



# Introduction aux bases de données relationnelles

M2 TNAH – 2018-2019



# Objectifs

Découvrir les principaux concepts des bases de données relationnelles : entité, attribut, association, cardinalité, clé primaire, clé secondaire, table, champ, enregistrement, opérations CRUD, etc.

- Cerner les principaux écueils de la modélisation.
- Construire une première base.
- Écrire des requêtes, exporter et exploiter les résultats.

**SQLite / MySQL**

# Une première base

Philippe RIGAUX, *Pratique de MySQL et PHP*, O'Reilly, 3e édition, 2005, chap. 5 (« Création d'une base MySQL »)

titre	année	nom_realisateur	prenom_realisateur	annee_naissance
Alien	1979	Scott	Ridley	1943
Vertigo	1958	Hitchcock	Alfred	1899
Psychose	1960	Hitchcock	Alfred	1899
etc.				

**Tableur : des tables et des enregistrements**  
**Problèmes ?**

## Insertion

- Unicité : comment empêcher de représenter plusieurs fois le même réalisateur ?
- Intégrité : un réalisateur peut apparaître plusieurs fois et décrit de différentes manières (annee\_naissance).

## Modification

- Maintenir la cohérence des données : si je modifie la date de naissance pour le premier enregistrement "Hitchcock" (*Vertigo*), comment garantir le report de l'information pour le second enregistrement "Hitchcock" (*Psychose*) ? Comment gérer les homonymes ?

## Suppression

- Perte d'information : on ne peut pas supprimer un film sans supprimer le réalisateur.

### films

titre	annee
Alien	1979
Vertigo	1958
Psychose	1960

### realisateurs

nom_realisateur	prenom_realisateur	annee_naissance
Scott	Ridley	1943
Hitchcock	Alfred	1899

Pour éviter les redondances, on peut représenter séparément les films et les réalisateurs.

**Problème** : comment identifier de manière unique un film et un réalisateur ?

## films

id	titre	annee
1	Alien	1979
2	Vertigo	1958
3	Psychose	1960

## realisateurs

id	nom_realisateur	prenom_realisateur	annee_naissance
1	Scott	Ridley	1943
2	Hitchcock	Alfred	1899

## Notions

- Entité
- Attribut
- Clé primaire (identifiant) : attribut qui identifie un enregistrement

## Problème

- Comment représenter le lien entre un film et un réalisateur ?

## films

id	titre	annee	id_realisateur
1	Alien	1979	1
2	Vertigo	1958	2
3	Psychose	1960	2

## realisateurs

id	nom_realisateur	prenom_realisateur	annee_naissance
1	Scott	Ridley	1943
2	Hitchcock	Alfred	1899

## Notions

- Clé étrangère : attribut qui référence le champ d'une autre table



# Problème

Ce référencement par clé étrangère ne fonctionne que dans le cas où nous considérons qu'un film est réalisé par un unique réalisateur.

Un film peut être coréalisé, par exemple :

*Les statues meurent aussi*, court métrage documentaire réalisé par Chris Marker, Alain Resnais et Ghislain Cloquet, sorti en 1953.

[https://fr.wikipedia.org/wiki/Les\\_statues\\_meurent\\_aussi](https://fr.wikipedia.org/wiki/Les_statues_meurent_aussi)



## films

id	titre	annee
1	Alien	1979
2	Vertigo	1958
3	Psychose	1960
4	Les statues...	1953

?

id_film	id_realisateur
1	1
2	2
3	2
4	3
4	4
4	5

## realisateurs

id	nom_realisateur	prenom_realisateur	annee_naissance
1	Scott	Ridley	1943
2	Hitchcock	Alfred	1899
3	Marker	Chris	1921
4	Resnais	Alain	1922
5	Cloquet	Ghislain	1924

**Association** est une liaison qui a une signification précise entre plusieurs entités.

Ici : “un réalisateur ? un film”



# Notions, interro surprise !

- Entité
- Attribut
- Table
- Enregistrement
- Association
- Clé primaire
- Clé étrangère



# Conception des bases de données : modèle Entité-Association

Deux phases :

1. **Réalisation du modèle conceptuel (MCD)**

La modélisation conceptuelle est une étape délicate : définition des données, de leur mode d'évolution dans le temps et des relations entre elles. Formalisme de type Entité-Association.

2. Traduction du MCD en un modèle logique, relationnel (MRD) dans notre cas  
Définition de l'ensemble des objets manipulables par le SGBD-R.



# Modèle conceptuel (1/2)

Le modèle conceptuel décrit les **entités** (*Film, Réalisateur*), leurs **attributs** (*titre, nom, date de naissance*), les **associations** (*réalise*) entre ces entités et leur **cardinalité**.

Une **entité** est une chose concrète ou abstraite qui peut être reconnue distinctement et qui est caractérisée par son unicité.

**NB: type-entité (*Film*) / entités (*Vertigo*)** : “population homogène d’individus” qui possèdent des propriétés communes”.

Par abus de langage, le terme entité est (toujours) utilisé pour désigner le type-entité et ses entités.

Une **association** est une liaison qui a une signification précise entre plusieurs entités : “Un réalisateur *réalise* un film.”

Un **attribut** (ou propriété) est une caractéristique associée à une entité (type-entité) ou à une association.

- Un attribut ne pas être partagé par plusieurs entités ou associations.
- Un attribut est une donnée élémentaire : il n’est ni calculé, ni dérivé (*date de naissance / âge*).
- Un attribut peut caractériser une association lorsqu’il dépend de toutes les entités liées par l’association (*rôle*).



## Modèle conceptuel (2/2)

La **cardinalité** est le **nombre d'occurrences minimal et maximal**, exprimé sous la forme d'un couple (*min*, *max*), d'une association par rapport à chaque occurrence d'une entité donnée.

Un réalisateur *réalise* **un à n** film(s) (1, n)

Un film *est réalisé par* **un à n** réalisateur(s) (1, n)

D'une entité donnée **vers une association** donnée :

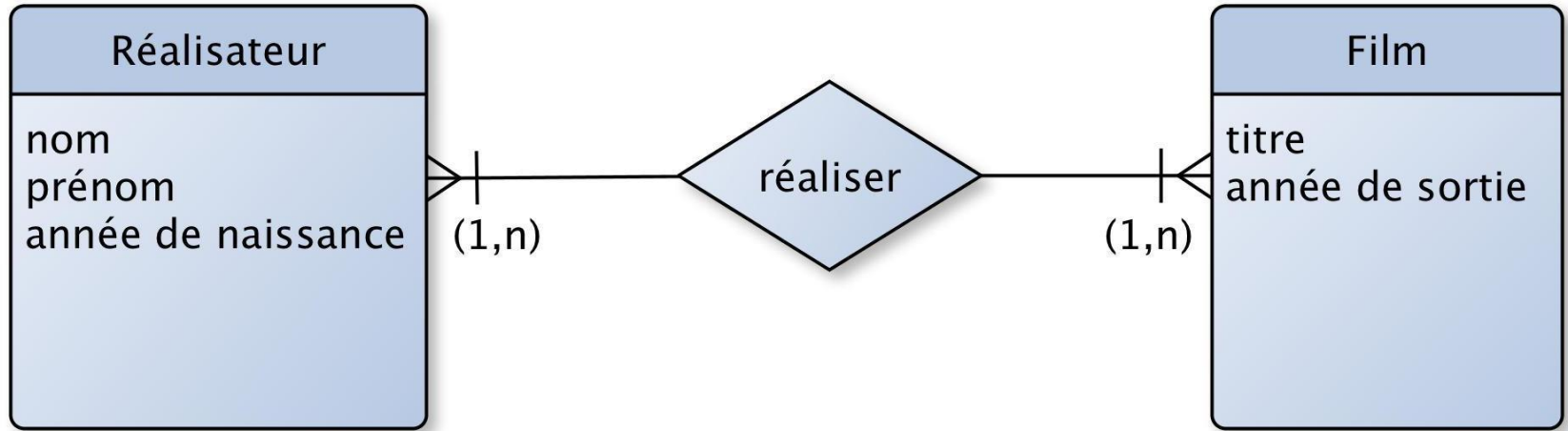
La cardinalité **minimale** peut être **0 ou 1**.

La cardinalité **maximale** peut être **1 ou n**.

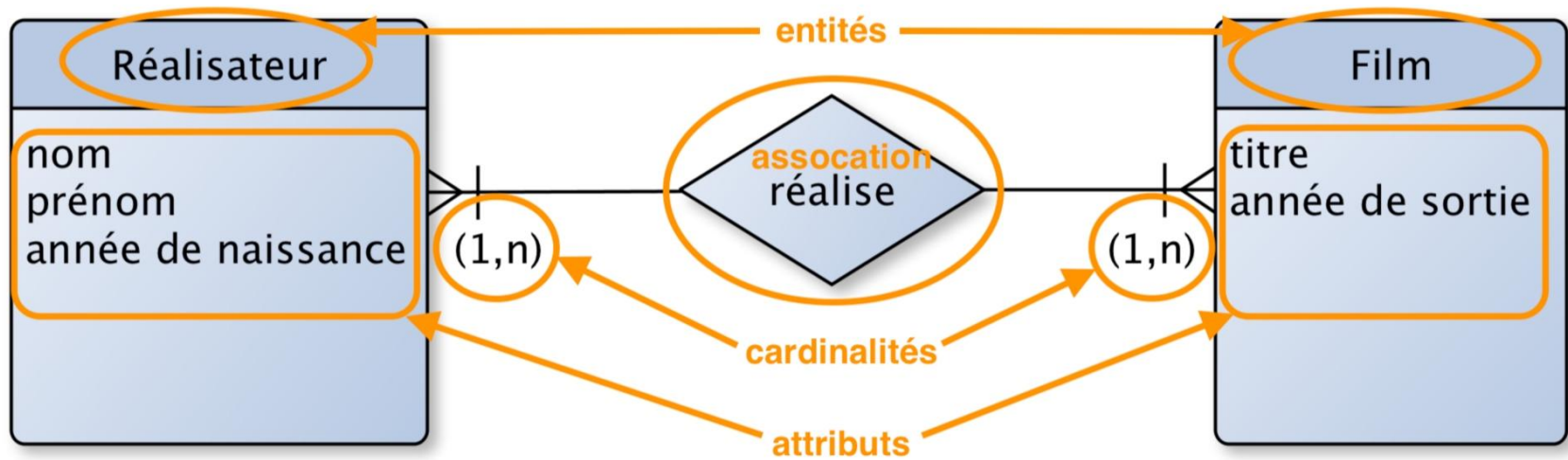
Une cardinalité minimale de 1 doit se justifier par le fait que les individus de l'entité en question ont besoin de l'association pour exister. Dans tous les autres cas, la cardinalité minimale vaut 0.

- Une personne réalise 0 à n films(s).
- Un réalisateur réalise 1 à n films(s).
- Un film est réalisé par 1 à n réalisateur(s).

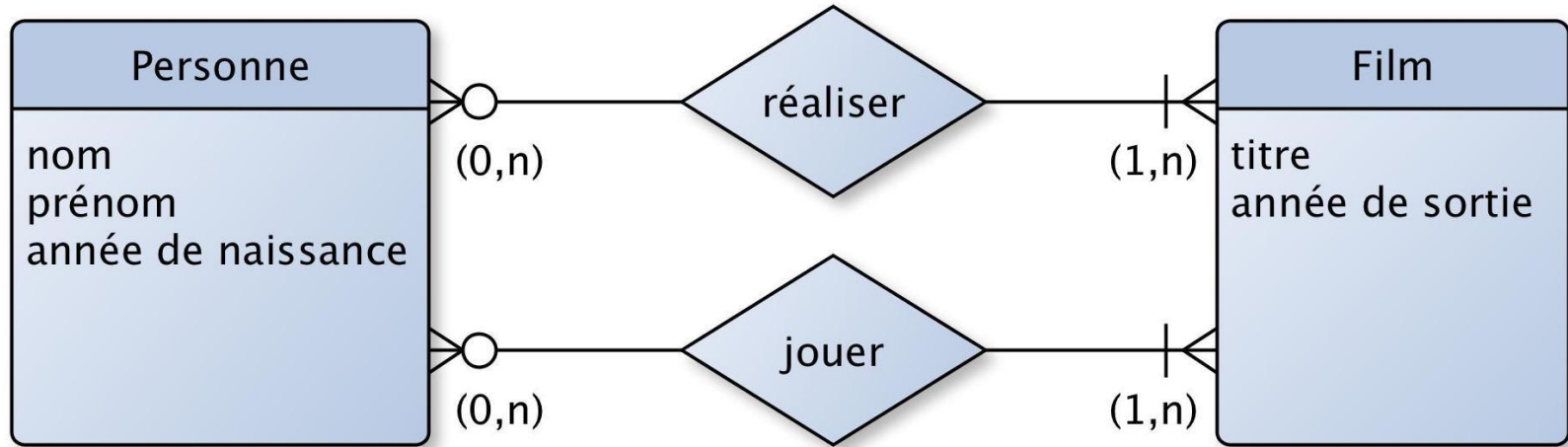
**Le choix des cardinalité est ESSENTIEL (moment décisif). Moment délicat de la modélisation.**



On peut représenter le MCD sous la forme d'un diagramme.

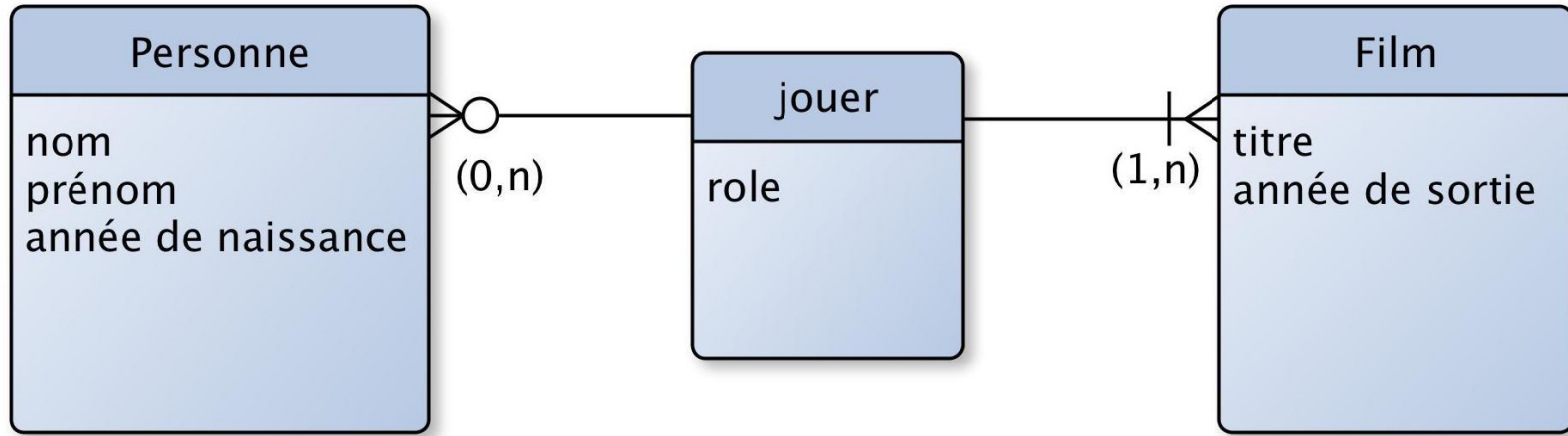


NB. Deux entités peuvent avoir plusieurs associations différentes...



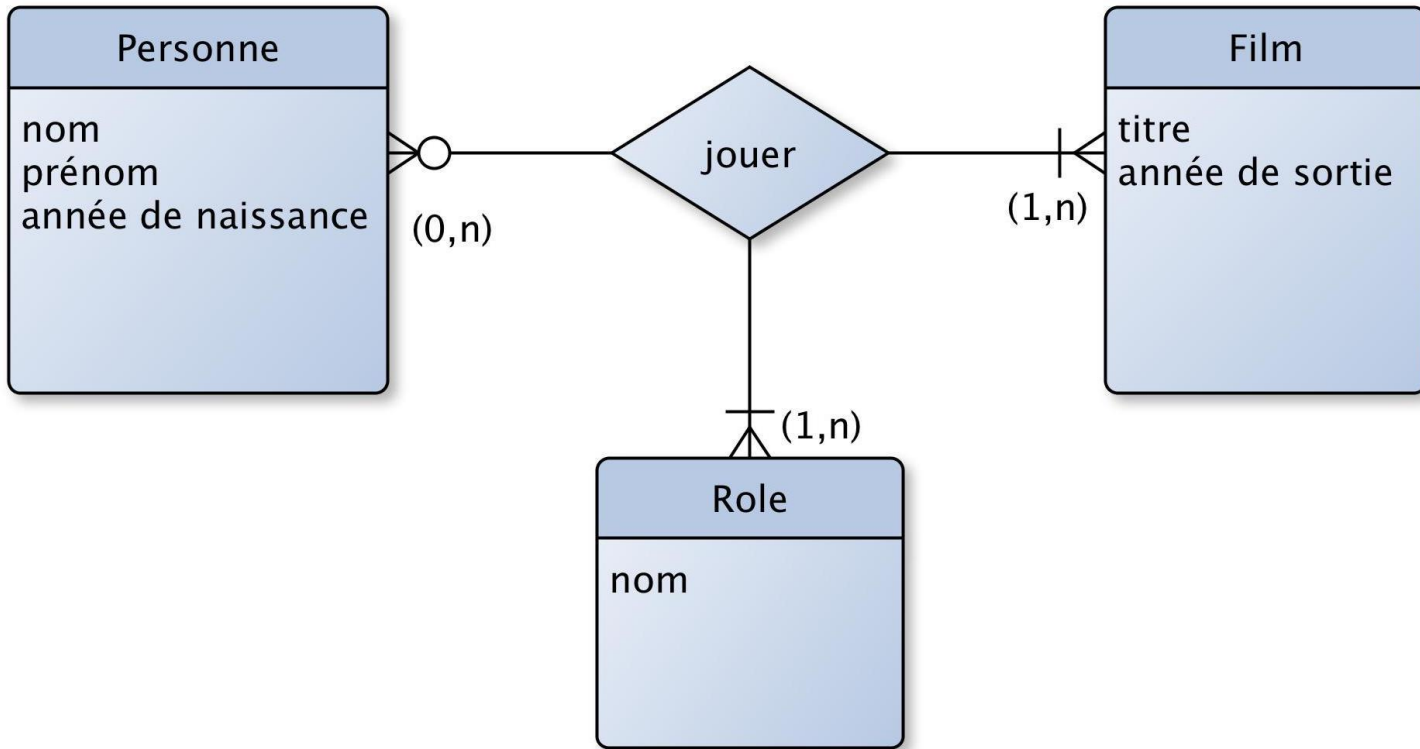
Exemple d'**association plurielle** : des associations différentes relient les mêmes entités.  
NB. Une association peut avoir des attributs...





**Comment faire si un acteur incarne plusieurs personnages dans un même film ?**

Par ex., Guillaume Gallienne dans *Les garçons et Guillaume, à table !* (son propre rôle et celui de sa mère) ou Christian Bale dans *Batman* (Bruce Wayne et Batman).

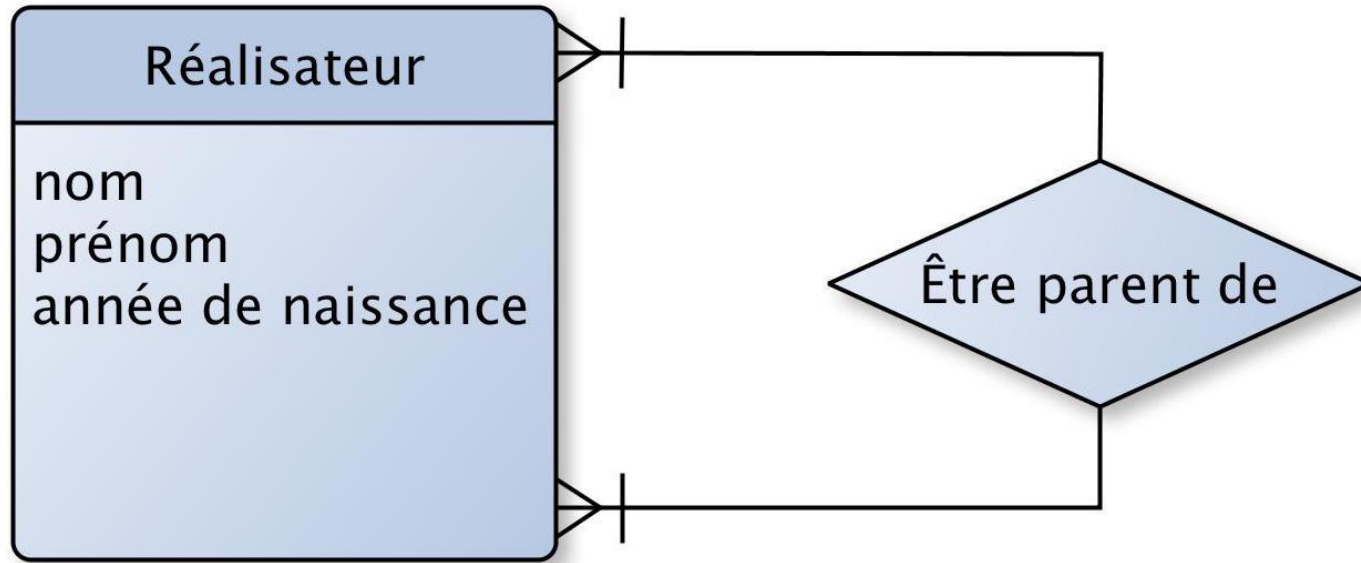


Une association n'est pas toujours binaire. Cas ici d'une **association ternaire**.

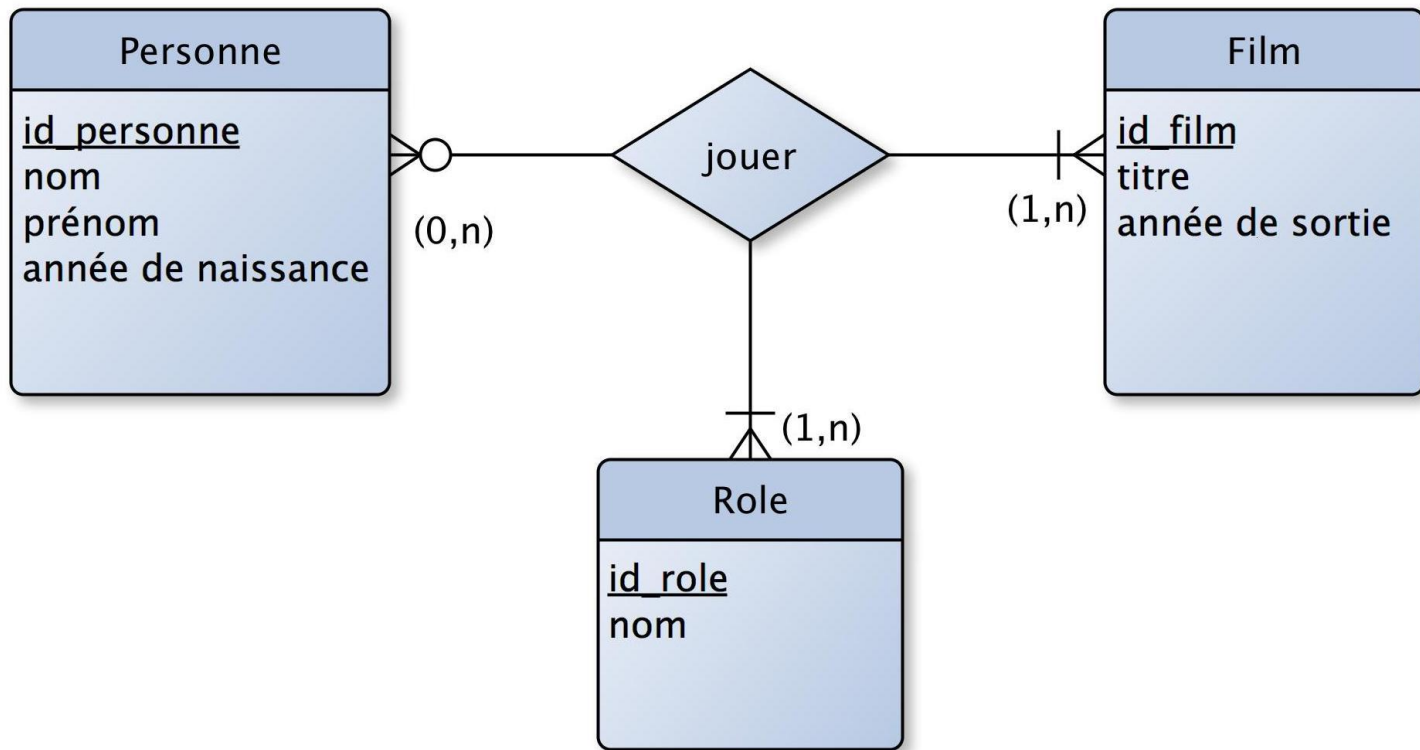
**Dimension ou "arité" d'une association** : le nombre d'entités contenus dans l'association (n-aire)

NB : les cas où  $n > 2$  sont rares et vite problématiques...

**Et si  $n = 1$  ?**



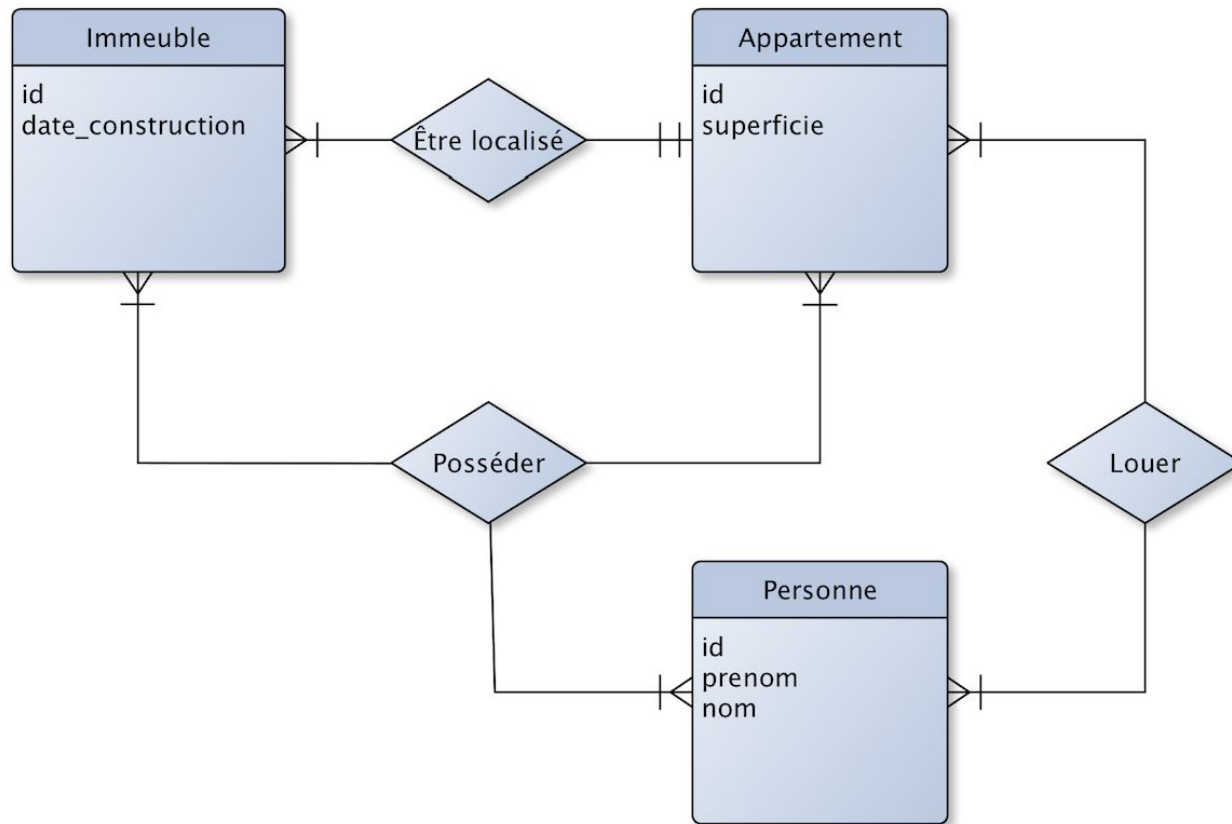
Exemple d'**association 1-aire** ("**unaire**") : Francis Ford et Sofia Coppola, Michel et Jacques Audiard.  
Pour finaliser le diagramme, il faut repérer (souligner) les clés primaires.



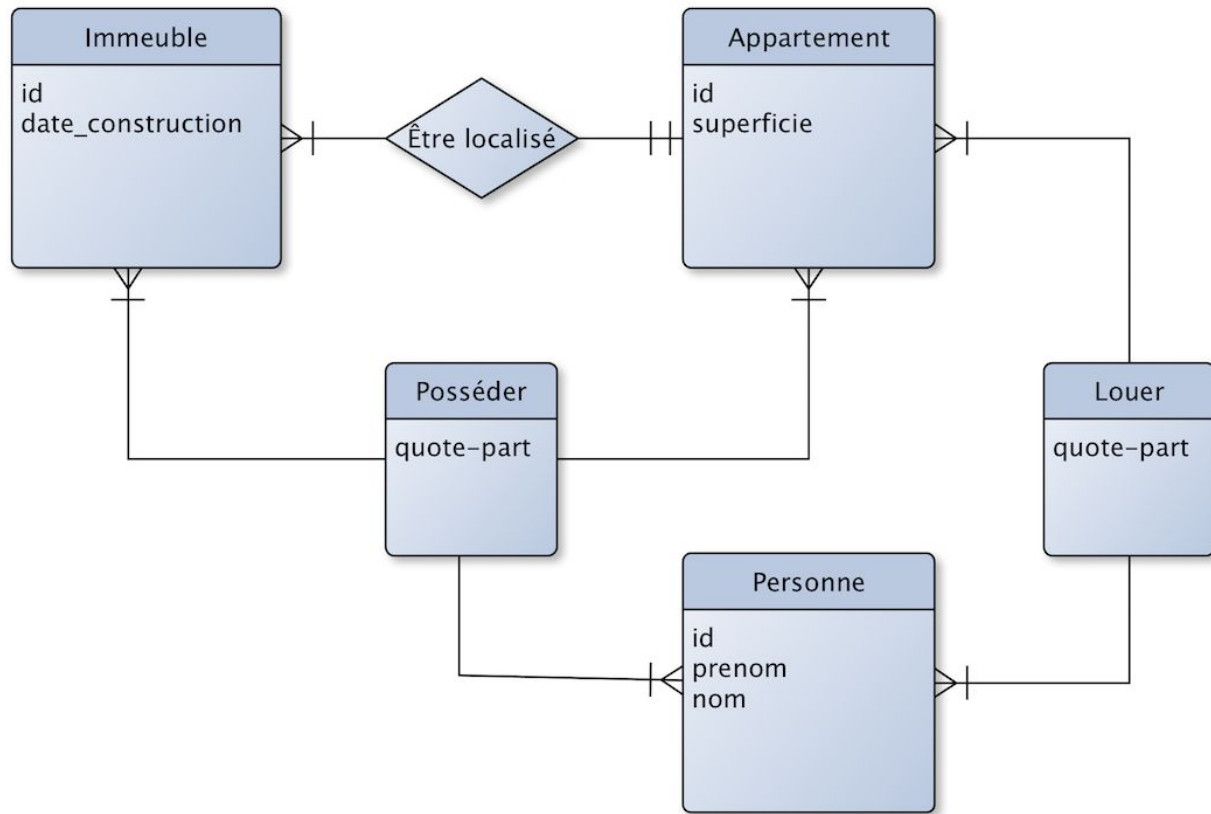


# MCD et cardinalités, exercice

- Immeuble / Appartement
- Ouvrage / Auteur / Type
- Musicien / Groupe / Album
- JO / Athlète / Pays
- Ville / Département / Région
- Ville / Département / Région / Pays ?
- Immeuble / Appartement / Propriétaire / Locataire



Immeuble, MCD



Immeuble, MCD + attributs sur les associations



# Modélisation – quelques règles

## Normalisation des attributs

- remplacer les attributs (factorisation) qui apparaissent dans plusieurs entités par une entité spécifique reliée aux premières via des associations (*Client.adresse* et *Fournisseur.adresse*).
- Ne pas avoir d'attribut calculable à partir des autres (*âge*).

**Identifiants.** Chaque entité possède au moins un identifiant, **éventuellement formé de plusieurs attributs**.

Normalisation des attributs d'association : ex. Un Lecteur emprunte un Livre ; que se passe-t-il si le lecteur emprunte plusieurs fois le même livre ?





# Modélisation : les formes normales

Une forme normale désigne un type de relation particulier entre les entités.

Le but essentiel de la normalisation est d'**éviter les anomalies transactionnelles** pouvant découler d'une mauvaise modélisation des données et ainsi éviter un certain nombre de problèmes potentiels tels que les anomalies de lecture, les anomalies d'écriture, la redondance des données et la contre performance.

Les formes normales s'emboîtent les unes dans les autres, tant et si bien que le respect d'une forme normale de niveau supérieur implique le respect des formes normales des niveaux inférieurs. Il existe huit formes normales, les trois premières étant les plus connues et usitées.

En pratique, la première et la deuxième forme normale sont nécessaires pour avoir un modèle relationnel juste. Les formes normales supplémentaires ont leurs avantages et leurs inconvénients.



# Modélisation : les formes normales

Les avantages sont :

- de limiter les redondances de données ;
- de limiter les incohérences de données qui pourrait les rendre inutilisables ;
- d'éviter les processus de mise à jour (réécritures).

Les inconvénients sont :

- des temps d'accès potentiellement plus longs si les requêtes sont trop complexes ;
- une plus grande fragilité des données étant donné la non redondance (lecture impossible)
- un manque de flexibilité au niveau de l'utilisation de l'espace disque

Pour des petites bases de données, se limiter à la troisième forme normale est généralement une des meilleures solutions d'un point de vue architecture de base de données.



# Première forme normale (1FN)

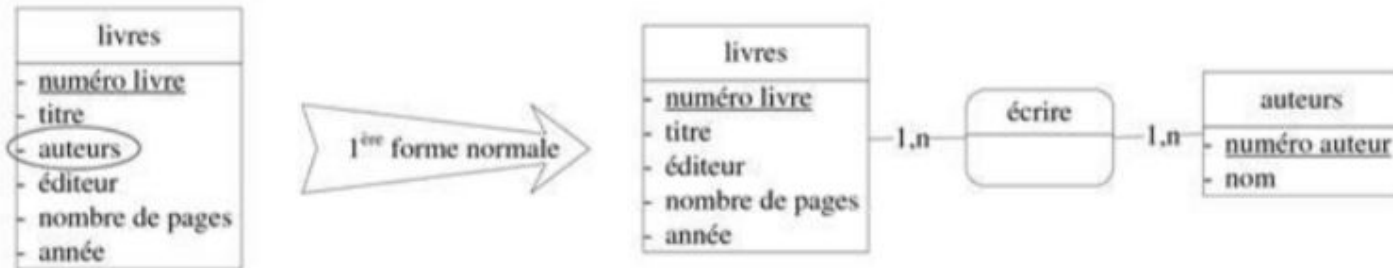
Relation dont tous les **attributs** :

- contiennent une **valeur atomique** (les valeurs ne peuvent pas être divisées en plusieurs sous-valeurs dépendant également individuellement de la clé primaire) ;
- contiennent des **valeurs non répétitives** (tout attribut a une valeur et non pas un ensemble ou une liste de valeurs) ;
- sont **constants dans le temps** (utiliser par exemple la date de naissance plutôt que l'âge).

Le non-respect de deux premières conditions de la 1FN rend la recherche parmi les données plus lente parce qu'il faut analyser le contenu des attributs. La troisième condition quant à elle évite qu'on doive régulièrement mettre à jour les données.

# 1FN, exemple

Si un attribut prend plusieurs valeurs, alors ces valeurs doivent faire l'objet d'une entité supplémentaire, en association avec la première.



*Application de la première forme normale  
Il peut y avoir plusieurs auteurs pour un livre donné.*



## Deuxième forme normale (2FN)

Respecte la deuxième forme normale, la relation respectant la 1FN et dont :

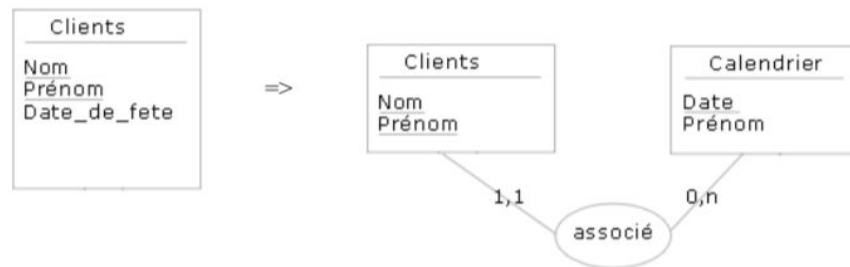
- Tout attribut ne composant pas un identifiant dépend d'un identifiant (de cet identifiant entier, non pas seulement d'une de ses parties).

Le non-respect de la 2FN entraîne une redondance des données qui encombre alors inutilement la mémoire et l'espace disque.

## 2FN, exemple

Un fleuriste stocke des données clients (nom, prénom, date\_de\_fête) afin d'offrir des fleurs lors de la fête de ses clients.

Dans l'entité *Client* la date de fête d'un client ne dépend pas de son identifiant en entier mais seulement de prénom. Elle ne doit pas figurer dans l'entité *Client*, il faut donc faire une entité calendrier à part, en association avec l'entité clients.





## 2FN, exercice

Gestion de compte bancaire

```
Operation(  
    numero_compte,  
    code_operation,  
    date_operation,  
    nom,  
    prenom,  
    libelle_operation,  
    montant)
```



## 2FN, solution

On note que :

- *nom* et *prenom* dépendent fonctionnellement de *numero\_compte*
- *libelle\_operation* dépend fonctionnellement de *code\_operation*

**Solution.** On va obtenir les entités et associations suivantes :

- Compte(numero\_compte, nom, prenom)
- Operation(code\_operation, date\_operation, libelle\_operation, montant)





# Conception des bases de données : modèle Entité-Association

Deux phases :

1. Réalisation du modèle conceptuel (MCD)

La modélisation conceptuelle est une étape délicate : définition des données, de leur mode d'évolution dans le temps et des relations entre elles. Formalisme de type Entité-Association.

2. **Traduction du MCD en un modèle logique, relationnel (MRD) dans notre cas**  
**Définition de l'ensemble des objets manipulables par le SGBD-R.**

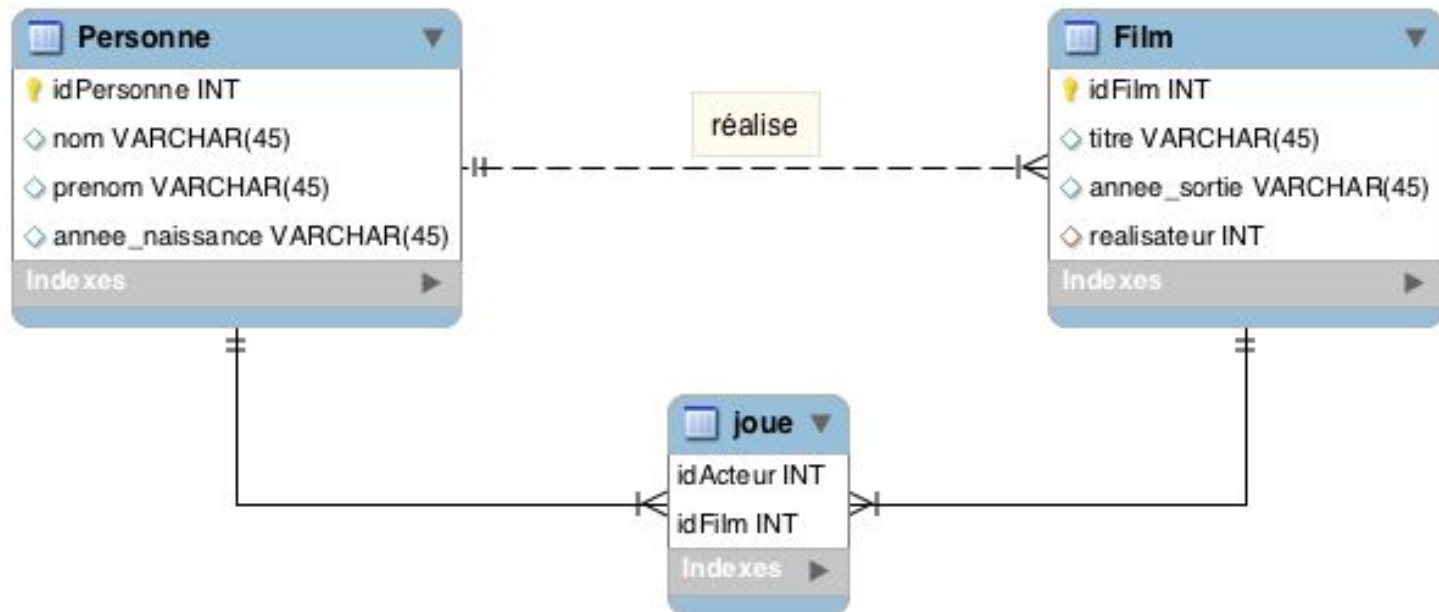



# Modèle relationnel (MRD)

Transformation du modèle conceptuel (MCD) en modèle logique, ici relationnel (MRD).

MCD	MRD
entité	table
attribut (propriété)	attribut
identifiant	clé primaire
association et cardinalité	clé étrangère (relation 1:n)   table (relation n:n)

$(0|1:1) \text{ — } (0|1:n) \rightarrow$  relation **1:n**  $\rightarrow$  la relation est matérialisée par l'ajout d'une clé étrangère.  
 $(0|1:n) \text{ — } (0|1:n) \rightarrow$  relation **n:n**  $\rightarrow$  la relation donne lieu à la création d'une table.





# MRD exercice 1/3 : Immeuble

Pour l'exemple "Immeuble / Appartement / Propriétaire" :

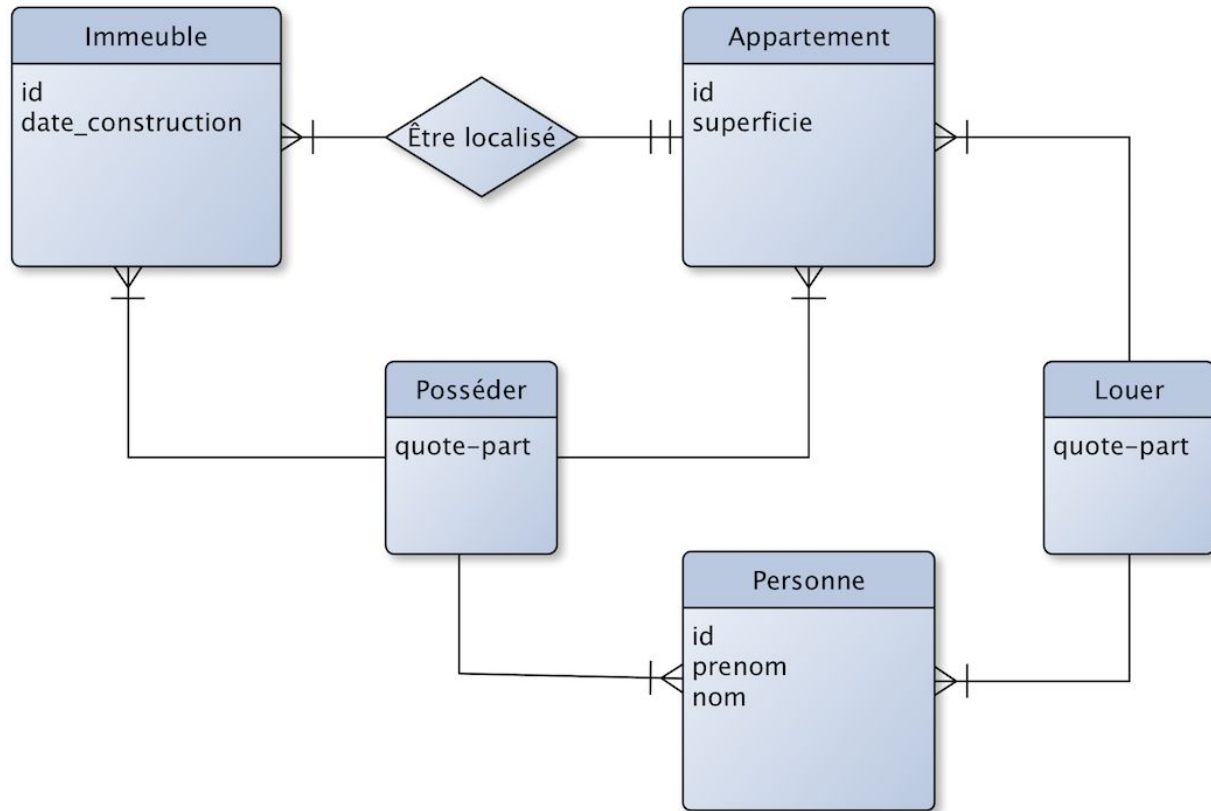
- Dessiner le MRD
- Installer MySQL Workbench  
<https://www.mysql.com/fr/products/workbench/>  
`sudo apt install mysql-workbench`
- Construire le MRD avec Workbench
- Exporter le script SQL pour générer la base



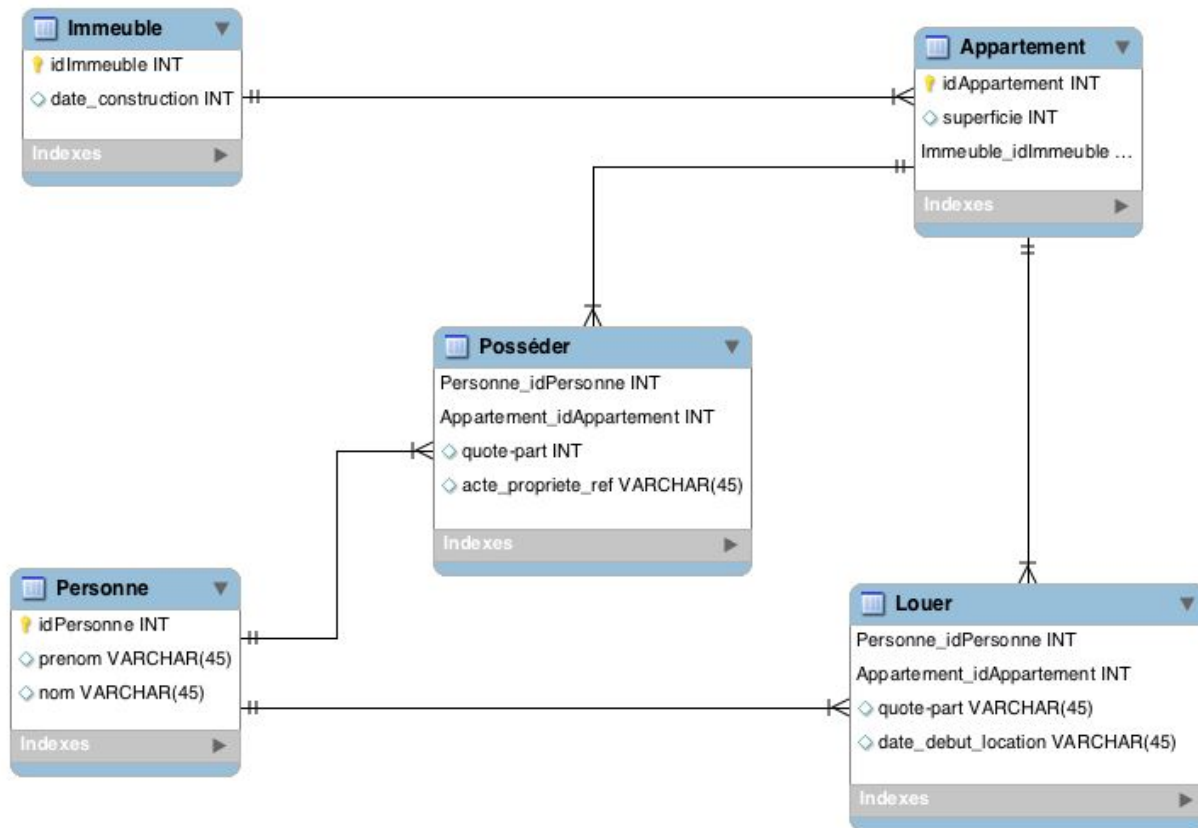
## *Identifying/non-identifying relationships*

<https://forums.mysql.com/read.php?125,590251,594011>

- **Identifying relationship** : lorsque l'identité/l'existence de l'entité enfant dépend uniquement de l'entité mère. Pas d'entité mère signifie pas d'entité enfant.  
Par **exemple**, si vous supprimez une commande (table *order*), vous voulez probablement que toutes les factures relatives à la commande disparaissent également (*order\_line\_item*). Ainsi, l'identité d'une facture dépend uniquement de l'existence d'une commande correspondante.  
→ **On inscrit la clé primaire du parent dans la colonne clé primaire de l'enfant (avec les autres colonnes nécessaires garantir l'unicité).**
- **Non-identifying relationship** : l'entité enfant peut se suffire à elle-même sans l'entité mère.  
Par **exemple**, dans le cas de tables *author* et *book*. On peut stocker des données sur un livre, sans savoir qui en est l'auteur. L'identité/l'existence du livre est indépendante de l'information de l'auteur.  
→ **On inscrit la colonne clé primaire du parent dans la table de l'enfant, mais pas dans la clé primaire.**



Immeuble / Appartement, MCD



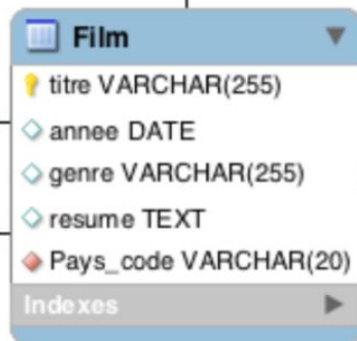
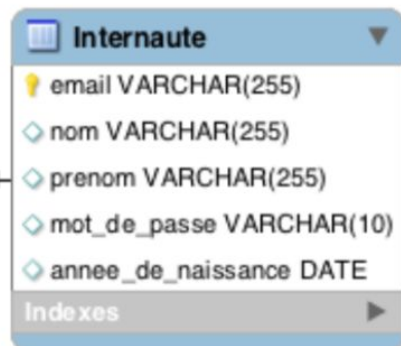
Immeuble / Appartement, MRD



## MRD exercice 2/3 : idem, Films

- Entités et propriétés
  - Un *Artiste* : nom, prénom, naissance
  - Un *Film* : titre, année, genre, résumé, pays
  - Un *Internaute* : id ?, nom, prénom, passwd, naissance
- Associations
  - Artiste réalise UN / joue dans DES films
  - Un Artiste ne tient qu'un rôle par film
  - Un internaute note des films





# MRD exercice 3/3 : Base prosopographique des architectes diocésains



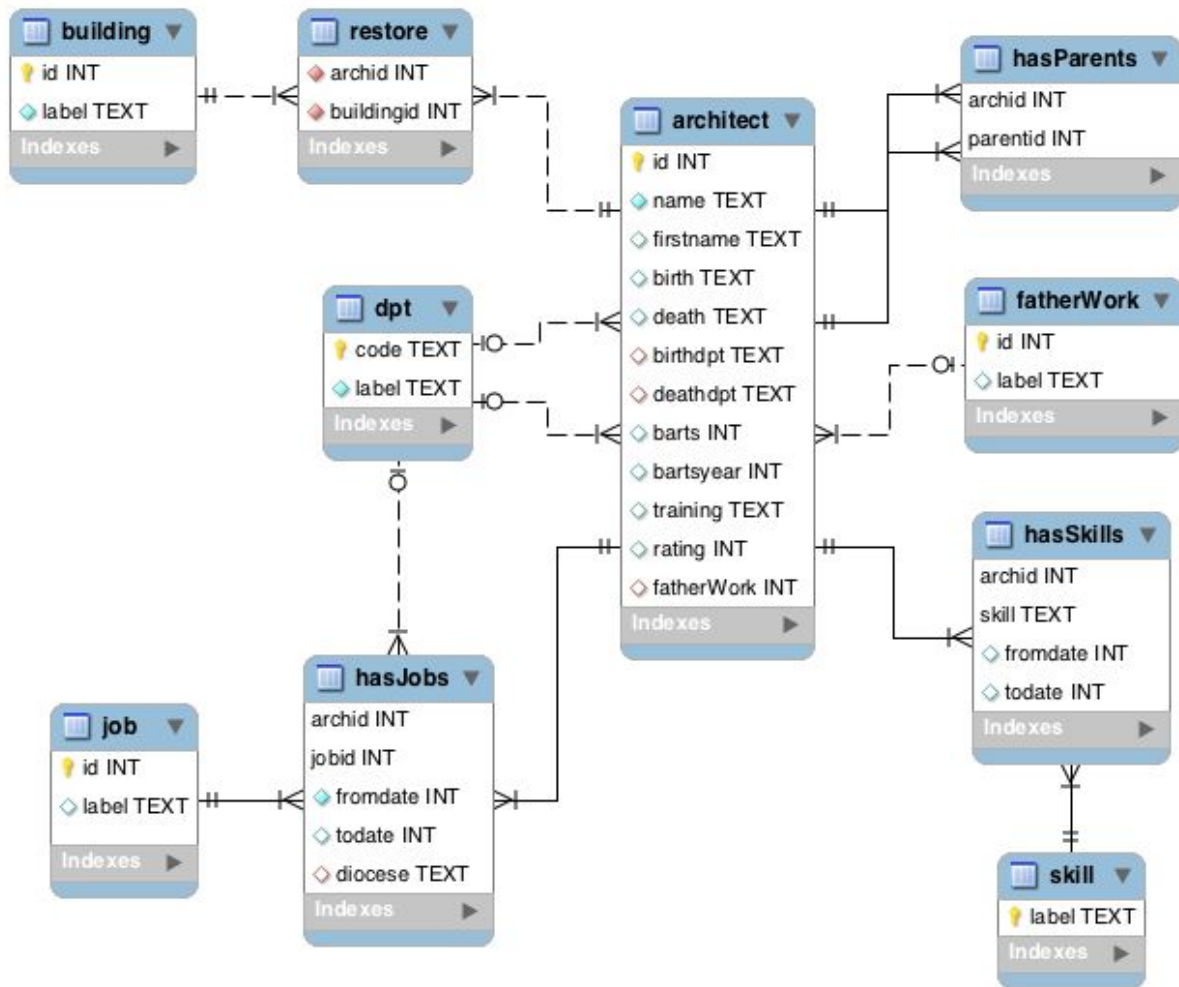
Dans les notices d'architectes,  
vous avez relevé les informations suivantes :

- *name*: nom
- *firstname*: prénom
- *birth*: date de naissance
- *death*: date de mort
- *birthdpt*: département de naissance
- *deathdpt*: lieu de mort
- *barts*: formation aux beaux-arts (oui|non)
- *bartsyear*: année de promotion aux beaux-arts
- *skill*: domaine(s) de compétence (cathédrale|séminaire|églises|tout)
- *rating*: appréciation de la hiérarchie en 1853
- *fatherWork*: milieu professionnel du père (liste d'autorité)
- *hasParent*: parenté avec un autre membre de la base
- poste(s) d'architecte diocésain :
  - *job*: attributions (architecte|inspecteur|suppléant|honoraire)
  - *fromdate*: date de début (unique)
  - *todate*: date de fin (unique)
  - *diocese*: diocèse (unique)
- chantier(s): *building* et *restore*

Associations :

- Un architecte (*Architect*) restaure (*restore*) des bâtiments (*Building*).
- Un architecte exerce (*hasJobs*) une / des fonctions (*Job*).
- Un architecte exerce cette fonction dans un département (*Dpt*).
- Un architecte peut être le parent (*hasParents*) d'un autre.
- Un architecte a (*hasSkills*) des spécialités (*Skill*).

1. Dessiner le MCD
2. Créer le diagramme du MRD avec Workbench.





# Générer le script de création de la base

- Dans MySQL Workbench :  
File > Export > Forward Engineer SQL CREATE Script...
- Lire le **script SQL** de création de la base de données
- Découvrir les premières **instructions** (*statements*)
- **Problème** : on est dépendant du SGBDR MySQL
- On va écrire à la main notre premier script SQL...



# DB Browser for SQLite

- Installer DB Browser (3.10.1 et PAS LA BETA !) : <https://sqlitebrowser.org/>
- Créer la base de données `architect:Fichier > Nouvelle base de données...`
- Vous pouvez l'enregistrer sur le bureau (`architect.db`).
- Explorer sa structure et ses données : c'est évidemment vide...
- Il faut créer les tables...

# CREATE TABLE Statement



[https://www.sqlite.org/lang\\_createtable.html](https://www.sqlite.org/lang_createtable.html)

<http://www.sqlitetutorial.net/sqlite-create-table/>

```
CREATE TABLE [IF NOT EXISTS] [schema_name].table_name (  
    column_1 data_type PRIMARY KEY,  
    column_2 data_type NOT NULL,  
    column_3 data_type DEFAULT 0,  
    table_constraint  
) [WITHOUT ROWID];
```

- **Attributs** : nom de la table, nom de la base de données, nom des colonnes et leur **data type** et les contraintes associées.
- **Clauses** [optionnelles] : IF NOT EXISTS
- **Contraintes** : PRIMARY KEY, UNIQUE, NOT NULL...
- **Data Types** : INTEGER, TEXT, BOOLEAN

# CREATE TABLE, exemple



```
CREATE TABLE architect (  
    id            INTEGER PRIMARY KEY,  
    name          TEXT NOT NULL,  
    firstname     TEXT,  
    birth         TEXT,  
    death         TEXT,  
    birthdpt      TEXT NULL REFERENCES dpt(code),  
    deathdpt      TEXT NULL REFERENCES dpt(code),  
    barts         BOOLEAN NOT NULL DEFAULT 0,  
    bartsyear     INTEGER,  
    training      TEXT,  
    rating        INTEGER,  
    fatherWork    INTEGER NULL REFERENCES fatherWork(id)  
);
```



## CREATE TABLE, exercice

- Exécuter l'exemple dans DB Browser.
- Explorer la structure et les données de la base `architect`.
- Rédiger et exécuter le script de création de la base de données `architect`.
- Créer un enregistrement dans notre base...