Bases de Données Relationnelles
TP2
MAM4 - SI3

Voici comment a été créée la seule et unique table de la base de données du TP2:

```
CREATE TABLE personne (
    numpers integer PRIMARY KEY,
    nom character varying(30),
    prenom character varying(30),
    pere integer,
    mere integer );
```

Cette table contient des informations sur des personnages. Les valeurs des attributs mere et pere sont les valeurs des attributs numpers de la mère et du père du personnage. si le père ou la mère sont inconnus, ces entiers ne sont pas renseignés.

1 Recherche de motifs

1. Afficher le nom et le prénom de tous les personnages dont le nom contient un 't', ordonné par nom puis par prénom

---

```
SELECT nom, prenom FROM personne
WHERE nom LIKE '%t%'
ORDER BY nom, prenom
```

---

2. Afficher le nom et le prénom de tous les personnages dont le nom ne contient pas de 't', sans tenir compte des majuscules et des minuscules. Ordonner les résultats par nom puis prénom.

```
SELECT nom, prenom FROM personne
WHERE NOT upper(nom) LIKE '%T%'
ORDER BY nom, prenom
```

---

3. Afficher le nom et le prénom des personnages dont le prénom contient un 'y' en deuxième position. Ordonner les résultats par nom puis par prénom.

```
SELECT nom, prenom
FROM personne
WHERE prenom  LIKE '_y%'
ORDER BY nom, prenom
```

---

4. les noms et prénom des personnages dont le prénom contient deux 'n' mais pas deux 'n' consécutifs.

```
SELECT DISTINCT nom, prenom FROM personne
WHERE prenom LIKE '%n%n%' AND NOT prenom LIKE '%nn%'
ORDER BY nom, prenom
```

Autre solution:

```
SELECT nom, prenom FROM personne
WHERE prenom LIKE '%n%n%'
EXCEPT
SELECT nom, prenom FROM personne
WHERE prenom LIKE '%nn%'
ORDER BY nom, prenom
```

---

2 Jointures

5. Afficher pour les 39 personnes de la table personne, leurs nom, prénom ainsi que le nom et le prénom de leur père.

   Les personnages doivent apparaître dans la réponse même si leur père est inconnu (plus précisément inconnu dans cette table).

   Les deux dernières colonnes s'appelleront "Nom du pere" et "Prenom du pere"

   Ordonner les résultats par nom puis prénom.

```
SELECT individu.nom, individu.prenom,
                papa.nom as "Nom␣du␣pere",
                papa.prenom AS "Prenom␣du␣pere"
FROM personne as individu
        LEFT JOIN personne AS papa
        ON individu.pere=papa.numpers
ORDER BY individu.nom, individu.prenom
```

---

6. Afficher pour les 39 personnes de la table personne, leurs nom, prénom ainsi que le nom et le prénom de leur grand-mère paternelle.

   A nouveau tous les personnages doivent apparaître dans la réponse.

   Les deux dernières colonnes s'appeleront "Nom de la grand-mere" et "Prenom de la grand-mere".

   Ordonner les résultats par nom puis prénom.

```
SELECT individu.nom, individu.prenom,
       grandma.nom AS "Nom␣de␣la␣grand-mere" ,
       grandma.prenom AS "Prenom␣de␣la␣grand-mere"
FROM personne AS individu
LEFT JOIN personne AS papa
ON individu.pere=papa.numpers
LEFT JOIN personne AS grandma
ON papa.mere=grandma.numpers
ORDER BY individu.nom, individu.prenom;
```

---

7. Afficher le nom et le prénom des personnages dont on connait le père mais pas le grand-père paternel.

   Ordonner les résultats par nom puis prénom.

```
SELECT individu.nom, individu.prenom
FROM personne AS individu
JOIN personne AS papa
ON individu.pere=papa.numpers
WHERE papa.pere is NULL
ORDER BY individu.nom, individu.prenom;
```

---

8. Afficher pour tous les personnages dont le père est connu, leur nom, leur prénom, le nom de leur père et celui de leur mère (qui peut être inconnue (null)).

   Les deux dernières colonnes seront intitulées "Nom du Père" et "Nom de la Mère".

   Ordonner par nom puis prénom.

```
SELECT individu.nom,
individu.prenom,
papa.nom AS "Nom du Pere",
mama.nom AS "Nom de la Mere"
FROM personne as individu
JOIN personne as papa
ON individu.pere=papa.numpers
LEFT JOIN personne as mama
ON individu.mere=mama.numpers
ORDER BY individu.nom, individu.prenom
```

---

3 Sous-requêtes

9. Ecrire l'ordre d'insertion de Jaime et Tyron Lannister. Leur père est Tywin Lannister, leur mère Joanna Lannister. Leur numPers doit etre plus grand que tous les numPers déjà présents, mais le plus petit possible. Utiliser des sous-requêtes pour calculer les entiers à insérer. Après l'ordre d'insertion, ajouter la requête SELECT * FROM personne ORDER BY numpers; afin de vérifier que les bons tuples ont été insérés.

```
INSERT INTO personne
SELECT MAX(numPers)+1, 'Lannister','Jaime',
       (SELECT papa.numPers
        FROM personne AS papa
        WHERE papa.nom='Lannister' AND papa.prenom='Tywin'),
       (SELECT mama.numPers
        FROM personne AS mama
        WHERE mama.nom='Lannister' AND mama.prenom='Joanna')
FROM personne ;

INSERT INTO personne
SELECT MAX(numPers)+1, 'Lannister','Tyron',
       (SELECT papa.numPers
        FROM personne AS papa
        WHERE papa.nom='Lannister' AND papa.prenom='Tywin'),
       (SELECT mama.numPers
        FROM personne AS mama
        WHERE mama.nom='Lannister' AND mama.prenom='Joanna')
FROM personne ;

SELECT * FROM personne ORDER BY numpers;
```

Autre solution:

```
INSERT INTO personne
VALUES ((SELECT MAX(numPers)+1 FROM personne), 'Lannister','Jaime',
        (SELECT papa.numPers
         FROM personne AS papa
         WHERE papa.nom='Lannister' AND papa.prenom='Tywin'),
        (SELECT mama.numPers
         FROM personne AS mama
         WHERE mama.nom='Lannister' AND mama.prenom='Joanna'));

INSERT INTO personne
VALUES ((SELECT MAX(numPers)+1 FROM personne), 'Lannister','Tyron',
```

```
        (SELECT papa.numPers
         FROM personne AS papa
         WHERE papa.nom='Lannister' AND papa.prenom='Tywin'),
        (SELECT mama.numPers
         FROM personne AS mama
         WHERE mama.nom='Lannister' AND mama.prenom='Joanna'));

SELECT * FROM personne ORDER BY numpers;
```

---

10. A l'aide d'une requête imbriquée afficher le nom et le prénom des personnages dont le père ou la mère est un Lannister.

    Ordonner par nom puis prénom.

```
SELECT nom, prenom FROM personne
WHERE pere IN (SELECT numPers FROM personne WHERE nom='Lannister')
UNION
SELECT nom, prenom FROM personne
WHERE mere IN (SELECT numPers FROM personne WHERE nom='Lannister')
ORDER BY nom, prenom
```

Autre solution:

```
SELECT DISTINCT nom, prenom
FROM personne
WHERE
 pere IN (SELECT numpers FROM personne WHERE nom = 'Lannister')
 OR
 mere IN (SELECT numpers FROM personne WHERE nom = 'Lannister')
ORDER BY nom, prenom
```

Autre solution sans requête imbriquée:

```
SELECT individu.nom, individu.prenom
FROM personne AS individu JOIN personne AS papa ON individu.pere=papa.numPers
WHERE papa.nom='Lannister'
UNION
SELECT individu.nom, individu.prenom
FROM personne AS individu JOIN personne AS mama ON individu.mere=mama.numPers
WHERE mama.nom='Lannister'
ORDER BY individu.nom, individu.prenom
```

---

11. A l'aide d'une requête corrélée afficher le nom et le prénom des personnages dont le père ne porte pas le même nom qu'eux.

    On n'affichera pas le nom des personnages dont le père est inconnu.

    Ordonner par nom puis prénom.

```
SELECT individu.nom, individu.prenom
FROM personne AS individu
WHERE individu.pere NOT IN
   (SELECT papa.numpers
    FROM personne papa
    WHERE papa.nom=individu.nom)
ORDER BY nom, prenom
```

---

12. Même question, mais à résoudre avec une jointure.

    Ordonner par nom puis prénom.

```
SELECT individu.nom, individu.prenom
FROM personne AS individu
JOIN personne AS papa
ON papa.numpers=individu.pere
WHERE papa.nom<>individu.nom OR
(individu.nom is NULL AND papa.nom is NOT NULL) OR
(individu.nom is NOT NULL AND papa.nom is NULL)
ORDER BY individu.nom, individu.prenom
```

---

13. Et voici qu'on s'apperçoit que Jaime Lannister figure deux fois dans la table.

    Supprimer celui des deux qui a le plus petit numéro de personnage.

    Faire suivre l'ordre de suppression de la requête SELECT * from personne; afin de verifier que tout s'est bien passé.

```
DELETE FROM personne
WHERE numpers = (SELECT min(numpers) FROM personne
                 WHERE nom='Lannister' AND prenom='Jaime') ;

SELECT * FROM personne;
```

---

14. Rétablir la vérité historique, en rendant à Jaime Lannister tous les enfants qui sont attribués à Robert Baratheon (avec des sous-requêtes bien sûr).

    Faire suivre votre requête de la requête SELECT * FROM personne; afin de vérifier que la mise à jour a bien été effectuée.

```
UPDATE personne
SET pere=
    (SELECT numpers FROM personne
    WHERE nom='Lannister' AND prenom='Jaime')
WHERE pere=
    (SELECT numpers FROM personne
    WHERE nom='Baratheon' AND prenom ='Robert');
SELECT * FROM personne;
```

---

4 Group By

15. Afficher par ordre alphabétique le nom et le prénom des personnages qui sont des parents avec leur nombre d'enfants.

    Les colonnes s'appeleront nom, prenom et progeniture.

```
SELECT mama.nom, mama.prenom, COUNT(individu.numpers) AS progeniture
FROM personne AS individu
JOIN personne AS mama ON mama.numpers=individu.mere
GROUP BY individu.mere, mama.nom, mama.prenom
UNION
SELECT papa.nom, papa.prenom, COUNT(individu.numpers) AS progeniture
FROM personne AS individu
JOIN personne AS papa ON papa.numpers=individu.pere
GROUP BY individu.pere, papa.nom, papa.prenom
ORDER BY 1, 2;
```

---

16. Afficher par ordre alphabétique le nom et le prénom des personnes avec leur nombre d'enfants (colonne progeniture), 0 si elles n'en ont pas.

Attention Moodle travaille avec sqlite qui ne supporte pas la jointure droite.

```
SELECT mama.nom, mama.prenom, COUNT(individu.numpers) as progeniture
FROM personne as individu JOIN personne as mama on mama.numpers=individu.mere
GROUP BY individu.mere, mama.nom, mama.prenom
UNION
SLECT papa.nom, papa.prenom, COUNT(individu.numpers) as progeniture
FROM personne as individu JOIN personne as papa on papa.numpers=individu.pere
GROUP BY individu.pere, papa.nom, papa.prenom
UNION
SELECT individu.nom, individu.prenom ,'0' as progeniture
FROM personne as individu
WHERE NOT EXISTS (SELECT * FROM personne p1 WHERE p1.pere = individu.numpers OR
ORDER BY 1, 2;
```

Autre solution:

```
SELECT personne.nom , personne.prenom , COALESCE(parent.enfants,0)
FROM personne LEFT JOIN
  (SELECT personne.pere AS numparent, COUNT(personne.numpers) AS enfants
   FROM personne WHERE  personne.pere IS NOT NULL
   GROUP BY personne.pere
   UNION
   SELECT personne.mere AS numparent , COUNT(personne.numpers) AS enfants
   FROM personne WHERE personne.mere IS NOT NULL
   GROUP BY personne.mere) AS parent
ON personne.numpers=parent.numparent
ORDER BY personne.nom, personne.prenom
```

COALESCE(E1,E2,E3) retourne E1 si E1 est non null, E2 si E1 est null mais pas E2, et sinon E3.

---

5 Récursivité

17. Afficher tous les personnages ayant des descendants avec leur nombre de descendants. Les afficher par ordre décroissant de fécondité. Départager les ex aequo par ordre alphabètique.

```
WITH RECURSIVE descendants (ancetre, descendant) AS (
  SELECT pere, numpers FROM personne WHERE pere IS NOT null
  UNION
  SELECT mere, numpers FROM personne WHERE mere is not null
  UNION
  SELECT d.ancetre, p.numpers
  FROM descendants d JOIN personne p
  ON p.pere=d.descendant OR p.mere=d.descendant )
SELECT p1.nom, p1.prenom, COUNT(descendant)
FROM descendants d JOIN personne p1 ON p1.numpers =d.ancetre
GROUP BY p1.nom, p1.prenom
ORDER BY 3 desc, 1, 2;
```

---

18. Afficher par ordre alphabétique le nom et le prénom de tous les descendants de Aegon Targaryen.

```
WITH RECURSIVE descendants (ancetre, descendant) AS (
  SELECT pere, numpers FROM personne WHERE pere IS NOT null
  UNION
  SELECT mere, numpers FROM personne WHERE mere is not null
  UNION
```

```
  SELECT d.ancetre, p.numpers
  FROM descendants d JOIN personne p
  ON p.pere=d.descendant OR p.mere=d.descendant )
SELECT p1.nom, p1.prenom
FROM descendants d JOIN personne p1 ON p1.numpers=d.descendant
WHERE d.ancetre =
  (SELECT P2.numpers
   FROM personne p2
   WHERE p2.nom='Targaryen' AND p2.prenom='Aegon')
ORDER BY p1.nom, p1.prenom;
```
