Modèle relationnel

Attribut, degré, n-uplet et cardinalité

Attribut : désigne une colonne d'une relation caractérisée par un nom prenant ses valeurs dans un domaine.

Exemple : La relation AVION a trois attributs : *numAvion*, *nomAvion*, et *localisation*.

Le *degré* (ou *arité*) d'une relation est son nombre d'attributs.

Un *n-uplet* ou *tuple* désigne une ligne d'une relation correspondant à un enregistrement.

La *cardinalité* d'une relation est son nombre de n-uplets.

Attribut, degré, n-uplet et cardinalité

Deux n-uplets d'une même relation doivent se distinguer selon au moins un attribut (pas de doublons).

On utilise la valeur spéciale « NULL » pour un attribut d'un n-uplet pour spécifier que sa valeur est inconnue (l'équivalent à une case vide d'une table).

L'ordre des n-uplets et des attributs est quelconque.

Spécification du schéma de relation

Le *schéma d'une relation* R est spécifié comme suit : < R (U), F > où :

- U : appelé *univers* de la relation R , désigne la liste des attributs et de leurs domaines respectifs,

- F : la liste des contraintes d'intégrité généralement réduites aux *dépendances fonctionnelles* entre les attributs.

Spécification du schéma de relation

Pour simplifier, on peut faire référence au schéma d'une relation en omettant la liste des domaines, celle des contraintes d'intégrité ou même celle des attributs.
$< R (A_1 ... A_n), F >$ ou $R (A_1 ... A_n)$ ou $R (U)$ ou R

Exemple : pour les deux relations AVION et TYPEAVION nous avons les schémas

AVION (numAvion, nomAvion, localisation)
TYPEAVION (nomAvion, capacite)

Nous nous restreindrons également à des domaines à valeur atomique, c'est-à-dire que nous interdirons les attributs à valeur multiple (du type liste ou ensemble de valeurs).

Base de données relationnelle

On définit une base de données relationnelle comme un ensemble de relations.

Le schéma d'une base de données est l'ensemble des schémas des relations la composant.

Clé d'une relation

Une clé d'une relation est un ensemble minimal d'attributs dont la valeur détermine un unique n-uplet de l'extension de la relation.

Une relation doit toujours avoir au moins une clé.

Une relation peut posséder plusieurs clés. Dans ce cas, on choisit, parmi les clés candidates, une clé appelée clé primaire.

L'usage est de souligner la partie clé primaire d'un schéma de relation.

Exemple :

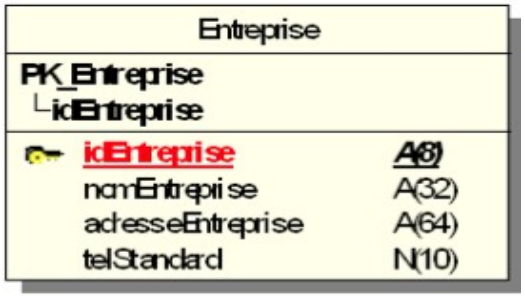
AVION (numAvion, nomAvion, localisation)
TYPEAVION (nomAvion, capacité)

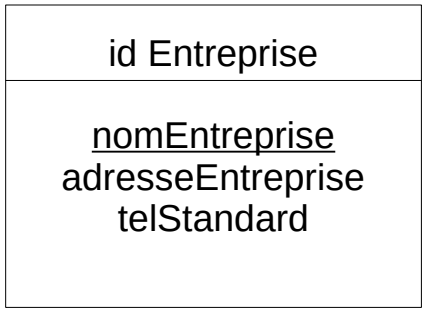
Formalisme graphique du modèle relationnel

La représentation graphique d'un modèle relationnel sous forme d'un ensemble de tables liées, appelée *Modèle Logique de Données Relationnel* (*MLDR*), permet au concepteur de mieux spécifier les liens sémantiques entre les tables et les contraintes d'intégrité référentielle associées.

La table, ses attributs et sa clé primaire

Entreprise (idEntreprise, nomEntreprise, adresseEntreprise, telStandard)



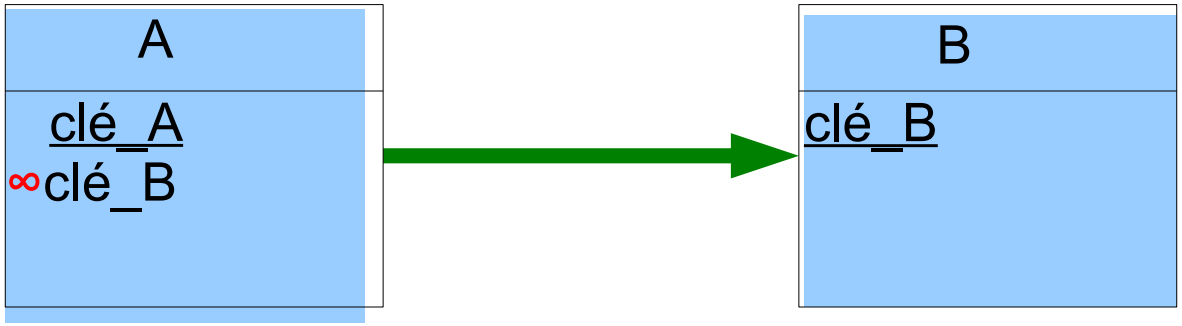


Win design
présence des domaines
de chaque attribut

Notation papier simplifiée
on se permet d'omettre les domaines
des attributs

Contraintes d'intégrité référentielle (CIR)

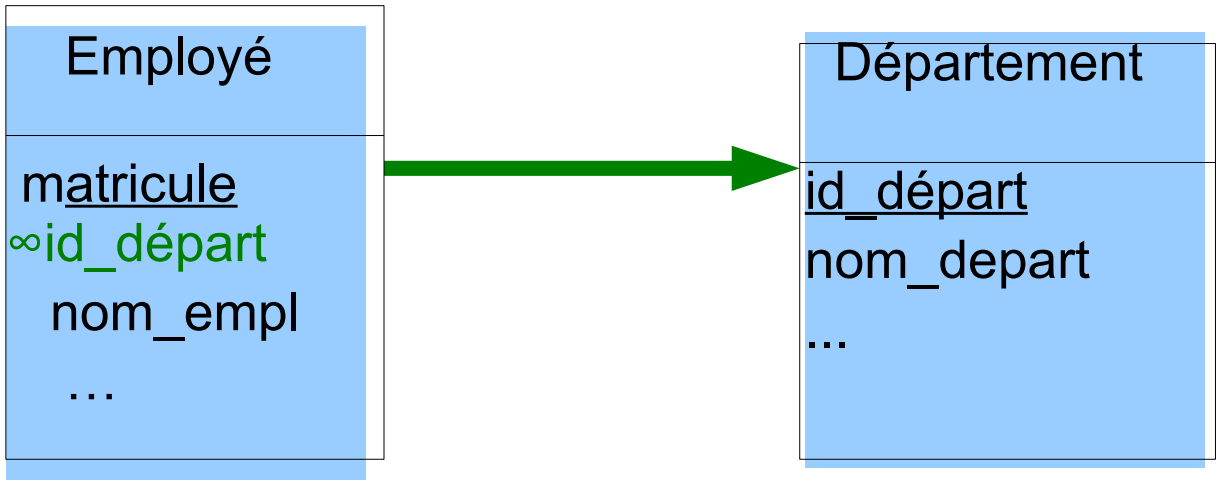
La table A fait référence à la table B dans le sens où une ligne la table A fait référence à au plus une ligne de la table B.



Cette liaison conduit à une duplication dans la table A de la clé primaire de la table B, qui devient ainsi dans la table A une clé étrangère ou clé externe.

Une clé étrangère d'une table A ne peut prendre comme valeur que NULL ou une valeur déjà existante (et unique) dans la table B de l'attribut référencé

Exemple de représentation de CIR dans un MLDR



Ce qui donne par simple lecture les schémas relationnels suivants :

Departement (id_départ,nom_départ,...)

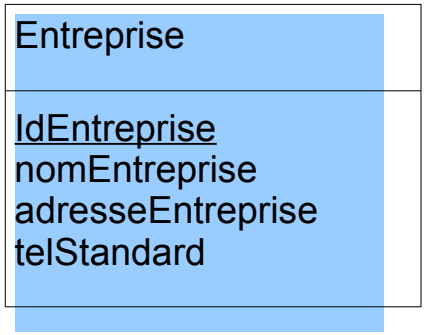
Employe (matricule, id_départ, nom_empl,...)

Dérivation de MLDR à partir de MCD

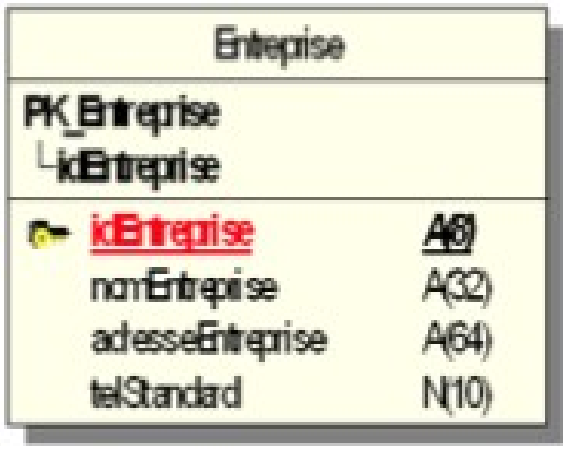
Règles de transformation : entité

Toute entité est transformée en table. Ses propriétés deviennent des attributs de la table. L'identifiant de l'entité devient la clé primaire de la table.

MCD



MLDR



Win design

ou

Notation papier simplifiée

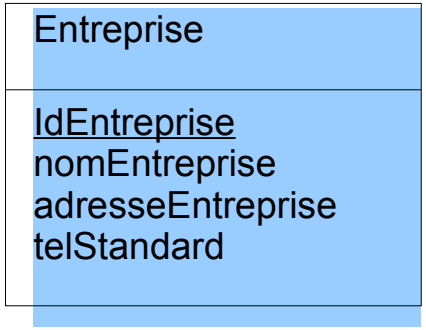


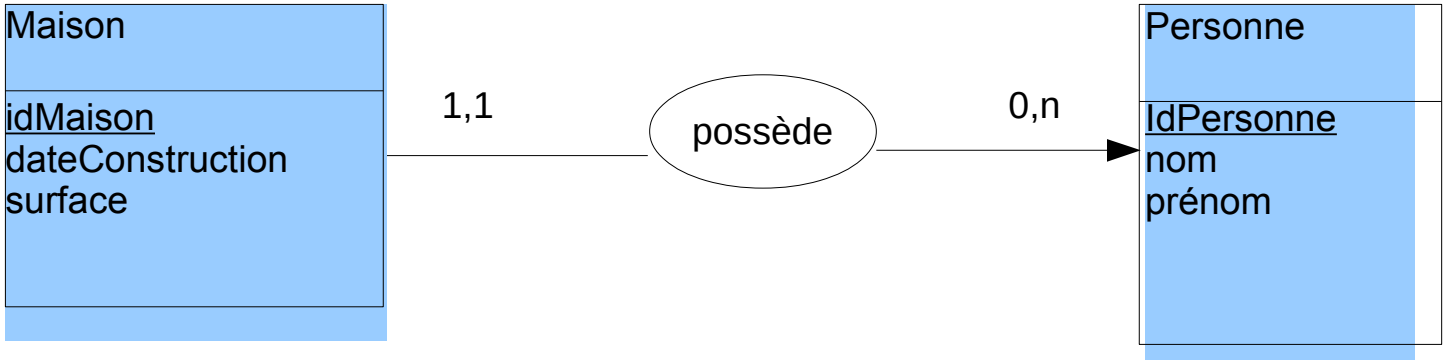
Schéma relationnel

Entreprise (idEntreprise, nomEntreprise, adresseEntreprise, telStandard)

Règles de transformation : association binaire (*,n) – (1,1)

On duplique la clé primaire de la table issue de l'entité du côté (*,n) dans la table issue de l'entité du côté (1,1) où elle devient clé étrangère avec une contrainte non NULL.

MCD



MLDR

Schéma relationnel

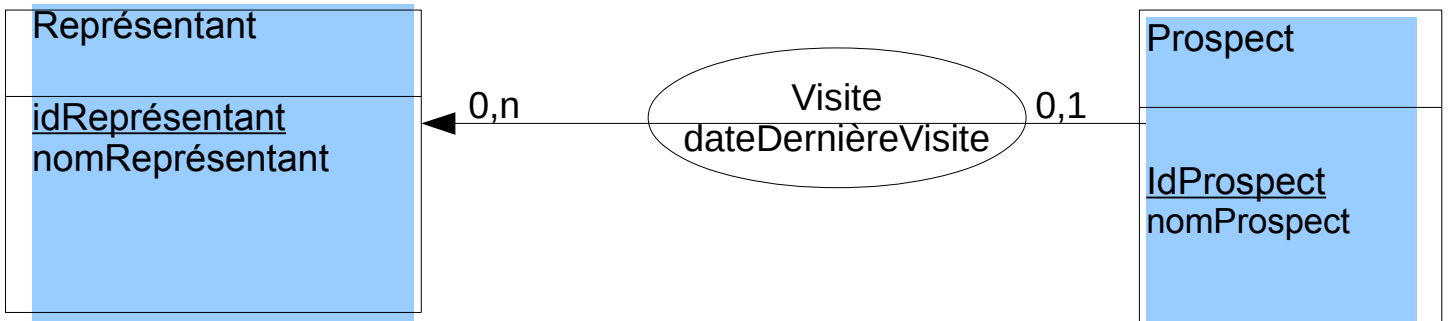
Maison (idMaison, idPersonne, dateConstruction, surface)
Personne (idPersonne, nom,prénom)

Règles de transformation : association binaire (*,n) – (0,1)

On duplique la clé primaire de la table issue de l'entité du côté (*,n) dans la table issue de l'entité du côté (0,1) où elle devient clé étrangère qui peut avoir la valeur NULL.

Les attributs de la relation (si elle en a) sont ajoutés à la table issue de l'entité du coté (0,1) et peuvent avoir la valeur NULL.

MCD



MLDR

Schéma relationnel

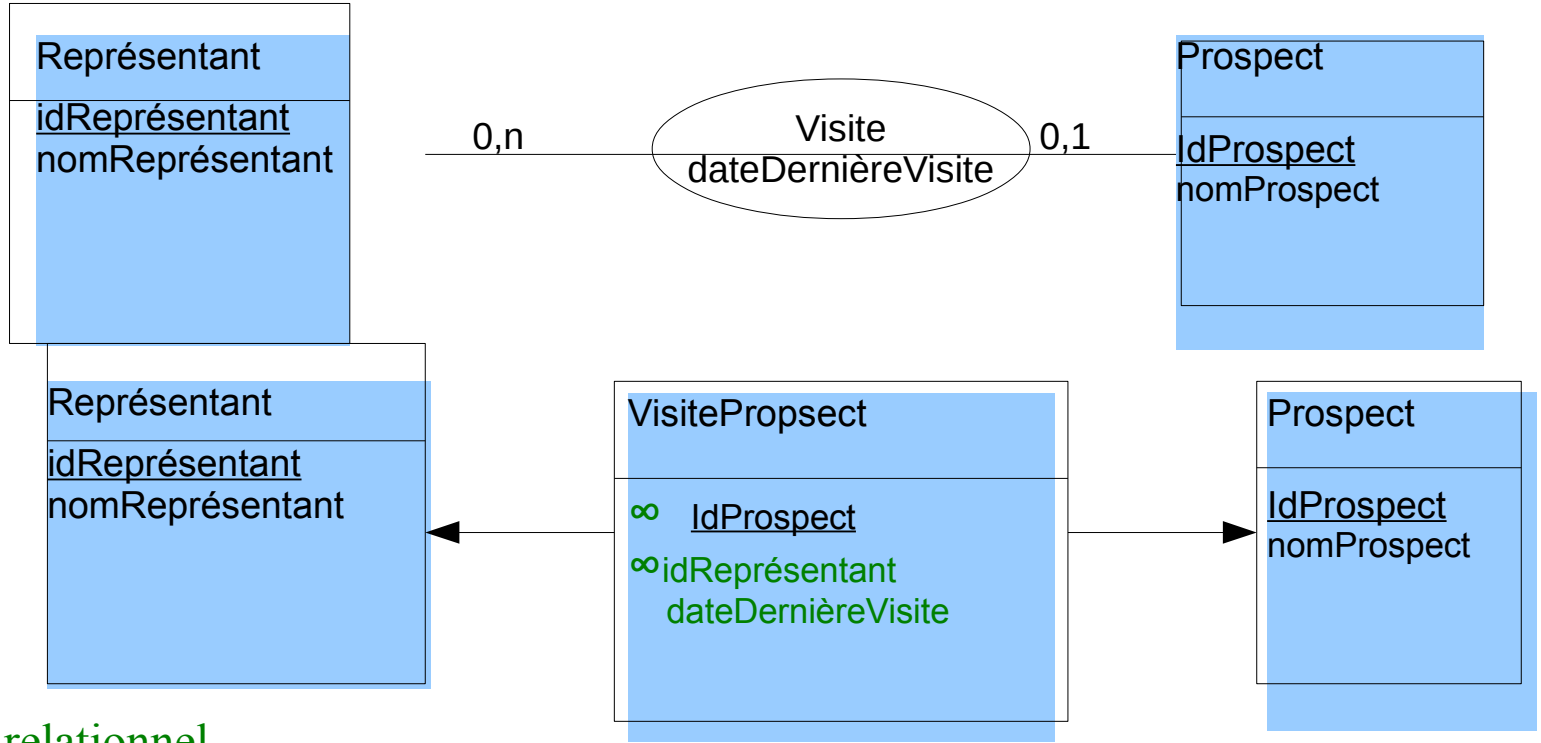
Prospect (idProspect, idRepréssentant, nomProspect, dateDernièreVisite)

Représentant(idRepréssentant, nomRepréssentant)

Règles de transformation : association binaire (*,n) – (0,1) : solution 2

La deuxième solution qui évite les valeurs NULL de la clé étrangère consiste à créer une nouvelle table avec comme clé primaire l'identifiant de l'entité du côté (0,1) ; l'identifiant de l'autre entité devient clé étrangère de cette table. Les éventuelles propriétés de l'association deviennent aussi attributs de la table issue de l'association.

MCD



MLDR

Schéma relationnel

Prospect (idProspect, nomProspect)

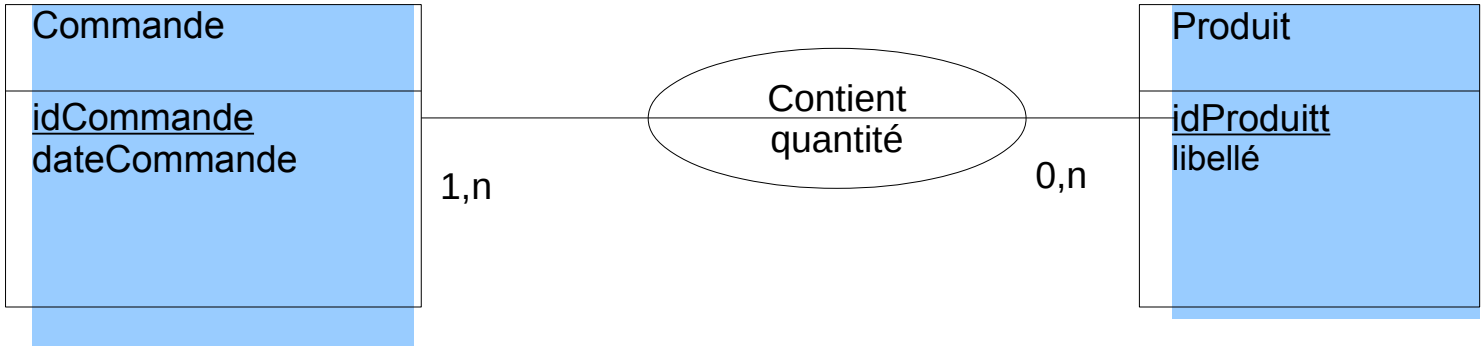
Représentant(idRepresentant, nomRepresentant)

VisiteProspect(idprospect, idRepresentant, dateDernièreVisite)

Règles de transformation : association binaire (*,n) – (*,n)

Créer une nouvelle table dont la clé primaire est composée des identifiants des deux entités. Chaque attribut qui compose la clé primaire de cette table constitue également une clé étrangère référençant la clé primaire correspondante. Les éventuelles propriétés de l'association deviennent des attributs de cette table issue de l'association.

MCD



MLDR

Schéma relationnel

Commande(idCommande, dateCommande)

Produit(idProduit,libellé)

LigneCommande(idCommande,idProduitndate,quantité)

Règles de transformation : association binaire (0,1) – (0,1)

Solution 1 :

La clé primaire de la table issue de l'une des entités est dupliquée dans la table issue de l'autre entité où elle devient clé étrangère qui peut avoir la valeur NULL.

Les attributs de la relation (si elle en a) sont ajoutés à la table dans laquelle nous avons ajouté la clé étrangère et peuvent avoir la valeur NULL.

Solution 2 :

La deuxième solution qui évite les valeurs NULL consiste à choisir une entité et créer une nouvelle table avec comme clé primaire l'identifiant de l'entité choisie.

L'identifiant de l'autre entité devient clé étrangère de cette table.

Les éventuelles propriétés de l'association deviennent aussi attributs de la table issue de l'association.

Règles de transformation : association binaire (0,1) – (1,1)

On duplique la clé primaire de la table issue de l'entité du côté (0,1) dans la table issue de l'entité du côté (1,1) où elle devient clé étrangère avec une contrainte non NULL.

Association ternaire et supérieure quelle que soit les cardinalités

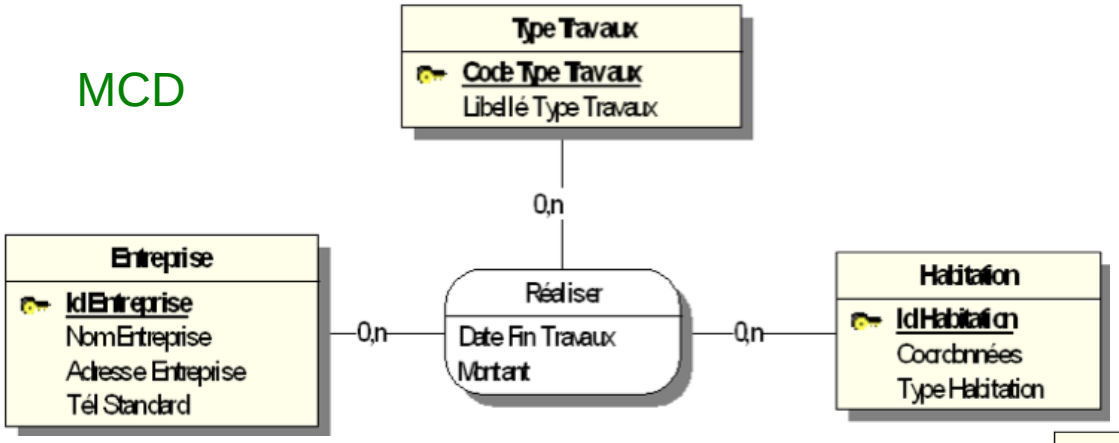
Créer une nouvelle table dont la clé primaire est composée des identifiants des diverses entités reliées par l'association considérée.

Chaque attribut qui compose la clé primaire de cette table constitue également une clé étrangère référençant la clé primaire correspondante.

Les éventuelles propriétés de l'association deviennent des attributs de cette table issue de la relation.

Exemple MCD → MLDR d'une association ternaire

MCD



MLDR

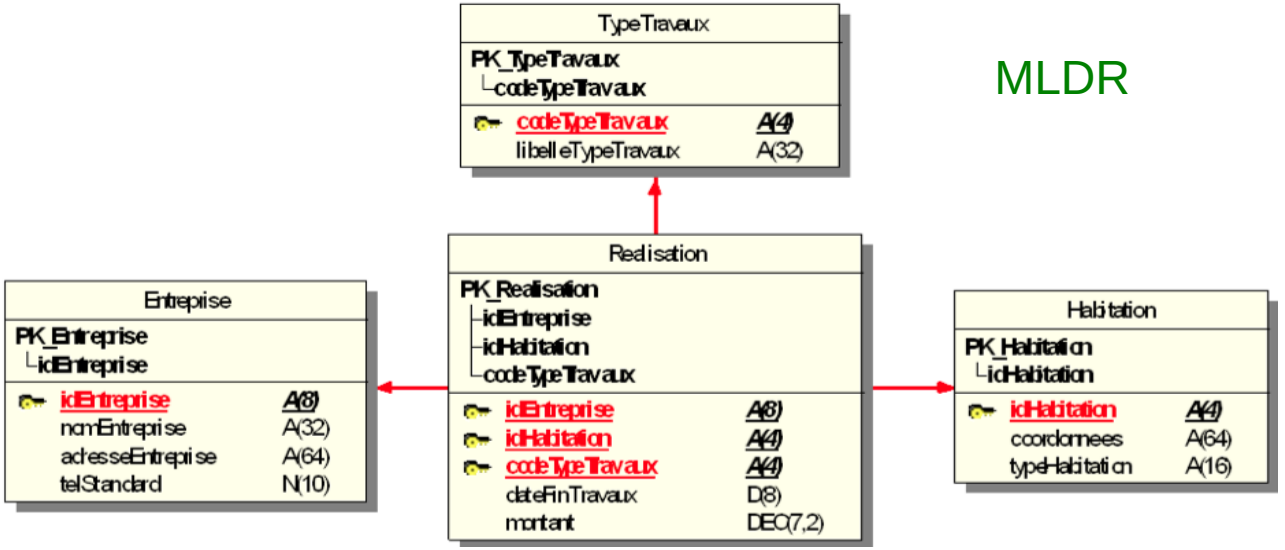


Schéma relationnel :

Entreprise (idEntreprise, nomEntreprise, adresseEntreprise, telStandard)
TypeTravaux (codeTypeTravaux, libelleTypeTravaux)
Habitation (idHabitation, coordonnees, typeHabitation)
Realisation (idEntreprise, idHabitation, codeTypeTravaux,dateFinTravaux, montant)

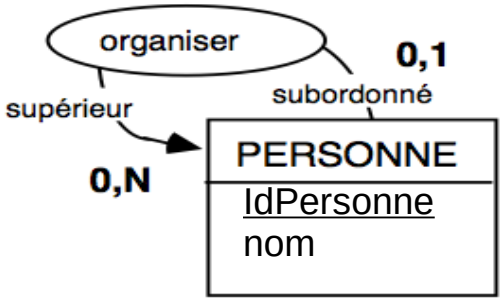
Règles de transformation : association réflexive (0,1) – (*,N)

L'identifiant de l'entité est dupliqué jouant pour sa première occurrence le rôle de clé primaire muni du rôle du côté (0,1) et pour sa deuxième occurrence comme clé étrangère muni du rôle du côté (*,N) et qui peut avoir la valeur NULL.

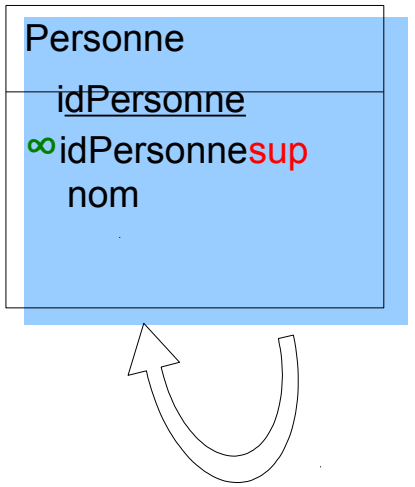
On procède à un changement d'appellation de l'attribut dupliqué en s'inspirant du nom de rôle du côté (*,N). D'ou l'importance de nommer les rôles pour les associations réflexive.

Les éventuelles propriétés de la relation deviennent des attributs de la table issue de l'entité.

MCD



MLDR



Schéma

Personne(idPersonne,idPersonnesup,nom)

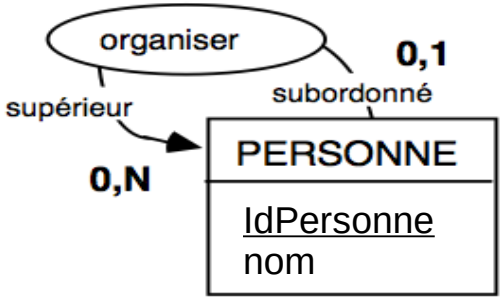
idPersonne=idPersonnesup

Règles de transformation : association réflexive (0,1) – (*,N) solution 2

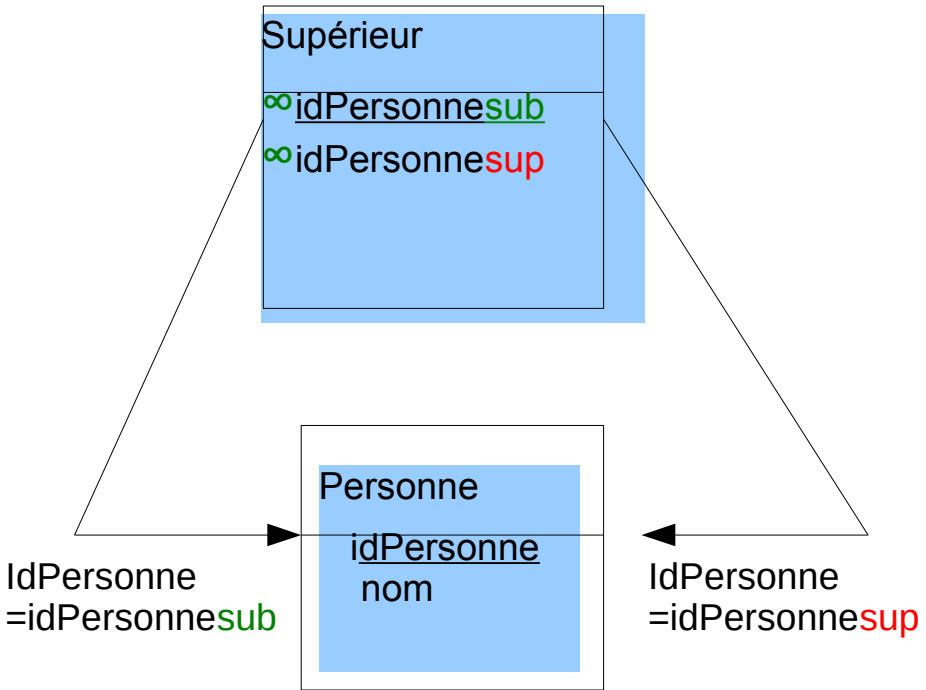
Deuxième solution qui évite les valeurs NULL à la clé étrangère crée une nouvelle table dans laquelle l'identifiant de l'entité se retrouve pour sa première occurrence, muni du rôle du côté (0,1), comme clé primaire et comme clé étrangère et pour sa deuxième occurrence, muni du rôle du côté (*,N), uniquement comme clé étrangère.

On procède à un double changement d'appellation de l'attribut dupliqué. Les éventuelles propriétés de la relation deviennent des attributs de la table issue de l'entité.

MCD



MLDR



Schéma

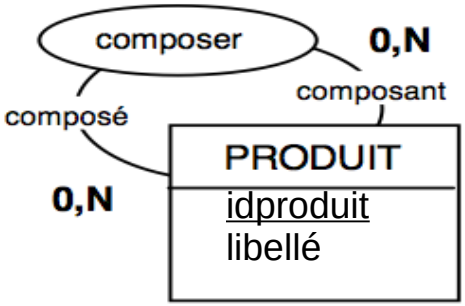
Personne(idPersonne,nom)
Supérieur(idPersonnesub,idPersonnesup)

Règles de transformation : association réflexive (*,N) – (*,N)

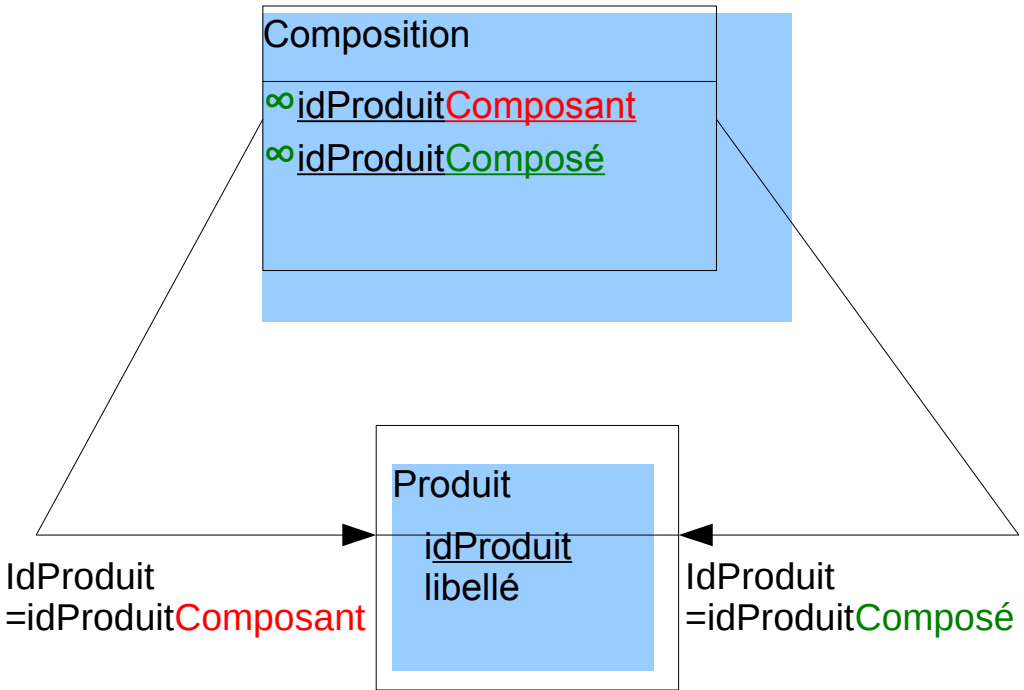
Créer une nouvelle table dont la clé est composée de deux occurrences de l'identifiant de l'entité. Les clés seront qualifiées par rapport aux rôles des pattes de la relation avec un double renommage.

Les éventuelles propriétés de la relation deviennent des attributs de cette table issue de la relation.

MCD



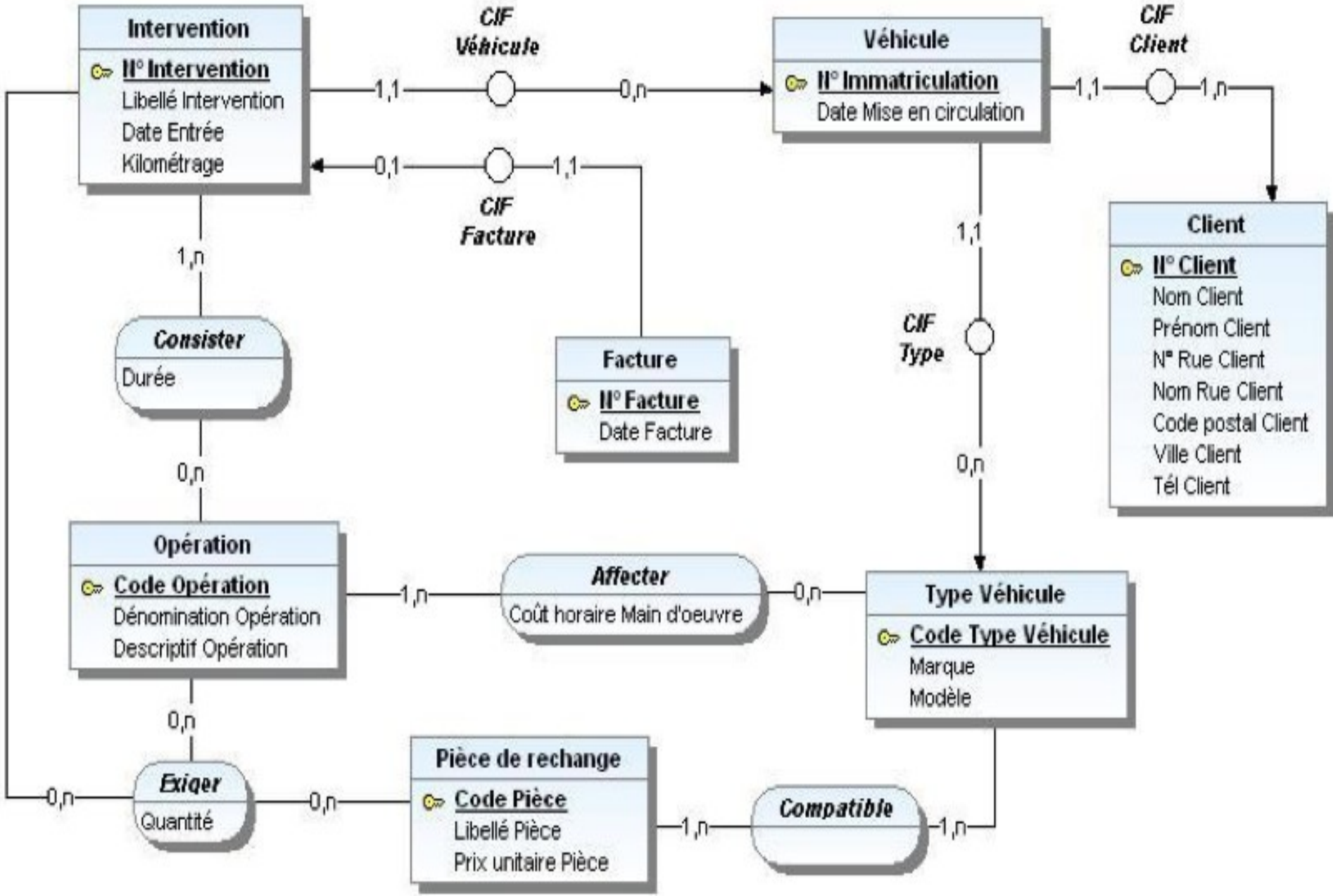
MLDR

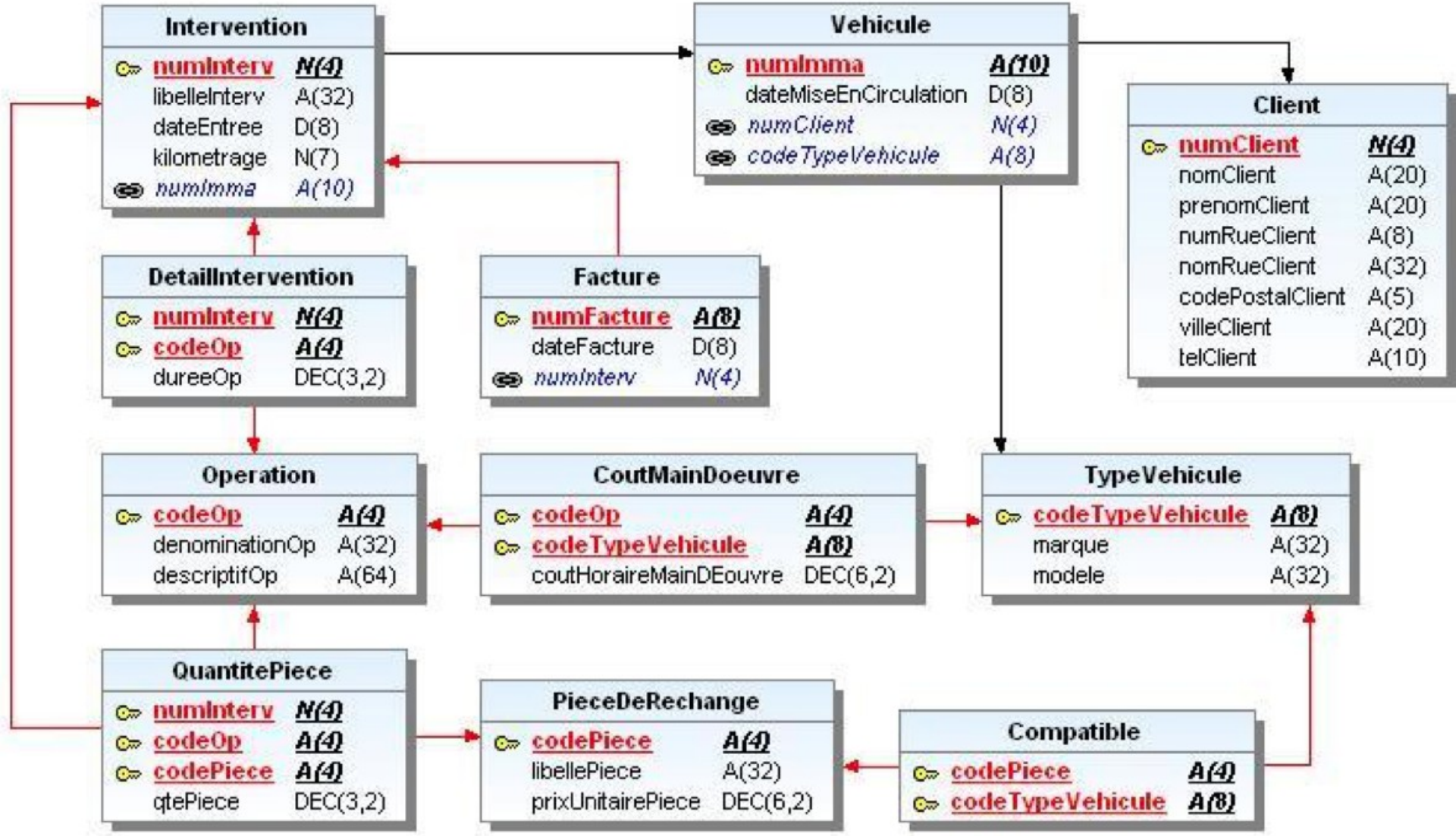


Schéma

Produit(idProduit,libellé)
Composition(idProduitComposant,
idProduitComposé)

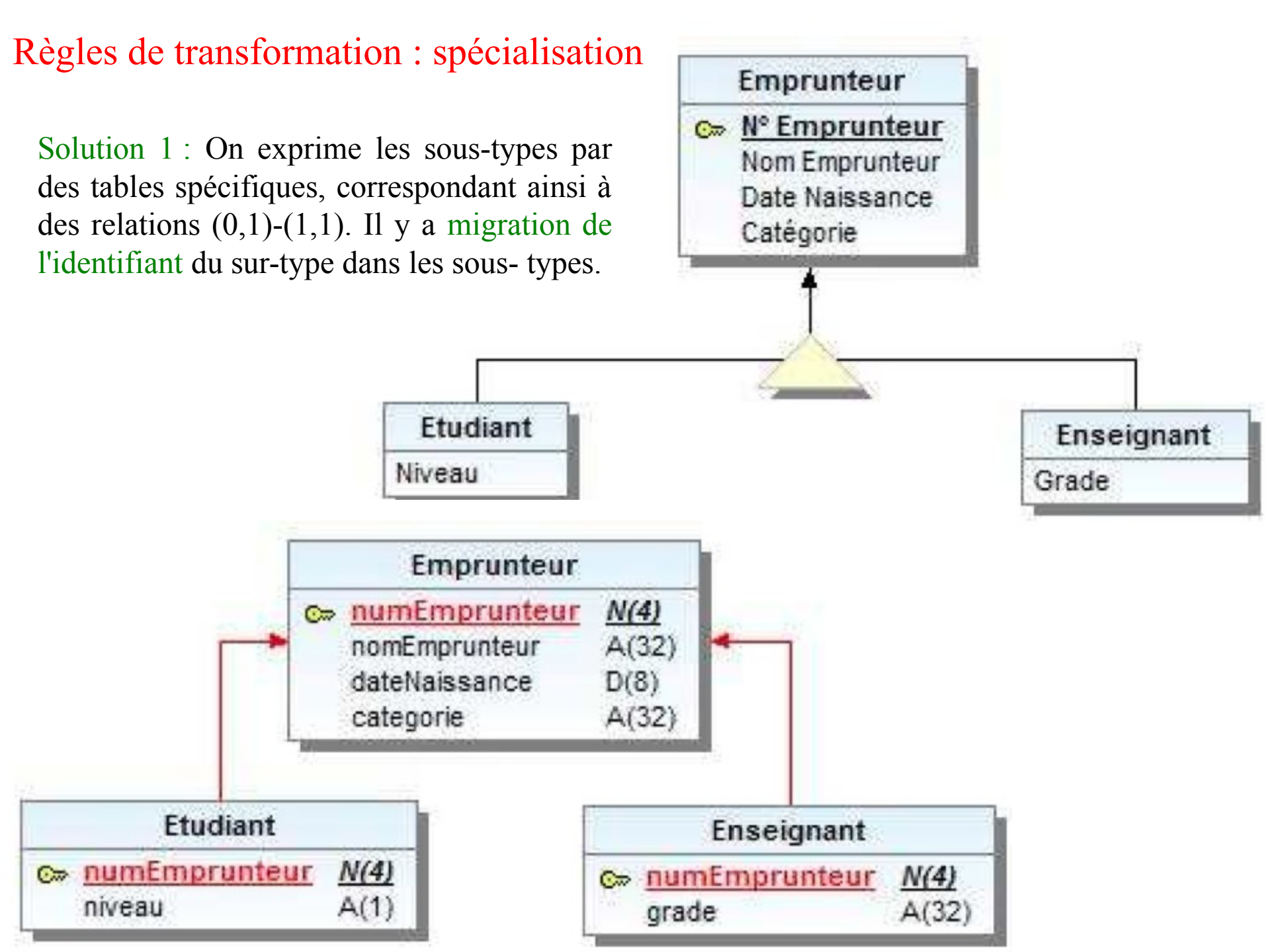
Transformons le MCD du garage automobile (cf. cours E.A)




Intervention
numInterv N(4)
libelleInterv A(32)
dateEntree D(8)
kilometrage N(7)
numImma A(10)
Vehicule
numImma A(10)
dateMiseEnCirculation D(8)
numClient N(4)
codeTypeVehicule A(8)
Client
numClient N(4)
nomClient A(20)
prenomClient A(20)
numRueClient A(8)
nomRueClient A(32)
codePostalClient A(5)
villeClient A(20)
telClient A(10)
DetailIntervention
numInterv N(4)
codeOp A(4)
dureeOp DEC(3,2)
Facture
numFacture A(8)
dateFacture D(8)
numInterv N(4)
Operation
codeOp A(4)
denominationOp A(32)
descriptifOp A(64)
CoutMainDoeuvre
codeOp A(4)
codeTypeVehicule A(8)
coutHoraireMainDEouvre DEC(6,2)
TypeVehicule
codeTypeVehicule A(8)
marque A(32)
modele A(32)
QuantitePiece
numInterv N(4)
codeOp A(4)
codePiece A(4)
qtePiece DEC(3,2)
PieceDeRechange
codePiece A(4)
libellePiece A(32)
prixUnitairePiece DEC(6,2)
Compatible
codePiece A(4)
codeTypeVehicule A(8)

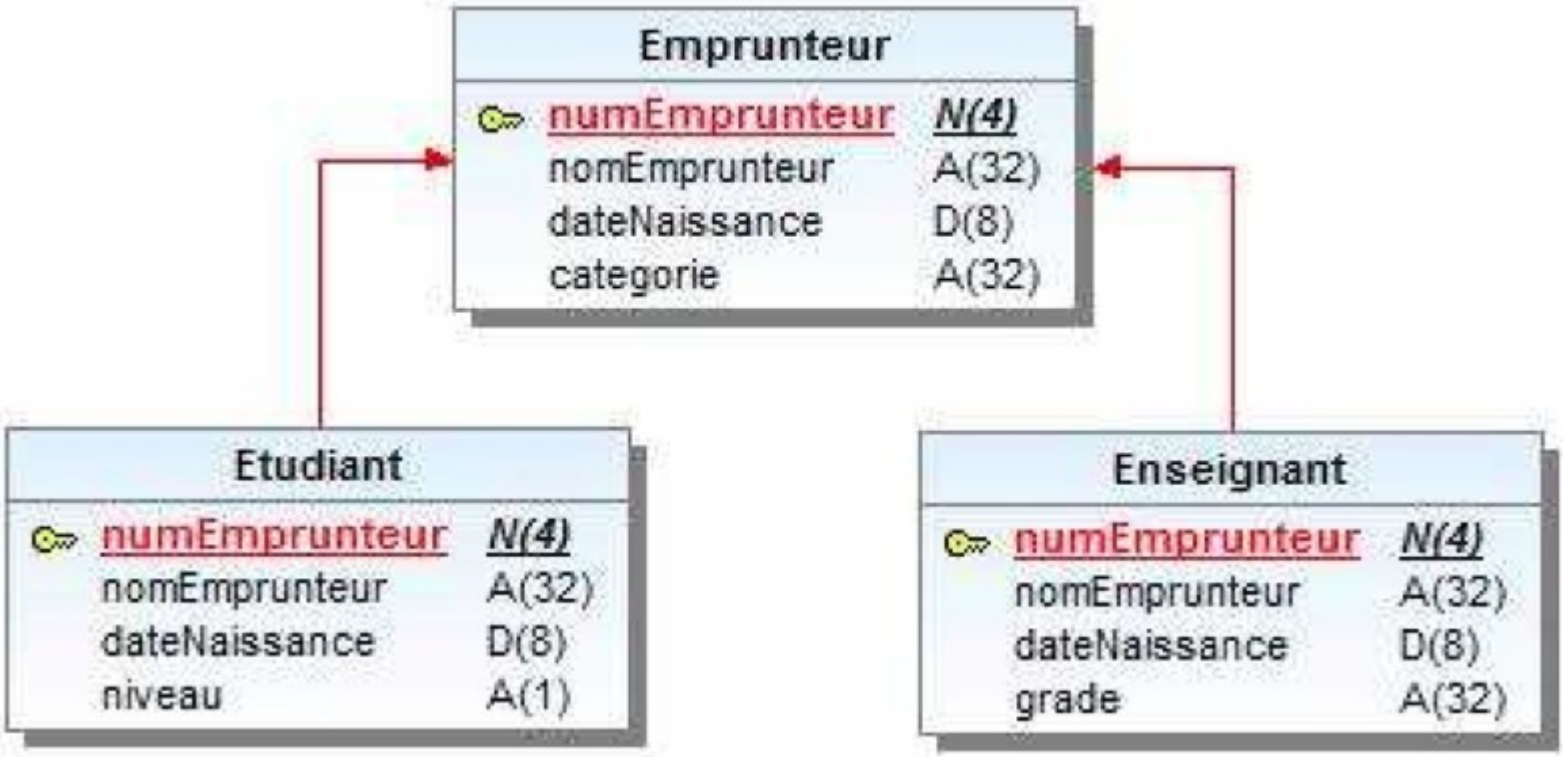
Règles de transformation : spécialisation

Solution 1 : On exprime les sous-types par des tables spécifiques, correspondant ainsi à des relations (0,1)-(1,1). Il y a migration de l'identifiant du sur-type dans les sous- types.



Règles de transformation : spécialisation

Solution 2 : on exprime les sous-types par des tables spécifiques. Il y a duplication des attributs du sur-type dans les sous-types, dont la mise à jour simultanée peut être réalisée à travers un mécanisme automatique implémentant l'héritage, par exemple par triggers.



Notez que l'attribut « Catégorie » n'est bien entendu pas copié vers les tables filles.

Règles de transformation : spécialisation

Solution 3 : on duplique la totalité des propriétés du sur-type dans les sous-types et on supprime le sur-type.

Cette solution n'est pas conseillée dans le cas où il existe, dans le MCD, des relations portant sur le sur-type.

Règles de transformation : spécialisation

Solution 4 : On transfère la totalité des propriétés des sous-types dans la table correspondant au sur-type.

On exprime les sous-types par des vues relationnelles de la table du sur- type : il faut disposer, par exemple, de l'attribut qui a permis la mise en évidence des sous-types, par exemple la catégorie dans l'entité « personne ».

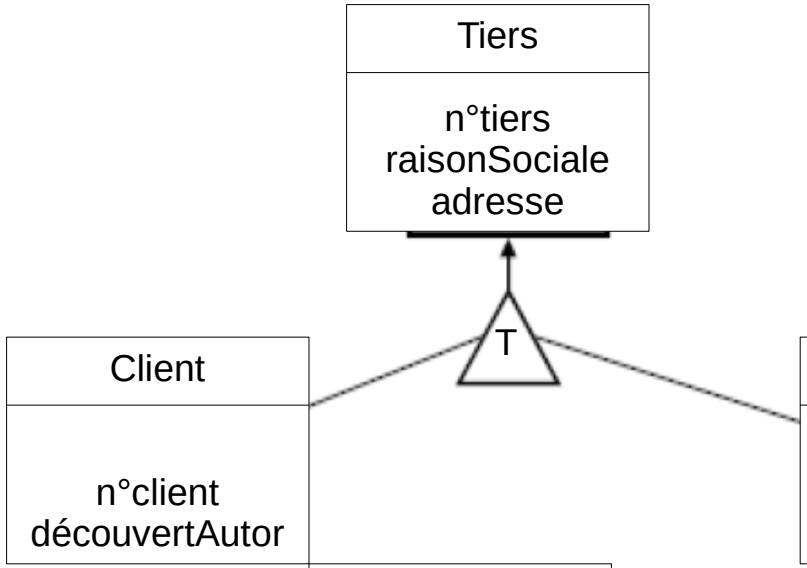
On doit alors ajouter deux vues supplémentaires :

EmprunteurEtudiant : sélection des individus dont la catégorie = étudiant
EmprunteurEnseignant : sélection des individus dont la catégorie = enseignant.

Règles de transformation : généralisation

Dans le cas d'une généralisation, les sous-types ont leurs propres identifiants. Ainsi, la solutions 1 précédente est possible sauf que l'identifiant migré du sur-type vers les sous-types ne sera pas clé primaire dans les tables sous types mais clé candidate (et étrangère).

MCD



MLDR

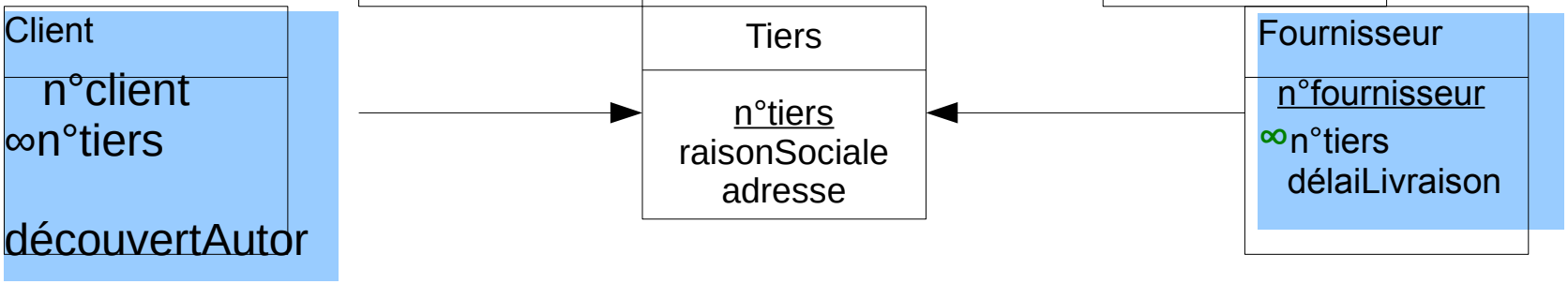
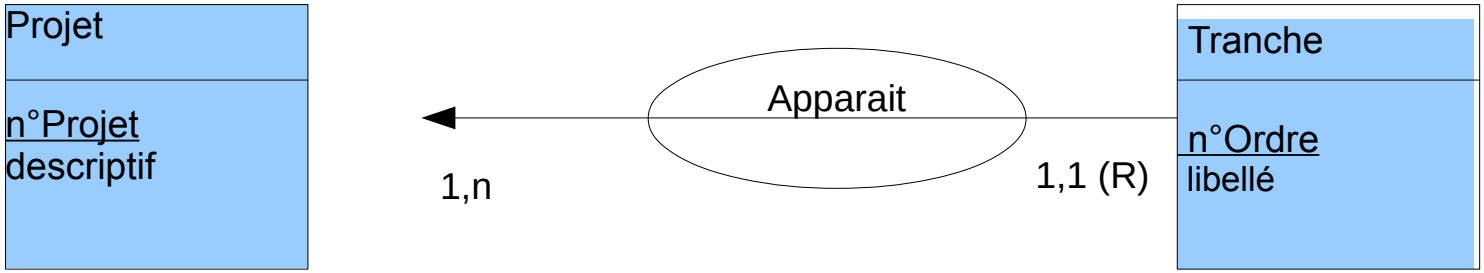


Schéma relationnel : Tiers(n°tiers, raisonSociale, adresse)
Client(n°client,n°tiers, découvertAutor)
Fournisseur(n°fournisseur,n°tiers, délaiLivraison)

Règles de transformation : identifiant relatif

La clé de la table issue de l'entité faible est constituée de la concaténation de l'identifiant de l'entité forte et de l'identifiant relatif de l'entité faible. L'identifiant de l'entité forte joue également le rôle de clé étrangère au sein de la table issue de l'entité faible.

MCD



MLDR

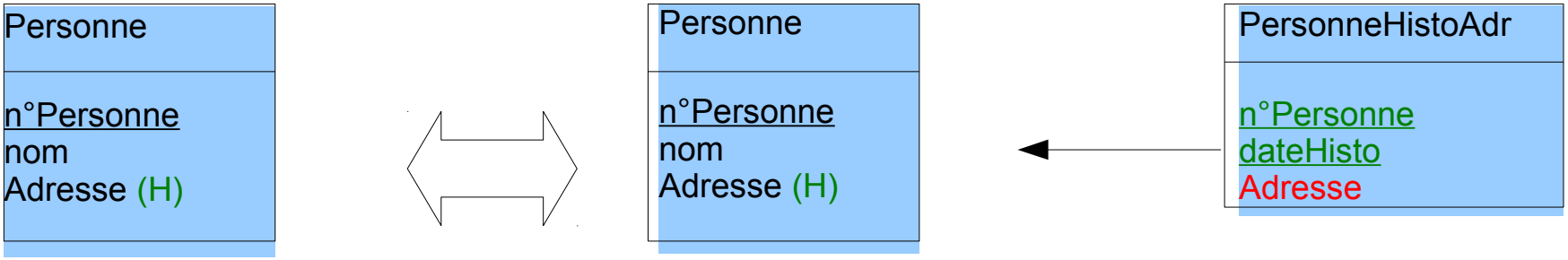
Schéma relationnel

projet(n°Projet,descriptif)

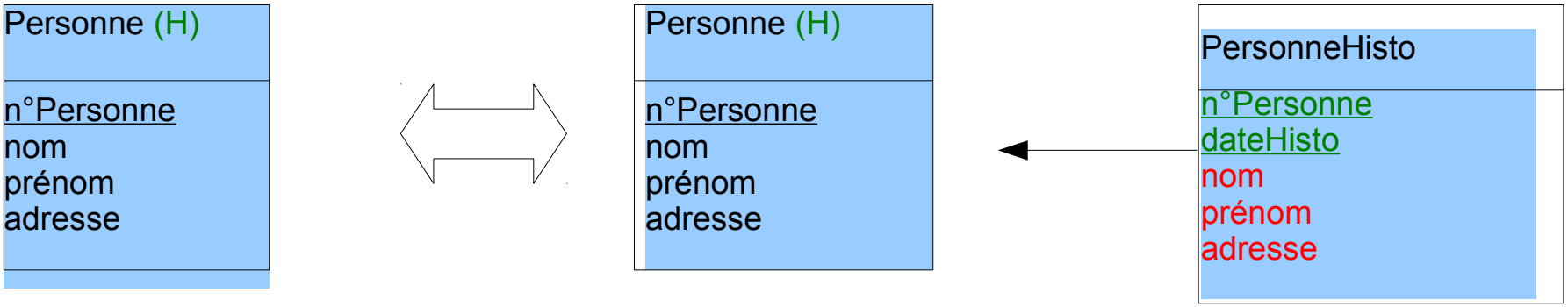
Tranche(n°Projet,n°Ordre,descriptif)

Règles de transformation : historisation

Attribut



Entité

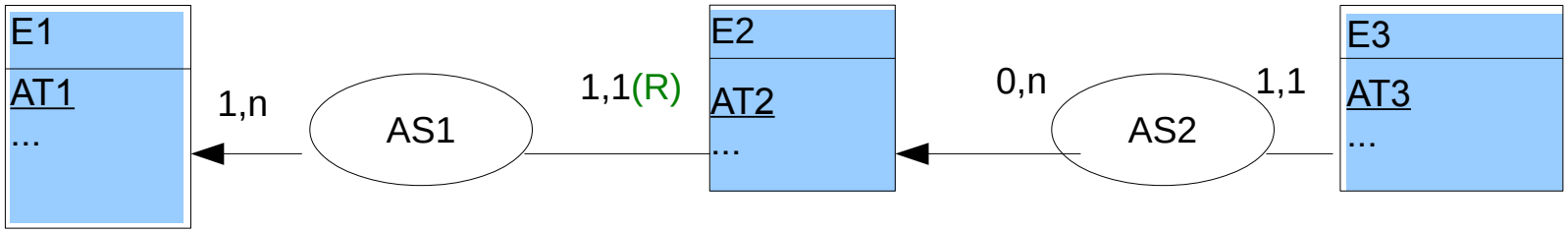


Même principe à appliquer en cas d'historisation d'une association et de ses attributs.

Remarques importantes

Nettoyage des tables : ne pas garder à la fin des tables avec qu'une seule colonne.

Propagez les modifications obtenues à chaque étape vers les tables déjà transformées, par exemple sur la traduction du schéma ci-dessous :



La bonne solution est : E1(AT1,...) E2(AT1,AT2,...) E3(AT3,AT1,AT2) Pourquoi ?

Si on transforme AS2 en premier on obtient E2(AT2,...) E3(AT3,AT2).

La transformation de AS1 donne E2(AT1,AT2) → la clé de E2 a changé → il faut mettre à jour E3 car elle était liée à E2 bien que l'on ait déjà transformé AS2 !

Il faut donc ajouter AT1 dans E3 pour obtenir le schéma final correct :
E1(AT1,...) E2(AT1,AT2,...) E3(AT3,AT1,AT2) et surtout pas
E1(AT1,...) E2(AT2,...) E3(AT3,AT2)