```
\i script_tables.sql

--1

/*
Les noms et pays des marques de la classe 10 qui n'ont jamais été vend
ues, classées par pays
et par noms.
*/
Select S.nom, S.pays  from marque S
       where classe  = 10 and S.Id not in (select marque  from vente)
order by pays, nom;

 /*
   Nom     | pays
-----------+------
 Exorcyl   | FR
 Exorcyl   | US
 Exorcyl   | UK
 Exorcyl   | MX
 Drobsol   | FR
 Drobsol   | US
 Drobsol   | UK
 Drobsol   | MX
 Cortisol  | FR
 Cortisol  | US
 Cortisol  | DE
 Cortislo  | UK
 Cortisol  | MX
(13 rows)
*/

-- 2
/*
Les noms des villes où sont domiciliées au moins 3 sociétés; on affich
era le nom de la ville, son pays et le nombre de sociétés localisées d
ans la ville en question.

*/
select ville,  pays, count(*) from societe group by ville, pays having
 count(*) >= 3;
/*
     ville      | pays | count
----------------+------+-------
 Boulogne       | FR   |     3
 Heathrow       | UK   |     4
 Hamburg        | DE   |     4
 Paris          | FR   |    10
 New-York City  | US   |     3
(5 rows)
*/

--3
/*
```

```
Les sociétés avec leur nombre de marques françaises; on affichera uniq
uement le nom, les pays et le nombre de marques pour les sociétés qui
possèdent plus de 2 marques françaises. Le libellé de la colonne "nomb
re de marques" dans la table résultante sera "Nb".
*/

select S.nom, S.pays, count(*) AS Nb from marque M, societe S
               where M.prop=S.id and M.pays ='FR'
                    group by S.Id,  S.nom, S.pays
                    having  count(*) >2;

 /*     nom          | pays | nb
--------------------+------+----
 Ste.Christian DIOR | FR   |  3
 Ste. Coca Cola     | FR   |  3
 Ste. GUERLAIN      | FR   |  3
 Ste. Orangina      | FR   |  3
(4 rows)*/

--4
/*
Afficher l'idenditifiant de la ou des sociétés qui ont effectué le plu
s de ventes. On affichera le nom, la ville et le pays et le nombre de
ventes effectuées
*/
select vendeur, count(*) from vente V
       group by V.vendeur
        having  count(*) >= all (select  count(*) from vente V group b
y V.vendeur);
/*
 vendeur | count
---------+-------
      46 |     9
(1 row)
*/
--5

/*
Utilisez une vue pour afficher le nom, la ville et le pays et le nombr
e de ventes effectuées de la ou des sociétés qui ont effectué le plus
de ventes.
*/

DROP VIEW IF EXISTS MaxV ;
CREATE VIEW MaxV As
 (select vendeur, count(*) AS N from vente V group by V.vendeur
 having  count(*) >= all (select  count(*) from vente V group by V.ven
deur)
 );

select S.nom, S.ville, S.pays, MaxV.N
       from  societe S, MaxV
       where S.Id=MaxV.Vendeur;
/*
    nom      |  ville  | pays | n
```

```
------------+---------+------+---
 PEPSI Inc. | Houston | US   | 9
*/

-- Partie  2

CREATE TABLE Employees
    (
        EmployeID  SERIAL PRIMARY KEY,
        Prenom varchar (30)  NOT NULL,
        Nom varchar(40) NOT NULL,
        Titre varchar(50) NOT NULL,
        DeptID smallint NOT NULL,
        ManagerID INTEGER REFERENCES Employees

    );

-- Les données de la table sont les suivantes :


insert into Employees values (1,   'Ken',   'Sanchez',  'Chief Executi
ve Officer',    16          );
insert into Employees values (273, 'Brian',  'Welcke', 'Vice President
 of Sales',    3     ,  1     );
insert into Employees values (16,  'David',   'Bradley',  'Marketing M
anager',      4     , 273    );
insert into Employees values (23,  'Mary',  'Gibson',   'Marketing Spe
cialist',     4     ,16      );
insert into Employees values (274, 'Stephe',  'Jiang',  'North America
n Sales Manager', 3     , 273    );
insert into Employees values (275, 'Michael', 'Blythe',   'Sales Repre
sentative',     3    ,274     );
insert into Employees values (276, 'Linda',   'Mitchell', 'Sales Repre
sentative',     3    , 274    );
insert into Employees values (285,  'Syed',  'Abbas',  'Pacific Sales
Manager',     3    , 273    );
insert into Employees values (286,  'Lynn',  'Tsoflias', 'Sales Repres
entative',      3     , 285    );


-- Ecrire la requête recursive qui retourne pour chaque employée tous
ses supérieurs direct ou indirect
-- classés par hauteur de hierarchie. Sur l'exemple donné le résultat
est le suivant:

WITH recursive Superieurs (EmployeID, ManagerID, Niveau)
AS
    (
-- initialisation
SELECT  e.EmployeID, e.ManagerID, 1 AS Level
FROM Employees AS e

UNION ALL
-- definition recursive
SELECT e.EmployeID,
```

```
       d.ManagerID,
       d.Niveau + 1
FROM Employees AS e
JOIN  Superieurs as d ON e.ManagerID = d.EmployeID
    )
SELECT e1.nom as "EmployÃ©",
       e2.nom as "Superieur",
       s.niveau as "Hauteur de hierarchie"
FROM Superieurs s
        join Employees e1 using(employeid)
        join Employees e2 on (e2.Employeid=s.Managerid);


/*
 Employée  | Superieur | Hauteur de hierarchie
----------+-----------+----------------------
 Welcke    | Sanchez   |                     1
 Bradley   | Welcke    |                     1
 Gibson    | Bradley   |                     1
 Jiang     | Welcke    |                     1
 Blythe    | Jiang     |                     1
 Mitchell  | Jiang     |                     1
 Abbas     | Welcke    |                     1
 Tsoflias  | Abbas     |                     1
 Abbas     | Sanchez   |                     2
 Jiang     | Sanchez   |                     2
 Bradley   | Sanchez   |                     2
 Gibson    | Welcke    |                     2
 Mitchell  | Welcke    |                     2
 Blythe    | Welcke    |                     2
 Tsoflias  | Welcke    |                     2
 Tsoflias  | Sanchez   |                     3
 Mitchell  | Sanchez   |                     3
 Blythe    | Sanchez   |                     3
 Gibson    | Sanchez   |                     3
(19 rows)
*/


-- Donner la requête récursive qui pour l'exemple donné retourne le ré
sultat suivant:
/*
Employe  & Intermediaires   & Grand Chef \\ \hline
Welcker  &                  & Sanchez    \\ \hline
Bradley  & Welcker          & Sanchez    \\ \hline
Abbas    & Welcker          & Sanchez    \\ \hline
Jiang    & Welcker          & Sanchez    \\ \hline
Gibson   & Bradley  Welcker & Sanchez    \\ \hline
Tsoflias & Abbas  Welcker   & Sanchez    \\ \hline
Mitchell & Jiang  Welcker   & Sanchez    \\ \hline
Blythe   & Jiang  Welcker   & Sanchez    \\ \hline
*/


WITH recursive Superieurs (EmployeID,
```

```
                            ManagerID,
                            nomEmploye,
                            nomManager,
                            Chaine,
                            Niveau)
    AS
 (
SELECT  e.EmployeID,  e.ManagerID,   e.nom,  m.nom, '', 1
        FROM Employees AS e  JOIN Employees as m   ON e.managerID=m.em
ployeID

 UNION ALL

SELECT e.EmployeID,  d.ManagerID, e.nom,  d.nomManager,  d.nomEmploye|
| '  ' ||d.chaine, d.Niveau+1
       FROM Employees AS e
       JOIN  Superieurs as d
       ON e.ManagerID = d.EmployeID
    )

SELECT s1.nomEmploye as employe ,
       s1.chaine as "Intermediaires",
           s1.nomManager as "Grand Chef"
    FROM Superieurs s1
        WHERE s1.Niveau=
        (SELECT max(s2.Niveau) FROM Superieurs s2
                WHERE s2.EmployeId=s1.EmployeId);


/*
 employe   |  Intermediaires    | Grand Chef
-----------+--------------------+------------
 Welcke    |                    | Sanchez
 Abbas     | Welcke             | Sanchez
 Jiang     | Welcke             | Sanchez
 Bradley   | Welcke             | Sanchez
 Tsoflias  | Abbas  Welcke      | Sanchez
 Mitchell  | Jiang  Welcke      | Sanchez
 Blythe    | Jiang  Welcke      | Sanchez
 Gibson    | Bradley  Welcke    | Sanchez
(8 rows)
*/

-- Partie : null et jointure
insert into table1 values (1,2);
insert into table1 values (1,3);
insert into table1 values (2,null);
insert into table1 values (null,4);
insert into table1 values (4,4);
insert into table2 select * from table1;

tp1=# select* from table1;
 a | b
---+---
 1 | 2
```

```
 1 | 3
 2 |
   | 4
 4 | 4
(5 rows)

SELECT * FROM table1
   WHERE a>2
   UNION
  SELECT * FROM table1
   WHERE a <=2;

/*
 a | b
---+---
 1 | 2
 1 | 3
 2 |
 4 | 4
*/

Select * FROM table1
         JOIN table2
         ON (table1.a=table2.a);
/*
 a | b | a | b
---+---+---+---
 1 | 2 | 1 | 2
 1 | 2 | 1 | 3
 1 | 3 | 1 | 2
 1 | 3 | 1 | 3
 2 |   | 2 |
 4 | 4 | 4 | 4
(6 rows)

*/

SELECT * FROM table1 FULL JOIN table2 USING (a,b);

/*
 a | b
---+---
 1 | 2
 1 | 3
 2 |
 2 |
 4 | 4
   | 4
   | 4
(7 rows)
*/

select * from table1 t1 full join table2 t2 on t1.a=t2.a and t1.b=t2.b
;
```

```
/*
 a | b | a | b
---+---+---+---
 1 | 2 | 1 | 2
 1 | 3 | 1 | 3
 2 |   |   |
   |   | 2 |
 4 | 4 | 4 | 4
   | 4 |   |
   |   |   | 4

(7 rows)
*/
```