|  |  |
| --- | --- |
|  |  |
| **SUPPORT DE COURS BLOC2** | |

**SQL**

|  |  |  |
| --- | --- | --- |
| **date** | **révision** | |
| 2019 | Timothée Robert : création du document | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  | |  |

|  |  |
| --- | --- |
| Les sous requêtes |  |

## Sources

SQL Server 2014 : F. Brouard & alii, Eyrolles

Discovering SQL: A. Kriegel, Wrox (Wiley)

Beginning SQL : P. Wilson & alii, Wrox (Wiley)

## Objectif

Les sous requêtes permettent de composer ***dynamiquement*** un résultat (ensemble de lignes) qui peut ensuite être utilisé dans le contexte d’une instruction SQL (extraction via un SELECT ou mise à jour).

## Définition requête interne et externe

On appelle requête interne la sous requête et requête externe la requête qui produit la sous requête.

## Types de sous requêtes pour une extraction (SELECT)

|  |  |  |
| --- | --- | --- |
| Dans le WHERE (ou HAVING) | Dans le FROM | Dans le SELECT |
| SELECT …. FROM t1  WHERE *expression operateur*  (SELECT … FROM t2 WHERE …) ; | SELECT ….  FROM t1 a1,  (SELECT … FROM t2 WHERE …) a2  WHERE …; | SELECT …,  (SELECT … FROM t2 WHERE …)  FROM t1  WHERE …; |

*expression operateur* pourra être selon les cas : =, IN, NOT IN, ANY, ALL, EXISTS, NOT EXISTS

## Règles

### Parenthèses

Ecrivez toujours vos sous requêtes entre parenthèses

### Utilisation

Les sous requêtes dans une clause WHERE ou HAVING permettent de programmer des jointures dites procédurales

Les sous requêtes dans une clause FROM permettent de créer une table dérivée à la volée utilisée par exemple dans une jointure

Les sous requêtes dans une clause SELECT permettent de fabriquer une colonne résultat répondant à des besoins avancés.

# Données utilisée pour les exemples

## Modèle relationnel

Pour les exemples on utilise 3 tables en rapport avec les compagnie aérienne, les avions et les pilotes de ligne.

Le modèle relationnel est le suivant :

**T\_compagnie\_cmp** (cmp\_comp, cmp\_pays, cmp\_nom)

Clé primaire : cmp\_comp

**T\_avion\_avi** (avi\_immat, avi\_type, avi\_hvol, cmp\_comp)

Clé primaire : avi\_immat

Clé étrangère: cmp\_comp reference T\_compagnie\_cmp(cmp\_comp)

**T\_pilote\_pil** (pil\_brevet, pil\_prenom, pil\_nom, pil\_hvol, cmp\_comp, pil\_chef)

Clé primaire : pil\_brevet

Clé étrangères:

cmp\_comp reference T\_compagnie\_cmp(cmp\_comp)

pil\_chef reference T\_pilote\_pil(pil\_brevet)

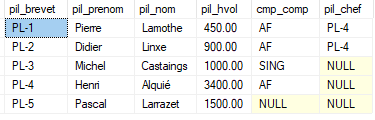
Remarque : le champ pil\_chef permet d’identifier le chef éventuel du pilote (via une self jointure : jointure de la table sur elle-même)

## Tables de données

Ci-dessous le contenu des 3 tables utilisées pour tous les exemples :

|  |  |
| --- | --- |
| Table **T\_compagnie\_cmp** | Table **T\_avion\_avi** |
|  |  |

Table **T\_pilote\_pil**



# Jointures procédurales (Where ou Having)

## Règles

Une sous requête retourne toujours des résultats sous forme d’une ou plusieurs lignes.

Si l’opérateur de jointure utilisé est un ***opérateur de comparaison*** mathématique (=, >, < …) ***alors une seule ligne*** doit être retournée par la sous requête

Si les opérateurs IN, ANY et ALL sont utilisés, alors il est possible que la sous requête retourne plusieurs lignes.

## Limitation

Les colonnes sont ***limitées à celles de la table figurant dans la requête externe*** (autrement dit on ne peut récupérer les colonnes extraites dans une sous requête).

## Sous requêtes monolignes (=, <, >)

|  |  |  |  |
| --- | --- | --- | --- |
| Opérateur | Objectif | Requête (avec sous requête) | Résultat |
| = | Pilotes de la compagnie Air France avec plus de 500 h de vol |  |  |
| Pilotes ayant pour chef le pilote nommé Alquié |  |  |
| > | Pilotes ayant plus d’expérience que le pilote de brevet ‘PL-2’ |  |  |

## Sous requêtes multilignes (IN, ALL et ANY)

### Usage des opérateurs multilignes

#### IN

IN compare un élément à une donnée ramenée par la sous requête.

L’opérateur NOT IN peut être utilisé pour obtenir le même résultat qu’une jointure externe à gauche avec restriction (à condition, sous SQL Server , de filtrer les valeurs NULL de la sous requête, cf exemple).

#### ANY

ANY compare l’élément à chaque donnée ramenée par la sous requête.

L’opérateur =ANY fait la même chose que IN.

L’opérateur <ANY signifie « inférieur à au moins une des valeurs » (donc inférieur au maximum).

L’opérateur >ANY signifie « supérieur à au moins une des valeurs » (donc supérieur au minimum).

#### ALL

ALL compare l’élément à tous ceux ramenés par la sous requête.

<ALL signifie « inférieur au minimum » (de toutes les valeurs ramenées par la sous requête)

>ALL signifie « supérieur au maximum » (de toutes les valeurs ramenées par la sous requête)

### Exemples avec IN et NOT IN

|  |  |  |  |
| --- | --- | --- | --- |
| **Opérateur** | **Demande** | **Requête avec sous requête** | **Résultat** |
| IN | Compagnies avec pilotes qui ont plus de 950 h de vol |  |  |
| = et IN | Somme des heures de vol des pilotes ayant pour chef un pilote d’Air France |  |  |
| NOT IN | Compagnies sans pilote |  |  |

La 2ème requête contient 2 sous requêtes imbriquées, la première avec l’opérateur IN (pour trouver un chef) et la 2ème avec l’opérateur = (qui soit de la compagnie Air France). Elle est équivalente à une requête simple avec 2 jointures (en utilisant 2 fois la table T\_pilote\_pil).

La dernière requête est équivalent à une jointure externe avec restriction entre T\_compagnie\_cmp et T\_pilote\_pil

### Exemples avec ANY et ALL

Non, il ne s’agit pas du film Annie Hall mais de SQL …

#### Exemples opérateur ANY

|  |  |  |
| --- | --- | --- |
| Demande | Requête (avec sous requête) | Résultat |
| Avions dont le nombre d’heures de vol est inférieur à celui de n’importe quel A320 |  |  |
| Compagnies et leurs avions avec un nombre d’heures supérieur à celui de n’importe quel avion de la compagnie SING |  |  |

Rappel : L’opérateur < ANY signifie « inférieur ***à au moins*** une des valeurs » (donc inférieur au maximum).

Les requêtes avec l’opérateur ANY, pour être bien comprises dans un premier temps, nécessitent d’opérer en 2 temps : exécution de la requête interne (sous requête) puis de la requête externe dans un second temps (en utilisant le résultat de la requête interne comme constante).

#### Exemple opérateur ALL

|  |  |  |
| --- | --- | --- |
| Demande / besoin | Requête | Résultat |
| Avions avec un nombre d’heures de vol inférieur à tous les A320 |  |  |
| Compagnies et leurs avions dont le nombre d’heures de vol est supérieur à celui de tous les avions de la compagnie Air France |  |  |

L’opérateur ALL implique que la condition de la requête externe soit validée pour ***toutes les lignes*** ramenées par la requête interne (sous requête).

Pour le 1er exemple, cf les données ramenées par la sous requête :

|  |  |
| --- | --- |
|  |  |

Pour le 2ème exemple, cf les données ramenées par la sous requête :

|  |  |
| --- | --- |
|  |  |

# Sous requête dans la clause FROM

## Introduction

SQL Server autorise comme la norme SQL2 le prévoit, de construire dynamiquement une table qui va être utilisée dans la clause FROM d’une requête.

Cette table dynamique construite dans la requête interne (sous requête) peut ensuite être utilisée dans une jointure ou dans un produit cartésien.

## Exemple

### Besoin

On souhaite savoir combien représente chaque compagnie en pourcentage de nombre d’avions.

### Requête

On décompose l’implémentation en plusieurs phases.

L’utilisation de sous requêtes est nécessaire car les données nécessaires ne sont pas au même niveau de détail (ou de regroupement) : nombre d’avions par compagnie et nombre d’avions total.

#### Nombre d’avions par compagnie

Ci-dessous la requête  et le résultat associé :

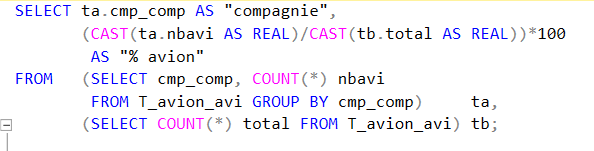
|  |  |
| --- | --- |
|  |  |

#### Nombre d’avions total

Ci-dessous la requête  et le résultat associé :

|  |  |
| --- | --- |
|  |  |

#### Requête finale



Résultat :



Remarque : la fonction SQL Server CAST permet une conversion de type. Par défaut, la donnée d’origine est un entier, dans les 2 cas (nbavi et total) et doit être transformée en REAL pour pouvoir effectuer une division qui va donner un nombre relatif inférieur à 1 (avant d’être multiplié par 100 pour obtenir un pourcentage).

## Sous requête corrélée (synchronisation)

### Définition

Une requête interne ou sous requête est dite corrélée (ou synchronisée) si elle utilise des colonnes de la table de la requête externe (du niveau supérieur).

Elle peut être utilisée dans une requête SELECT , UPDATE ou DELETE.

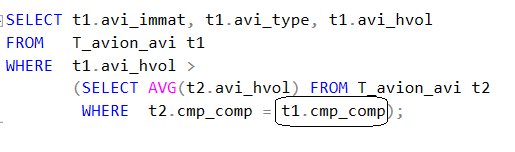
Les opérateurs utilisés peuvent être : =, <, > et EXISTS

### Exemple avec opérateur de comparaison

#### Besoin / demande

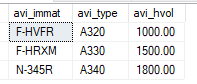
On souhaite disposer des avions pour lesquels le nombre d’heures de vol est supérieur au nombre d’heures de vol moyen de leur compagnie

#### Requête



Remarque : on voit dans la sous requête l’utilisation d’une colonne de la requête externe, entourée par moi. Pour chaque ligne de la requête externe on va vérifier si la valeur de la colonne avi\_hvol est supérieure à la valeur moyenne correspondante (pour la même compagnie).

#### Résultat



Remarque : les valeurs moyennes pour chaque compagnie étaient les suivantes (cf ci-dessous requête et résultat).

|  |  |
| --- | --- |
|  |  |

### Exemple avec opérateurs EXISTS et NOT EXISTS

#### Règles de fonctionnement

L’opérateur EXISTS permet d’interrompre la sous interrogation de la 1ère ligne trouvée vérifiant la condition de la sous requête. Si aucune ligne n’est extraite dans la requête interne (sous requête) alors la valeur FALSE est retournée.

L’opérateur NOT EXISTS retourne la valeur TRUE si aucune ligne n’est extraite dans la requête interne (sous requête).

#### Exemple sous requête corrélée avec EXISTS

Besoin / demande : on souhaite connaître les pilotes qui sont chefs (qui sont responsables d’au moins 1 pilote).

|  |  |
| --- | --- |
|  |  |

Remarque : il y a évidemment d’autres manières de répondre au besoin, via NOT IN et même par jointure simple.

#### Exemple sous requête corrélée avec NOT EXISTS

Besoin / demande : on souhaite connaître les compagnies qui n’ont embauché aucun pilote.

|  |  |
| --- | --- |
|  |  |