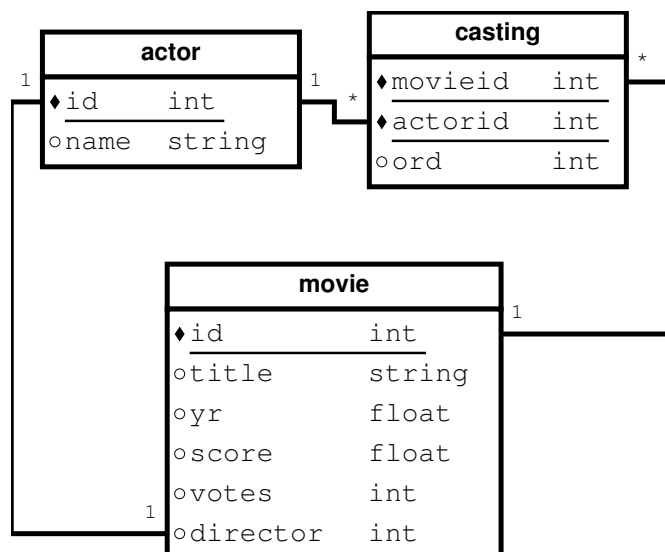


COURS-TP : Sql 2

Récupérer le fichier `movies.sqlite` (qu'il faut copier sur son espace personnel), il sera peut-être aussi nécessaire de récupérer et d'installer le logiciel DB BROWSER FOR SQLITE. Cette base est censée porter sur tous les films de cinéma, leur casting, acteurs et réalisateurs jusqu'à l'année 2000. La table `actor` contient les acteurs mais aussi les réalisateurs.



Quand il y a plusieurs tables (ce qui est le cas ici) il est important de connaître le contenu de toutes les tables et surtout les attributs qui permettent de les relier, ce jeu d'information est appelé *schéma de la base de données*, c'est ce que représente notre graphique (les clés primaires sont soulignées, vous remarquerez que dans la table `casting` elle est constituée de deux attributs).

Ici on a plusieurs tables, en effet l'un des objectifs quand on crée une base de données est d'éviter la redondance de données (car cela prend plus de places, mais c'est aussi sensible aux erreurs de frappe), ainsi si dans une entrée de la table `movie` l'attribut `director` vaut 5, c'est que le réalisateur de ce film est l'entrée de la table `actor` dont l'attribut `id` vaut 5, un réalisateur peut avoir réalisé plusieurs films, mais (dans notre cas) chaque film possède un réalisateur, c'est ce qu'on appelle une association 1 – *. Ainsi l'attribut `director` de la table `movie` fait référence à la clé primaire de la table `director`, on dit que `director` est une *clé étrangère*.

Il existe aussi des associations 1 – 1 (non présent dans cette base).

Si on considère les acteurs, plusieurs acteurs jouent dans un film et un acteur peut jouer dans plusieurs films, cela correspond à une association * – *, dans notre table, cette association a été séparée en deux associations 1 – * grâce à la table `casting`, par exemple dans cette table `casting` on a l'enregistrement (1,6,2) qui dit, par exemple que l'acteur dont l'id est 6 (Harrison Ford) a joué dans le film dont l'id est 1 (Star Wars) et qu'il occupait le second rôle, mais on a aussi (37,6,1) (Harrison Ford occupait le premier rôle dans Indiana Jones) et (1,462,3) (Carrie Fisher occupait le troisième rôle de Star Wars).

I] Opérateurs ensemblistes et jointures simples

Il existe différentes opérations sur les tables, par exemple l'union (`UNION`), l'intersection (`INTERSECT`) et la "différence" (`EXCEPT`), cette dernière donne les éléments qui sont présents dans la première table mais pas dans la seconde. Il y aura quelques exemples plus tard.

Quand on a deux tables, on peut aussi réaliser le produit cartésien des tables, cela correspond à toutes les combinaisons possibles des enregistrements des deux tables, par exemple :

Cependant ce qu'on va majoritairement utiliser, ce sont les jointures, par exemple si on veut obtenir une table qui contient, par exemple, les noms des films et les noms des réalisateurs

```
SELECT movie.title, rea.name
FROM movie
      JOIN actor AS rea ON movie.director=rea.id
```

On notera la syntaxe `FROM ... JOIN ... ON ...` qui permet d'avoir la sélection des éléments du produit cartésien qui vérifient une certaine condition¹. On remarquera que quand on utilise plusieurs tables on précise à quelle table appartient l'attribut considéré (pour éviter des ambiguïtés, par exemples si on avait utilisé des attributs en français on aurait pu avoir le nom du film et le nom de l'acteur ...)

Q1. Afficher le nombre de films de chaque réalisateurs (on mettra en premier les réalisateurs qui ont réalisés le plus de films).

Remarque : Il est aussi possible d'avoir une association 1 – 1 d'une table avec elle-même, par exemple si on a la liste des employés d'une entreprise avec comme attribut `id` (identifiant de l'employé, c'est la clé primaire), `nom` (son nom), ... , `id_sup` (identifiant de son supérieur hiérarchique direct), ainsi l'attribut `id_sup` d'un enregistrement fait référence à l'attribut `id` d'un autre. Dans ce contexte on parle d'*autojointure*.

II] À vous de jouer

A vous maintenant d'écrire les requêtes pour répondre aux questions, il sera nécessaire d'utiliser une ou plusieurs jointures (`FROM ... JOIN ... ON ... JOIN ... ON ...`).

Q2. Un même acteur peut jouer différents rôles (par exemple Harrison Ford est Han Solo dans Star Wars et Indiana Jones dans le film du même nom), où doit-on rajouter cette donnée?

Q3. Pour se mettre en bouche :

- Afficher les (titres des) 10 films les mieux notés de l'année 1995.
- Afficher le nombre de films contenus dans la base.
- Afficher la liste des acteurs/réalisateurs dont les initiales sont T.G.
- Combien d'acteurs/réalisateurs ont un nom commençant par la lettre A ? (la question n'est pas assez précise, est-ce le cas de "Alfred Hitchcock" ? de "Woody Allen" ? des deux ?)
- Combien de films ont obtenu une note supérieure à 8 ?

Q4. On continue avec un peu de jointures :

- Afficher les (titres des) films où on peut voir Harrison Ford.
- Afficher les films où le réalisateur est aussi un acteur.
- Sortir le casting du film "Star Wars".
- Afficher les noms de tous les réalisateurs.
- Afficher les noms des 10 réalisateurs dont la moyenne des scores de leurs films est la plus basse.
- Afficher le nom des acteurs qui n'ont joués dans aucun film présent dans cette base (comme exemple utilisation de `EXCEPT`).

III] A savoir

Vous devez connaître :

- le vocabulaire lié aux bases de données : tables ou relations, attributs ou colonnes, domaine, schéma de tables, enregistrements ou lignes, types de données ;
- la notion de clé primaire ;
- le modèle entité-association et la notion de clé étrangère ;
- la syntaxe de base SQL : `SELECT ... FROM ... WHERE`, le renommage `AS`, les opérateurs `+`, `-`, `*`, `/`, `=`, `<>`, `<`, `<=`, `>`, `>=`, `AND`, `OR` et `NOT`, l'utilisation des mots-clés `DISTINCT`, `LIMIT`, `OFFSET`, `ORDER BY` ;
- les agrégations avec les fonctions `MIN`, `MAX`, `SUM`, `AVG` et `COUNT`, et ce avec ou sans `GROUP BY`, ainsi que le filtrage des agrégats avec `HAVING` ;
- Les opérateurs ensemblistes `UNION`, `INTERSECT` et `EXCEPT`, ainsi que le produit cartésien ;
- Les jointures interne `... JOIN ... ON ...` et les autojointures.

1. il s'avère qu'on pourrait obtenir le même résultat avec un produit cartésien et une condition `SELECT movie.title, rea.name FROM movie, actor AS rea WHERE movie.director=rea.id`