id
1	5
(d) output	

(c) points

```
WITH pairwisedistances (id1, id2, dist) AS
    (SELECT A.id, B.id, (A.x-B.x)*(A.x-B.x)+(A.y-B.y)*(A.y-B.y)
    FROM points as A, points as B
    WHERE A.id < B.id)

SELECT A.id1, A.id2
FROM pairwisedistances AS A
WHERE A.dist <= ALL (SELECT B.dist FROM pairwisedistances AS B);</pre>
```

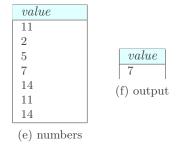
```
numbers (value integer)
```

Each record contains an integer number.

Chaque ligne contient un nombre entier.

Write a query to output the largest number among all the numbers that only appear once in the table. Écrivez une requête pour calculer le plus grand nombre parmi tous les nombres qui n'apparaissent qu'une seule fois dans le tableau.

Example:



```
WITH distinctnums (value) AS
    (SELECT value
    FROM numbers
    GROUP BY value
    HAVING count(*) = 1)

SELECT MAX(value)
FROM distinctnums;
```

```
customer (id integer, product_id integer)
```

Each record is a customer id and a product_id that they bought. Chaque ligne est un id client et un product_id qu'ils ont acheté.

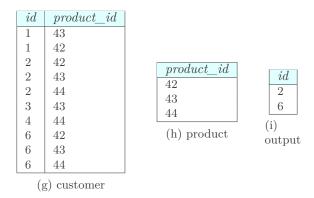
```
product (product_id integer)
```

Each record is a product_id of a product.

Chaque ligne est un product_id d'un produit.

Write a query to output the customers who bought *all* product_id's. Écrivez une requête pour afficher les clients qui ont acheté *tous* product_id.

Example:



```
project (employee_id integer, project_id integer)
```

Each record is an employee_id and a project_id on which the employee worked. Chaque ligne est un employee_id et un project_id sur lesquels l'employé a travaillé.

```
employee (employee_id integer, name text, experience integer)
```

Each record is an employee_id, his name and his years of experience.

Chaque ligne est un employé_id, son nom et ses années d'expérience.

Write a query to output, for each project, the average experience of the employees that worked on it. Écrivez une requête pour afficher, pour chaque projet, l'expérience moyenne des employés qui y ont travaillé.

Example:

$employee_id$	$project_id$
1	23
1	51
2	77
2 2	51
2	23
3	23
4	51
5	23
5	77

(i)	project
(J)	project

$employee_id$	name	experience
1	john	2
2	james	5
3	joe	3
4	joe jamil	10
5	jack	20

(k) employee

$project_id$	average
77	12.5
51	5.6666
23	7.5

(l) output

```
SELECT project_id, avg(experience)
FROM project LEFT OUTER JOIN employee USING (employee_id)
GROUP BY project_id;
```

```
project (employee_id integer, project_id integer)
```

Each record is an employee_id and a project_id on which the employee worked. Chaque ligne est un employee_id et un project_id sur lesquels l'employé a travaillé.

```
employee (employee_id integer, name text, experience integer)
```

Each record is an employee_id, his name and his years of experience. Chaque ligne est un employé_id, son nom et ses années d'expérience.

Write a query to that lists all the projects that have the most employees. Écrivez une requête qui répertorie tous les projets qui ont le plus d'employés.

Example:

emplog	yee_id	project_	$_id$
1		23	
1		51	
2		77	
2		51	
2		23	
3		23	
4		51	
5		23	
5		77	
	(m) pro	oject	

$employee_id$	name	experience
1	john	2
2	james	5
3	joe jamil	3
4	jamil	10
5	jack	20

project_id
23
(o) output

(n) employee

```
project (employee id integer, project id integer)
```

Each record is an employee_id and a project_id on which the employee worked. Chaque ligne est un employee_id et un project_id sur lesquels l'employé a travaillé.

```
employee (employee id integer, name text, experience integer)
```

Each record is an employee_id, his name and his years of experience.

Chaque ligne est un employé_id, son nom et ses années d'expérience.

Write a query to that lists, for each project, the most experienced employees in that project. List all if there is more than one maximum.

Écrivez une requête qui répertorie, pour chaque projet, les employés les plus expérimentés de ce projet. Listez-les tous s'il y a plus d'un maximum.

Example:

$employee_id$	$project_id$
1	23
1	51
2	77
2	51
2	23
3	23
4	51
5	23
5	77

$employee_id$	name	experience
1	john	2
2	james	5
3	joe	3
4	joe jamil	10
5	jack	20
() 1		

(q) employee

$project_id$	$employee_id$	
23	5	
51	4	
77	5	
(r) output		

(p) project

```
WITH mytable (project id, maxexp) AS
        (SELECT project_id, max(employee.experience)
         FROM project INNER JOIN employee USING (employee id)
         GROUP BY project id)
SELECT project_id, employee_id
FROM (mytable INNER JOIN project USING (project id))
           INNER JOIN employee USING (employee id)
WHERE employee.experience = mytable.maxexp
GROUP BY project id, employee id;
```

```
product (product_id integer, new_price integer, change_date date)
```

Each record is a product with a new_price and the date it was changed to this new price.

Chaque ligne est un produit avec un nouveau_prix et la date à laquelle il a été modifié à ce nouveau prix.

Write a query to find the price of all products on the day 05-10-2023 (format: month-day-year). Assume that at the start, each product has price 10.

Écrivez une requête pour trouver le prix de tous les produits le jour 05-10-2023 (format : mois-jour-année). Supposons qu'au départ, chaque produit ait un prix de 10.

Example:

$product_id$	new_price	$change_date$
1	34	10-21-2023
2	412	10-12-2023
3	73	11-15-2023
4	823	03-19-2023
5	31	02-17-2023
6	84	12-22-2023

S	product
10/	product

$product_id$	new_price
1	10
2	10
3	10
4	823
5	31
6	10
6	10

(t) output

```
WITH mytable (product_id, maxdate) AS (
    SELECT product_id, max(change_date)
    FROM product
    WHERE change_date <= '05-10-2023'
    GROUP BY product_id)

SELECT product.product_id,
    (CASE WHEN mytable.product_id IS NULL THEN 10
    ELSE new_price
    END)

FROM mytable RIGHT OUTER JOIN product ON (mytable.product_id,
    mytable.maxdate) = (product.product_id, product.change_date);</pre>
```

```
ascenseur (ordering integer, weight integer)
```

Each record is a person, in order, entering an ascenseur with a given weight in kg.

Chaque ligne correspond à une personne, dans ordre, entrant dans un ascenseur avec un poids donné en kg.

Write a query to find out how many people, in the order in which they arrived, will fit in the ascenseur. Assume that the maximum weight capacity of the ascenseur is 200kg.

Rédigez une requête pour savoir combien de personnes, dans l'ordre d'arrivée, peuvent prendre place dans l'ascenseur. Supposons que la capacité de poids maximale de l'ascenseur soit de 200 kg.

Example:

ordering	weight
1	70
2	80
3	60
4	50
5	75
6	55

ordering 2
(v) output

(u) ascenseur

```
match (host_id integer, host_goals integer, guest_id integer, guest_goals integer)
```

Each record is a football match between host_id and guest_id, and the number of goals scored by both teams.

Chaque ligne est un match de football entre host_id et guest_id, et le nombre de goals marqués par les deux équipes.

Write a query to list the total points for each team, where a team gets:

3 points if it wins a match,

1 point if it is a draw, and

0 points if it loses.

Écrivez une requête pour afficher le total de points de chaque équipe, où une équipe obtient :

3 points si elle remporte un match,

1 point s'il s'agit d'un match nul, et

0 point si elle perd.

Example:

$host_id$	$host_goals$	$guest_id$	$guest_goals$
1	4	2	1
2	2	5	3
2	5	4	5
1	4	2	1
3	1	1	2
5	4	2	1

(w) match

$team_id$	points
1	9
2	1
3	0
4	1
5	6
/ >	

(x) output

```
WITH mytable(id, points) AS (
      (SELECT host_id, (CASE
                          WHEN host_goals > guest_goals THEN 3
                          WHEN host_goals = guest_goals THEN 1
                          ELSE 0
                                 END)
      FROM match)
         UNION ALL
      (SELECT guest_id, (CASE
                          WHEN host_goals > guest_goals THEN 0
                          WHEN host_goals = guest_goals THEN 1
                          ELSE 3 END)
      FROM match) )
SELECT id, sum(points)
FROM mytable
GROUP BY id;
```

```
number (id integer)
```

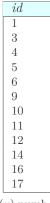
Each record is a positive integer id.

Chaque ligne est un entier positif id.

Write a query to list all maximum sequences of *consecutive numbers*, giving the starting and the ending number of each such subsequence.

Écrivez une requête pour lister toutes les séquences maximales de *nombres consécutifs*, en indiquant le numéro de début et de fin de chacune de ces sous-séquences.

Example:



id_start	id_end	
3	6	
9	12	
(z) output		

(y) number