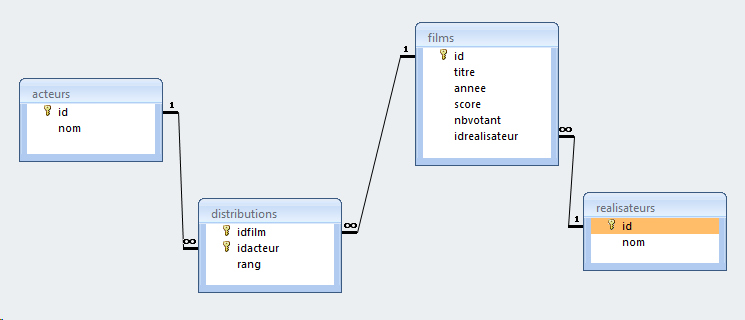
Soit une **base de données de films** et d'informations sur ces films (acteurs, réalisateurs, ...). Cette base est inspirée de ce [site](http://www.imdb.com/). Le schéma relationnel de cette base est décrit ci-dessous :



Il se concrétise dans les relations suivantes dans la base de données du TP (les clés sont soulignées et les clés étrangères sont en gras) :

* films (id, titre, annee, score, nbvotant, **idrealisateur**)
* acteurs (id, nom)
* distributions (**idfilm, idacteur**, rang)
* realisateurs(id, nom)

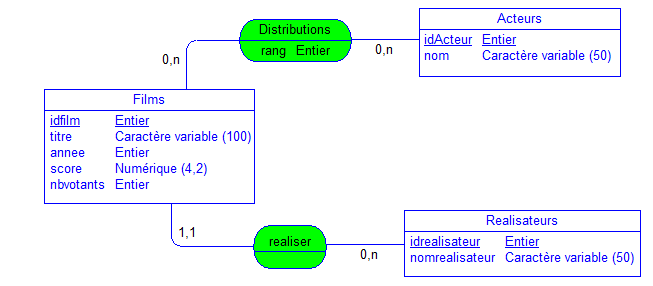
Un **film** est décrit de manière unique par un identifiant (entier, clé de la relation), un titre, une année (première sortie du film), un score (moyenne de tous les votes pour ce film), un nombre de votants et un réalisateur (entier, clé étrangère sur réalisateurs).

Un **acteur** est décrit de manière unique par un identifiant (entier, clé de la relation) et un nom.

Un acteur peut jouer un rôle dans zéro ou plusieurs films et un film a zéro ou plusieurs acteurs dans sa distribution. L'attribut rangindique le statut de l'acteur dans le film (rang=1 indique que l'acteur est la vedette du film). Le couple (idacteur, idfilm) est la clé de la relation **distributions**. idacteur est une clé étrangère sur acteurs et idfilm est une clé étrangère sur films.

Un **réalisateur** est identifié de manière unique par son identifiant (entier, clé de la relation) et un nom.

Pour votre information, le schéma de conception de cette base (schéma entité-association)est le suivant. Un chapitre de cours est consacré à l'étude des schéma entité-association.



**1. Questions**

**1.1** Donner la liste des films (id, titre et annee) triée par titre.

160 tuples

SELECT id, titre, annee

FROM films

order by titre

**1.2** Donner la liste des films avec un score supérieur ou égal à 9 ?

4 tuples

SELECT id, titre, annee, score

FROM films

where score >= 9

**1.3** Quels sont les acteurs principaux (rang=1) des films sortis en 2000 ? On veut l'id et les noms triés par nom.

6 tuples

SELECT a.id, nom

FROM films f

join distributions d on f.id = d.idfilm

join acteurs a on a.id = d.idacteur

where rang = 1

and annee = 2000

order by nom

**1.4** Donner les id des films sortis avant 1930 ou bien dans lesquels joue l'acteur de id 12.

7 tuples (ou 8 pour <=)

SELECT distinct f.id

FROM films f join distributions d on f.id = d.idfilm

where d.idacteur = 12

or annee <= 1930

**1.5** Donner le nom des vedettes (rang=1) du film ayant l'id 11, dans l'ordre alphabétique.

1 tuple

SELECT nom

FROM acteurs a join distributions d on a.id = d.idacteur

where rang = 1

and idfilm = 11

order by nom

**1.6** Donner les noms des réalisateurs de films sortis avant 1970 et ayant '*the*' dans leur titre.

25 tuples (SQLite n'est pas sensible à la casse de '*the*'. PostgreSQL renverrait 11 tuples, en différenciant '*the*' de '*THE*')

SELECT distinct nom

FROM realisateurs r

join films f on r.id = f.idrealisateur

where annee < 1970

and titre like '%the%'

**1.7** Donner les id et titre des films n'ayant aucun acteur dans leur distribution.

6 tuples

select films.id , films.titre

from films

WHERE id NOT IN

( select idfilm

from distributions )

**1.8** Donner les films (id et titre) sortis en 2000 et ayant au moins deux acteurs dans leur distribution.

6 tuples

select films.id , films.titre , count( distributions.idacteur)

from films

JOIN distributions ON films.id = distributions.idfilm

WHERE annee = 2000

GROUP BY films.id , films.titre

HAVING count( distributions.idacteur ) > 2

**1.9** Donner le(s)  film(s)  (id, titre) avec le score le plus haut.

4 tuples

select films.id , films.titre

from films

WHERE score = (select max(score) from films)

1.10 Donner la moyenne des votants par année de sortie de film. 72 tuples

SELECT annee, avg(nbvotant)

FROM films

GROUP BY annee

1.11 Donner le nombre de films par réalisateur (nom du réalisateur), trié par nombre de films décroissant.

103 tuples

SELECT nom, count(f.id) as [Nombre de films réalisés]

FROM films f JOIN realisateurs r on f.idrealisateur=r.id

GROUP BY nom

ORDER BY 2 desc

1.12 Donner les noms des acteurs n’ayant jamais joué, trié par nom d’acteur.30 tuples

SELECT distinct nom

FROM acteurs

where id NOT IN

( SELECT distinct idacteur

FROM distributions)

ORDER BY nom

1.13 Donner les noms des acteurs n’ayant jamais été au rang 1. Le résultat sera trié par nom d’acteur.

779 tuples

SELECT distinct nom

FROM acteurs a JOIN distributions d on a.id = d.idacteur

where a.id NOT IN

( SELECT distinct idacteur

FROM distributions

WHERE rang = 1)

ORDER BY nom

**1.14** Pour chaque film afficher son titre, son année de sortie, le nom du réalisateur, les noms de chacun des acteurs avec leur rang. Le résultat sera trié année de sortie du film décroissant, titre du film croissant et par rang de l’acteur croissant. Les jointures devront être faites avec la clause JOIN. 1135 tuples

SELECT titre, annee, r.nom, a.nom, rang

FROM acteurs a

JOIN distributions d ON a.id = d.idacteur

JOIN films f ON d.idfilm = f.id)

JOIN realisateurs r ON f.idrealisateur = r.id

ORDER BY annee DESC, titre,rang