# Modèle relationnel

Dr N. BAME

# PLAN

- Modèle  relationnel
- Passage E/A vers modèle relationnel
- Normalisation du MR

# Rappels : construction des BD

1. Analyse des besoins

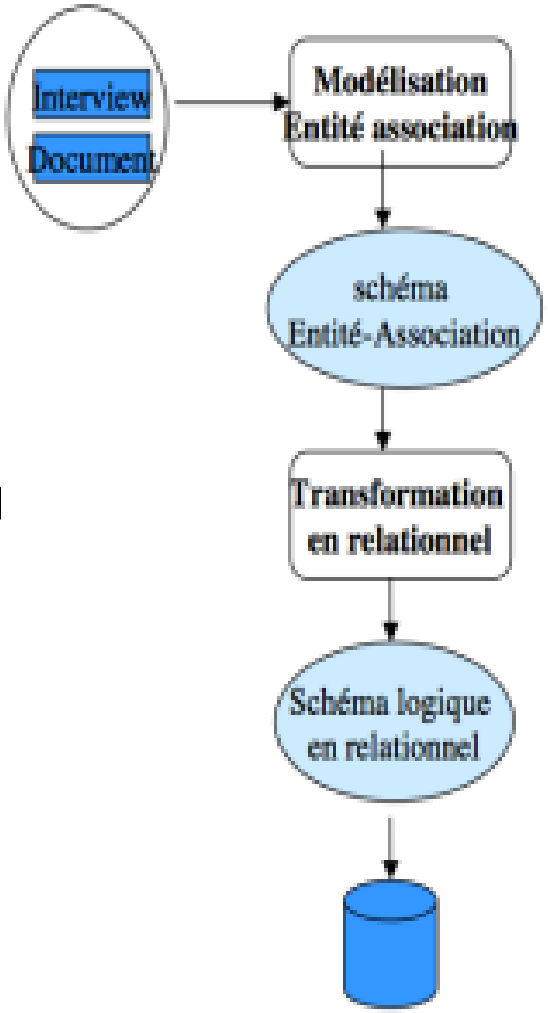
   →Document technique décrivant les données de l'applications

2. Conception du schéma de données

   →Schéma dans un langage de haut niveau (Entité-Association)

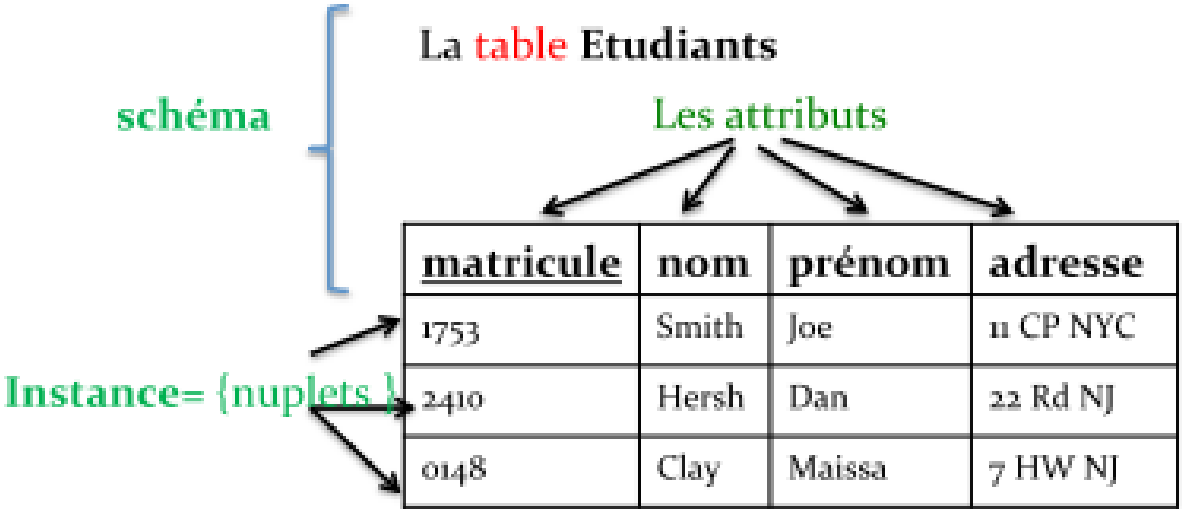
3. Implantation des données

   →Schéma relationnel



# Le modèle relationnel

- Données = collection de **tables (ou relations)**
- Table = **ensemble** de n-uplets avec les mêmes attributs
- N-uplet = séquence de **valeurs atomiques** (nombre, chaine de caractères, date, …)




La table **Etudiants**

Les attributs

| matricule | nom | prénom | adresse |
|---|---|---|---|
| 1753 | Smith | Joe | 11 CP NYC |
| 2410 | Hersh | Dan | 22 Rd NJ |
| 0148 | Clay | Maissa | 7 HW NJ |

La table **Modules**

| code | intitulé | niveau |
|---|---|---|
| LI341 | Bases de Données | L3 |
| LI345 | BD Web | L3 |
| LI399 | Crypto | L3 |

# Avantages du modèle relationnel

**Proche de la réalité** et **simple**

- La plupart des entités du monde réel partagent les mêmes attributs
- **Familiarité** des utilisateurs avec les **tableaux**

Repose sur des fondements solides

- Théorie des Ensembles
- Logique du Premier Ordre

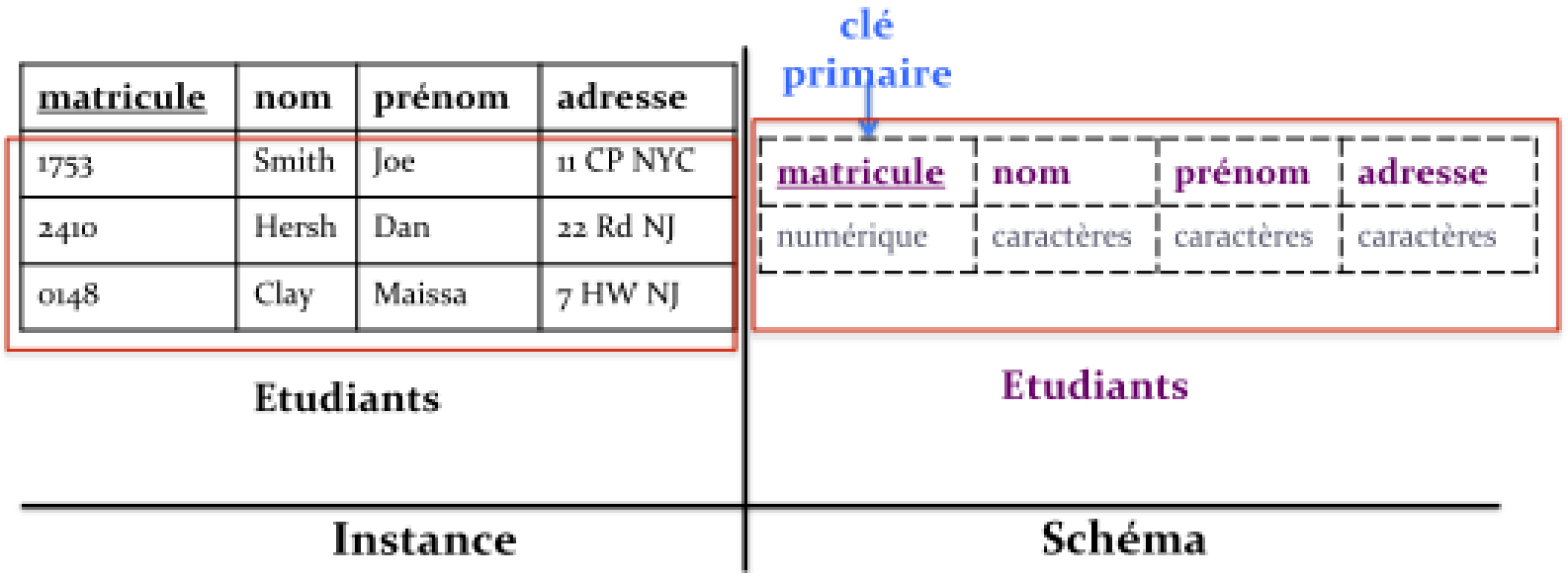
Doté de **langages de requêtes puissant**

- **Algèbre relationnelle**, Calcul des Prédicats
- **SQL** (Structured Query Language)

# Schéma relationnel : aperçu

Un schéma définit les instances valides

- ***Nom*** et contenu des ***attributs*** des ***tables***
- ***Domaine*** des attributs (nombre, chaine de caractères, date, …)
- ***Clés*** des tables et ***contraintes d'intégrité***



# Schéma relationnel

- Un schéma de relation

$$\mathbf{R(A_1:D_1 , A_2:D_2 , \ldots, A_n:D_n)}$$

- est composé
  - d'un ***nom*** *de relation* R et
  - d'un ***ensemble d'attributs*** $A=\{A_1, A_2 , \ldots, A_n\}$ définis sur *n* **domaines** de valeurs $D=\{D_1 , D_2 , \ldots, D_n\}$
  - Autre notation (sans domaines):

$$\boldsymbol{R(A_1 , A_2 , \ldots, A_n)}$$

- Un **schéma relationnel** est un **ensemble de schémas de relations**.

# Schéma relationnel

- *Schéma d'une BD* = schéma relationnel

  **S = {$SR_1$, $SR_2$, …, $SR_n$}** où **$SR_i$** schéma de relation

**Etudiants**(matricule :*Number*, nom : *Varchar*, prenom : *Varchar*, adresse : *Varchar*)

**Modules**(code : *Number*, intitule : *Varchar*, niveau : *Varchar*, salle : *Number*)

**Salles**(numero : *Number*, capacite : *Number*)

# Schéma relationnel

**EMP**(ENO, ENAME, TITLE)

**PROJ** (PNO, PNAME, BUDGET)

**WORKS**(#ENO,#PNO, RESP, DUR)

Les **attributs soulignés** sont les ***clés*** (identificateurs de n-uplets)

EMP

| ENO | ENAME | TITLE |
|---|---|---|

PROJ

| PNO | PNAME | BUDGET |
|---|---|---|

WORKS

| ENO | PNO | RESP | DUR |
|---|---|---|---|

# Clés

Sous-ensemble d'attributs qui ***identifient*** un n-uplet

Les valeurs de ces attributs doivent être ***distinctes***!

| matricule | nom | prénom | dateNaiss | adresse |
|---|---|---|---|---|
| 1753 | Smith | Joe | 1992-01-12 | 11 CP NYC |
| 9832 | Smith | Dan | 1989-04-03 | 22 Rd NJ |
| 4755 | Smith | Joe | 1994-11-29 | 7 HW NJ |
| 6842 | Roy | Ian | NULL | NULL |

Est-ce que les ensembles suivants peuvent être des clés?

{nom, prénom}

{nom, prénom, dateNaiss}

# Clé primaire

Clé dont **chacun** **des attributs est** **renseigné** (**≠** de **NULL**)

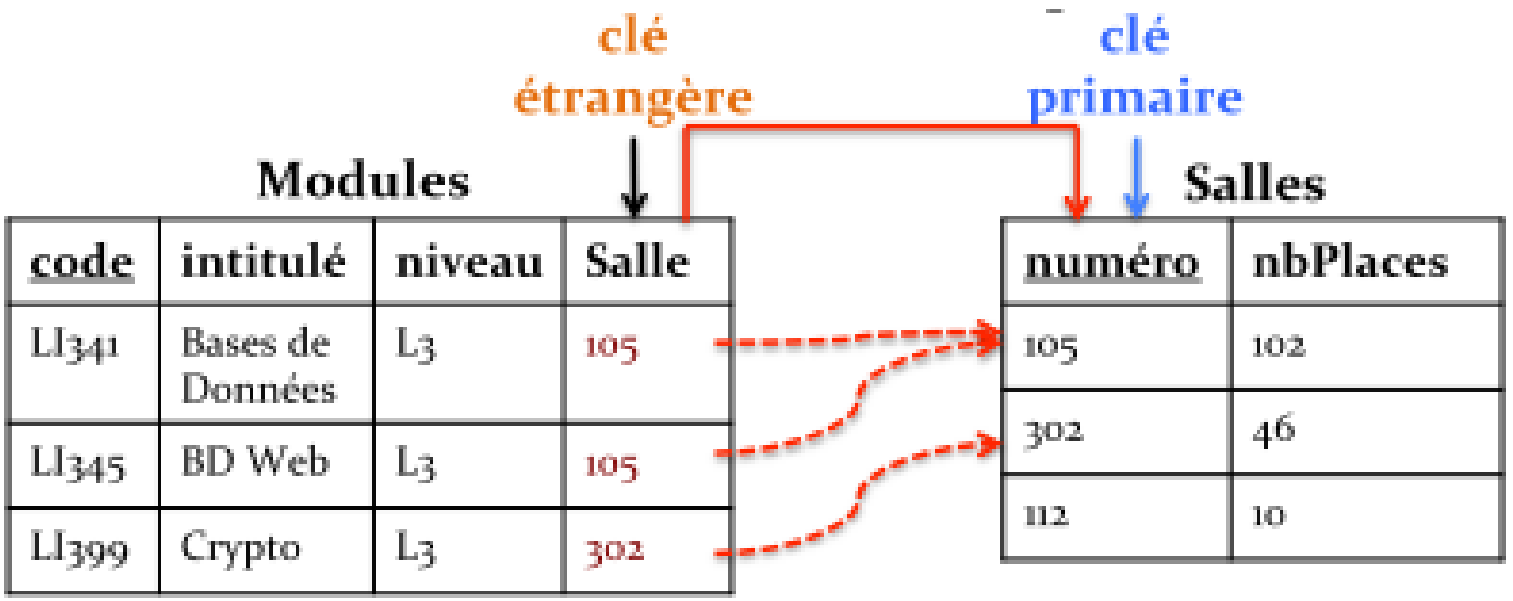
**Une seule** par table (peut servir à organiser physiquement la table)

| matricule | nom | prénom | dateNaiss | adresse |
|---|---|---|---|---|
| 1753 | Smith | Joe | 1992-01-12 | 11 CP NYC |
| 9832 | Smith | Dan | 1989-04-03 | 22 Rd NJ |
| 4755 | Smith | Joe | 1994-11-29 | 7 HW NJ |
| 6842 | Roy | Ian | NULL | NULL |

La clé primaire est {matricule}

# Clés étrangères

- Une clé étrangère = sous-ensemble d'attributs dont les **valeurs *proviennent*** des **clés primaires** de la **même table ou d'autre table**
- Mécanisme de **référencement** des n-uplets




| code | intitulé | niveau | Salle |
|---|---|---|---|
| LI341 | Bases de Données | L3 | 105 |
| LI345 | BD Web | L3 | 105 |
| LI399 | Crypto | L3 | 302 |

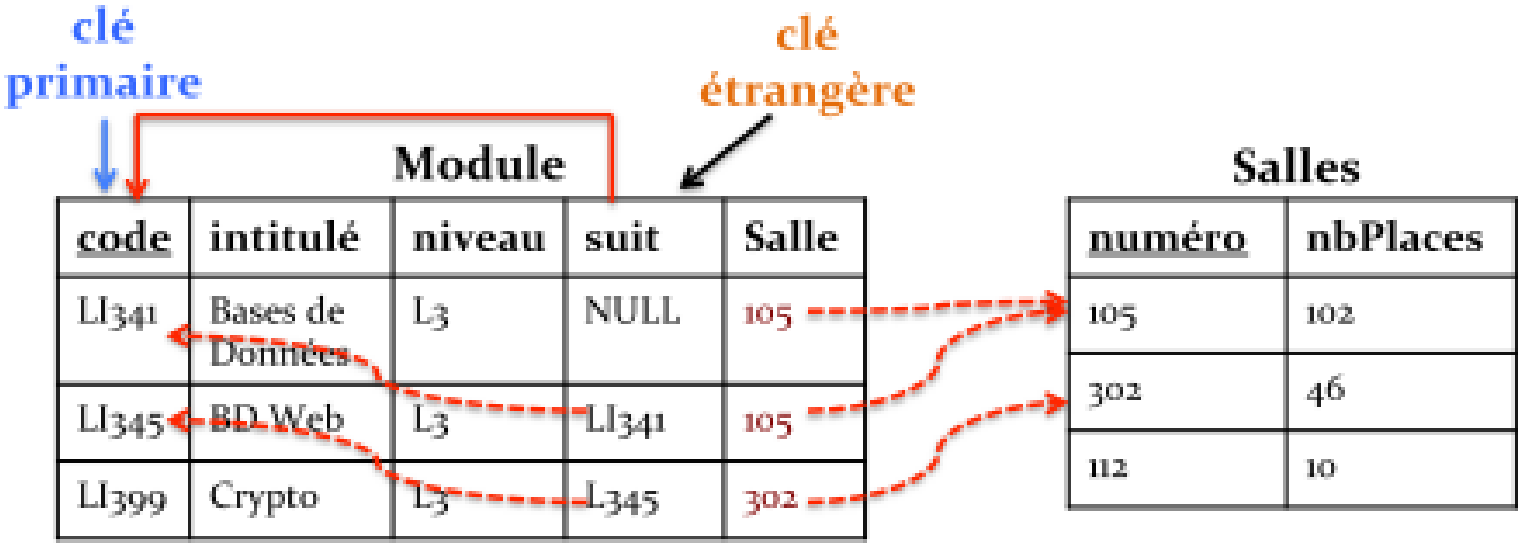
| numéro | nbPlaces |
|---|---|
| 105 | 102 |
| 302 | 46 |
| 112 | 10 |

# Clés étrangères : autre exemple

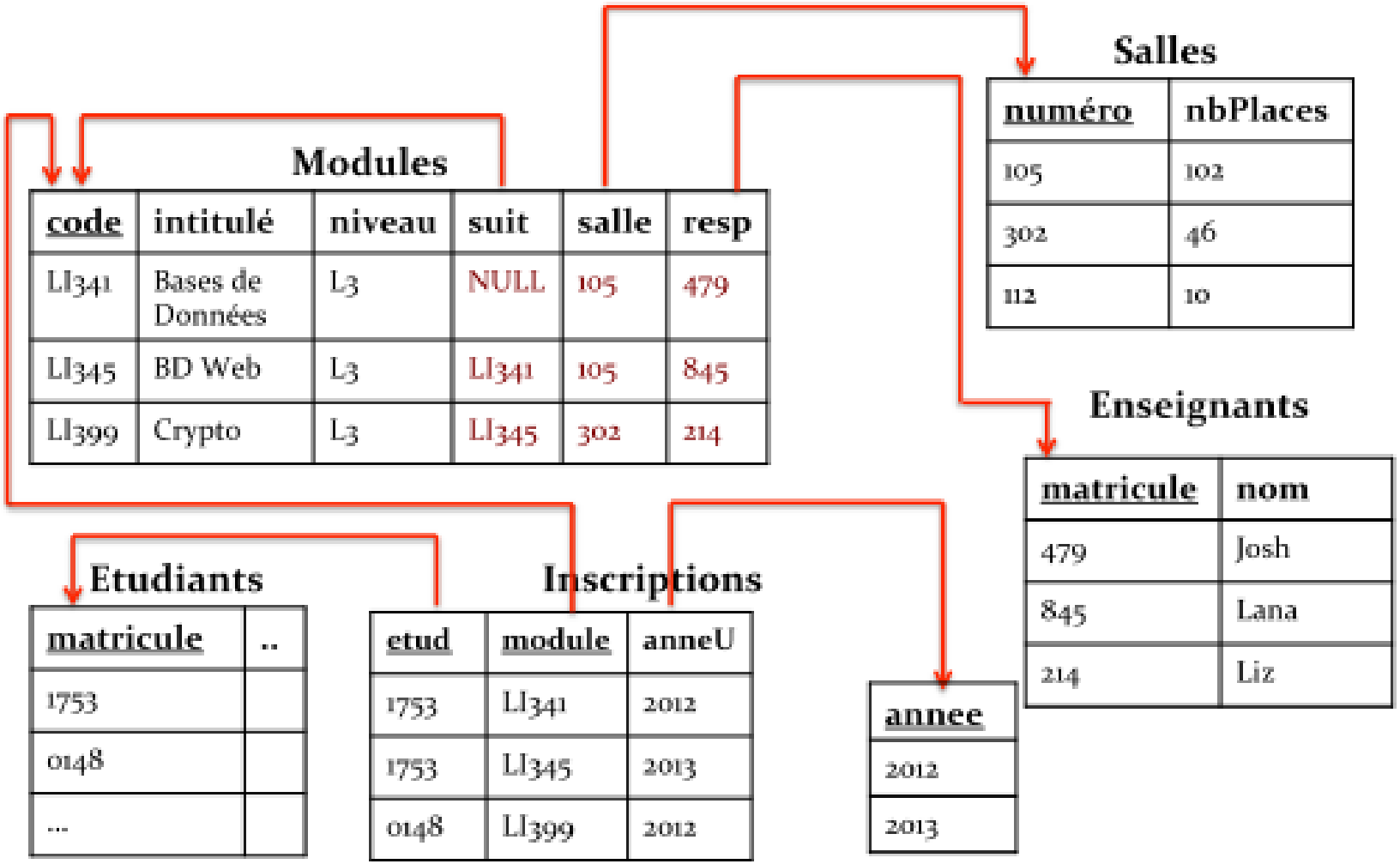
Une clé étrangère = sous-ensemble d'attributs dont les valeurs proviennent des clés primaires de la même table ou d'autre table.

Mécanisme de référencement des n-uplets.

**Matérialise les assocations de E/A.**



# Clés étrangères : exemple complet



**Modules**

| code | intitulé | niveau | suit | salle | resp |
|---|---|---|---|---|---|
| LI341 | Bases de Données | L3 | NULL | 105 | 479 |
| LI345 | BD Web | L3 | LI341 | 105 | 845 |
| LI399 | Crypto | L3 | LI345 | 302 | 214 |

**Salles**

| numéro | nbPlaces |
|---|---|
| 105 | 102 |
| 302 | 46 |
| 112 | 10 |

**Enseignants**

| matricule | nom |
|---|---|
| 479 | Josh |
| 845 | Lana |
| 214 | Liz |

**Etudiants**

| matricule | .. |
|---|---|
| 1753 | |
| 0148 | |
| ... | |

**Inscriptions**

| etud | module | anneU |
|---|---|---|
| 1753 | LI341 | 2012 |
| 1753 | LI345 | 2013 |
| 0148 | LI399 | 2012 |

| annee |
|---|
| 2012 |
| 2013 |

# Notations

Convention de notation

- Clé primaire : **soulignement**
- Clés étrangères : **dièse** ou astérisque et désignation de la table référencée (**#** ou ***)

**Etudiants**(matricule, nom, prenom, adresse, #collaborateur)

Collaborateur fait référence à (la clé primaire de) Etudiants

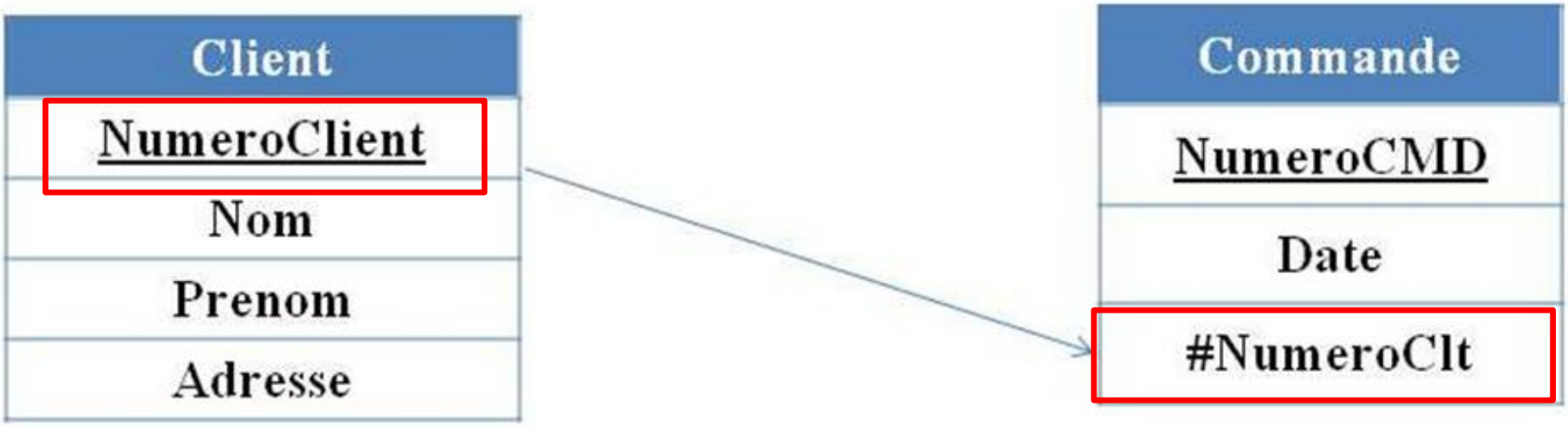
**Modules**(code, intitule, niveau, #salle)

Salle fait référence à Salles

**Salles**(numero, capacite, précédente*, suivante*)

Précédente et suivante font chacune référence à Salles

# Concepts et notations




Nous pouvons aussi adopter la notation textuelle suivante :

***Client(NumeroClient**, nom client, prenom, adresse)*

***Commande(NumeroCmd**, date, **#NumeroClt**)*

# Conception de schémas relationnels

- Une schéma relationnel peut contenir des centaines de tables avec des milliers d'attributs.

- Problème: comment éviter des erreurs de conception?

- **Solution :** Génération (automatique) à partir d'un **schéma E/A**

## Règles de passage du Modèle E/A au MR

**Notations**

On dit qu'une association binaire (entre deux entités ou réflexive) est de type :

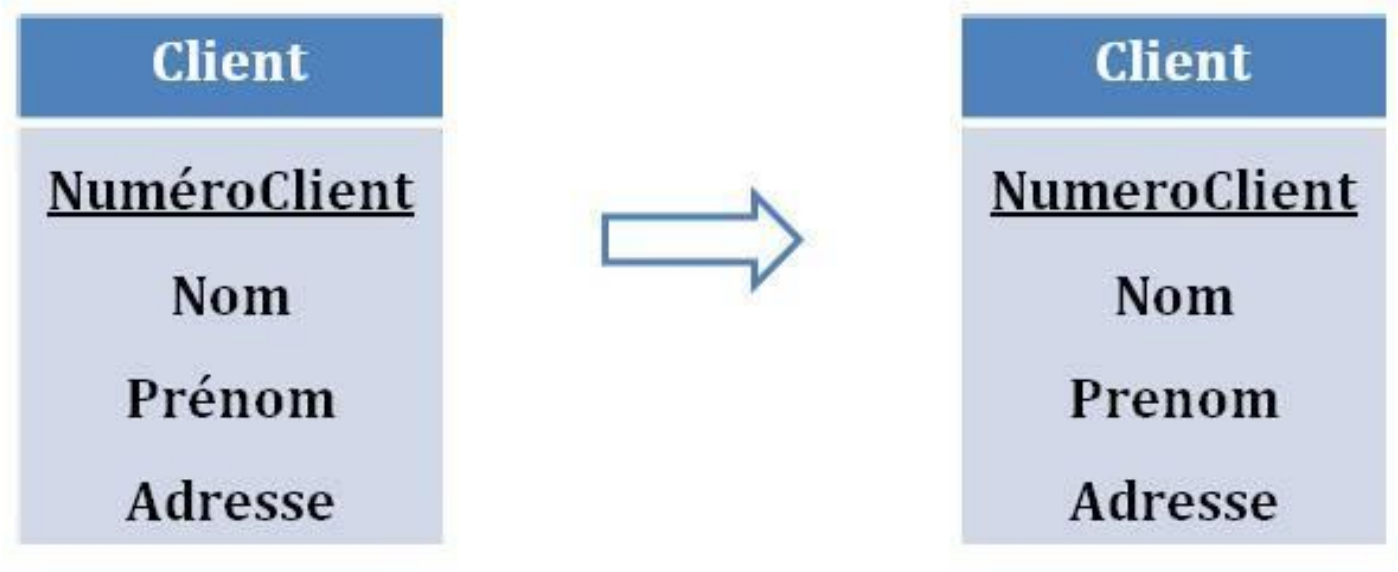
- **1 : 1** (**un à un**) si aucune des 2 **cardinalités maximales n'est n**
- **1 : n** (**un à plusieurs**) si **une** des 2 **cardinalités maximales est n**
- **n : m** (**plusieurs à plusieurs**) si les 2 **cardinalités maximales sont n**

# Règle 1 : Entités

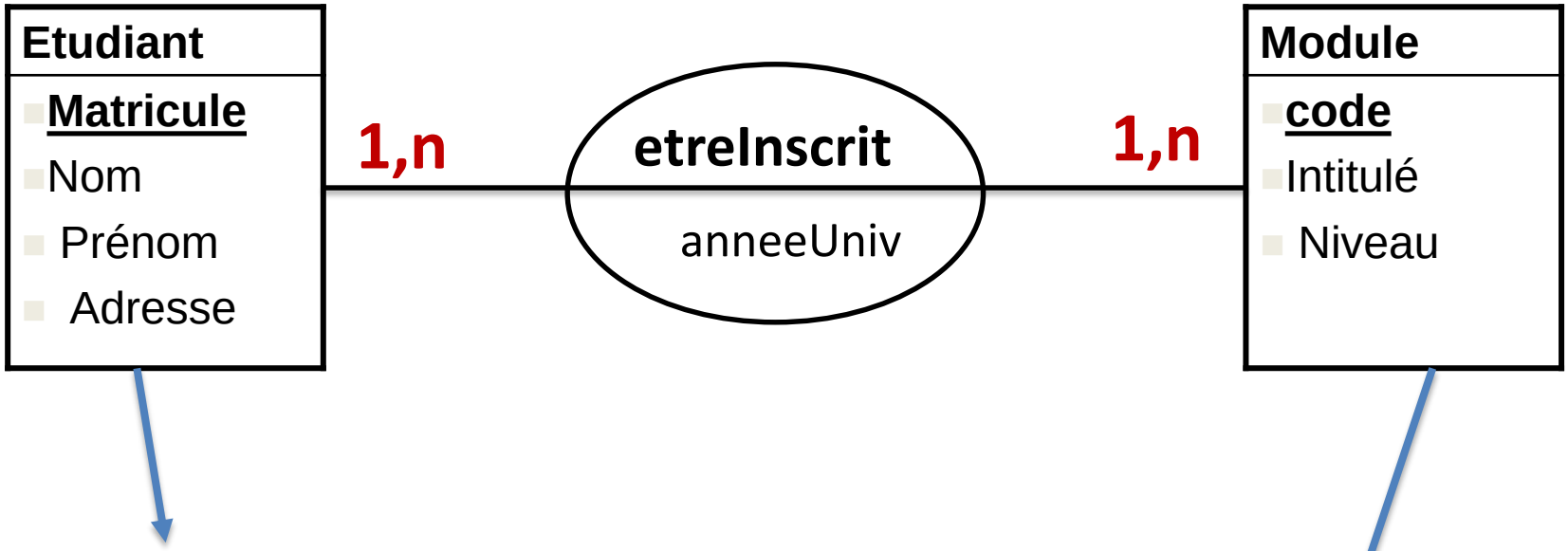
Créer pour *chaque entité* une ***table***

***Attributs entité*** *deviennent* ***attributs de la table***

***Identifiant entité*** *deviennent* ***clé primaire de la table***



# Exemple

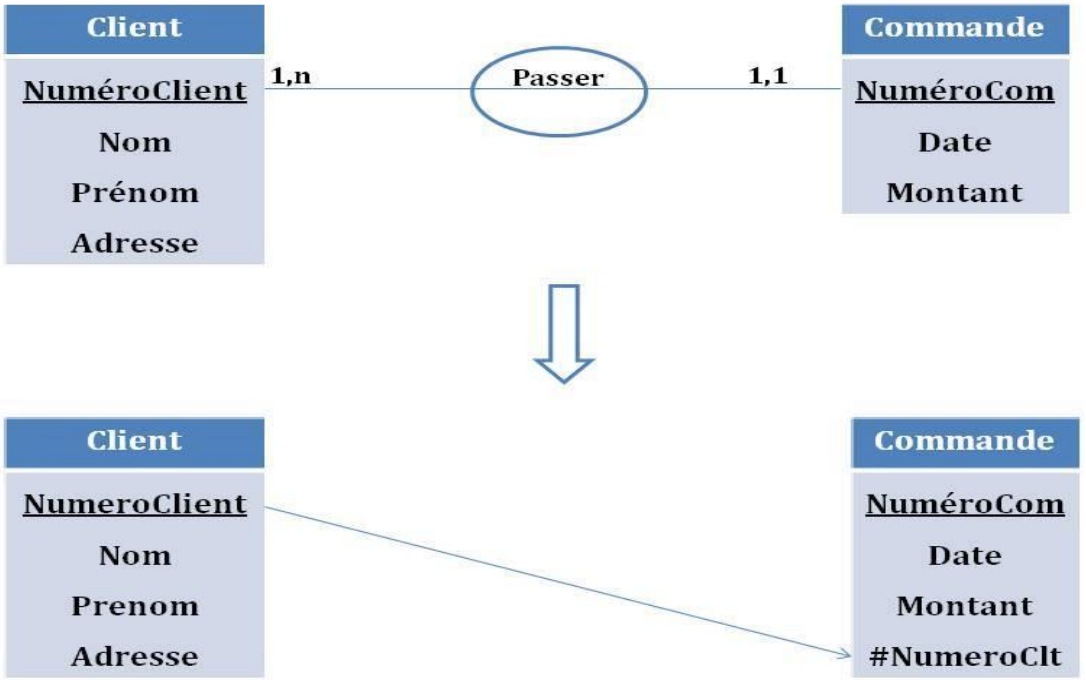



**Etudiant(**Matricule, Nom, Prenom, Adresse**)**

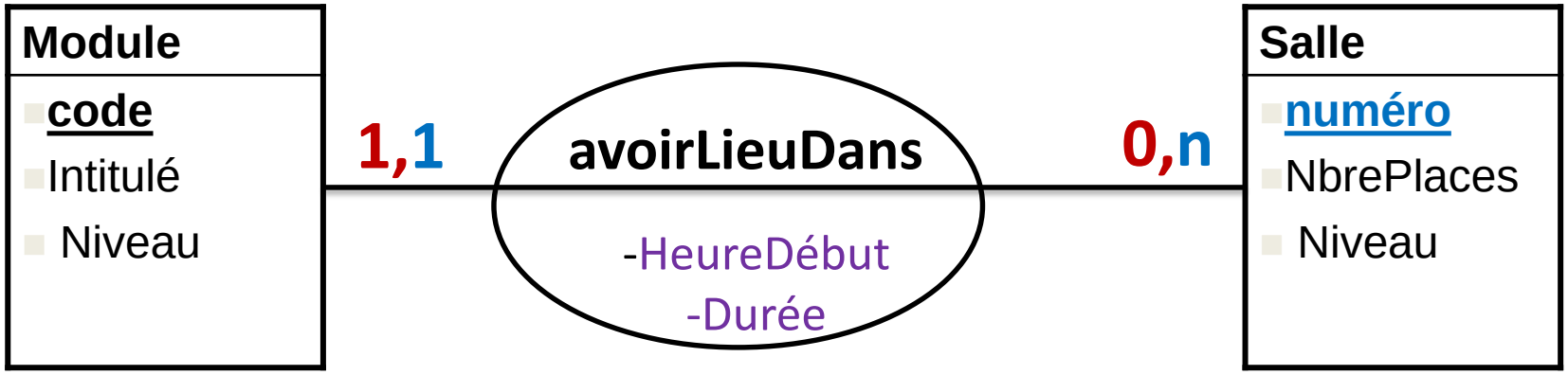
**Module(**Code, Intitule, Niveau**)**

# Règle 2 : Association de type un-à-plusieurs

- Une **association binaire** de type ***1 : n*** disparaît, au profit d'une clé étrangère **dans la table du coté *0,1* ou *1,1*** qui référence la clé primaire de l'autre table.
  - Cette clé étrangère ne peut **pas** recevoir la valeur **vide** si la cardinalité est ***1,1***.



# Exemple



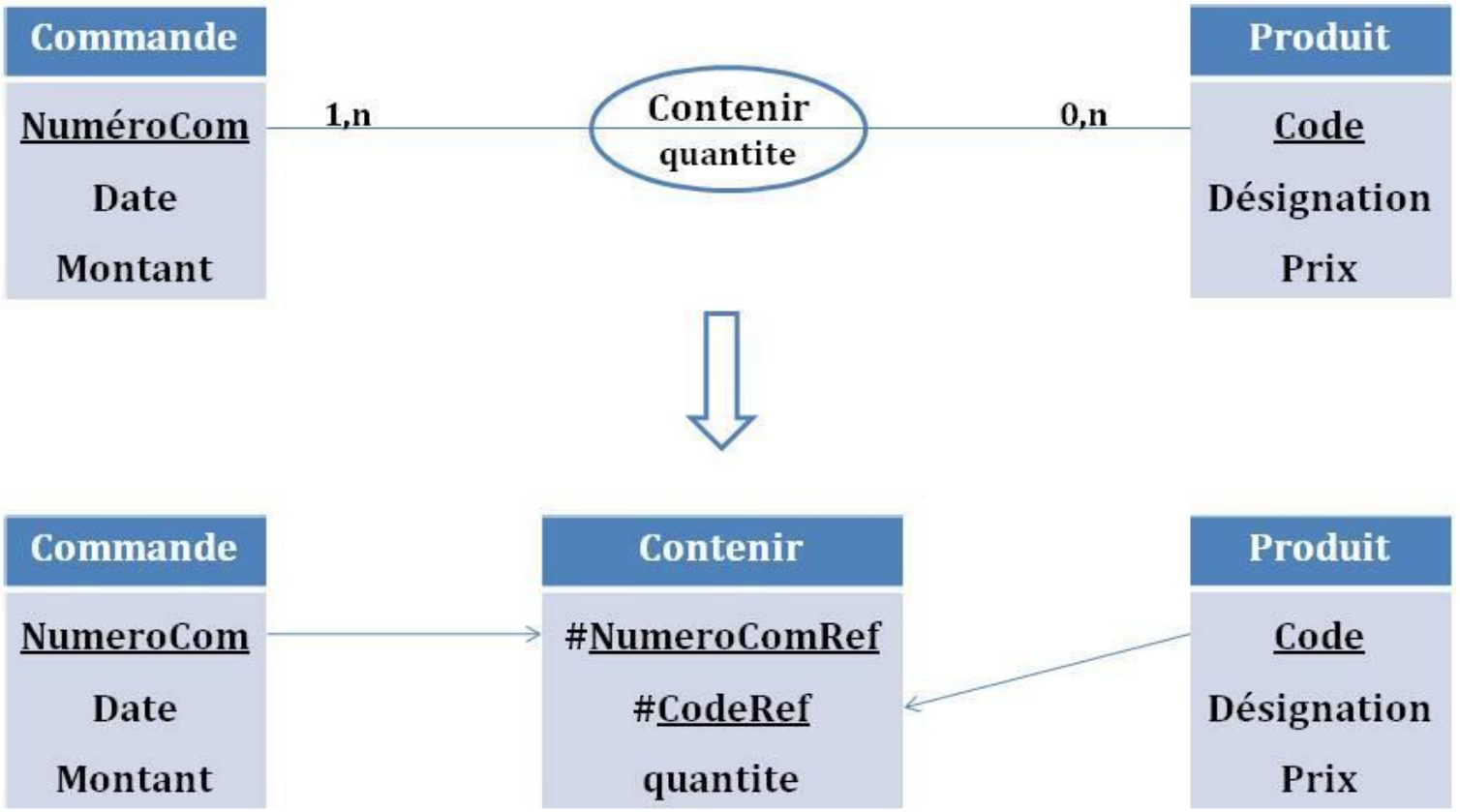

**Module(**Code, Intitule, Niveau,**#numeroSalle**, heureDebut, duree**)**

**NumeroSalle** référence **numero** de la table **Salle**

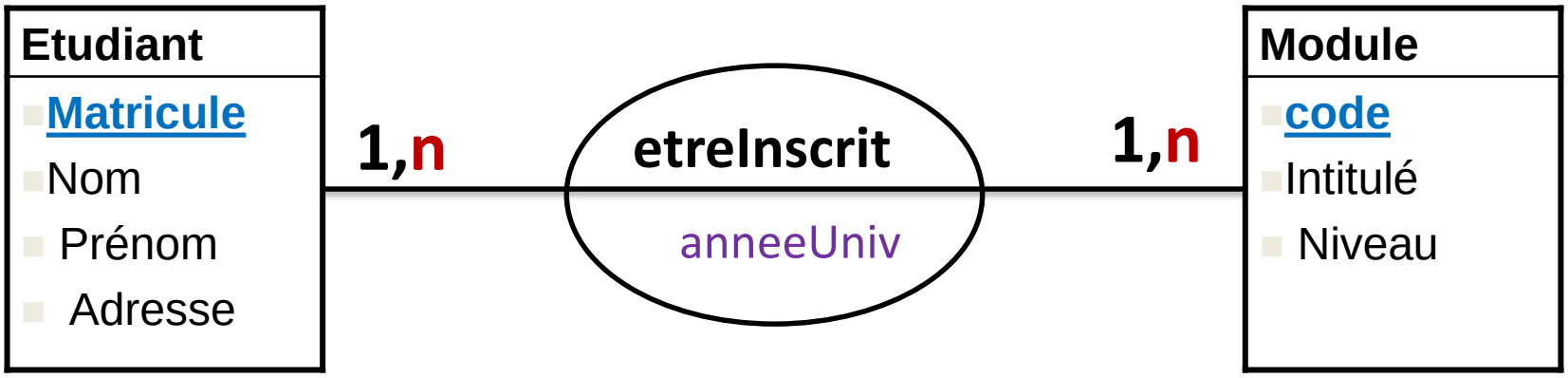
**Salle(**numero, NbrePlaces**)**

# Règle 3 : Association de type plusieurs-à-plusieurs

- Une association binaire de type ***n : m*** devient une **table supplémentaire** (table de jonction) dont la **clé primaire** est **composée** des **deux clés étrangères**.



# Exemple




**Etudiant(**matricule, nom, prenom, adresse**)**

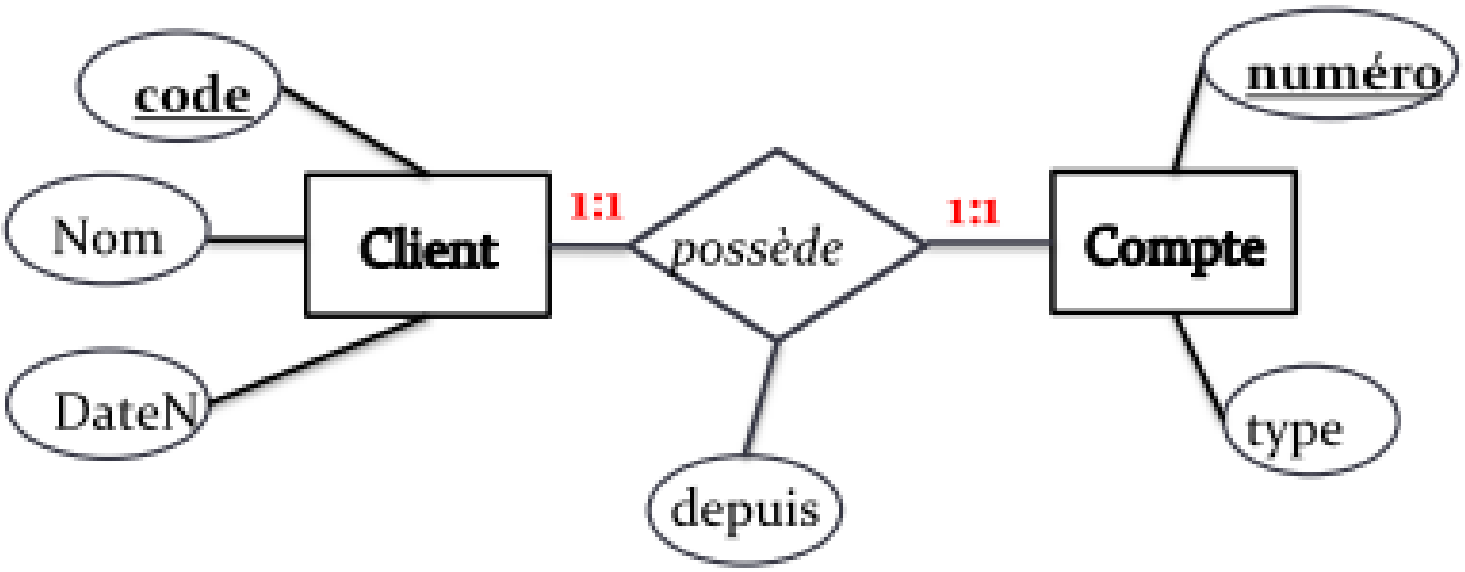
**Module(**code, intitule, niveau, adresse**)**

**Inscription(**matricule*, code*, anneeUniv**)**

# Règle 4 : Association de type un-à-un

- Une association binaire de type ***1 : 1*** est traduite comme une association binaire de type ***1 : n***
  - sauf que la clé étrangère se voit imposer une contrainte d'unicité

# Exemple



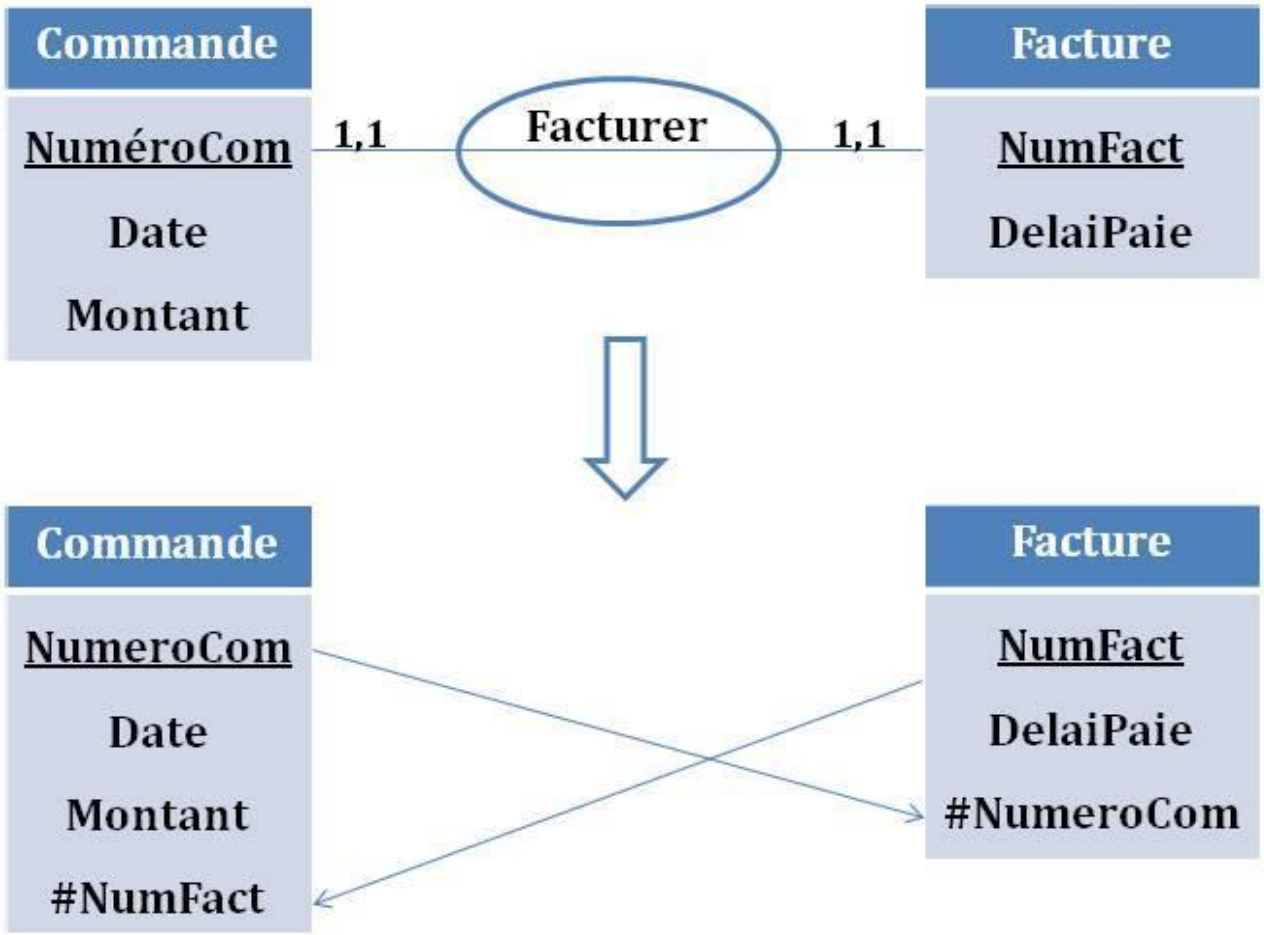

**Compte**(<u>numéro</u>, type)

**Client**(<u>code</u>, Nom, DateN, NumeroCompte*, dateOuverture)

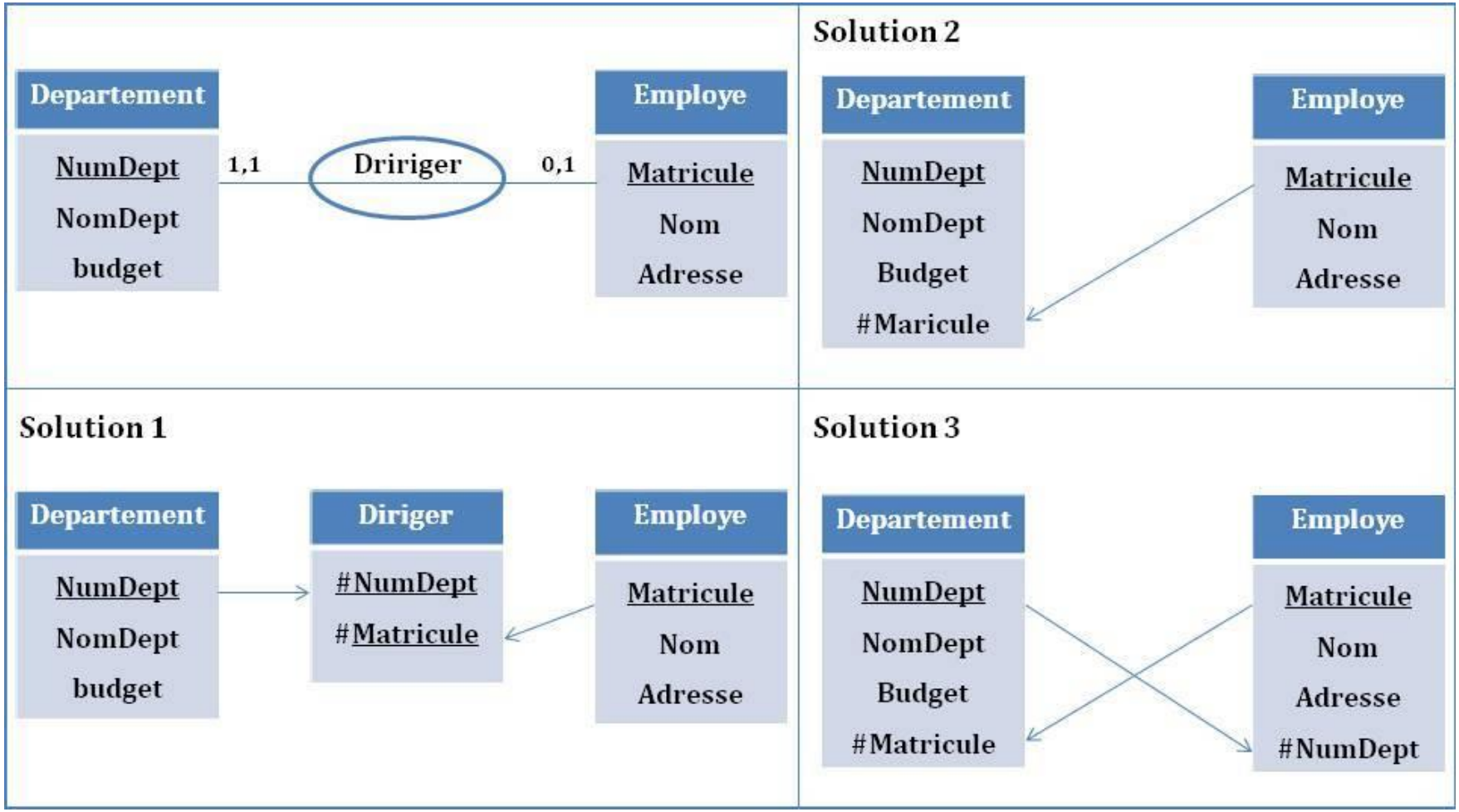
NumeroCompte référence numéro de la table Compte

*Si Compte n'est associé à aucune autre entité, mettre tout dans la relation Client*

# Exemple 1 : cardinalité 1,1 des deux cotés

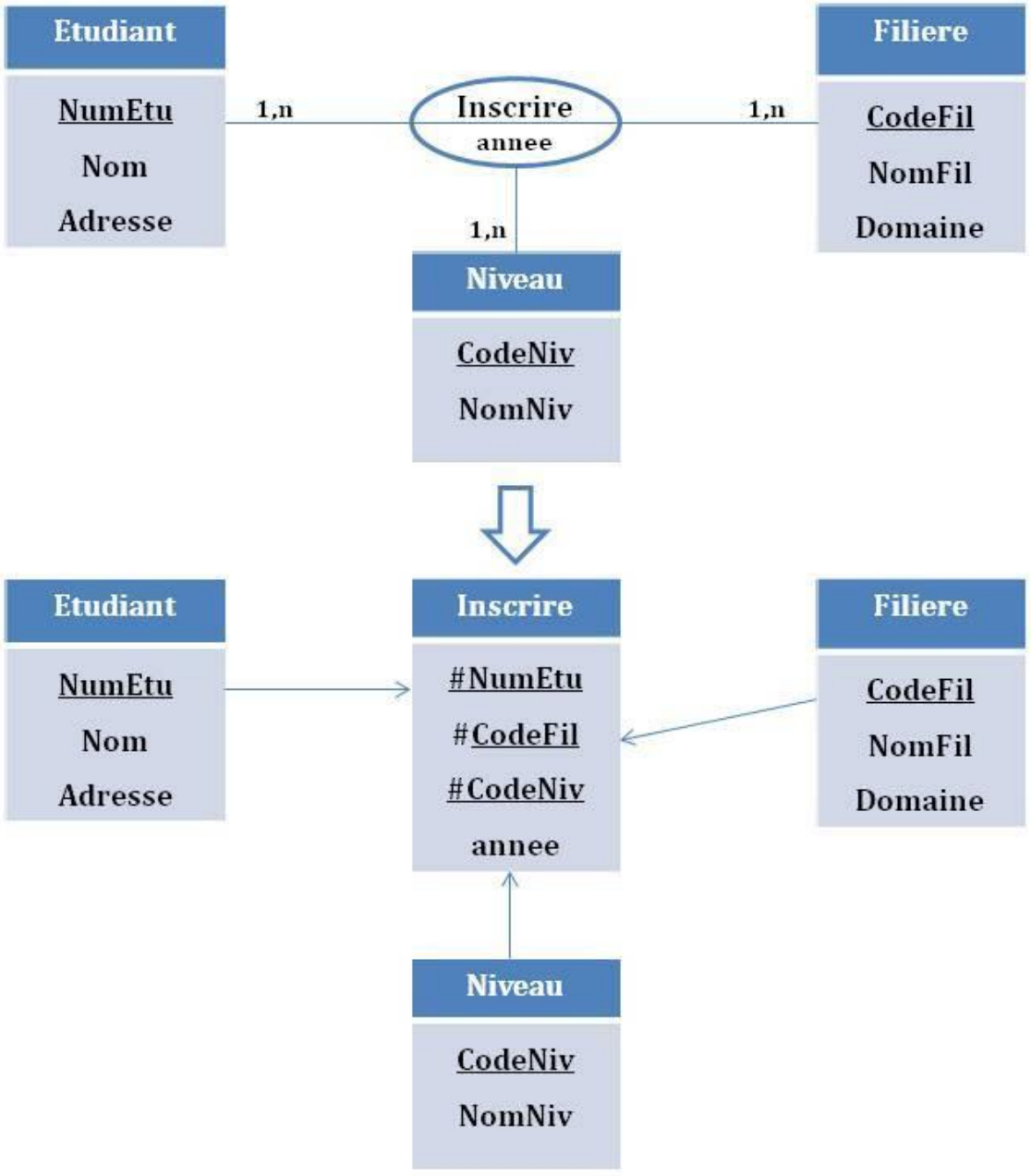


# Exemple 2 : cardinalité 0,1 d'un coté et 1,1 de l'autre



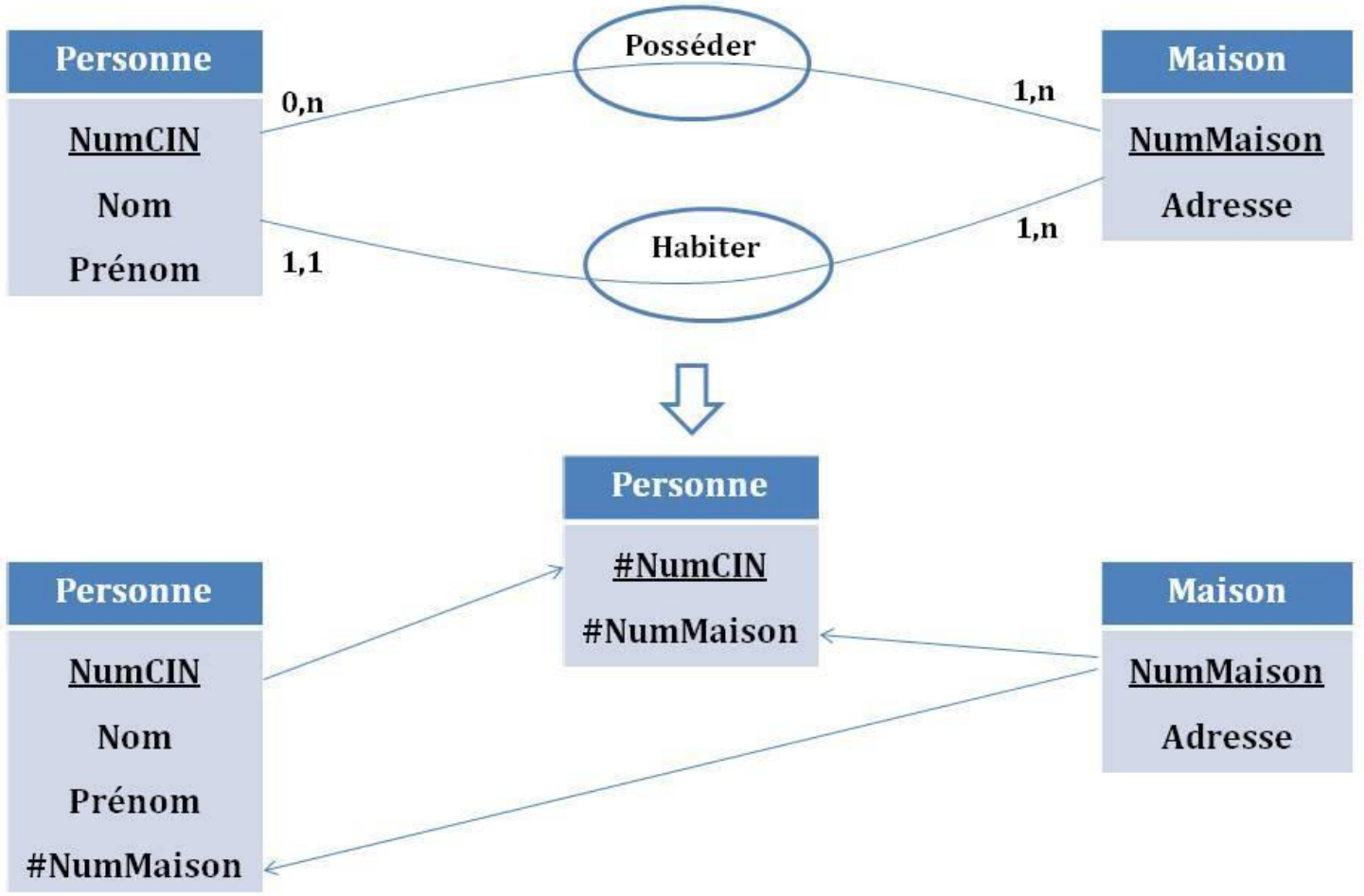
# Règle 5 : association n-aire n:m:k

- **Une association non binaire** est traduite par une **table supplémentaire** dont la clé primaire est composée d'autant de clés étrangères que d'entité en association.
- Les attributs de l'association deviennent les colonnes de cette nouvelle table.


Etudiant
NumEtu
Nom
Adresse
1,n
Inscrire
annee
1,n
Filiere
CodeFil
NomFil
Domaine
1,n
Niveau
CodeNiv
NomNiv
Etudiant
NumEtu
Nom
Adresse
Inscrire
#NumEtu
#CodeFil
#CodeNiv
annee
Filiere
CodeFil
NomFil
Domaine
Niveau
CodeNiv
NomNiv

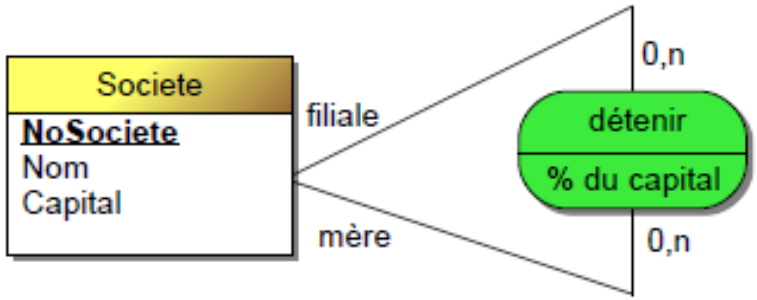
# Règle 6 : plusieurs associations entre les mêmes entités

- Les règles générales s'appliquent

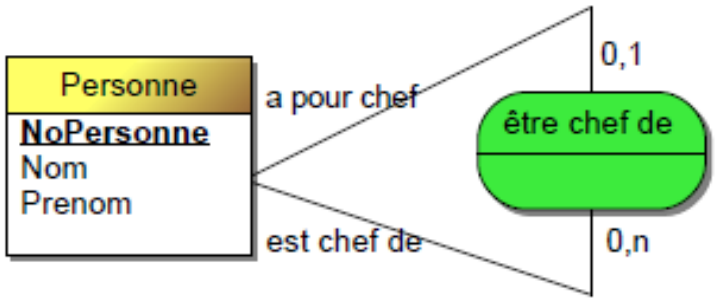


# Règle 7 : Relations réflexives

- On applique les règles générales avec la seule différence que la **relation est reliée deux fois à la même entité**.



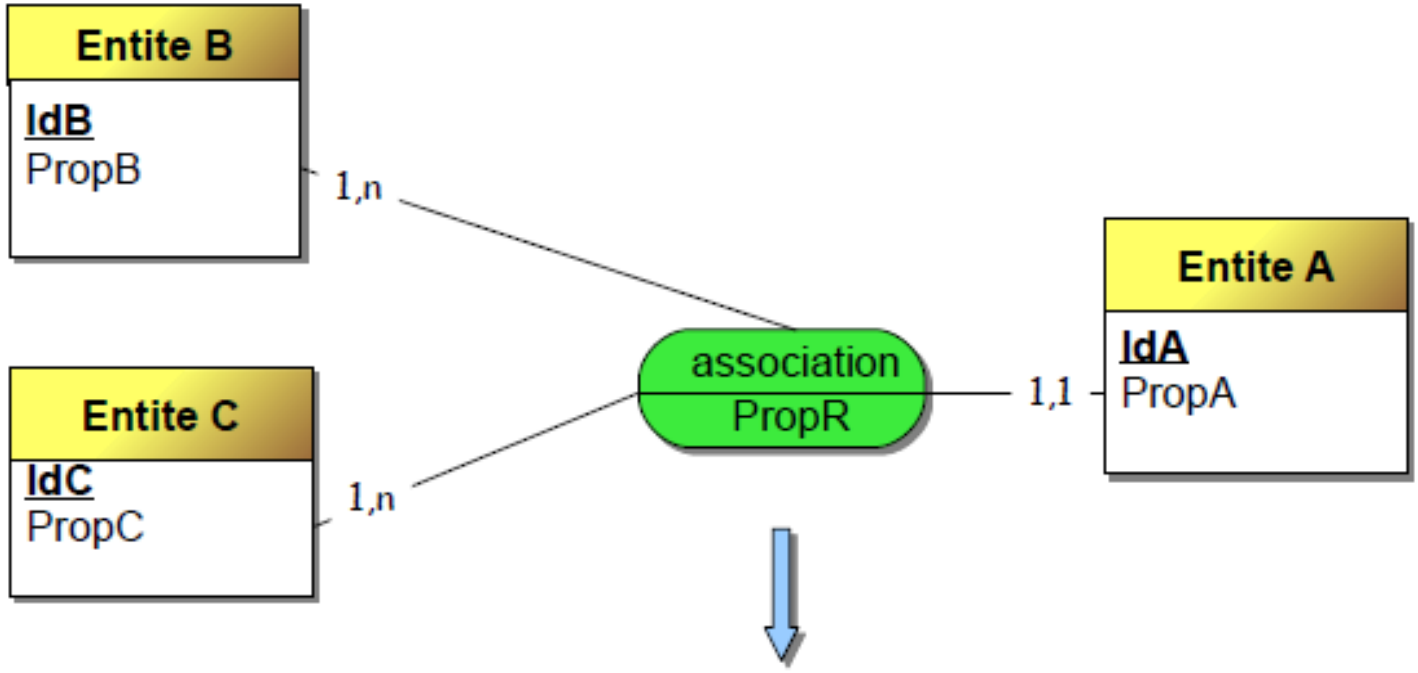

Societe(**NoSociete**,Nom,Capital)
Detenir(**NoSociete**,**NoSocieteFiliale**,Pourcent)




Personne(**NoPersonne**,Nom,Prenom,NoChef)

# Règle 8 : association n-aire avec 1:n:m..k

- Si une **association n-aire** possède une branche avec une cardinalité **1 :1**, on place les références dans la table reliée par 1 :1.

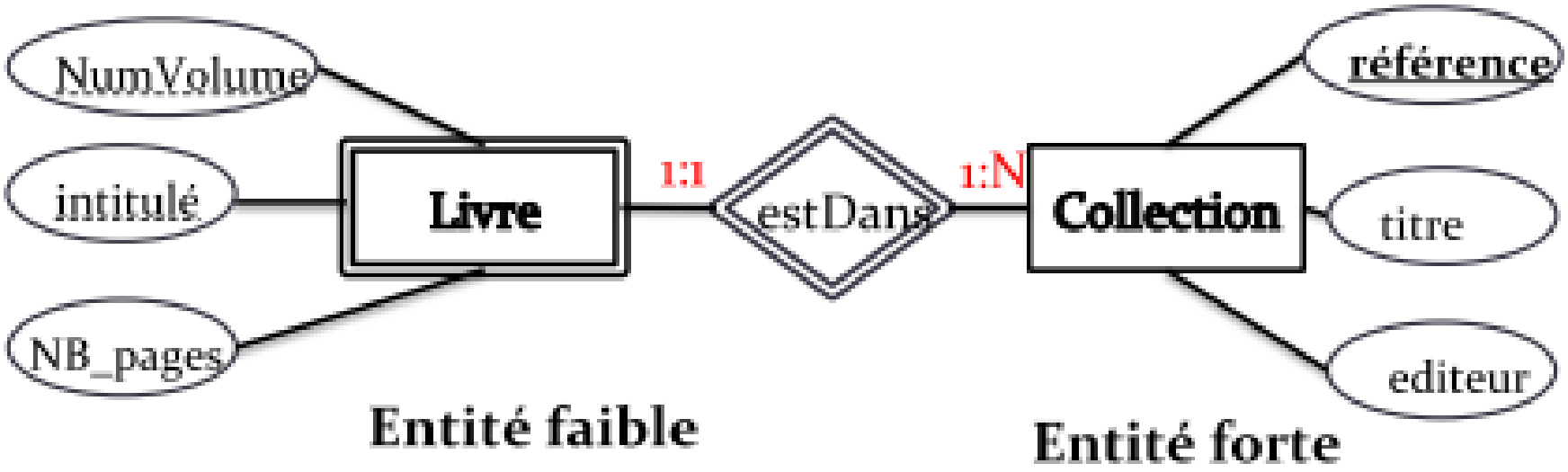


# **Règle 9 :** Traduction des **entités faibles**

- Créer une **table** pour **l'entité faible**
- Clé= **identifiant de l'entité forte** + attributs discriminants de l'entité faible
  - Identifiant entité forte est une **clé étrangère** et (*) fait partie de la clé

Entité faible se traite comme une association 1:1 avec (*) en plus

# Exemple

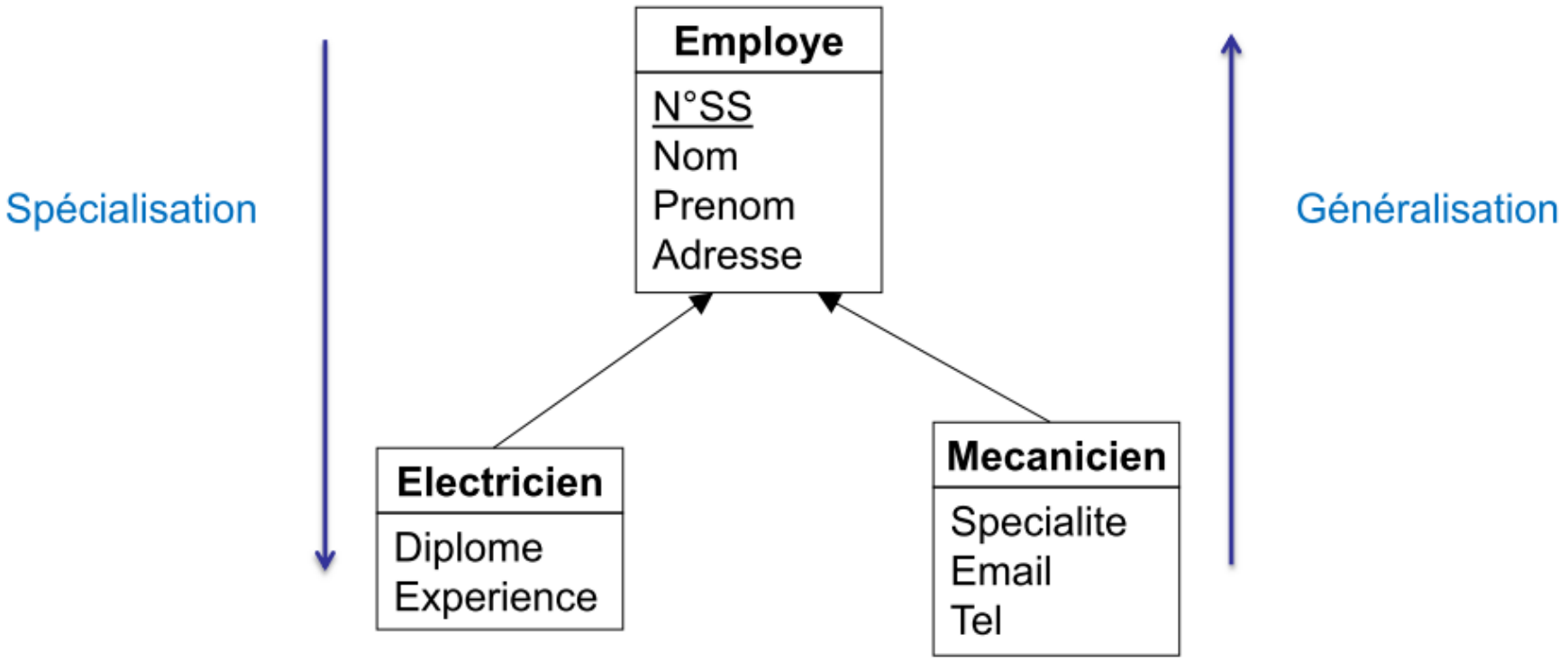



Livre(référenceLivre*, numVolume, intitulé, NB_pages)

# Règle 10 : Traduction de **la spécialisation**

- **Créer une table par sous-entités**
  - Attributs des sous-entités →attributs des tables
  - **Identifiant de la super-entité** devient clé primaire des sous-entités

# Exemple




**Employe(NumeroSS, Nom, prenom, Adresse)**

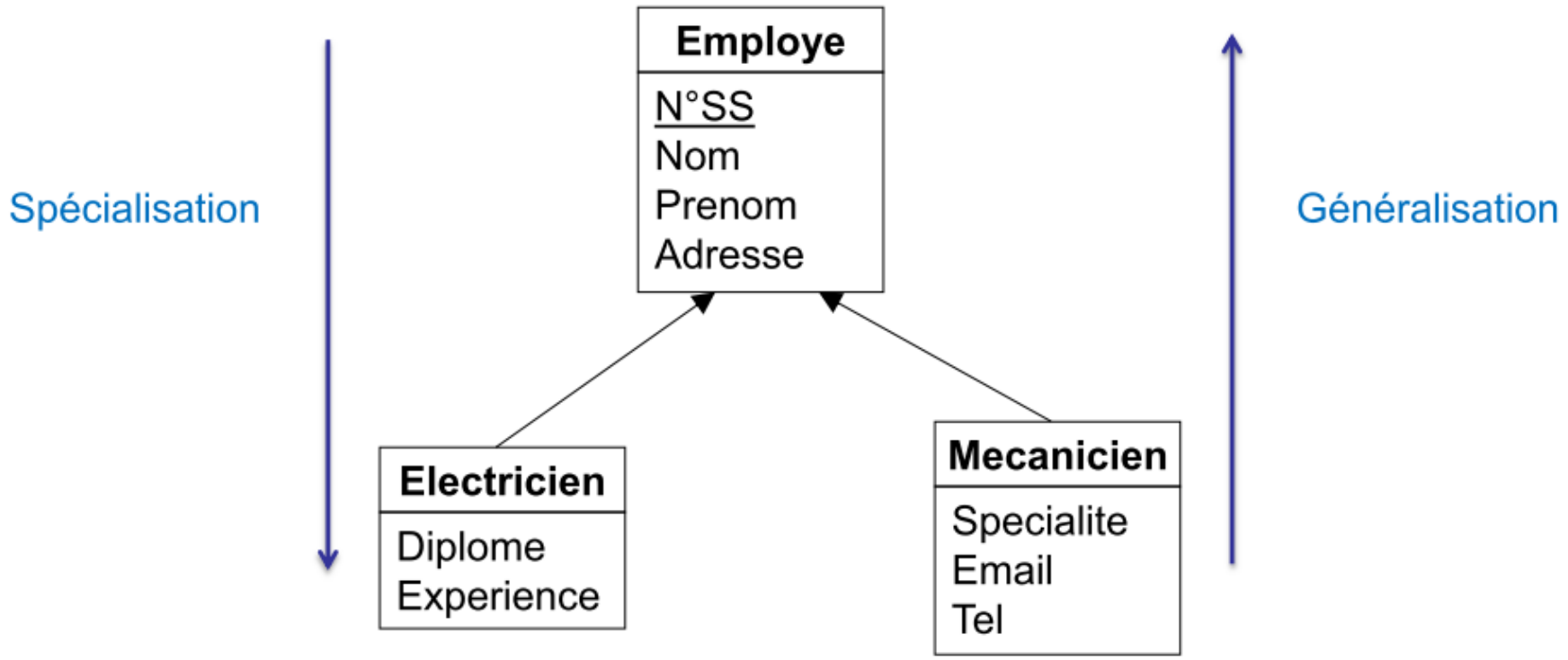
**Electricien(#NumeroSS, Diplome, Experience)**

**Mecanicien(#NumeroSS, Specialite, Email, Tel)**

# Traduction de la spécialisation

- Créer une table par sous-entités
  - Attributs des  sous-entités →attributs des tables
  - Identifiant de la super-entité devient clé primaire des sous-entités
  - **Cas particulier :** si la super entité est abstraite (pas d'instance), on peut supprimer la table qui la représente mais il faut rajouter ses attributs aux sous-entités

# Exemple




~~**Employe(NumeroSS, Nom, prenom, Adresse)**~~

**Electricien(NumeroSS, Nom, prenom, Adresse Diplome, Experience)**

**Mecanicien(NumeroSS, Nom, prenom, Adresse Specialite, Email, Tel)**

# Normalisation du modèle relationnel

- Le modèle relationnel procure des outils destinés à **tester la qualité et la cohérence des relations** dans un schéma relationnel.
- Cette étape, **appelée normalisation**, permettra de vérifier certaines propriétés des relations et le cas échéant de les transformer.
- La normalisation d'un schéma relationnel **suggère une autre méthode pour obtenir un ensemble de relations**.
- L'objectif de la normalisation est de construire un **schéma de base de données cohérent**.
  - Un **mauvais schéma relationnel peut conduire à un certain nombre d'anomalies** pendant la phase d'exploitation de la base de donnée.
- Pour qu'un modèle relationnel soit normalisé, il faut qu'il respecte certaines **contraintes appelées les formes normales**.
- Les **formes normales** s'appuient sur les **dépendances fonctionnelles** entre attributs.

# DF : Définition

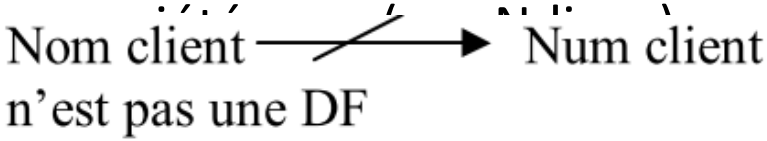
- On dit que **b** est en dépendance fonctionnelle (DF) de **a**
- si à **une valeur quelconque** de l'attribut **a**, on **ne peut faire correspondre qu'une seule valeur au plus** de l'attribut **b**.

On note **a → b**

- Autrement dit, **si on connaît la valeur de a**, **on peut en déduire une seule valeur de b**.
- **Mais la réciproque n'est pas vrai** (si on connaît b, on ne peut pas en déduire a).

**Num_client → Nom_client**

  - Il existe une DF entre num_client et Nom_client, car si on connaît une valeur de la propriété num_client (ex : 4553), il ne peut lui correspondre qu'une seule valeur de la




- **La réciproque est fausse :**
  - Si l'on connaît la valeur de la propriété Nom_client, on ne peut pas en déduire la propriété Num_client, car il peut y avoir des homonymes.

# DF : Terminologie

- Si on a une dépendance fonctionnelle **a→ b**, on peut employer les expressions suivantes de façon équivalente :
  - Il y a une **dépendance fonctionnelle de a vers b**
  - **b** est en **dépendance fonctionnelle** de **a**
  - **b dépend fonctionnellement** de **a**
  - **a** est la **source** et **b** est le **but** (ou la cible) de la **dépendance fonctionnelle**

**DF à partir de propriétés concaténées (partie gauche composée de plusieurs attributs)**

- Il peut exister des dépendances fonctionnelles **à partir de propriétés concaténées**, c'est-à-dire qui **forment un tout indissociable**, comme si elles étaient soudées.

**Exemple** : Considérons une commande qui comporte plusieurs produits

- **Num_Commande + Ref_Produit → quantité commandée**

**Propriétés des DF**

- **Union:**
  - Si on a deux DF ayant la même source, on peut les rassembler en une seule, en séparant les cibles par une virgule.
  - Si **a → b** et **a → c** alors ont peut écrire que **a → b,c**
- **Transitivité :**
  - Si **a → b** et **b → c** alors on a : **a → c**

# Première forme normale

- **s'intéresse au contenu des champs**.
- Interdit la présence de **plusieurs valeurs dans un même champ** d'une relation.
  - En effet, la notion de dépendance fonctionnelle entre les champs ne peut plus être vérifiée s'ils **possèdent plusieurs valeurs**.
- Elle s'exprime de la manière suivante :

**Tout champ contient une valeur atomique.**

# Comment passer en première forme normale ?

- La relation suivante n'est pas en première forme normale ; le champ 'Auteurs' contient plusieurs valeur

Pub

| NumPubli | Titre | Auteurs |
|---|---|---|
| 13490 | Le vin et l'avenir | Jean Lasso, Hubert De la Tuque, Stanislas Wilski |
| 21322 | Bière et progrès social | Aristide Salem, Jean Lasso, Salome Dupont |
| 45333 | Le champagne et la France | Penelope Light, Vanessa Martinez, Salome Dupont |

# Comment passer en première forme normale ?

- Une solution pour résoudre ce problème **est de décomposer** le champ '**Auteurs**' en 'Auteur_1', 'Auteur_2', 'Auteur_3', 'Auteur_4', **etc**.
- Ainsi, la relation est bien en première forme normale.
- L'ennui est que l'on ne sait pas à l'avance le nombre d'auteurs que peut posséder une publication, et le problème consiste ***donc à savoir combien de champs ajouter***.
- La solution plus correcte, mais également plus lourde à mettre en œuvre, est de **décomposer cette relation en trois relations** :

  **Publication**(NumPubli, Titre),

  **Auteur**(NumAuteur, Nom, Prenom)

  **EstEcrite**(#NumPubli, #NumAuteur)
- Ces trois relations sont la représentation de la réalité « **une publication est écrite par des auteurs** ».
- Elle se modélise par deux entités '**Publication**' et '**Auteur**' reliées par l'association '**EstEcrite**'.

- On obtient alors le résultat suivant

**Auteur(NumAuteur, Nom, Prenom)**

| NumAuteur | Nom | Prenom |
|---|---|---|
| 1 | Lasso | Jean |
| 2 | De la Tuque | Hubert |
| 3 | Wilski | Stanislas |
| 4 | Salem | Aristide |
| 5 | Dupont | Salome |
| 6 | Light | Penelope |
| 7 | Martinez | Vanessa |

**EstEcrite(#NumPubli, #NumAuteur)**

| NumPubli | NumAuteur |
|---|---|
| 13490 | 1 |
| 13490 | 2 |
| 13490 | 3 |
| 21322 | 4 |
| 21322 | 1 |
| 21322 | 5 |
| 45333 | 6 |
| 45333 | 7 |
| 45333 | 5 |

**Publication(NumPubli, Titre)**

| NumPubli | Titre |
|---|---|
| 13490 | *Le vin et l'avenir* |
| 21322 | Bière et progrès social |
| 45333 | Le champagne et la France |

- On doit alors effectuer des **jointures** sur les différentes relations afin de reconstituer l'information dans son intégralité.
- Cette décomposition en trois relations **se fait sans perte d'information**.

# Deuxième forme normale

- Bien évidemment, une relation **doit déjà se trouver en première forme normale** pour être en **deuxième forme normale**.
- Cette dernière **recherche la redondance d'information** dans une relation.
- Elle **interdit les dépendances fonctionnelles** possibles **entre les champs** qui composent la **clé** et les **autres champs**.
- On peut l'exprimer de la manière suivante :

  **La relation est en première forme normale.**

  **Tout champ qui n'appartient pas à la clé ne dépend pas d'une partie de la clé.**

# Comment passer en deuxième forme normale ?

- La solution consiste à **décomposer la relation en deux relations**.
- La **nouvelle relation** créée a pour **clé** la **partie de la clé dont dépendent** les autres champs qui **constituent ses champs.**

**Produit(Article, Fournisseur, Adresse, Prix)**

| Article | Fournisseur | Adresse | Prix |
|---|---|---|---|
| Marteau | SOGENO | Paris | 5 |
| Tournevis | ARTIFACT | Lille | 10 |
| Tournevis | SOGENO | Paris | 23 |
| Pince | LEMEL | Paris | 34 |
| Mètre | ARTIFACT | Lille | 24 |

- La clé est constituée des champs '**Article**' et '**Fournisseur**'.
- Or, il y a une relation **de dépendance entre** le champ '**Fournisseur**', qui est une partie de la clé, et le champ '**Adresse**'.
  - On **décompose** alors la relation pour **éliminer la redondance** ainsi créée.
  - La nouvelle relation aura pour clé la partie de la clé de la relation d'origine dont dépendent fonctionnellement les autres champs. Il s'agit du champ 'Fournisseur'.
  - Les autres champs dépendants constituent le reste de la relation. Il s'agit ici du champ '**Adresse**'.

- On obtient alors le résultat suivant

**Fournisseur(Fournisseur, Adresse)**

| Fournisseur | Adresse |
|---|---|
| SOGENO | Paris |
| ARTIFACT | Lille |
| LEMEL | Paris |

**Produit(Article, #Fournisseur, Prix)**

| Article | Fournisseur | Prix |
|---|---|---|
| Marteau | SOGENO | 5 |
| Tournevis | ARTIFACT | 10 |
| Tournevis | SOGENO | 23 |
| Pince | LEMEL | 34 |
| Mètre | ARTIFACT | 24 |

- Comme précédemment, il est nécessaire de faire une **jointure** pour reconstituer l'information.
- La décomposition en deux relations **se fait sans perte d'information**.

## Remarque

- Si la clé d'une relation est **atomique**, c'est-à-dire **composée d'un seul champ**, elle est **naturellement** en **deuxième forme normale**.

# Troisième forme normale

- La troisième forme normale **recherche également la redondance d'information dans une relation**.
- On cherche s'il existe une **dépendance entre deux champs qui ne font pas partie d'une clé**.
- Si c'est le cas, on se trouve dans la situation où **un champ dépend d'un autre** champ **qui dépend lui même d'une clé**.
- La clé considérée peut être primaire ou secondaire.
- La troisième forme **normale interdit donc les dépendances fonctionnelles dites « transitives »** entre les champs.
- Elle s'exprime de la manière suivante :

  **La relation est en deuxième forme normale (donc en première forme normale).**

  **Tout champ n'appartenant pas à une clé ne dépend pas d'un autre champ non clé.**

# Comment passer en troisième forme normale ?

- La solution est également de **décomposer** la relation de **départ en deux relations**.
- La **nouvelle relation** créée a **pour clé le champ dont dépendent** les autres champs qui constituent ainsi la dépendance transitive.

**Baladeur(NumBal, Marque, Type, Couleur)**

| NumBal | Marque | Type | Couleur |
|---|---|---|---|
| 12 | Apple | Ipod | Blanc |
| 43 | Creative | Zen | Noir |
| 23 | Apple | Ipod | Noir |
| 29 | Creative | Zen | Gris |
| 34 | Sony | MZ-RH910 | Rouge |

# Comment passer en troisième forme normale ?

**Baladeur(NumBal, Marque, Type, Couleur)**

| NumBal | Marque | Type | Couleur |
|---|---|---|---|
| 12 | Apple | Ipod | Blanc |
| 43 | Creative | Zen | Noir |
| 23 | Apple | Ipod | Noir |
| 29 | Creative | Zen | Gris |
| 34 | Sony | MZ-RH910 | Rouge |

- La clé de cette relation est un numéro, '**NumBal**', car il peut y avoir dans notre stock **plusieurs baladeurs de même marque**, de même **type** et de même **couleur**.
- Les marques déposent **les noms des objets** qu'elles fabriquent de façon à les **identifier sur le marché**.
  - Il existe donc une dépendance fonctionnelle entre le champ '**Type**' (qui n'appartient pas à la clé) et le champ '**Marque**' (qui n'appartient pas à la clé).
- On **décompose** la relation **en créant une nouvelle** qui **a pour clé le champ** dont dépendent les autres champs constituant la dépendance transitive.
  - Il s'agit dans ce cas du champ '**Type**'.
  - Les autres champs de la nouvelle relation sont composés des champs qui en dépendent fonctionnellement : ici, le champ '**Marque**

**Baladeur(NumBal, Marque, Type, Couleur)**

| NumBal | Marque | Type | Couleur |
|---|---|---|---|
| 12 | Apple | Ipod | Blanc |
| 43 | Creative | Zen | Noir |
| 23 | Apple | Ipod | Noir |
| 29 | Creative | Zen | Gris |
| 34 | Sony | MZ-RH910 | Rouge |

**Baladeur(NumBal, #Type, Couleur)**

| NumBal | Type | Couleur |
|---|---|---|
| 12 | Ipod | Blanc |
| 43 | Zen | Noir |
| 23 | Ipod | Noir |
| 29 | Zen | Gris |
| 34 | MZ-RH910 | Rouge |

**Baladeur_type(Type, Marque)**

| Type | Marque |
|---|---|
| Ipod | Apple |
| MZ-RH910 | Sony |
| Zen | Creative |

- Comme précédemment, il est nécessaire **de faire une jointure pour reconstituer** l'information dans son intégralité.
- La décomposition en deux relations se fait sans perte d'information.

# Conclusion

- Présentation du modèle relationnel
  - Tables  de n-uplets
  - Clés et contraintes d'intégrité
- Traduction d'un schéma EA vers un schéma relationnel
- Schéma EA à partir d'un schéma relationnel
- Normalisation d'un schéma relationnel