Bases de données
Ecole doctorale - Année 2023-2024
Corrigé

Sylvie Damy

13 mai 2024

1 Requêtes SQL

1.1 Ecriture en SQL de requêtes simples

1. Nom des cinémas

```
SELECT NomCinema
FROM cinema
```

2. Nom des cinémas sans doublon

```
SELECT DISTINCT NomCinema
FROM cinema
```

3. Nom des cinémas du 5 arrondissement ($CP = 75005$)

```
SELECT NomCinema
FROM cinema
WHERE CodePostalCinema = 75005
```

4. Nom des cinémas qui sont situés à Besançon

```
SELECT NomCinema FROM cinema
WHERE VilleCinema='Besançon'
```

5. Nom des cinémas situés à BESANCON.

```
SELECT NomCinema FROM cinema
WHERE VilleCinema='BESANCON'
```

Remarque : on obtient le même résultat que pour la requête précédente. Cette possibilité est apparue ces dernières années, elle permet de trouver des chaînes de caractères proches de celle décrite dans la requête, aux caractères accentués ou spéciaux près.

6. Information sur les cinémas dont le nom contient *'cin'*.

```
  SELECT *
FROM cinema
WHERE NomCinema LIKE '%cin%'
```

7. Nom et ville des cinémas qui nont pas de numéro de téléphone.

```
  SELECT NomCinema, VilleCinema
FROM cinema
WHERE TelephoneCinema IS NULL
```

8. Titre des films dont l'année de sortie est supérieure ou égale à *1928*.

```
  SELECT Titre
FROM film
WHERE Annee >= 1928
```

9. Titre des films dont l'année de sortie est comprise entre *1928* et *1980*.

```
SELECT Titre
FROM film
WHERE Annee between 1928 and 1980


  SELECT Titre
FROM film
WHERE Annee >= 1928 and Annee <= 1980
```

10. Numéro des cinémas projetant le film *N°5*.

```
SELECT NumCinema
  FROM affiche
  WHERE NumCinema = 5
```

11. Numéro des cinémas projetant le film *N°5* **ou** le film *N°4*.

```
SELECT NumCinema
  FROM affiche
  WHERE (NumCinema = 5) OR (NumCinema = 4)
```

12. Nom des cinémas qui ont un numéro de téléphone

```
SELECT NomCinema
FROM cinema
WHERE TelephoneCinema IS NOT NULL
```

1.2 Ecriture de requêtes avec jointure

1. Numéro des cinémas projetant le film *"Le bal"*

```
SELECT NumCinema
FROM affiche, film
WHERE affiche.IDFilm = film.IDFilm
AND Titre = 'Le bal'
```

2. Le nom et l'adresse des cinémas projetant le film *"Le bal"*

```
SELECT NomCinema, AdresseCinema, VilleCinema, CodePostalCinema
FROM cinema,affiche,film
WHERE cinema.NumCinema = affiche.NumCinema
AND affiche.IDFilm = film.IDFilm
AND Titre = 'Le bal'
```

3. Le nom des metteurs en scène des films à l'affiche du cinéma *Le Grand Rex*

```
SELECT MetteurEnScene
    FROM cinema, affiche , film
    WHERE cinema.NumCinema=affiche.NumCinema
    and affiche.IDFilm = film.IDFilm
and NomCinema = 'le grand rex'

SELECT MetteurEnScene
    FROM (film INNER JOIN affiche ON film.IDFilm=affiche.IDFilm)
             INNER JOIN cinema ON cinema.numCinema = affiche.numCinema
    WHERE NomCinema = 'le grand rex'
```

```
SELECT MetteurEnScene
    FROM affiche NATURAL JOIN film NATURAL JOIN cinema
    WHERE NomCinema = 'le grand rex'
```

Remarque : Avec différentes expressions de la jointure.

4. Le nom des acteurs apparaissant dans la distribution des films à l'affiche dans le 5 arrondissement (*CP = 75005*)

```
SELECT DISTINCT NomActeur
FROM acteur,JoueDans, affiche,cinema
WHERE cinema.numCinema = affiche.numCinema
AND affiche.IDFilm = JoueDans.IDFilm
AND JoueDans.IDActeur=acteur.IDActeur
AND CodePostalCinema = 75005
```

```
SELECT NomActeur
    FROM ((acteur INNER JOIN jouedans ON acteur.IDActeur = jouedans.IDActeur)
     INNER JOIN affiche ON jouedans.IDFilm = affiche.IDFilm)
 INNER JOIN cinema ON affiche.NumCinema = cinema.NumCinema
    WHERE CodePostalCinema = 75005
```

5. Le titre des films avec *Humphrey Bogart* à l'affiche à *Paris*.

```
SELECT Distinct Titre FROM film
         inner join jouedans ON jouedans.IDFilm=film.IDFilm
         INNER JOIN acteur ON acteur.IDActeur=jouedans.IDActeur
         INNER JOIN affiche ON affiche.IDFilm=film.IDFilm
         INNER JOIN cinema ON cinema.NumCinema=affiche.NumCinema
   WHERE NomActeur = 'Humphrey Bogart'
   AND cinema.VilleCinema = 'Paris'
```

ou

```
SELECT DISTINCT Titre from film, acteur, jouedans, affiche, cinema
WHERE film.IDFilm=jouedans.IDFilm
AND acteur.IDActeur=jouedans.IDFilm
AND  affiche.IDFilm=jouedans.IDFilm
AND cinema.NumCinema=affiche.NumCinema
AND cinema.VilleCinema='Paris'
AND acteur.NomActeur='Humphrey Bogart'
```

6. Le titre des *comédies musicales* à l'affiche à *Paris*, et le nom du cinéma où ils sont à l'affiche.

```
SELECT DISTINCT Titre, NomCinema
from film, affiche, cinema
WHERE cinema.NumCinema=affiche.NumCinema
AND film.IDFilm=affiche.IDFilm
AND cinema.VilleCinema='Paris'
```

ou

```
SELECT Titre, NomCinema
    FROM (film INNER JOIN affiche ON film.IDFilm=affiche.IDFilm)
        INNER JOIN cinema ON affiche.NumCinema=cinema.NumCinema
    WHERE Genre= 'Comédie musicale'
     AND VilleCinema = 'Paris'
```

7. Le noms des cinémas ayant le même code postal que le cinéma de nom *Le Champo*
   Première sous-requête

```
SELECT cinema.NomCinema
FROM cinema
WHERE CodePostalCinema IN
    ( SELECT CodePostalCinema
      FROM cinema
      WHERE NomCinema = 'le champo')
```

   ou solution utilisant l'autojointure

```
SELECT C1.NomCinema
  FROM cinema AS C1, cinema AS C2
  WHERE C1.CodePostalCinema= C2.CodePostalCinema
  AND C2.NomCinema='le champo'
```

1.3 Ecriture de requêtes de comptage

1. Le nombre de Cinémas à Paris

```
SELECT COUNT(NumCinema) AS NbreCineParis
FROM cinema
WHERE VilleCinema ='Paris'

SELECT COUNT(DISTINCT NumCinema) AS NbreCineParis
FROM cinema
WHERE VilleCinema ='Paris'

SELECT COUNT(*) AS NbreCineParis
FROM cinema
WHERE VilleCinema ='Paris'
```

   Remarque : Ces différentes utilisation de la fonction COUNT donnent ici le même résultat, car chaque t-uplet de cinema représente un cinéma. Les numéros de cinéma sont tous différents, donc count(NumCinema) et COUNT(DISTINCT NumCinema) donnent le même résultat et COUNT(*) qui donne le nombre de lignes ou t-uplets donne aussi la même valeur.

2. Pour chaque cinéma, donnez le nombre de films à l'affiche
   Première requête avec un regroupement.

```
SELECT cinema.NumCinema, COUNT(IDFilm) AS NbreFilm
FROM affiche INNER JOIN cinema ON affiche.NumCinema=cinema.NumCinema
GROUP BY cinema.NumCinema
```

3. Le nom des cinémas ainsi que le nombre de films à l'affiche dans ceux-ci

```
SELECT NomCinema, COUNT(IDFilm) AS NbreFilm
FROM cinema, affiche
WHERE cinema.numCinema = affiche.numCinema
GROUP BY cinema.NumCinema, NomCinema

SELECT NomCinema, COUNT(IDFilm) AS NbreFilm
FROM cinema, affiche
WHERE cinema.numCinema = affiche.numCinema
GROUP BY cinema.NumCinema
```

   Remarque : Normalement le regroupement devrait être fait sur NumCinema et NomCinema, mais les dernières versions des SGBD supportent le fait de ne pas mettre NomCinema (qui est unique pour un cinéma).

4. Le numéro des cinémas ayant au moins deux films à l'affiche

```
SELECT cinema.NumCinema
FROM cinema INNER JOIN affiche ON cinema.numCinema = affiche.numCinema
GROUP BY cinema.NumCinema
HAVING COUNT(IDFilm) >= 2
```

5. Le nom des cinémas ayant au moins deux films à l'affiche

```
SELECT NomCinema
FROM cinema, affiche
WHERE cinema.numCinema = affiche.numCinema
GROUP BY cinema.NumCinema, NomCinema
HAVING COUNT(IDFilm) >= 2
```

6. Le numéro des cinémas qui ont autant de films à l'affiche que le cinéma Numéro 1

```
SELECT cinema.NumCinema
FROM cinema INNER JOIN affiche ON cinema.numCinema = affiche.numCinema
GROUP BY cinema.NumCinema
HAVING COUNT(IDFilm) = (SELECT COUNT(IDFilm)
FROM affiche
WHERE NumCinema = 1)
```

7. Pour chaque cinéma, le nombre de films qu'il a à l'affiche. Si le cinéma n'a aucun film à l'affiche, la valeur 0 devra apparaître en face du numéro de cinéma.
   Première jointure externe.

```
SELECT cinema.NumCinema, COUNT(IDFilm) AS NbreFilm
FROM cinema LEFT OUTER JOIN affiche ON cinema.numCinema = affiche.numCinema
GROUP BY cinema.NumCinema
```

8. Pour chaque acteur, le nombre de films dans lesquels il joue. Si l'acteur ne joue dans aucun film, la valeur 0 devra apparaître en face du nom de l'acteur.

```
SELECT acteur.NomActeur, COUNT(IDFilm) AS NbreFilm
FROM acteur LEFT OUTER JOIN jouedans ON acteur.IDActeur = jouedans.IDActeur
GROUP BY acteur.IDActeur
```

9. Pour chaque cinema, le nombre de films à laffiche qui nont pas de genre

```
SELECT NumCinema, COUNT(affiche.IDFilm)
FROM affiche INNER JOIN film ON affiche.IDFilm=film.IDFilm
WHERE Genre IS NOT NULL
GROUP BY NumCinema
```

10. Pour chaque ville, donner le nombre de cinémas dont le numéro de téléphone est défini

```
SELECT VilleCinema, COUNT(TelephoneCinema)
FROM cinema
GROUP BY VilleCinema

SELECT VilleCinema, COUNT(*)
FROM cinema
WHERE TelephoneCinema IS NOT NULL
GROUP BY VilleCinema
```

1.4 Ecriture de requêtes pour aller plus loin ...

Les requêtes proposées ici supposent l'écriture de sous-requêtes.

1.4.1 Définition des sous-requêtes

Les sous-requêtes permettent l'expression d'un prédicat à partir du résultat d'un SELECT. Leur syntaxe est la suivante :

```
SELECT columns
FROM TABLE1
WHERE col Op  (SELECT col
    FROM TABLE2
    WHERE condition)
```

Comment, par exemple, exprimer que l'on veut les titres des films qui ont été réalisés par le même réalisateur que le film "Le bal" ? Nous avons deux types de solutions. La première utilise une auto-jonction de la table FILM :

```
SELECT F1.Titre
FROM FILM F1 INNER JOIN FILM F2 ON F1.MetteurEnScene = F2.MetteurEnScene
WHERE F2.Titre = 'Le bal'.
```

La deuxième solution, qui est exprimée avec une sous-requête, permet de sélectionner les titres de films, appartenant à l'ensemble des titres de films qui ont le même réalisateur que le film *"les voleurs"*. En d'autres termes, nous définissons une requête qui donne le nom de ce réalisateur, et nous l'utilisons dans notre requête principale, ce qui nous donne la requête suivante :

```
SELECT Title
FROM FILM
WHERE MetteurEnScene = (SELECT MetteurEnScene
       FROM FILM
       WHERE Title = 'Le bal')
```

Cette deuxième solution s'exprime beaucoup plus naturellement.

1.5 Definition d'une sous-requête indépendante

Les sous-requêtes permettent d'exprimer un prédicat en utilisant le résultat d'un SELECT, comme le montre la figure **??**. La sous-requête indépendante est évaluée en premier indépendamment de la

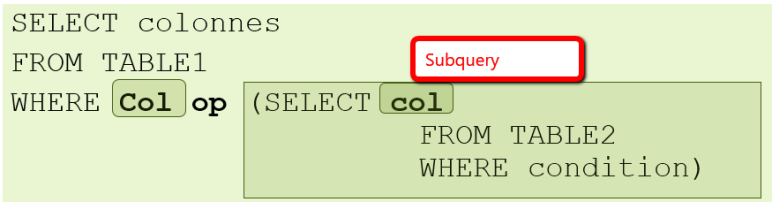


FIGURE 1 – Sous-requête

requête principale, qui est évaluée ensuite. Reprenons la requête présentée ci-dessus. La requête est évaluée :

```
SELECT MetteurEnScene
       FROM FILM
       WHERE Titre = 'Le bal'
```

et donne : "Wilhelm Thiele". Dans un deuxième temps la requête principale ets évaluée et donne :

```
SELECT Titre
FROM FILM
WHERE MetteurEnScene = "Wilhelm Thiele"
```

1.5.1 Expression d'une sous-requête

L'opérateur "op" de la figure 1 peut prendre différentes valeurs.
Si la sous-requête ne renvoie qu'une seule valeur, il s'agit d'un opérateur de comparaison classique : = <> < > <= et >=.

En revanche, si la sous-requête peut renvoyer plusieurs valeurs, il est nécessaire d'utiliser des opérateurs d'ensemble tels que : IN, ANY, ALL.

- **IN** est un opérateur d'ensemble classique : il permet de vérifier que l'attribut spécifié dans la requête principale lié à la sous-requête, a une valeur qui appartient à l'ensemble des valeurs retournées par la sous-requête. Il peut être composé avec l'opérateur NOT : la valeur de l'attribut ne doit pas appartenir à l'ensemble des valeurs retournées par la sous-requête.
- Opérateurs associés avec des opérateurs de comparaison :
  - ANY : La comparaison retourne Vrai : si elle est vraie pour au moins un élément de l'ensemble. La comparaison renvoie Faux : si l'ensemble est vide.
  - ALL : La comparaison renvoie Vrai : si elle est vraie pour tous les éléments de l'ensemble. Si l'ensemble est vide, la comparaison renvoie Vrai.

1.5.2 Exemples de sous-requêtes

Acteurs qui ont déjà joué dans un film avec l'acteur 12.

```
SELECT NomActeur
      FROM JoueDans
      WHERE IDFilm IN (SELECT IDFilm
                           WHERE IDActor = 12)
```

Acteurs qui n'ont jamais joué dans un film avec l'acteur 12.

```
SELECT NomActeur
      FROM JoueDans
      WHERE IDFilm NOT IN (SELECT IDFilm
                           WHERE IDActor = 12)
```

ou

```
SELECT NomActeur
      FROM JoueDans
      WHERE IDFilm <> ALL (SELECT IDFilm
                           WHERE IDActor = 12)
```

1.5.3 Ecriture de sous-requêtes

1. Le nom des cinémas de Paris ayant à l'affiche une *comédie* et une *comédie musicale*
   Il s'agit d'une requête d'intersection.

   ```
   SELECT NomCinema
   FROM film AS F1, film AS F2,affiche AS A1, affiche AS A2,cinema
   WHERE cinema.numCinema = A1.numCinema
   AND A1.IDFilm = F1.IDFilm
   AND F1.Genre = 'comédie'
   AND cinema.numCinema = A2.numCinema
   AND A2.IDFilm = F2.IDFilm
   AND F2.Genre = 'comédie musicale'
     AND VilleCinema ='Paris'

     SELECT NomCinema
   ```

```
FROM film, affiche,cinema
WHERE cinema.numCinema = affiche.numCinema
AND affiche.IDFilm = film.IDFilm
AND film.Genre = 'comédie'
AND cinema.numCinema IN
  (SELECT cinema.NumCinema
   FROM film, affiche,cinema
   WHERE cinema.numCinema = affiche.numCinema
   AND affiche.IDFilm = film.IDFilm
   AND film.Genre = 'comédie musicale')
  AND VilleCinema ='Paris'
```

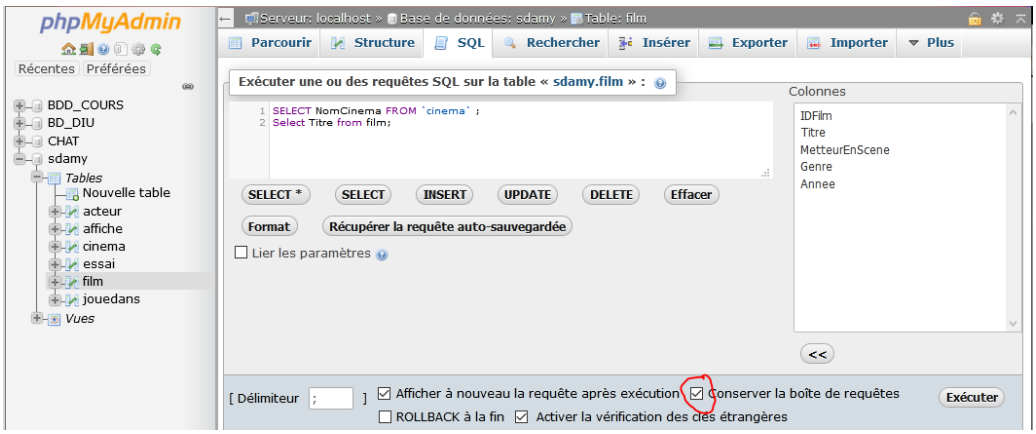
2. Le nom des cinémas n'ayant pas de film de genre *comédie musicale* à l'affiche
   Il s'agit d'une requête de différence

```
SELECT Distinct NomCinema
FROM cinema
WHERE VilleCinema = 'Paris'
AND cinema.NumCinema NOT IN
     (SELECT NumCinema
      FROM affiche INNER JOIN film ON affiche.IDFilm = film.IDFilm
      WHERE Genre = 'comédie musicale')
```

2 Environnements pour écrire les requêtes

Il est possible d'utiliser des outils tels que : phpStorm, Atom, IntelliJ ou Visual Code Studio pour être guidé lorsque l'on rédige des requêtes.

Ces outils peuvent être paramétrés pour prendre en compte le SQL.
Enfin pour conserver les requêtes tapées sous phpmyadmin, il est possible sous l'onglet SQL de cocher l'option "Conserver la boîte de requêtes". On peut ainsi enregistrer plusieurs requêtes qui apparaitront automatiquement à chaque session lorsque l'on se positionnera sur la table et l'onglet SQL. De plus il



est possible de mettre des commentaires dans un script SQL sous la forme :

```
/*      ...          */
```