SQL : Regroupement et requête imbriquée

Thibaud Lemanissier

PC Lycée Carnot

2026-2027

Présentation de la base de donnée

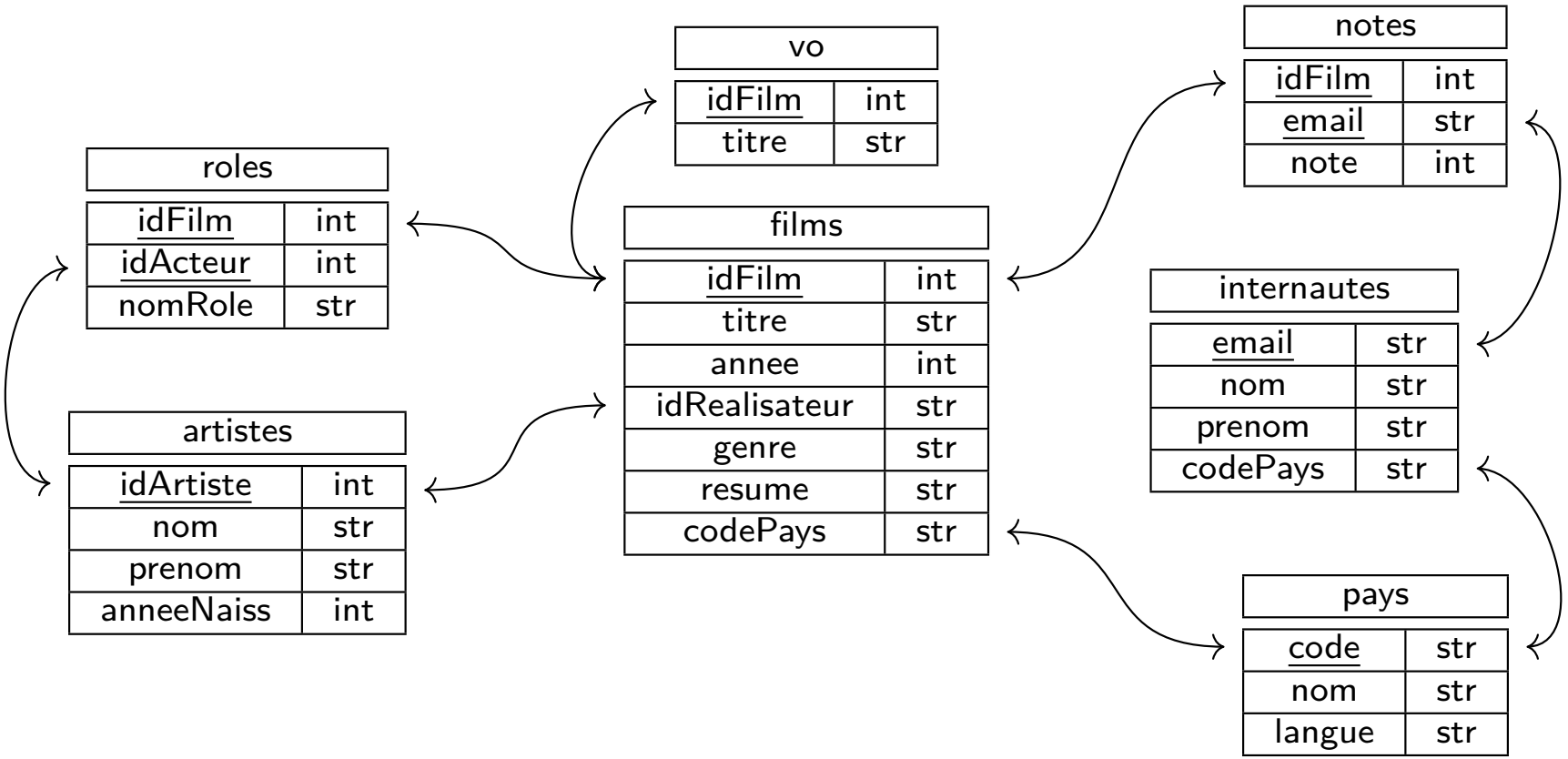


Figure – Une base de donnée contenant quelques films

Modèle Entités et Associations

Entité

Une **entité** est une famille d'objets qui possèdent des caractéristiques communes. Ces objets possèdes des propriétés qui nous permettent de les décrire que l'on appelle **attribut**.

Modèle Entités et Associations

Entité

Une **entité** est une famille d'objets qui possèdent des caractéristiques communes. Ces objets possèdes des propriétés qui nous permettent de les décrire que l'on appelle **attribut**.

Par exemple, les films forment une entité dont on peut sélectionner quelques attribues : titre, année de sortie, genre,...

Modèle Entités et Associations

Entité

Une **entité** est une famille d'objets qui possèdent des caractéristiques communes. Ces objets possèdes des propriétés qui nous permettent de les décrire que l'on appelle **attribut**.

Par exemple, les films forment une entité dont on peut sélectionner quelques attribues : titre, année de sortie, genre,...

Association

Une **association** est une relation entre plusieurs entités.

Modèle Entités et Associations

Entité

Une **entité** est une famille d'objets qui possèdent des caractéristiques communes. Ces objets possèdes des propriétés qui nous permettent de les décrire que l'on appelle **attribut**.

Par exemple, les films forment une entité dont on peut sélectionner quelques attribues : titre, année de sortie, genre,...

Association

Une **association** est une relation entre plusieurs entités.

Par exemple, un acteur a joué dans un film est une association entre l'entité acteur et l'entité film.

Modèle Entités et Associations

Entité

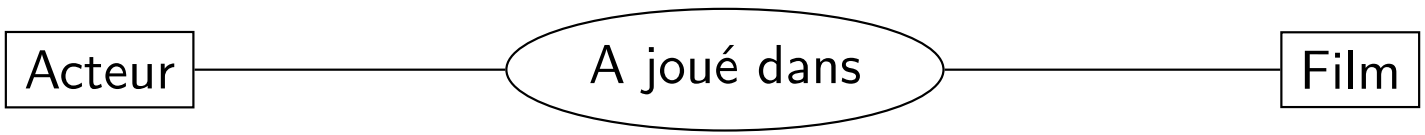
Une **entité** est une famille d'objets qui possèdent des caractéristiques communes. Ces objets possèdes des propriétés qui nous permettent de les décrire que l'on appelle **attribut**.

Par exemple, les films forment une entité dont on peut sélectionner quelques attribues : titre, année de sortie, genre,...

Association

Une **association** est une relation entre plusieurs entités.

Par exemple, un acteur a joué dans un film est une association entre l'entité acteur et l'entité film.
On peut le représenter de la manière suivante :



Cardinalité

Cardianlité

Pour un lien entre entité et association, on peut spécifier la **cardinalité** de cette liaison, c'est-à-dire (dans ce contexte) le nombre minimal et maximal que l'entité peut apparaître dans ce lien. On l'écrit sous la forme d'un couple (a, b) où a est le minimum et b est le maximum. S'il n'y a pas de maximum, on le note $*$.

Cardinalité

Cardianlité

Pour un lien entre entité et association, on peut spécifier la **cardinalité** de cette liaison, c'est-à-dire (dans ce contexte) le nombre minimal et maximal que l'entité peut apparaître dans ce lien. On l'écrit sous la forme d'un couple (a, b) où a est le minimum et b est le maximum. S'il n'y a pas de maximum, on le note $*$.

Par exemple pour l'association « un acteur a joué dans un film », un acteur peut avoir joué dans un nombre arbitraire de film (mais au moins 1) et un film peut avoir un nombre arbitraire d'acteur (mais dans le cas qui nous intéresse, il y a au moins 1 acteur par film). On a donc :

Cardinalité

Cardianlité

Pour un lien entre entité et association, on peut spécifier la **cardinalité** de cette liaison, c'est-à-dire (dans ce contexte) le nombre minimal et maximal que l'entité peut apparaître dans ce lien. On l'écrit sous la forme d'un couple (a, b) où a est le minimum et b est le maximum. S'il n'y a pas de maximum, on le note $*$.

Par exemple pour l'association « un acteur a joué dans un film », un acteur peut avoir joué dans un nombre arbitraire de film (mais au moins 1) et un film peut avoir un nombre arbitraire d'acteur (mais dans le cas qui nous intéresse, il y a au moins 1 acteur par film). On a donc :

Cardinalité

Pour ce qui est de l'association, on peut spécifier la valeur maximal des deux entités concernées. On a trois cas notable :

Cardinalité

Pour ce qui est de l'association, on peut spécifier la valeur maximal des deux entités concernées. On a trois cas notable :

- $1-1$ s'il y a correspondance univoque entre deux entités.
  Par exemple : un film a un unique titre en vo ;

Cardinalité

Pour ce qui est de l'association, on peut spécifier la valeur maximal des deux entités concernées. On a trois cas notable :

- $1-1$ s'il y a correspondance univoque entre deux entités.
  Par exemple : un film a un unique titre en vo ;
- $1-*$ si pour l'une (et une seule) des entités il peut y avoir plusieurs occurrences dans cette association.
  Par exemple : un film peut avoir plusieurs notes mais ces notes ne correspondent qu'à un seul film ;

Cardinalité

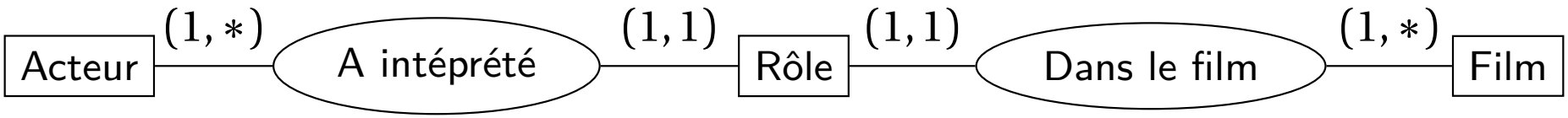
Pour ce qui est de l'association, on peut spécifier la valeur maximal des deux entités concernées. On a trois cas notable :

- $1-1$ s'il y a correspondance univoque entre deux entités.
  Par exemple : un film a un unique titre en vo ;
- $1-*$ si pour l'une (et une seule) des entités il peut y avoir plusieurs occurrences dans cette association.
  Par exemple : un film peut avoir plusieurs notes mais ces notes ne correspondent qu'à un seul film ;
- $*-*$ s'il peut y avoir plusieurs occurrences pour les deux entités de l'association.
  Par exemple : dans un film il y a plusieurs acteurs et un acteur peut avoir joué dans plusieurs films.

L'association $* - *$

Une association $* - *$ peut être découpé en deux associations $1 - *$ en ajoutant une entité. C'est par exemple le cas de l'association :

Dans laquelle on peut ajouter l'entité rôle pour avoir :



L'association $1-*$ en SQL

En SQL, le rôle des entités est joué par les tables et les associations $1-1$ et $1-*$ peuvent se traduire à l'aide des clés primaires et étrangaire :

L'association $1-*$ en SQL

En SQL, le rôle des entités est joué par les tables et les associations $1-1$ et $1-*$ peuvent se traduire à l'aide des clés primaires et étrangaire :

- lorsque l'association ce fait entre deux clés primaires de table, on a alors une association $1-1$.
  Par exemple : les tables vo et films ;

L'association $1-*$ en SQL

En SQL, le rôle des entités est joué par les tables et les associations $1-1$ et $1-*$ peuvent se traduire à l'aide des clés primaires et étrangaire :

- lorsque l'association ce fait entre deux clés primaires de table, on a alors une association $1-1$.
  Par exemple : les tables vo et films ;
- lorsque l'association ce fait entre un clés primaire et un clé étrangaire, on a alors une association $1-*$.
  Par exemple : les tables notes et films ;

Fonction d'agrégation

On a vu que certaine fonctions peuvent regrouper les enregistrements pour en tirer une information globale. Les exemples que nous avons vu sont :

- `MIN` pour obtenir le minimum sur les enregistrements ;
- `MAX` pour obtenir le maximum sur les enregistrements ;
- `COUNT` pour obtenir le nombre d'enregistrements ;
- `SUM` pour obtenir la somme sur les enregistrements ;
- `AVG` pour obtenir la moyenne sur les enregistrements.

Fonction d'agrégation

Pour obtenir le nombre de films sortis avant 2000, on peut écrire :

Fonction d'agrégation

Pour obtenir le nombre de films sortis avant 2000, on peut écrire :

```sql
SELECT COUNT(*) AS nb_films
FROM films
WHERE annee >= 2000 ;
```

Fonction d'agrégation

Pour obtenir le nombre de films sortis avant 2000, on peut écrire :

```sql
SELECT COUNT(*) AS nb_films
FROM films
WHERE annee >= 2000 ;
```

et on obtient :

nb_films
82

Regrouper des enregistrements

Pour considérer plusieurs groupes d'enregistrement possédant des propriétés communes, on peut utiliser les mots clés `GROUP BY`. Cela permet d'utiliser les fonctions d'agrégation sur plusieurs groupes d'enregistrements.

Regrouper des enregistrements

Pour considérer plusieurs groupes d'enregistrement possédant des propriétés communes, on peut utiliser les mots clés `GROUP BY`. Cela permet d'utiliser les fonctions d'agrégation sur plusieurs groupes d'enregistrements.
Pour afficher les identifiants de chaque film avec le nombre de rôle listé dans la base, on peut écrire :

Regrouper des enregistrements

Pour considérer plusieurs groupes d'enregistrement possédant des propriétés communes, on peut utiliser les mots clés `GROUP BY`. Cela permet d'utiliser les fonctions d'agrégation sur plusieurs groupes d'enregistrements.
Pour afficher les identifiants de chaque film avec le nombre de rôle listé dans la base, on peut écrire :

```
1 SELECT idFilm, COUNT(idActeur) AS nb_Acteur
2 FROM roles
3 GROUP BY idFilm ;
```

idFilm	nb_Acteur
11	8
24	11
28	7
33	7
38	4
59	5
62	5
77	6

Regrouper des enregistrements

Si on les veut dans l'ordre croissant de nombre d'acteur en utilisant (ou non un alias), on écrit :

Regrouper des enregistrements

Si on les veut dans l'ordre croissant de nombre d'acteur en utilisant (ou non un alias), on écrit :

```sql
SELECT idFilm, COUNT(idActeur) AS nb_Acteur
FROM roles
GROUP BY idFilm
ORDER BY nb_Acteur ;
```

idFilm	nb_Acteur
1398	1
133919	1
181812	1
153	3
269	3
426	3
571	3
649	3

Regrouper des enregistrements

Pour faire apparaitre le titre du film plutôt que leurs identifiants, on peut faire une jointure :

Regrouper des enregistrements

Pour faire apparaitre le titre du film plutôt que leurs identifiants, on peut faire une jointure :

```sql
SELECT f.titre, COUNT(f.idFilm) AS nb_Acteur
FROM roles AS r JOIN films AS f
ON r.idFilm = f.idFilm
GROUP BY r.idFilm
ORDER BY nb_Acteur ;
```

Regrouper des enregistrements

Pour faire apparaitre le titre du film plutôt que leurs identifiants, on peut faire une jointure :

```sql
SELECT f.titre, COUNT(f.idFilm) AS nb_Acteur
FROM roles AS r JOIN films AS f
ON r.idFilm = f.idFilm
GROUP BY r.idFilm
ORDER BY nb_Acteur ;
```

titre	nb_Acteur
Stalker	1
Scènes de la vie conjugale	1
Star Wars : L'Ascension de Skywalker	1
Lost in Translation	3
À bout de souffle	3
Sueurs froides	3
Les Oiseaux	3
Belle de jour	3

Condition sur des enregistrements regroupés

Pour ajouter une condition sur des enregistrements regroupés on n'utilise pas `WHERE`, on utilise `HAVING`.

Condition sur des enregistrements regroupés

Pour ajouter une condition sur des enregistrements regroupés on n'utilise pas `WHERE`, on utilise `HAVING`.
Pour obtenir la note moyenne (sur 10) des films qui ont une moyenne plus grande que 6, on peut écrire :

Condition sur des enregistrements regroupés

Pour ajouter une condition sur des enregistrements regroupés on n'utilise pas `WHERE`, on utilise `HAVING`.

Pour obtenir la note moyenne (sur 10) des films qui ont une moyenne plus grande que 6, on peut écrire :

```sql
SELECT idFilm, AVG(note) AS moyenne
FROM notes
GROUP BY idFilm
HAVING AVG(note) >= 6
ORDER BY moyenne ;
```

idFilm	moyenne
769	6.04348
861	6.05556
423	6.05882
424	6.09375
266	6.14286
273481	6.8

Condition sur des enregistrements regroupés

Pour ajouter une condition sur des enregistrements regroupés on n'utilise pas `WHERE`, on utilise `HAVING`.

Pour obtenir la note moyenne (sur 10) des films qui ont une moyenne plus grande que 6, on peut écrire :

```sql
SELECT idFilm, AVG(note) AS moyenne
FROM notes
GROUP BY idFilm
HAVING AVG(note) >= 6
ORDER BY moyenne ;
```

idFilm	moyenne
769	6.04348
861	6.05556
423	6.05882
424	6.09375
266	6.14286
273481	6.8

On peut aussi utiliser l'alias dans le `HAVING` et pour ordonner les données :

Condition sur des enregistrements regroupés

Pour ajouter une condition sur des enregistrements regroupés on n'utilise pas `WHERE`, on utilise `HAVING`.

Pour obtenir la note moyenne (sur 10) des films qui ont une moyenne plus grande que 6, on peut écrire :

```sql
SELECT idFilm, AVG(note) AS moyenne
FROM notes
GROUP BY idFilm
HAVING AVG(note) >= 6
ORDER BY moyenne ;
```

idFilm	moyenne
769	6.04348
861	6.05556
423	6.05882
424	6.09375
266	6.14286
273481	6.8

On peut aussi utiliser l'alias dans le `HAVING` et pour ordonner les données :

```sql
SELECT idFilm, AVG(note) AS moyenne
FROM notes
GROUP BY idFilm
HAVING moyenne >= 6
ORDER BY moyenne ;
```

idFilm	moyenne
769	6.04348
861	6.05556
423	6.05882
424	6.09375
266	6.14286
273481	6.8

Le tout avec une jointure

Le tout avec une jointure

Pour afficher le titre du film plutôt que ces identifiants, on peut faire une jointure :

```sql
SELECT f.titre, AVG(n.note) AS moyenne
FROM notes AS n JOIN films AS f
ON f.idFilm = n.idFilm
GROUP BY f.idFilm
HAVING moyenne >= 6
ORDER BY moyenne ;
```

Le tout avec une jointure

Pour afficher le titre du film plutôt que ces identifiants, on peut faire une jointure :

```sql
SELECT f.titre, AVG(n.note) AS moyenne
FROM notes AS n JOIN films AS f
ON f.idFilm = n.idFilm
GROUP BY f.idFilm
HAVING moyenne >= 6
ORDER BY moyenne ;
```

titre	moyenne
Les Affranchis	6.04348
Total Recall	6.05556
Le Pianiste	6.05882
La liste de Schindler	6.09375
Le Mépris	6.14286
Sicario	6.8

Comparaison WHERE et HAVING

Les requêtes ne sont pas exectuée dans l'ordre d'écriture. Dans l'ordre :

Comparaison WHERE et HAVING

Les requêtes ne sont pas exectuée dans l'ordre d'écriture. Dans l'ordre :

- sélection des enregistrements avec `WHERE` ;

Comparaison WHERE et HAVING

Les requêtes ne sont pas exectuée dans l'ordre d'écriture. Dans l'ordre :

- sélection des enregistrements avec `WHERE` ;
- regroupement des résultats avec `GROUP BY` ;

Comparaison WHERE et HAVING

Les requêtes ne sont pas exectuée dans l'ordre d'écriture. Dans l'ordre :

- sélection des enregistrements avec `WHERE` ;
- regroupement des résultats avec `GROUP BY` ;
- calcul sur les données regroupé avec `MIN`, `MAX`, `SUM`,... ;

Comparaison WHERE et HAVING

Les requêtes ne sont pas exectuée dans l'ordre d'écriture. Dans l'ordre :

- sélection des enregistrements avec `WHERE` ;
- regroupement des résultats avec `GROUP BY` ;
- calcul sur les données regroupé avec `MIN`, `MAX`, `SUM`,... ;
- sélection des enregistrements avec `HAVING`.

Comparaison WHERE et HAVING

Pour obtenir la note moyenne (sur 10) des films qui ont une moyenne plus grande que 8.7 lorsque l'on se limite aux notes supérieures à 6, on peut écrire :

Comparaison WHERE et HAVING

Pour obtenir la note moyenne (sur 10) des films qui ont une moyenne plus grande que 8.7 lorsque l'on se limite aux notes supérieures à 6, on peut écrire :

```sql
SELECT f.titre, AVG(n.note) AS moyenne
FROM notes AS n JOIN films AS f
ON f.idFilm = n.idFilm
WHERE n.note >= 6
GROUP BY f.idFilm
HAVING moyenne >= 8.5
ORDER BY moyenne ;
```

titre	moyenne
Harry, un ami...	8.5
Apocalypse Now	8.58
Manhattan	8.58
Minuit à Paris	8.6
Le Mépris	8.63
OSS 117 : Le ...	8.64
Manchester by ...	8.65
Rebecca	8.67
Le sacrifice	8.67
Star Wars : L'...	8.67
Reservoir Dogs	8.69
Rencontres du ...	8.73
Le grand bain	8.77
E.T. l'extra-...	8.8

Requête imbriqué

Il est parfait nécessaire de faire des calculs intermédiaires pour obtenir les résultats d'une requête. Pour cela, on fait des requêtes imbriquées.

Requête imbriqué

Il est parfait nécessaire de faire des calculs intermédiaires pour obtenir les résultats d'une requête. Pour cela, on fait des requêtes imbriquées.
Par exemple, si l'on souhaite obtenir les films qui sont sortis au plus 20 ans après le film le plus ancien, on doit commencer par déterminer la date du film le plus ancien, on peut donc écrire :

Requête imbriqué

Il est parfait nécessaire de faire des calculs intermédiaires pour obtenir les résultats d'une requête. Pour cela, on fait des requêtes imbriquées.
Par exemple, si l'on souhaite obtenir les films qui sont sortis au plus 20 ans après le film le plus ancien, on doit commencer par déterminer la date du film le plus ancien, on peut donc écrire :

```
1 SELECT titre, annee
2 FROM films
3 WHERE annee <= 20+(SELECT MIN(annee) FROM films ) ;
```

Requête imbriqué

Il est parfait nécessaire de faire des calculs intermédiaires pour obtenir les résultats d'une requête. Pour cela, on fait des requêtes imbriquées.
Par exemple, si l'on souhaite obtenir les films qui sont sortis au plus 20 ans après le film le plus ancien, on doit commencer par déterminer la date du film le plus ancien, on peut donc écrire :

```sql
SELECT titre, annee
FROM films
WHERE annee <= 20+(SELECT MIN(annee) FROM films ) ;
```

titre	annee
Rebecca	1940
La Grande Illusion	1937
M le maudit	1931
Le dictateur	1940
Les temps modernes	1936
Le Kid	1921
Soupçons	1941
La Charge de la brigade légère	1936

Requête imbriqué

On peut aussi faire des requêtes imbriquées avec un regroupement.

Requête imbriqué

On peut aussi faire des requêtes imbriquées avec un regroupement.
Si on souhaite obtenir la liste des identifiants de film qui ont une note moyenne supérieure à la moyenne de toute les notes données, on peut écrire :

Requête imbriqué

On peut aussi faire des requêtes imbriquées avec un regroupement.
Si on souhaite obtenir la liste des identifiants de film qui ont une note moyenne supérieure à la moyenne de toute les notes données, on peut écrire :

```
1 SELECT idFilm, AVG(note) AS moyenne
2 FROM notes
3 GROUP BY idFilm
4 HAVING moyenne >= (SELECT AVG(note) FROM notes) ;
```

Requête imbriqué

On peut aussi faire des requêtes imbriquées avec un regroupement.
Si on souhaite obtenir la liste des identifiants de film qui ont une note moyenne supérieure à la moyenne de toute les notes données, on peut écrire :

```sql
SELECT idFilm, AVG(note) AS moyenne
FROM notes
GROUP BY idFilm
HAVING moyenne >= (SELECT AVG(note) FROM notes) ;
```

idFilm	moyenne
28	5.154
38	5.097
62	5.121
85	5.886
87	5.513
89	5.302
98	5.903
145	5.257

Requête imbriqué

On peut ajouter une jointure pour faire apparaitre le titre du film et les ranger dans l'ordre croissant :

Requête imbriqué

On peut ajouter une jointure pour faire apparaitre le titre du film et les ranger dans l'ordre croissant :

```sql
SELECT f.titre, AVG(n.note) AS moyenne
FROM notes AS n JOIN films AS f
ON n.idFilm = f.idFilm
GROUP BY n.idFilm
HAVING moyenne >= (SELECT AVG(note) FROM notes)
ORDER BY moyenne ;
```

Requête imbriqué

On peut ajouter une jointure pour faire apparaitre le titre du film et les ranger dans l'ordre croissant :

```sql
SELECT f.titre, AVG(n.note) AS moyenne
FROM notes AS n JOIN films AS f
ON n.idFilm = f.idFilm
GROUP BY n.idFilm
HAVING moyenne >= (SELECT AVG(note) FROM notes)
ORDER BY moyenne ;
```

titre	moyenne
Casablanca	5.0
Star Wars, épisode III - La Revanche des Sith	5.0
La Grande Vadrouille	5.0
Jusqu'à la garde	5.02
Mulholland Drive	5.03
Melancholia	5.03
La Grande Illusion	5.03
Thelma et Louise	5.03

Requête imbriqué

La requête imbriqué peut-être dans la partie sélectionnée.

Requête imbriqué

La requête imbriqué peut-être dans la partie sélectionnée.
Si on souhait avoir la différence entre la note moyenne des films et la moyenne de toute les notes données, on peut écrire :

Requête imbriqué

La requête imbriqué peut-être dans la partie sélectionnée.
Si on souhait avoir la différence entre la note moyenne des films et la moyenne de toute les notes données, on peut écrire :

```
1 SELECT f.titre, AVG(n.note)-(SELECT AVG(note) FROM notes)  AS  →
      difference
2 FROM notes AS n JOIN films AS f
3 ON n.idFilm = f.idFilm
4 GROUP BY n.idFilm
5 ORDER BY difference ;
```

Requête imbriqué

La requête imbriqué peut-être dans la partie sélectionnée.
Si on souhait avoir la différence entre la note moyenne des films et la moyenne de toute les notes données, on peut écrire :

```
1 SELECT f.titre, AVG(n.note)-(SELECT AVG(note) FROM notes)  AS  →
      difference
2 FROM notes AS n JOIN films AS f
3 ON n.idFilm = f.idFilm
4 GROUP BY n.idFilm
5 ORDER BY difference ;
```

titre	difference
La Porte du paradis	-1.626
Impitoyable	-1.231
Fenêtre sur cour	-1.084
Skyfall	-1.081
Matrix	-0.981
Le Corbeau	-0.95
Jackie Brown	-0.921
Les Quatre Cents Coups	-0.864