



Bases de données

SQL - Sous-requêtes



Sommaire

- 1. Intérêt des sous-requêtes
- 2. Principe des sous-requêtes
- 3. Sous-requêtes simples
- 4. Sous-requêtes corrélées
- 5. Opérateurs ensemblistes
- 6. Division de relations



1. Intérêt des sous-requêtes

■ Exemple

- Donner le titre et l'année du film le plus long
 - le plus long : calcul vertical (max)
 - le titre et l'année : attributs standards
 - → impossible en une seule requête, cf. cours précédent
- Il faut procéder en 2 étapes
 - requête 1 : calculer la durée maximale d'un film
 - requête 2 : afficher les films de cette durée
 - → passage par un « résultat intermédiaire »
- On veut automatiser l'utilisation de résultats intermédiaires
 - → sous-requête



1. Intérêt des sous-requêtes

- Exemple

- Donner le titre et l'année du film le plus long

- ```
SELECT Ftitre, Fannee
FROM film
WHERE Fduree = (SELECT MAX(Fduree)
 FROM film)
```

# 1. Intérêt des sous-requêtes

## ■ Exemple


- Donner le titre et l'année du film le plus long

- `SELECT Ftitre, Fannee`  
`FROM film`

- `WHERE Fduree = (SELECT MAX(Fduree)`  
`FROM film)`

Sous-requête

- cette sous-requête est indépendante (non corrélée)
- → elle est calculée en premier



|             |
|-------------|
| MAX(Fduree) |
| 290         |

# 1. Intérêt des sous-requêtes

## ■ Exemple

- Donner le titre et l'année du film le plus long

- ```
SELECT Ftitre, Fannee
FROM film
WHERE Fduree =
```

290

Sous-requête

- elle est remplacée par sa valeur

MAX(Fduree)

290

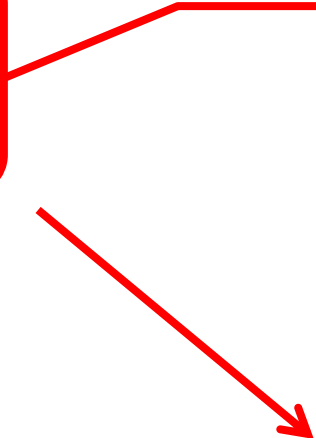
1. Intérêt des sous-requêtes

■ Exemple

- Donner le titre et l'année du film le plus long

- ```
SELECT Ftitre, Fannee
FROM film
WHERE Fduree = 290
```

Requête principale



| Ftitre         | Fannee |
|----------------|--------|
| Les misérables | 1933   |

- La requête principale est exécutée



# 1. Intérêt des sous-requêtes

---

- Les sous-requêtes permettent le calcul de résultats intermédiaires qui sont utilisés dans une requête principale
- ➔ permettent ainsi des calculs qui sont impossibles directement par une simple requête





# Sommaire

---

- 1. Intérêt des sous-requêtes
- 2. Principe des sous-requêtes
- 3. Sous-requêtes simples
- 4. Sous-requêtes corrélées
- 5. Opérateurs ensemblistes
- 6. Division de relations



## 2. Principe des sous-requêtes

---

- Définition de sous-requête (ou requêtes imbriquées)
  - requête SQL (SELECT) apparaissant dans une autre requête
  - Requête interne : la sous-requête
  - Requête externe : celle qui la contient
- Forme de la sous-requête
  - identique aux autres requêtes...
  - ... mais pas destinée à être affichée  
→ pas de clause ORDER BY
  - placée entre parenthèses



## 2. Principe des sous-requêtes

---

### ■ Exécution de la sous-requête

- Soit **une seule fois avant** la requête externe
  - seulement si son résultat ne dépend pas de la requête externe !  
→ sous-requête non corrélée (indépendante)
- Soit **pour chaque ligne** de la table du FROM de la requête externe
  - lorsque son résultat dépend de la requête externe  
→ sous-requête corrélée
  - 👍 puissante (plus grand pouvoir expressif)
  - 👎 coûteuse (ss-requête exécutée plusieurs fois)



## 2. Principe des sous-requêtes

---

- **Position de la sous-requête**
  - dans les différentes clauses de la requête externe
    - sauf `GROUP BY` et `ORDER BY`  
dans lesquelles on ne trouve que des *noms de colonnes*
  - *là où il faut utiliser le résultat intermédiaire !*
- **Aucune limite au nombre d'imbrications**



# Sommaire

---

- 1. Intérêt des sous-requêtes
- 2. Principe des sous-requêtes
- 3. Sous-requêtes simples
- 4. Sous-requêtes corrélées
- 5. Opérateurs ensemblistes
- 6. Division de relations

### 3. Sous-requêtes simples

- **Sous-requête « simple » = non corrélée**

- résultat indépendant de la requête externe
- évaluée 1 seule fois, avant la requête externe
- résultat utilisé dans la requête externe

- **4 types de résultats attendus**

- 1 valeur (1 ligne / 1 colonne)
- 1 colonne (n lignes / 1 colonne)
- 1 ligne (1 ligne / m colonnes)
- 1 table (n lignes / m colonnes)

remarque : les entêtes ne comptent pas


| C | B |
|---|---|
| 2 | a |
|   | c |
|   | a |

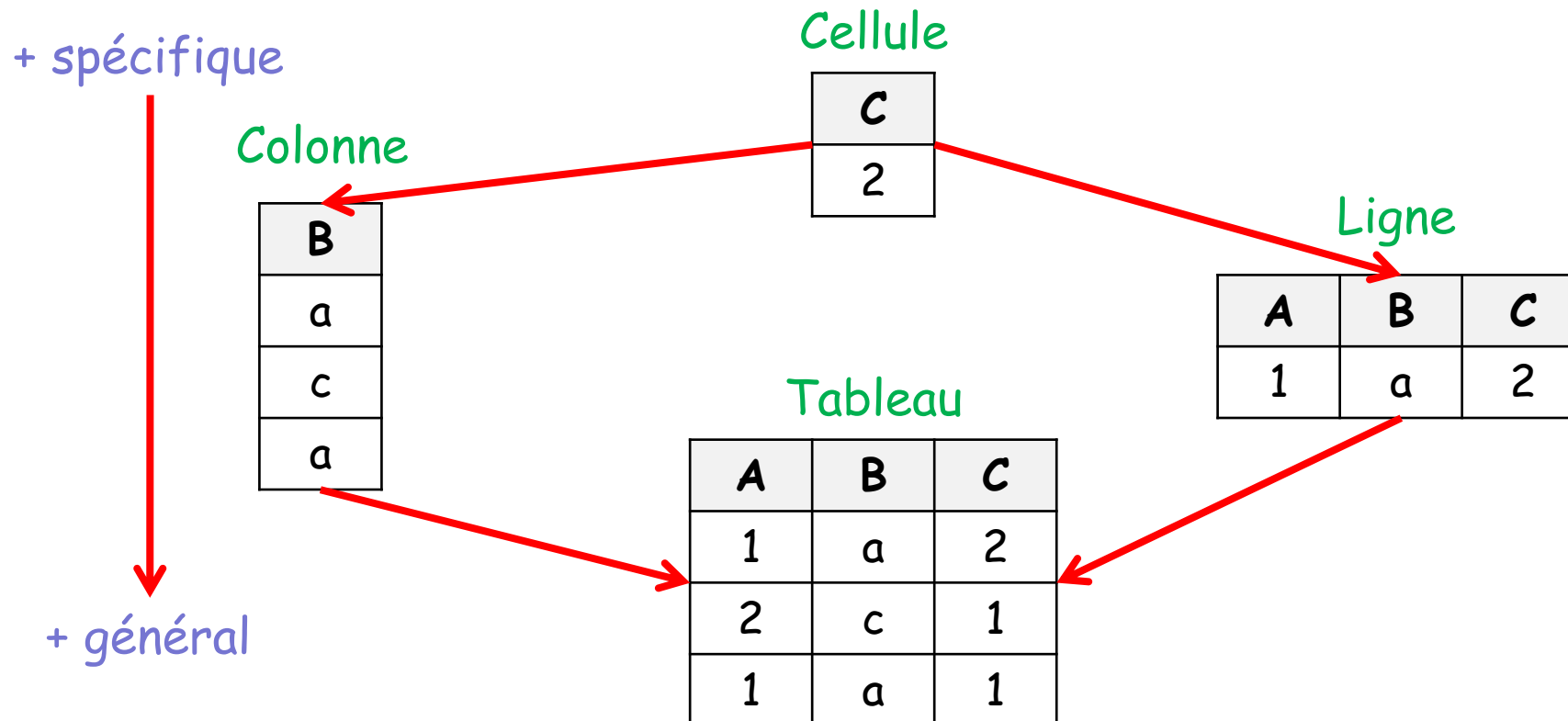
| A | B | C |
|---|---|---|
| 1 | a | 2 |

| A | B | C |
|---|---|---|
| 1 | a | 2 |
| 2 | c | 1 |
| 1 | a | 1 |

### 3. Sous-requêtes simples

- Lien entre les types de résultat

-  relation « est un »
- ce qui marche avec le + général, marche avec le + spécifique





## 3. Sous-requêtes simples

---

- **3.1 Sous-requêtes simples dans WHERE**
  - Cas 1 : 1 ligne / 1 colonne
  - Cas 2 : n lignes / 1 colonne
  - Cas 3 : 1 ligne / n colonnes
  - Cas 4 : m lignes / n colonnes
- **3.2 Sous-requêtes et jointures**
- **3.3 Sous-requêtes simples dans HAVING**
- **3.4 Sous-requêtes simples dans SELECT / ORDER BY**
- **3.5 Sous-requêtes simples dans FROM**





## 3.1 Ss-req simples dans WHERE - cas 1

- **Cas 1 : Résultat = 1 valeur**

|   |
|---|
| C |
| 2 |

- 1 ligne / 1 colonne comme résultat de la sous-requête
- sous-requête utilisée comme une **valeur constante**
- peut apparaître là où on peut mettre une constante
  - le type doit être compatible...



## 3.1 Ss-req simples dans WHERE - cas 1

---

### ■ Exemple

- Donner le titre et l'année du film le plus long
- ```
SELECT Ftitre, Fannee
FROM film
WHERE Fduree = (SELECT MAX(Fduree)
                FROM film)
```
- déjà vu en introduction...
- la sous-requête vaut 290, et peut être utilisée comme 290 (opérateurs de comparaison, calculs...)

3.1 Ss-req simples dans WHERE - cas 2

- **Cas 2 : Résultat = 1 colonne**

- n lignes / 1 colonne comme résultat de la sous-requête
- sous-requête utilisée comme une **liste de valeurs**
 - ensemble de valeurs
 - entre parenthèses
 - séparées par des virgules
- la condition doit utiliser des **opérateurs sur les listes**

B
a
c
a

3.1 Ss-req simples dans WHERE - cas 2

- Sous-requête = liste de valeurs

- Exemple : les identifiants des films français

- ```
SELECT FilmID
FROM film
WHERE Fnat = 'FRANCE'
```



- Équivalent à la liste  
( 'F030' , 'F032' , 'F034' , 'F040' , 'F050' ,  
 'F081' , 'F058' , 'F201' , 'F202' , 'F204' )

| FilmID |
|--------|
| F030   |
| F032   |
| F034   |
| F040   |
| F050   |
| F081   |
| F058   |
| F201   |
| F202   |
| F204   |



## 3.1 Ss-req simples dans WHERE - cas 2

---

- Opérateurs sur les listes

- IN

- test d'appartenance à la liste
    - vrai si la valeur de l'attribut appartient à la liste

- `<attr> IN <liste valeurs>`

- Exemple de condition

- `Fnat IN (SELECT DISTINCT Anat FROM artiste WHERE ...)`
    - `Fnat IN ('FRANCE', 'USA', 'CANADA')`

- NOT IN

- test de non appartenance à la liste
    - `<attr> NOT IN <liste valeurs>`

## 3.1 Ss-req simples dans WHERE - cas 2

### ■ Opérateurs sur les listes

#### ■ ALL

- associé à un opérateur de comparaison
- vrai si la comparaison est vraie **pour toutes** les valeurs de la liste

■ `<attr> <op_comp> ALL <liste valeurs>`

Obtenue par sous-requête

#### ■ Exemple de condition

- `Fannee > ALL (SELECT DISTINCT Amort FROM artiste...)`
- `Fannee > ALL (1985, 1992, 1979)`
- vrai si strictement plus grand que les 3 valeurs

#### ■ il existe des équivalences

- ex : `<> ALL` est équivalent à `NOT IN`

Attention : ne peut pas être testé !  
> ALL valide seulement avec une sous-requête

## 3.1 Ss-req simples dans WHERE - cas 2

### ■ Opérateurs sur les listes

#### ■ ANY / SOME (synonymes)

- associé à un opérateur de comparaison
- vrai si la comparaison est vraie **pour une** des valeurs de la liste

■ `<attr> <op_comp> ANY <liste valeurs>`

Obtenue par sous-requête

#### ■ Exemple de condition

- `Fannee > ANY (SELECT DISTINCT Amort FROM artiste...)`
- `Fannee > ANY (1985, 1992, 1979)`
- vrai si strictement plus grand qu'une des 3 valeurs (au moins)

#### ■ il existe des équivalences

- ex : `= ANY` est équivalent à `IN`

Attention : ne peut pas être testé !  
`> ANY` valide seulement avec une sous-requête



## 3.1 Ss-req simples dans WHERE - cas 2

---

### ■ Exemple 1

- Les réalisateurs des films de 3 heures 30 ou plus
- ```
SELECT Anom, Aprenom
FROM artiste
WHERE ArtisteID IN (SELECT DISTINCT FrealisateurID
                    FROM film
                    WHERE Fduree >= 210)
```
- remarque : peut se faire sans sous-requête...

3.1 Ss-req simples dans WHERE - cas 2


■ Exemple 1

- Les réalisateurs des films de 3 heures 30 ou plus

- `SELECT Anom, Aprenom`
`FROM artiste`
`WHERE ArtisteID IN`

Sous-requête

```
(SELECT DISTINCT FrealisateurID  
FROM film  
WHERE Fduree >= 210)
```



FrealisateurID
A032
A035
A005

3.1 Ss-req simples dans WHERE - cas 2

■ Exemple 1

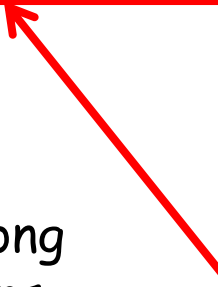
- Les réalisateurs des films de 3 heures 30 ou plus

- `SELECT Anom, Aprenom`
`FROM artiste`
`WHERE ArtisteID IN ('A032', 'A035', 'A005')`

- Sans `DISTINCT` dans la ss-requête

- résultat inchangé
- mais temps de calcul généralement plus long car recherche dans une liste avec doublons

```
('A032', 'A032', 'A032', 'A035', 'A035', 'A035', 'A032',  
'A032', 'A035', 'A035', 'A005', 'A032', 'A035', ...)
```




FrealisateurID
A032
A035
A005

3.1 Ss-req simples dans WHERE - cas 2

■ Exemple 1

- Les réalisateurs des films de 3 heures 30 ou plus

```
SELECT Anom, Aprenom  
FROM artiste  
WHERE ArtisteID IN ('A032', 'A035', 'A005')
```

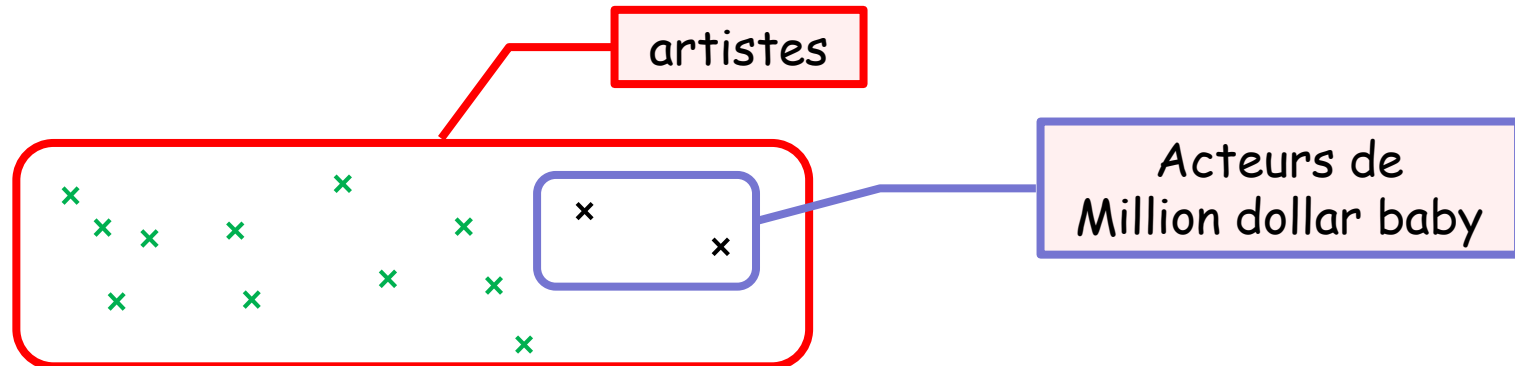


Anom	Aprenom
LEONE	Sergio
BERNARD	Raymond
LE CHANOIS	Jean-Paul

3.1 Ss-req simples dans WHERE - cas 2

- Exemple 2 : complémentaire d'un ensemble
 - Les artistes qui n'ont pas joué dans « Million dollar baby »
 - 1. construire la liste des artistes qui ont joué dedans
 - 2. donner tous les artistes sauf ceux-là

Les bonnes réponses
sont les croix vertes !



- Le calcul des acteurs des autres films que « million dollar baby » ne donne pas la bonne réponse !...
- ... car certains acteurs de « million dollar baby » ont aussi joué dans d'autres films



3.1 Ss-req simples dans WHERE - cas 2

- **Exemple 2 : complémentaire d'un ensemble**
 - Les artistes qui n'ont pas joué dans « Million dollar baby »
 - 1. construire la liste des artistes qui ont joué dedans
 - 2. donner tous les artistes sauf ceux-là
 - ```
SELECT Anom, Aprenom
FROM artiste
WHERE ArtisteID NOT IN
 (SELECT JArtisteID
 FROM film JOIN joue ON FilmID = JFilmID
 WHERE Ftitre = 'Million dollar baby')
```

## 3.1 Ss-req simples dans WHERE - cas 3

- **Cas 3 : Résultat = 1 ligne**

| A | B | C |
|---|---|---|
| 1 | a | 2 |

- 1 ligne / n colonnes comme résultat de la sous-requête
- dans WHERE, sous-requête utilisée comme un **n-uplet**
  - ensemble de valeurs qui peuvent être de nature différente
  - entre parenthèses
  - séparées par des virgules
- Ex : ('Clint', 'Eastwood')  
(Aprenom, Anom)

(1, 'a', 2)



## 3.1 Ss-req simples dans WHERE - cas 3

### ■ Exemple

- Les films de même genre et de même nationalité que Titanic

- ```
SELECT Ftitre
FROM film
WHERE (Fgenre, Fnat) =
      (SELECT Fgenre, Fnat
       FROM film
       WHERE Ftitre = 'Titanic')
```

- on suppose qu'il n'y a qu'un seul film de titre « titanic » !

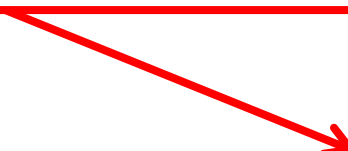
3.1 Ss-req simples dans WHERE - cas 3

■ Exemple

- Les films de même genre et de même nationalité que Titanic

```
SELECT Ftitre
FROM film
WHERE (Fgenre, Fnat) =
```

```
(SELECT Fgenre, Fnat
FROM film
WHERE Ftitre = 'Titanic')
```



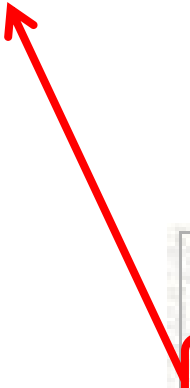
Fgenre	Fnat
Drame	USA

3.1 Ss-req simples dans WHERE - cas 3

■ Exemple

- Les films de même genre et de même nationalité que Titanic

```
SELECT Ftitre  
FROM film  
WHERE (Fgenre, Fnat) = ('Drame', 'USA')
```



Fgenre	Fnat
Drame	USA

3.1 Ss-req simples dans WHERE - cas 3

■ Exemple

- Les films de même genre et de même nationalité que Titanic

- ```
SELECT Ftitre
FROM film
WHERE (Fgenre, Fnat) = ('Drame', 'USA')
```

| Ftitre                        |
|-------------------------------|
| Million dollar baby           |
| Eyes wide shut                |
| La nuit du chasseur           |
| Titanic                       |
| Atlantique latitude 41        |
| Gangs of New-York             |
| Il était une fois en Amérique |

## 3.1 Ss-req simples dans WHERE - cas 3

- Remarque : utilité du n-uplet

```
SELECT Ftitre
FROM film
WHERE (Fgenre, Fnat) = (SELECT Fgenre, Fnat
 FROM film
 WHERE Ftitre = 'Titanic')
```

peut s'écrire avec 2 sous-requêtes (mais double calcul...)

```
SELECT Ftitre
FROM film
WHERE Fgenre = (SELECT Fgenre
 FROM film
 WHERE Ftitre = 'Titanic')
AND Fnat = (SELECT Fnat
 FROM film
 WHERE Ftitre = 'Titanic')
```



## 3.1 Ss-req simples dans WHERE - cas 3

- **Comparaison de n-uplets**

- seulement égalité et différence

- = et <> (ou !=)

- car il n'existe **pas d'ordre total** entre les n-uplets en général

- certains ne sont pas comparables selon < <= > >=

- **Exemple** : qui est le « plus grand » de (A, 2) et (B, 1) ?

- (A, 2) selon le 2<sup>e</sup> critère (le chiffre)
    - (B, 1) selon le 1<sup>er</sup> critère (la lettre)
    - → au final ils ne sont pas comparables !

## 3.1 Ss-req simples dans WHERE - cas 4

| A | B | C |
|---|---|---|
| 1 | a | 2 |
| 2 | c | 1 |
| 1 | a | 1 |

### ■ Cas 4 : Résultat = 1 table

- m lignes / n colonnes comme résultat de la sous-requête
- dans WHERE, sous-requête utilisée comme **liste de n-uplets**

■ ex : **(('Clint', 'Eastwood'), ('Stanley', 'Kubrick'))**

■ = liste de 2 n-uplets (Aprenom, Anom)

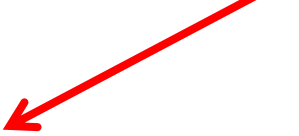
| Aprenom | Anom     |
|---------|----------|
| Clint   | EASTWOOD |
| Stanley | KUBRICK  |

## 3.1 Ss-req simples dans WHERE - cas 4

### ■ Exemple

- Les films de même genre et de même nationalité que ceux tournés par Kubrick

- ```
SELECT Ftitre
FROM film
WHERE (Fgenre, Fnat) IN
      (SELECT DISTINCT Fgenre, Fnat
       FROM film JOIN artiste ON
           FrealisateurID = ArtisteID
       WHERE Anom = 'Kubrick')
```



3.1 Ss-req simples dans WHERE - cas 4

■ Exemple

- Les films de même genre et de même nationalité que ceux tournés par Kubrick

```
■ SELECT Ftitre  
FROM film  
WHERE (Fgenre, Fnat) IN
```

Fgenre	Fnat
Drame	USA
Fantastique	USA
SF	UK


```
(SELECT DISTINCT Fgenre, Fnat  
FROM film JOIN artiste ON  
FrealisateurID = ArtisteID  
WHERE Anom = 'Kubrick')
```

3.1 Ss-req simples dans WHERE - cas 4

■ Exemple

- Les films de même genre et de même nationalité que ceux tournés par Kubrick

```
■ SELECT Ftitre
FROM film
WHERE (Fgenre, Fnat) IN
      ( ('Drame', 'USA'),
        ('Fantastique', 'USA'),
        ('SF', 'UK') )
```



Fgenre	Fnat
Drame	USA
Fantastique	USA
SF	UK

3.1 Ss-req simples dans WHERE - cas 4

■ Exemple

- Les films de même genre et de même nationalité que ceux tournés par Kubrick

```
■ SELECT Ftitre
FROM film
WHERE (Fgenre, Fnat) IN
      ( ('Drame', 'USA'),
        ('Fantastique', 'USA'),
        ('SF', 'UK') )
```



Ftitre
Million dollar baby
Eyes wide shut
La nuit du chasseur
Titanic
Atlantique latitude 41
Shining
Gangs of New-York
Il était une fois en Amérique
Orange mécanique
Dr Jekyll et Mr Hyde



3. Sous-requêtes simples

- 3.1 Sous-requêtes simples dans WHERE

- Cas 1 : 1 ligne / 1 colonne
- Cas 2 : n lignes / 1 colonne
- Cas 3 : 1 ligne / n colonnes
- Cas 4 : m lignes / n colonnes

→ ■ 3.2 Sous-requêtes et jointures

- 3.3 Sous-requêtes simples dans HAVING

- 3.4 Sous-requêtes simples dans SELECT / ORDER BY

- 3.5 Sous-requêtes simples dans FROM



3.2 Sous-requêtes et jointures

- **Remplacer une jointure par une sous-requête**
 - cas particulier d'utilisation d'une sous-requête dans WHERE
 - pas toujours possible
 - souvent plus simple à écrire
 - e.g. évite le renommage dans les auto-jointures
 - moins efficace en machine en général
 - car liste fournie par ss-req. pas indexée
 - mais SGBD souvent capables de transformer
 - méthode systématique de passage de l'un à l'autre
→ simple à mettre en œuvre



3.2 Sous-requêtes et jointures

- **Exemple**

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre  
FROM film JOIN artiste ON FrealisateurID = ArtisteID  
WHERE Anat = 'USA' AND Fannee >= 1980
```

3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```

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

```
(SELECT ...
FROM ...
WHERE )
```

- Pb 1 : 2 tables, 2 requête. Laquelle, où ?

3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```

```
SELECT Ftitre
FROM film
WHERE
```

```
(SELECT ...
FROM artiste
WHERE )
```

- Pb 1 : 2 tables, 2 requête. Laquelle, où ?
- affichage d'une colonne de film → film dans la req. principale

3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```

```
SELECT Ftitre
FROM film
WHERE
```

```
(SELECT ...
FROM artiste
WHERE )
```

- Pb 2 : lien entre les 2 tables ?

3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```

```
SELECT Ftitre
FROM film
WHERE FrealisateurID      (SELECT ArtisteID
                             FROM artiste
                             WHERE
```

- Pb 2 : lien entre les 2 tables ?
- lien clé étrangère / clé primaire associée → idem dans ss-req

3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```

```
SELECT Ftitre
FROM film
WHERE FrealisateurID      (SELECT ArtisteID
                             FROM artiste
                             WHERE
```

- Pb 3 : opérateur de liaison ?

3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```



```
SELECT Ftitre
FROM film
WHERE FrealisateurID IN (SELECT ArtisteID
                        FROM artiste
                        WHERE
```

- Pb 3 : opérateur de liaison ?
- ss-req donne une liste (cas général) → = devient IN

3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```

```
SELECT Ftitre
FROM film
WHERE FrealisateurID IN (SELECT ArtisteID
                        FROM artiste
                        WHERE
```

- Pb 4 : où mettre les conditions de sélection ?

3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```

```
SELECT Ftitre
FROM film
WHERE FrealisateurID IN (SELECT ArtisteID
                        FROM artiste
                        WHERE Anat = 'USA')
AND Fannee >= 1980
```

- Pb 4 : où mettre les conditions de sélection ?
- dans la requête qui contient la table concernée

3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```

```
SELECT Ftitre
FROM film
WHERE FrealisateurID IN (SELECT ArtisteID
                        FROM artiste
                        WHERE Anat = 'USA')
AND Fannee >= 1980
```

- Pb 5 : s'il y avait eu `SELECT DISTINCT`, quid du `DISTINCT` ?
- le plus souvent, se place dans la sous-requête
 - → production d'une liste sans doublons → `IN` plus efficace



3.2 Sous-requêtes et jointures

■ Exemple

- les titres des films réalisés par un américain après 1980

```
SELECT Ftitre
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```

```
SELECT Ftitre
FROM film
WHERE FrealisateurID IN (SELECT ArtisteID
                        FROM artiste
                        WHERE Anat = 'USA')
AND Fannee >= 1980
```



3.2 Sous-requêtes et jointures

- Peut-on toujours remplacer une jointure par une sous-requête ?



3.2 Sous-requêtes et jointures

- Peut-on toujours remplacer une jointure par une sous-requête ?
- **Non !**
 - seulement si attributs d'une seule des tables utilisés dans la requête principale (dans SELECT, ORDER BY...)
 - et attention à la condition du WHERE...
- **Contre-exemple**
 - les titres et réalisateurs des films réalisés par un américain après 1980

```
SELECT Ftitre, Anom, Aprenom
FROM film JOIN artiste ON FrealisateurID = ArtisteID
WHERE Anat = 'USA' AND Fannee >= 1980
```
 - nécessite *film* et *artiste* dans la requête principale → jointure



Questions ?
