23.10.18

**SQL**

Introduction aux bases de données relationnelles

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
* Ecrire des requêtes, exporter et exploiter les résultats

SQLite / MySQL = plein de moteurs de base de données qui implémentent

MySQL = architecture client/serveur => serveur qui exécute le SQL (un compte, un log, etc.)

SQLite = un fichier (+ light que MySQL) permet de consulter la base

**Une première base :**

Définir des identifiants pour créer des bases de données à rentrées multiples pouvant se répéter.

Préférer faire deux entités plutôt que de rajouter des colonnes = deux entités partagent les mêmes attributs.

Notions :

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

Problème :

* Comment représenter le lien entre un film et un réalisateur ?
* Clé étrangère : attribut qui référence le champ d’une autre table

Problème : le référencement par clé étrangère ne fonctionne que dans le cas où nous considérons qu’un film est réalisé par un unique réalisateur. Or un film peut être coréalisé.

On a besoin d’une table d’association > une table transitoire. Une liaison entre deux entités.

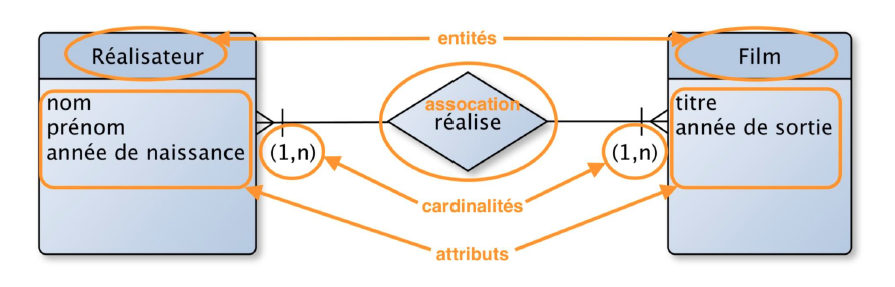
* Association : une liaison qui a une signification précise entre plusieurs entités

(« un réalisateur ? un film »)

**Notions, interro surprise :**

* Entité : concept identifiés qui se caractérisent par des attributs communs
* Attribut : caractérise une entité
* Table : une table par entité, c’est le tableau dans lequel on inscrit les attributs de l’entité
* Enregistrement : enregistre les attributs de la table
* Association : une liaison qui a une sémantique précise entre 2 entités
* Clé primaire : attribut qui identifie un enregistrement / elle est unique
* Clé étrangère : id qui n’est pas obligatoirement unique, c’est un attribut qui référence la clé primaire d’une autre entité.

**Modèle conceptuel :**



MCD, exercice :

3 entités : clients, articles, fournisseurs

* Quelles sont les associations entre ces entités ?
* Placer les attributs ci-dessous dans le schéma entités associations :

30.10.18

**Pour établir un Modèle Relationnel (MRD) :**

**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’E° dans le temps et des relations entre elles. Formalisme de type Entité-Association.

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

**Etape 1 :**

* **Modèle conceptuel :** Décrit les **entités** (films, réalisateurs), 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 utiliser 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 peut 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*)

**La cardinalité** est le nb 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 1 à n film(s) (1, n)

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

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

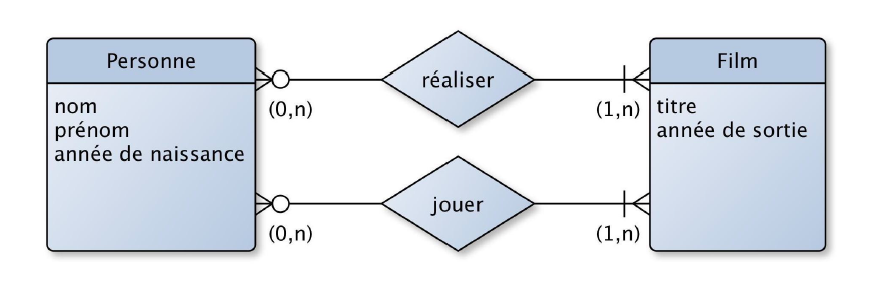
MCD. On peut représenter cette association et sa cardinalité sous la forme d’un schéma. C’est le modèle conceptuel de données (MCD).

Réalisateur(0,n)--------------réalise------------(1,n)film(s)

Il peut y avoir des **associations plurielles**: des associations différentes relient les mêmes entités > c’est parce qu’il y a des associations plurielles que la cardinalité ne peut pas se placer entre deux entités, mais entre une entité et une association.

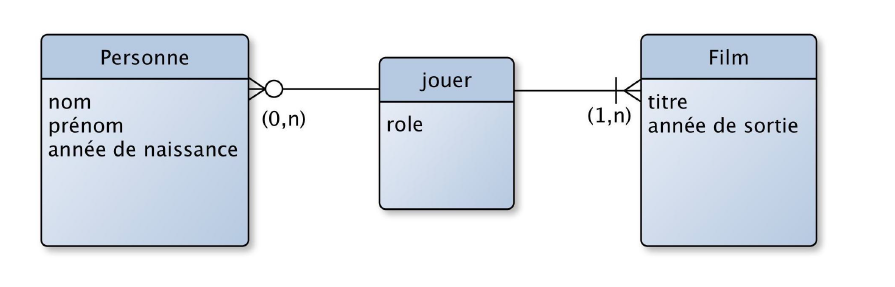
**Une association n’est pas toujours binaire**: par exemple si un acteur joue plusieurs rôles (Ch Bale qui joue Batman et B. Wayne) c’est ce qu’on appelle une **association ternaire**. Dimension ou « arité » d’une association : le nombre d’entités contenus dans l’association (n-aire). NB : les cas où un père a une fille.

Des **associations unaires** existent également : quand l’association renvoie à l’entité de laquelle elle émane.

Exemple d’association plurielle : des associations différentes relient les mêmes entités.

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).

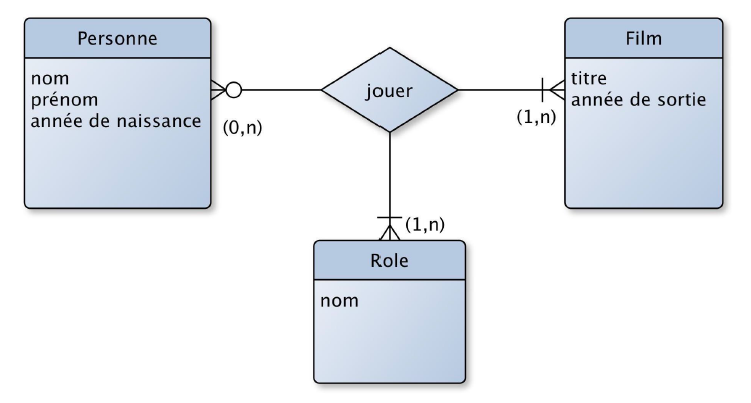


Une association n’est pas toujours binaire. Cas ici d’une association ternaire.

Dimension ou “arité” d’une association : le nombre d’entités contenus dans l’association (n-aire)

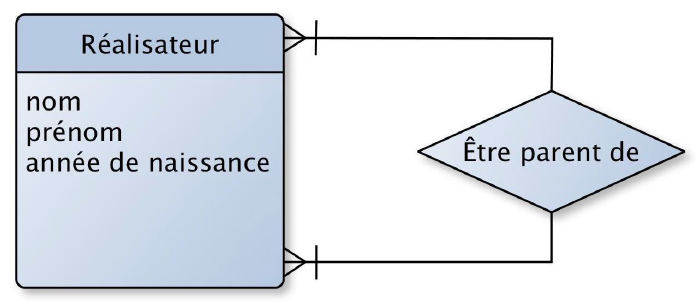
NB : les cas où n > 2 sont rares et vite problématiques…

Et si n = 1 ?

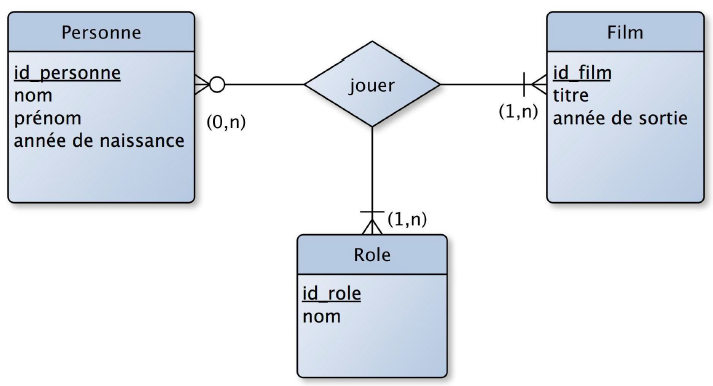


Exemple d’association 1-aire (“unaire”) : Francis Ford et Sofia Coppola, Michel et Jacques Audiard.

Pour finaliser le diagramme, il faut repérer (souligner) les clés primaires.



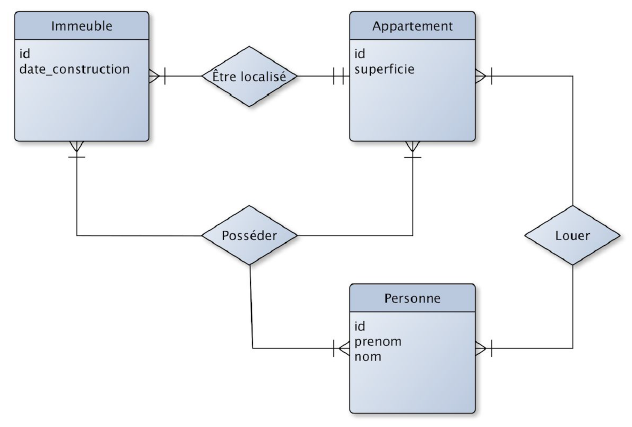
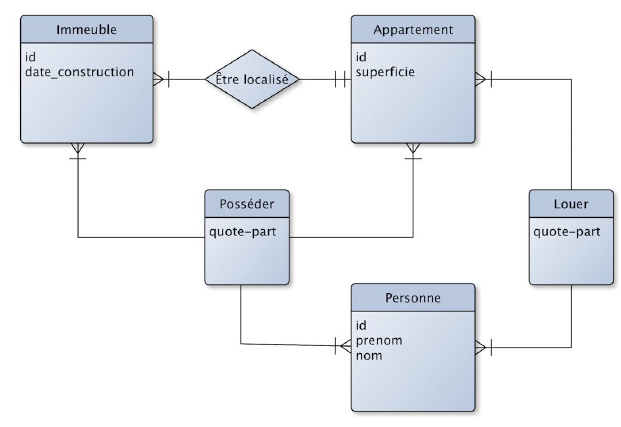
Association ternaire avec « id » :



13.11.18

*Exercice : Sur la modélisation = appartement ; propriétaire ; locataire ; immeuble.*

* Locataire et propriétaire = une entité personne qui a comme attribut = propriétaire et/ou locataire.



**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’associations : 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 qui peuvent découler de mauvaises modélisation des données, et ainsi éviter un certain nb 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’emboitent les unes dans les autres, tant et si bien que le respect d’une forme normale de niveau sup 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 mode le relationnel juste. Les formes normales supplémentaires ont leurs avantages et leurs inconvénients.

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

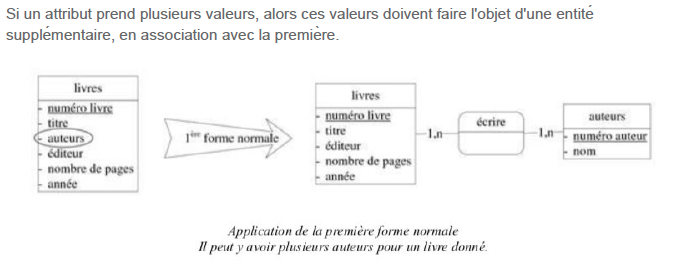
Pour des petites bases de données, se limiter à la troisième forme normale est généralement une des meilleures solutions d'un point de vue architecture de base de données.

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.

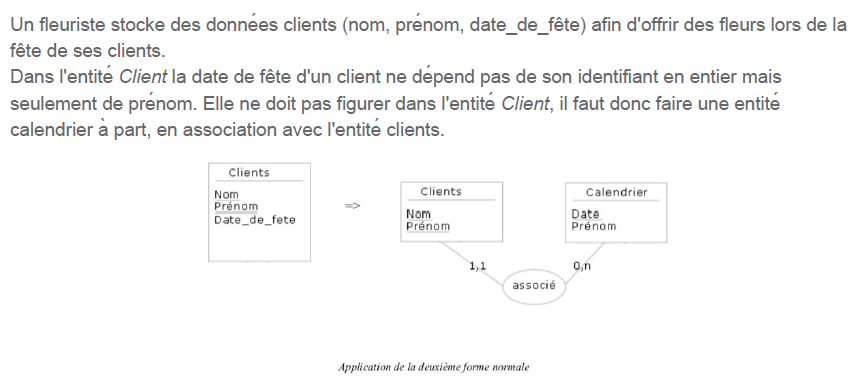


Deuxième forme normale (2FN) :

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 entraine une redondance des données qui encombrent alors inutilement la mémoire et l’espace.



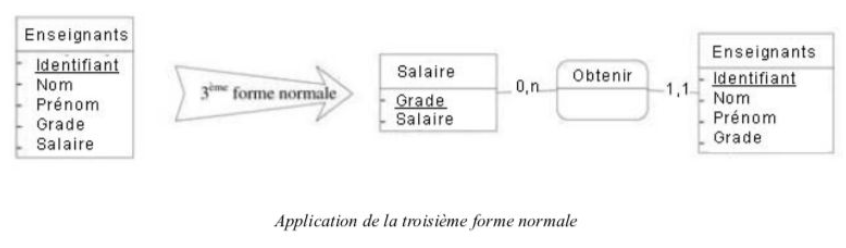
Troisième forme normale (3FN) :

Relation respectant la 2FN et dont :

● tout attribut ne composant pas un identifiant dépend d'un identifiant (où tout attribut

n'appartenant pas à une clé ne dépend pas d'un attribut non clé).

Tous les attributs d'une entité doivent dépendre directement de son identifiant et d'aucun autre attribut. Si ce n'est pas le cas, il faut placer l'attribut pathologique dans une entité séparée, mais en association avec la première.

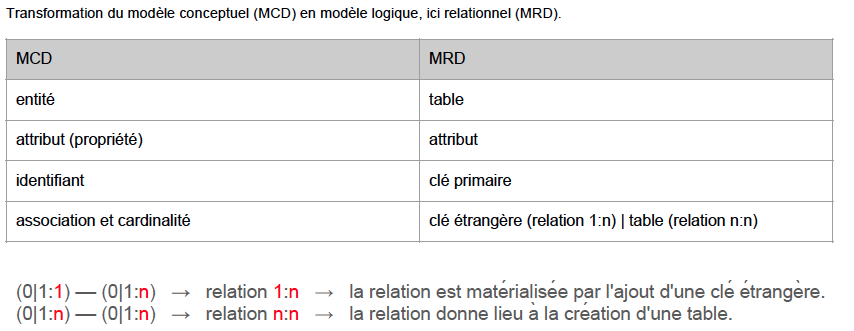


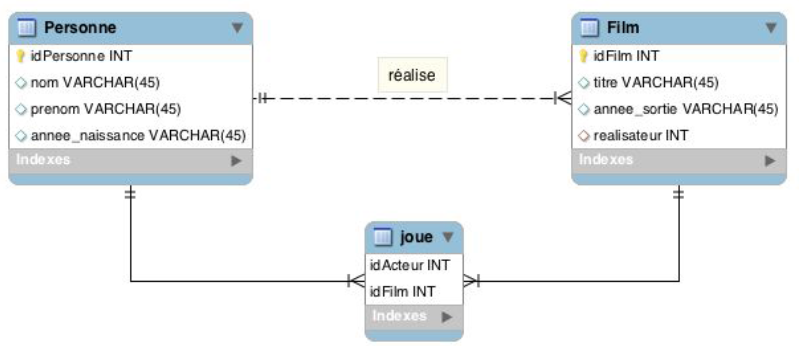
20.11.18

MRD ≠ MCD

Poser les cardinalités entre les entités – retenir les cardinalités maximales – définir la relation avec les cardinalités maximales (+ petit, + grand). Une fois qu’on a les cardinalités du modèle relationnel on peut « supprimer » les cardinalités du modèle conceptuel.

**Passage à l’étape 2 = Traduction du MCD en MRD**

****

****

**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.

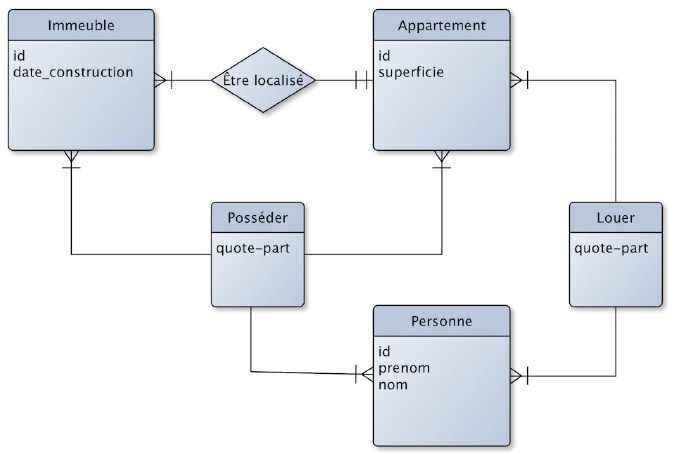
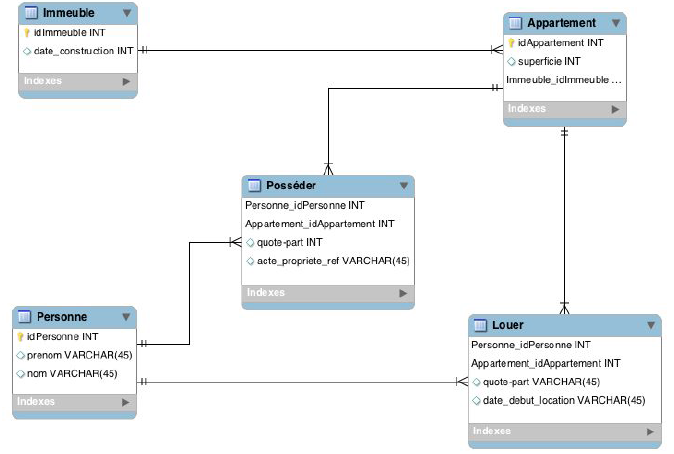
Par exemple, si vous supprimez une commande (table order), vous voulez probablement que toutes les factures relatives à la commande disparaissent également (order\_line\_item). Ainsi, l'identité d'une facture dépend uniquement de l'existence d'une commande correspondante.

→ **On inscrit la clé primaire du parent dans la colonne clé primaire de l'enfant** (avec les autres colonnes nécessaires **garantir l’unicité**).

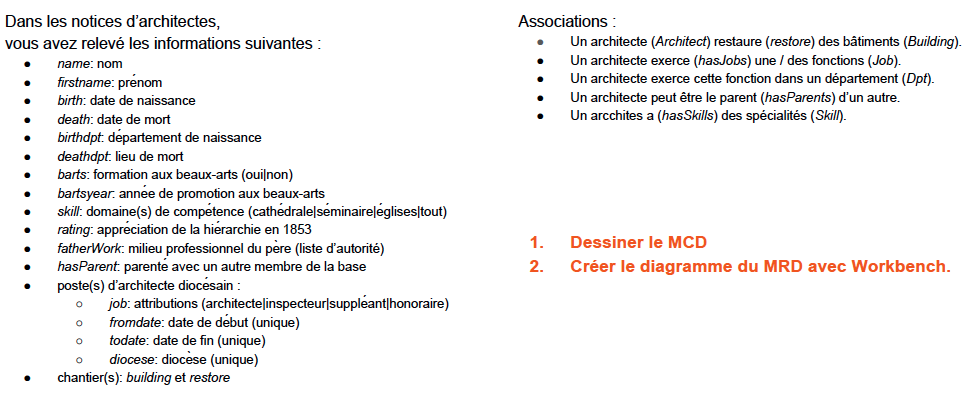
**Non-identifying relationship** : l'entité enfant peut se suffire à elle-même sans l'entité mère.

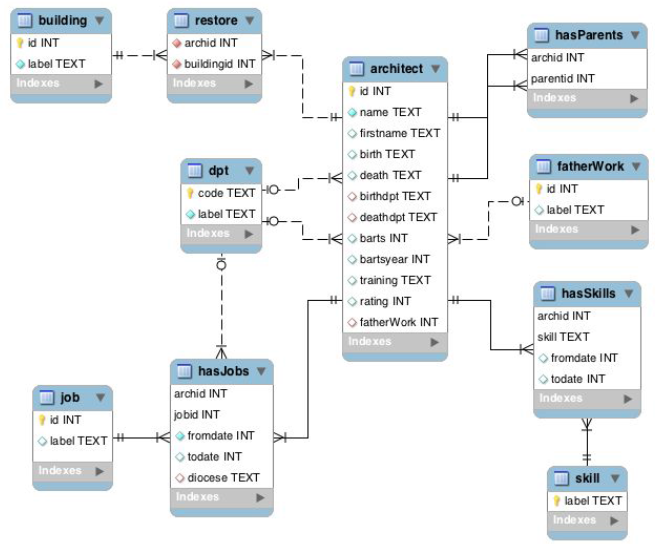
Par exemple, dans le cas de tables author et book. On peut stocker des données sur un livre, sans savoir qui en, est l'auteur. L'identité/l'existence du livre est indépendante de l'information de l'auteur.

→ **On inscrit la colonne clé primaire du parent dans la table de l'enfant, mais pas dans la clé primaire**.

****

**Exercice MDR = base prosopographique des architectes diocésains**

****

****

**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 notre premier script SQL…

● 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>

<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)

);