4. CONCEPTION LOGIQUE RELATIONNELLE

Version 2 - Septembre 2012

Support du chapitre 19, *Conception logique d'une base de données relationnelle* de l'ouvrage *Bases de données*, J-L Hainaut, Dunod 2018.

4. CONCEPTION LOGIQUE RELATIONNELLE

Contenu

4.1 INTRODUCTION

Contenu

a) Objectifs de la conception logique

b) Un exemple élémentaire

c) Principes de la conception logique

4.1 Introduction - Objectifs de la conception logique

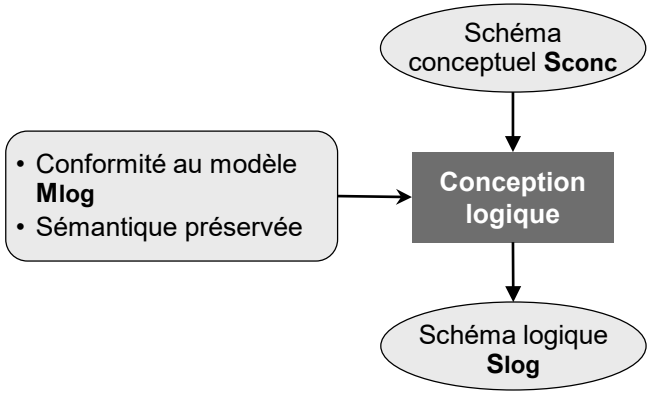
En (très) bref :

La conception logique consiste à produire un schéma SQL2

équivalent à un schéma Entité-association.

On introduira ensuite les constructions SQL3.

4.1 Introduction - Objectifs de la conception logique



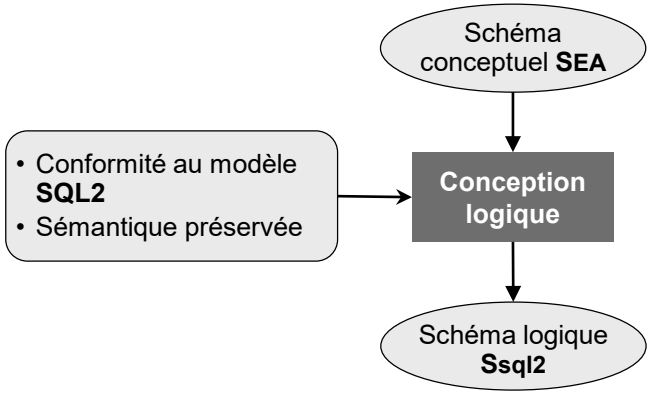
Plus généralement :

Soit Mlog un modèle logique (SQL2, SQL3, XML, fichiers, CODASYL, IMS, etc.)

La conception logique selon le modèle Mlog d'une base de données de schéma conceptuel Sconc consiste à produire un schéma Slog tel que :

1. **Slog est conforme à Mlog : il ne comporte que des constructions offertes par Mlog**
2. **Slog est équivalent à Sconc : Slog et Sconc expriment la même sémantique (externe)**

4.1 Introduction - Objectifs de la conception logique SQL2



Soit, ici :

La conception logique selon le modèle logique SQL2 d'une base de données de schéma conceptuel SEA consiste à produire un schéma Ssql2 tel que :

1. **Ssql2 est conforme à SQL2 : il ne comporte que des constructions offertes par SQL2 (*tables, colonnes, identifiants, clés étrangères,* etc.)**
2. **Ssql2 est équivalent à SEA : Ssql2 et SEA décrivent le même domaine d'application, ils ont la même sémantique externe**

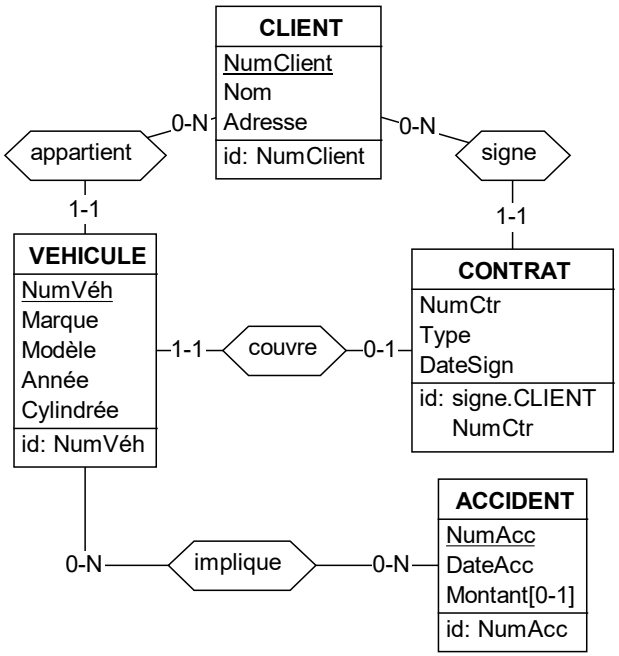
4.1 Introduction - Objectifs de la conception logique SQL2

Pourquoi SQL2 ?

- la plupart des BD en développement sont en SQL2;
- la conception logique strictement SQL3 (*relationnel objet*) pose des problèmes complexes (voir annexe G);
- la conception logique SQL3 est vue comme une extension de la conception logique SQL2

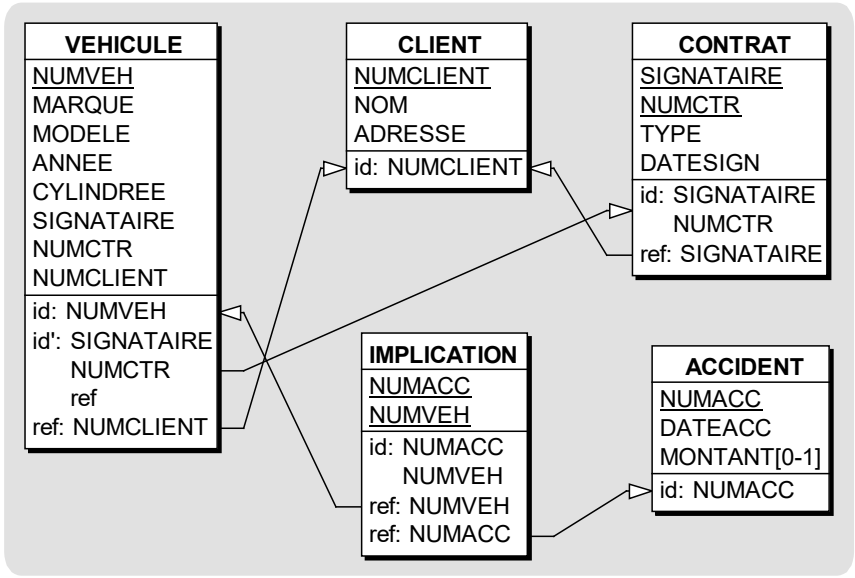
4.1 Introduction - Un exemple élémentaire

EA



⇒

SQL2



Note : ⇒ ou ⇔ ?

4.1 Introduction - Principe de la conception logique

On observe qu'un schéma EA contient des constructions directement exprimables en SQL2 :

- *type d'entités → table*
- *attribut simple → colonne*
- *identifiant primaire formé d'attributs → primary key*
- *identifiant secondaire formé d'attributs → prédicat **unique**()*

= constructions "*conformes*"

En revanche, les autres constructions demandent une conversion plus élaborée :

- *type d'associations → ?*
- *attribut composé → ?*
- *attribut multivalué → ?*
- *identifiant comportant un rôle → ?*
- *relations is-a → ?*
- *etc.*

= constructions "*non conformes*"

4.1 Introduction - Principe de la conception logique

Développement d'une stratégie de conception logique pour SQL2 :

- *identifier les constructions non conformes à SQL2*
- *pour chacune d'elles, répertorier les techniques de transformation de mise en conformité*
- *pour chaque construction non conforme, sélectionner une technique de transformation*
- *ordonnancer l'application de ces techniques pour constituer une procédure générale*
- *traduire cette procédure sous la forme d'un guide ou d'un logiciel.*

4.2 LE MODELE LOGIQUE RELATIONNEL

Contenu

4.2 Avant toutes choses ...

Avant toute chose :

- identifier les constructions EA *conformes* à SQL2
- par complémentarité, on en déduit les constructions EA *non conformes* à SQL2

On considère un modèle EA *étendu*, comportant notamment des attributs non ensemblistes (multi-ensembles, listes, tableaux) et des attributs de référence (= clés étrangères).

Constructions EA conformes :

- modèle SQL2 de base (comprend les constructions EA strictement conformes)
- modèle SQL2 enrichi (un peu plus laxiste, par facilité)

4.2 Le modèle SQL2 de base

Modèle SQL2 de base

- le schéma comporte des **types d'entités** (correspondant à des *tables*),
- tout type d'entités possède **un** ou plusieurs **attributs** (correspondant à des *colonnes*),
- un **attribut** est **mono-valué** et **atomique**; il est obligatoire ou facultatif,
- les seules contraintes sont celles qui sont induites par les **identifiants** et les **attributs de référence** (appelés *clés étrangères*),
- le schéma relationnel ne contient pas d'autres constructions,
- les noms des objets respectent la syntaxe SQL.

4.2 Le modèle SQL2 enrichi

Modèle SQL2 enrichi

- le schéma comporte des **types d'entités** (correspondant à des *tables*),
- tout type d'entités possède **un** ou plusieurs **attributs** (correspondant à des *colonnes*),
- un **attribut** est **mono-valué** et **atomique**; il est obligatoire ou facultatif,
- les seules contraintes sont celles qui sont induites par les **identifiants**, les **clés étrangères** et les **clés étrangères totales** (*equ*) ainsi que les **contraintes d'existence** (*coex, excl, at-least-1, exact-1, if*),
- le schéma relationnel ne contient pas d'autres constructions,
- les noms des objets respectent la syntaxe SQL.

4.2 Les constructions non conformes à SQL2

On en déduit :

Les constructions EA *non conformes* ou *invalides*

- les attributs multivalués,
- les attributs composés,
- les types d'associations,
- les relations *is-a*,
- les types d'entités sans attribut,
- les contraintes autres que celles des identifiants, des clés étrangères (éventuellement **equ**) et d'existence,
- les noms d'objets non conformes à la syntaxe SQL.

4.2 La suite . . .

Nous étudierons donc les techniques de transformation :

- d'un attribut complexe
  - composé
  - multivalué
  - composé multivalué
- d'un type d'associations
  - fonctionnel
  - complexe
  - cyclique
  - etc.
- de relations *is-a*,
- etc.

4.3 TRANSFORMATION D'UN ATTRIBUT COMPLEXE

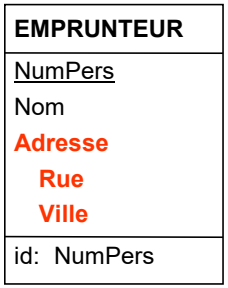
Contenu

a) **Attribut composé**

b) **Attribut multivalué**

c) **Attribut composé multivalué**

4.3 Transformation d'un attribut composé



Transformation d'un attribut composé

a) **Désagrégation** (attribut remplacé par ses composants)

b) Transformation en **type d'entités** (représentation des instances)

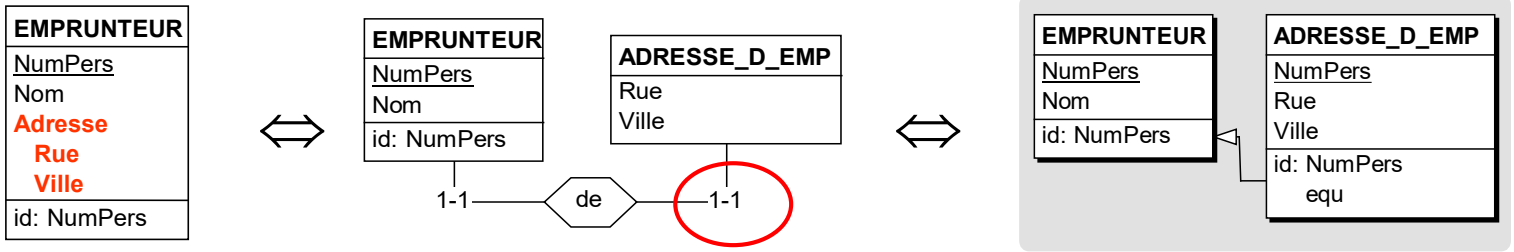
c) Transformation en **type d'entités** (représentation des valeurs)

d) Représentation déconseillée

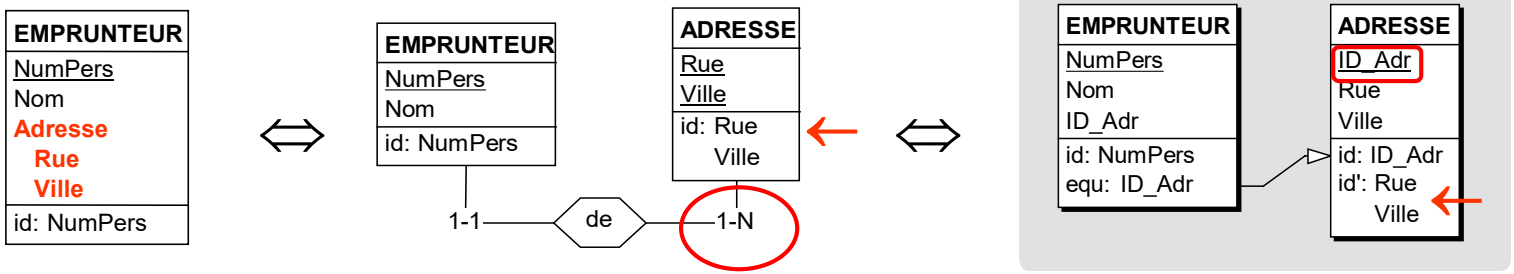
4.3 Transformation d'un attribut composé (obligatoire)

a)

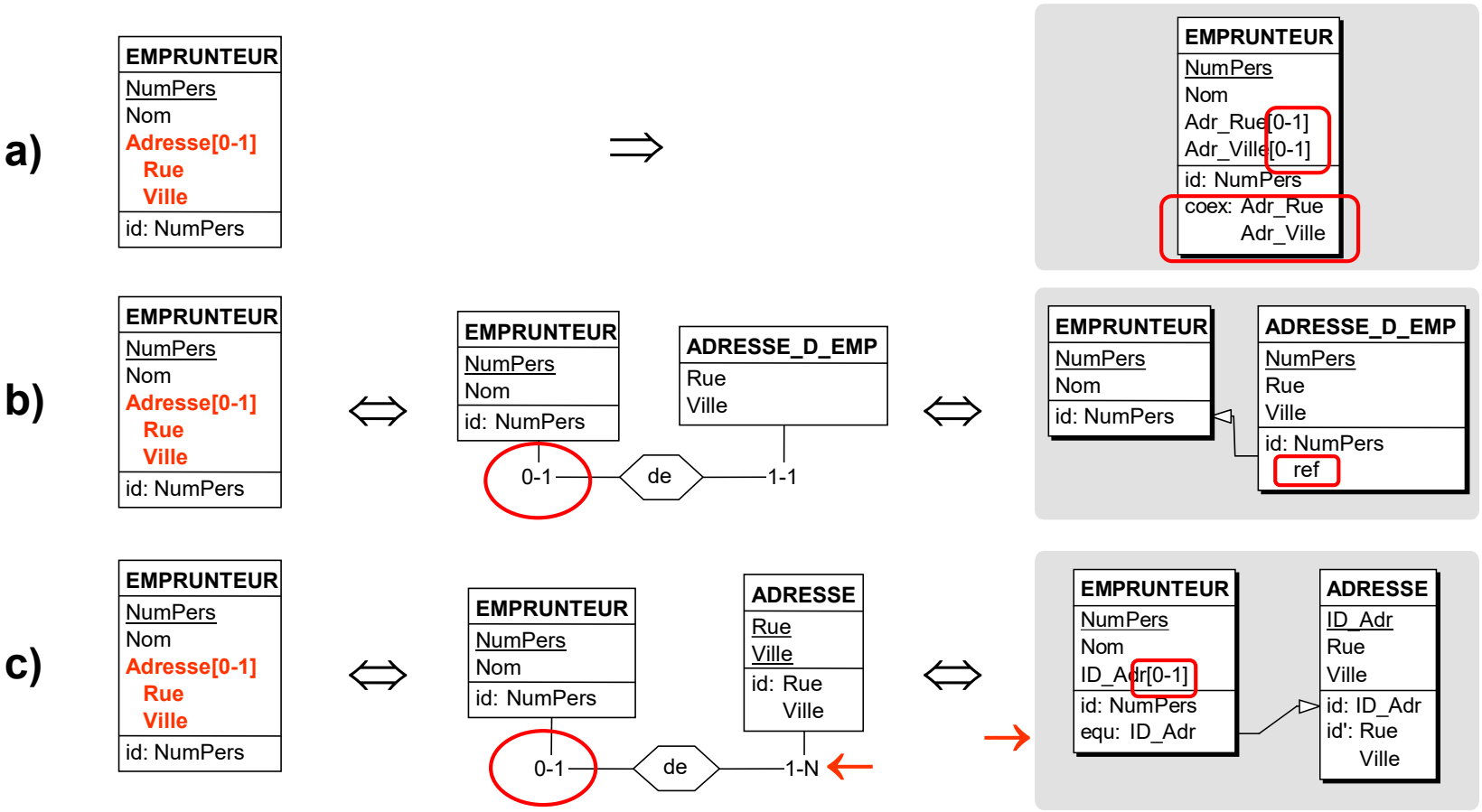
b)



c)

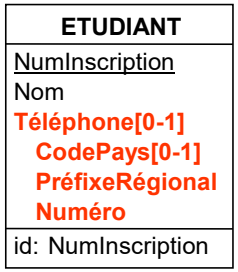


4.3 Transformation d'un attribut composé (facultatif)

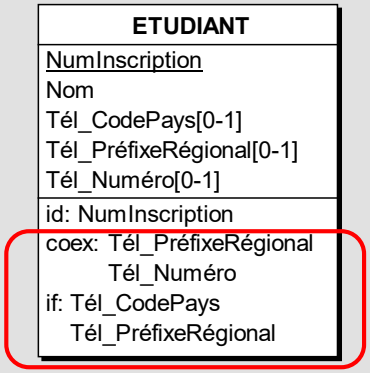


4.3 Transformation d'un attribut composé (facultatif à composants facultatifs)

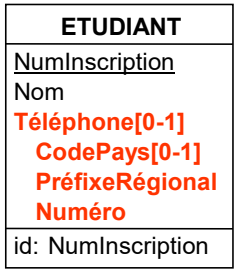
a)



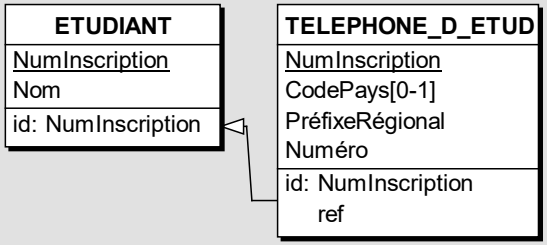
⇒



b)



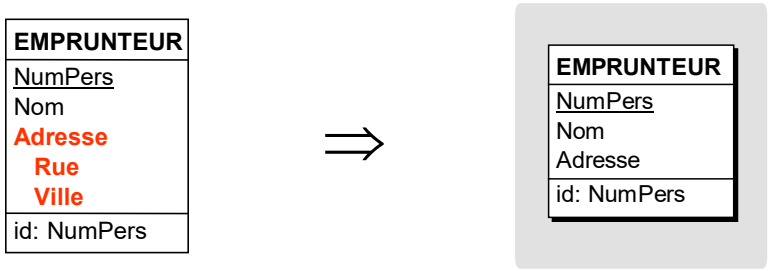
⇔



pourquoi pas par " *représentation des valeurs*" ?

4.3 Transformation d'un attribut composé (déconseillée)

c) concaténation :



4.3 Transformation d'un attribut multivalué

DOCUMENT
ID-Doc Titre **MotClé[0-10]**
id: ID-Doc

Transformation d'un attribut multivalué

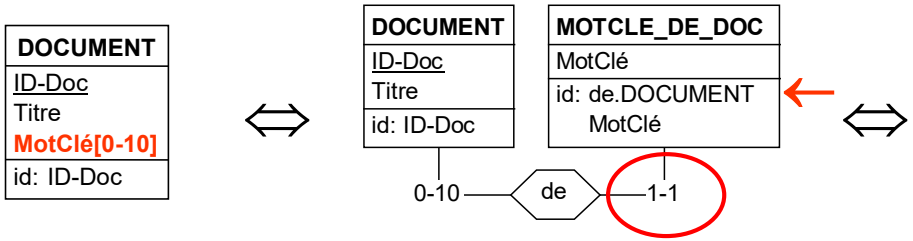
a) Transformation en **type d'entités** (représentation des instances)

b) Transformation en **type d'entités** (représentation des valeurs)

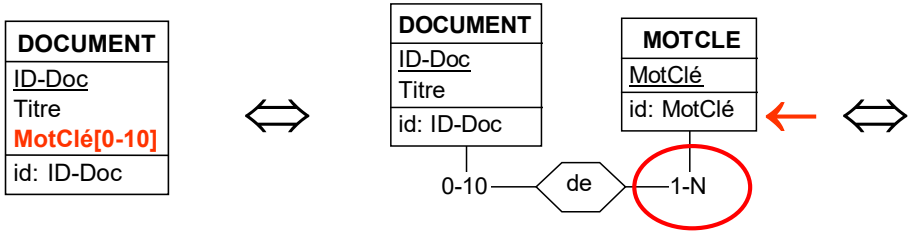
c) Représentations déconseillées

4.3 Transformation d'un attribut multivalué

a)



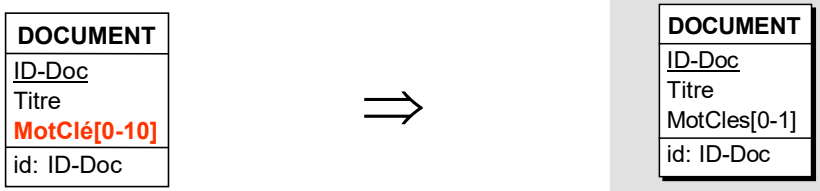
b)



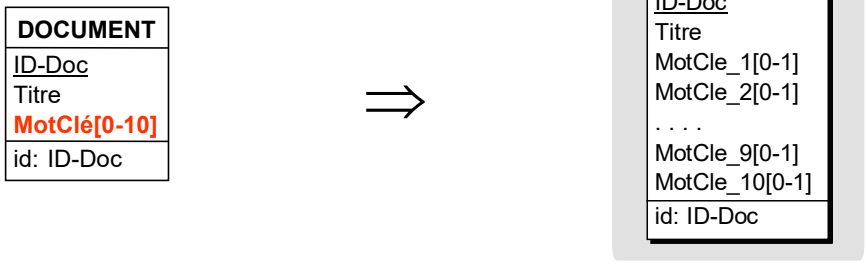
Etrange : *le schéma (a) est un sous-ensemble du schéma (b) ! Qu'en penser ?*

4.3 Transformation d'un attribut multivalué (représentations déconseillées)

c1) concaténation :

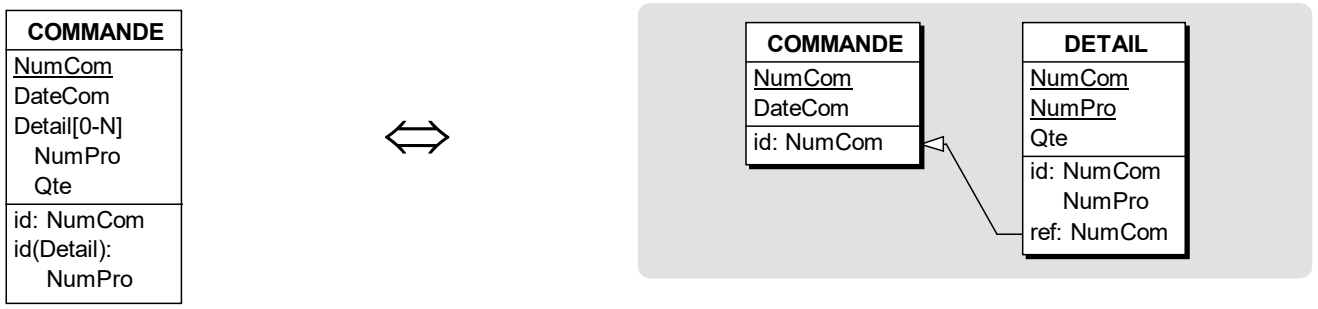


c2) instanciation :



pourquoi " *déconseillées*" ?

4.3 Transformation d'un attribut composé multivalué



4.4 TRANSFORMATION D'UN TYPE D'ASSOCIATIONS

Contenu

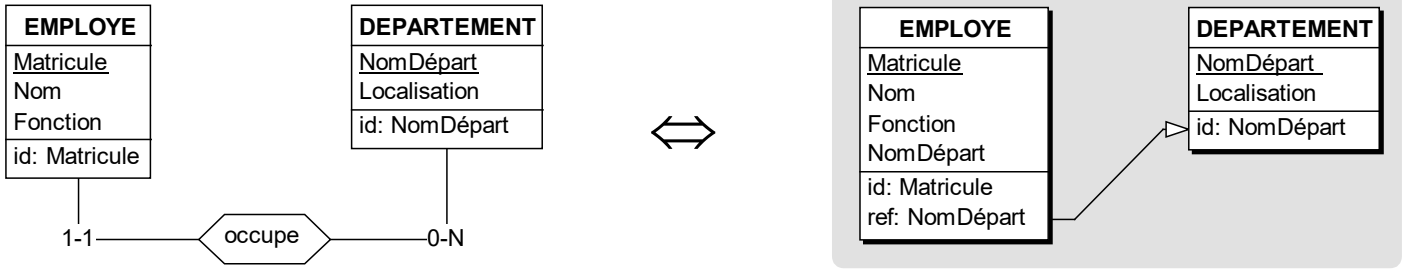
a) **Type d'associations fonctionnel**

b) **Type d'associations complexe**

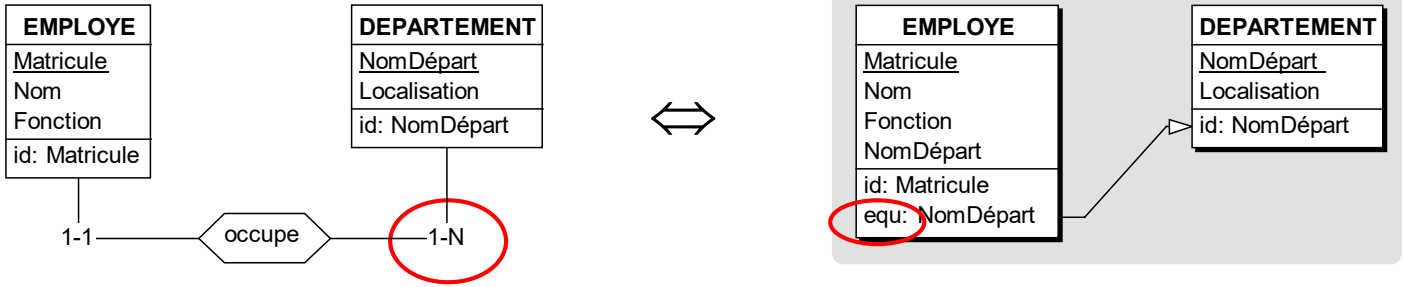
c) **Type d'associations cyclique**

d) **Type d'associations à rôle multitype (polymorphique)**

4.4 Transformation d'un type d'associations fonctionnel (N:1)

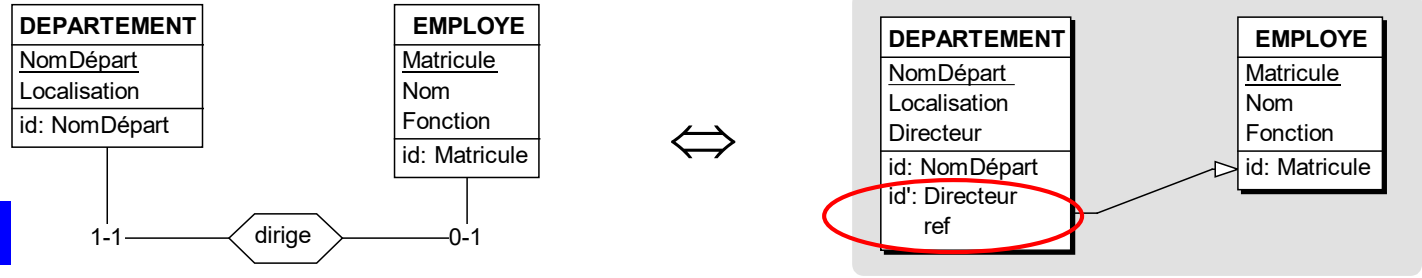


!

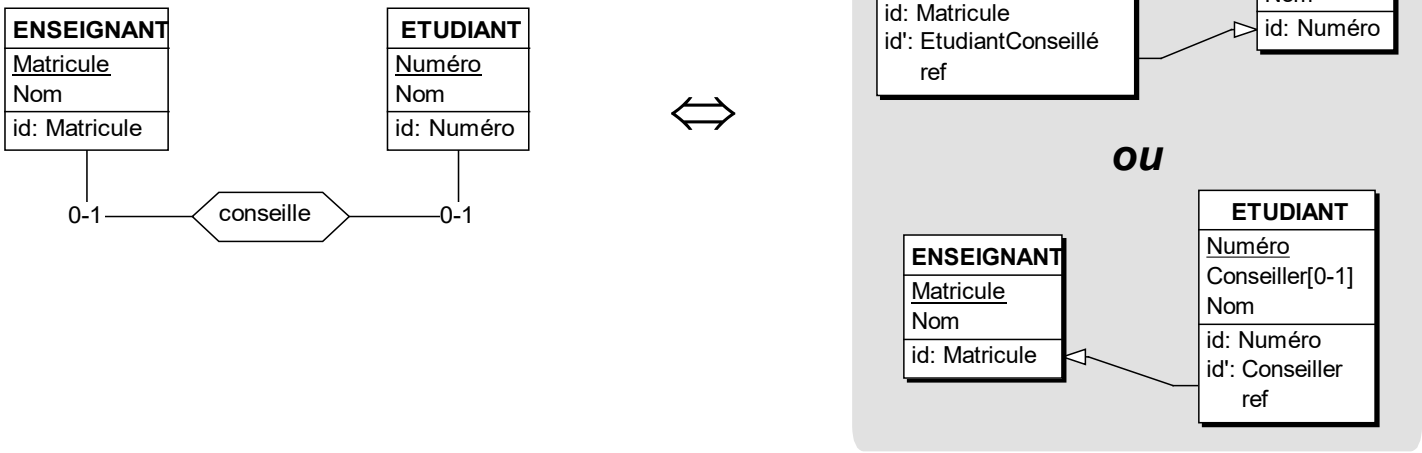


4.4 Transformation d'un type d'associations fonctionnel (1:1)

asymétrique

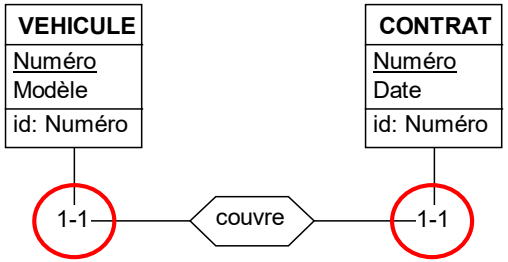


symétrique

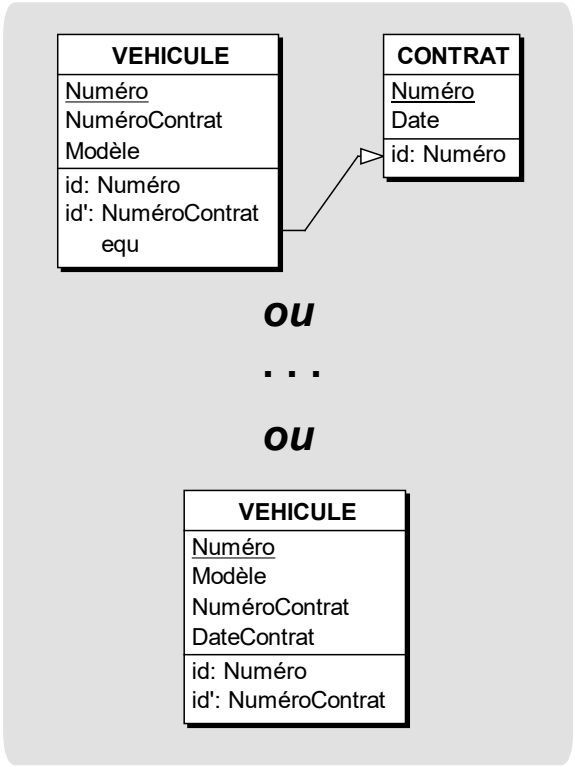


4.4 Transformation d'un type d'associations fonctionnel (1:1, bi-obligatoire)

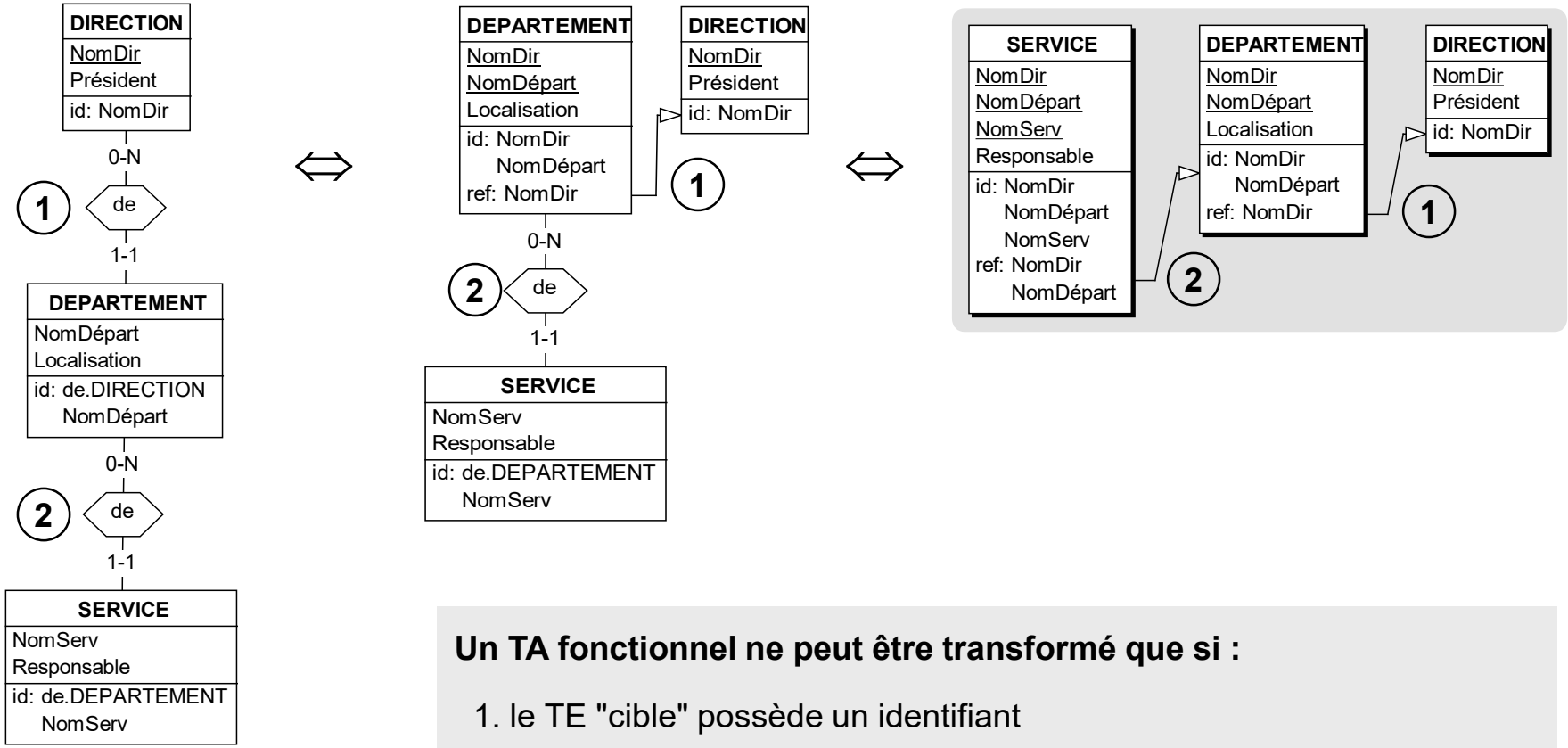
symétrique



⇔



4.4 Transformation d'un type d'associations fonctionnel (! identifiants hybrides)

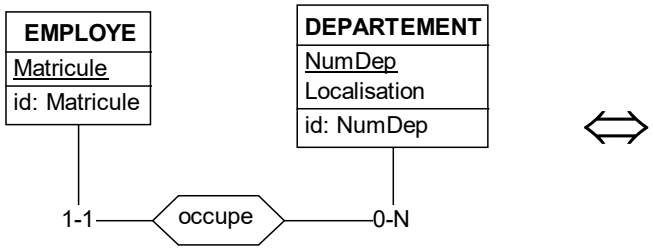


Un TA fonctionnel ne peut être transformé que si :

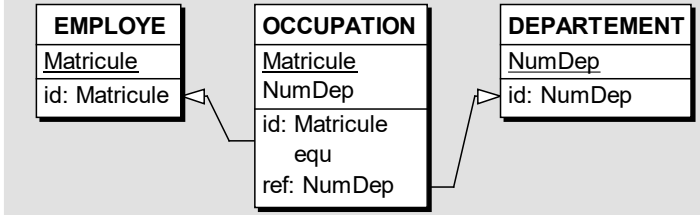
1. le TE "cible" possède un identifiant
2. cet identifiant n'est constitué que d'attributs;

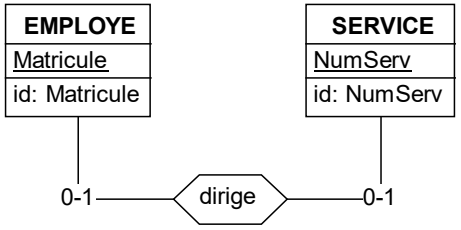
⇒ les TA doivent être transformés dans un certain ordre imposé par les identifiants hybrides.

4.4 Transformation d'un type d'associations fonctionnel (en type d'entités)

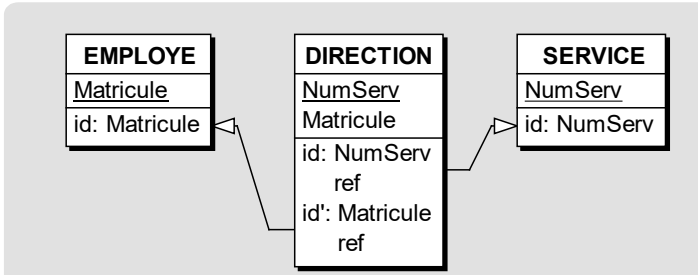


⇔





⇔



quel intérêt ?

4.4 Transformation d'un type d'associations complexe

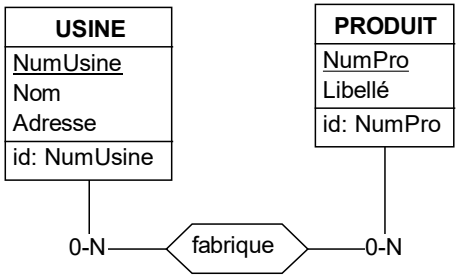
Type d'associations complexe =

1. N:N
2. n-aire
3. doté d'attributs

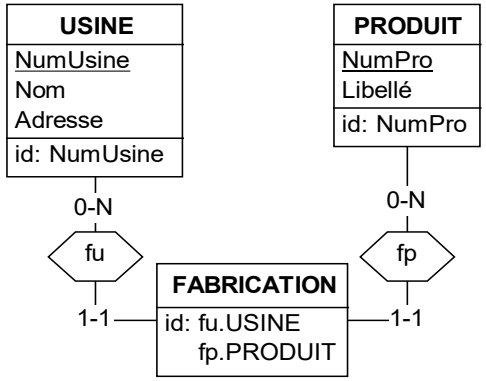
Représentation d'un type d'associations complexe R :

1. transformation de **R** en 1 TE **R'** + n TA fonctionnels **R1**, **R2**, .., **Rn**
2. transformation des TA fonctionnels **R1**, **R2**, .., **Rn**

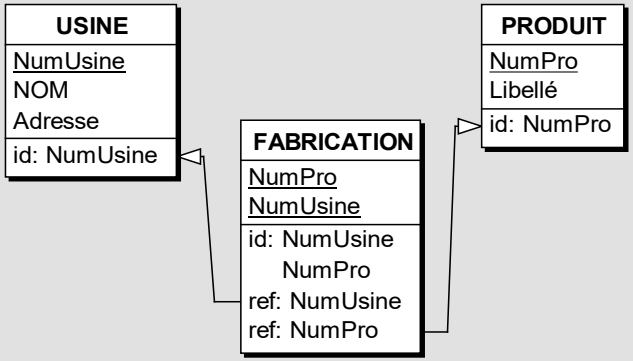
4.4 Transformation d'un type d'associations complexe (binaire)



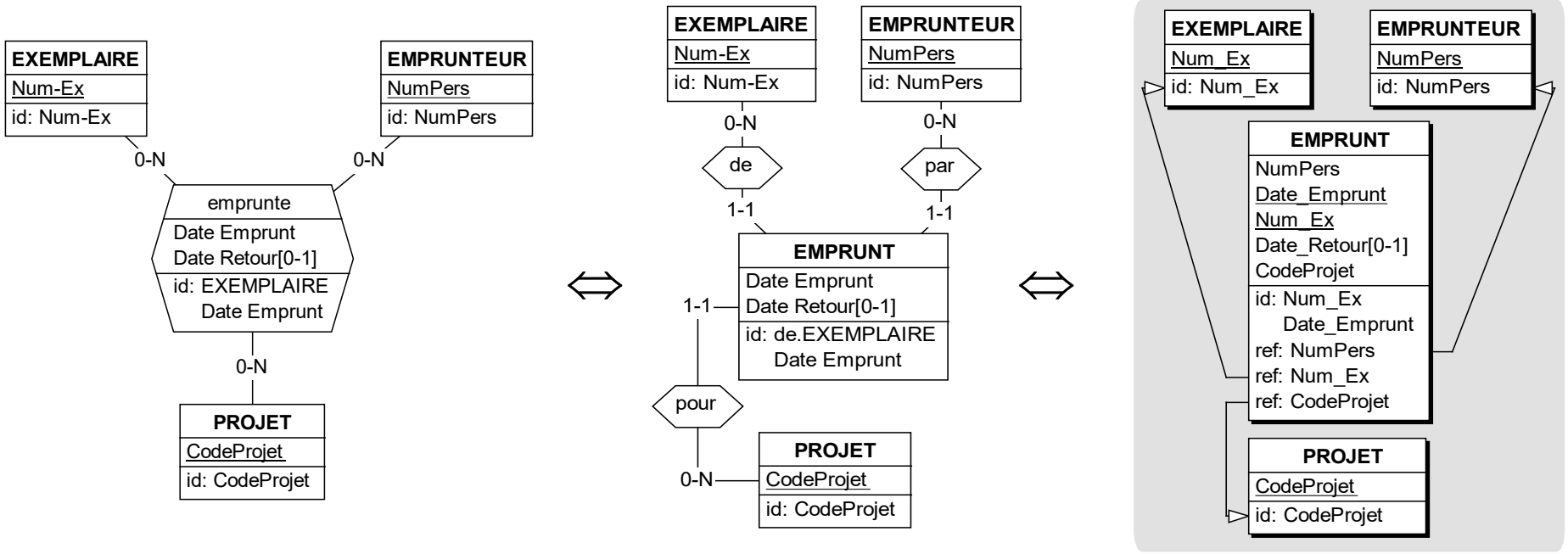
⇔



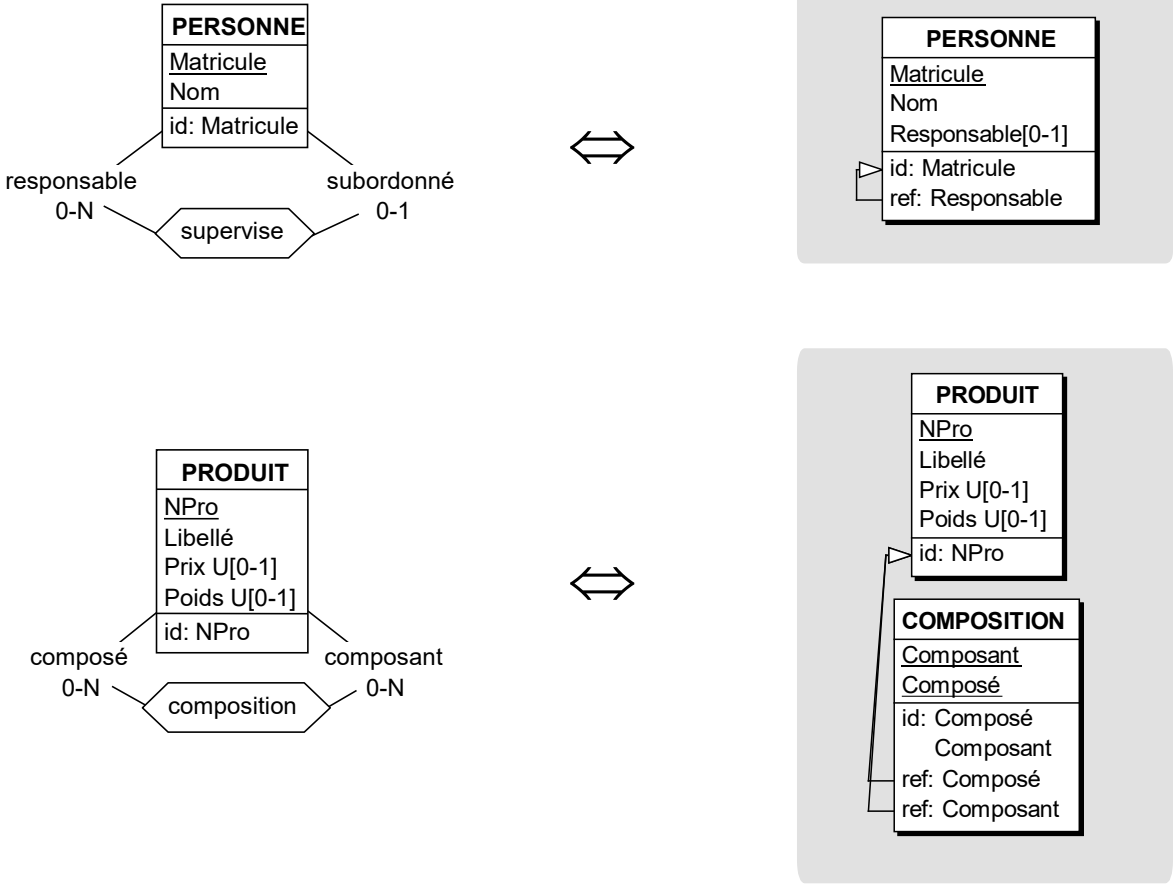
⇕



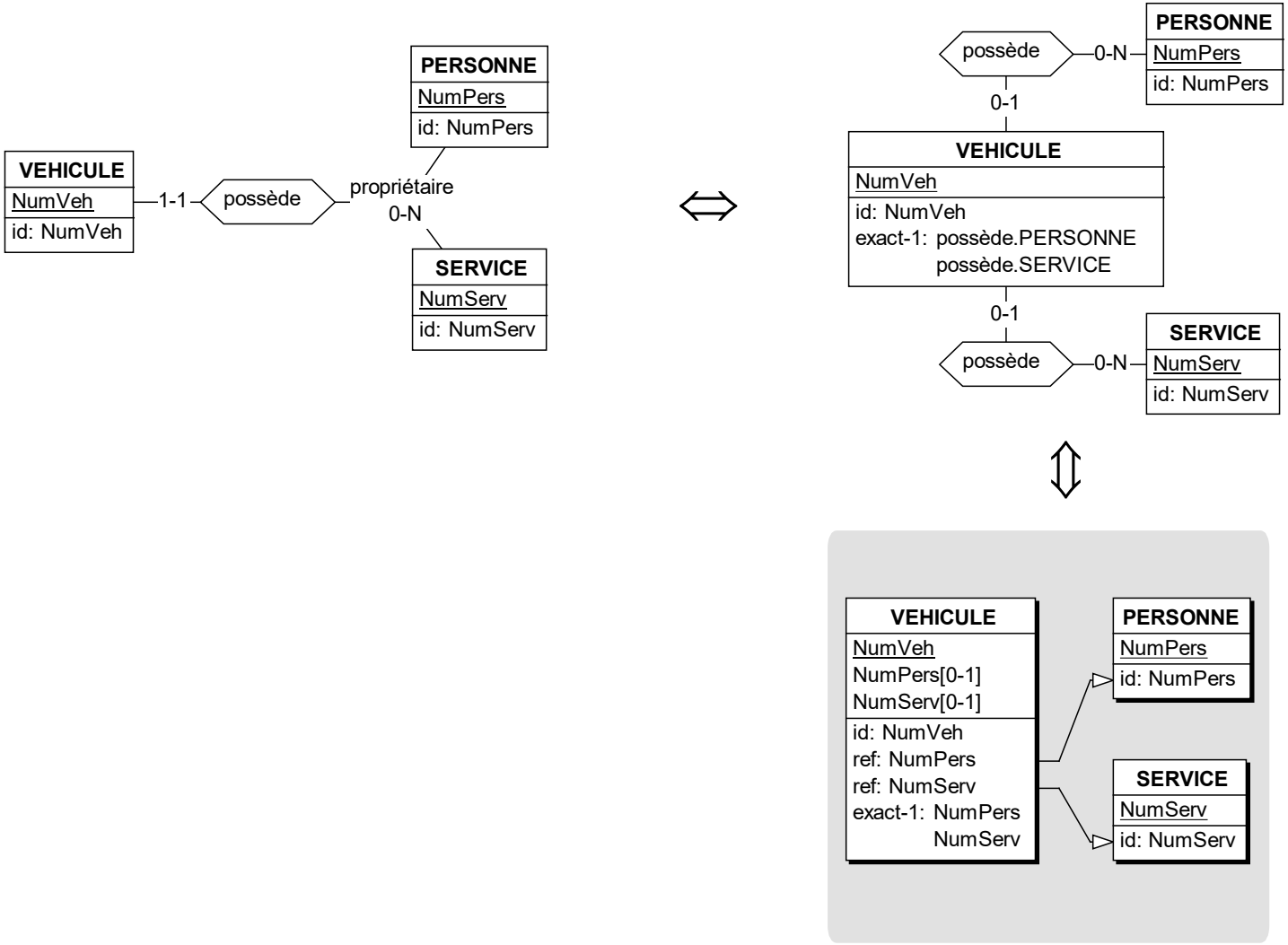
4.4 Transformation d'un type d'associations complexe (n-aire)



4.4 Transformation d'un type d'associations cyclique

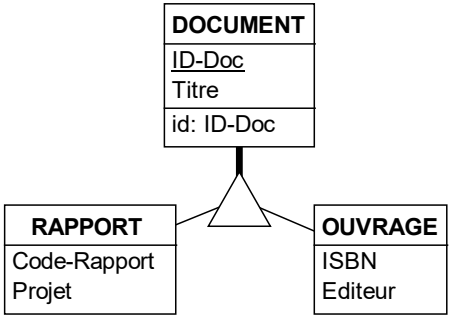


4.4 Transformation d'un type d'associations à rôle multitype



4.5 TRANSFORMATION DE RELATIONS *is-a*

4.5 Les relations *is-a*



Transformation d'une relation *is-a*

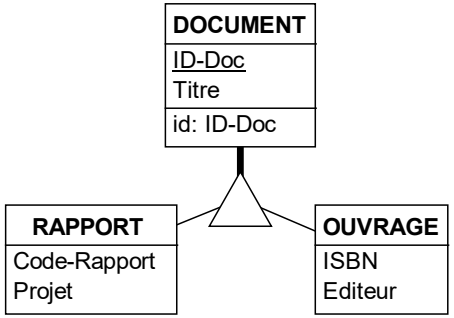
Trois familles de techniques :

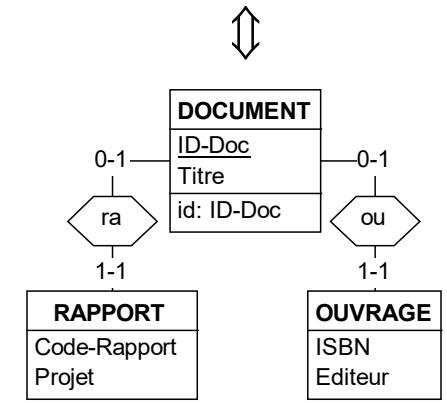
a) Matérialisation des relations *is-a*

b) Héritage descendant

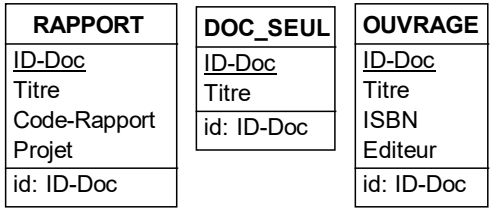
c) Héritage ascendant

4.5 Les relations *is-a*

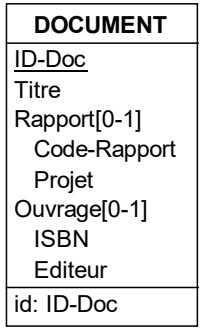




matérialisation
ou
une table par TE

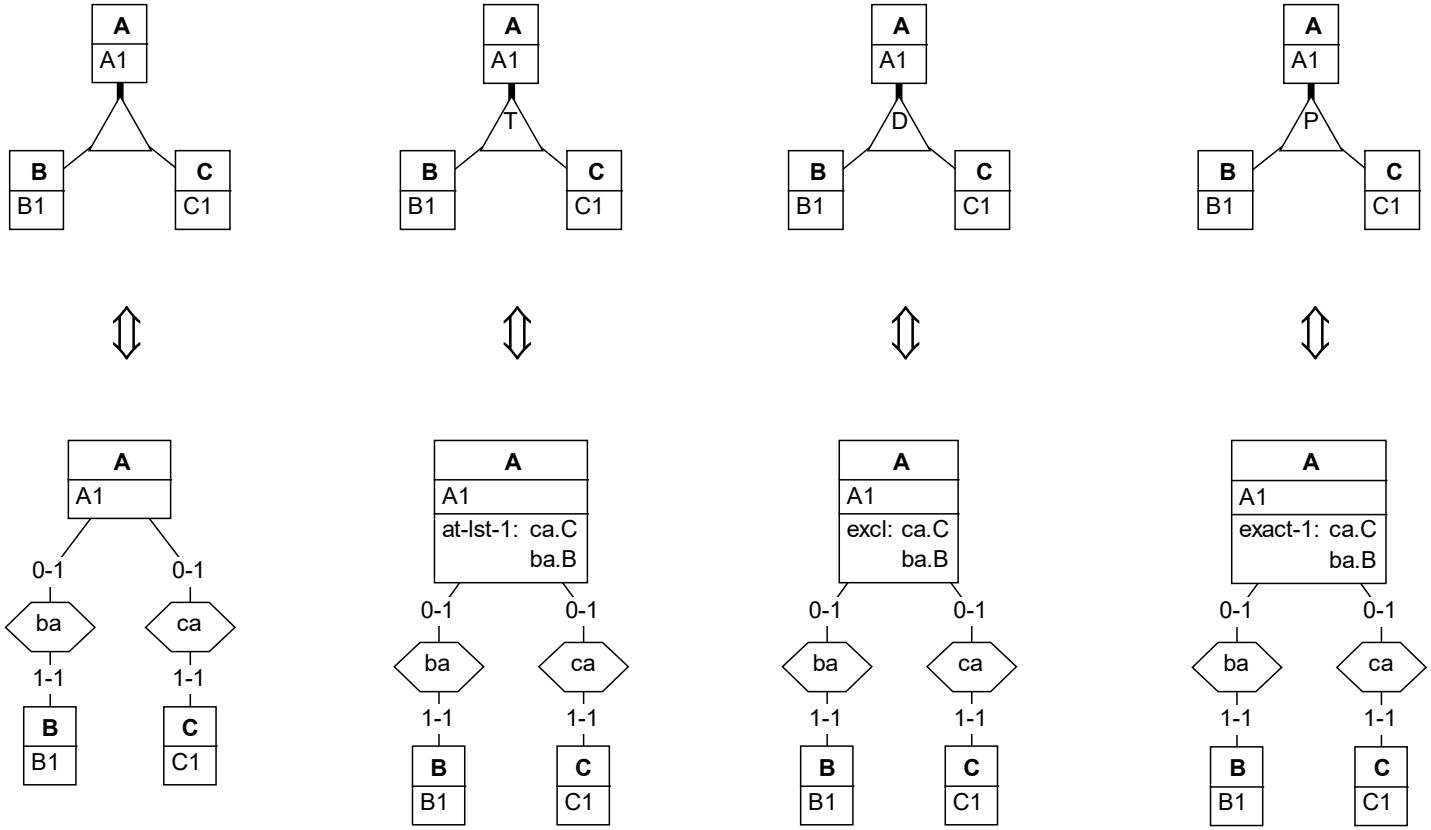


héritage descendant
ou
une table par sous-type



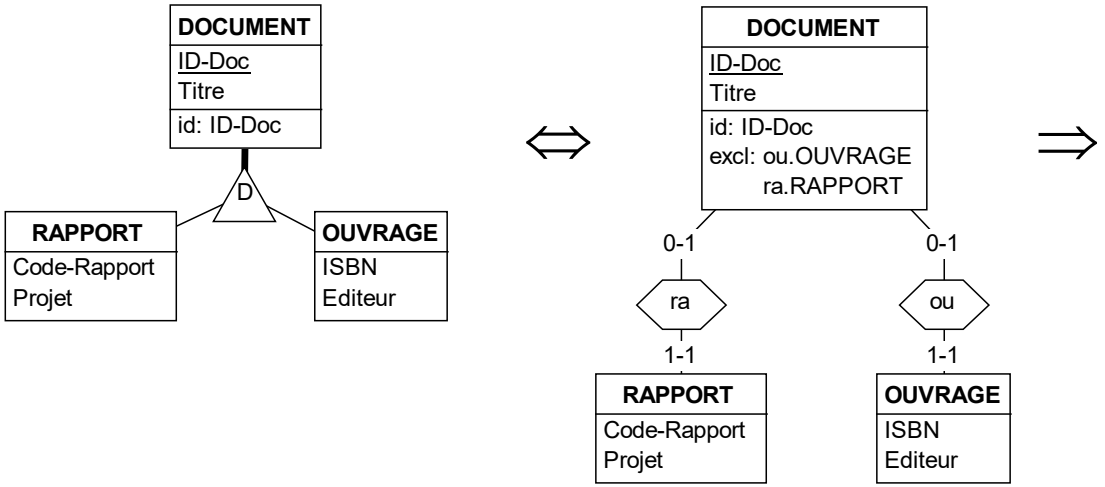
héritage ascendant
ou
une table par surtype

4.5 Les relations *is-a* - Matérialisation



4.5 Les relations *is-a* - Matérialisation et transformation des TA 1:1

Exemple de la disjonction



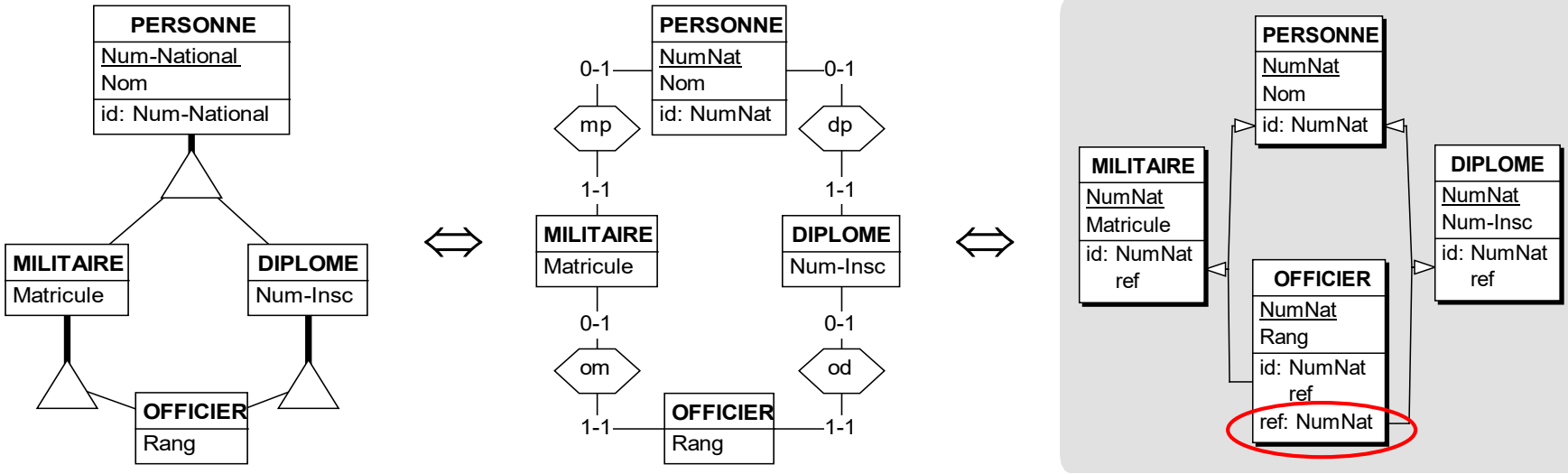
+ contrainte similaire pour OUVRAGE

Conditions :

1. le surtype possède un identifiant (primaire)
2. le surtype possède des indicateurs de sous-type

4.5 Les relations *is-a* - Matérialisation d'une hiérarchie multiple

***Attention** aux clés étrangères de même valeur*

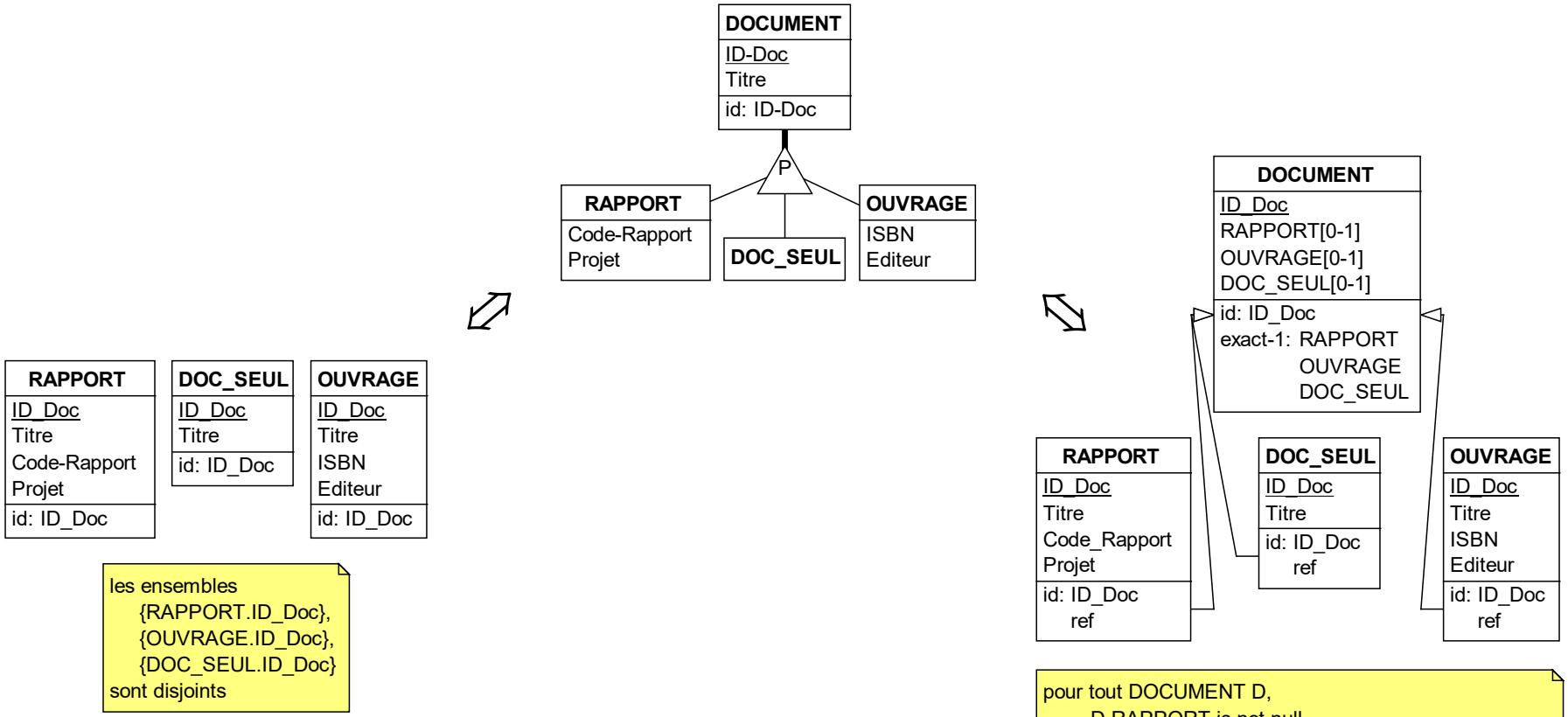


(om*mp)[OFFICIER,PERSONNE]
= (od*dp)[OFFICIER,PERSONNE]

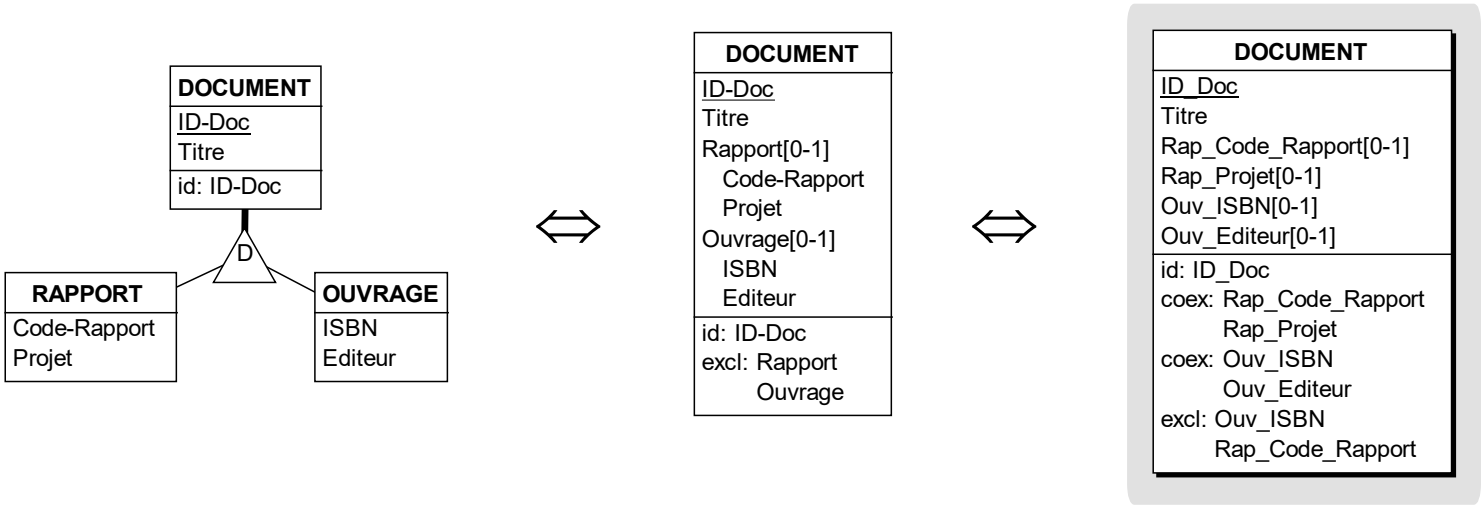
Un seul attribut NumNat !

4.5 Les relations *is-a* - Héritage descendant

A éviter en cas de non-disjonction



4.5 Les relations *is-a* - Héritage ascendant



4.6 COMPLEMENTS

Contenu

a) Identifiants facultatifs

b) Identifiants primaires complexes

c) Noms des objets du schéma

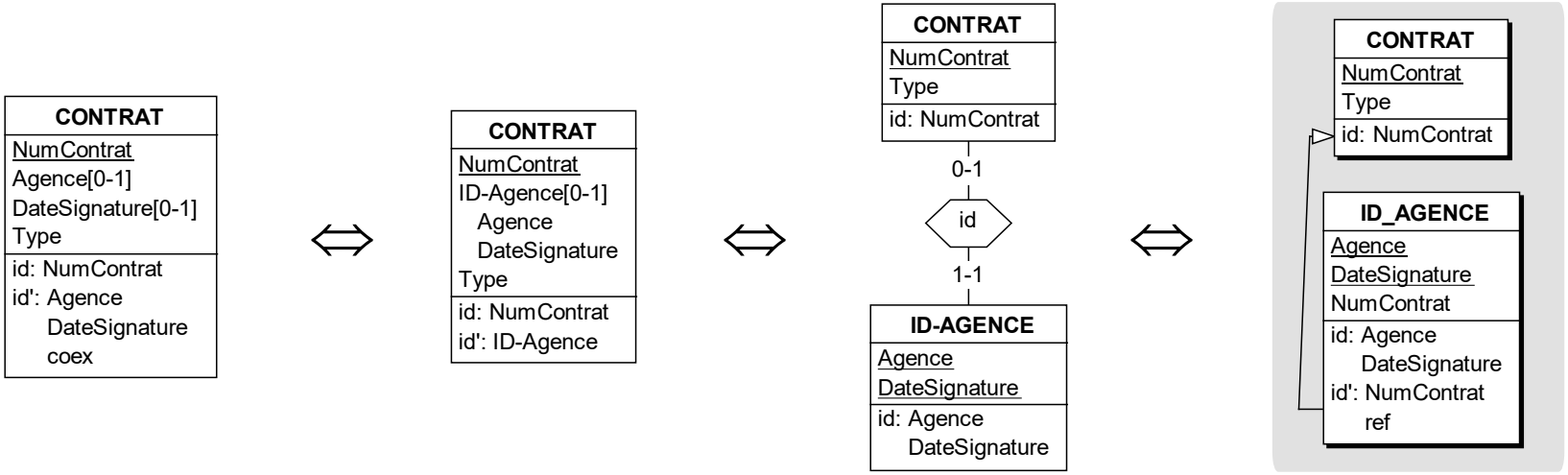
d) Rôles de cardinalité [1-N]

e) Traduction des vues conceptuelles

4.6 Compléments - Identifiants facultatifs

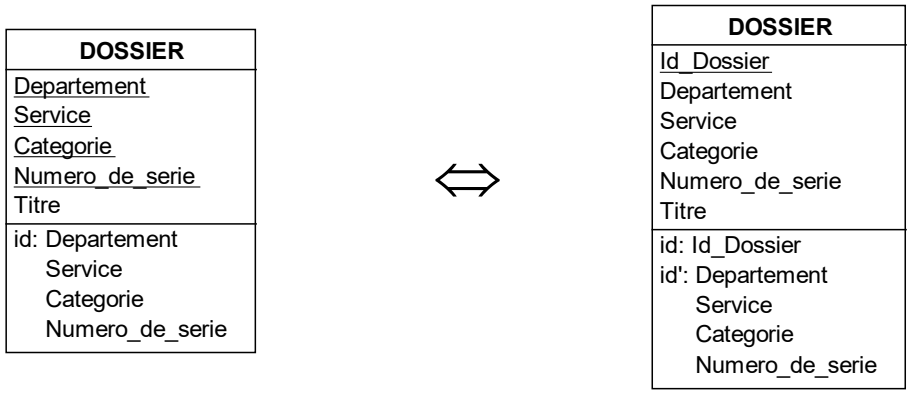
Identifiants facultatifs à conserver ou à transformer selon *idéologie locale* et capacité du SGBD

Un pattern complexe à simplifier :



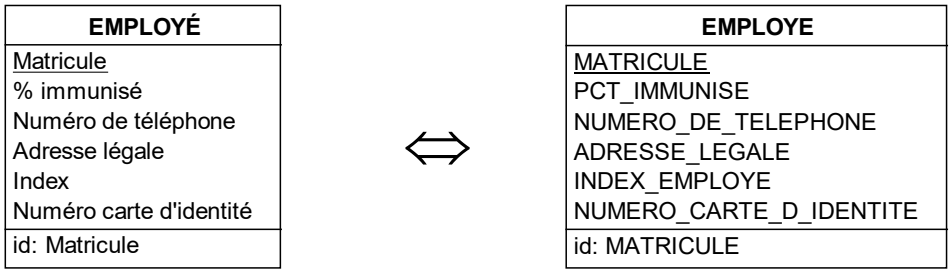
4.6 Compléments - Identifiants primaires complexes

Un identifiant primaire multicomposants peut être remplacé par un identifiant technique



4.6 Compléments - Noms des objets du schéma

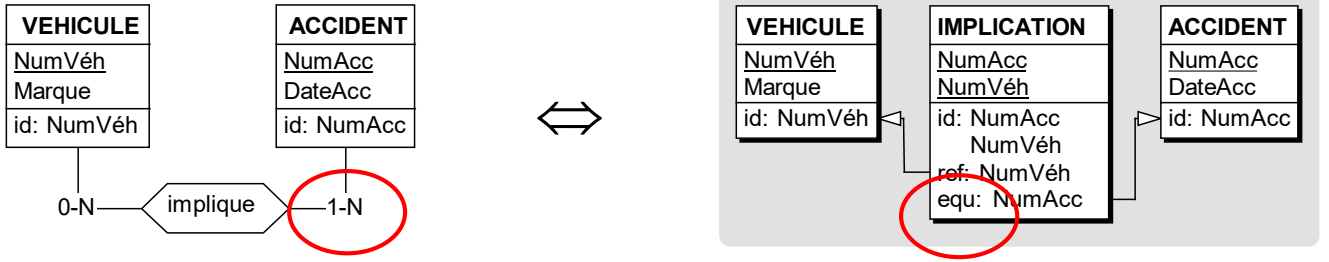
Les noms doivent respecter la syntaxe SQL - Choisir des règles systématiques



4.6 Compléments - Rôles de cardinalité [1-N]

Utiles mais complexes à gérer :

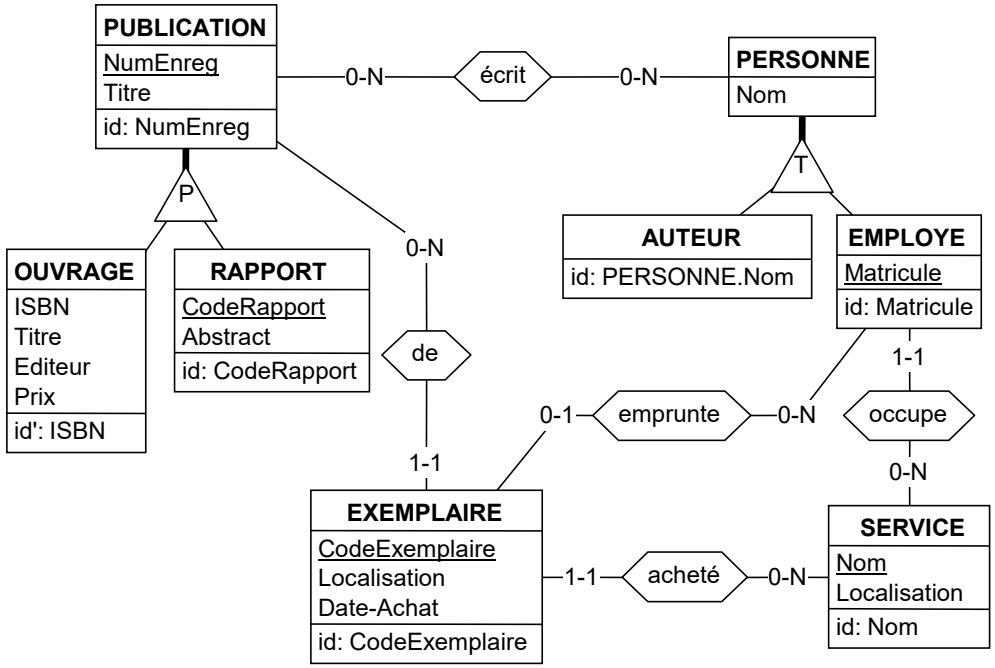
- dynamique de modification des données problématique (IMPLICATION et/ou ACCIDENT)
- impossibles à traduire par des *checks* ou des *triggers*



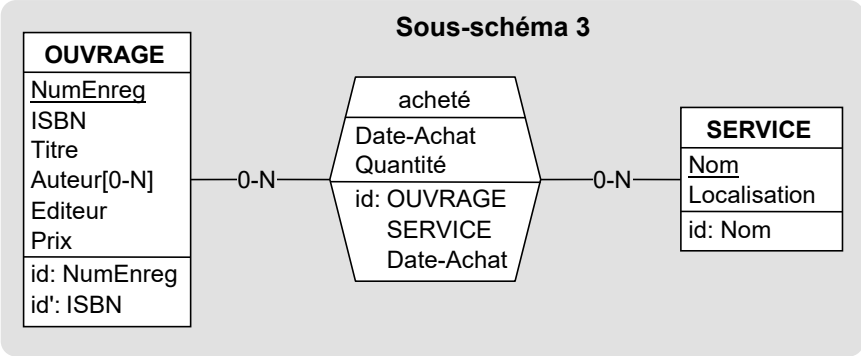
Approches :

- transactions
- procédures SQL (*stored procedures*): créer un accident et ses implications
- méthodes de table typée (SQL3): créer un accident et ses implications

4.6 Compléments - Traduction des vues conceptuelles

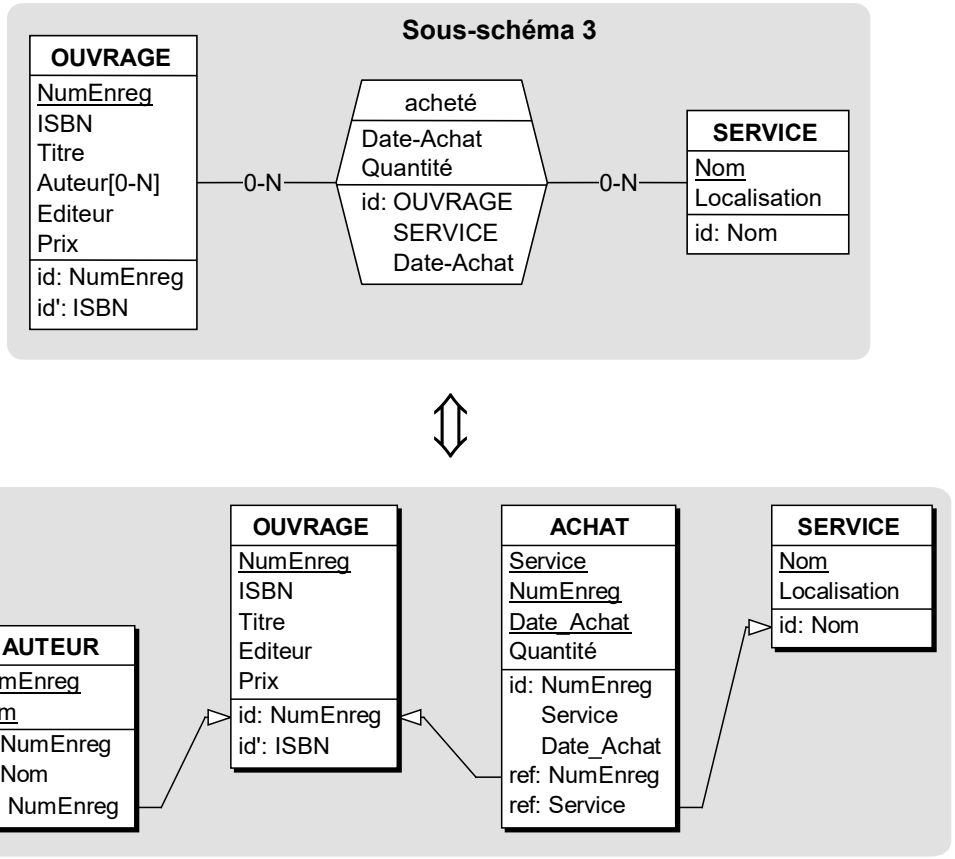


Dans la leçon 3 :
le ***Sous-schéma 3*** a servi à construire le schéma conceptuel global.
Inversement, il est ensuite considéré comme une vue sur celui-ci.



4.6 Compléments - Traduction des vues conceptuelles

Une vue conceptuelle se convertit comme le schéma conceptuel complet



4.7 PLAN DE TRANSFORMATION POUR LA CONCEPTION LOGIQUE RELATIONNELLE

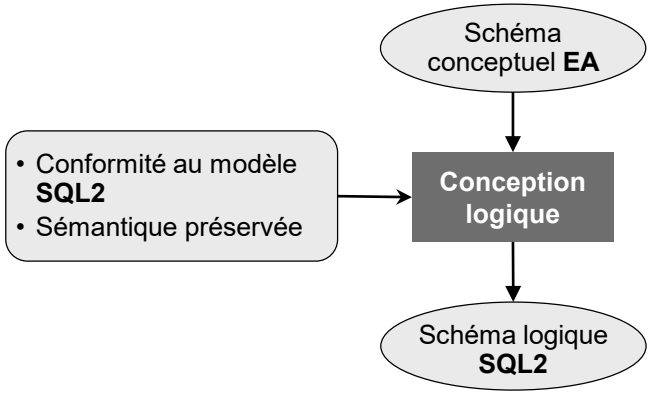
Contenu

a) **Principes**

b) **Choix des techniques de transformation**

c) **Plan de transformation pour SQL2**

4.7 Principes



Plan de transformation **:** ordonnancement des transformations des constructions EA non conformes au modèle SQL2

4.7 Principes

Les constructions invalides (rappel) ou à éviter :

- les relations *is-a*,
- les attributs composés,
- les attributs multivalués,
- les types d'associations fonctionnels,
- les types d'associations N:N ou complexes,
- les contraintes non conformes,
- les noms d'objets non conformes à la syntaxe SQL,
- les identifiants facultatifs (si nécessaire),
- les identifiants primaires complexes (si nécessaire).

Pour chacune :

- on sélectionne **une** (chaîne de) transformation(s) de mise en conformité,

4.7 Choix des techniques de transformation

Traitement des relations *is-a*

- transformation par matérialisation,
- transformation des TA 1:1 en clés étrangères,
- ajout d'identifiants techniques si nécessaire.

Traitement des attributs composés,

- les attributs composés de niveau 1 sont désagrégés,
- + contrainte de coexistence si nécessaire.

Traitement des attributs multivalués,

- transformation des attributs multivalués de niveau 1 en types d'entités par représentation des instances,
- transformation des TA fonctionnels en clés étrangères,
- ajout d'identifiants techniques si nécessaire.

4.7 Choix des techniques de transformation

Traitement des types d'associations fonctionnels,

- transformation en clés étrangères,
- possible si le TE "cible" possède un identifiant primaire formé d'attributs,
- ajout d'identifiants techniques si nécessaire.

Traitement des types d'associations N:N ou complexes,

- transformation en types d'entités,
- transformation des TA fonctionnels en clés étrangères,
- possible si le TE "cible" possède un identifiant primaire formé d'attributs,
- ajout d'identifiants techniques si nécessaire.

4.7 Choix des techniques de transformation

Traitement des contraintes non conformes,

- sous forme d'annotations.

Traitement des noms d'objets non conformes à la syntaxe SQL,

- suppression des accents et autres signes diacritiques,
- remplacement des espaces et autres séparateurs par "_",
- remplacement des mots réservés SQL.

4.7 Choix des techniques de transformation

Traitement des identifiants facultatifs (si nécessaire),

- transformation des composants en un type d'entités par représentation des instances,
- transformation des TA fonctionnels en clés étrangères,
- possible si le TE "cible" possède un identifiant primaire formé d'attributs,
- ajout d'identifiants techniques si nécessaire.

Traitement des identifiants primaires complexes (si nécessaire).

- statut secondaire
- ajout d'un attribut technique constituant l'identifiant primaire.

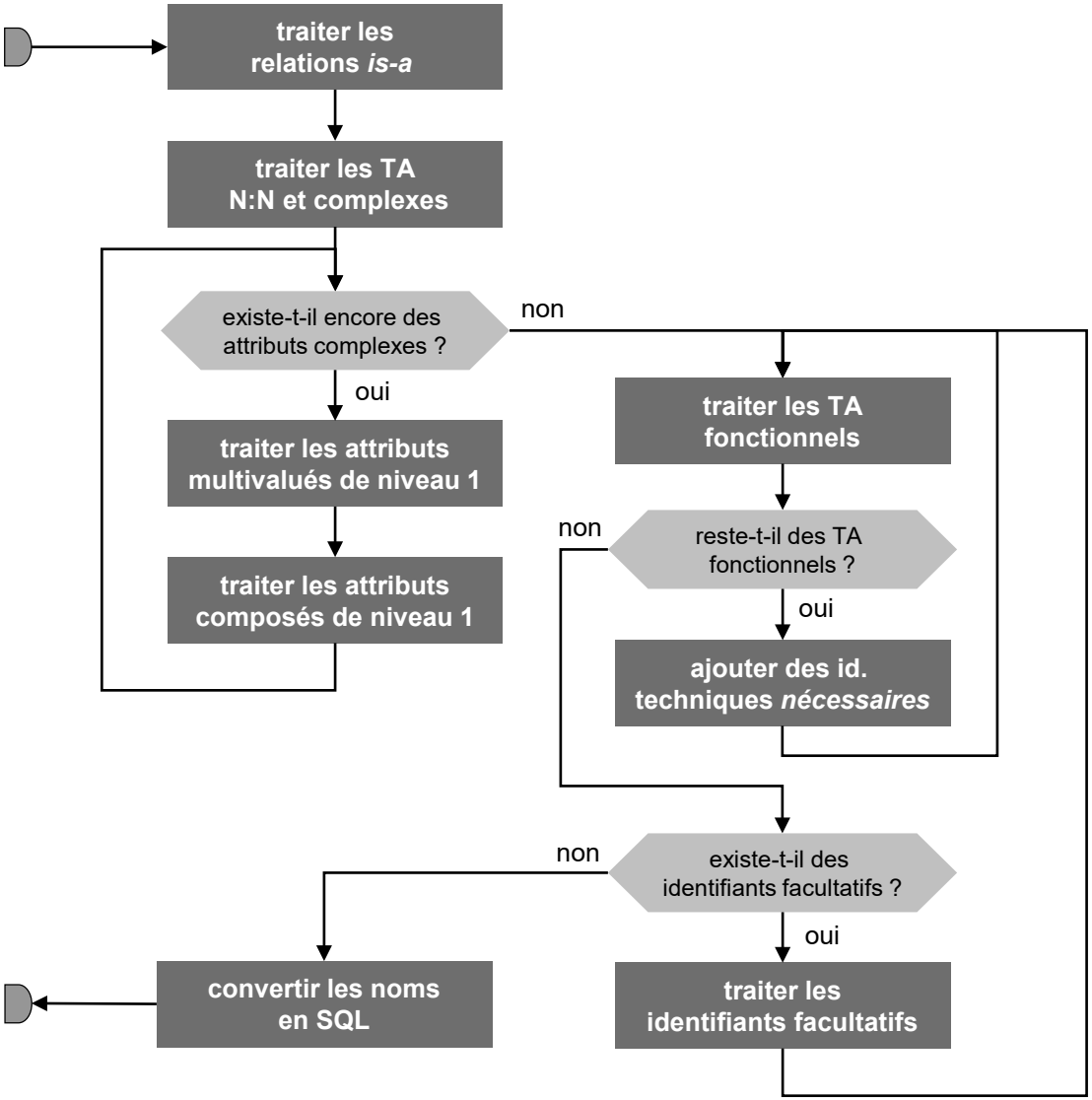
4.7 Plan de transformation pour SQL2

On en déduit un plan de transformation

... par ordonnancement des transformations élémentaires,

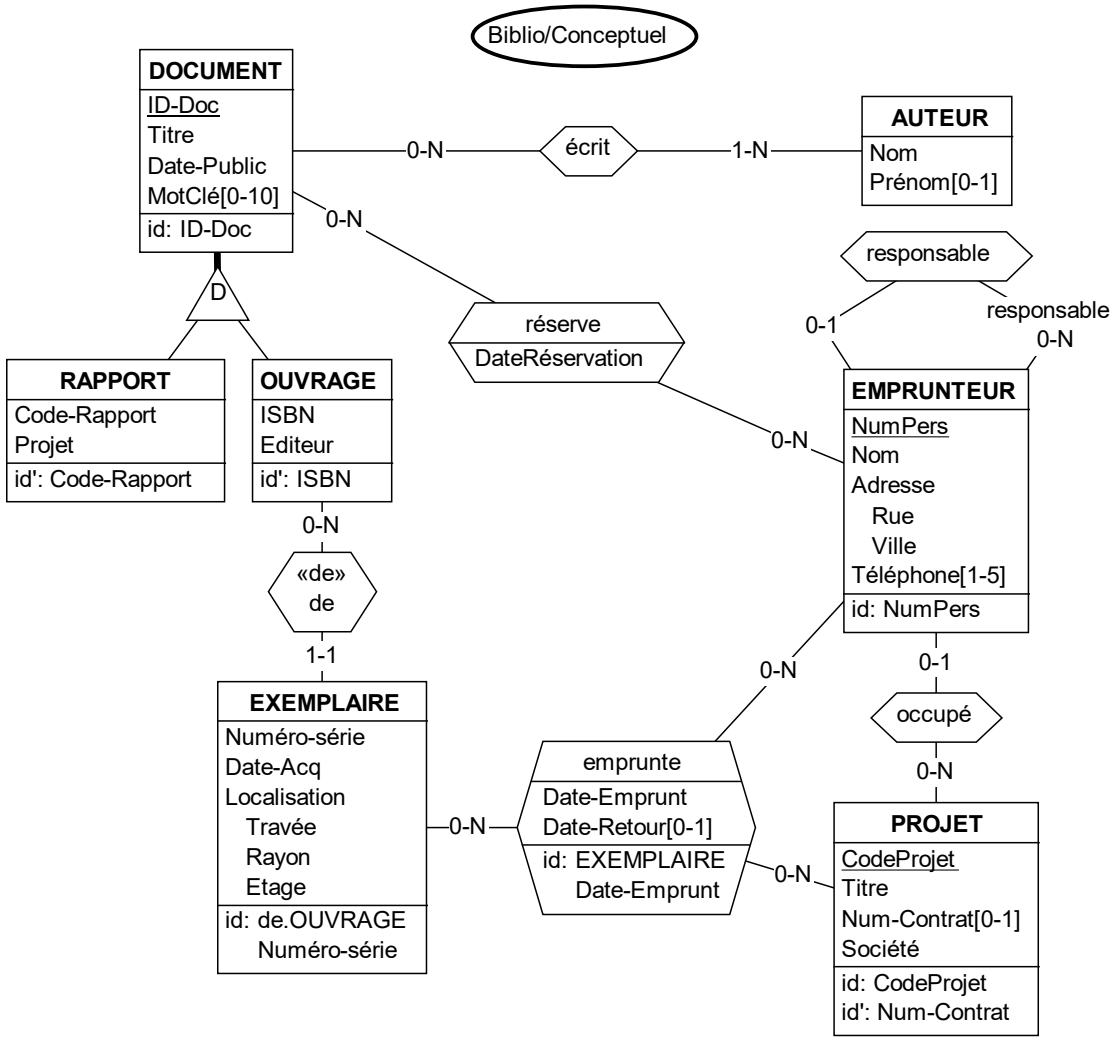
- regrouper les opérations similaires,
- retarder une opération de manière à ne l'exécuter qu'une seule fois (si possible),
- définir les itérations pour les constructions définies récursivement (attribut complexes, TA intervenant dans un identifiant hybride par exemple).

4.7 Plan de transformation pour SQL2

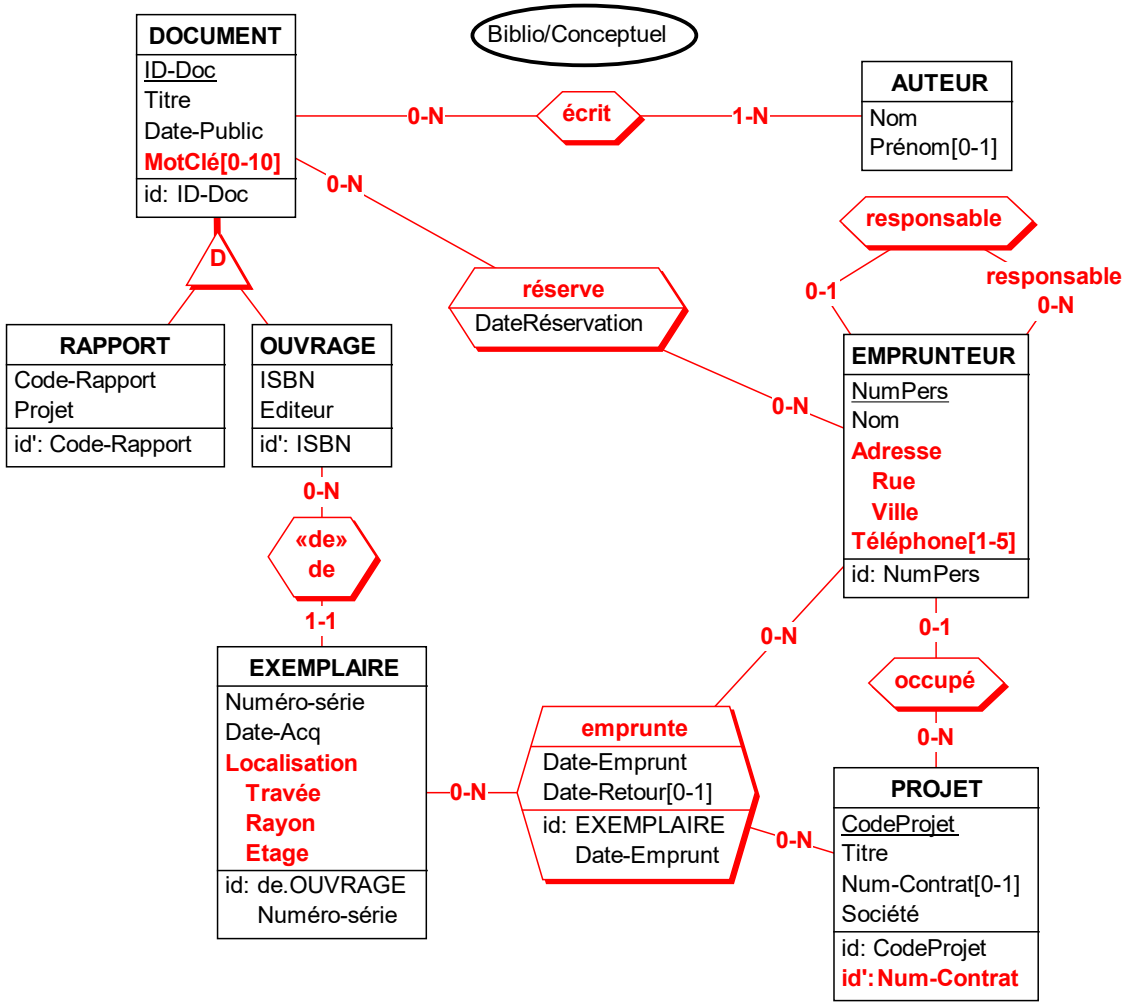


4.8 EXEMPLE DE CONCEPTION LOGIQUE

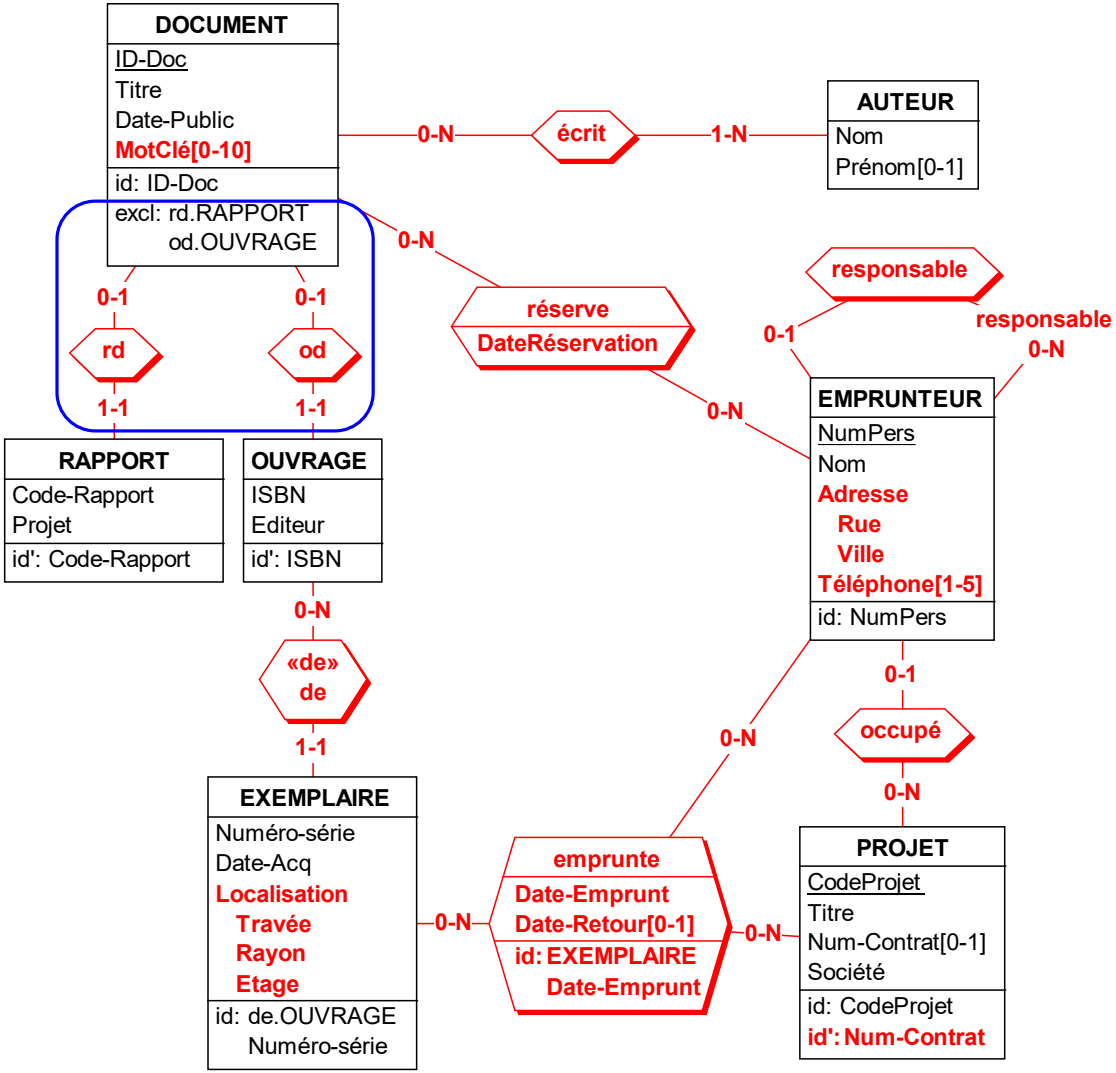
4.8 Exemple de conception logique - Le schéma conceptuel



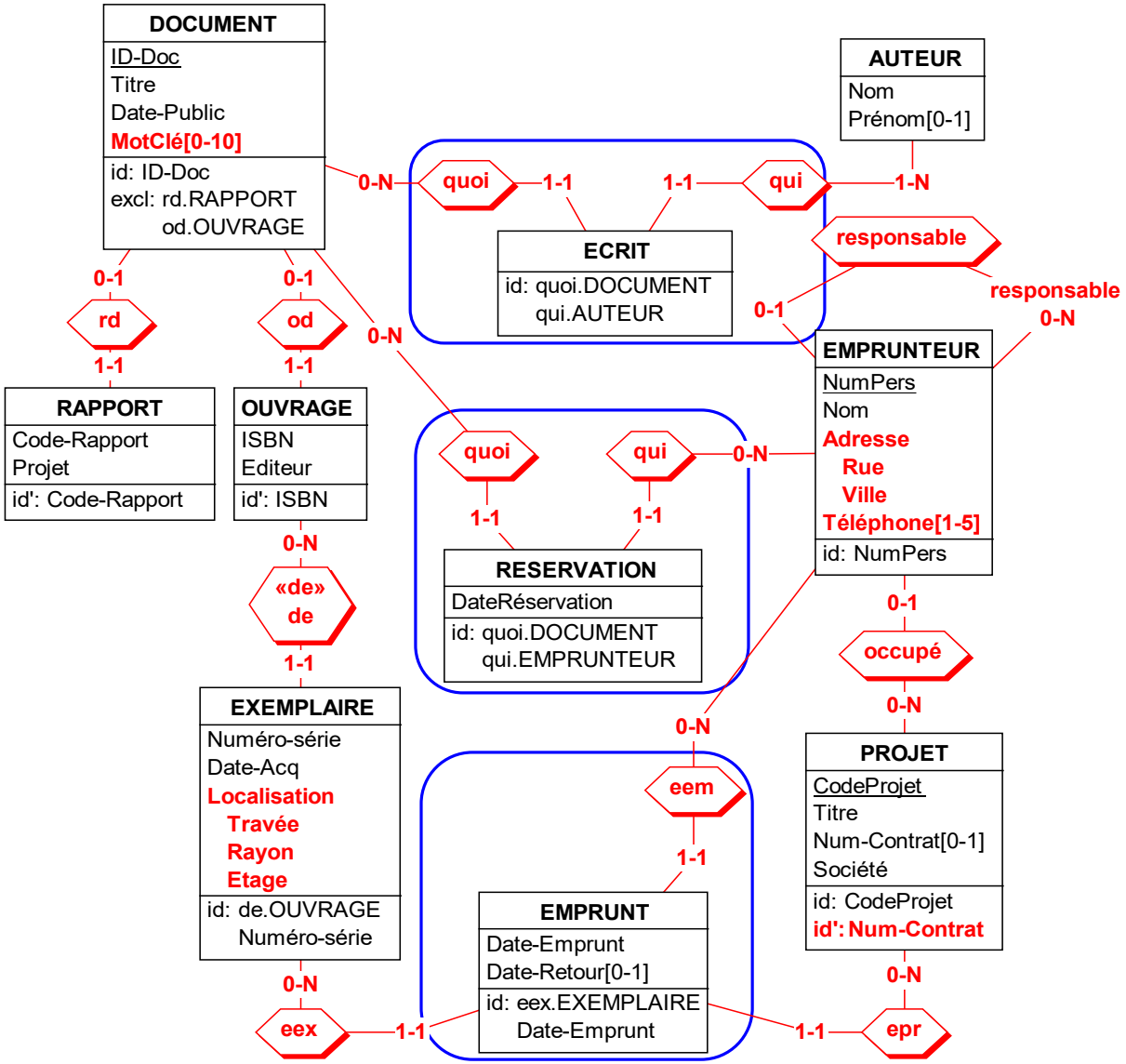
4.8 Exemple de conception logique - Les constructions invalides



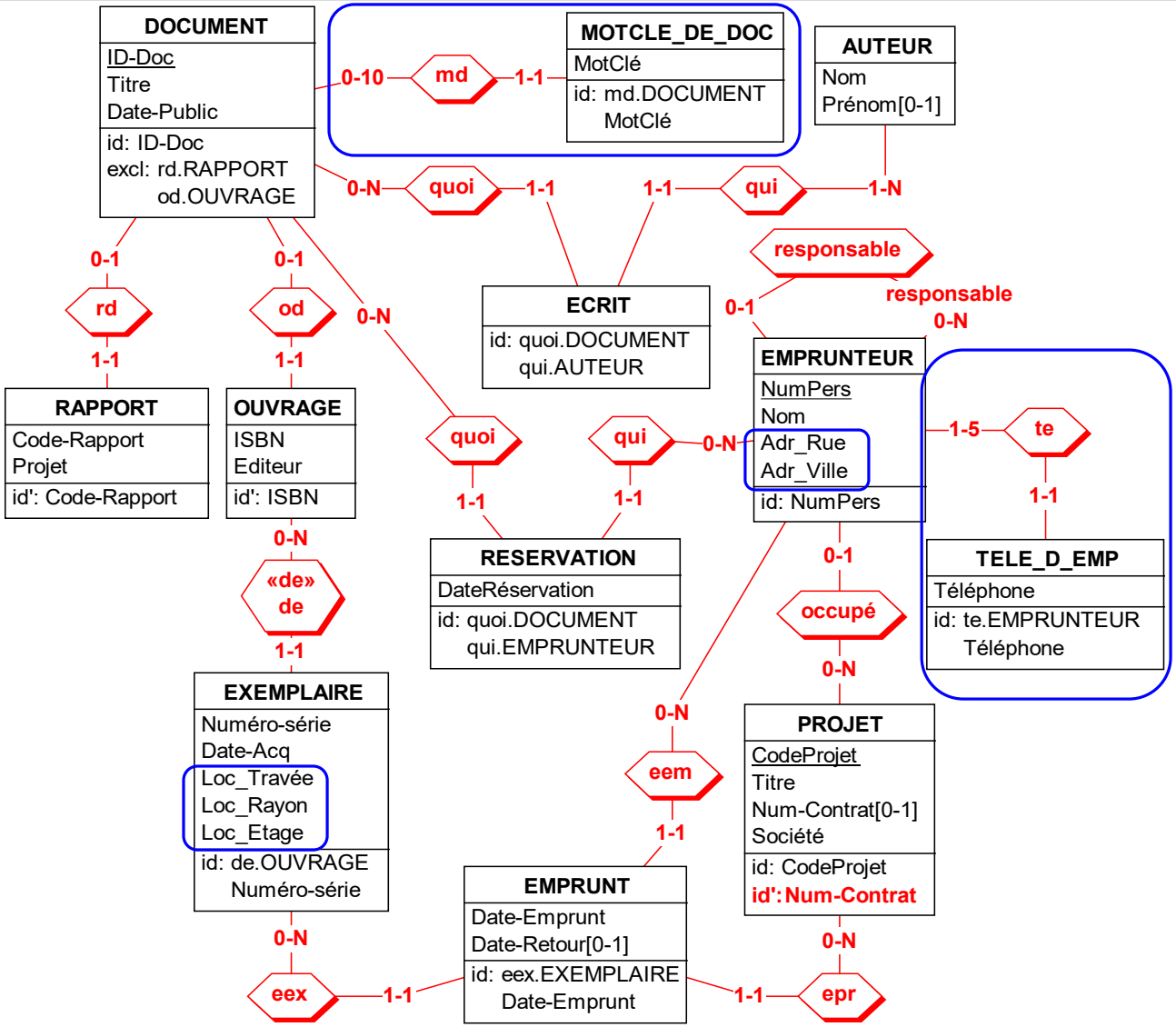
4.8 Exemple de conception logique - Traitement des relations *is-a*



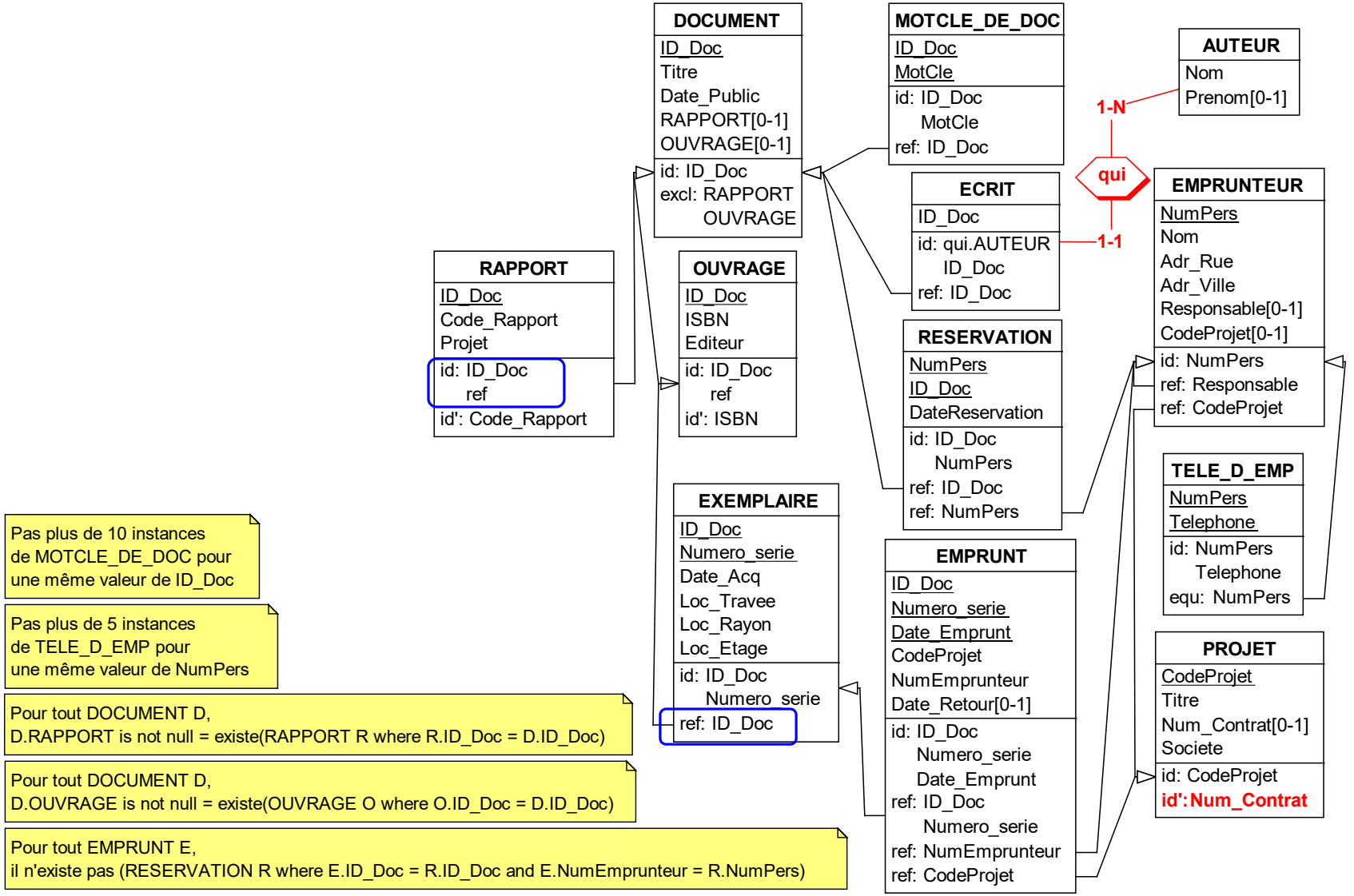
4.8 Exemple de conception logique - Traitement des TA complexes



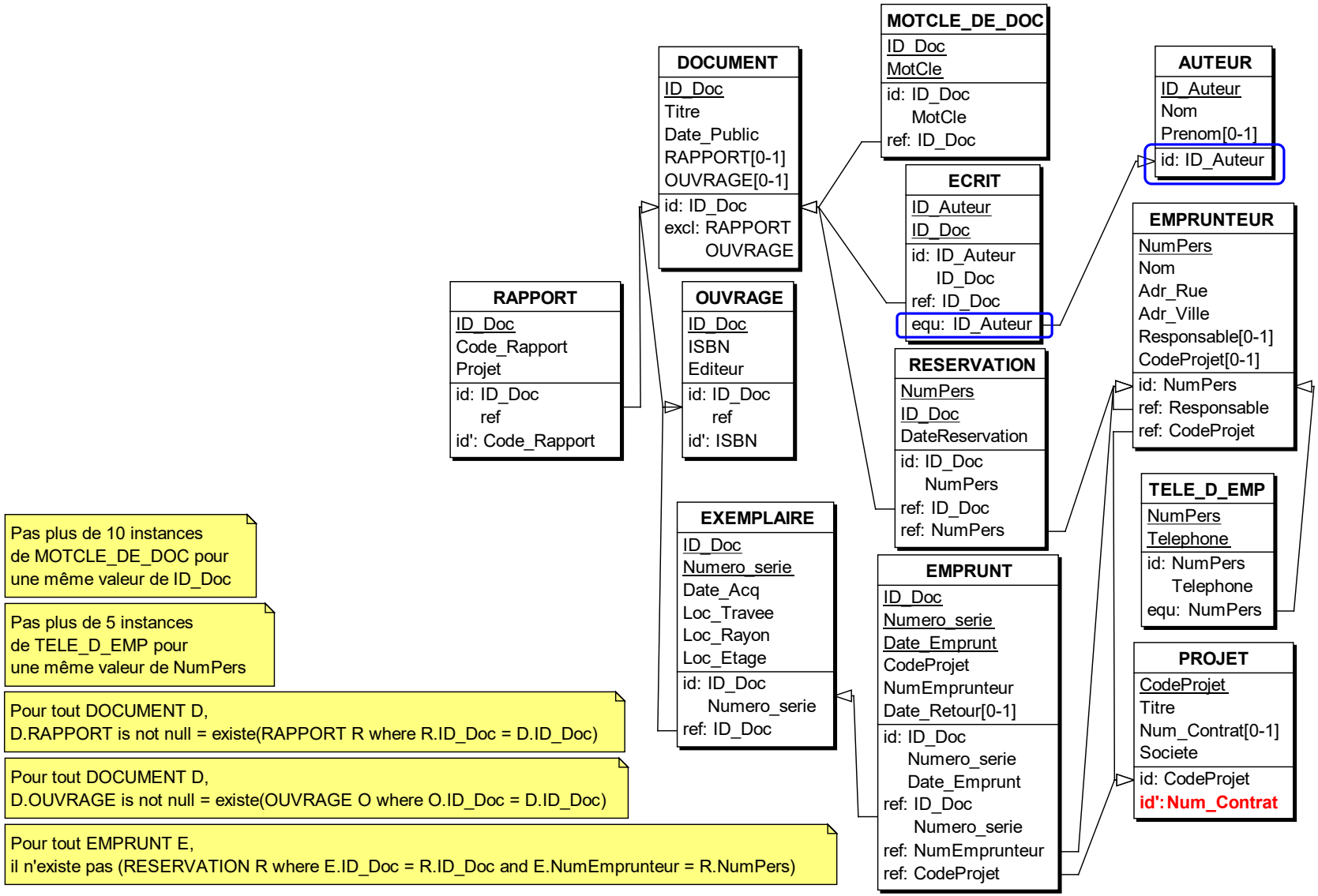
4.8 Exemple de conception logique - Traitement des attributs complexes



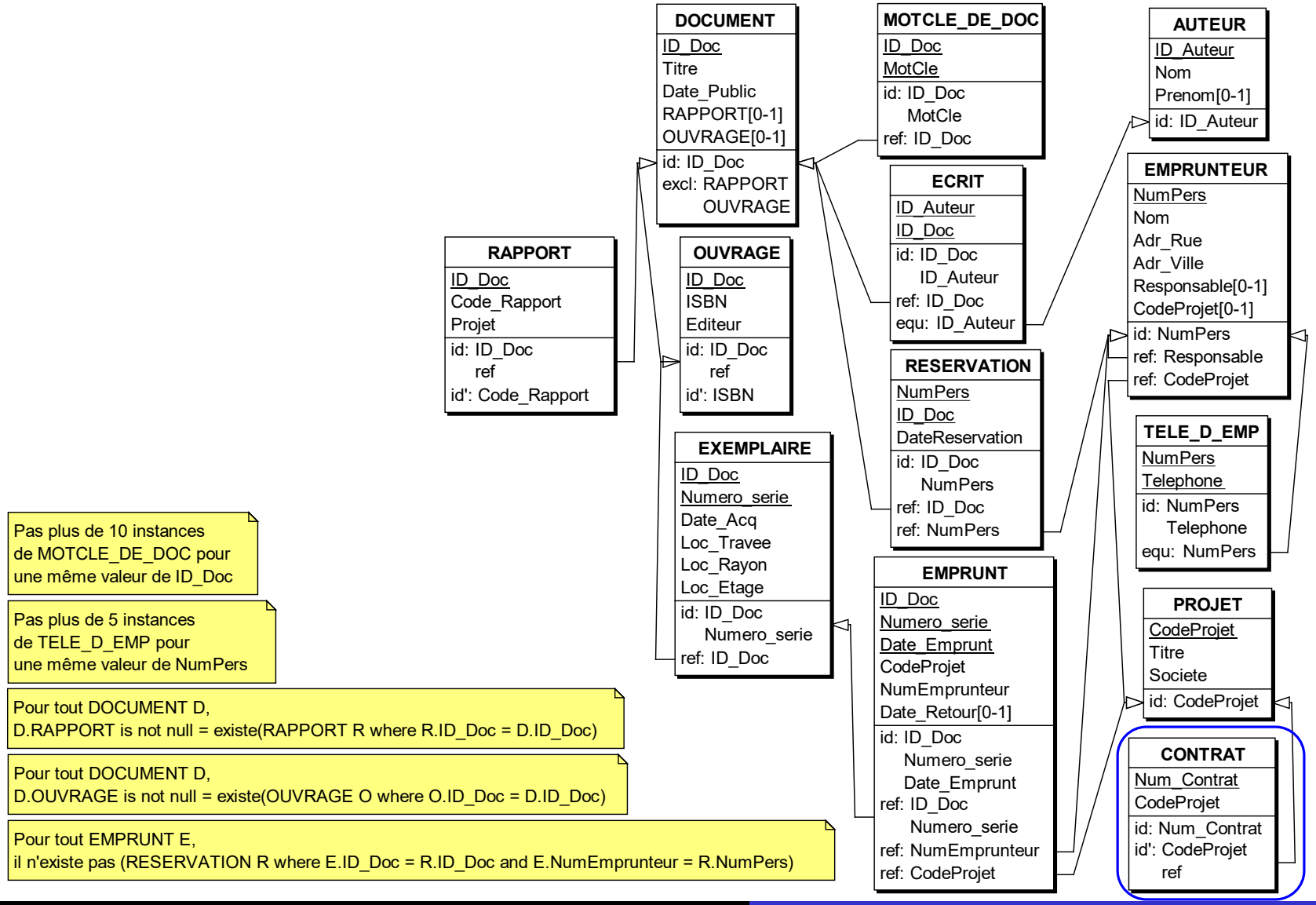
4.8 Exemple de conception logique - Traitement des TA fonctionnels (1)



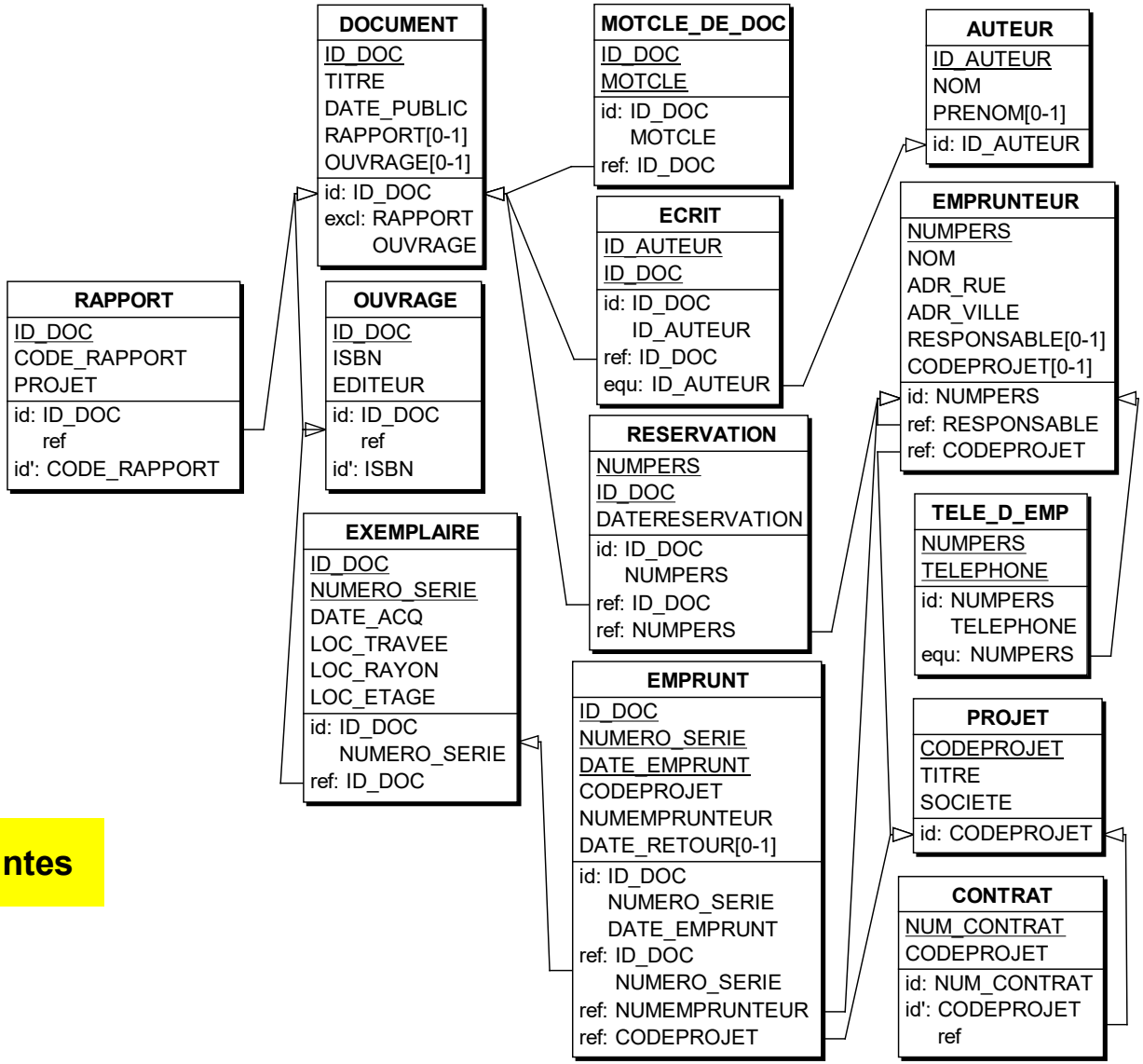
4.8 Exemple de conception logique - Traitement des TA fonctionnels (2)



4.8 Exemple de conception logique - Traitement id. facultatifs



4.8 Exemple de conception logique - Traitement des noms



+ contraintes

4.9 OUTILS POUR LA CONCEPTION LOGIQUE

Contenu

a) Transformations

b) Assistants de transformation

c) Assistant d'analyse

4.9 Outils pour la conception logique - Transformations

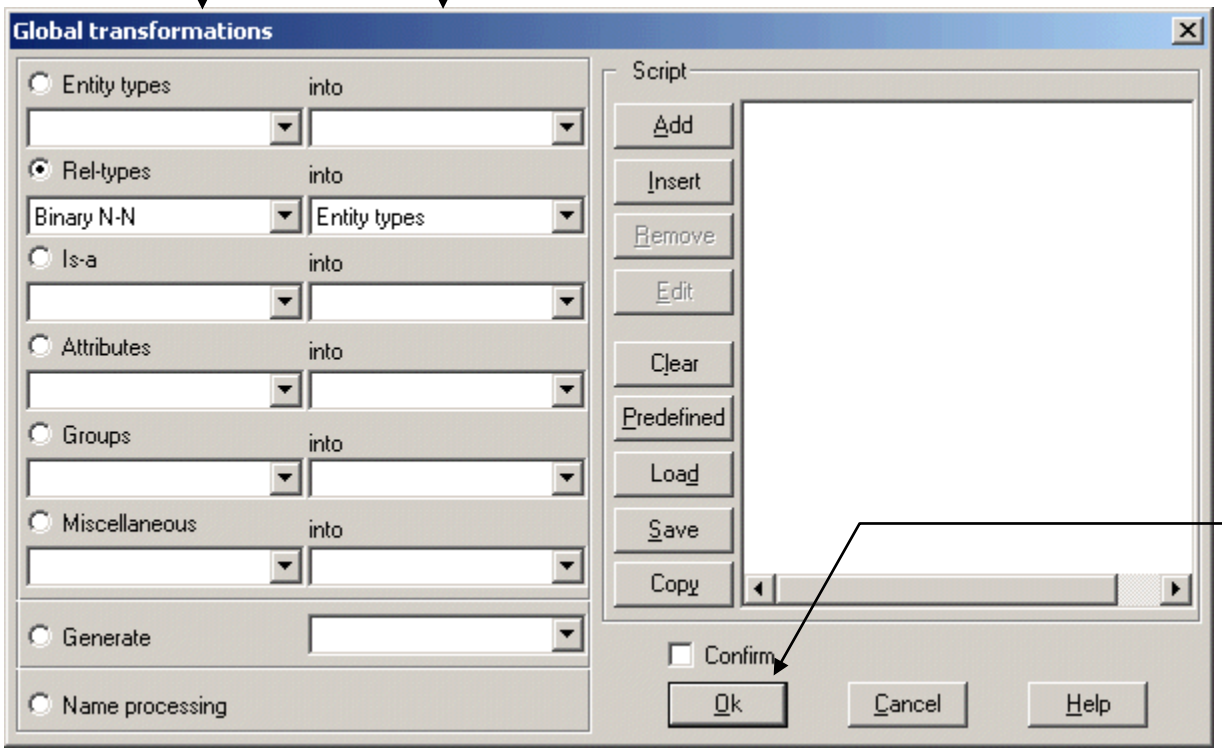
Quatre niveaux de transformation sont nécessaires

- **transformations élémentaires**
  T(O); la transformation T est appliquée à l'objet O du schéma S;
- **transformations composées**
  T(O), T = T3 o T2 o T1; la transformation T est construite par composition de transformations plus élémentaires;
- **transformations à base de prédicat** (*predicate-driven transformation*)
  $T_{(p)}$: la transformation T est appliquée à tous les objets du schéma courant qui vérifient le prédicat p;
- **transformation à base de modèle** (*model-driven transformation*)
  $T_{m1.m2}(S1)$: la transformation $T_{m1.m2}$ est appliquée au schéma S1 conforme au modèle M1 et produit un schéma S2 conforme au modèle M2; $T_{m1.m2}$ est définie par un plan de transformation constitué de transformations à base de prédicats. = *transformation inter-modèles*

4.9 Outils pour la conception logique - Assistant de transformation

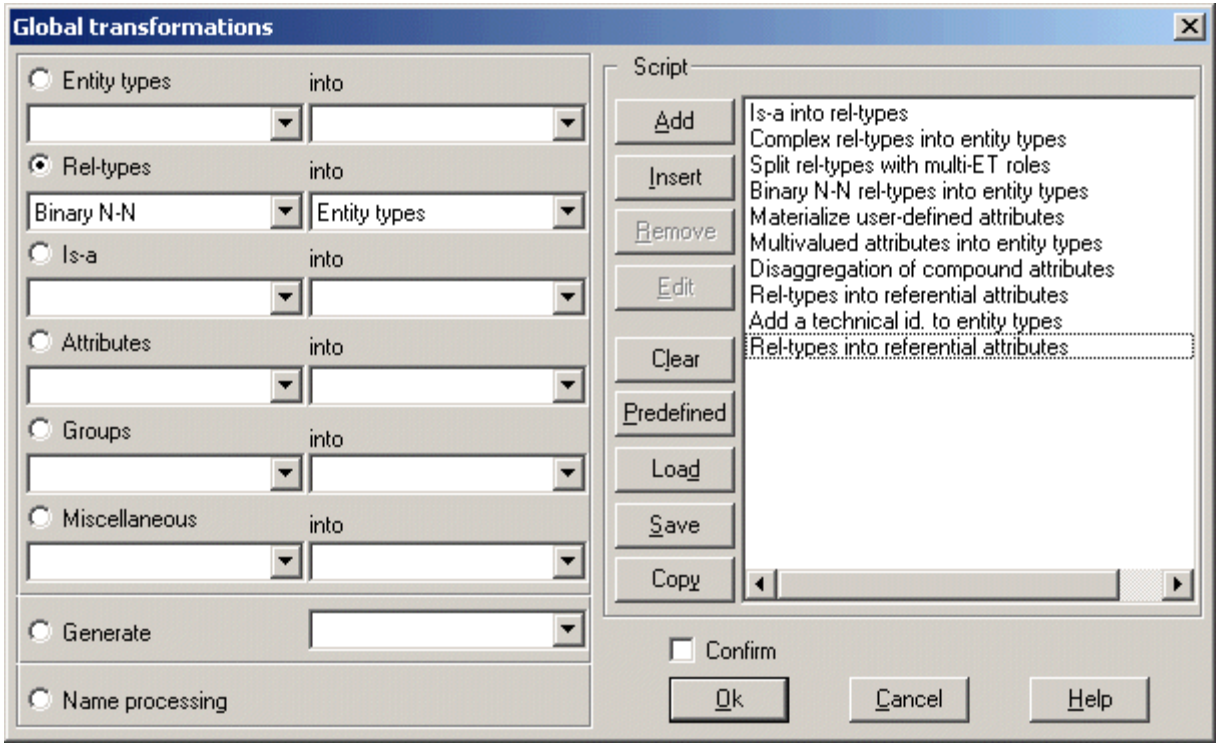
1. sélectionner un pattern = prédicat (p)

2. sélectionner une action (T)

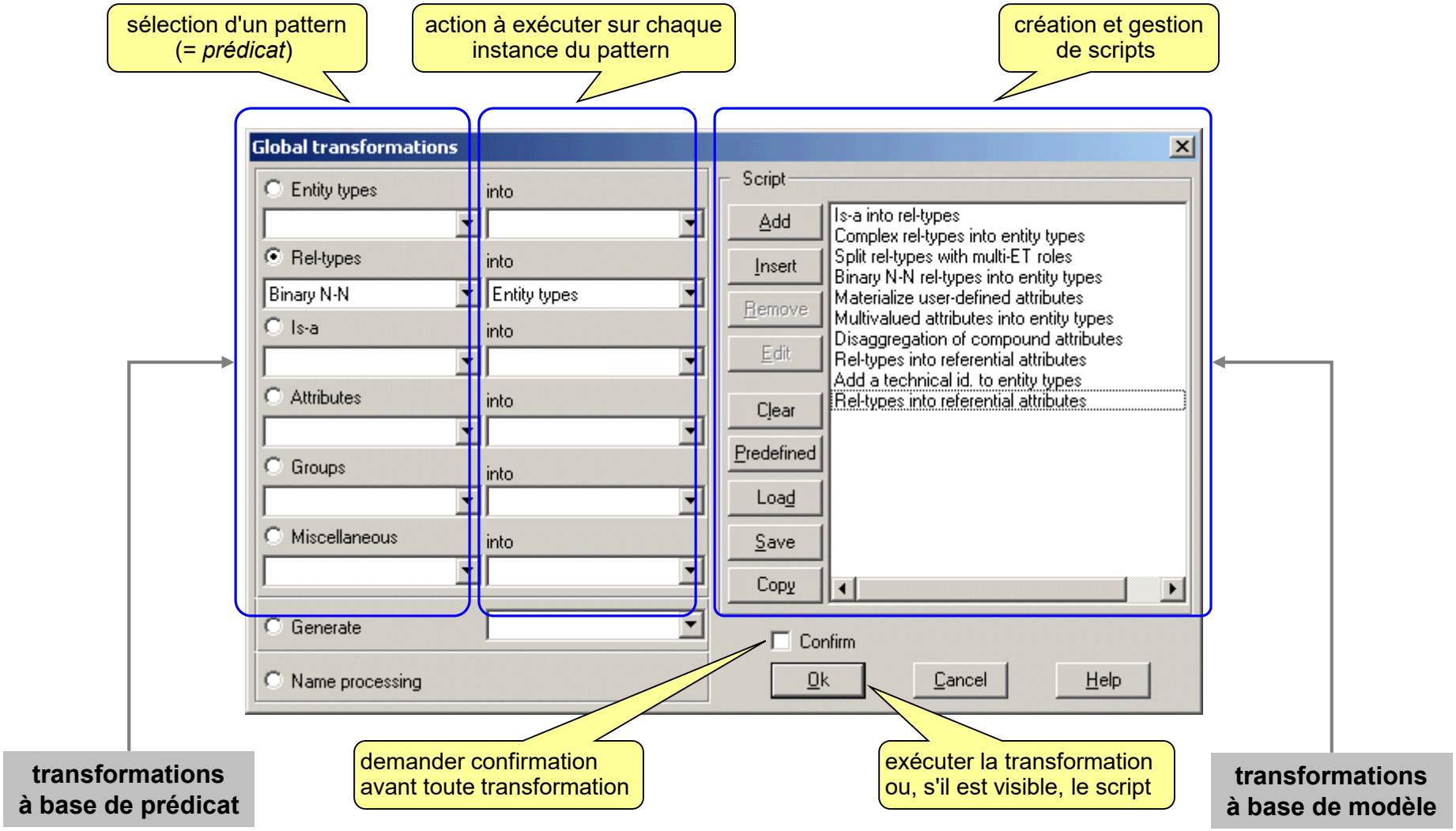


3. exécuter T(p)

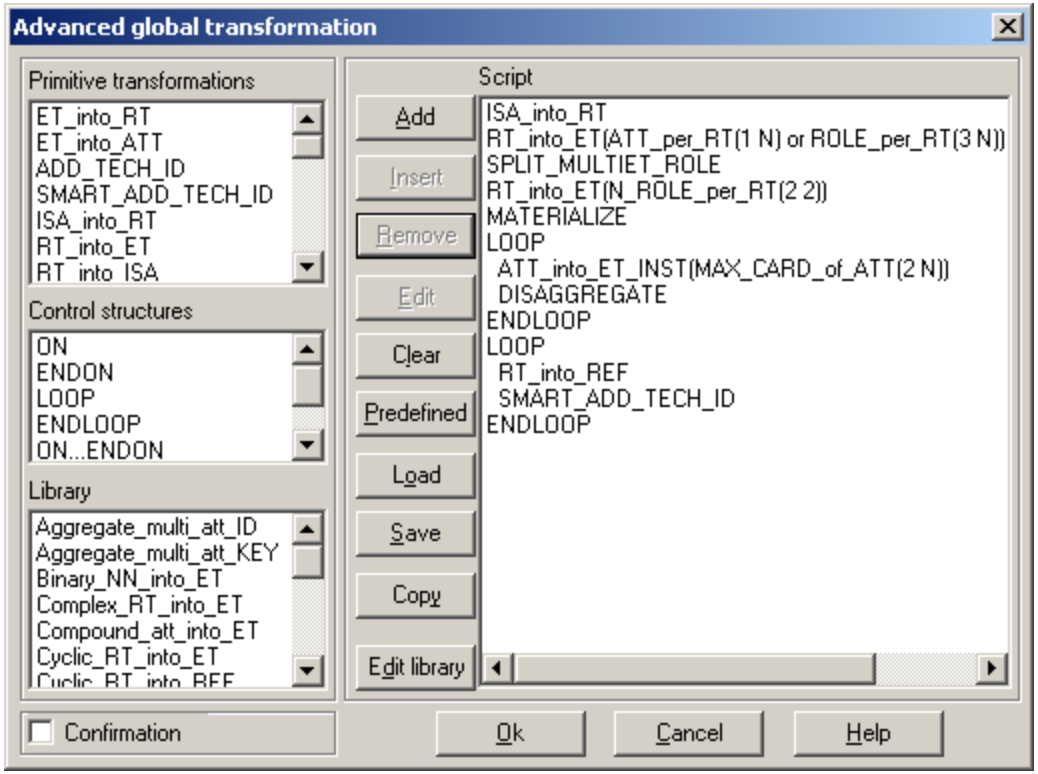
4.9 Outils pour la conception logique - Assistant de transformation



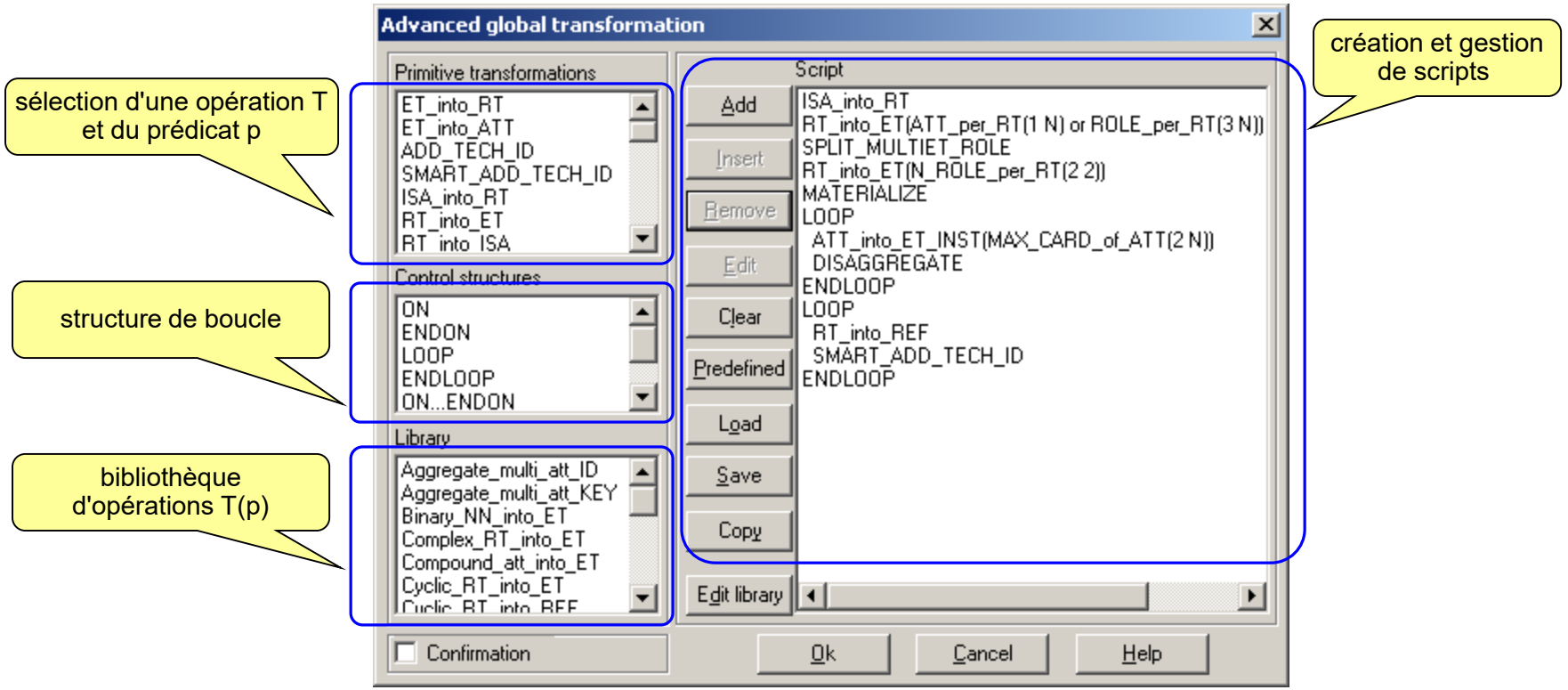
4.9 Outils pour la conception logique - Assistant de transformation



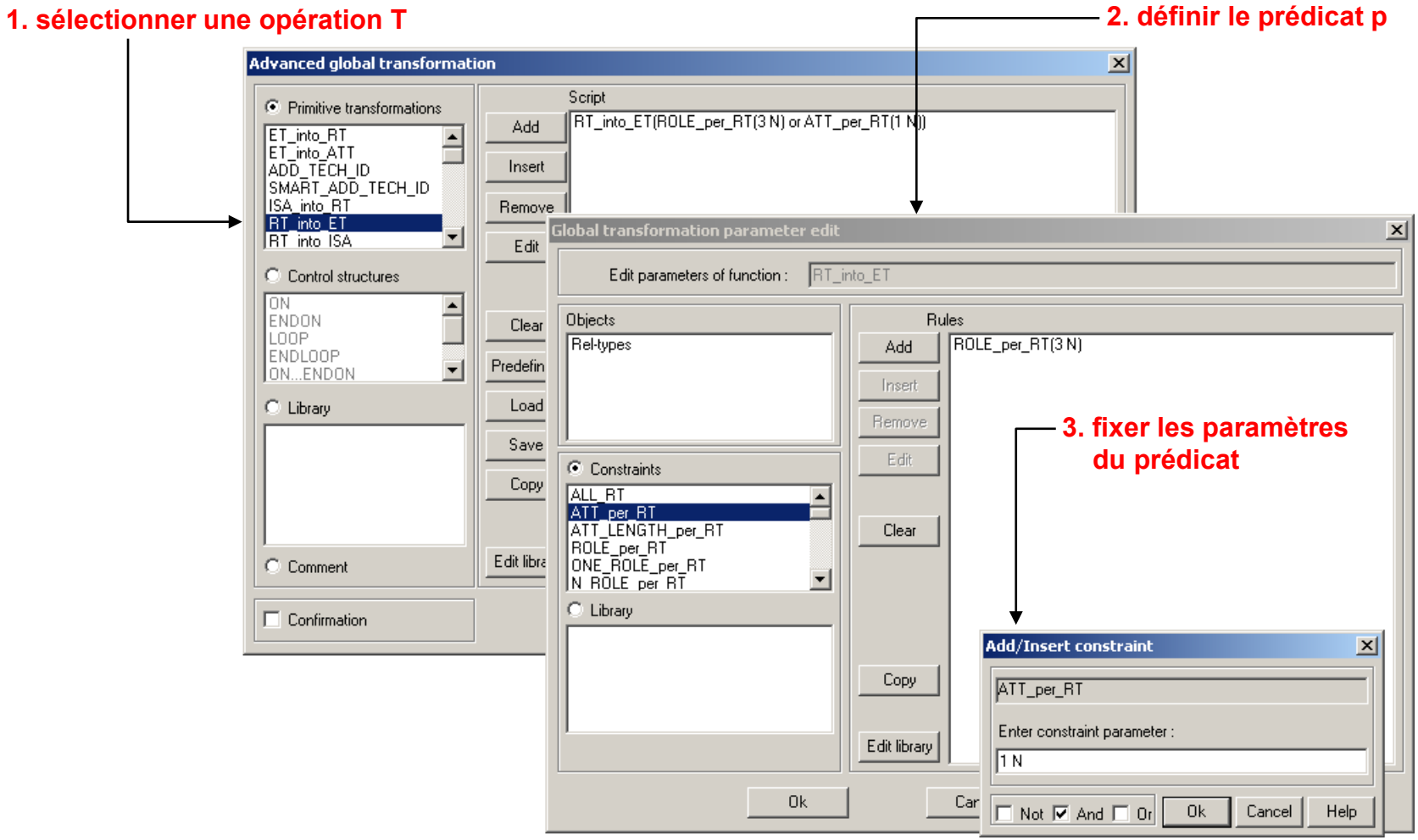
4.9 Outils pour la conception logique - Assistant de transformation avancé



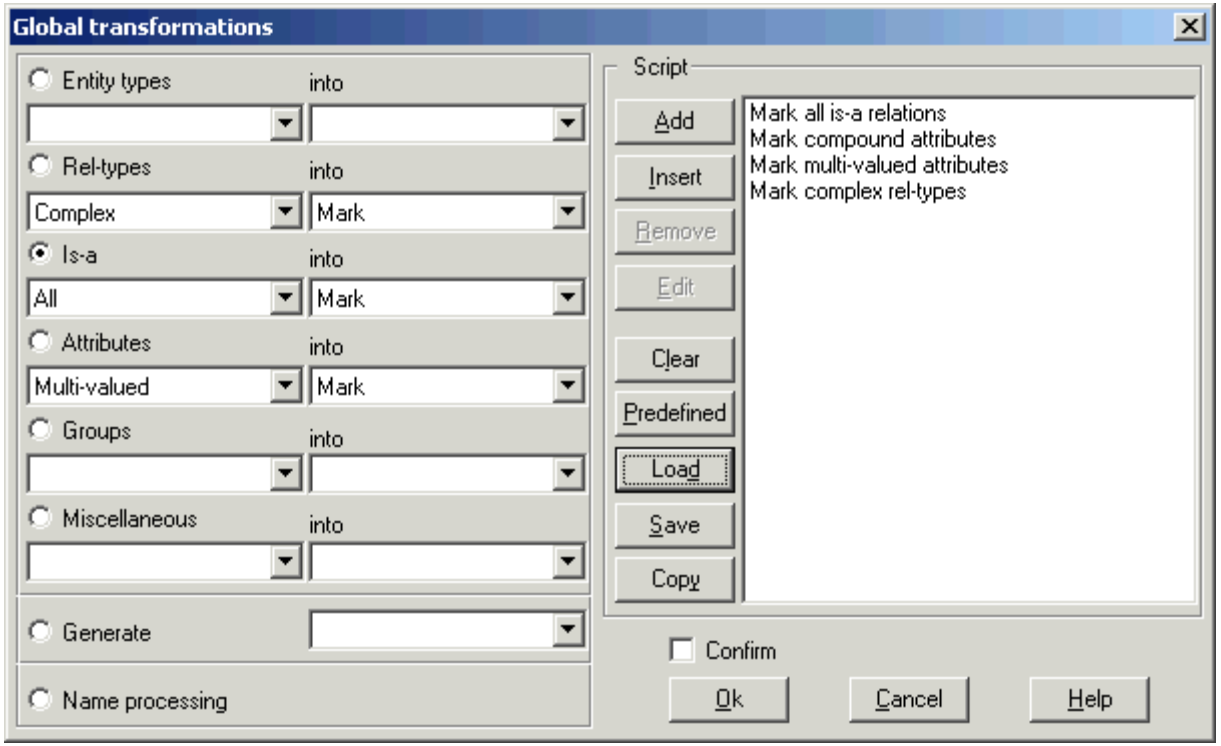
4.9 Outils pour la conception logique - Assistant de transformation avancé



4.9 Outils pour la conception logique - Assistant de transformation avancé

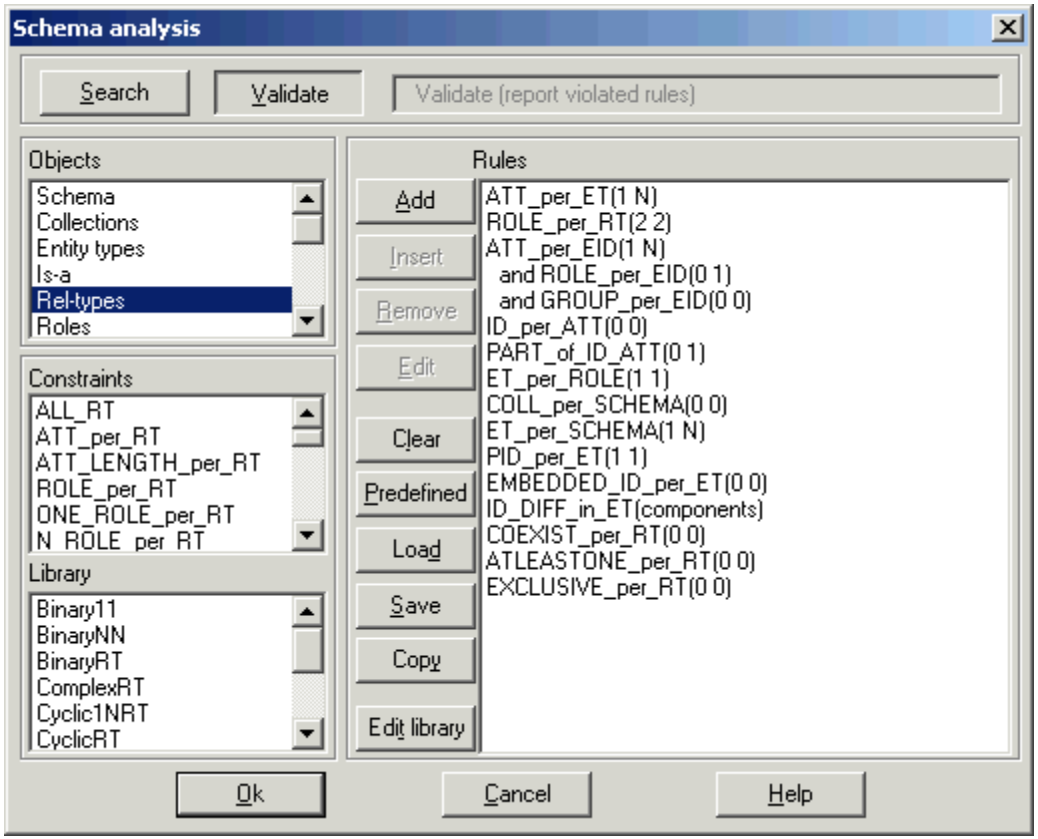


4.9 Outils pour la conception logique - Assistant d'analyse



L'assistant de transformation peut servir d'analyseur : on marque les objets sélectionnés au lieu de les transformer.

4.9 Outils pour la conception logique - Assistant d'analyse



Search : recherche les éléments qui satisfont une des règles

Validate : recherche les éléments qui violent une des règles

4.10 Extensions du modèle SQL3

Contenu

a) Les attributs complexes

b) Les relations is-a

4.10 Extensions du modèle SQL3

On observe qu'un schéma EA contient des constructions directement exprimables en SQL3 :

= constructions "*conformes*"

- *type d'entités → table*
- ***hiérarchies is-a simples*** *(disjonction, supertype unique, distribution unique)*
- *attribut simple → colonne*
- ***attribut composé*** *→ constructeur* ***row***
- ***attribut multivalué*** *→ constructeur* ***array***
- *identifiant primaire formé d'attributs → primary key*
- *identifiant secondaire formé d'attributs → prédicat* ***unique****()*

En revanche, les autres constructions demandent une conversion plus élaborée :

- *type d'associations → ?*
- *identifiant comportant un rôle → ?*
- *hiérarchies is-a complexes ?*
- *etc.*

= constructions "*non conformes*"

4.10 Extensions du modèle SQL3

Tout schéma SQL2 est un schéma SQL3

- toutes les règles précédentes sont d'application
- mais on peut vouloir profiter de la plus grande puissance du modèle SQL3

Apport de SQL3

- représentation directe des attributs complexes;
- représentation directes des relations is-a simples;
- autres possibilités mais ignorées ici.

4.10 Extensions du modèle SQL3

Les attributs complexes

Attribut composé

- peut être représenté par une colonne de type **row**;

Attribut multivalué

- peut être représenté par une colonne de type **array**;
- mais : constructeur **non ensembliste** !

⇒ taille limitée
valeurs ordonnées
unicité non gérée
valeurs manquantes possibles

- voir si on peut vivre avec ces caractéristiques et ces lacunes indésirables ! Par exemple les ignorer ou les gérer par des contraintes ou des triggers
- sinon, procéder comme en SQL2

4.10 Extensions du modèle SQL3

Les relations *is-a* simples

Relations is-a simples

- disjonction (D) ou partition (P) - donc pas de recouvrement
- hiérarchie simple (surtype unique) - donc pas de surtypes multiples
- répartition simple (surtype unique) - donc pas de répartitions multiples

⇒ **traduction directe (sans transformation) en**

- hiérarchies de TDU
- hiérarchie de tables

4.10 Extensions du modèle SQL3

Les relations *is-a* complexes

Relations is-a complexes

- recouvrement (libre ou T) - transformer (voir annexes E et G)
- surtypes multiples - transformer (voir annexes E et G)
- répartitions multiples - transformer (voir annexes E et G)

Mais … le schéma logique devient irrégulier, complexe, illisible ...

4.10 Extensions du modèle SQL3

Les relations *is-a* complexes

Règle pratique :

si **toutes** les relations *is-a* sont **naturellement simples** (sans tricher !)

et si cette propriété est **garantie à long terme**,

alors on utilise les constructions SQL3 (hiérarchies de TDU et de tables typées)

sinon on procède comme en SQL2

4.11 Quelques réflexions sur les plans de transformation

Contenu

a) Qualités d'un plan de transformation

b) Unicité des plans de transformation

c) Conception logique et conversion inter-modèles

d) Equivalence de schémas

4.11 Qualités d'un plan de transformation

Quelles sont les qualités d'un bon plan de transformation ?

- ***terminaison*** *: quel que soit le schéma source S1, la procédure se termine*
- ***conformité*** *: quel que soit le schéma source S1, le schéma résultant S2 est conforme au modèle M2*
- ***équivalence*** *: quel que soit le schéma source S1, les schémas S1 et S2 sont équivalents*
- ***idempotence*** *: P(P(S)) = P(S)*

(pour autant que S1 satisfasse les préconditions du plan)

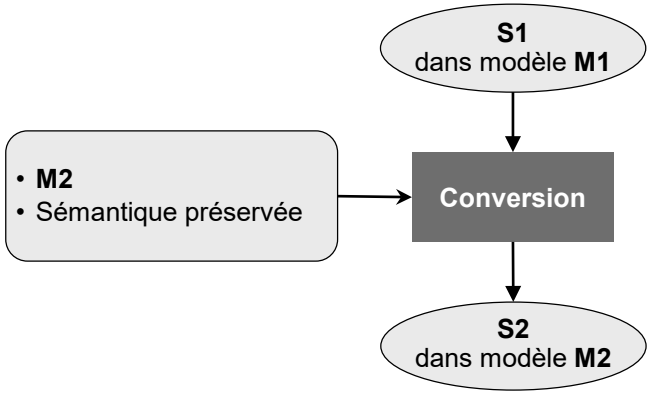
4.11 Unicité des plans de transformation

Le plan de transformation SQL2 est-il unique ?

- dépend du modèle logique (quelle variante de SQL2 ?),
- dépend du traitement choisi pour chaque construction invalide,
- dépend du degré de complétude des traitements (*Ciel ! on a oublié les rôles multitypes, les identifiants cycliques, les TDU et les delete/update modes des clés étrangères !*),
- dépend des préconditions ad hoc sur les schémas sources (*pas plus de 3 niveaux dans un attribut complexe*),
- dépend du degré de précision (constructions laissées à l'initiative du concepteur).

⇒ nombreux plans de transformation SQL2 possibles

4.11 Conception logique et transformations inter-modèles



La conception logique est un cas d'application du processus général de transformation inter-modèles (M1 ⇒ M2) :

conversion d'un schéma S1 exprimé dans le modèle M1

en un schéma équivalent S2 exprimé selon le modèle M2

4.11 Equivalence de schémas

Qu'est-ce que l'équivalence de deux schémas S1 et S2 ?

- S1 et S2 ont la même sémantique (*vague*)
- S2 et S2 décrivent le même domaine d'application (*mieux mais invérifiable*)
- on suppose qu'aux schémas S1 et S2 correspondent des instances; pour toute requête Q1 appliquée à S1, il existe une requête Q2 appliquée à S2 qui produit le même résultat, et inversement; (*excellent mais difficile à vérifier*)
- il existe une chaîne de transformations C1 qui, appliquée à S1, produit S2, et il existe une autre chaîne C2 qui, appliquée à S2, produit S1, telles que :
  S1 = C2(C1(S1)) (*mieux mais insuffisant; il existe des contre-exemples*)
- à la chaîne C1 correspond une transformation de données c1 qui, appliquée à une instance s1 de S1, produit une instance s2 de S2, et il existe une autre chaîne c2 qui, appliquée à une instance s2 de S2, produit une instance de S1, telles que :
  s1 = c2(c1(s1)) (*beaucoup mieux ! Mais encore insuffisant*)
- C1, C2, c1, c2 sont telles que :
  S2 = C1(S1); S1 = C2(S2);
  s2 = c1(s1); s1 = c2(s2); (*parfait : vérifiable et opérationnel*)

Méthode : utiliser des transformations à sémantique constante ou réversibles

Fin du module 4

Module suivant :

5. Conception physique relationnelle