Corrigé du R2.06-Exploitation BD (Séance n° 4)
Recherche récursive, Division et Requêtes complexes

Première étape : Expression des recherches récursives

Q1 Donner les feuilles de la hiérarchie des thèmes (identifiant, libellé).

```
SELECT      IdT, Libelle
FROM        THEME
WHERE       IdT NOT IN
                    (SELECT IdTPere
                    FROM THEME
                    WHERE IdTPere IS NOT NULL) ;
```

Q2 Donner toute la hiérarchie (identifiant, libellé) de classification du thème Jointure externe. Triez cette liste du thème le plus général au thème le plus spécifique.

```
SELECT      IdT, Libelle
FROM        THEME
CONNECT BY  IdT = PRIOR IdTPere
START WITH  Libelle = 'JOINTURE EXTERNE'
ORDER BY LEVEL DESC ;
```

Q3 Donner les racines de la hiérarchie des thèmes (identifiant, libellé).

```
SELECT      IdT, Libelle
FROM        THEME
WHERE       IdTPere IS NULL ;
```

Q4 Quels sont les thèmes (identifiant, libellé) spécialisant (i.e. plus précis) le thème Langage de requêtes ?

```
SELECT      IdT, Libelle
FROM        THEME
CONNECT BY  PRIOR IdT = IdTPere
START WITH  Libelle = 'LANGAGE DE REQUETES' ;
```

Q5 Quels sont les thèmes (identifiant, libellé) rattachés directement ou pas au thème Jointure à l'exception du thème Jointure imbriquée et de ses sous-thèmes ?

```
SELECT      IdT, Libelle
FROM        THEME
CONNECT BY  PRIOR IdT = IdTPere
AND         Libelle <> 'JOINTURE IMBRIQUEE'
START WITH  Libelle = 'JOINTURE' ;
```

Q6 Quels sont les thèmes (identifiant, libellé) rattachés directement ou pas au thème Jointure à l'exception du thème Jointure imbriquée ?

```
SELECT      IdT, Libelle
FROM        THEME
```

```
WHERE       Libelle <> 'JOINTURE IMBRIQUEE'
CONNECT BY  PRIOR IdT = IdTPere
START WITH  Libelle = 'JOINTURE' ;
```

Q7 Donnez la liste des questions (identifiant, numéro de TP) se rapportant à un thème rattaché directement ou non au thème Jointure.

```
SELECT      QUESTION.IdQ, NumTP
FROM        QUESTION, THEMQUEST
WHERE       QUESTION.IdQ = THEMQUEST.IdQ AND IdT IN
            (SELECT     IdT
            FROM        THEME
            CONNECT BY  PRIOR IdT = IdTPere
            START WITH  Libelle = 'JOINTURE') ;
```

Q8 Quels sont les thèmes (identifiant, libellé) généralisant les thèmes Jointure ou Sélection simple ?

```
SELECT      DISTINCT IdT, Libelle
FROM        THEME
CONNECT BY  IdT = PRIOR IdTPere
START WITH  Libelle IN ('JOINTURE', 'SELECTION SIMPLE') ;
```

Q9 Quels sont les thèmes (identifiant, libellé) généralisant les thèmes Jointure et Sélection simple ?

```
SELECT      IdT, Libelle
FROM        THEME
CONNECT BY  IdT = PRIOR IdTPere
START WITH  Libelle = 'JOINTURE'
INTERSECT
SELECT      DISTINCT IdT, Libelle
FROM        THEME
CONNECT BY  IdT = PRIOR IdTPere
START WITH  Libelle = 'SELECTION SIMPLE' ;
```

```
SELECT      IdT, Libelle
FROM        THEME
WHERE       IDT IN
            (SELECT DISTINCT IdT
            FROM  THEME
            CONNECT BY  IdT = PRIOR IdTPere
            START WITH  Libelle = 'SELECTION SIMPLE')
CONNECT BY  IdT = PRIOR IdTPere
START WITH  Libelle = 'JOINTURE' ;
```

Q10 Quels sont les thèmes (identifiant, libellé) rattachés directement ou pas au thème SQL LDD et qui ne sont utilisés par aucune question ?

```
SELECT      IdT, Libelle
FROM        THEME
WHERE       IdT NOT IN (SELECT IdT FROM THEMQUEST)
CONNECT BY  PRIOR IdT = IdTPere
START WITH  Libelle = 'SQL LDD' ;
```

Q11 Quels sont les thèmes (identifiant, libellé) feuilles de la hiérarchie des thèmes et qui sont rattachés (directement ou pas) à la même racine que celle du thème Jointure.

```
SELECT      IdT, Libelle
FROM        THEME
WHERE       IdT NOT IN (SELECT IdTPere FROM THEME WHERE IDTPere IS NOT NULL)
CONNECT BY  PRIOR IdT = IdTPere
START WITH  IdT IN
```

```
        (SELECT IdT                          -- La sous-requête parcourt tous les
        FROM THEME                           -- ancêtres du theme Jointure et ne
        WHERE IdTPere IS NULL                -- sélectionne que la racine (Cf.
        CONNECT BY IdT = PRIOR IdTPere       -- clause WHERE
        START WITH Libelle = 'JOINTURE') ;
```

Q12 Quels sont les thèmes (identifiant, libellé) se situant dans la même branche que le thème Jointure (au-dessus ou en dessous) ?

```
SELECT      IdT, Libelle,  - (level - 1)
FROM        THEME
CONNECT BY  IdT = PRIOR IdTPere
START WITH  Libelle = 'JOINTURE'
UNION
SELECT      IdT, Libelle, level - 1
FROM        THEME
CONNECT BY  PRIOR IdT = IdTPere
START WITH  Libelle = 'JOINTURE' ;
```

Deuxième étape : Expression des divisions

Q13 Donner les nom et prénom des étudiants ayant fait toutes les questions du TP 1. Donnez les deux paraphrasages et les formulations SQL associées.

Quels sont les étudiants ayant fait autant de questions du TP n° 1 qu'il en existe.

```
SELECT      ETUDIANT.NumEt, Nom, Prenom
FROM        ETUDIANT, EVALUATION, QUESTION
WHERE       ETUDIANT.NumEt = EVALUATION.NumEt AND EVALUATION.IdQ = QUESTION.IdQ AND
            NumTP = 1
GROUP BY    ETUDIANT.NumEt, Nom, Prenom
HAVING      COUNT(*) =
            (SELECT     COUNT(*)
            FROM        QUESTION
            WHERE       NumTP = 1) ;
```

Quels sont les étudiants tels qu'il n'existe aucune question du TP n° 1 qu'ils n'aient pas faite.

```
SELECT      NumEt, Nom, Prenom
FROM        ETUDIANT E
WHERE       NOT EXISTS
            (SELECT     *
            FROM        QUESTION Q
            WHERE       NumTP = 1 AND NOT EXISTS
                        (SELECT     *
                        FROM        EVALUATION
                        WHERE       E.NumEt = EVALUATION.NumEt AND
                                    EVALUATION.IdQ = Q.IdQ)) ;
```

Q14 Quels sont les groupes dans lesquels tous les types de BAC sont représentés ? Donnez un paraphrasage et la formulation SQL associée.

Quels sont les groupes ayant autant de Types de BAC qu'il en existe.

```
SELECT      Groupe
FROM        ETUDIANT
```

```
GROUP BY    Groupe
HAVING      COUNT(DISTINCT TypeBAC) = (SELECT   COUNT(DISTINCT TypeBAC)
                                       FROM     ETUDIANT) ;
```

Q15 Quels sont les groupes pour lesquels au moins un étudiant a été évalué pour chaque TP.

Quels sont les groupes pour lesquels il y a autant de TP évalués qu'il en existe.

```
SELECT      Groupe
FROM        ETUDIANT, EVALUATION, QUESTION
WHERE       ETUDIANT.NumEt = EVALUATION.NumEt AND EVALUATION.IdQ = QUESTION.IdQ
GROUP BY    Groupe
HAVING      COUNT(DISTINCT NumTP) = (SELECT     COUNT(DISTINCT NumTP)
                                     FROM       QUESTION) ;
```

Quels sont les groupes tels qu'il n'existe aucun TP pour lequel ils n'aient pas été évalué.

```
SELECT      DISTINCT Groupe
FROM        ETUDIANT E
WHERE       NOT EXISTS
            (SELECT     *
            FROM        QUESTION Q
            WHERE       NOT EXISTS
                        (SELECT     *
                        FROM        EVALUATION, ETUDIANT, QUESTION
                        WHERE       E.Groupe = ETUDIANT.Groupe AND
                                    EVALUATION.NumEt = ETUDIANT.NumEt AND
                                    EVALUATION.IdQ = QUESTION.IdQ AND
                                    QUESTION.NumTP = Q.NumTP)) ;
```

Troisième étape : Requêtes complexes

Q16 Donner les nom et prénom des étudiants ayant eu (au moins) un résultat 'Faux' au TP numéro 2 et au TP n°3. Donnez trois formulations.

```
SELECT      DISTINCT Nom, Prenom
FROM        ETUDIANT, EVALUATION, QUESTION
WHERE       ETUDIANT.NUMET = EVALUATION.NUMET
            AND EVALUATION.IdQ = QUESTION.IdQ
            AND Resultat = 'FAUX' AND NumTP = 2 AND ETUDIANT.NumEt IN
            (SELECT     ETUDIANT.NumEt
            FROM        ETUDIANT, EVALUATION, QUESTION
            WHERE       ETUDIANT.NUMET = EVALUATION.NUMET
                        AND EVALUATION.IdQ = QUESTION.IdQ
                        AND Resultat = 'FAUX' AND NumTP = 3) ;
```

```
SELECT      DISTINCT Nom, Prenom
FROM        ETUDIANT, EVALUATION E2, QUESTION  Q2, EVALUATION E3,
            QUESTION Q3
WHERE       ETUDIANT.NUMET = E2.NUMET
            AND E2.IdQ = Q2.IdQ
            AND E2.Resultat = 'FAUX' AND Q2.NumTP = 2 AND
            ETUDIANT.NumEt = E3.NumEt AND E3.IdQ = Q3.IdQ
            AND E3.Resultat = 'FAUX' AND Q3.NumTP = 3 ;
```

```
SELECT      Nom, Prenom
FROM        ETUDIANT
WHERE       NumEt IN
            (SELECT     NumEt
            FROM        EVALUATION, QUESTION
            WHERE       EVALUATION.IdQ = QUESTION.IdQ
                        AND Resultat = 'FAUX' AND NumTP = 2
            INTERSECT
            SELECT      NumEt
            FROM        EVALUATION, QUESTION
            WHERE       EVALUATION.IdQ = QUESTION.IdQ
                        AND Resultat = 'FAUX' AND NumTP = 3) ;
```

Q17 Donner les numéros des groupes dont aucun étudiant n'a obtenu un résultat 'Faux' au TP n° 1.

```
SELECT      DISTINCT Groupe
FROM        ETUDIANT
WHERE       Groupe  NOT IN
            (SELECT     Groupe
            FROM        ETUDIANT, EVALUATION, QUESTION
            WHERE       ETUDIANT.NumEt = EVALUATION.NumEt
                        AND EVALUATION.IdQ = QUESTION.IdQ
                        AND Resultat = 'FAUX' AND NumTP = 1) ;
```

Q18 Donner les nom et prénom des étudiants dans le même groupe et ayant le même type de BAC que l'étudiante Marie Dujardin.

```
SELECT      E.Nom, E.Prenom
FROM        ETUDIANT E, ETUDIANT EMD
WHERE       EMD.Nom = 'DUJARDIN' AND EMD.Prenom = 'MARIE' AND
            E.Groupe = EMD.Groupe AND E.TypeBAC = EMD.TypeBAC ;

SELECT      Nom, Prenom
FROM        ETUDIANT
WHERE       (Groupe, TypeBAC) IN
            (SELECT     Groupe, TypeBAC
            FROM        ETUDIANT
            WHERE       Nom = 'DUJARDIN' AND Prenom = 'MARIE') ;
```

Q19 Donner les nom et prénom des étudiants ayant réalisé le nombre de variantes demandées pour une question du TP numéro 2.

```
SELECT      DISTINCT Nom, Prenom
FROM        ETUDIANT, EVALUATION, QUESTION
WHERE       ETUDIANT.NUMET = EVALUATION.NUMET AND NumTP = 2
            AND EVALUATION.IdQ = QUESTION.IdQ
            AND EVALUATION.NbVariantes = QUESTION.NbVariantes ;

SELECT      DISTINCT Nom, Prenom
FROM        ETUDIANT, EVALUATION
WHERE       ETUDIANT.NUMET = EVALUATION.NUMET
            AND (IdQ, NbVariantes) IN
            (SELECT     IdQ, NbVariantes
            FROM        QUESTION
            WHERE       NumTP = 2) ;
```

Q20 Donner la liste des questions (identifiant et numéro de TP) portant à la fois sur le thème 'Jointure imbriquée' et sur le thème 'Fonction agrégative ou statistique'.

```
SELECT      QUESTION.IdQ, NumTP
FROM        QUESTION, THEME, THEMQUEST
WHERE       QUESTION.IdQ = THEMQUEST.IdQ AND THEMQUEST.IdT = THEME.IdT
            AND Libelle = 'JOINTURE IMBRIQUEE'
            AND QUESTION.IdQ IN
            (SELECT     QUESTION.IdQ
            FROM        QUESTION, THEME, THEMQUEST
            WHERE       QUESTION.IdQ = THEMQUEST.IdQ AND
                        THEMQUEST.IdT = THEME.IdT
                        AND Libelle = 'FONCTION AGREGATIVE OU STATISTIQUE') ;
```

Q21 Donner le nombre de points obtenu par Marie Dujardin pour chaque TP.

```
SELECT      NumTP , SUM(EVALUATION.NbPoints)
FROM        ETUDIANT, EVALUATION, QUESTION
WHERE       ETUDIANT.NumEt = EVALUATION.NumEt AND
            EVALUATION.IdQ = QUESTION.IdQ AND Nom = 'DUJARDIN'
            AND PRENOM = 'MARIE'
GROUP BY    NumTP ;
```

Q22 Donner le nom des étudiants ayant eu au moins trois résultats justes à des questions complexes.

```
SELECT      ETUDIANT.NumEt, Nom
FROM        ETUDIANT, EVALUATION, QUESTION
WHERE       ETUDIANT.NumEt = EVALUATION.NumEt AND
            EVALUATION.IdQ = QUESTION.IdQ AND Niveau = 'COMPLEXE'
            AND RESULTAT = 'JUSTE'
GROUP BY    ETUDIANT.NumEt, Nom
HAVING      COUNT(*) >= 3 ;
```

Q23 Pour chaque étudiant ayant au moins une réponse juste au TP 3, donner le nombre total de questions réalisées à tous les TP.

```
SELECT      ETUDIANT.NumEt, Nom, COUNT(*)
FROM        ETUDIANT, EVALUATION
WHERE       ETUDIANT.NumEt = EVALUATION.NumEt AND
            (ETUDIANT.NumEt, IdQ) IN
            (SELECT     NumEt, EVALUATION.IdQ
            FROM        EVALUATION, QUESTION
            WHERE       RESULTAT = 'JUSTE' AND NumTP = 3 AND
                        EVALUATION.IdQ = QUESTION.IdQ)
GROUP BY    ETUDIANT.NumEt, Nom ;
```