SQL : Requêtes imbriquées

Dr N. BAME

Syntaxe

Requête imbriquée dans la clause WHERE d'une requête externe:

SELECT …
FROM …
WHERE [Opérande] Opérateur (SELECT …
FROM …
WHERE …)

Trois imbrications (opérateurs) possibles :

- **(A_1 ,…A_n) IN <sous-req>** exprime une inclusion ensembliste
- **EXISTS <sous-req>** exprime une condition d'existence
- **(A_1 ,…A_n) <comp> [ALL|ANY] <sous-req>**
  exprime comparaison avec quantificateur (ANY par défaut)

Opérateur IN

SELECT …

FROM …

WHERE (A_1 ,…,A_n) [NOT] IN (SELECT B_1 ,…,B_n

FROM …

WHERE …)

Sémantique : la condition est **vraie** si le n-uplet désigné par *(A_1 ,…, A_n) de* la requête ***externe*** **appartient** (**n'appartient pas**) au résultat de la requête ***interne***.

Exemple avec IN

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans des villes où il y a des projets de budget inférieur à 50?

Exemple avec IN

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans des villes où il y a des projets de budget inférieur à 50?

```
SELECT Ename
FROM Emp
WHERE City IN (SELECT City
               FROM  Project
               WHERE Budget < 50)
```

Exemple avec IN

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans des villes où il n'y a pas de projets de budget inférieur à 50?

Exemple avec IN

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans des villes où il n'y a pas de projets de budget inférieur à 50?

```
SELECT Ename
FROM Emp
WHERE City NOT IN (SELECT City
                   FROM  Project
                   WHERE Budget < 50)
```

Imbrication de requêtes avec IN : remarques

- La requête interne est effectuée dans un premier temps et son résultat est stocké temporairement en RAM (ou sur disque si trop gros!)

- La requête interne **n'utilise pas** la (les) table(s) de la requête externe

- Le IN étant ensembliste, les n-uplets retournés par la requête interne doivent être de même format que les n-uplet avant le IN (même nombre d'attributs et de même domaine, pas forcément de même nom)

- Le SELECT de la requête externe **ne peut retourner que des attributs appartenant aux tables** placées dans son FROM

Opérateur EXISTS

Q

```
SELECT ...
FROM ...
WHERE
```

[NOT] EXISTS

Q'

```
(SELECT *
 FROM ...
 WHERE P)
```

- *Sémantique procédurale*

  Pour **chaque n-uplet x** de la requête externe Q, exécuter la requête interne Q'; **s'il existe au moins** un [*s'il n'existe aucun*] n-uplet y dans le **résultat** de la requête interne, alors sélectionner x

Imbrication avec EXISTS

- Les deux requêtes sont **corrélées** : la condition **P** dans la requête interne Q' exprime une jointure entre les tables de Q' et les tables de la requête externe Q.

- => requête interne exécutée pour chaque tuple de la requête externe

- **Seule l'existence d'un résultat importe**, d'où l'utilisation de * dans le SELECT de la **requête interne**

- Le SELECT de la **requête externe ne peut retourner que des attributs** appartenant aux tables placées dans son FROM

Exemples avec EXISTS

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans une ville où il y a un projet?

Exemples avec EXISTS

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans une ville où il y a un projet?

```
SELECT Ename FROM Emp
WHERE EXISTS  (SELECT  *
                FROM    Project
                WHERE   Emp.City=Project.City)
```

Pour chaque employé on va vérifier si un projet est localisé dans sa ville.

Si oui, EXISTS vaut vrai et on retourne le nom de l'employé

Exemples avec EXISTS

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans une ville où il n'y a pas de projet localisé ?

Exemples avec EXISTS

Emp(Eno, Ename, Title, City)
Pay(Title, Salary)
Project(Pno, Pname, Budget, City)
Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans une ville sans projet ?

```
SELECT Ename FROM Emp
WHERE NOT EXISTS  (SELECT  *
                   FROM    Project
                   WHERE   Emp.City=Project.City)
```

- Impossible à faire avec condition dans le WHERE

- Exercice : exprimer cette requête à l'aide d'un NOT IN

Exemples avec EXISTS

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des projets ayant le plus grand budget?

Exemples avec EXISTS

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des projets ayant le plus grand budget?

```
SELECT Pname FROM Project P1
WHERE NOT EXISTS (SELECT  *
                  FROM    Project P2
                  WHERE   P1.Budget < P2.Budget)
```

ALL/ANY

SELECT …

FROM …

WHERE $(A_1, \ldots, A_n)$ **θ ALL/ANY (SELECT** $B_1, \ldots, B_n$

FROM …

WHERE …)

- On peut utiliser une comparaison **θ** ϵ {=, <, <=, >, <=, <>} et **ALL () ou ANY ()** : la **condition est alors vraie** si la **comparaison est vraie pour tous les n-uplets /au moins un n-uplet** de la requête interne

- Comment peut-on exprimer **IN** avec **ALL/ANY**

Imbrication avec ALL/ANY

Comme **IN**

- La requête interne est effectuée dans un premier temps
- La requête interne **n'utilise pas les tables de la requête externe**
- Les n-uplets retournés par la requête interne **doivent être au même format** que le **nuplet avant** le **ANY/ALL**
- Le SELECT de la **requête externe ne peut retourner que des attributs** appartenant aux tables placées dans son FROM

Le **ANY/ALL** est obligatoire devant le comparateur car le SELECT interne retourne ***plusieurs nuplets*** auxquels on veut comparer un seul nuplet

- sauf si la requête interne retourne un seul nuplet

Alors que le **IN ne couvre que l'égalité** (appartenance ensembliste), le ANY/ALL permet de faire des inégalités

Exemple avec “ANY”

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des projets Thiessois de budget supérieur à (au moins) un budget de projet Dakarois ?

Exemple avec “ANY”

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des projets Thiessois de budget supérieur à (au moins) un budget de projet Dakarois ?

```
SELECT Pname
FROM Projet
WHERE City = 'Thies'
AND Budget > ANY (SELECT Budget
                  FROM Project
                  WHERE City= 'Dakar' )
```

Pas possible avec un IN... mais comment faire avec une jointure ?

Exemple avec “ALL”

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des professions ayant le plus petit salaire?

Exemple avec “ALL”

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des professions ayant le plus petit salaire?

```
SELECT Title
FROM Pay
WHERE Salary <= ALL (SELECT Salary
                     FROM Pay );
```

Exercice

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans une ville où il y a un projet?

Exercice

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans une ville où il y a un projet?

```
SELECT Ename
FROM Emp
WHERE EXISTS (SELECT *
              FROM Project
              WHERE Emp.City=Project.City)
```

Exprimer avec un IN ou un ANY.

Exercice

Emp(Eno, Ename, Title, City)

Project(Pno, Pname, Budget, City)

Pay(Title, Salary)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans une ville sans projet ?

Exercice

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des employés qui habitent dans une ville sans projet?

```
SELECT Ename
FROM Emp
WHERE NOT EXISTS ( SELECT  *
                   FROM    Project
                   WHERE   Emp.City=Project.City )
```

Exprimer avec NOT IN ou ALL

Exercice

Emp(Eno, Ename, Title, City)

Pay(Title, Salary)

Project(Pno, Pname, Budget, City)

Works(Eno, Pno, Resp, Dur)

- Noms des projets ayant le plus grand budget?

Exercice

Emp(Eno, Ename, Title, City)
Pay(Title, Salary)

Project(Pno, Pname, Budget, City)
Works(Eno, Pno, Resp, Dur)

- Noms des projets ayant le plus grand budget?

```
SELECT Pname
FROM Project P1
WHERE NOT EXISTS ( SELECT  *
                   FROM    Project P2
                   WHERE P1.Budget < P2.Budget )
```

Exprimer avec un NOT IN (si possible)

Puis avec ALL !

Equivalence IN/ANY, NOT IN/ALL

Toute requête avec un IN peut-être écrite avec un ANY:

Donner le nom des employés habitant la ville d'un ingénieur ?

```
SELECT Ename FROM Emp
WHERE City IN (SELECT City FROM Emp WHERE Title='ingénieur')
SELECT Ename FROM Emp
WHERE City = ANY(SELECT City FROM Emp WHERE Title='ingénieur')
```

Toute requête avec un NOT IN peut-être écrite avec un ALL :

Donner le nom des employés habitant une ville sans ingénieur ?

```
SELECT Ename FROM Emp
WHERE City NOT IN (SELECT City FROM Emp WHERE Title='ingénieur')
SELECT Ename FROM Emp
WHERE City <> ALL(SELECT City FROM Emp WHERE Title='ingénieur')
```

Equivalence ANY/EXISTS, ALL/NOT EXISTS

Toute requête avec un ANY peut-être écrite avec un EXISTS :
Donner les travaux durant plus qu'un travail dirigé par 'Prof' ?

```
SELECT * FROM Work
WHERE Dur > ANY (SELECT Dur FROM Work
                 WHERE Resp='Prof')
SELECT * FROM Work W1
WHERE EXISTS (SELECT * FROM Work W2
              WHERE Resp='Prof'
              AND W1.Dur > W2.Dur)
```

Toute requête avec un ALL peut-être écrite avec un NOT EXISTS :
Donner les travaux durant plus que tous les travaux dirigés par 'Prof' ?

```
SELECT * FROM Work WHERE Dur > ALL (SELECT Dur
                                   FROM Work
                                   WHERE  Resp='Prof')
SELECT * FROM Work W1
WHERE NOT EXISTS (SELECT * FROM Work W2
                  WHERE Resp='Prof'
                  AND W1.Dur < W2.Dur)
```