INF5180

Conception et exploitation d'une base de données

Cours 5

Zied Zaier, PhD
Département d'informatique
Université du Québec à Montréal

Cours 5

CONCEPTION LOGIQUE: TRADUCTION DU MODÈLE CONCEPTUEL DE DONNÉES EN SCHÉMA CONCEPTUEL RELATIONNEL

CE DOCUMENT EST INSPIRÉ DES TRAVAUX DES PROFESSEURS ROBERT GODIN, VALTCHEV PETKO ET FATIHA SADAT.

Sommaire

Partie I du cours

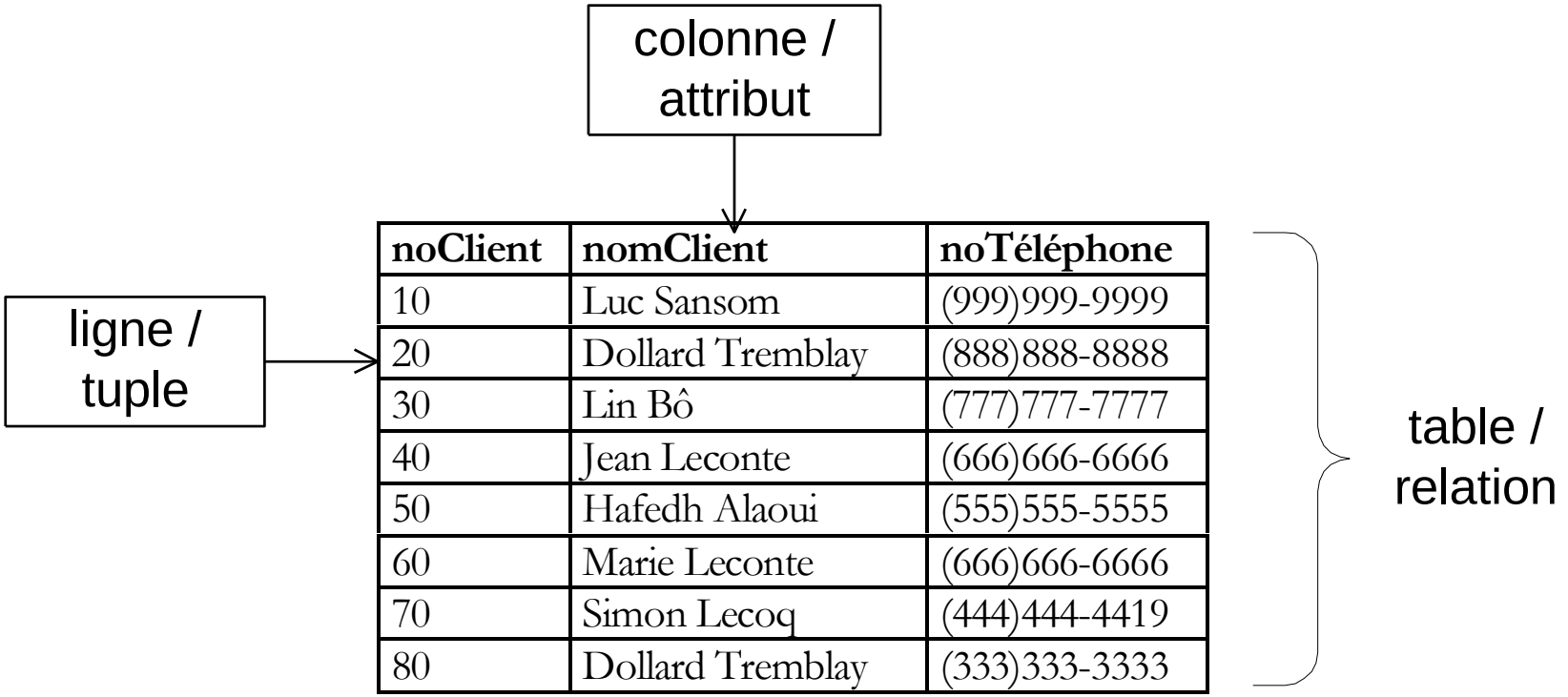
- Première ébauche du schéma: une table par classe
- Traduction des attributs
- Réalisation de l'identité par des clés primaires

Partie II du cours

- Traduction des associations
- Relations de généralisation/spécialisation
- Traduction des autres contraintes

Concepts de base

- Domaine : ensemble de valeurs
- Relation : ensemble de n-uplets (tuples)
- SQL : multi-ensemble



noClient	nomClient	noTéléphone
10	Luc Sansom	(999)999-9999
20	Dollard Tremblay	(888)888-8888
30	Lin Bô	(777)777-7777
40	Jean Leconte	(666)666-6666
50	Hafedh Alaoui	(555)555-5555
60	Marie Leconte	(666)666-6666
70	Simon Lecoq	(444)444-4419
80	Dollard Tremblay	(333)333-3333

Concepts de base (Suite)

- **Deux facettes du concept de table**
  - **Schéma d'une table (table schema)**
    - définition de son type (intention)
    - ex: Client(noClient, nomClient, noTéléphone)
  - **Instance ou extension d'une table**
    - état de la table
    - ~ variable qui contient un ensemble de lignes
    - modifications d ’état
- **Schéma relationnel**
  - ensemble de schémas de tables

Instance de BD VentesPleinDeFoin

Table *Commande*		
noCommande	**dateCommande**	**noClient**
1	01/06/2000	10
2	02/06/2000	20
3	02/06/2000	10
4	05/07/2000	10
5	09/07/2000	30
6	09/07/2000	20
7	15/07/2000	40
8	15/07/2000	40

Table *Client*		
noClient	**nomClient**	**noTéléphone**
10	Luc Sansom	(999)999-9999
20	Dollard Tremblay	(888)888-8888
30	Lin Bô	(777)777-7777
40	Jean Leconte	(666)666-6666
50	Hafedh Alaoui	(555)555-5555
60	Marie Leconte	(666)666-6666
70	Simon Lecoq	(444)444-4419
80	Dollard Tremblay	(333)333-3333

Table *LigneCommande*		
noCommande	**noArticle**	**quantité**
1	10	10
1	70	5
1	90	1
2	40	2
2	95	3
3	20	1
4	40	1
4	50	1
5	70	3
5	10	5
5	20	5
6	10	5
6	40	1
7	50	1
7	95	2
8	20	3

Table *DétailLivraison*			
noLivraison	**noCommande**	**noArticle**	**quantitéLivrée**
100	1	10	7
100	1	70	5
101	1	10	3
102	2	40	2
102	2	95	1
100	3	20	1
103	1	90	1
104	4	40	1
105	5	70	2

Table *Livraison*	
noLivraison	**dateLivraison**
100	3/06/2000
101	4/06/2000
102	4/06/2000
103	5/06/2000
104	7/07/2000
105	9/07/2000

Table *Article*			
noArticle	**description**	**prixUnitaire**	**quantitéEnStock**
10	Cèdre en boule	10.99	10
20	Sapin	12.99	10
40	Epinette bleue	25.99	10
50	Chêne	22.99	10
60	Erable argenté	15.99	10
70	Herbe à puce	10.99	10
80	Poirier	26.99	10
81	Catalpa	25.99	10
90	Pommier	25.99	10
95	Génévrier	15.99	10

Contraintes d'intégrité du modèle relationnel
Contrainte de domaine et de valeur non nulle

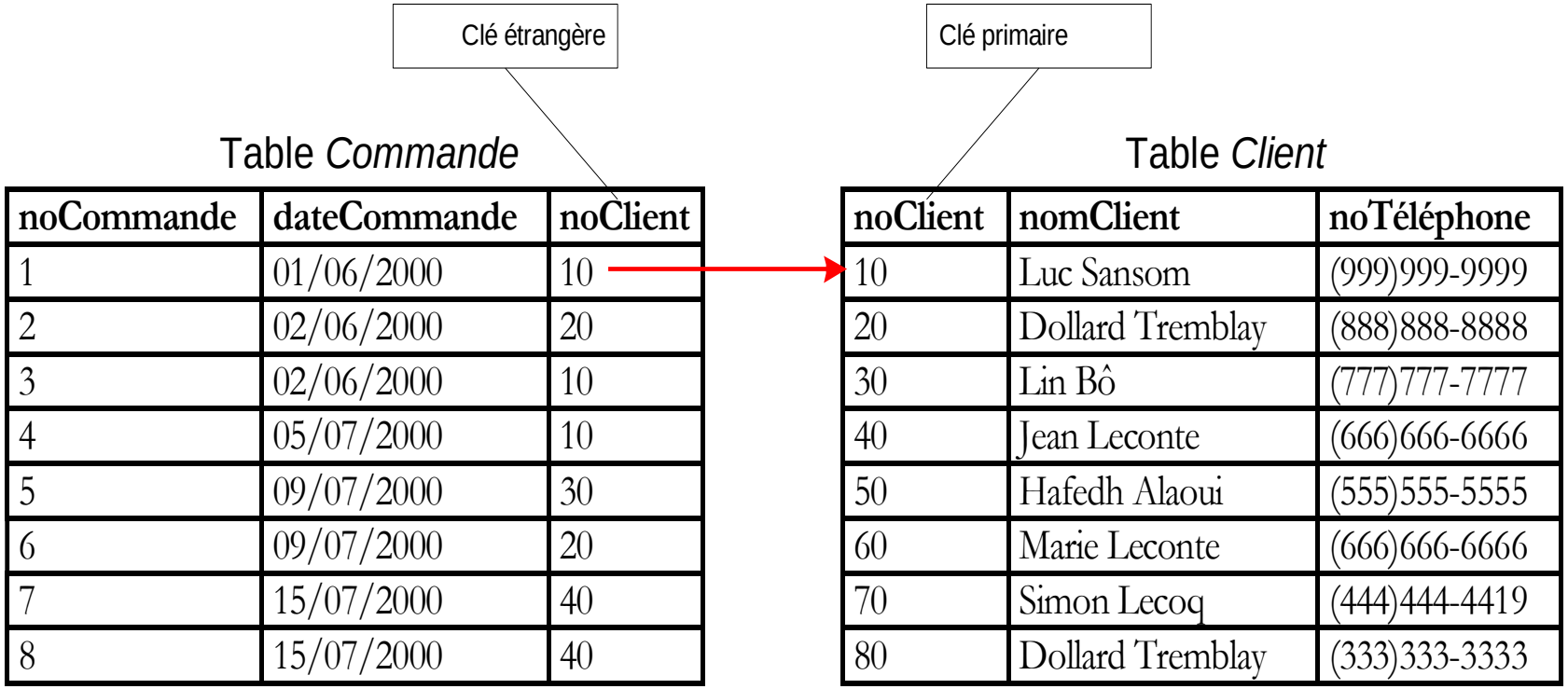
- $T(A_1 : D_1, A_2 : D_2, \ldots, A_n : D_n)$
  - D_i : domaine de A_i
- Valeur nulle
  - Comportement particulier
  - Valeur inconnue
  - Valeur non applicable
  - …

Clé primaire et contrainte d'entité

- **Clé unique (unique key)**
  - **ou superclé (superkey)**
  - identifiant
- **Clé candidate (candidate key)**
  - clé unique minimale
- **Clé primaire (primary key)**
  - sert de mécanisme de référence aux lignes de la table
- **Contrainte d'entité (entity constraint)**
  - ∃ clé primaire
  - non nulle

Contrainte d'intégrité référentielle

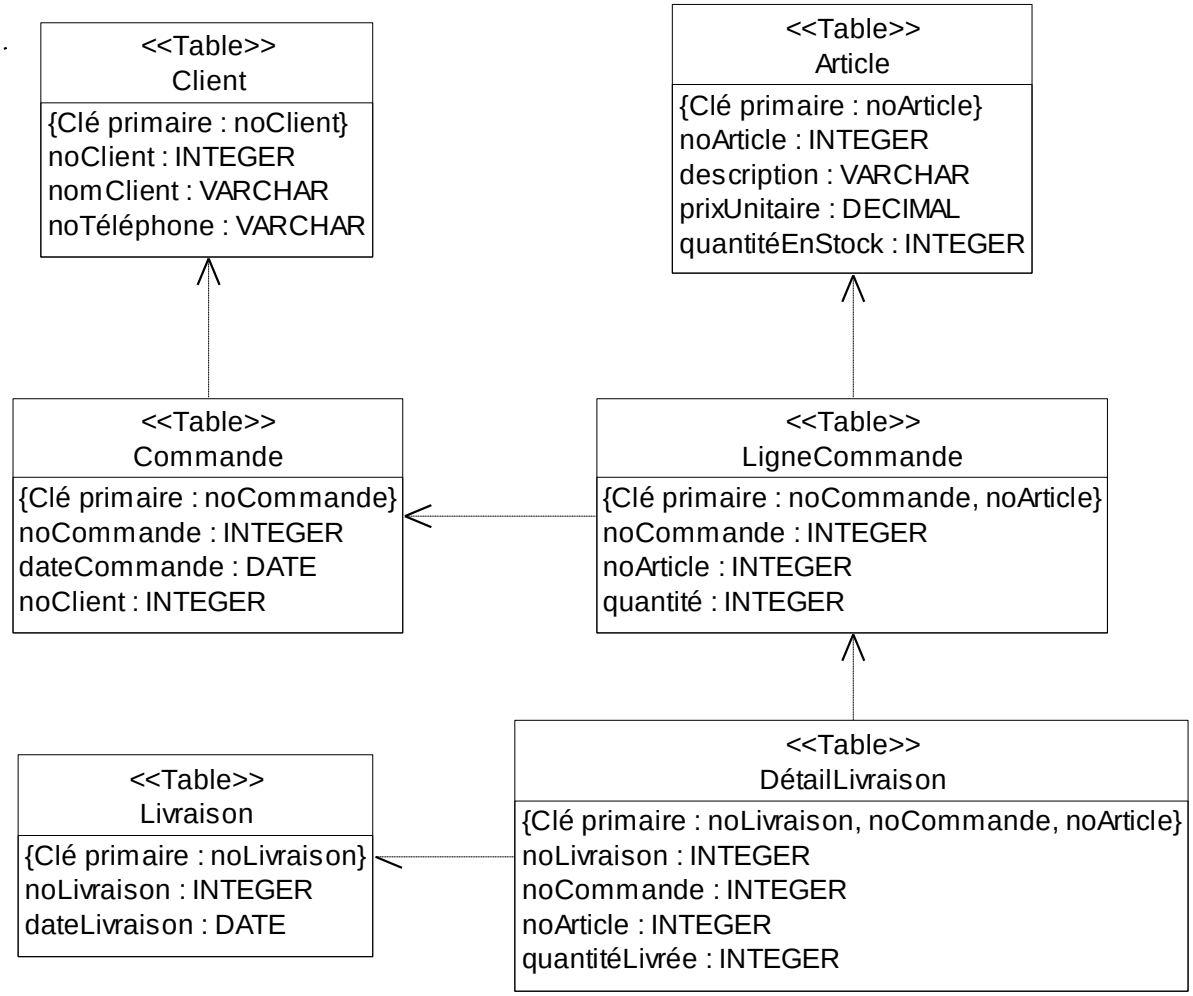
- Clé étrangère non nulle → clé primaire



noCommande	dateCommande	noClient
1	01/06/2000	10
2	02/06/2000	20
3	02/06/2000	10
4	05/07/2000	10
5	09/07/2000	30
6	09/07/2000	20
7	15/07/2000	40
8	15/07/2000	40

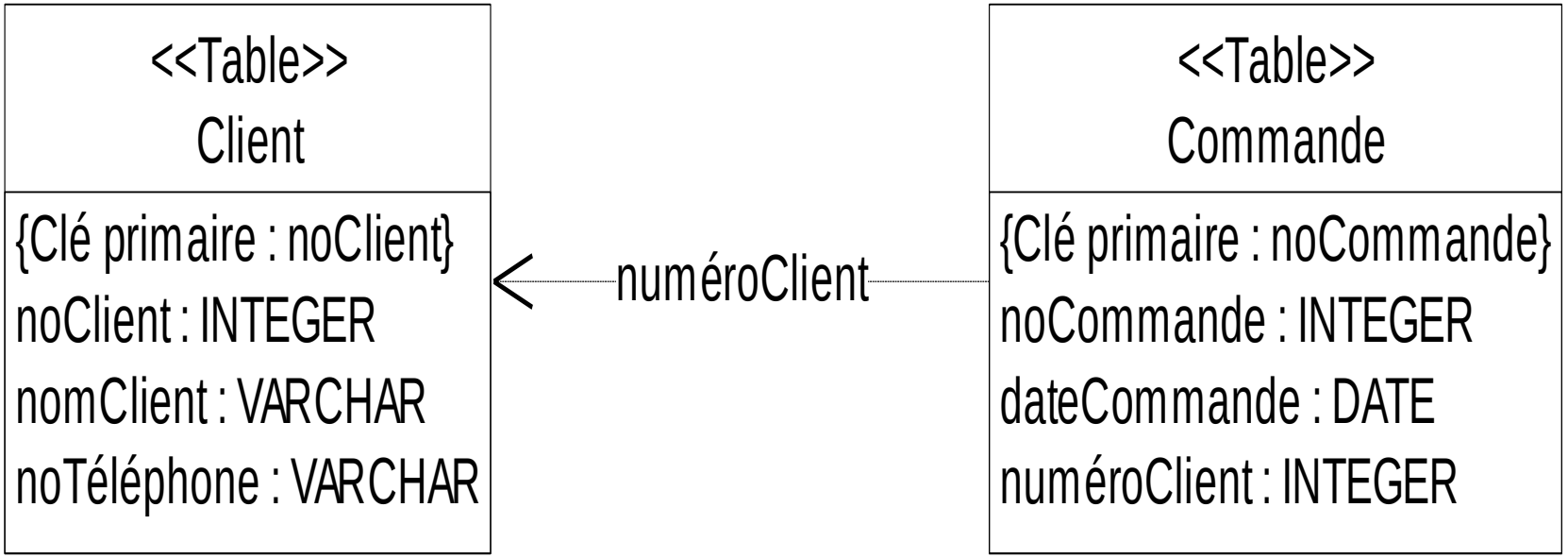
noClient	nomClient	noTéléphone
10	Luc Sansom	(999)999-9999
20	Dollard Tremblay	(888)888-8888
30	Lin Bô	(777)777-7777
40	Jean Leconte	(666)666-6666
50	Hafedh Alaoui	(555)555-5555
60	Marie Leconte	(666)666-6666
70	Simon Lecoq	(444)444-4419
80	Dollard Tremblay	(333)333-3333

Représentation UML



Nom clé étrangère ≠ nom clé primaire

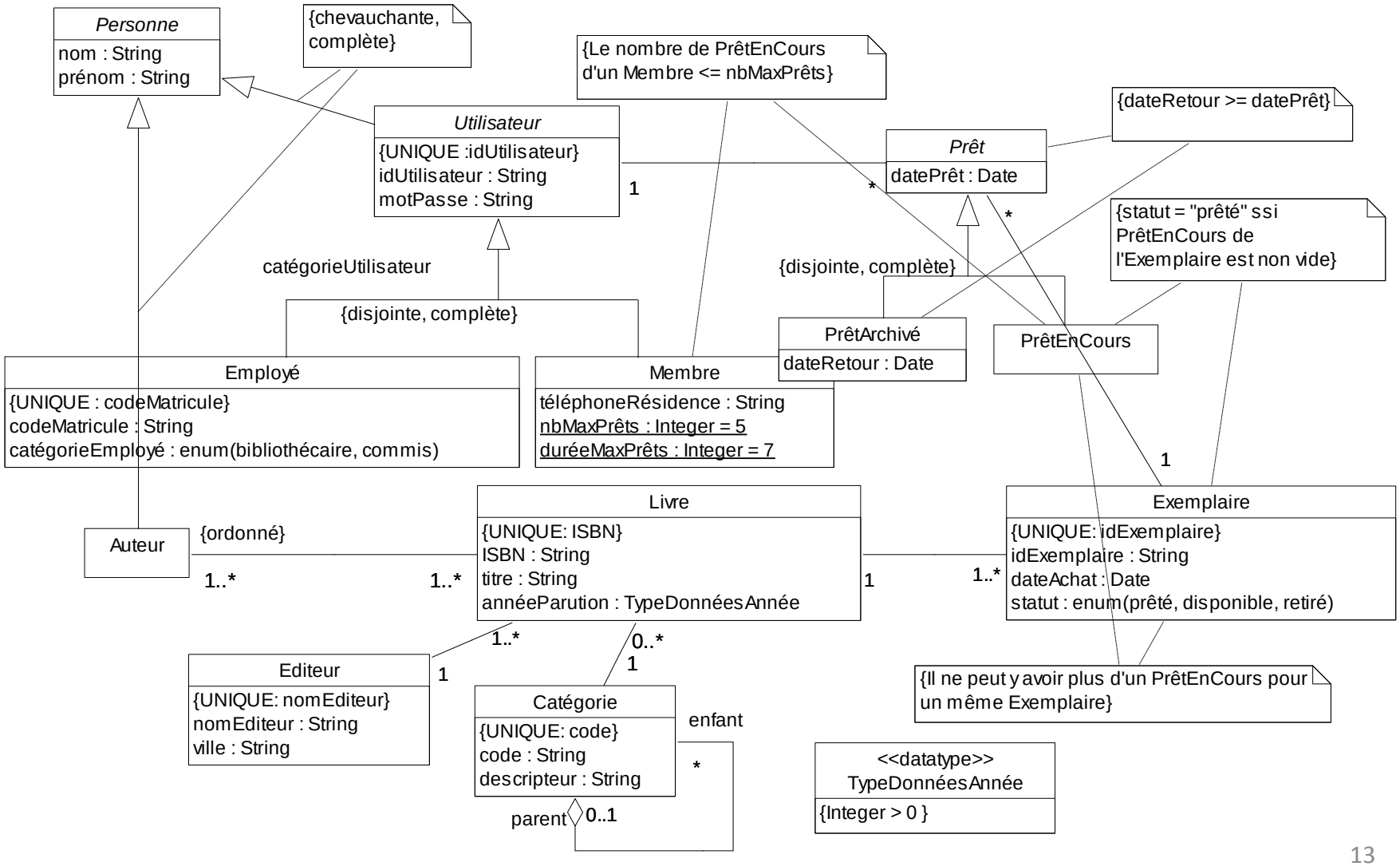
- Étiquette de la relation de dépendance



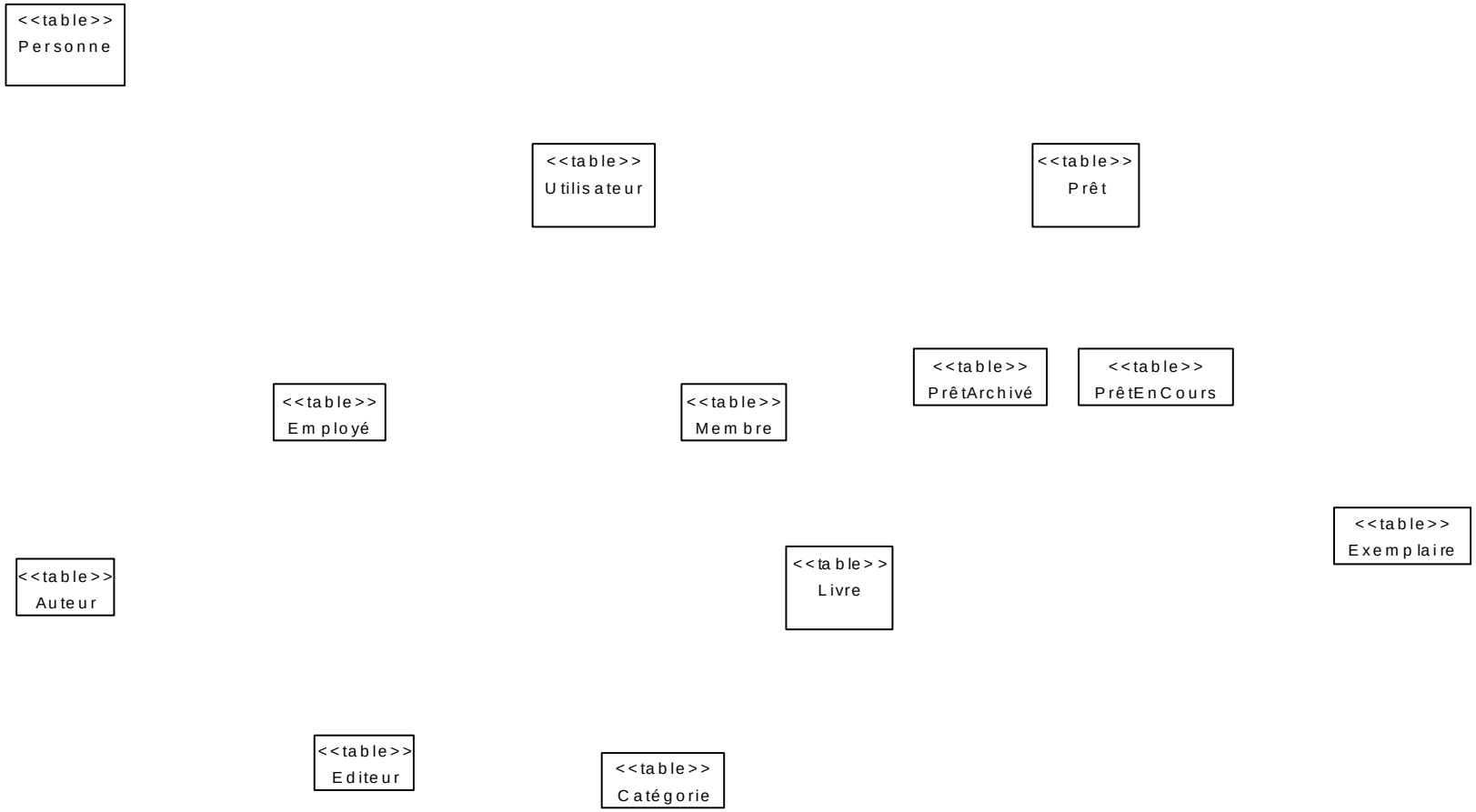
Conception logique : traduction du modèle conceptuel de données en schéma conceptuel relationnel

- Output : schéma relationnel (niveau conceptuel)
  - tables
  - colonnes
    - types, domaines, ...
  - contraintes d ’intégrité
    - clés primaires, étrangères, ...
- Input à la conception
  - modèle conceptuel des données (*en UML*)
- Appelée aussi conception logique de la BD
  - ne fait pas intervenir les considérations de performance
- Mise à jour possible à l’étape conception

Rappel - modèle conceptuel des données - SYLERAT

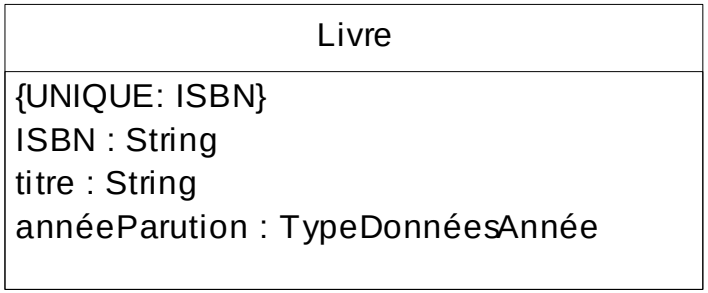


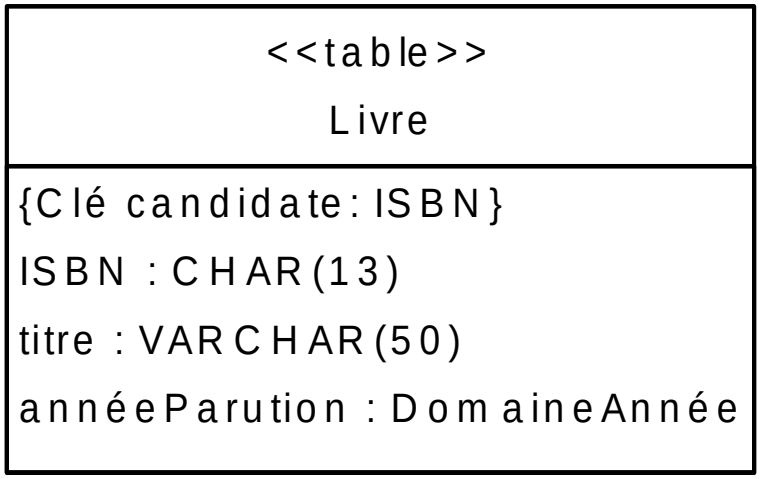
Première ébauche du schéma : une table par classe



Traduction des attributs

- Attribut de la classe -> colonne de la table

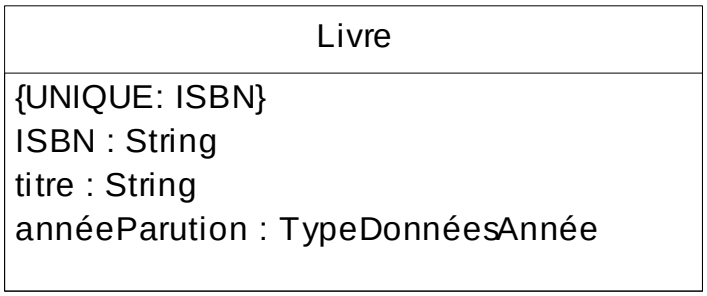


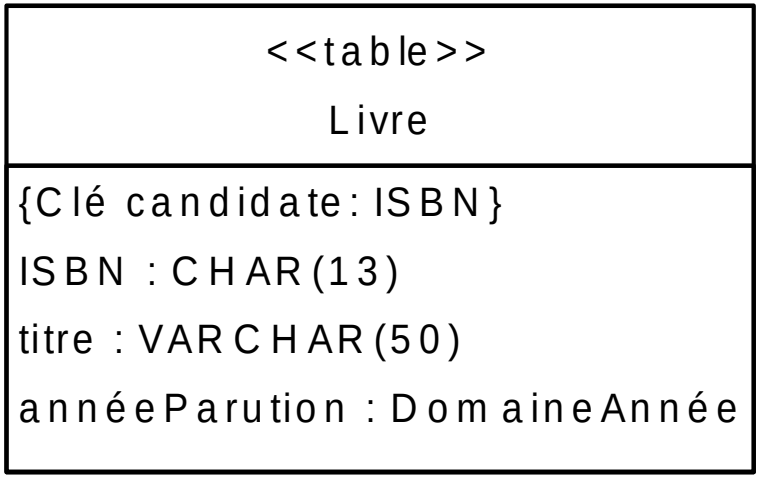


- La multiplicité minimale de 1 est transformée en contrainte de colonne obligatoire
- La multiplicité 0 devient une colonne optionnelle

TRADUCTION DES CONTRAINTES D'IDENTIFICATION (UNIQUE)

- Contrainte UNIQUE -> clé candidate relationnelle
- – clé primaire ?





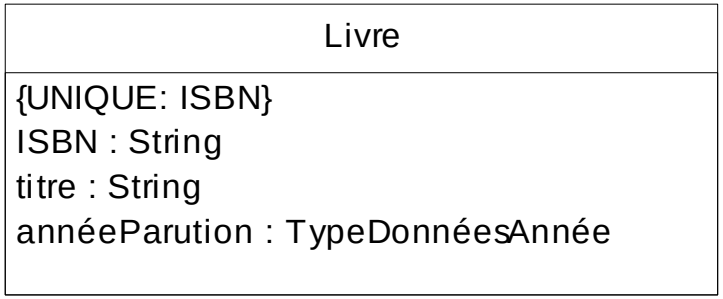
- La clé candidate peut être considérée comme clé primaire

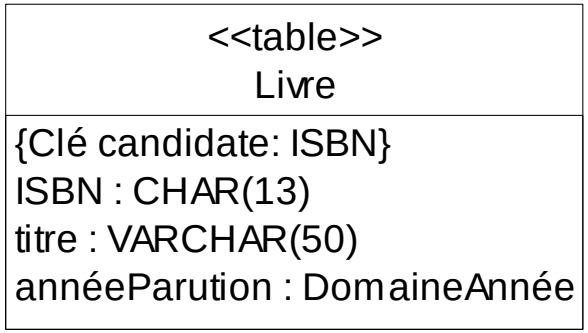
TRADUCTION DES TYPES DE DONNÉES

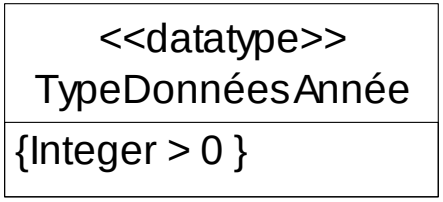
Type OCL	**Type SQL2**	**Oracle 8**
Boolean	BIT(1)	Non supporté (utiliser CHAR(1) + CHECK)
Integer	INTEGER ou SMALLINT	NUMBER(n)
String	CHARACTER (CHAR) (n), CHARACTER VARYING (VARCHAR) (n)	VARCHAR2(n), LONG ou LONG VARCHAR (chaîne jusqu'à 2G), CLOB (chaîne jusqu'à 4G), NCLOB (chaîne pour caractères encodés sur plusieurs octets)
Real	NUMERIC(p,s) (précision exacte), DECIMAL(p,s), REAL, DOUBLE PRECISION, FLOAT(p)	NUMBER(p,s)
Enum{v1,…vn}	CHARACTER (CHAR) ou VARCHER + CHECK … IN (v1,…,vn) (possibilité de création de domaine)	Domaine non supporté
	DATE	DATE inclut TIME
	TIME	
	TIMESTAMP	
	BIT(n), BIT VARYING(n)	RAW(n : max = 255), LONG RAW (binaire jusqu'à 2G), BLOB (binaire jusqu'à 4G)
		BFILE (pointeur à un fichier externe)

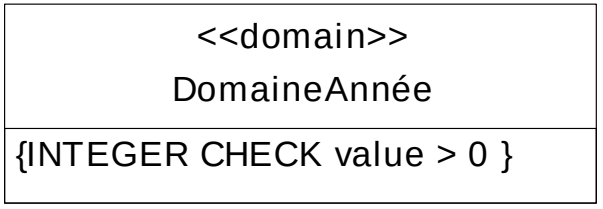
Types de données déclarés

- Domaine pas toujours supporté par le dialecte SQL
  - La classe stéréotypée ««datatype»» sera traduit en domaine au sens relationnel
  - Ex. Année parution sera du type : DomaineAnnée dans la table Livre









TYPES ÉNUMÉRÉS

- Petit domaine invariant
  - création d ’un domaine VARCHAR + CHECK

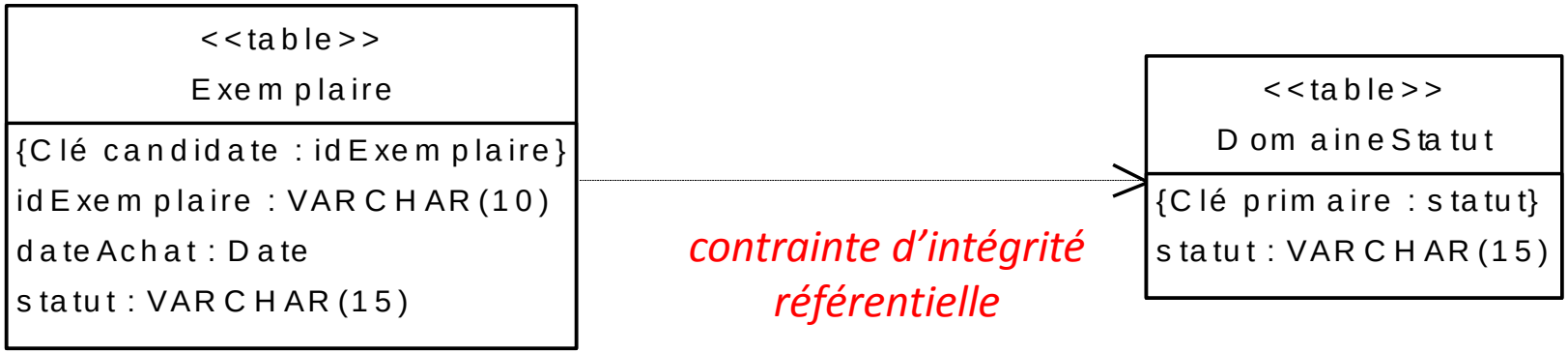
Exemplaire
{UNIQUE: idExemplaire} idExemplaire : String dateAchat : Date statut : enum(prêté, disponible, retiré)

<<table>> Exemplaire
{Clé candidate : idExemplaire} idExemplaire : VARCHAR(10) dateAchat : Date statut : DomaineStatut

<<domain>> DomaineStatut
{VARCHAR(15) CHECK value IN ('prêté','disponible','retiré')}

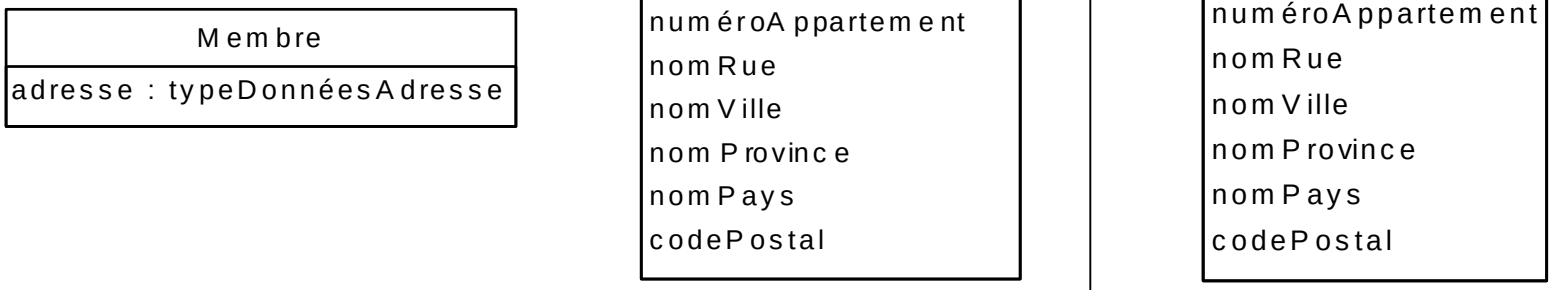
Création d ’une table

- Gros domaine ou extensible
  - création d ’une table à part
  - utilisé comme liste de valeurs
  - introduction d ’une clé primaire artificielle ?



TYPES COMPLEXES

Option 1. Représentation explicite des attributs du type complexe

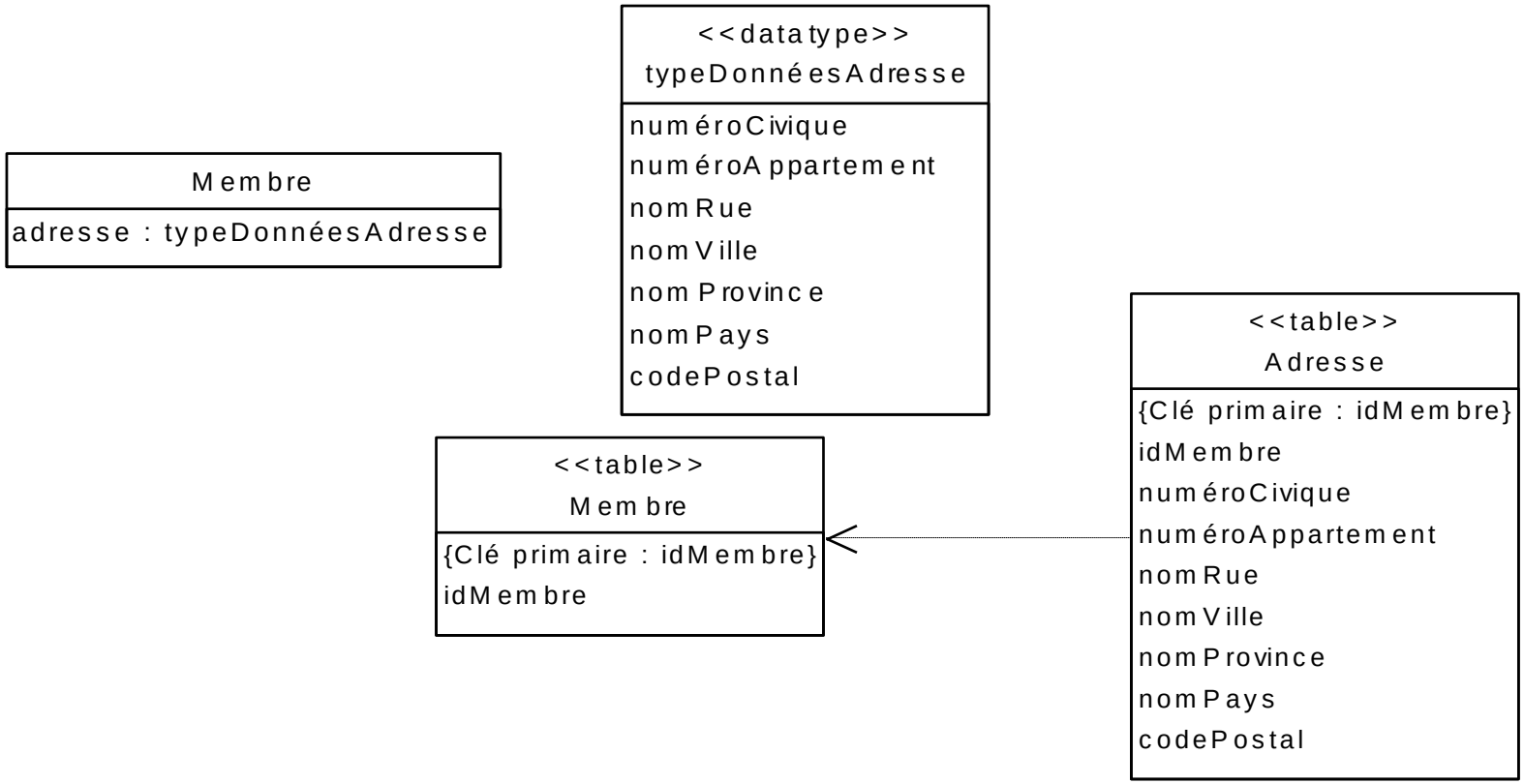


Les attributs représentent les colonnes de la table

TYPES COMPLEXES

Option 2. Création d'une nouvelle table

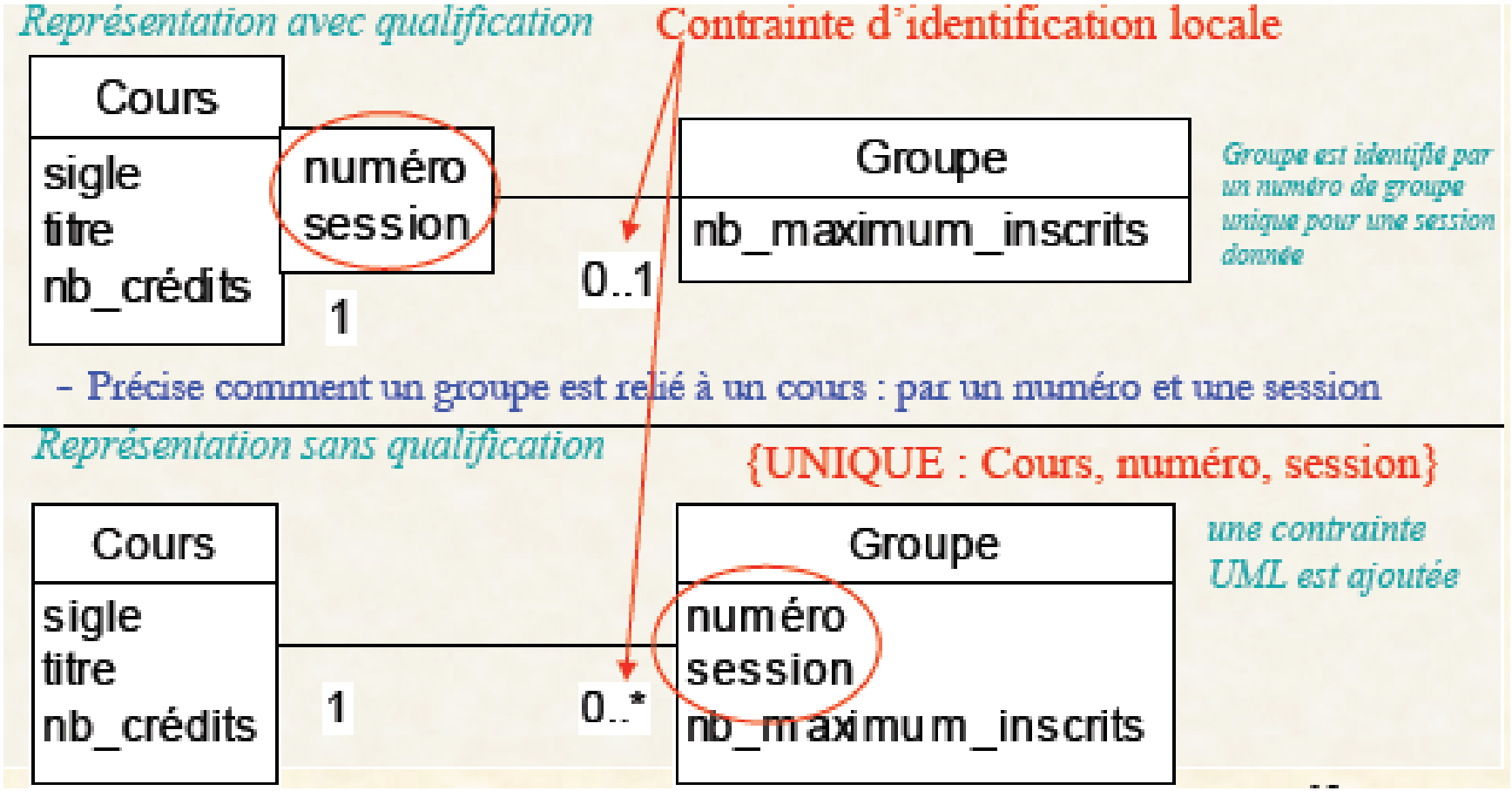
- N.B. Pas de partage de la même adresse !



- nouvelle table Adresse , même clé primaire dans les deux tables, contrainte
- d'intégrité référentielle :: représentation plus coûteuse

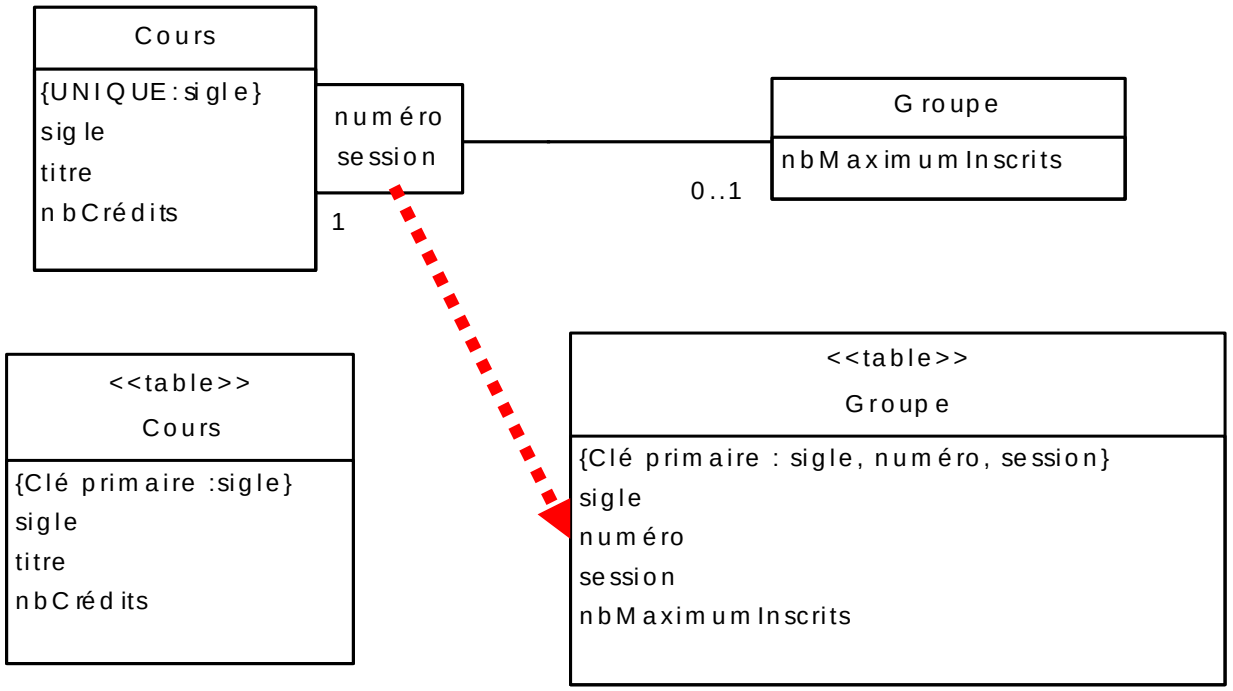
QUALIFICATEUR

- Partition des objets associés par rapport aux valeurs d'un ensemble d'attributs



QUALIFICATEUR

- Colonne dans la table du rôle opposé

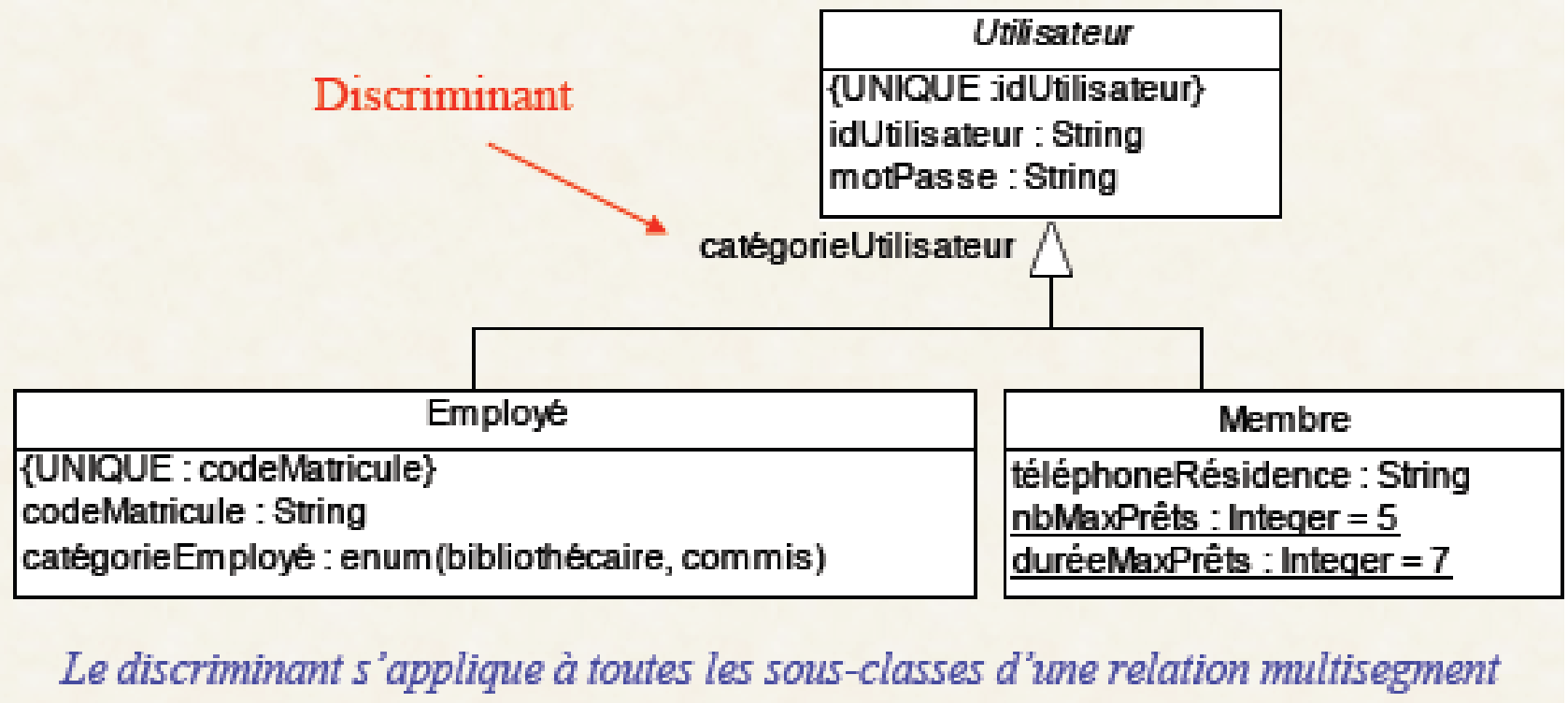


Le qualificateur devient une composante de la clé primaire de la table opposée

DISCRIMINANT

- Le discriminant est un attribut indiquant le critère de partitionnement des objets de la superclasse en sous-classes

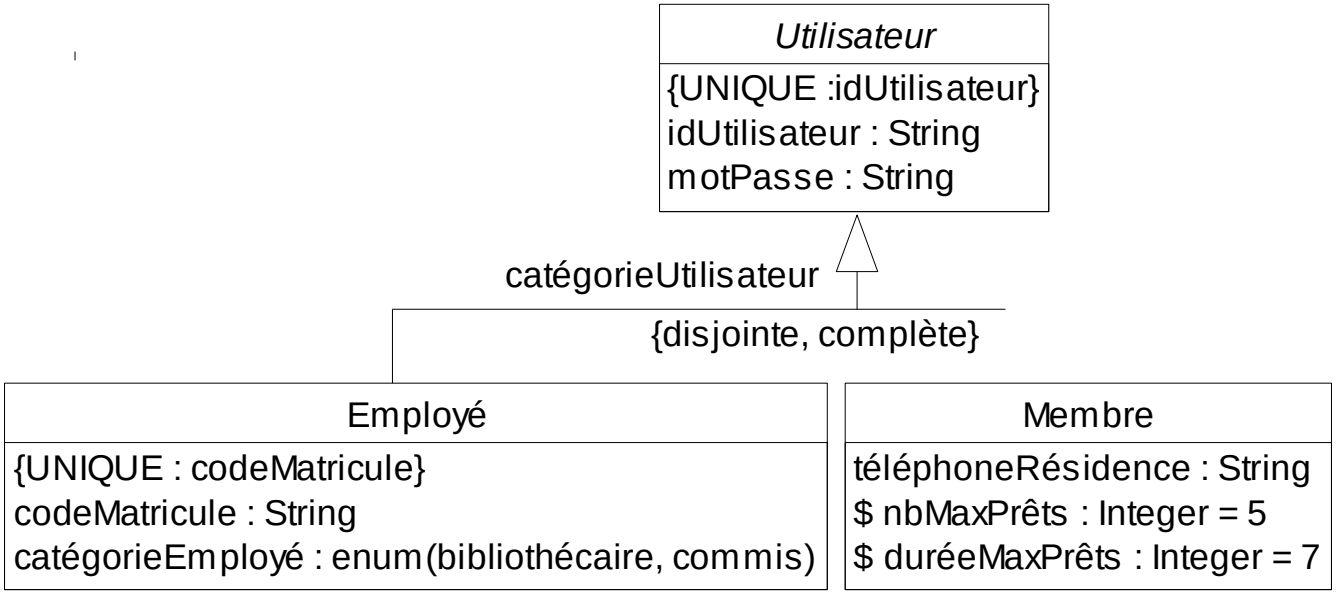
indique à quelle sous-classe l'objet appartient?



Le discriminant s'applique à toutes les sous-classes d'une relation multisegment

DISCRIMINANT

- Redondant ?



<<table>> Utilisateur
{Clé candidate : idUtilisateur}
idUtilisateur : VARCHAR(10)
motPasse : VARCHAR(10)
catégorieUtilisateur : DomaineCatégorieUtilisateur

<<domain>> DomaineCatégorieUtilisateur
{VARCHAR(14) CHECK value IN ('employé, 'membre')}

- **Le discriminant est traduit par une colonne dans la table parent**
- **Un domaine est créé avec des valeurs (noms des classes spécialisées correspndantes)**

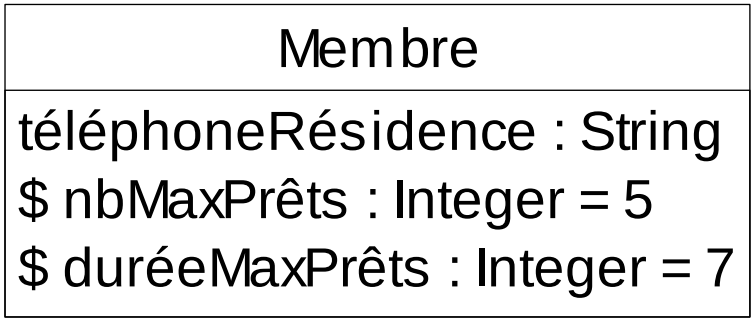
ATTRIBUT MULTIVALUÉ

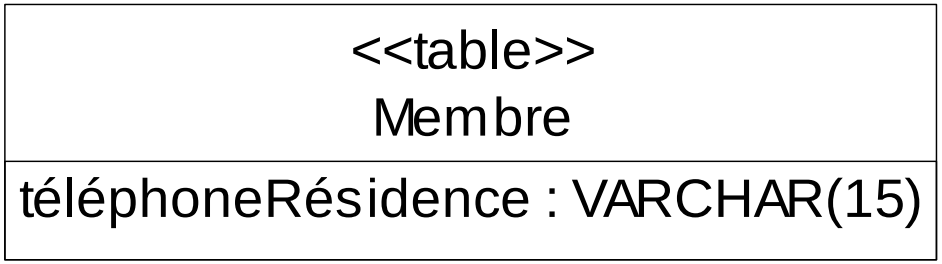
- Table à part
  - clé étrangère + colonne pour l ’attribut
- Petit tableau de taille fixe
  - n colonnes (valeurs nulles)
- Encodage
  - invisible au SGBD
- Oracle8
  - VARRAY, NESTED TABLE

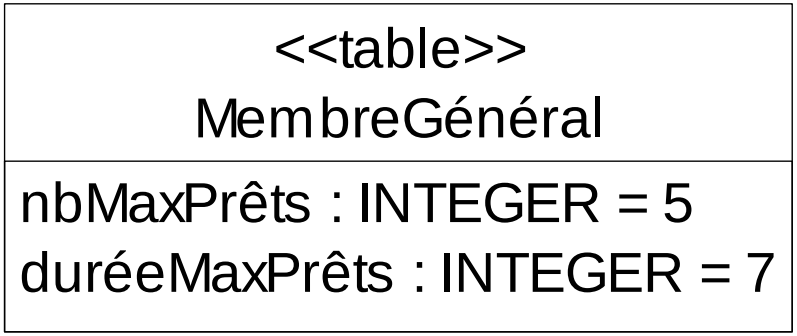
COMPRESSION DES VALEURS DES ATTRIBUTS DE GRANDE TAILLE

- Supporté par le SGBD ?
- Multimédia

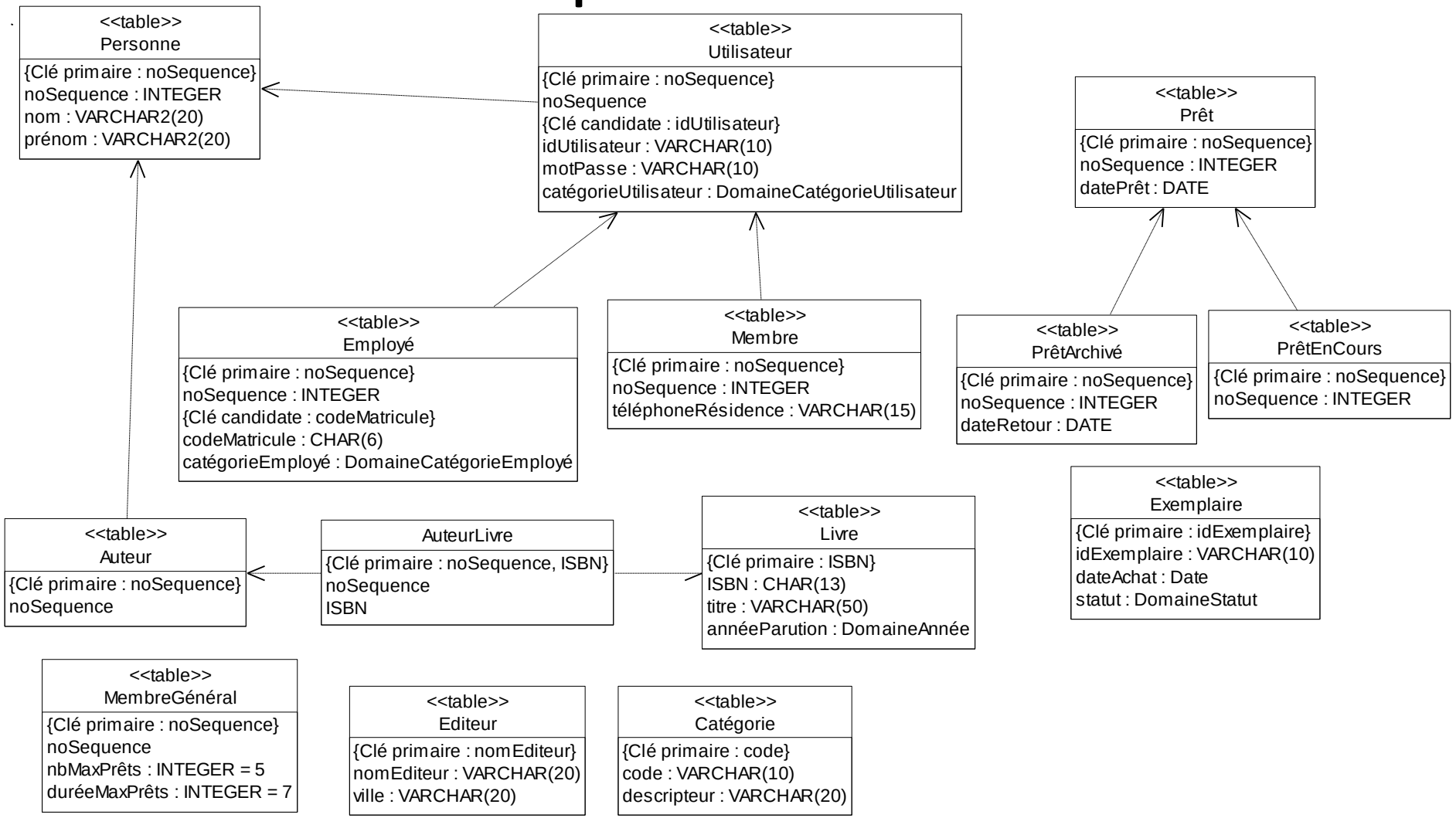
CRÉATION DE TABLES SUPPLÉMENTAIRES POUR LES ATTRIBUTS DE CLASSE



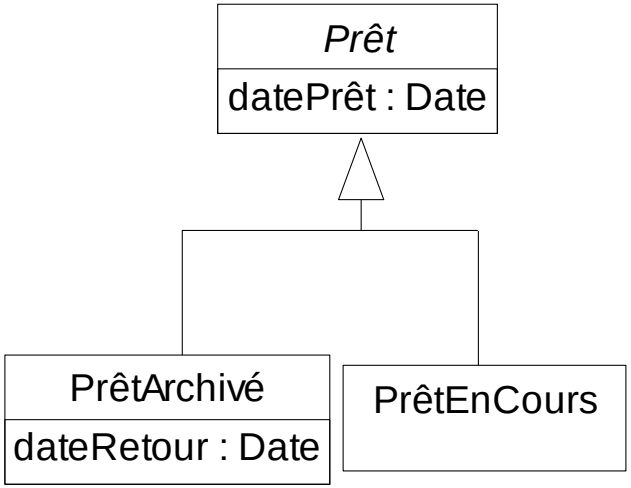


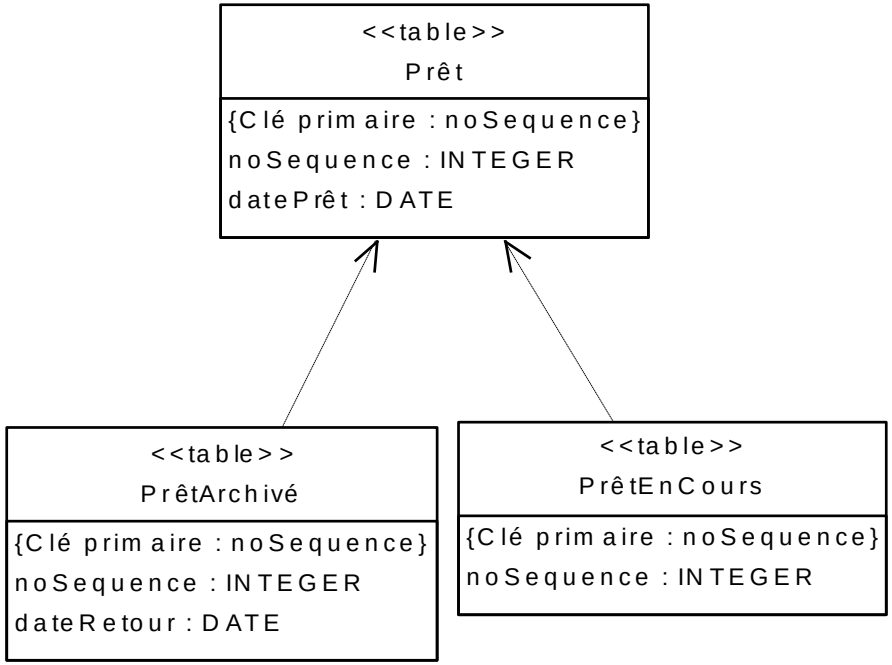


Réalisation de l'identité par les clés primaires



CRÉATION D'UNE CLÉ PRIMAIRE ARTIFICIELLE



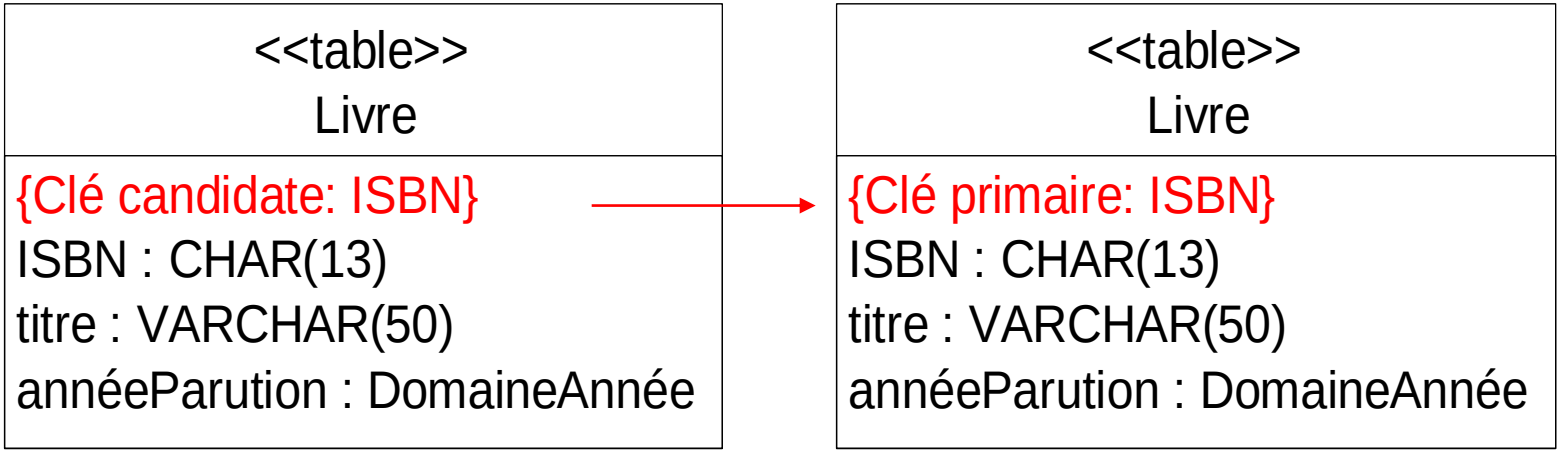


La clé primaire est introduite dans les tables de la spécialisation

- De manière systématique ?
- Mécanisme de SEQUENCE Oracle

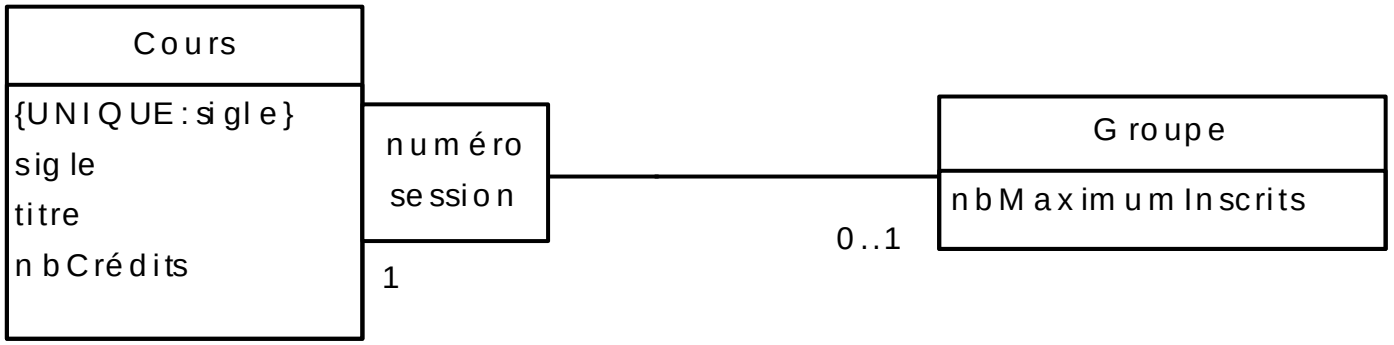
UTILISATION D'UN IDENTIFIANT NATUREL COMME CLÉ PRIMAIRE

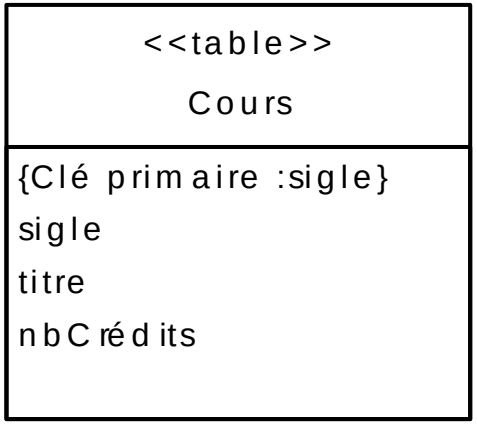
- Utilisation d'un identifiant naturel (UNIQUE)

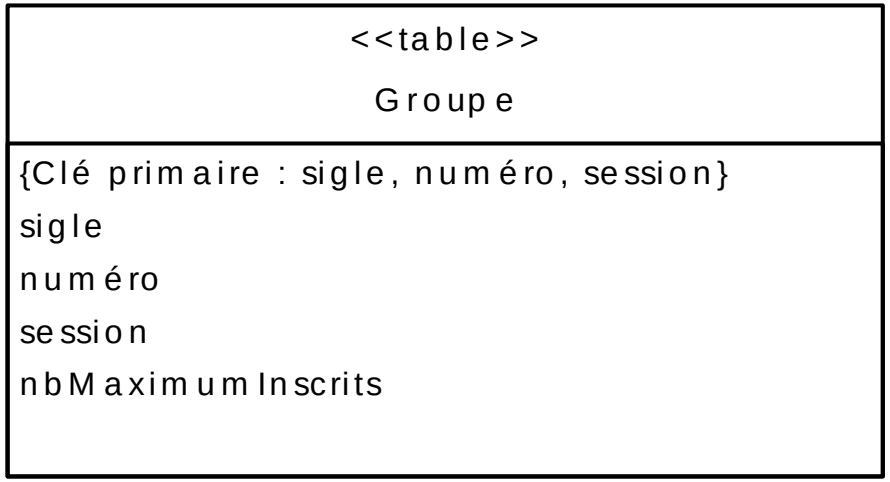


- Si identifiant naturel trop lourd
  - introduire clé primaire artificielle

Utilisation du qualificateur







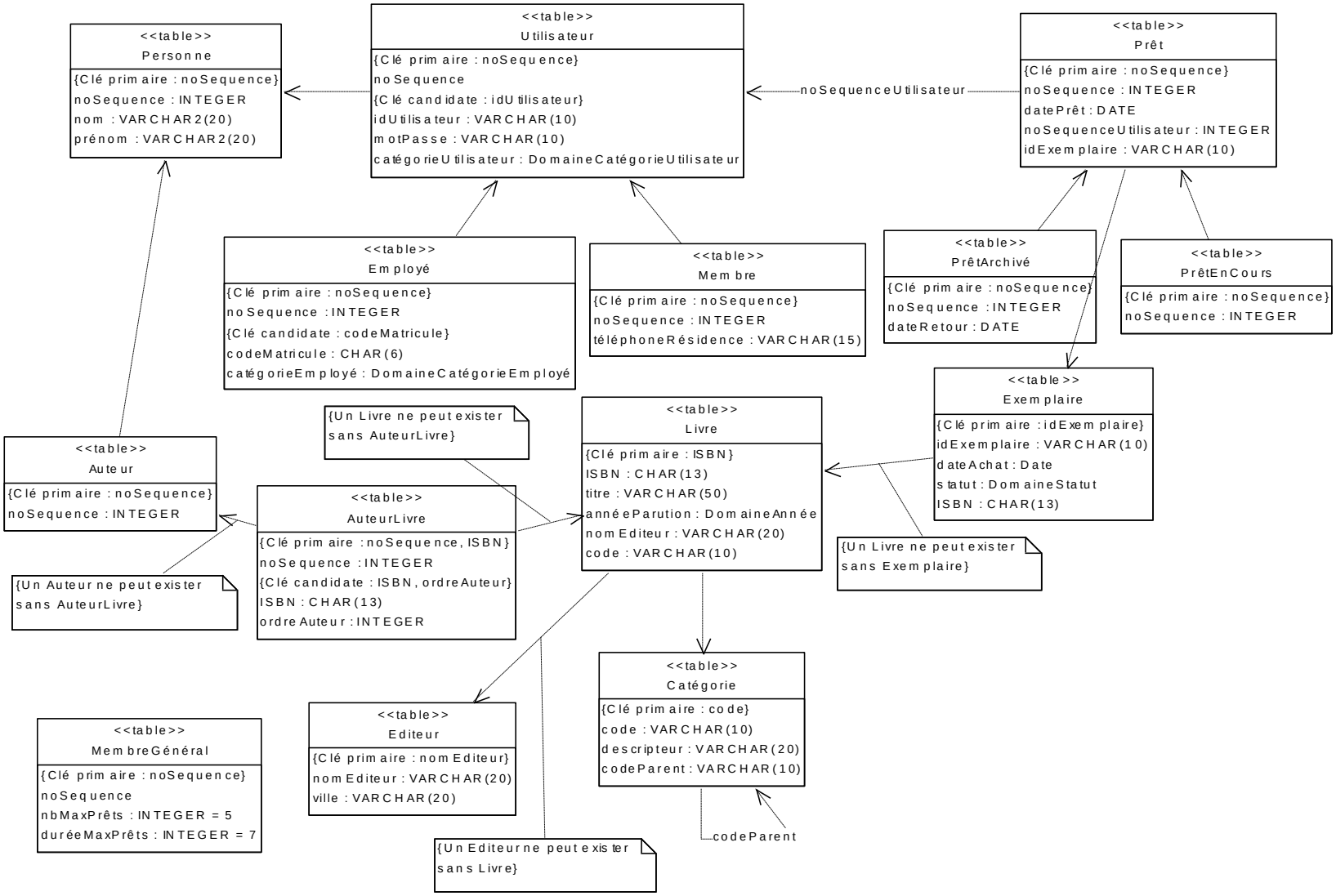
IDENTIFIANT NATUREL POUR UNE SPÉCIALISATION

- Éviter de changer de clé primaire dans le contexte d ’une hiérarchie de généralisation
- Si la table du parent est omise
  - considérer identifiant naturel de la spécialisation

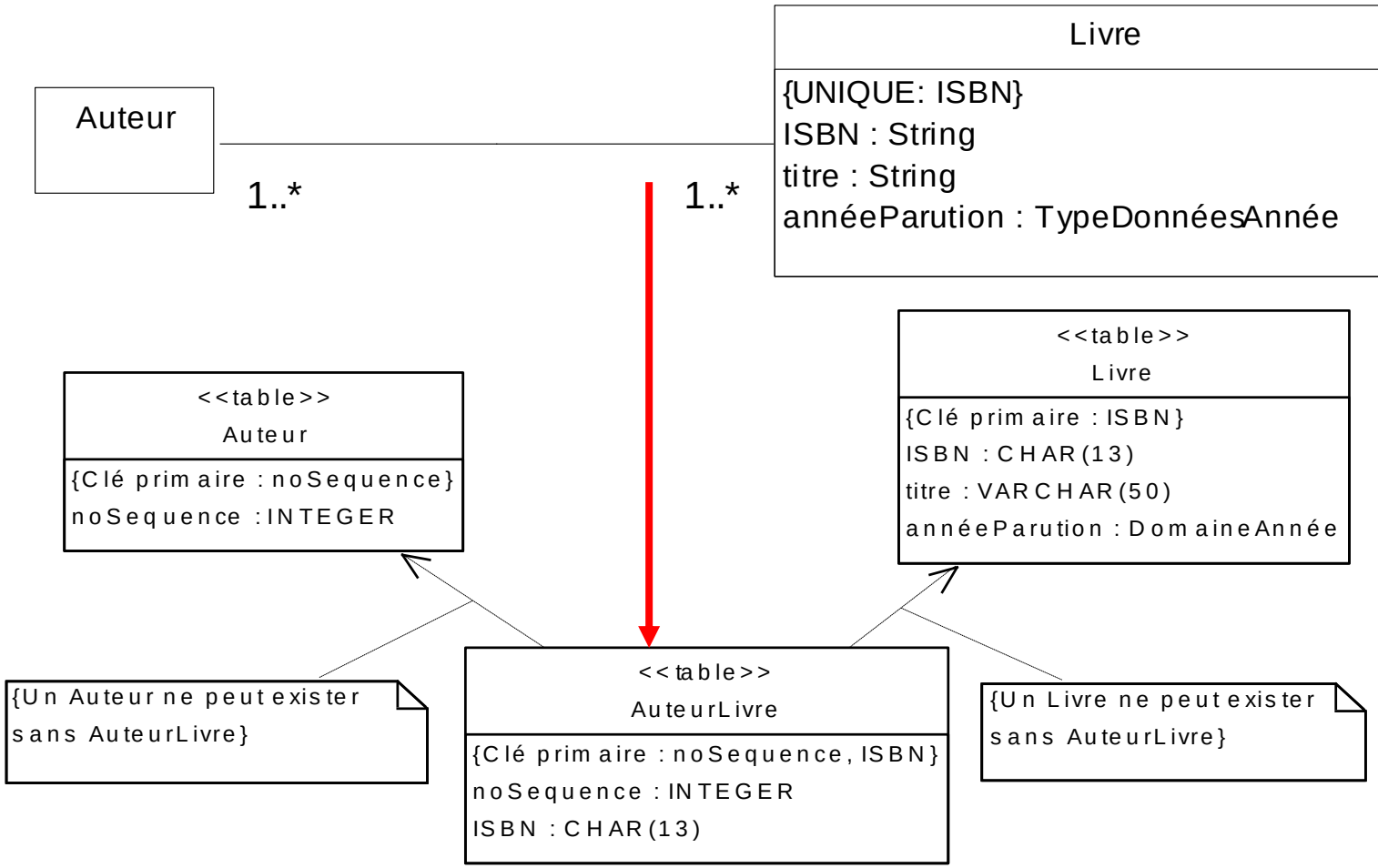
Traduction des associations

- À l ’aide des clés étrangères
  - permettent de représenter directement une association
    - un à un ou un à plusieurs
- introduction de nouvelles tables (d’intersection) pour les autres cas
  - Voir l’association entre les tables auteur et livre
  - Cas d’association binaires --- deux clés étrangères
  - Cas d’associations n-aires --- n clés étrangères

Traduction des associations

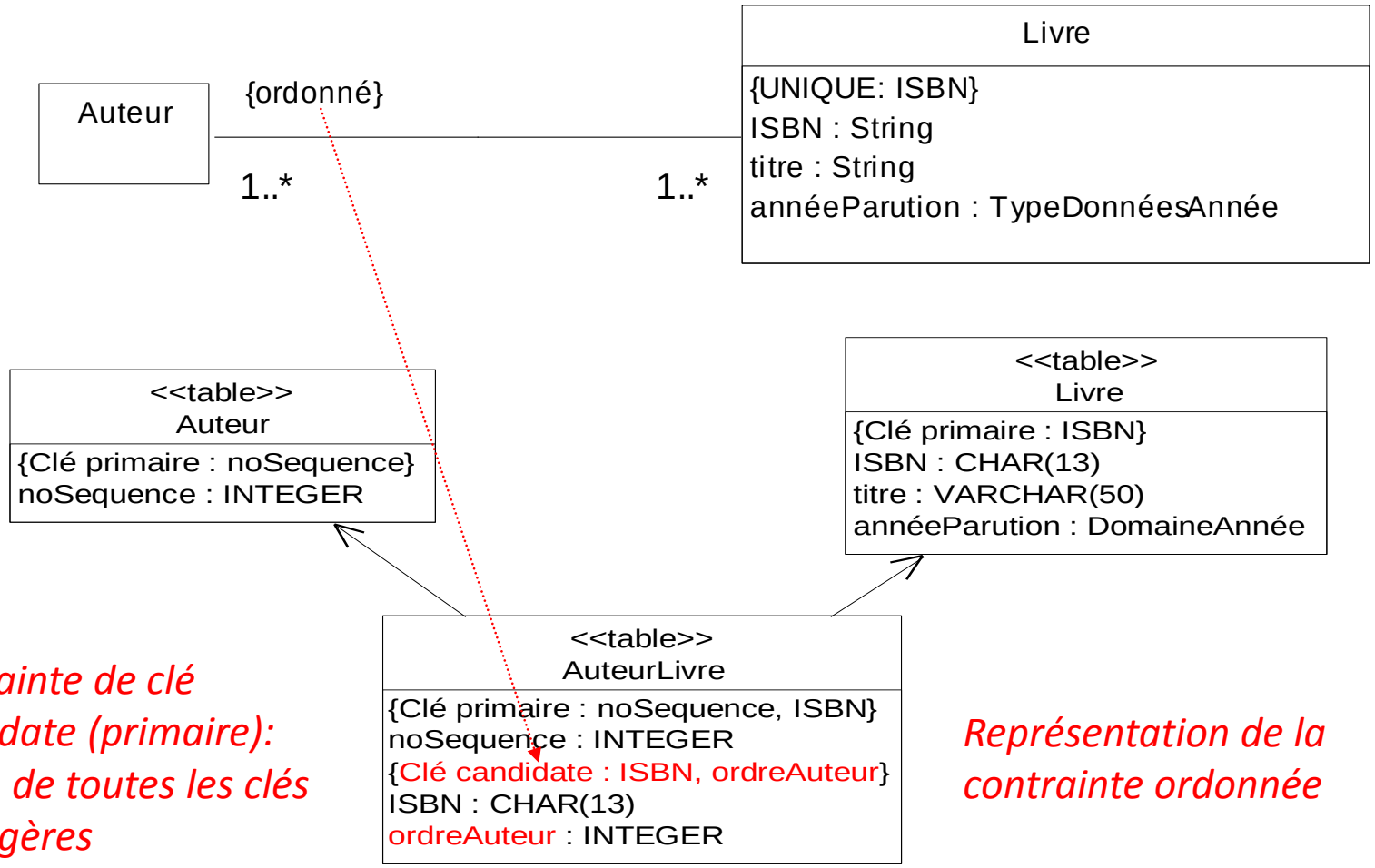


CAS GÉNÉRAL : TRADUCTION D'UNE ASSOCIATION PAR UNE TABLE



contrainte de clé candidate (primaire):
union de toutes les clés étrangères

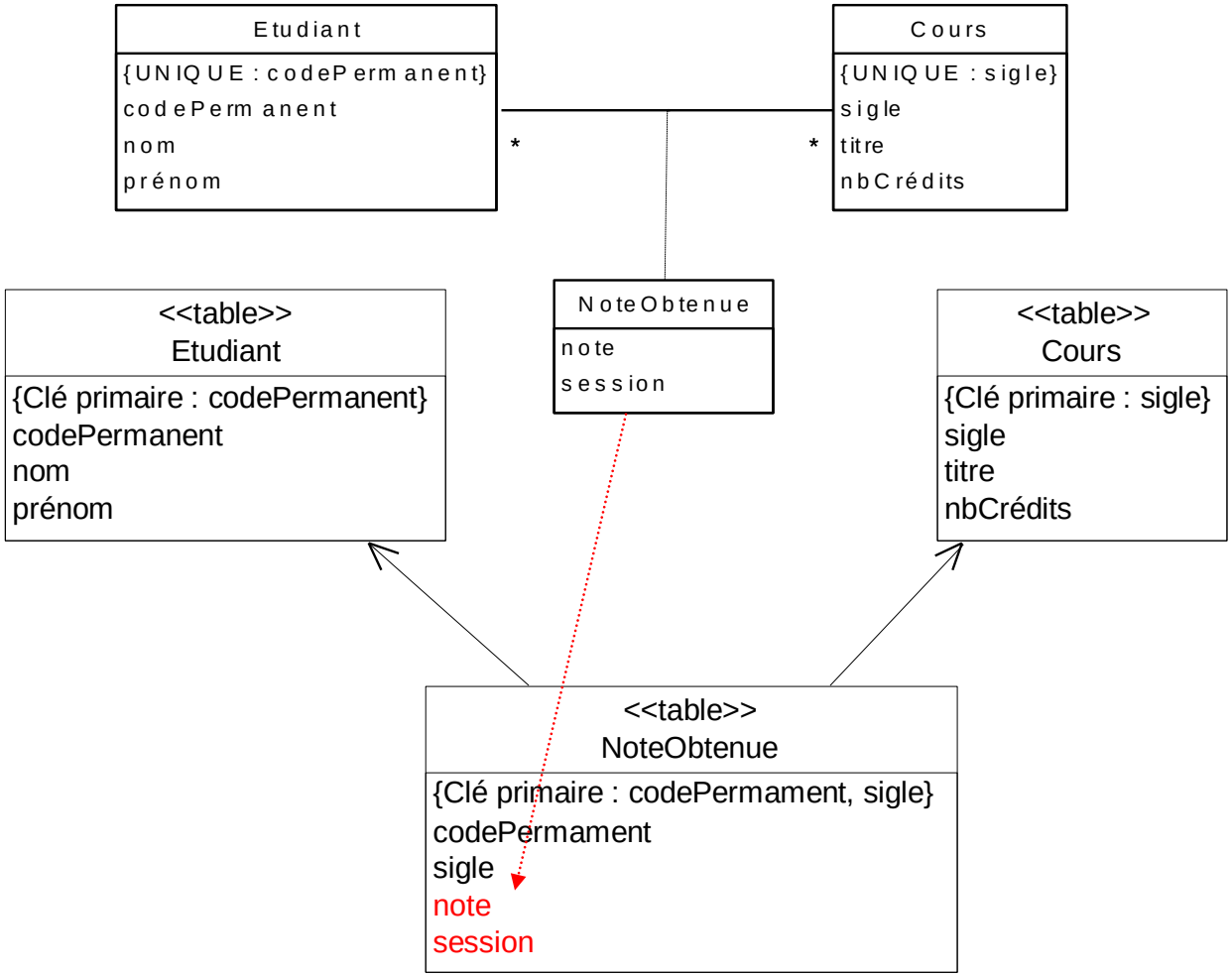
TRADUCTION D'UN RÔLE ORDONNÉ POUR UNE ASSOCIATION



contrainte de clé candidate (primaire): union de toutes les clés étrangères

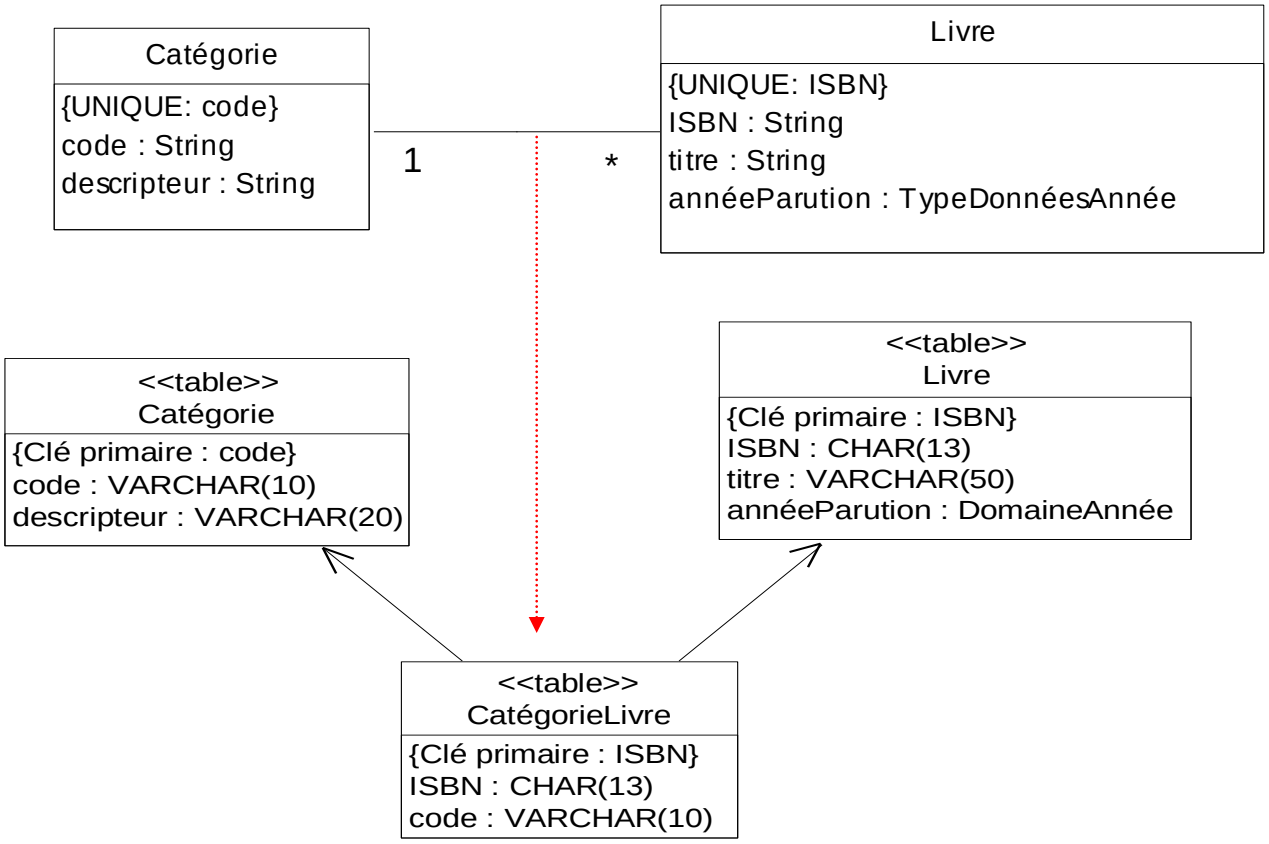
Représentation de la contrainte ordonnée

CLASSE ASSOCIATIVE



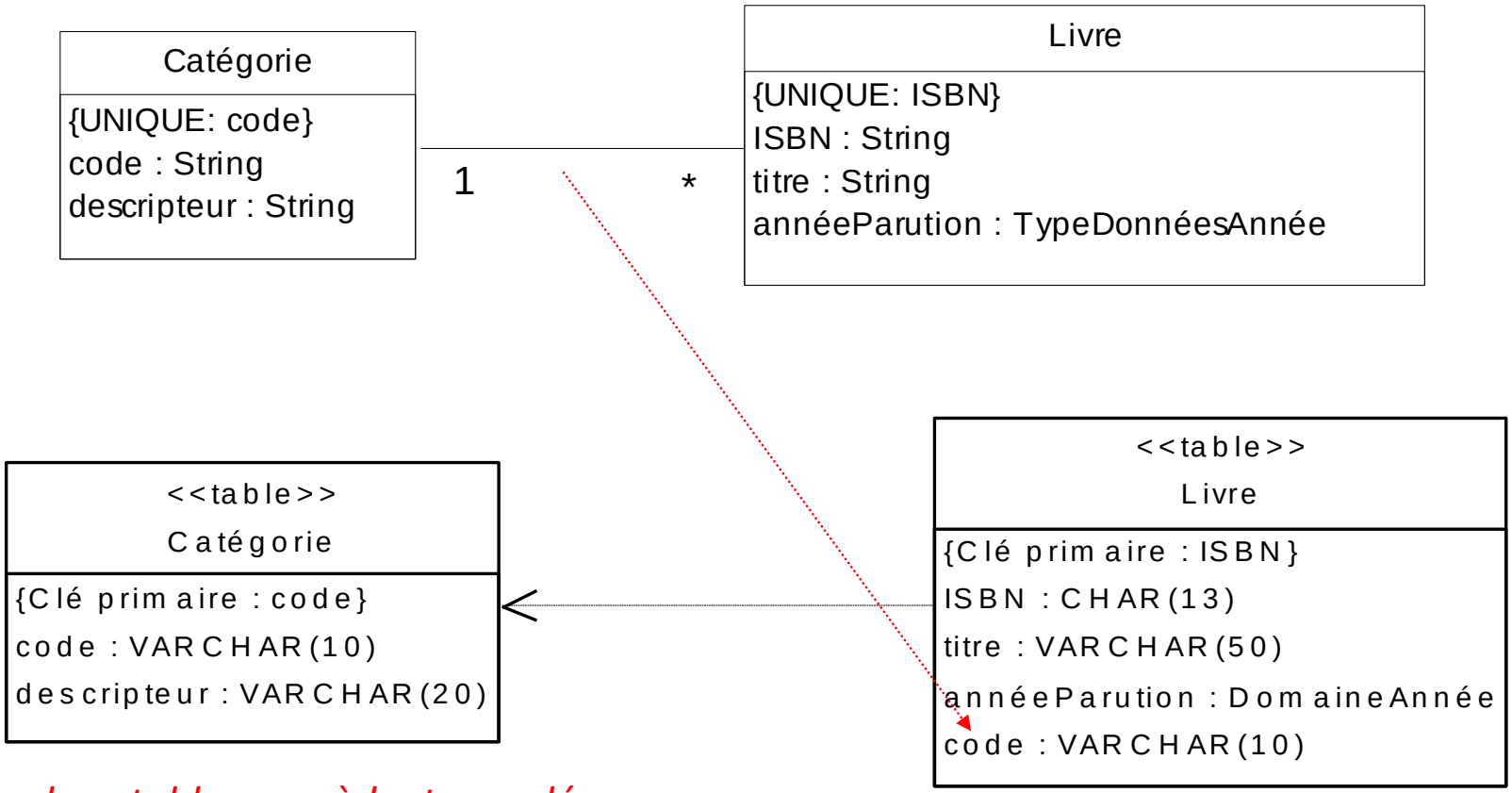
CAS UN À PLUSIEURS

- Option 1. Traduction par une table ?



Traduction par l'ajout d'une clé étrangère à privilégier

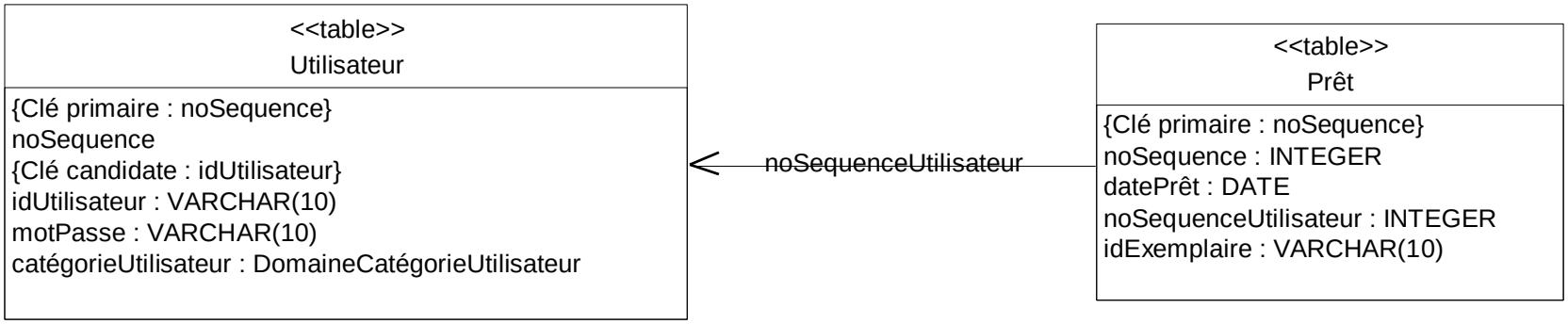
- Option 2. Navigation plus performante



Lorsque deux tables possèdent une clé commune, on considère de les fusionner

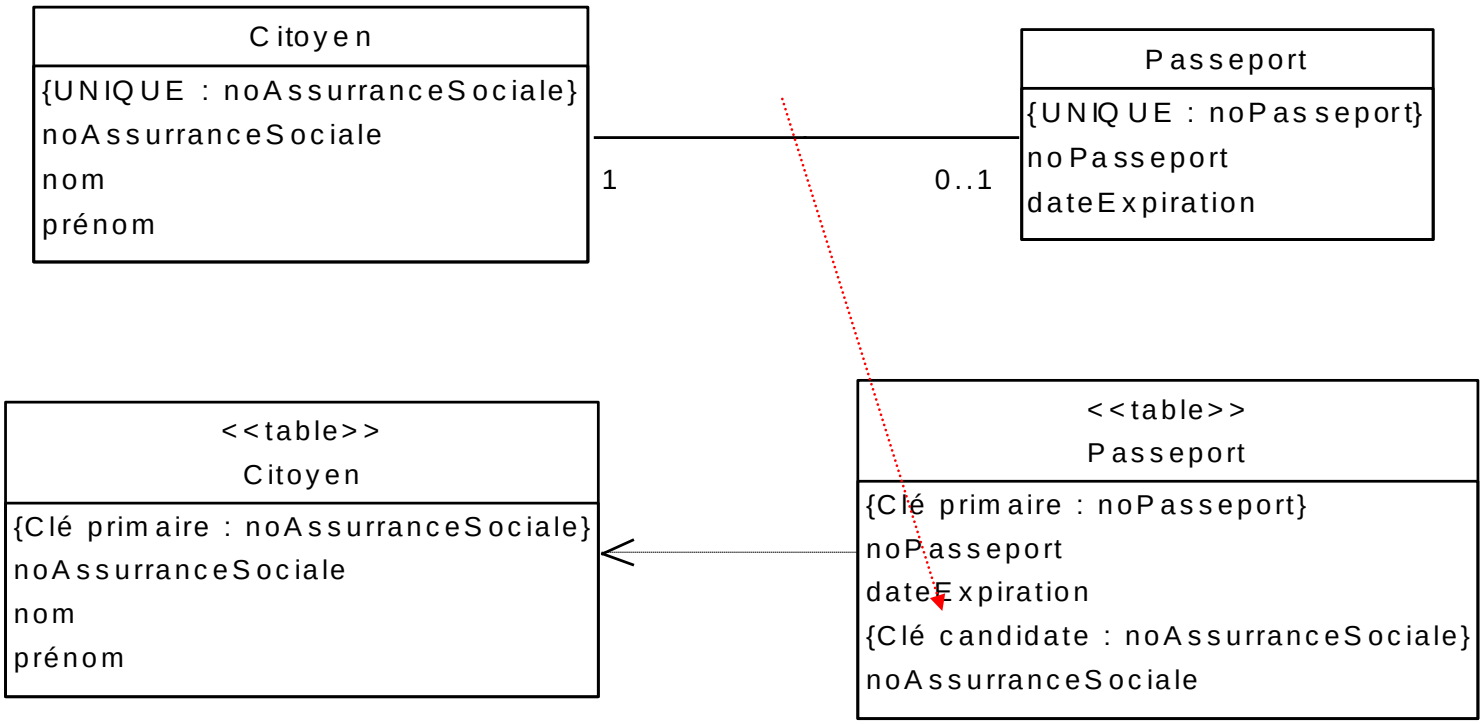
Renommer la clé étrangère au besoin

afin d'éviter les ambiguïtés



CAS UN À UN

- Option 1. Une clé étrangère (du côté obligatoire)

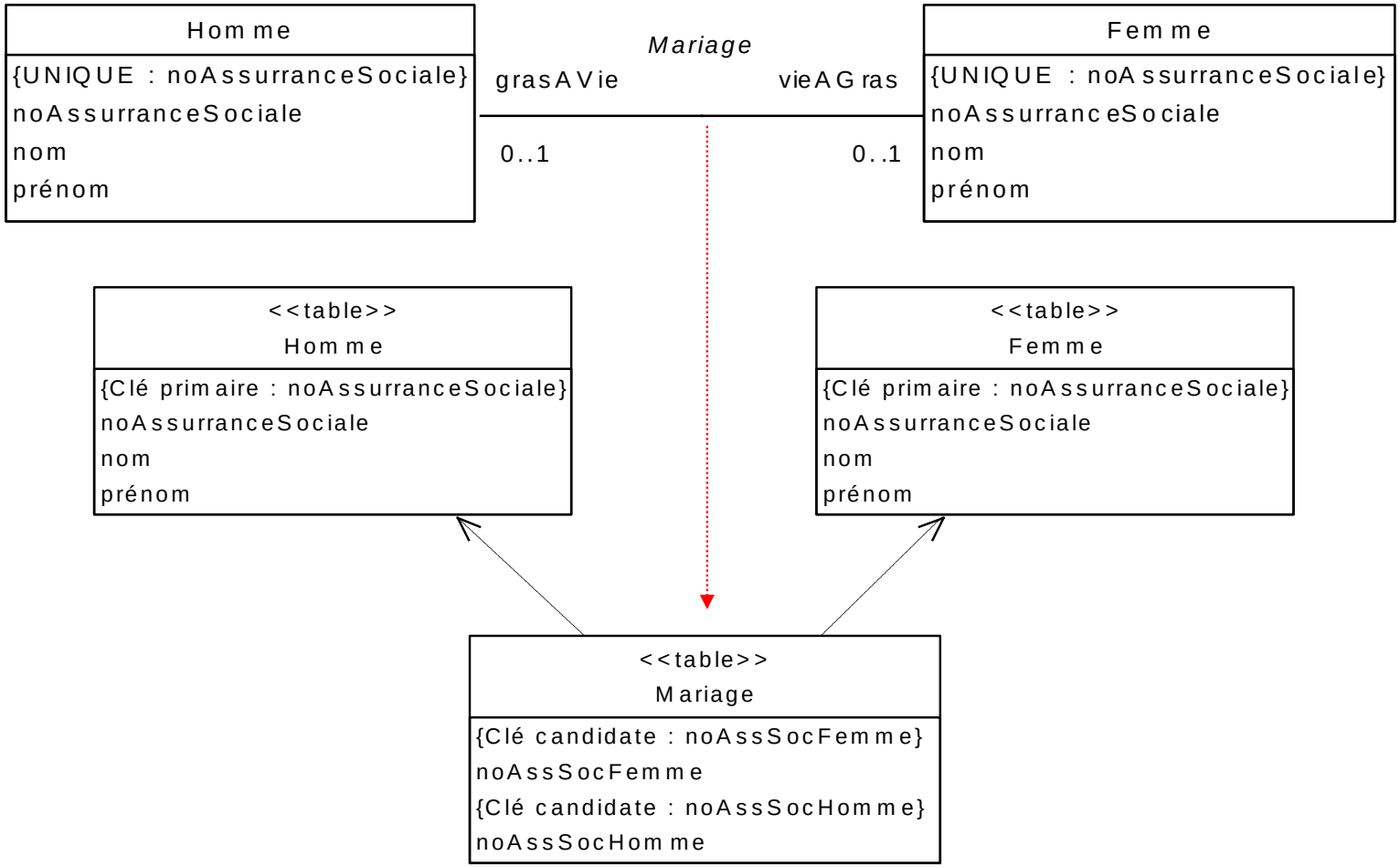


Une contrainte de clé candidate doit être spécifiée pour la clé étrangère

Une nouvelle table

Option 2.

Lorsque les deux rôle sont optionnels et que les valeurs nulles résultant de clés étrangères sont fréquentes

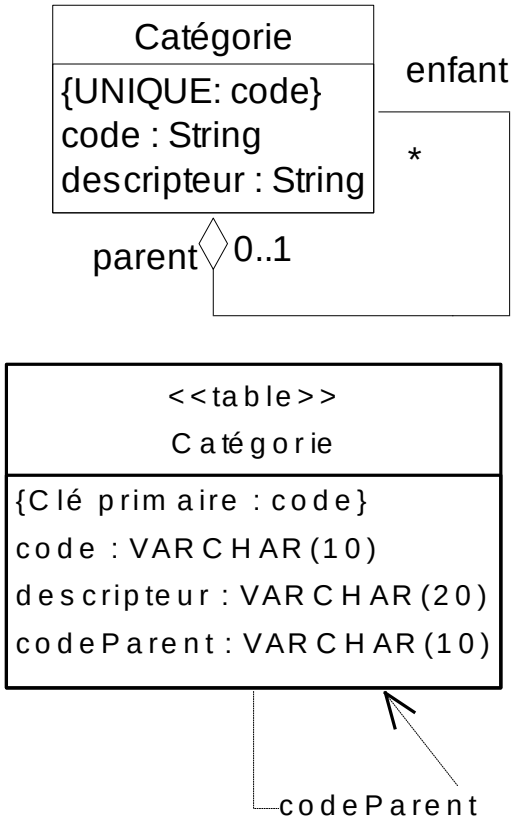


Fusion

- Option 3.
  - Cas 1-1 obligatoire
    - Considérer la fusion des deux tables

CAS DE L'AGRÉGATION

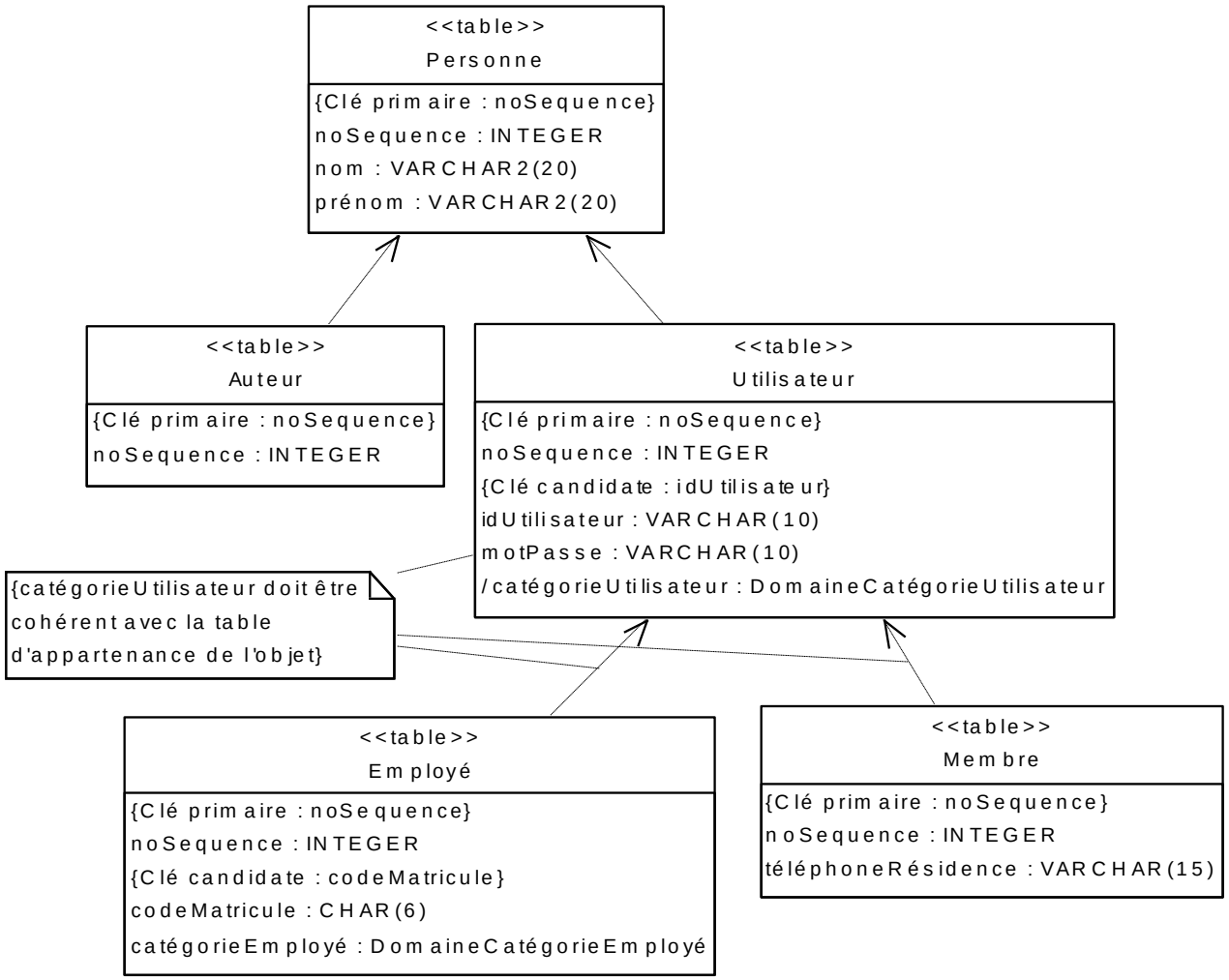
- Comme une association normale



Traduction des relations de généralisation/spécialisation

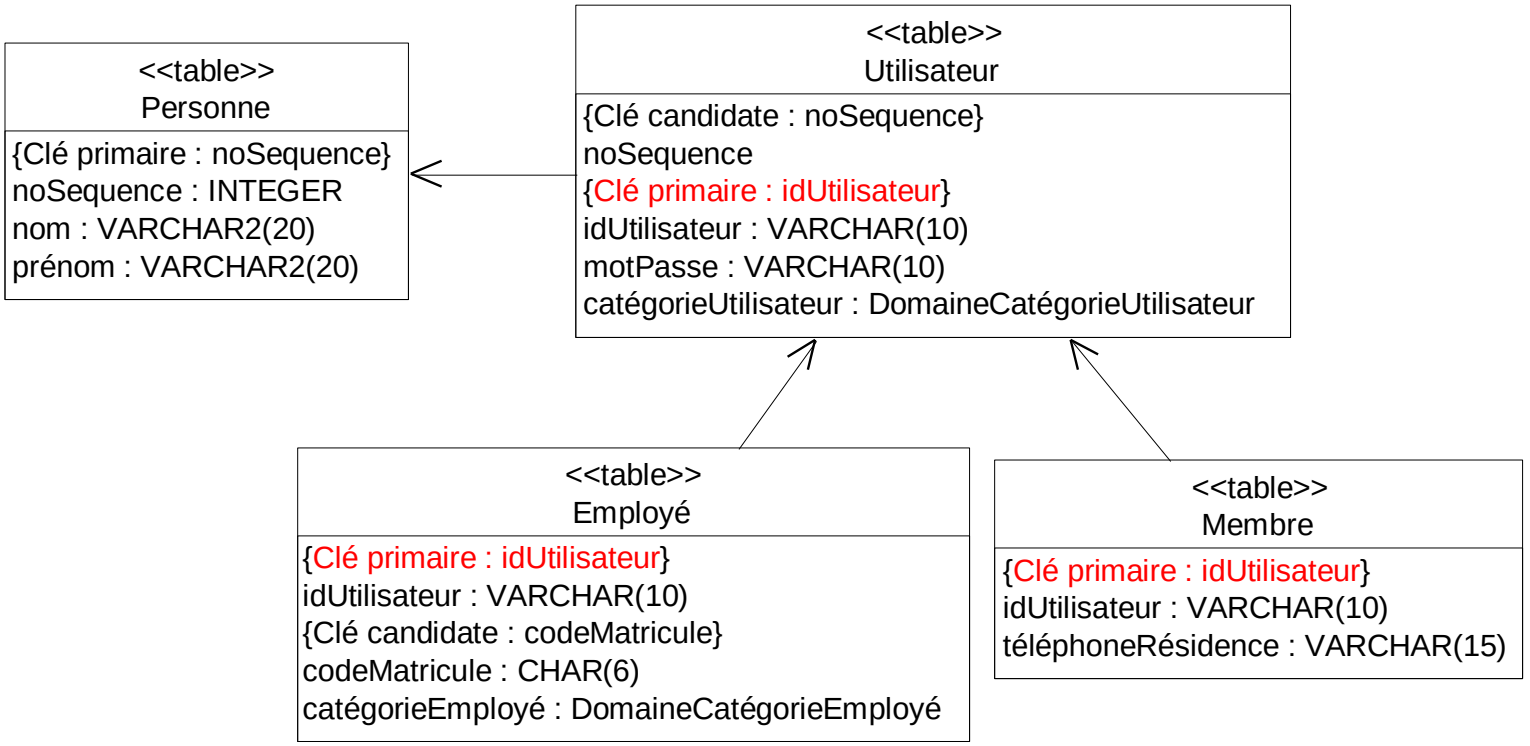
- Délégation
- Fusion
- Concaténation

DÉLÉGATION

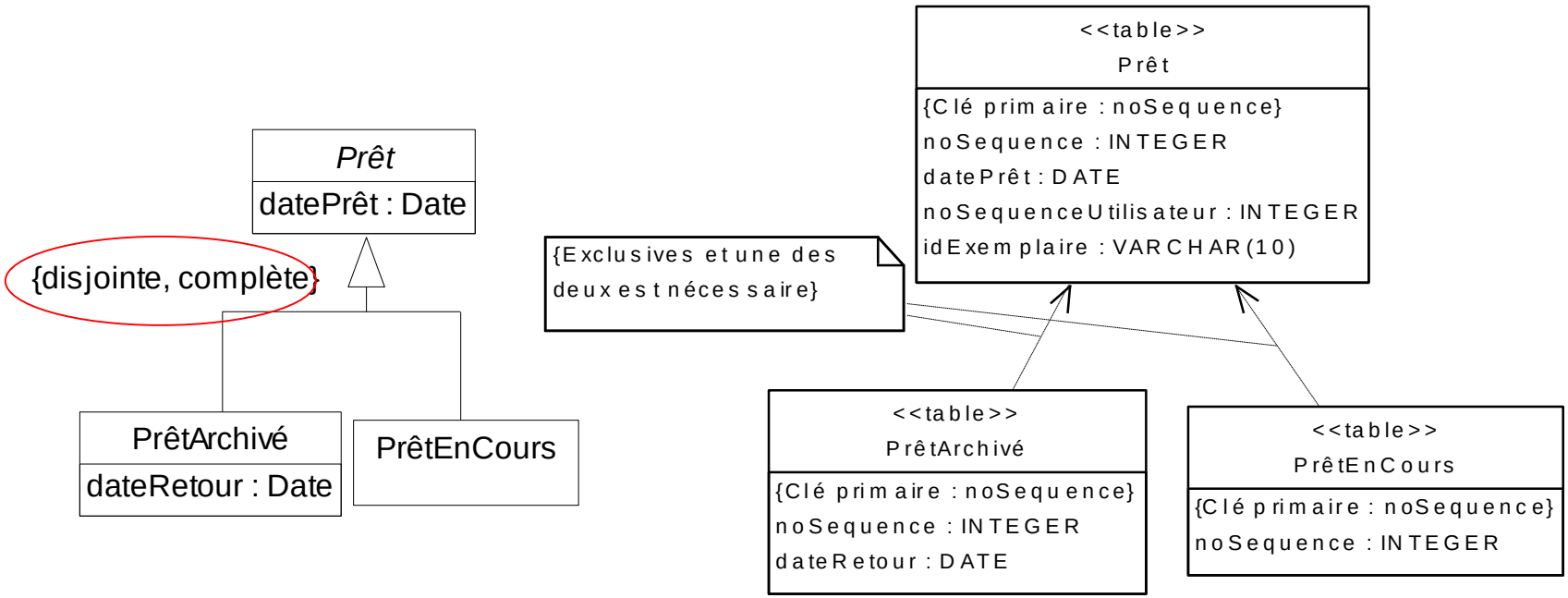

<<table>>
Personne
{Clé primaire : noSequence}
noSequence : INTEGER
nom : VARCHAR2(20)
prénom : VARCHAR2(20)
<<table>>
Auteur
{Clé primaire : noSequence}
noSequence : INTEGER
<<table>>
Utilisateur
{Clé primaire : noSequence}
noSequence : INTEGER
{Clé candidate : idUtilisateur}
idUtilisateur : VARCHAR(10)
motPasse : VARCHAR(10)
/catégorieUtilisateur : DomaineCatégorieUtilisateur
{catégorieUtilisateur doit être cohérent avec la table d'appartenance de l'objet}
<<table>>
Employé
{Clé primaire : noSequence}
noSequence : INTEGER
{Clé candidate : codeMatricule}
codeMatricule : CHAR(6)
catégorieEmployé : DomaineCatégorieEmployé
<<table>>
Membre
{Clé primaire : noSequence}
noSequence : INTEGER
téléphoneRésidence : VARCHAR(15)

Changement de clé primaire ?

Extraire le téléphone résidence d'un Membre possédant l'idUtilisateur 12345

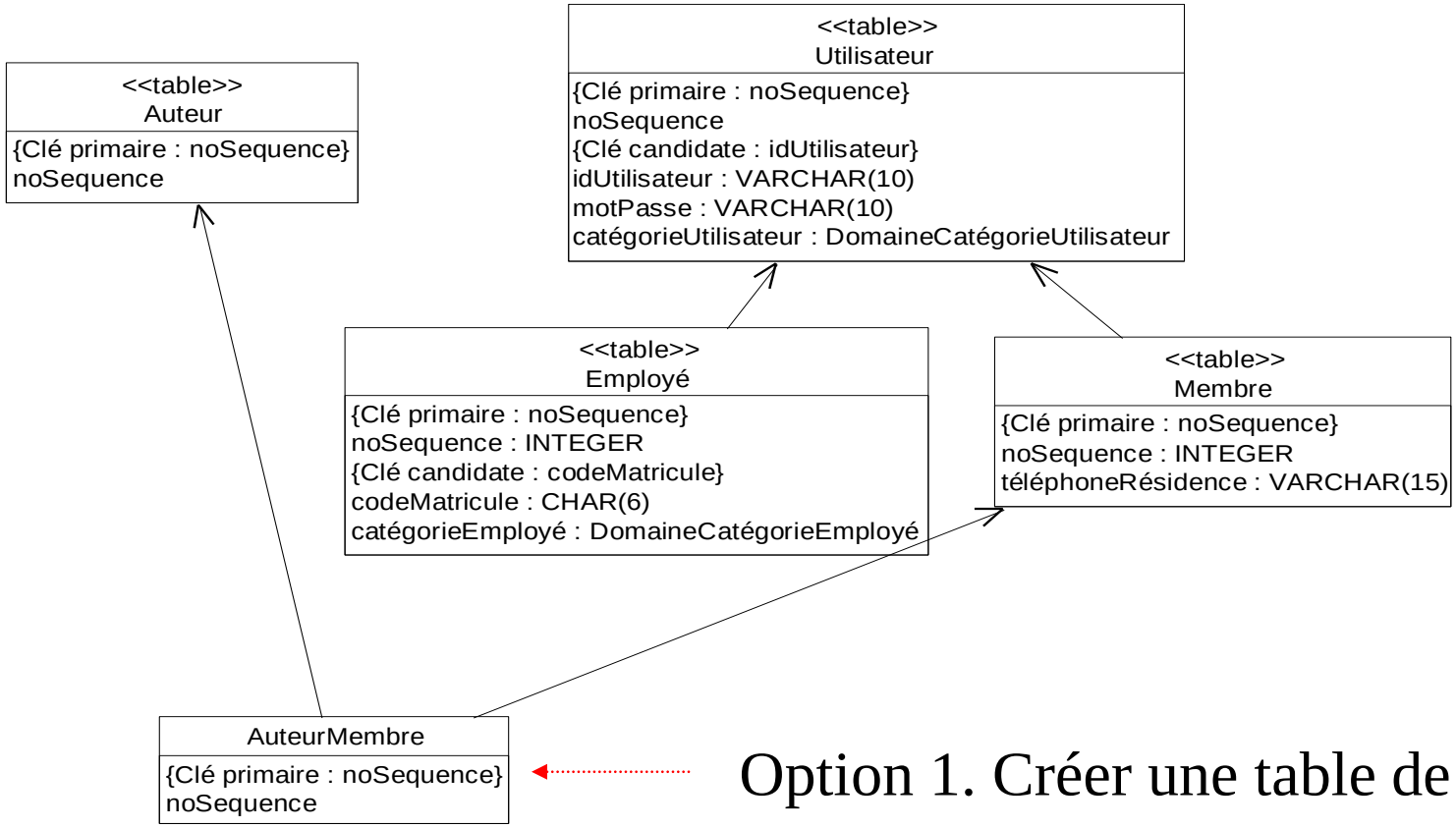


Traduction de la contrainte {disjointe, complète}



Cas de la multi-classification et de l'héritage multiple

Lorsque les relations de généralisations ne sont pas disjointes --- le même objet peut faire partie de plusieurs sous-classes en même temps (objet de jointure)



Option 1. Créer une table de jointure

Alternatives

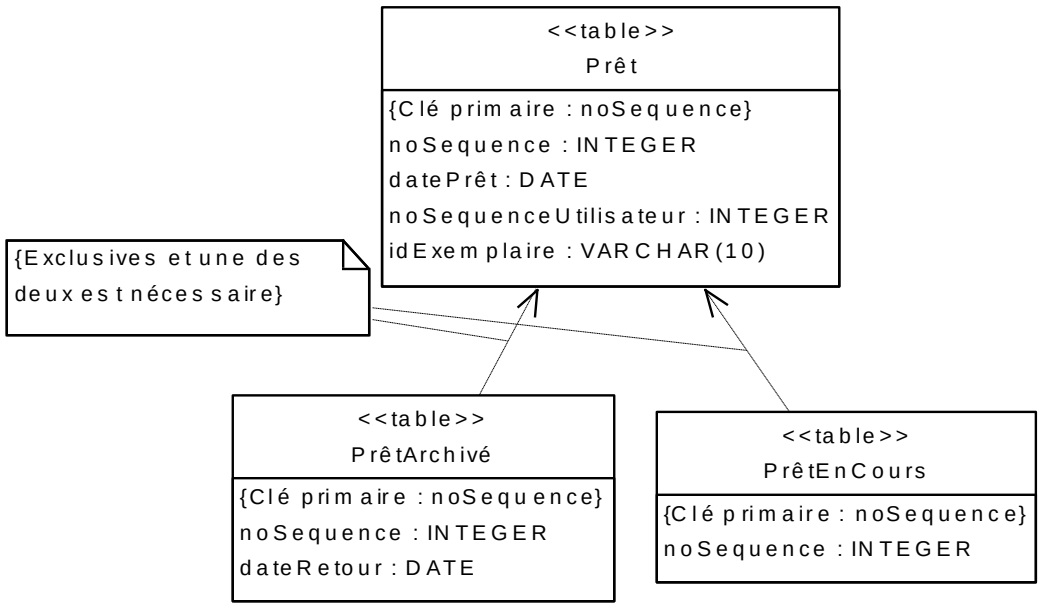
- Option 2. Identité commune sans table de jointure
  - vérifier la valeur de la clé primaire (no séquence) – doit être la même dans les deux tables
- Option 3. Identité séparée et redondance de données
  - considérer deux objets différents sans lien entre eux et dupliquer les données

ANALOGIE AVEC UNE ASSOCIATION UN À UN

- Clé étrangère
  - dans l ’enfant
  - dans le parent ???
- Fusion
  - vers le parent (approche par fusion)
  - vers l ’enfant (approche par concaténation)
- Nouvelle table ???

FUSION

Le résultat est une seule table parent contenant toutes les colonnes du parent et des enfants



<<table>>
Prêt

{Clé primaire : noSequence}
noSequence : INTEGER
datePrêt : DATE
noSequenceUtilisateur : INTEGER
idExemplaire : VARCHAR(10)
{catégoriePrêt = 'prêtEnCours' ssi dateRetour nulle}
dateRetour[0..1] : DATE
catégoriePrêt : DomaineCatégoriePrêt

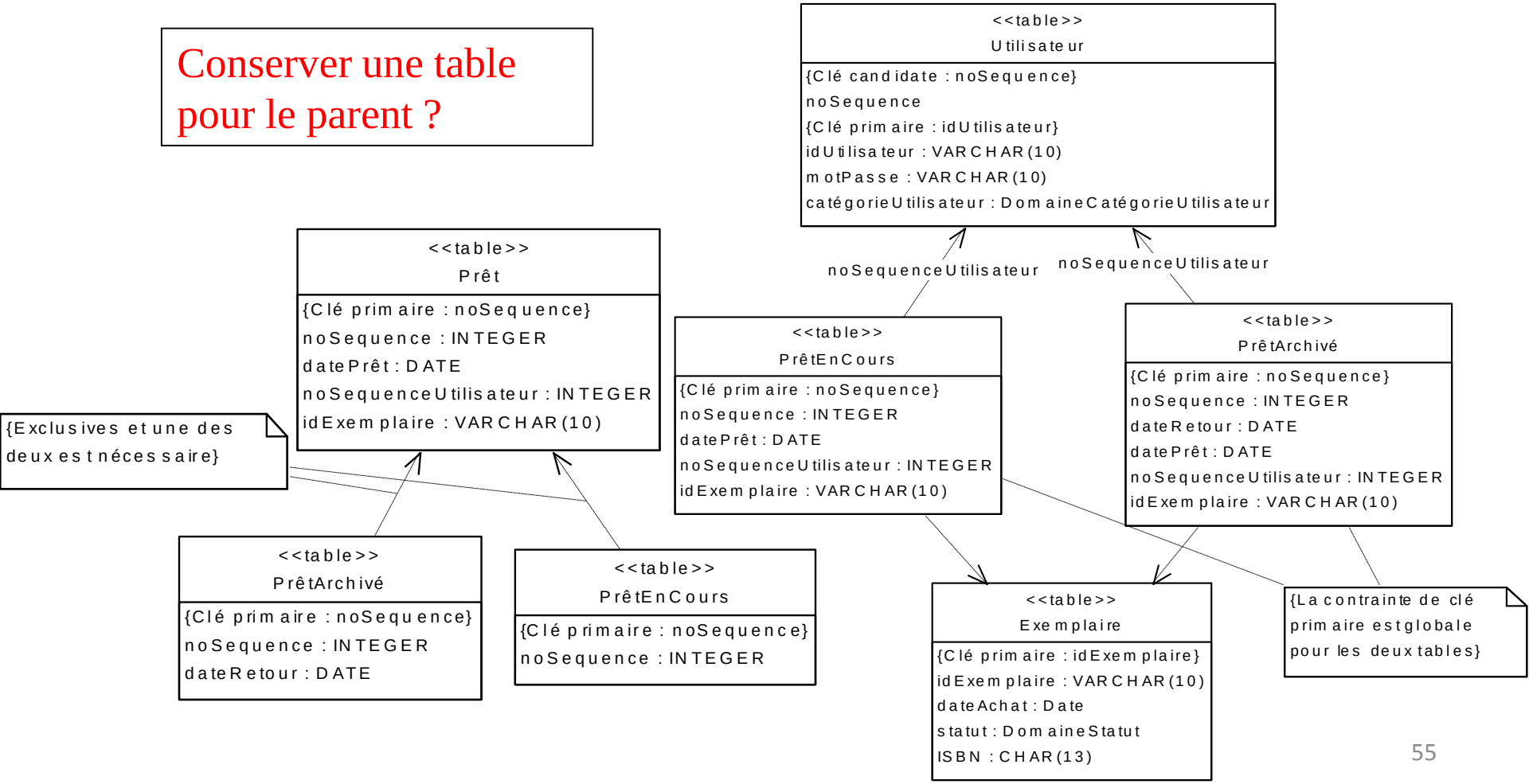
<<domain>>
DomaineCatégoriePrêt

{VARCHAR(15) CHECK valeur IN ('prêtEnCours', 'prêtArchivé')}

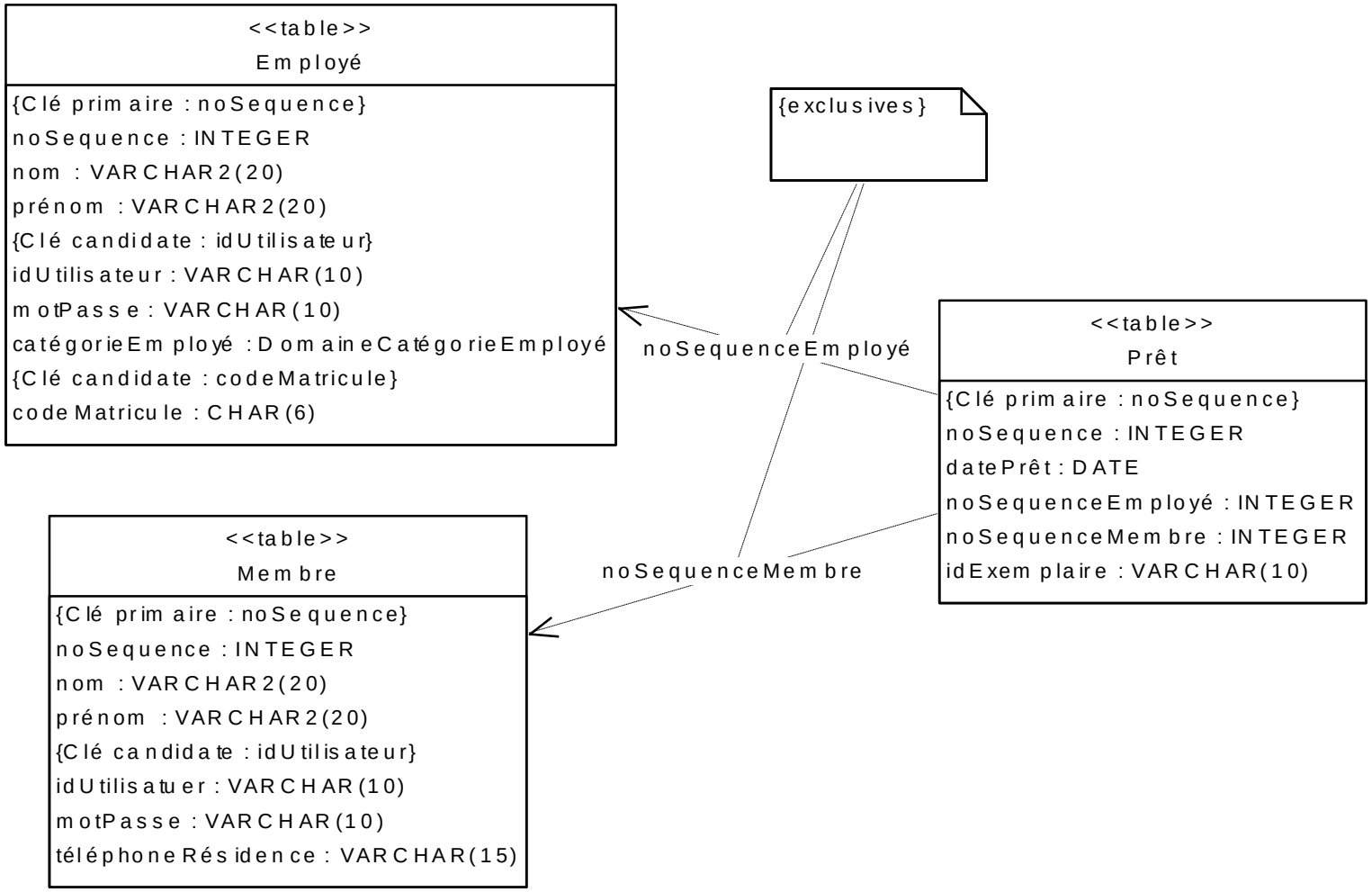
ajout d'un discriminant (colonne) et d'une contrainte pour déterminer la sous-classe d'appartenance

CONCATÉNATION

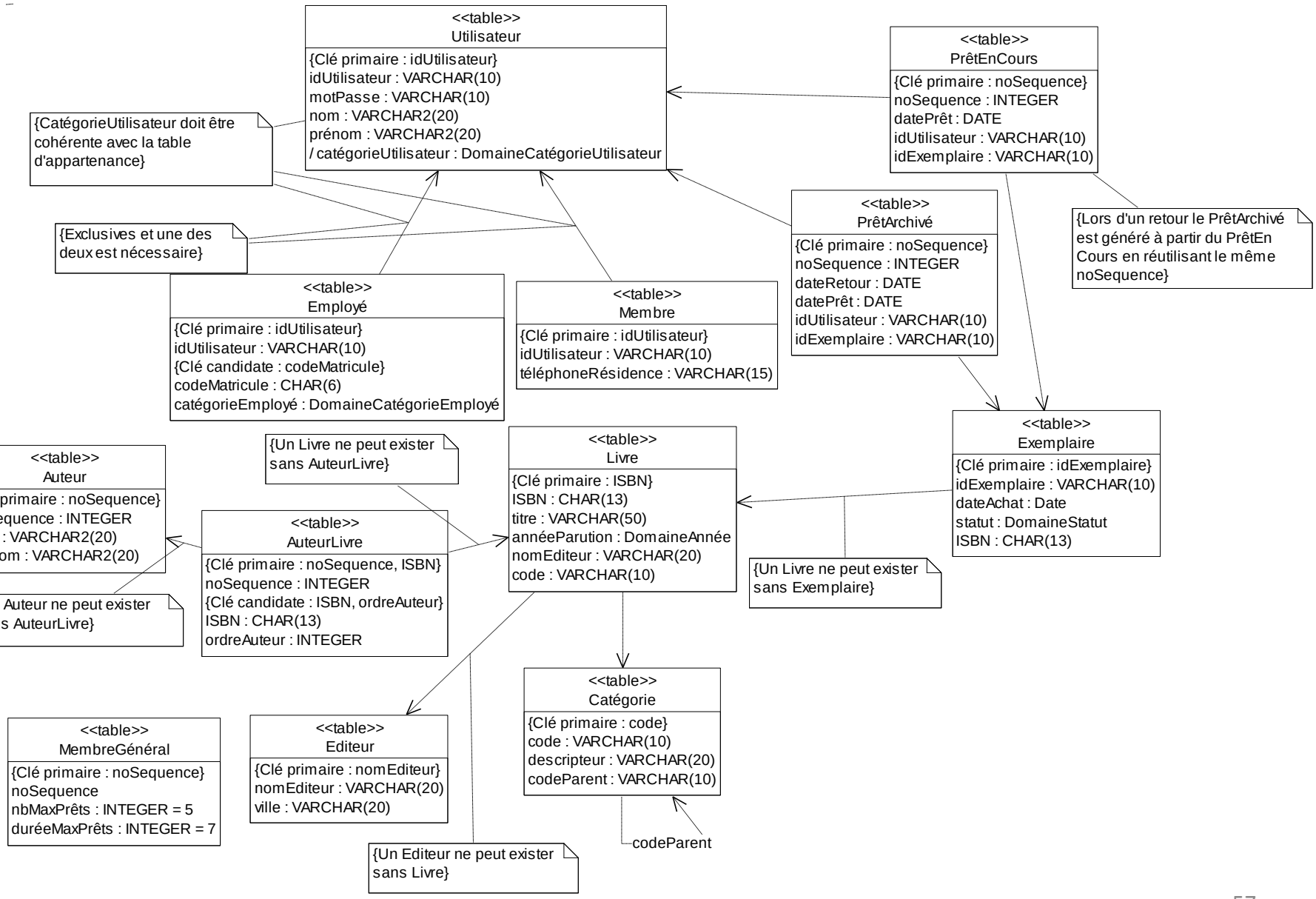
Dupliquer les colonnes et contraintes référentielles de la classe parent dans chacune des tables des enfants



Cas d ’une référence au parent



TRADUCTION DES GÉNÉRALISATIONS POUR SYLERAT



Traduction des autres contraintes