## Correction LogiSoft™ (séries 1 et 2)

Série 1 :

**R1 :** le nom et le prénom du salarié le mieux payé.

```
SELECT nomSalarie, prenomSalarie
FROM Salaries
WHERE salaireSalarie >= ALL (SELECT salaireSalarie
                             FROM Salaries)
```

```
SELECT nomSalarie, prenomSalarie
FROM Salaries
WHERE salaireSalarie = (SELECT MAX(salaireSalarie)
                        FROM Salaries)
```

**R2 :** le code et le nom des projets de type Cascade qui ont un budget plus important qu'un des projets de type Agile.

```
SELECT codePorijet, nomProjet
FROM Projets
WHERE typeProjet = 'Cascade'
AND budgetProjet > ANY (SELECT budgetProjet
                        FROM Projets
                        WHERE typeProjet = 'Agile')
```

```
SELECT codeProjet, nomProjet
FROM Projets
WHERE typeProjet = 'Cascade'
AND budgetProjet > (SELECT MIN(budgetProjet)
                    FROM Projets
                    WHERE typeProjet = 'Agile')
```

Il est également possible de faire cette requête avec une auto-inéquijointure

```
SELECT DISTINCT pc.codeProjet, pc.nomProjet
FROM Projets pc
JOIN Projets pa ON pc.budgetProjet > pa.budgetProjet
WHERE pc.typeProjet = 'Cascade'
AND pa.typeProjet = 'Agile'
```

**R3 :** le nom du projet qui possède le plus de salariés.

```
SELECT nomProjet
FROM Projets
NATURAL JOIN EtreAffecte
GROUP BY codeProjet, nomProjet
HAVING COUNT(*) = (SELECT MAX(COUNT(*))
                   FROM EtreAffecte
                   GROUP BY codeProjet);
```

```
SELECT nomProjet
FROM Projets
NATURAL JOIN EtreAffecte
GROUP BY codeProjet, nomProjet
HAVING COUNT(*) >= ALL (SELECT COUNT(*)
                        FROM EtreAffecte
                        GROUP BY codeProjet);
```

```
SELECT nomProjet
FROM Projets
NATURAL JOIN EtreAffecte
GROUP BY codeProjet, nomProjet
HAVING COUNT(*) = (SELECT MAX(nb)
                   FROM (SELECT COUNT(*) AS nb
                         FROM EtreAffecte
                         GROUP BY codeProjet));
```

```
WITH AffectationsProjets AS
     (SELECT codeProjet, nomProjet,
             COUNT(*) AS nbSalaries
      FROM Projets
      NATURAL JOIN EtreAffecte
      GROUP BY codeProjet, nomProjet)
SELECT nomProjet
FROM AffectationsProjets
WHERE nbSalaries = (SELECT MAX(nbSalaries)
                    FROM AffectationsProjets);
```

**R4 :** le nom et prénom des salariés qui ne sont pas affectés à un projet qui contient plus de trois salariés.

```
WITH projetPlusDeTrois AS (SELECT codeProjet
                           FROM Projets
                           NATURAL JOIN EtreAffecte
                           GROUP BY codeProjet
                           HAVING COUNT(*) > 3)
SELECT nomSalarie, prenomSalarie
FROM Salaries s
WHERE NOT EXISTS (SELECT *
                  FROM EtreAffecte ea
                  NATURAL JOIN projetPlusDeTrois
                  WHERE ea.numSalarie = s.numSalarie)
```

**R5 :** le pourcentage de salariés qui ont un salaire supérieur à 3500 €.

```
SELECT riches.rich / total.tot * 100 AS pourcentage
FROM (SELECT COUNT(*) AS tot FROM Salaries) total
CROSS JOIN (SELECT COUNT(*) AS rich FROM Salaries WHERE salaireSalarie> 3500) riches

WITH riches AS (SELECT COUNT(*) AS rich FROM Salaries WHERE salaireSalarie> 3500),
     total AS (SELECT COUNT(*) AS tot FROM Salaries)
SELECT riches.rich / total.tot * 100 AS pourcentage
FROM riches
CROSS JOIN total
```

On peut aussi réaliser cette requête avec des sous-requêtes scalaires dans le SELECT

```
SELECT (SELECT COUNT(*) AS rich FROM Salaries WHERE salaireSalarie> 3500) /
       (SELECT COUNT(*) AS tot FROM Salaries) * 100 AS pourcentage
FROM DUAL
```





**R6 :** le nom et le prénom des salariés qui connaissent toutes les technologies.

```
SELECT nomSalarie, prenomSalarie
FROM Salaries s1
WHERE NOT EXISTS (SELECT *
                  FROM Technologies t2
                  WHERE NOT EXISTS (SELECT *
                                    FROM Connaitre t3
                                    WHERE t3.numSalarie = s1.numSalarie
                                    AND t3.codeTechnologie = t2.codeTechnologie))
```

**R7 :** le nom et le prénom des salariés qui connaissent toutes les technologies de la catégorie Système.

```
SELECT nomSalarie, prenomSalarie
FROM Salaries s1
WHERE NOT EXISTS (SELECT *
                  FROM Technologies t2
                  WHERE categorieTechnologie = 'Système'
                  AND NOT EXISTS (SELECT *
                                  FROM Connaitre t3
                                  WHERE t3.numSalarie = s1.numSalarie
                                  AND t3.codeTechnologie = t2.codeTechnologie))
```

**R8 :** le nom et le prénom des salariés qui connaissent toutes les technologies connues par le salarié Mélanie Zétofrais.

```
SELECT nomSalarie, prenomSalarie
FROM Salaries s1
WHERE NOT EXISTS (SELECT *
                  FROM Connaitre c2
                  NATURAL JOIN Salaries s2
                  WHERE nomSalarie = 'Zétofrais'
                  AND prenomSalarie = 'Mélanie'
                  AND NOT EXISTS (SELECT *
                                  FROM Connaitre t3
                                  WHERE t3.numSalarie = s1.numSalarie
                                  AND t3.codeTechnologie = c2.codeTechnologie))
```

**R9 :** pour chacun des diplômes de la table Diplômes, le nombre de salariés titulaires du diplôme.

```
SELECT nomDiplome, COUNT(numSalarie) AS nb
FROM Diplomes
NATURAL LEFT OUTER JOIN Posseder
GROUP BY referenceDiplome, nomDiplome
ORDER BY nb DESC ;
```

**R10 :** pour chaque client de la table Clients, le nom du client ainsi que le budget moyen de ses projets.

```
SELECT nomClient, AVG(budgetProjet) AS budgetMoyen
FROM Clients c
LEFT JOIN Projets p ON c.numClient = p.numClient
GROUP BY c.numClient, nomClient
ORDER BY budgetMoyen DESC NULLS LAST
```

**R11 :** Pour chaque salarié de la table Salariés, le budget moyen des projets sur lesquels il a été affecté.

```
SELECT nomSalarie, prenomSalarie, ROUND(AVG(budgetProjet),0) AS budgetMoyen
FROM Salaries s
LEFT JOIN EtreAffecte ea ON s.numSalarie = ea.numSalarie
LEFT JOIN Projets p ON ea.codeProjet = p.codeProjet
GROUP BY s.numSalarie, nomSalarie, prenomSalarie
ORDER BY budgetMoyen DESC NULLS LAST
```

**R12 :** Le code et le nom des projets de type Agile qui ont un budget inférieur à un des projets de type Cascade.

```
SELECT DISTINCT pa.codeProjet, pa.nomProjet
FROM Projets pa
JOIN Projets pc ON pa.budgetProjet < pc.budgetProjet
WHERE pc.typeProjet = 'Cascade'
AND pa.typeProjet = 'Agile'
```

On peut également réaliser cette requête avec un quantificateur ou bien avec une fonction

```
SELECT codeProjet, nomProjet
FROM Projets
WHERE typeProjet = 'Agile'
AND budgetProjet < ANY (SELECT budgetProjet
                        FROM Projets
                        WHERE typeProjet = 'Cascade')
```

```
SELECT codeProjet, nomProjet
FROM Projets
WHERE typeProjet = 'Agile'
AND budgetProjet < (SELECT MAX(budgetProjet)
                    FROM Projets
                    WHERE typeProjet = 'Cascade')
```

Série 2 :

**R13 :** le nom et le prénom du salarié qui possède le moins de diplômes.

```
WITH SalariesDiplomes AS
      (SELECT s.numSalarie, nomSalarie, prenomSalarie, COUNT(referenceDiplome) AS nbDiplomes
       FROM Salaries s
       LEFT OUTER JOIN Posseder p ON s.numSalarie = p.numSalarie
       GROUP BY s.numSalarie, nomSalarie, prenomSalarie)
 SELECT nomSalarie, prenomSalarie
 FROM SalariesDiplomes
 WHERE nbDiplomes = (SELECT MIN(nbDiplomes)
                     FROM SalariesDiplomes)

 SELECT nomSalarie, prenomSalarie
 FROM Salaries s
 LEFT OUTER JOIN Posseder p ON s.numSalarie = p.numSalarie
 GROUP BY s.numSalarie, nomSalarie, prenomSalarie
 HAVING COUNT(referenceDiplome) <= ALL (SELECT COUNT(referenceDiplome)
                                        FROM Salaries s
                                        LEFT OUTER JOIN Posseder p ON s.numSalarie = p.numSalarie
                                        GROUP BY s.numSalarie)
```

**R14 :** le code et le nom des projets pour lesquels aucun salarié affecté ne possède plusieurs diplômes.

```
WITH SalariesPlusieursDiplomes AS
      (SELECT numSalarie
       FROM Salaries
       NATURAL JOIN Posseder
       GROUP BY numSalarie
       HAVING COUNT(*) >= 2)
 SELECT codeProjet, nomProjet
 FROM Projets p
 WHERE NOT EXISTS (SELECT *
                   FROM EtreAffecte e
                   NATURAL JOIN SalariesPlusieursDiplomes
                   WHERE p.codeProjet = e.codeProjet);
```

**R15 :** le nom et le prénom des salariés qui possèdent tous les diplômes qu'a obtenu le salarié 'Jean Beaux-Nau'.

```
SELECT nomSalarie, prenomSalarie
FROM Salaries s1
WHERE NOT EXISTS (SELECT *
                  FROM Salaries s2
                  JOIN Posseder p2 ON s2.numSalarie = p2.numSalarie
                  WHERE nomSalarie = 'Beaux-Nau'
                  AND prenomSalarie = 'Jean'
                  AND NOT EXISTS (SELECT *
                                  FROM Posseder p3
                                  WHERE p3.numSalarie = s1.numSalarie
                                  AND p3.referenceDiplome = p2.referenceDiplome)) ;
```

On peut également réaliser cette division comme en 1ère année :

```
SELECT nomSalarie, prenomSalarie
FROM Salaries s1
WHERE NOT EXISTS (SELECT referenceDiplome
                  FROM Salaries s
                  JOIN Posseder p ON s.numSalarie = p.numSalarie
                  WHERE nomSalarie = 'Beaux-Nau'
                  AND prenomSalarie = 'Jean'
                  MINUS
                  SELECT referenceDiplome
                  FROM Posseder p2
                  WHERE p2.numSalarie = s1.numSalarie);
```

```
SELECT nomSalarie, prenomSalarie
FROM Salaries
NATURAL JOIN Posseder
WHERE referenceDiplome IN (SELECT referenceDiplome
                           FROM Posseder
                           NATURAL JOIN Salaries
                           WHERE nomSalarie = 'Beaux-Nau'
                           AND prenomSalarie = 'Jean')
GROUP BY numSalarie, nomSalarie, prenomSalarie
HAVING COUNT(referenceDiplome) = (SELECT COUNT(referenceDiplome)
                                  FROM Posseder
                                  NATURAL JOIN Salaries
                                  WHERE nomSalarie = 'Beaux-Nau'
                                  AND prenomSalarie = 'Jean')
```

**R16 :** pour chaque client, le nombre de salariés différents qui ont travaillé sur ses projets.

```
SELECT nomClient, COUNT(DISTINCT numSalarie) AS nbSalaries
FROM Clients c
LEFT JOIN Projets p ON p.numClient = c.numClient
LEFT JOIN EtreAffecte ea ON ea.codeProjet = p.codeProjet
GROUP BY c.numClient, nomClient
ORDER BY nbSalaries DESC
```





**R17 :** pour chaque salarié, le nom et prénom du salarié ainsi que le nombre (de salariés) subordonnés directs.

```
SELECT chef.nomSalarie, chef.prenomSalarie, COUNT(sub.numSalarie) AS nbSubordonnes
FROM Salaries chef
LEFT JOIN Salaries sub ON chef.numSalarie = sub.numSalarieChef
GROUP BY chef.numSalarie, chef.nomSalarie, chef.prenomSalarie
ORDER BY nbSubordonnes DESC
```

**R18 :** pour chacun des clients de la table Clients, afficher le nombre de projets de plus de 500 000 € qui ont été contractés.

La solution suivante ne marche pas … La séléction est réalisée après la jointure externe.

```
SELECT nomClient, COUNT(*) AS nb
FROM Clients
NATURAL LEFT OUTER JOIN Projets
WHERE budgetProjet > 500000
GROUP BY numClient, nomClient
ORDER BY nb DESC;
```

En effet on obtiendrait le résultat suivant

| NOMCLIENT | COUNT(*) |
|---|---|
| Renault | 2 |
| Peugeot | 1 |
| EDF | 1 |

Il faut en effet réaliser la sélection concernant les projets de plus de 500 000 € avant de faire la jointure externe :

```
SELECT c.nomClient, COUNT(codeProjet) AS nb
FROM Clients c
NATURAL LEFT OUTER JOIN
      (SELECT numClient, codeProjet
       FROM Projets
       WHERE budgetProjet > 500000)
GROUP BY numClient, nomClient
ORDER BY nb DESC;
```

Ou encore

```
WITH ProjetsImportants AS
      (SELECT numClient
       FROM Projets
       WHERE budgetProjet > 500000)
SELECT c.nomClient, COUNT(p.numClient) AS nb
FROM Clients c
LEFT JOIN ProjetsImportants p ON c.numClient = p.numClient
GROUP BY c.numClient, c.nomClient
ORDER BY nb DESC;
```

Ou encore

```
SELECT nomClient, COUNT(p.numClient) AS nb
FROM Clients c
LEFT OUTER JOIN Projets p ON c.numClient = p.numClient
AND budgetProjet > 500000
GROUP BY c.numClient, nomClient
ORDER BY nb DESC;
```

**R19 :** le nom et le budget des 5 projets qui possèdent les plus gros budgets.

```
WITH ValeurCinquiemeBudget AS
      (SELECT MIN(budgetProjet) AS valeur
       FROM (SELECT budgetProjet
             FROM Projets
             ORDER BY budgetProjet DESC)
       WHERE rownum <= 5)
SELECT nomProjet, budgetProjet
FROM Projets
JOIN ValeurCinquiemeBudget ON budgetProjet >= valeur
ORDER BY budgetProjet DESC;
```

```
SELECT p1.nomProjet, p1.budgetProjet
FROM Projets p1
JOIN Projets p2 ON p1.budgetProjet <= p2.budgetProjet
GROUP BY p1.codeProjet, p1.nomProjet, p1.budgetProjet
HAVING COUNT(DISTINCT p2.budgetProjet) <= 5
ORDER BY p1.budgetProjet DESC;
```

```
SELECT p1.nomProjet, p1.budgetProjet
FROM Projets p1
WHERE 5 >= (SELECT COUNT(DISTINCT budgetProjet)
            FROM Projets p2
            WHERE p1.budgetProjet <= p2.budgetProjet)
ORDER BY budgetProjet DESC;
```

Ou encore, comme nous verrons la semaine prochaine ...

```
SELECT nomProjet, budgetProjet
FROM (SELECT nomProjet, budgetProjet, RANK() OVER (ORDER BY budgetProjet DESC) AS rang
      FROM Projets)
WHERE rang <= 5;
```