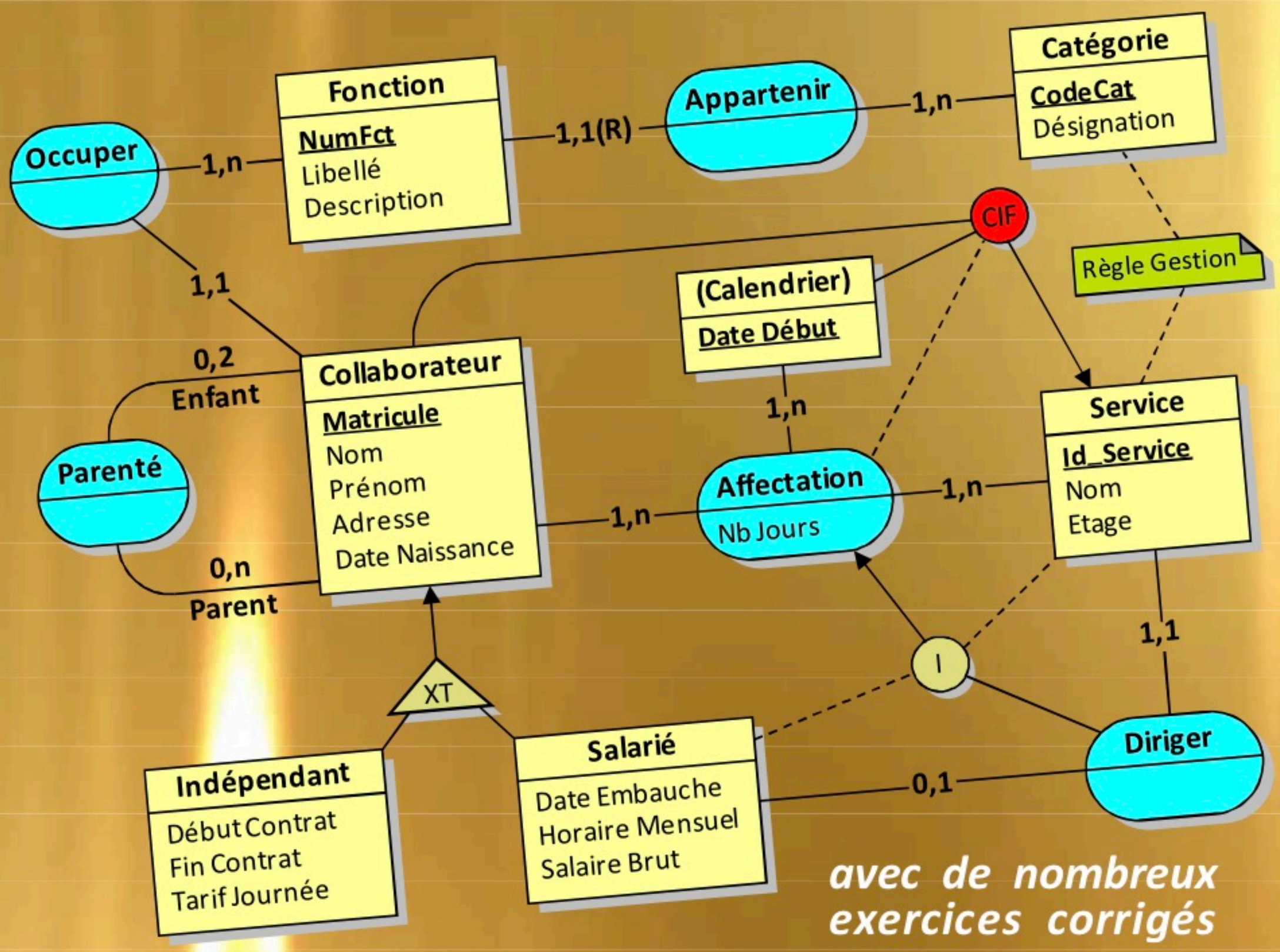
Touchez le centre de la page pour découvrir d'autres options de lecture ou quitter le livre.

Patrick Bergougnoux

Modélisation Conceptuelle de Données

– UNE DÉMARCHE PRAGMATIQUE –



avec de nombreux exercices corrigés

Touchez le centre de la page pour découvrir d'autres options de lecture ou quitter le livre.

Patrick Bergougnoux est Professeur des Universités au Département Informatique de l'IUT de Toulouse III où il enseigne l'Analyse et la Conception des Systèmes d'Information ainsi que les Bases de Données.
En contact permanent avec le milieu professionnel, il a mené de nombreux projets opérationnels.

L'objectif de cet ouvrage n'est pas d'être exhaustif et de traiter tout ce qui peut exister en termes de modélisation de bases de données. Bien au contraire, l'idée maîtresse est de proposer une démarche pédagogique la plus simple possible, permettant de réaliser des modèles conceptuels de données sans faire appel à de quelconques connaissances préalables, et en limitant les aspects théoriques à leur strict minimum. Une approche concrète et pratique est privilégiée avec, en support, de nombreux schémas, illustrations et exercices corrigés.

Les modèles proposés dans cet ouvrage s'appuient essentiellement sur le formalisme entité-association, mais un chapitre entier est consacré aux diagrammes de classes UML, avec une mise en correspondance complète des deux formalismes.

Ce livre s'adresse aux étudiants de DUT, BTS, licences, masters, classes préparatoires, écoles d'ingénieurs, à leurs professeurs, ainsi qu'à tous les professionnels désireux d'adopter une approche conceptuelle simple et rigoureuse pour la modélisation de leurs bases de données.

Pour cela, une démarche pragmatique et progressive est développée en cinq étapes :

- Recensement des informations
- Bases de la modélisation
- Informatisation des modèles
- Modélisation avancée
- Études de cas complètes

Avec de nombreux exercices corrigés tout au long de l'ouvrage.

L'ensemble des modèles conceptuels de données proposés dans cet ouvrage ont été réalisés avec *Looping*, logiciel de modélisation gratuit et libre d'utilisation.

https://www.looping-mcd.fr

TOUS LES BÉNÉFICES DES VENTES DE CE LIVRE SERONT REVERSÉS À

ACTION CONTRE LA FAIM

Sommaire

Recensement des informations

- Analyse des flux d'information
- Modèle organisationnel de flux
- Dictionnaire des données
- Exercices de recensement

Bases de la modélisation

- Répartition en classes d'entités
- Association des classes d'entités
- Règles de cohérence des modèles
- Exercices de modélisation

Informatisation des modèles

- Mise en relation des tables
- Modèle logique de données
- Requêtes SQL de création de tables
- Reprise exercices avec MLD et SQL

Modélisation avancée

- Identifiant relatif
- Association n-aire
- Compléments de modélisation
- Diagramme de classes UML

Études de cas complètes

- Association sportive et culturelle
- Compagnie d'assurances
- Entreprise générale du bâtiment
- Société de services

La simplicité est la sophistication suprême

Léonard de Vinci

Patrick Bergougnoux

Modélisation Conceptuelle de Données

– UNE DÉMARCHE PRAGMATIQUE –

Contact avec l'auteur

Patrick.Bergougnoux@univ-tlse3.fr

Le Code de la propriété intellectuelle et artistique n'autorisant, aux termes des alinéas 2 et 3 de l'article L.122-5, d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective » et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale, ou partielle, faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause, est illicite » (alinéa 1er de l'article L.122-4). Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait donc une contrefaçon sanctionnée par les articles 425 et suivants du Code pénal.

L'autorisation de reproduire intégralement ou partiellement le présent ouvrage, sur quelque support que ce soit, doit être obtenue auprès de l'Éditeur ou du Centre Français d'Exploitation du Droit de Copie, 20 rue des Grands-Augustins, 75006 Paris.

© Patrick Bergougnoux, 2019 ISBN livre imprimé : 979-10-699-3087-2

À mes parents,
à mon épouse et mes enfants.

Avant-propos

Il existe déjà bon nombre d'ouvrages de qualité concernant la modélisation de données. Alors pourquoi celui-ci ?

Tout d'abord, l'objectif de ce livre n'est pas d'être exhaustif et de traiter tout ce qui peut exister en termes de modélisation de bases de données. Bien au contraire, le principe majeur de cet ouvrage est de proposer une démarche la plus simple possible, permettant de réaliser des modèles conceptuels de données sans faire appel à de quelconques connaissances préalables, et en limitant les aspects théoriques à leur strict minimum. Une approche concrète et pratique est privilégiée avec, en support, de nombreux exercices corrigés.

Par ailleurs, les démarches les plus souvent proposées ont pour postulat de départ la construction d'une base de données informatisée : la modélisation est alors une simple phase préalable à la définition de la structure de la base de données. De plus, l'implémentation de cette base diffère quasi-systématiquement du modèle conceptuel du fait de la prise en compte de choix techniques souvent incompatibles avec une approche purement conceptuelle.

Conséquence : dans la grande majorité des cas, les développeurs négligent l'étape conceptuelle et écrivent directement le schéma relationnel de leur base.

Pour remédier à tout cela, ce livre propose donc **une approche pragmatique**, alliant simplicité de réalisation des modèles conceptuels et adéquation avec la base de données correspondante.

Structure de l'ouvrage

Abréviations

Tout d'abord, il convient de préciser quelques abréviations qui pourront être utilisées dans cet ouvrage et que l'on retrouve systématiquement dans un contexte de modélisation de bases de données :

- SI : Système d'Information
- MOF : Modèle Organisationnel de Flux
- MCD : Modèle Conceptuel de Données
- MLD : Modèle Logique de Données
- E/A : Entité/Association
- UML : Unified Modeling Language (Langage de Modélisation Unifié)
- CIF : Contrainte d'Intégrité Fonctionnelle
- BD : Base de Données
- SGBD : Système de Gestion de Base de Données
- SQL : Structured Query Language (Langage de requête structuré)

Leur signification précise et leur utilisation seront détaillées au fur et à mesure de la démarche proposée dans ce livre.

Pictogrammes

Différents pictogrammes viennent régulièrement ponctuer le livre dans le but de mettre en évidence certains aspects spécifiques à des moments précis de la démarche de modélisation :

Définition, conseil ou remarque importante à retenir pour la bonne compréhension de la démarche.

Question à un moment clé de la démarche pour mettre en évidence une problématique importante (toujours suivie de la réponse correspondante).

Attention ! Mise en avant d'un point crucial : erreur à éviter, étape à ne pas oublier, … avec des conséquences importantes pour la suite de la démarche.

Astuce concernant le logiciel ***Looping*** : explication sur la façon de mettre en œuvre un point particulier d'un modèle avec le logiciel de modélisation.

© Patrick Bergougnoux

Une approche pédagogique à technicité croissante

Alors que le domaine de la modélisation de données nécessite habituellement de bonnes bases mathématiques (théorie des ensembles, algèbre relationnelle, …), le principe essentiel de la démarche proposée dans cet ouvrage est de ne jamais rentrer dans des considérations trop théoriques et de proposer une méthode simple, pratique et fonctionnelle pour réaliser les modèles.

Le livre est structuré en cinq parties composées de quatre chapitres chacune. L'approche est très progressive, chaque chapitre s'appuyant sur les connaissances acquises aux chapitres précédents.

La première partie consiste à identifier et recenser l'ensemble des informations contenues dans le système étudié. Les techniques de base de la modélisation sont ensuite étudiées afin de produire de manière simple les premiers modèles conceptuels de données. La partie suivante fait le lien entre concept et pratique avec l'évolution des modèles vers la base de données correspondante : la boucle est alors bouclée avec une démarche partant du recensement des informations jusqu'à leur implémentation au sein d'une base de données. La quatrième partie revient ensuite sur la modélisation conceptuelle avec des modèles plus complexes utilisant toutes les possibilités offertes par le formalisme ; l'utilisation des diagrammes de classes UML est également traitée. La dernière partie est consacrée à la mise en œuvre de quatre études de cas complètes.

L'ouvrage s'appuie, chapitre après chapitre, sur de nombreux exercices corrigés qui illustrent systématiquement les différents aspects de la démarche.

Quelques références bibliographiques

Comme indiqué précédemment, le but de ce livre n'est pas d'être exhaustif : il se focalise sur une démarche simple permettant aux lecteurs de réaliser des modèles conceptuels de données sans forcément maitriser tous les aspects théoriques et tout l'état de l'art relatif à la modélisation des bases de données.

D'autres ouvrages peuvent donc venir en complément pour celles et ceux qui souhaitent approfondir certains aspects de ce vaste domaine. Voici les références qui me paraissent intéressantes pour compléter ce livre :

- « *The Entity-Relationship Model : Toward a Unified View of Data* »,
  Peter Chen – ACM Transactions on Database Systems, Vol. 1-1, 1976.
- « *Bases de données et systèmes relationnels* »,
  Claude Delobel et Michel Abida – Dunod informatique, 1983.
- « *Ingénierie des systèmes d'information : Merise, deuxième génération* »,
  Dominique Nanci et Bernard Lespinasse – Vuibert, 4e édition, 2001.
- « Modélisation des bases de données – UML et les modèles E/A »,
  Christian Soutou – Eyrolles, 4e édition, 2017.

Cette liste n'est bien évidemment pas exhaustive.

Utilisation du logiciel de modélisation *Looping*

Il existe de nombreux logiciels permettant de réaliser des modèles conceptuels de données. Parmi les principaux, citons « *PowerAMC* », version française de PowerDesigner de Sybase, et « *Win'Design* » de la société française Cecima. Tous deux supportent parfaitement les formalismes entité-association et diagrammes de classes UML. La capacité de ces outils va bien au-delà de la seule modélisation conceptuelle de données, ce qui les rend certes très complets, mais également très complexes. Leur coût est également conséquent.

Dans le cadre de mon enseignement à l'IUT Informatique de Toulouse III, j'ai lancé, en collaboration avec Éric Mathet de la société Soft & technique informatique, le développement d'un logiciel spécifique dédié à la modélisation conceptuelle de données.

Ce logiciel, ***Looping***, propose bien sûr les fonctionnalités classiques de modélisation que l'on retrouve dans tous les logiciels du même type :

- Modélisation entité-association (E/A).
- Associations réflexives avec définition de rôles.
- Classes d'entités fictives non générées dans le MLD.
- Définition des propriétés et des types de données.
- Héritage avec spécialisations et généralisations.
- Contraintes d'intégrité fonctionnelle (CIF) et inter-associations.
- Règles de gestion sur entités et associations.
- Insertion de textes, graphiques et images au sein des modèles.

Certaines fonctions sont un peu plus originales et ont été développées dans un esprit pédagogique pour permettre aux utilisateurs de mieux comprendre, entre autres, la correspondance E/A et UML, ainsi que les liens entre les MCD, les schémas relationnels et les requêtes de création des bases de données correspondantes :

- Diagramme de classes UML généré simultanément avec bascule instantanée E/A ⇔ UML.
- Possibilité de transformer une association en classe d'entités avec mise en œuvre d'identifiants relatifs.
- Modèle logique de données (MLD) sous forme de schéma relationnel affiché en temps-réel au fur et à mesure de la construction du MCD.
- Affichage également en temps-réel des requêtes SQL de création des tables de la base de données.
- Exportation de scripts SQL pour la création de bases de données MS-Access, MySQL, Oracle et PostgreSQL.

© Patrick Bergougnoux

- Exportation des MCD dans tous les formats images, ou directement dans le presse-papier pour un copier-coller instantané dans d'autres documents.
- Affichage multi-vue et multi-zoom des modèles E/A et UML.
- Possibilité de créer des modèles de flux et autres diagrammes avec des connexions d'objets libres.
- Présentation entièrement personnalisable avec de nombreux paramètres d'affichage des modèles.

Quelques limites :

- La modélisation des diagrammes de classes UML se limite à l'équivalence avec les modèles E/A : les autres aspects de la modélisation UML ne sont pas implémentés.
- Les requêtes SQL générées automatiquement se limitent à la création des tables et du schéma relationnel de la base de données et n'intègrent pas, par exemple, les procédures et déclencheurs (triggers) liés aux règles de gestion et aux contraintes d'intégrité inter-associations. Des instructions SQL gérant ces contraintes peuvent néanmoins être rajoutées au script de génération du schéma relationnel grâce à la définition de règles contenant le code approprié.
- En fait, ***Looping*** est avant tout un logiciel de modélisation conceptuelle de données avec une approche pédagogique pour la génération des MLD et des tables de bases de données.

Par ailleurs, ***Looping*** possède certains avantages particulièrement intéressants :

- Tout d'abord, il est entièrement **gratuit** : développé en interne à l'IUT informatique de Toulouse III, il est totalement libre d'utilisation tant pour l'enseignement que pour un usage personnel ou professionnel.
- Son installation est extrêmement facile et rapide : une simple copie du fichier « *Looping.exe* » (moins de 5 Mo) le rend immédiatement exécutable depuis n'importe quel support (disque local, serveur, clé USB, ...).
- Il fonctionne sous Windows 7 et supérieur, 32 et 64 bits, ainsi que sous macOS et Linux avec un outil tel que Wine.
- Son interface est conforme aux derniers standards Windows et permet une mise en œuvre intuitive et très rapide des modèles.
- Et enfin, sa fiabilité a été éprouvée par quelques milliers d'utilisateurs.

Le logiciel se télécharge en quelques secondes sur : **www.looping-mcd.fr**

Même si tous les modèles mis en œuvre dans cet ouvrage sont réalisés avec ***Looping***, la démarche proposée est parfaitement transposable à l'ensemble des outils de conception du marché.

Quelques conseils pour la lecture de ce livre

Suivre l'ordre des chapitres

Beaucoup d'ouvrages techniques ne prennent pas en compte la progression pédagogique et, dans un souci d'exhaustivité, se présentent comme une accumulation de concepts dont il est parfois difficile d'extraire la cohérence globale ; il est alors courant de les lire en diagonale, de sauter certains chapitres, pour ne s'intéresser qu'à un aspect particulier. Ces ouvrages n'en restent pas moins très intéressants car ils sont souvent complets et un lecteur averti y trouvera son intérêt.

Dans notre cas, le plan du livre est très progressif, chaque nouveau concept s'appuyant sur ceux précédemment traités. Aussi surprenant que cela puisse paraitre, la première lecture doit s'effectuer un peu comme un roman qui amène le lecteur d'une situation de novice jusqu'à la compréhension des problèmes et des solutions associées.

L'ordre peut alors paraitre déroutant avec une première étape pour réaliser des modèles de base, puis un passage par des aspects plus techniques, et enfin un retour vers une modélisation plus avancée. En effet, le principal objectif est d'amener le lecteur à très vite concrétiser l'approche conceptuelle, de manière à bien comprendre, avec des modèles les plus simples possibles, les liens entre conception et pratique. Et c'est seulement après cette étape que la dernière partie de l'ouvrage revient sur les aspects les plus avancés de la modélisation.

Par conséquent, à moins que vous ne soyez déjà expérimenté, il est fortement conseillé de suivre l'ordre du livre. Bien sûr, une fois que vous avez bien appréhendé la démarche, il est intéressant de pouvoir revenir sur n'importe quel chapitre pour approfondir la maitrise de certaines notions.

Pratiquer au fur et à mesure de la lecture

Chaque concept traité dans le livre est accompagné d'exemples et d'exercices permettant de le concrétiser. N'hésitez pas à reproduire les différents modèles proposés afin de bien ancrer vos connaissances et ce, au fur et à mesure de la lecture des différents chapitres. N'attendez pas d'avoir terminé le livre pour commencer à pratiquer. Utilisez votre logiciel de modélisation pour réaliser les MCD de l'ouvrage, mais également pour mettre en œuvre vos propres exercices ; et n'hésitez pas à générer la base de données correspondante pour bien comprendre les liens entre les niveaux conceptuel, logique et physique.

Lorsque vous reprenez les exercices du livre, refaites les modèles sans regarder la correction : un exercice corrigé parait souvent évident ; partir d'une feuille blanche l'est beaucoup moins…

© Patrick Bergougnoux

Remerciements

Ce livre est le résultat d'une collaboration de plus d'une vingtaine d'années avec mes collègues du département informatique de l'IUT de Toulouse. Année après année, nous avons affiné notre enseignement en nous focalisant sur le côté pragmatique de la démarche pédagogique, et en nous faisant violence pour laisser de côté tous les aspects trop théoriques qui, même s'ils sont importants pour nous, enseignants, rebutent trop souvent les étudiants et les détournent de la matière. Ce livre est le résultat de cette démarche collective et j'y associe donc tous les collègues qui ont, directement ou indirectement, contribué à ce travail.

L'ouvrage est construit autour d'un outil de modélisation « maison », ***Looping***, réalisé en collaboration avec la société Soft & technique informatique, et en particulier avec Éric Mathet qui a su parfaitement répondre à toutes les exigences propres à un logiciel certes très professionnel, mais aussi et surtout particulièrement adapté à une démarche pédagogique.

Je souhaite également remercier ma famille qui partage mon quotidien, me soutient sans réserve dans tous mes projets, et a largement contribué à la relecture et à la finalisation de ce livre.

Enfin, merci à vous, chers lecteurs ! Si vous avez des questions ou remarques concernant cet ouvrage ou sur l'utilisation du logiciel ***Looping***, n'hésitez pas à me contacter : **Patrick.Bergougnoux@univ-tlse3.fr**

Bonne lecture !

© Patrick Bergougnoux

Table des matières

Recensement des informations 1

1 Analyse des flux d'information 3

Le système d'information 3
De quoi s'agit-il ? 3
Les sous-systèmes dans les organisations 4
L'informatisation 7

Identifier l'information 7
Une tâche plus complexe qu'il n'y parait 7
Information ou état d'âme 8

De l'audit au modèle de flux 8
Audit et compte-rendu d'interview 8
Abstraction et formalisation 9
Vers un modèle focalisé sur l'inventaire des flux 10

Ce qu'il faut retenir 10

2 Modèle organisationnel de flux 11

Structure du modèle 11
Une première étape simple mais déterminante 11
L'acteur 12
Le flux 12

Un objectif bien précis 13
Un modèle volontairement limité 13
Un modèle graphiquement épuré 13
Quelques erreurs à éviter 14

Mise en pratique avec un exemple simple 16
Compte-rendu d'interview 16
Domaine d'étude et acteurs 16
Recherche des échanges 16
Réalisation du modèle organisationnel de flux 17

Ce qu'il faut retenir 18

3 Dictionnaire des données 19

Analyse des informations issues des flux 19
Représentation des flux 19
Repérage des informations 22
Constitution d'un tableau des rubriques 25

Évolution vers le dictionnaire des données 26
Gestion des rubriques calculables 26
Pièges à éviter 27
Paramètres 30
Complément du tableau des rubriques 31
Dictionnaire des données 32

Ce qu'il faut retenir 32

4 Exercices de recensement 33

Gestion des achats d'une administration 34
Compte-rendu d'interview 34
Documents annexes 35
Dictionnaire des données 39

Société canine 40
Compte-rendu d'interview 40
Documents annexes 41
Tableau des rubriques 44
Dictionnaire des données 45

© Patrick Bergougnoux

Bibliothèque universitaire 46
Compte-rendu d'interview *46*
Tableau des rubriques *47*
Dictionnaire des données *47*

Bases de la modélisation 51

5 Répartition en classes d'entités 53

Répartition des données 53
Nécessité d'organiser les données en sous-ensembles *53*
Trouver des objets significatifs du domaine d'étude *54*
Éléments syntaxiques et représentation *55*

Règle d'affectation à une classe d'entités 56
Affectation exclusive de toutes les rubriques *56*
Principale règle d'affectation *56*
Application à l'exemple *56*

Affectation des rubriques orphelines 59
Vérifier qu'aucune classe d'entités ne convient *59*
Pas une, mais deux classes concernées *59*
Définition d'une nouvelle classe d'entités *60*

Ce qu'il faut retenir 60

6 Association des classes d'entités 61

Sémantique des modèles 61
Nommer les associations existantes *61*
Rajouter les associations décrites lors de l'analyse *62*

Cardinalités 62
Étude des liens possibles dans une association *62*
Notation des cardinalités dans le modèle de données *65*
Choix des verbes *67*

Associations réflexives 68
Liens entre entités d'une même classe *68*
Association réflexive symétrique *69*
Association réflexive asymétrique *69*
Association réflexive hiérarchique *70*

Ce qu'il faut retenir 70

7 Règles de cohérence des modèles 71

Règle 1 71
Une et une seule affectation par rubrique *71*
Énoncé de la règle *72*

Règle 2 72
Une seule valeur par rubrique pour une entité ou une association *72*
Énoncé de la règle *72*

Règle 3 72
Même valeur de rubrique pour plusieurs entités d'une classe *72*
Premier énoncé de la règle et ses limites *74*
Nouvel énoncé de la règle *76*
Conséquence : règle 3bis *76*

Règle 4 77
Jamais de rubrique dans une association avec patte simple *77*
Énoncé de la règle et son explication *77*

Règle 5 80
Jamais deux cardinalités 1,1 pour une même association *80*
Énoncé de la règle *81*

Ce qu'il faut retenir 82

8 Exercices de modélisation 83

Gestion des achats d'une administration 84
Rappel du dictionnaire des données *84*
Répartition des données dans les classes d'entités *84*
Modèle conceptuel de données *86*
Ce que le modèle met en évidence *89*

Société canine 90
Rappel du dictionnaire des données *90*
Répartition des données dans les classes d'entités *90*
Modèle conceptuel de données *92*
Ce que le modèle met en évidence *95*

Bibliothèque universitaire 96
Rappel du dictionnaire des données *96*
Répartition des données dans les classes d'entités *96*
Modèle conceptuel de données *97*
Ce que le modèle met en évidence *98*

Guide gastronomique 99
Compte-rendu d'interview *99*
Tableau des rubriques *99*
Dictionnaire des données *100*

© *Patrick Bergougnoux*

Répartition des données dans les classes d'entités *101*
Modèle conceptuel de données *102*
Ce que le modèle met en évidence *103*

Représentations théâtrales 104
Compte-rendu d'interview *104*
Tableau des rubriques *104*
Dictionnaire des données *105*
Répartition des données dans les classes d'entités *106*
Modèle conceptuel de données *107*
Ce que le modèle met en évidence *109*

Location de logements 110
Compte-rendu d'interview *110*
Tableau des rubriques *110*
Dictionnaire des données *111*
Répartition des données dans les classes d'entités *112*
Modèle conceptuel de données *113*
Ce que le modèle met en évidence *115*

Agence de travail temporaire 116
Compte-rendu d'interview *116*
Tableau des rubriques *116*
Dictionnaire des données *117*
Répartition des données dans les classes d'entités *118*
Modèle conceptuel de données *119*
Ce que le modèle met en évidence *120*

Gendarmes et voleurs 121
Compte-rendu d'interview *121*
Tableau des rubriques *121*
Dictionnaire des données *122*
Répartition des données dans les classes d'entités *123*
Modèle conceptuel de données *124*
Ce que le modèle met en évidence *125*

Informatisation des modèles 129

9 Mise en relation des tables 131

Représentation des données 131
Format tabulaire *131*
Représentation graphique des liens *133*

Référencement d'une entité au sein d'une autre classe 133
Création de nouvelles colonnes de liaison 133
Propriétés nécessaires aux identifiants des classes d'entités..... 135
Choix de l'identifiant 137
Modèle conceptuel de données informatisé 139

Table de correspondance.....140
Limites du référencement direct d'une table vers une autre 140
Nécessité de rajouter une table de correspondance 142
Le dilemme de la cardinalité 0,1 143
Le doute sur les cardinalités 1,1 et 1,n 144

Ce qu'il faut retenir 144

10 Modèle logique de données 145

Représentation de la structure de données.....145
Modèle graphique 145
Schéma relationnel textuel 146
Remarque concernant le libellé des clés étrangères 147

Introduction des types de données.....148
Des types variés et différents en fonction des SGBD 148
Usage spécifique du type « COUNTER » 150
Interdiction de la valeur « NULL »..... 150
Affichage des types dès le MCDi..... 151

Génération automatique du MLD 152
Un passage du MCDi au MLD facilement automatisable..... 152
Un processus en trois étapes simples 152

Ce qu'il faut retenir 152

11 Requêtes SQL de création de tables 153

Utilisation de SQL 153
Un langage normalisé 153
Un langage d'interrogation structuré..... 154

Création du schéma relationnel.....154
Requêtes de création de tables 154
L'instruction « CREATE TABLE »..... 154
Exécution des requêtes..... 156

Ce qu'il faut retenir 156

© Patrick Bergougnoux

12 Reprise exercices avec MLD et SQL 157

Gestion des achats d'une administration 158
MCDi *158*
MLD *158*
Requêtes SQL *159*
Schéma relationnel Access *160*

Société canine 161
MCDi *161*
MLD *161*
Requêtes SQL *162*
Schéma relationnel Access *163*

Bibliothèque universitaire 164
MCDi *164*
MLD *164*
Requêtes SQL *164*
Schéma relationnel Access *165*

Guide gastronomique 166
MCDi *166*
MLD *166*
Requêtes SQL *167*
Schéma relationnel Access *168*

Représentations théâtrales 169
MCDi *169*
MLD *169*
Requêtes SQL *170*
Schéma relationnel Access *171*

Location de logements 172
MCDi *172*
MLD *172*
Requêtes SQL *172*
Schéma relationnel Access *173*

Agence de travail temporaire 174
MCDi *174*
MLD *174*
Requêtes SQL *175*
Schéma relationnel Access *176*

Gendarmes et voleurs 177
MCDi *177*
MLD *177*
Requêtes SQL *178*
Schéma relationnel Access *179*

Modélisation avancée 183

13 Identifiant relatif 185

Utilisation poussée des clés étrangères 185
Mise en évidence de la problématique avec un exemple simple 185
Mise en œuvre au sein du MCDi 187
Requêtes SQL 188
Un usage très spécifique 188

Décomposition d'une association 189
Problèmes d'historisation 189
Transformation d'une association en classe d'entités 190
Équivalence stricte de la décomposition 191
Associer une classe d'entités à une association 192

Gestion de l'irréductibilité 195
Réduction éventuelle d'une clé primaire 195
Clé primaire et dépendance fonctionnelle 197

Récapitulatif sur l'usage des identifiants relatifs 197
Une solution répondant à tout type de modélisation 197
Transgresser en connaissance de cause 198

Ce qu'il faut retenir 198

14 Association n-aire 199

Les risques d'utilisation d'associations n-aires 200
Représentation graphique 200
MLD d'une association ternaire 200
Exemples de mauvaise utilisation d'associations n-aires 201
Présence d'une cardinalité simple dans une association n-aire 203

Décomposition systématique des associations n-aires 204
Des problèmes de cardinalités difficiles à appréhender 204
Détection des problèmes de clé minimale 207

L'historisation et la tentation des tripattes 209
Utilisation d'une classe d'entités de type Calendrier 209
Classe d'entités fictive 211
À utiliser avec précaution 213

Les contraintes d'intégrité fonctionnelle 213
Une alternative à la décomposition 213
Exemples d'utilisation de CIF 213

Ce qu'il faut retenir 214

© Patrick Bergougnoux

15 Compléments de modélisation 215

Héritage : spécialisation et généralisation 215
Rubriques communes à plusieurs classes d'entités 215
Sur-type et sous-type 217
Spécialisation ou généralisation 218
Attention aux abus 220

Contraintes inter-associations 222
Mise en œuvre dans le cadre de l'héritage 222
Représentation dans le formalisme de base 223

Expression d'autres contraintes 225
Règles complémentaires 225
Pas d'incidence sur le MLD 226

Ce qu'il faut retenir 226

16 Diagramme de classes UML 227

Présentation d'UML 228
MERISE vs UML 228
Schéma entité-association et diagramme de classes UML 228

MCD et diagramme de classes UML 229
Un formalisme utilisé partiellement 229
Équivalence avec le formalisme entité-association 230

Reprise des exercices en UML 231
Gestion des achats d'une administration 231
Société canine 232
Bibliothèque universitaire 233
Guide Gastronomique 234
Représentations théâtrales 235
Location de logements 236
Agence de travail temporaire 237
Gendarmes et voleurs 238
Compagnie aérienne 239
Cinéma 240
Établissement scolaire 241
Club sportif 242

Ce qu'il faut retenir 243

Études de cas complètes 247

17 Association sportive et culturelle 249

Compte-rendu d'interview 249
Description du système d'information 249
Annexes.... 250

Recensement des informations 252
Modèle organisationnel de flux 252
Tableau des rubriques.... 252
Dictionnaire des données 253

Modèle conceptuel de données....254
Schéma entité-association 254
Diagramme de classes UML.... 255

Schéma relationnel 256
Modèle logique de données 256
Requêtes SQL de création des tables 256
Base de données Access.... 258

Commentaires sur le modèle 258

18 Compagnie d'assurances 259

Compte-rendu d'interview 259
Description du système d'information 259
Annexes.... 260

Recensement des informations 262
Modèle organisationnel de flux 262
Tableau des rubriques.... 262
Dictionnaire des données 263

Modèle conceptuel de données....264
Schéma entité-association 264
Diagramme de classes UML.... 265

Schéma relationnel 266
Modèle logique de données 266
Requêtes SQL de création des tables 266
Base de données Access.... 269

Commentaires sur le modèle 270

© Patrick Bergougnoux

19 Entreprise générale du bâtiment 271

Compte-rendu d'interview 271
Description du système d'information 271
Annexes.... 272

Recensement des informations 276
Modèle organisationnel de flux 276
Tableau des rubriques.... 277
Dictionnaire des données 279

Modèle conceptuel de données....280
Schéma entité-association 280
Diagramme de classes UML.... 281

Schéma relationnel 282
Modèle logique de données 282
Requêtes SQL de création des tables 282
Base de données Access.... 285

Commentaires sur le modèle 286

20 Société de services 287

Compte-rendu d'interview 287
Description du système d'information 287
Annexe 288

Recensement des informations 288
Modèle organisationnel de flux 288
Tableau des rubriques.... 289
Dictionnaire des données 289

Modèle conceptuel de données....290
Schéma entité-association 290
Diagramme de classes UML.... 291

Schéma relationnel 292
Modèle logique de données 292
Requêtes SQL de création des tables 292
Base de données Access.... 294

Commentaires sur le modèle et conclusion 294

Première partie

Recensement des informations

Dans cette première partie, nous nous intéressons uniquement au recueil des informations qui seront ensuite nécessaires à la construction des modèles conceptuels de données.

Quelles sont ces informations ? Où se trouvent-elles ? Comment les recenser facilement ? Ce sont autant de questions auxquelles les quatre chapitres de cette partie apportent des réponses simples et concrètes.

Chapitre 1

Analyse des flux d'information

Ce chapitre pose les bases de toute étude visant à automatiser la gestion des données d'une entreprise ou d'une quelconque organisation. Nous verrons à quoi correspond vraiment un système d'information et comment identifier de façon simple mais rigoureuse l'information qui s'y trouve.

Le système d'information

De quoi s'agit-il ?

On a souvent tendance à attribuer à la notion de « système d'information » des définitions diverses en fonction du contexte, qu'il soit simplement organisationnel ou associé à une démarche d'informatisation. Or, le plus simple est peut-être de revenir à la définition de base en prenant chaque mot et en regardant dans un dictionnaire à quoi il correspond.

Système

Bien sûr, de nombreuses définitions sont proposées. En voici une très générique et qui correspond bien à notre problématique : « *Un système est un ensemble d'éléments interagissant selon certaines règles, et concourant à un même objectif* ».

Information

Là aussi, les définitions sont multiples ; retenons-en une simple : « *Renseignement que l'on donne ou que l'on obtient sur quelqu'un ou quelque chose* ».

Nous pouvons donc définir un système d'information comme un ensemble de renseignements que l'on donne ou que l'on obtient sur quelqu'un ou quelque chose, qui interagissent selon certaines règles et qui concourent à un même objectif.

Le système étudié peut concerner tout ou partie d'une entreprise, le service d'une administration, ou de n'importe quelle organisation. Dans la suite de cet ouvrage, nous parlerons de « **domaine d'étude** ».

Les sous-systèmes dans les organisations

Quel que soit le domaine d'étude, un système peut toujours être étudié à travers deux sous-systèmes.

Le système opérant

On y retrouve les éléments de production, de transformation. C'est à ce niveau que la « matière première » est travaillée par les différents acteurs du domaine d'étude pour obtenir un « produit fini ».

Nous ne ferons pas ici un cours d'économie, mais il est intéressant de constater que tous les systèmes opérants, quels que soient les domaines, fonctionnent sur le même principe de « producteur-consommateur ».

Pour opérer, le système opérant a besoin de recevoir sa « matière première » quelle que soit sa nature (fourniture, personnel, finance, …) : il s'agit là d'un « flux entrant ».

Dès lors qu'il est stimulé par un flux, le système opérant va le consommer pour réaliser le produit fini, celui-ci étant ensuite envoyé sous forme d'un « flux sortant ».

Ce flux sortant représente alors un flux entrant pour l'autre système auquel il est destiné, qui le transformera à son tour… et c'est ainsi que fonctionnent et interagissent toutes les organisations.

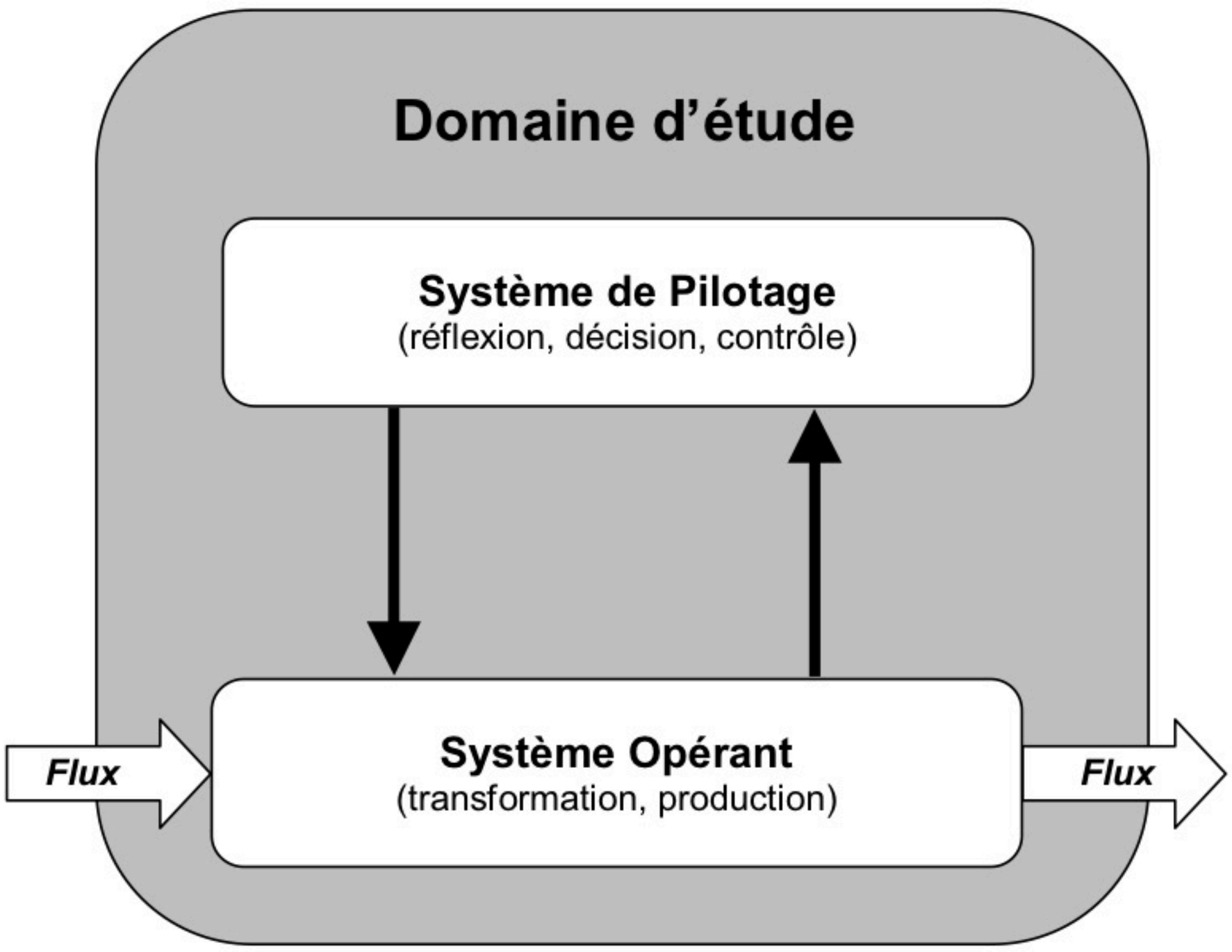
Le système de pilotage

C'est à ce niveau que se situent les aspects décisionnels, la réflexion sur l'évolution organisationnelle, le contrôle du bon fonctionnement du système opérant.

© Patrick Bergougnoux

La distinction entre le système opérant et le système de pilotage n'est pas forcément basée sur une distinction des personnes composant le domaine d'étude. Certes, dans les grandes organisations, les sous-systèmes correspondent à des services différents, mais au sein d'une petite entreprise, les mêmes personnes se retrouvent souvent à tous les niveaux : un artisan va souvent « opérer » dans la journée et « piloter » le soir ou le week-end.

Bien évidemment, ces deux sous-systèmes communiquent constamment pour la bonne marche de l'organisation, le système de pilotage donnant les instructions au système opérant et ce dernier remontant les éléments utiles au contrôle de l'organisation. Les échanges au sein du domaine d'étude peuvent donc être schématisés de la façon suivante :



Le problème se situe au niveau de la manière d'échanger : les acteurs du système de pilotage ne vont généralement pas aller sur le site de production pour en orchestrer le fonctionnement. De même, les acteurs du système opérant ne vont pas aller voir leurs dirigeants pour les informer de chacune de leurs actions.

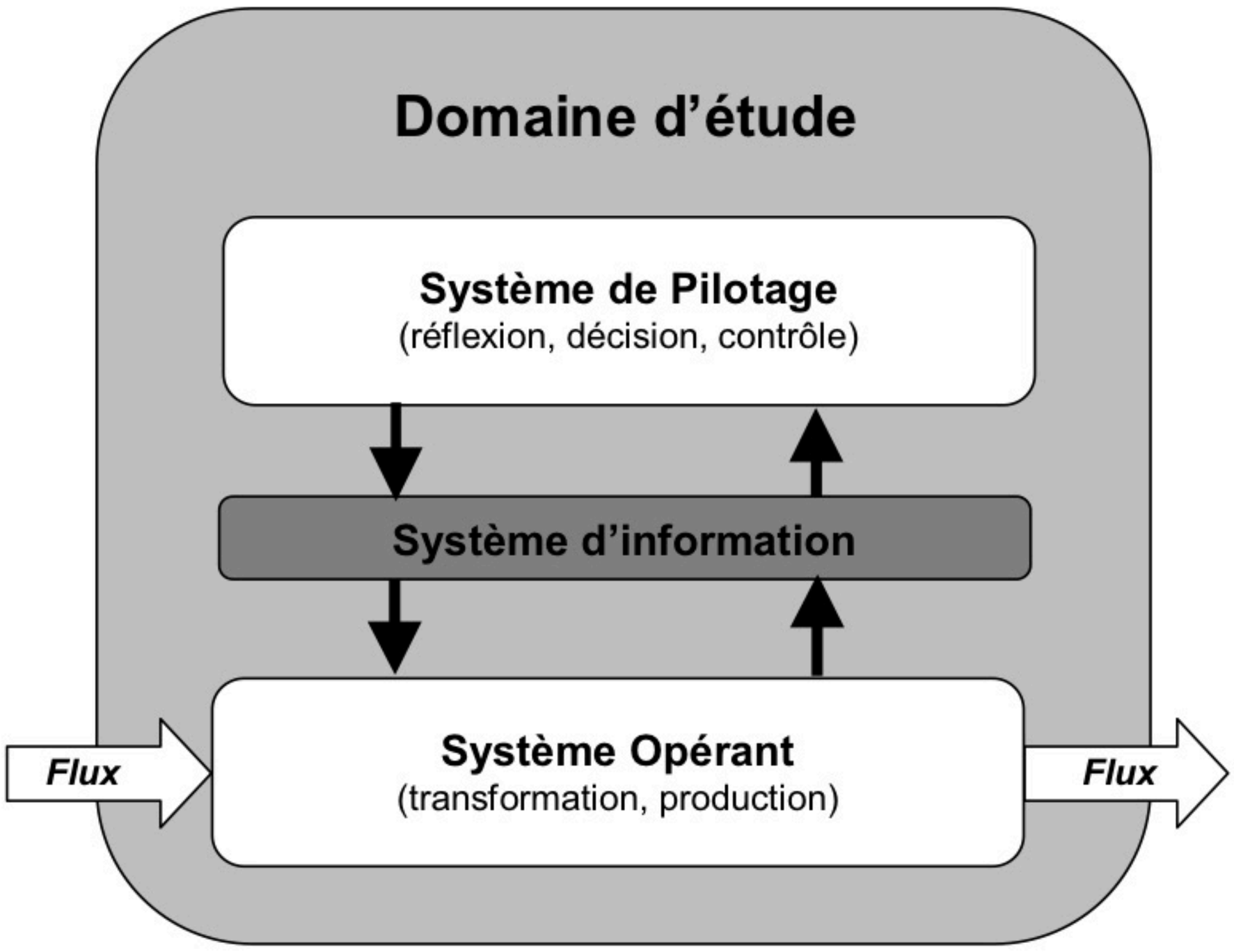
Comment les échanges entre les deux sous-systèmes peuvent-ils s'effectuer de manière efficace et formelle ?

Il faut pour cela une interface entre les deux sous-systèmes, une interface dans laquelle chacun va pouvoir stocker et traiter l'ensemble des informations du système organisationnel.

Cette interface, c'est le **système d'information**.

Le système d'information

Tout ce qui est manipulé par le système opérant est traduit sous forme d'information et stocké dans le système d'information. Ces données peuvent alors être consultées et manipulées par le système de pilotage qui contrôle et dirige ainsi le fonctionnement du système opérant, comme le montre le schéma ci-dessous :



Le système d'information est la représentation de l'activité du système opérant sous forme d'informations ; il est conçu à l'initiative du système de pilotage en fonction des objectifs à atteindre et de l'organisation choisie.

© Patrick Bergougnoux

L'informatisation

Informatique… Un mot récent, néologisme créé par contraction des mots « information » et « automatique », et adopté par l'Académie française en 1967 pour permettre de désigner la « science du traitement de l'information ».

Le mot « informatique » porte en lui toute la signification de ce qu'est vraiment son objectif : automatiser l'information.

Les anglo-saxons lui préfèrent généralement le terme « Computer Science » (science des ordinateurs). Pour ma part, je trouve ce terme très restrictif, les ordinateurs n'étant qu'un moyen d'automatiser l'information : l'algorithmique, la programmation, la modélisation et bien d'autres spécialités de l'informatique vont bien au-delà de la seule science des ordinateurs.

Vous l'avez compris, je suis un fan inconditionnel de l'origine française de ce mot, d'autant plus qu'il correspond parfaitement aux fondamentaux qui sont les miens dès qu'il est question d'informatisation.

En effet, vous aurez beau être le maitre absolu des sciences de l'ordinateur, concevoir une super-architecture matérielle sur un super-réseau avec des super-programmes, si les informations que vous traitez ne sont pas les bonnes, cela servira super-à-rien !

Pour automatiser une information, encore faut-il réussir à l'identifier !

Identifier l'information

Une tâche plus complexe qu'il n'y parait

Imaginez que vous arriviez dans une entreprise et que vous deviez faire l'inventaire de toutes les informations qui s'y trouvent dans le but d'établir le système d'information.

Où trouver ces informations ?

Lorsque je pose la question, les réponses que j'obtiens sont toujours très évasives : « *dans les documents* », « *il faut demander au personnel* », « *dans les archives* », … et la meilleure de toutes : « *partout* » !

Nous voilà bien avancés… Mais alors, par où commencer ?

Information ou état d'âme

Le plus simple est peut-être de commencer par déterminer ce qui n'est pas une information.

Imaginez un employé qui arrive dans le bureau de son chef et lui dit : « *J'ai une solution pour doubler notre chiffre d'affaires* », puis tourne les talons et rentre chez lui… L'entreprise va-t-elle pouvoir doubler son chiffre d'affaires ? Non… pourtant l'un de ses employés prétend avoir la solution, mais le système d'information de l'entreprise ne peut intégrer l'information qui le permettrait.

Prenons un autre exemple dans le cadre de l'enseignement ; c'est une expérience que je fais tous les ans : j'entre dans l'amphi avec mes connaissances sur mon cours et je ne dis rien pendant plusieurs minutes… puis je sors de l'amphi (bien sûr, je reviens de suite après !). Le système d'information qui régit mon enseignement intègre-t-il alors les informations relatives à ce cours ? Non… pourtant je les connais parfaitement.

Que manque-t-il à ces informations pour qu'elles puissent faire partie du système ?

La réponse est : **l'échange**.

Pour appartenir à un système d'information, une information doit forcément être échangée au sein de ce système.

Si un renseignement n'est jamais échangé, que ce soit en interne (collègues) ou avec des partenaires extérieurs (clients, fournisseurs, …), c'est-à-dire que la personne qui la détient la garde pour elle, alors ce n'est pas une information du domaine d'étude. Personnellement, c'est ce que j'appelle un « état d'âme » !

De l'audit au modèle de flux

Audit et compte-rendu d'interview

Pour identifier les informations du système, la première étape consiste à interroger les différents acteurs du domaine d'étude et recueillir tous les éléments relatifs à leur activité.

Cet audit donne lieu à la rédaction d'un compte-rendu d'interview qui décrit en détail le fonctionnement du domaine d'étude. On y retrouvera également, souvent sous forme d'annexes, des exemples de tous les documents utilisés.

© Patrick Bergougnoux

Ce compte-rendu permet également de recueillir les nouveaux besoins exprimés par les acteurs.

La spécificité de ce travail est néanmoins son manque de formalisme : il s'agit effectivement d'un document relatant de manière très détaillée, sous forme de prose, le fonctionnement du système. Par conséquent, il est assez difficile de repérer et d'isoler les éléments importants pour la suite de notre démarche.

Abstraction et formalisation

Un système d'information peut être observé à différents niveaux d'abstraction :

- Niveau **Technique** : tous les détails sont pris en compte.
- Niveau **Organisationnel** : on fait abstraction des aspects purement techniques qui décrivent comment fonctionne le système, et on se focalise sur l'organisation.
- Niveau **Conceptuel** : on fait abstraction des éléments d'organisation indiquant qui intervient et dans quel timing, et on ne retient que ce à quoi sert le système, son concept.

On peut ainsi porter un regard plus ou moins macroscopique ou microscopique sur le système d'information, en faisant ou non abstraction de certains aspects de son fonctionnement.

Ces niveaux peuvent se résumer de la façon suivante :

Niveau	Aspects pris en compte
Conceptuel	Quoi
Organisationnel	Quoi / Qui / Quand
Technique	Quoi / Qui / Quand / Comment

Le compte-rendu d'interview se situe au niveau technique : il regorge de détails mais peut difficilement être formalisé. À partir du niveau organisationnel, il va être possible de créer des modèles formels, faisant certes abstraction des détails techniques, mais ne laissant aucune place à l'incertitude quant au fonctionnement organisationnel du domaine d'étude.

Sur la base du compte-rendu d'interview, différents modèles peuvent donc être réalisés pour décrire l'organisation du système étudié. Parmi eux, le modèle organisationnel de traitements, issu de la méthode MERISE (Méthode d'Étude et de Réalisation Informatique pour les Systèmes d'Entreprise), décrit avec précision les tâches mises en œuvre par les acteurs du domaine d'étude. D'autres modèles, tels que certains diagrammes UML (cas d'utilisation, états-transitions, activité, séquence, communication, interaction, temps, …) permettent également de décrire le fonctionnement d'une organisation.

Vers un modèle focalisé sur l'inventaire des flux

Tous ces modèles sont intéressants si l'on souhaite une analyse précise de l'organisation d'un système. Dans notre cas, nous avons vu que l'information est contenue dans les échanges, quels que soient les traitements qui sont ensuite effectués. Nous allons donc nous focaliser sur ces échanges : pour cela, nous n'avons besoin de modéliser qu'une partie de l'organisation axée sur les flux et non sur les tâches.

Cela rend immédiatement la démarche beaucoup plus simple ; en effet, alors qu'on ne savait pas où chercher l'information dans un compte-rendu d'interview bourré de détails, nous savons maintenant qu'il nous suffit de repérer les échanges car, comme nous l'avons vu précédemment, l'information a forcément été échangée à un moment ou à un autre dans le système.

L'objectif est simplement de réaliser un modèle qui recense tous les flux impliquant le domaine d'étude afin d'en extraire les informations qu'ils contiennent.

En interrogeant les différents acteurs du système, on récupère les éléments reçus et envoyés par chacun d'entre eux, et il est ainsi possible d'établir aisément un premier modèle très simple : le modèle organisationnel de flux.

Ce qu'il faut retenir

- Le système d'information est la représentation de l'activité du domaine d'étude sous forme d'informations.
- Une information est forcément échangée.
- Pour identifier les informations, il faut commencer par recenser tous les flux impliquant le domaine d'étude.

© Patrick Bergougnoux

Chapitre 2
Modèle organisationnel de flux

Dans ce chapitre, nous détaillons le formalisme utilisé pour modéliser les flux. Nous nous plaçons à un niveau d'abstraction assez élevé puisque ce modèle se situe au niveau organisationnel et néglige même certains aspects de l'organisation. L'objectif est de mettre en œuvre un modèle le plus simple possible tout en restant exhaustif en termes d'échanges d'informations.

Structure du modèle

Une première étape simple mais déterminante

Il s'agit donc de recenser les flux circulant au sein de l'organisation et avec son environnement, et de les représenter sous la forme d'un modèle.

Ce modèle, au-delà de la première étape qu'il représente pour l'identification des informations, permet également d'appréhender simplement, mais de façon formelle, le fonctionnement global de l'entreprise. Il est en effet étonnant de voir à quel point un modèle aussi simpliste peut donner une première idée aussi réaliste de l'activité de l'entreprise ou de l'organisation en question.

Pour une entreprise, on pourrait dire : « Dis-moi ce que tu échanges et je te dirai qui tu es. »

L'acteur

Il s'agit du premier concept de base pour la réalisation d'un modèle de flux. Un acteur n'est pas forcément une personne physique : il peut s'agir d'un service entier, d'une entreprise externe, voire d'une machine. En fait, toute unité active produisant ou consommant des flux du domaine d'étude est considérée comme un acteur.

La principale caractéristique d'une unité active est sa capacité à effectuer des actions sur les flux qu'elle reçoit, et éventuellement de les renvoyer à un autre acteur une fois traités. Cette notion s'oppose à celle d'une unité passive qui ne peut pas effectuer d'actions : une unité passive ne doit alors pas être considérée comme un acteur et n'est pas susceptible, par conséquent, d'échanger des flux.

Un acteur est une unité active stimulée par des flux qu'elle traite et qu'elle renvoie éventuellement à un autre acteur.

On distingue les acteurs internes qui appartiennent au domaine d'étude, et les acteurs externes qui sont en dehors du domaine et qui collaborent avec lui.

Le flux

Nous retrouvons là un concept qui est la base de tout ce début d'ouvrage. Les flux représentent donc les échanges porteurs d'information. Comme indiqué précédemment, toute information, quelle que soit sa nature, est forcément échangée à un moment ou à un autre.

Un flux est un échange entre deux acteurs, c'est-à-dire deux unités actives. Cette précision est importante : en effet, par exemple, si un acteur jette un document à la poubelle (celle-ci ne pouvant raisonnablement pas être considérée comme une unité active !), alors cette action ne peut être représentée par un flux. Nous reviendrons un peu plus loin sur une série d'exemples du même genre.

Un flux est émis par un acteur à destination d'un autre acteur. Un flux ne vient pas de nulle part et n'est pas envoyé n'importe où. À chaque flux son émetteur et son récepteur, qu'il soit interne ou externe au domaine d'étude.

Notons également que les flux entre acteurs externes ne nous intéressent pas même si nous en avons connaissance : ils ne sont donc pas représentés dans le modèle organisationnel de flux de notre domaine d'étude. Les informations échangées entre les acteurs externes concernent leur propre système d'information, pas le notre. Et si certaines de ces informations nous concernent, elles nous arriverons forcément par d'autres flux vers nos acteurs internes.

© Patrick Bergougnoux

Un objectif bien précis

Un modèle volontairement limité

Le but de ce modèle est certes de donner une première idée du fonctionnement du domaine d'étude, mais son principal intérêt est de recenser tous les flux qui, une fois analysés, donneront l'ensemble des informations échangées.

La chronologie des échanges n'a d'ailleurs aucune importance : ce qui nous intéresse c'est que l'information existe et doive alors être prise en compte dans le système d'information, et non pas le moment où elle est utilisée. La chronologie des échanges n'est donc pas représentée dans le modèle organisationnel de flux.

Il en est de même pour les actions de mémorisation ; toutes les informations échangées sont recensées afin qu'elles fassent partie du système d'information : par conséquent, il est inutile de représenter les actions de mémorisation au sein du modèle organisationnel de flux.

Certains aspects de l'organisation, en particulier le « Quand », sont donc volontairement négligés : certes, cela limite le niveau de représentation de l'organisation contrairement à un modèle organisationnel de traitements qui détaille les tâches et leur chronologie ; mais cela rend le modèle plus simple, plus abordable et, surtout, suffisant pour le recensement de l'information et la suite de notre démarche.

Un modèle graphiquement épuré

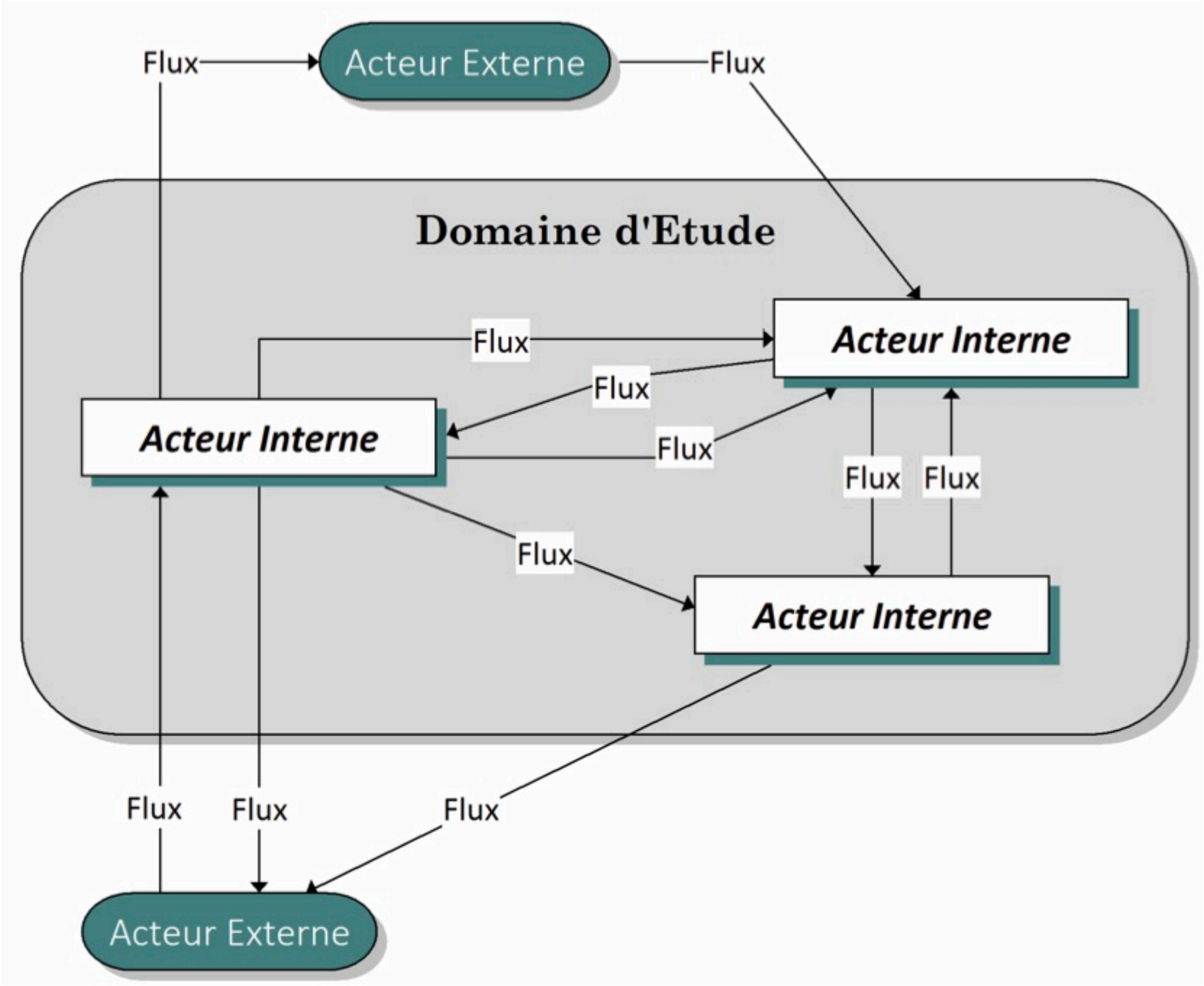
L'objectif de ce modèle organisationnel de flux est d'être le plus simple possible afin de bien mettre en évidence les différents échanges entre les acteurs qui serviront de base au recensement des informations du domaine d'étude.

Les acteurs internes sont placés au sein du domaine d'étude contrairement aux acteurs externes ; ceux-ci ont également une représentation graphique différente afin de bien mettre en évidence le système d'information étudié.

Les flux, pour leur part, sont naturellement représentés par des flèches allant de l'émetteur du flux vers son destinataire.

Par ailleurs, comme indiqué précédemment, les échanges entre les acteurs externes, même s'ils sont connus, n'intéressent pas le domaine d'étude et ne sont donc pas représentés dans le modèle organisationnel de flux.

Pour réaliser un modèle organisationnel de flux avec le logiciel ***Looping***, il faut utiliser les « Objets libres » et en particulier l'outil « Texte » qui permet de définir les acteurs ; l'outil « Lien » permet pour sa part de créer les flux.



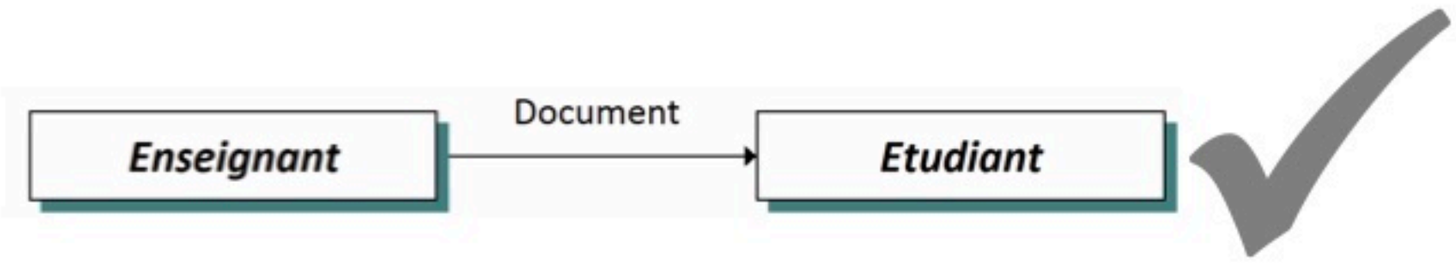
Quelques erreurs à éviter

Les erreurs le plus souvent commises sont liées à la définition même de ce qu'est un acteur. En effet, comme vu précédemment, un acteur est une unité active capable d'actions, contrairement aux unités passives qui ne sont souvent que des réceptacles, des lieux de passage ou de stockage.

Prenons par exemple le cas d'un enseignant qui dépose un document dans un casier afin que les étudiants viennent le récupérer ; on pourrait être tenté de considérer le casier comme un acteur :

© Patrick Bergougnoux

Or, le casier n'est en rien une unité active : il n'est en effet pas susceptible de faire des actions sur les documents qu'il reçoit et sert juste de support de communication entre le professeur et ses étudiants ; la modélisation du flux se résume donc à :



De même, un serveur de fichiers sur lequel l'enseignant dépose des documents pour ses étudiants ne doit pas être considéré comme un acteur, mais comme un moyen de transmission (non représenté dans les modèles de flux). Par contre, si le serveur réalise un traitement automatique comme c'est souvent le cas, par exemple, pour les serveurs Web, il sera bien considéré comme une unité active, c'est-à-dire un acteur.

Il n'est pas nécessaire d'indiquer la façon dont un flux est transmis, ni son support (papier, courriel, texto, téléphone, …) : la seule chose qui compte est l'échange entre les deux acteurs.

Autre exemple évoqué précédemment : un acteur jette un document à la poubelle… Comment représenter cette action dans le modèle de flux ? La réponse est très simple : on ne la représente pas ! En effet, la poubelle n'est pas une unité active et n'est donc pas un acteur ; par ailleurs, on voit mal comment le fait de jeter quelque chose à la poubelle pourrait apporter une quelconque information au système.

Dernier exemple : un acteur met un document de côté et le récupère le lendemain pour le traiter ; quelle représentation peut-on envisager pour cela dans le modèle de flux ? Une boucle réflexive sur l'acteur ? Aucun intérêt : en effet, rappelons que si l'information n'est échangée à aucun moment avec un autre acteur, ce n'est pas vraiment une information du domaine d'étude. C'est le fameux état d'âme vu au chapitre précédent… et les états d'âme ne sont pas représentés dans le modèle organisationnel de flux !

Les seuls éléments à représenter dans le modèle organisationnel de flux sont les **échanges** impliquant les acteurs internes : tout le reste est inutile pour notre quête de recensement des informations.

Mise en pratique avec un exemple simple

Compte-rendu d'interview

L'entreprise Beaux Rivages propose à ses clients toute une gamme de mobilier de jardin. Le domaine d'étude se limite ici à la gestion quotidienne de la facturation des produits.

Le matin à partir de 9 heures, le secrétariat saisit les factures des produits vendus la veille aux clients. Elles sont imprimées au sein du service et mises en attente jusqu'en début d'après-midi. À 14 heures, elles sont transmises à la direction qui les vérifie et les renvoie au secrétariat avant 16 heures avec d'éventuelles corrections. Une fois les modifications effectuées, les factures sont envoyées aux clients par courrier et une copie est transmise au service comptabilité.

Lors de l'arrivée du courrier vers 11 heures, le service comptabilité réceptionne les paiements de clients : pour simplifier, on considère que le paiement d'une facture est effectué par un et un seul chèque (1 facture ➔ 1 chèque).

En fin de mois, le service comptabilité réalise les statistiques de ventes du mois passé, prépare un mail avec le document en pièce jointe, et l'envoie à la direction et aux actionnaires.

Domaine d'étude et acteurs

Le domaine d'étude est composé de trois acteurs internes :

- Le secrétariat qui réalise et gère la transmission des factures.
- Le service comptabilité qui s'occupe de la réception des paiements et de l'établissement des statistiques.
- La direction qui contrôle les factures et étudie les statistiques.

Deux acteurs externes sont concernés :

- Les clients qui reçoivent les factures et les payent.
- Les actionnaires qui réceptionnent les statistiques.

Recherche des échanges

Lors de la lecture du compte-rendu d'interview, on se focalise sur les échanges :

> *L'entreprise Beaux Rivages propose à ses clients toute une gamme de mobilier de jardin. Le domaine d'étude se limite ici à la gestion quotidienne de la facturation des produits.*
>
> *Le matin à partir de 9 heures, le secrétariat saisit les factures des produits vendus la veille aux clients. Elles sont imprimées au sein du service et mises en*

© Patrick Bergougnoux

attente jusqu'en début d'après-midi. À 14 heures, elles sont transmises à la direction qui les vérifie et les renvoie au secrétariat avant 16 heures avec d'éventuelles corrections. Une fois les modifications effectuées, les factures sont envoyées aux clients par courrier et une copie est transmise au service comptabilité.

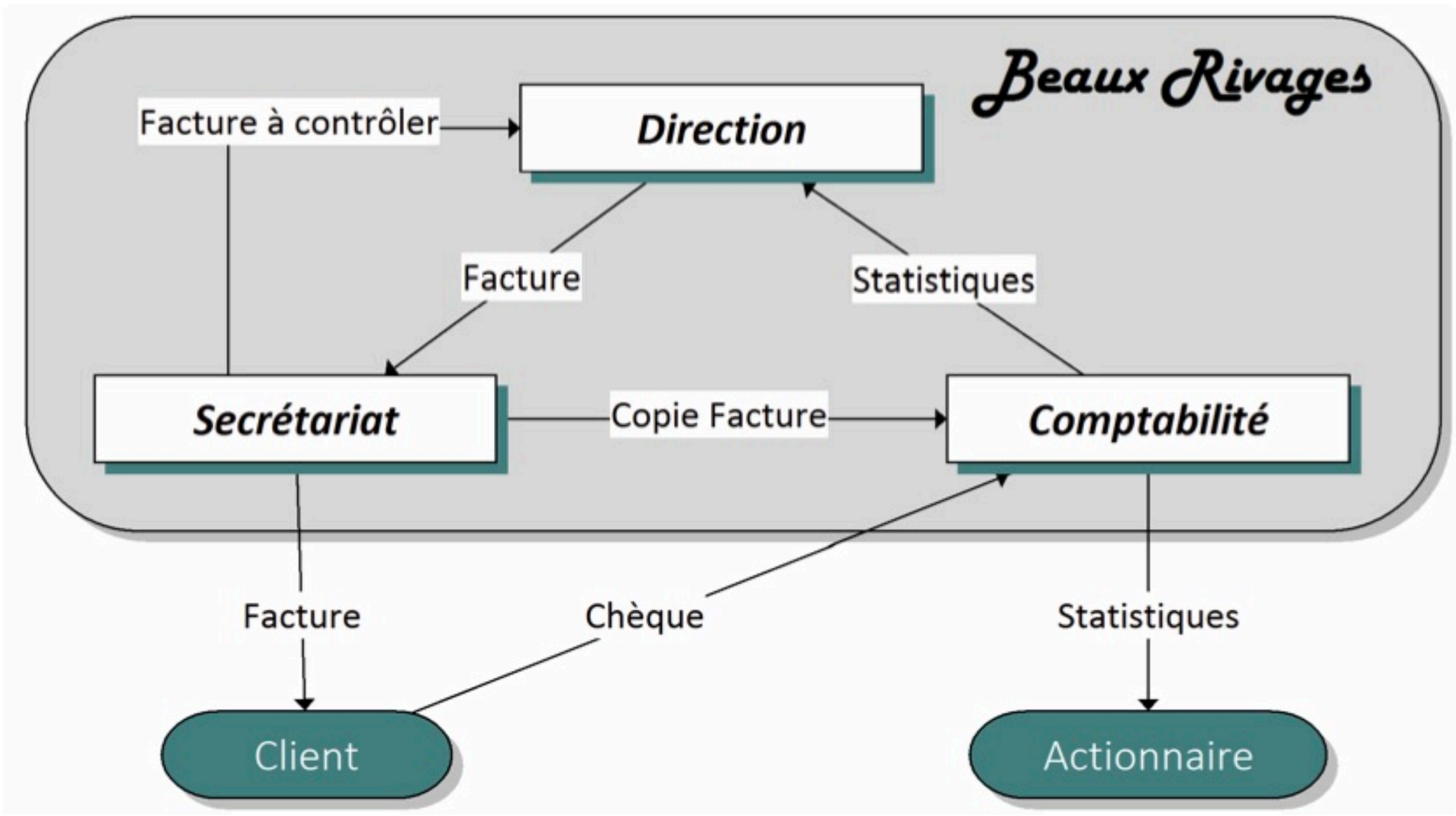
Lors de l'arrivée du courrier vers 11 heures, le service comptabilité réceptionne les paiements de clients : pour simplifier, on considère que le paiement d'une facture est effectué par un et un seul chèque (1 facture ➔ 1 chèque).

En fin de mois, le service comptabilité réalise les statistiques de ventes du mois passé, prépare un mail avec le document en pièce jointe, et l'envoie à la direction et aux actionnaires.

Sept flux ont ainsi été identifiés. Le reste du compte-rendu d'interview (horaires, mise en attente de document, travail personnel des acteurs, détails techniques, …) ne nous intéresse pas pour la mise en œuvre du modèle organisationnel de flux ; ces indications, que l'on qualifierait de « bruits » en génie logiciel, ont certes un intérêt pour la compréhension du système, mais pas pour le recensement des informations.

Réalisation du modèle organisationnel de flux

Une fois les acteurs et les flux identifiés, la réalisation du modèle ne pose aucun problème. Il faut juste faire attention de bien positionner les acteurs de manière à éviter, autant que possible, le croisement des flux.



Cet exemple sera également utilisé dans le chapitre suivant et régulièrement tout au long de l'ouvrage.

Ce qu'il faut retenir

- Un acteur est une unité active capable de réaliser des actions.
- Un flux est forcément émis par un acteur à destination d'un autre acteur.
- Le modèle organisationnel de flux se focalise sur l'existence des échanges entre acteurs et fait abstraction des aspects techniques et temporels.

© Patrick Bergougnoux

Chapitre 3
Dictionnaire des données

À partir du modèle organisationnel de flux qui indique tout ce qui est échangé dans le cadre du domaine d'étude, nous allons pouvoir analyser en détail chacun des flux et recenser toutes les informations qu'ils contiennent.

Sur la base de l'exemple du chapitre précédent, analysons la représentation des différents flux pour en extraire l'ensemble des rubriques qui les composent. Ces rubriques sont souvent directement décrites dans le compte-rendu d'interview et parfois représentées dans des documents annexés à ce compte-rendu.

Analyse des informations issues des flux

Représentation des flux

Dans notre exemple, trois types de documents font l'objet de flux entre les acteurs :

- les factures,
- les chèques,
- les statistiques.

Afin de rendre cette première analyse la plus abordable possible, nous avons pris une version simple de ces documents. Dans la réalité, ils seraient plus complets et complexes, mais la démarche serait la même.

Facture

Cet exemple de facture, bien que très simple, est assez représentatif des informations que l'on trouve habituellement sur ce type de documents. Précisons également que, dans cet exercice, tous les produits de l'entreprise sont soumis au même taux de TVA.

Beaux Rivages
www.beauxrivages.com

coupon à joindre au règlement

Numéro	FA19010123
Date	14/01/2019
Montant	3 375,12 €

FACTURE

Numéro FA19010123
Date 14 janvier 2019

HIBULAIRE Pat
15 chemin bonne humeur
75014 PARIS

Réf	Qté	Désignation	Prix Unitaire	Montant
120	2	Table rectangulaire 3x1m	120,00	240,00
230	4	Chaise plastique	13,00	52,00
200	6	Chaise bois	48,60	291,60
300	1	Bar roulant	254,00	254,00
100	5	Table ronde 2m	395,00	1 975,00
			Total HT	2 812,60
			TVA 20%	562,52
			TOTAL TTC	**3 375,12**

Facture établie en €uros – Valeur en votre aimable règlement

Chèque

Afin de simplifier l'analyse, nous considérons que le client utilise toujours le même compte bancaire et que le montant du chèque est toujours égal au total TTC précis de la facture correspondante. Par conséquent, un et un seul chèque vient systématiquement payer une et une seule facture.

© Patrick Bergougnoux

Statistiques

En se contentant de lister le chiffre d'affaires annuel des clients avec une simplicité extrême, ce document est particulièrement caricatural, mais suffisant pour expliquer la démarche.

Beaux Rivages
www.beauxrivages.com

STATISTIQUES

CLIENT	CA HT ANNUEL
DUPONT	12 345 €
HIBULAIRE	23 456 €
LOMBEC	9 876 €
TOTAL HT	**45 677 €**

Repérage des informations

Il faut donc recenser l'ensemble des informations contenues dans ces documents ; pour cela, il suffit de les prendre les uns après les autres, dans n'importe quel ordre et de repérer les différentes rubriques qui s'y trouvent.

Qu'est-ce qu'une rubrique ?

Une rubrique, également appelé champ, est un élément qui peut contenir différentes données variables jouant le même rôle au sein du système et possédant le même type (texte, numérique, date, …) et les mêmes propriétés (nombre de caractères, précision, …). Il peut s'agir, par exemple, d'un nom de client, d'une référence de produit, d'une date de naissance, d'un prix de vente, d'une date de commande, d'un numéro de facture, …

Ne pas confondre contenant et contenu : la rubrique est le contenant (ex : Nom client) et elle accueille des valeurs qui sont donc contenues dans ce contenant (ex : Dupont, Durand, …).

Pour repérer ces rubriques et afin de n'en oublier aucune, le plus visuel est de les entourer ou de les surligner sur le document où elles apparaissent, et de leur affecter un numéro.

Lors de ce recensement des rubriques, certains pièges sont à éviter :

- Les **redondances** : bon nombre de rubriques sont présentes sur plusieurs documents, voire plusieurs fois sur le même document. Il faut alors prendre garde à ne les recenser qu'une seule fois en leur affectant le même numéro.
- Les **synonymes** : certaines rubriques peuvent paraitre distinctes parce que leur désignation est différente. Il s'agit néanmoins parfois de la même rubrique présentée différemment. Comme précédemment, il ne faut les recenser qu'une seule fois en leur affectant le même numéro.
- Les **homonymes** : des rubriques peuvent posséder la même désignation sur les documents sans forcément représenter la même information. Il faut alors les distinguer et leur affecter des numéros différents.

Certaines rubriques peuvent aussi être issues d'informations données directement dans le texte du compte-rendu d'interview, sans forcément être représentées dans des pièces jointes en annexe.

Dans ce premier exemple, ce n'est pas le cas : toutes les informations sont présentes sur les trois documents que nous allons analyser.

© Patrick Bergougnoux

Commençons par analyser la facture :

Beaux Rivages 1
www.beauxrivages.com 2

coupon à joindre au règlement

Numéro 3 FA19010123
Date 4 14/01/2019
Montant 5 3 375,12 €

FACTURE

Numéro 3 FA19010123
Date 4 14 janvier 2019

6 HIBULAIRE Pat 7
8 15 chemin bonne humeur
9 75014 PARIS 10

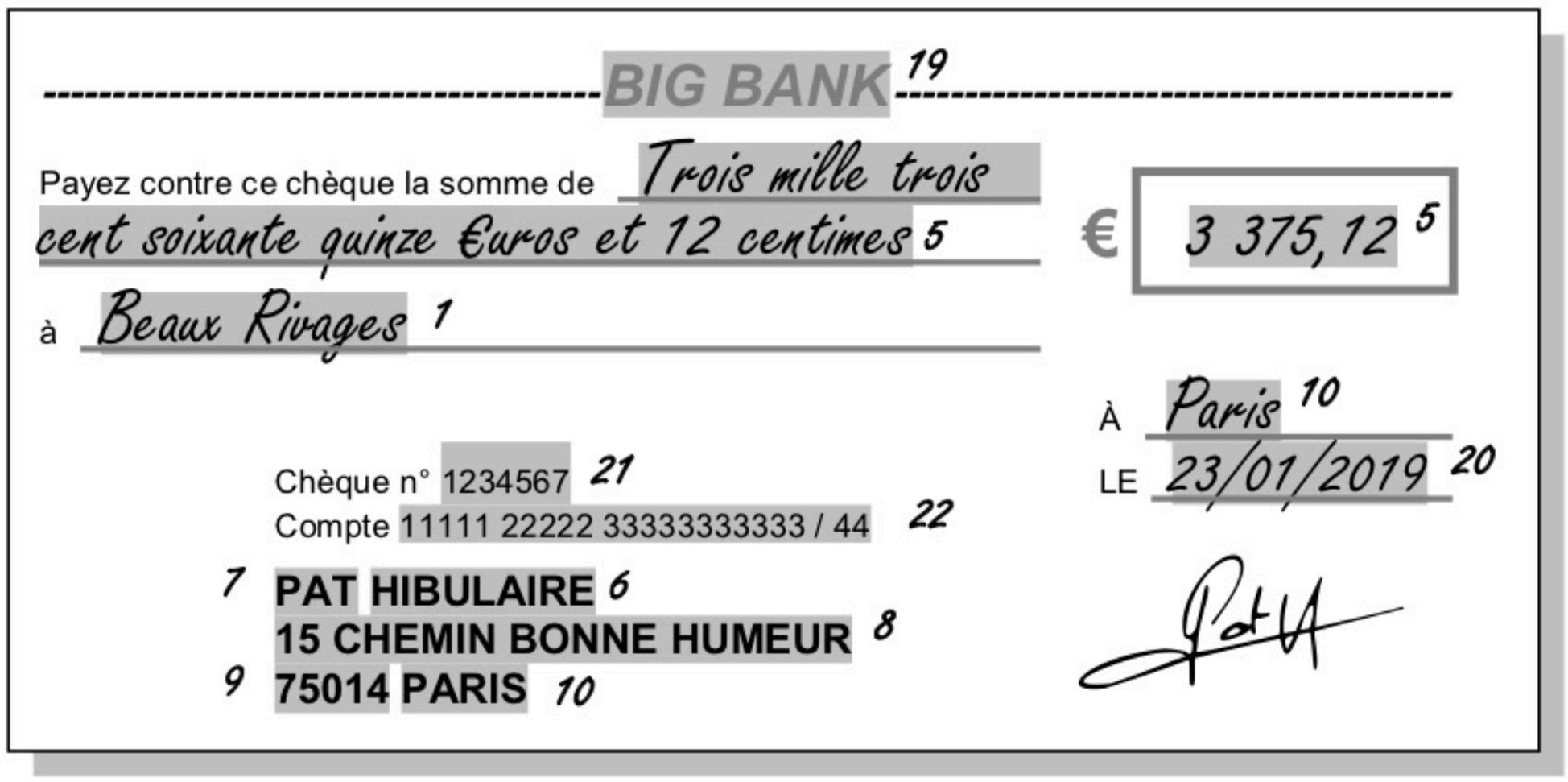
Réf	Qté	Désignation	Prix Unitaire	Montant	
11 120	12 2	13 Table rectangulaire 3x1m	14 120,00	240,00	15
11 230	12 4	13 Chaise plastique	14 13,00	52,00	15
11 200	12 6	13 Chaise bois	14 48,60	291,60	15
11 300	12 1	13 Bar roulant	14 254,00	254,00	15
11 100	12 5	13 Table ronde 2m	14 395,00	1 975,00	15
			Total HT	2 812,60	16
			TVA 20% 17	562,52	18
			TOTAL TTC	**3 375,12**	5

Facture établie en €uros – Valeur en votre aimable règlement

Sur ce document, on remarque :

- Des redondances : le numéro et la date de facture apparaissent plusieurs fois ; de même, il y a de nombreuses références de produits pour lesquelles le contenu est certes différent, mais le contenant (la rubrique) est le même. On leur affecte donc le même numéro ; il en est de même pour les quantités, désignations, prix et montants constituant le corps de la facture.
- Des synonymes : « TOTAL TTC » est la même rubrique que « Montant » sur le coupon en haut de la facture, même si les libellés sont différents.
- Des homonymes : le libellé « Montant » est utilisé à la fois dans le coupon pour indiquer le total TTC et dans le corps de la facture pour indiquer le total de la ligne.

Passons maintenant à l'analyse du chèque (on rappelle que, dans notre cas, le chèque envoyé par le client correspond exactement au montant de la facture) :


BIG BANK 19

Payez contre ce chèque la somme de Trois mille trois cent soixante quinze Euros et 12 centimes 5

€ 3 375,12 5

à Beaux Rivages 1

À Paris 10

LE 23/01/2019 20

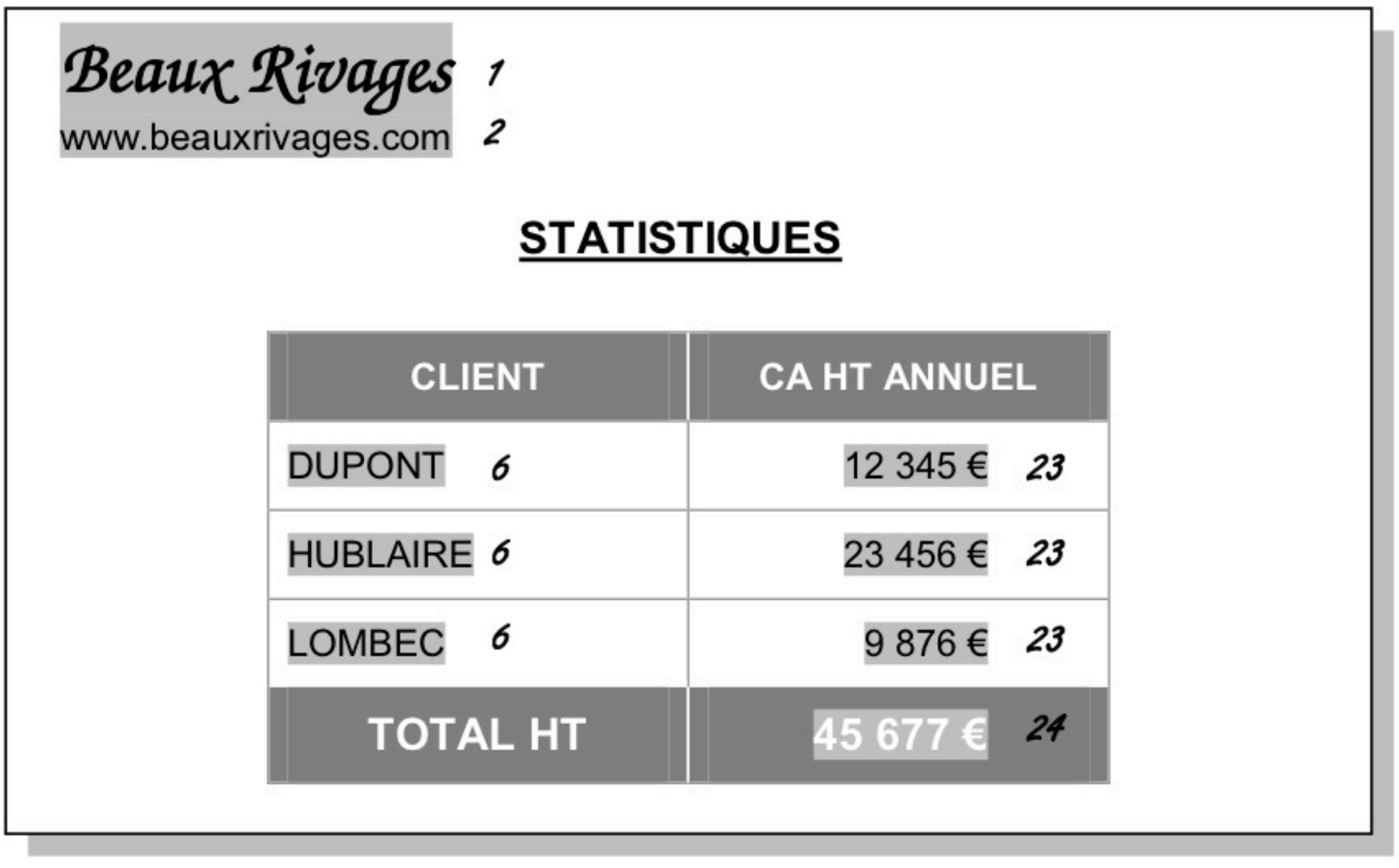
Chèque n° 1234567 21

Compte 11111 22222 33333333333 / 44 22

7 PAT HIBULAIRE 6
15 CHEMIN BONNE HUMEUR 8
9 75014 PARIS 10

On constate que, au fur et à mesure que l'on avance dans l'analyse des documents, on retrouve les mêmes rubriques, et on en recense donc de moins en moins de nouvelles. Concernant la signature, il n'est pas convenable de la recenser dans notre système d'information sauf autorisation expresse du client !

Terminons avec l'analyse du document des statistiques qui ne rajoute que deux nouvelles rubriques :


Beaux Rivages 1

www.beauxrivages.com 2

STATISTIQUES

CLIENT	CA HT ANNUEL
DUPONT 6	12 345 € 23
HUBLAIRE 6	23 456 € 23
LOMBEC 6	9 876 € 23
TOTAL HT	45 677 € 24

© Patrick Bergougnoux

Constitution d'un tableau des rubriques

Les rubriques ainsi repérées sur les documents peuvent être reportées dans un tableau au fur et à mesure de leur recensement. En cas de présence d'homonymes, il convient de préciser le libellé pour les distinguer.

La colonne « Exemple » est très importante : en effet, elle permet de ne pas confondre **contenant** et **contenu**.

Si, lors de l'analyse du document, on considère « Dupont » comme une rubrique, on se rendra compte de l'erreur au moment de préciser l'exemple en constatant que « Dupont » est l'exemple, c'est-à-dire un contenu et non un contenant.

Par ailleurs, **les documents ou les listes ne sont jamais des rubriques : ils contiennent des rubriques** ; ainsi, « Liste produits » n'est pas une rubrique : elle contient des rubriques telles que « Réf produit», « Désignation produit », « Prix unitaire produit », … et on s'en rend immédiatement compte en donnant un exemple qui ne doit en aucun cas être un document complet.

Les deux dernières colonnes seront traitées par la suite.

N°	Libellé	Exemple		
1	Nom entreprise	Beaux Rivages		
2	Site web entreprise	www.beauxrivages.com		
3	Numéro facture	FA19010123		
4	Date facture	14/01/2019		
5	Montant TTC facture	3 375,12 €		
6	Nom client	Hibulaire		
7	Prénom client	Pat		
8	Adresse client	15 chemin bonne humeur		
9	Code postal client	75014		
10	Ville client	Paris		
11	Réf produit	230		
12	Quantité vendue produit	4		
13	Désignation produit	Chaise plastique		
14	Prix unitaire produit	13,00 €		
15	Montant ligne facture	52,00 €		
16	Total HT facture	2 812,60 €		
17	Taux TVA	20%		
18	Montant TVA facture	562,52 €		
19	Banque client	Big Bank		
20	Date paiement	23/01/2019		
21	Numéro chèque	1234567		
22	Numéro compte client	11111 22222 3333333…		
23	CA HT annuel	12 345 €		
24	TOTAL HT CA	45 677 €		

Évolution vers le dictionnaire des données

Gestion des rubriques calculables

Nous venons de recenser toutes les rubriques présentes sur les documents échangés dans le cadre de notre domaine d'étude. Le recensement est donc exhaustif, mais une question se pose :

Toutes ces rubriques doivent-elles être mémorisées dans le système d'information ?

Prenons un exemple : si je vous demande de mémoriser une somme en euros, par exemple 123 456 789 euros, et en francs ce qui donne 809 823 449,42 francs, allez-vous encombrer votre mémoire avec les deux valeurs, ou allez-vous en retenir juste une (plutôt celle en euros dans notre exemple !) sachant que vous pourrez recalculer la deuxième ? À priori, la deuxième solution semble la plus pertinente.

Le raisonnement est le même pour un système d'information dont nous n'allons pas encombrer la mémoire avec des rubriques qu'il pourra aisément recalculer quand il en aura besoin, et ce très rapidement compte-tenu de la puissance des systèmes informatiques actuels.

Cependant, même si le gain de place en mémoire était important il y a quelques décennies, ce n'est plus vraiment le cas aujourd'hui avec les énormes capacités de stockage de nos ordinateurs. Nous pourrions donc nous payer le luxe de stocker des rubriques calculables.

Mais le principal problème ne se situe pas vraiment au niveau du gain de mémoire, mais beaucoup plus au niveau de la cohérence et de l'intégrité du système d'information à tout moment. En effet, dans le cas de notre facture, si le total d'une ligne est stocké et qu'une quantité est changée, juste après la validation de la nouvelle quantité, le système se retrouve dans une situation incohérente avec un total de ligne qui n'est plus égal à la multiplication du prix unitaire par la quantité… sans parler du total HT de la facture, du montant de la TVA, du total TTC, du chiffre d'affaires annuel du client dans les statistiques…

Toutes ces valeurs, si elles sont mémorisées dans le système, doivent être recalculées à chaque modification d'un des éléments qui est à l'origine de son calcul. Et, si le système connait un incident entre la modification de l'élément en question et le recalcul des rubriques qui en dépendent, le système d'information devient incohérent. Ne soyons cependant pas alarmiste : les systèmes informatiques mettent en œuvre des procédures permettant de maintenir cette cohérence, et il est donc possible (et même parfois préférable)

© Patrick Bergougnoux

de mémoriser des rubriques calculables. Mais, dans notre approche, nous partirons du principe que seules les rubriques qui ne sont pas calculables seront mémorisées dans le système d'information : il s'agit de **rubriques primaires**.

Une rubrique primaire doit être mémorisée dans le système d'information car elle ne peut pas être calculée à partir d'autres rubriques du système.

Pièges à éviter

Référencements circulaires

Si la rubrique X est calculée à partir de la rubrique Y, elle-même calculée à partir de la rubrique Z, elle-même calculée à partir de la rubrique X, il va fatalement manquer des données de base pour calculer X, Y et Z !

Dans notre exemple, on pourrait ainsi considérer que :

- « Montant ligne » est calculé à partir de « Prix unitaire » x « Quantité ».
- « Prix unitaire » est calculé à partir de « Montant ligne » / « Quantité ».
- « Quantité » est calculée à partir de « Montant ligne » / « Prix unitaire ».

Les trois rubriques seraient alors considérées comme calculables : le problème est que chacune fait référence aux deux autres et qu'aucune information n'est finalement mémorisée dans le système pour calculer ces trois rubriques. Dans notre cas, deux d'entre elles devront donc être primaires : en l'occurrence, le meilleur choix semble être « Prix unitaire » et « Quantité » qui permettront de calculer « Montant ligne ».

Rubriques en constante évolution

Certaines rubriques, qui ne peuvent pourtant pas être calculées à partir d'autres rubriques présentes sur les documents analysés, ne sont pas propices à une mémorisation dans leur état initial. Il s'agit en particulier de rubriques qui évoluent constamment ; prenons par exemple l'âge d'une personne qui apparait dans un document : doit-on considérer cet âge comme une rubrique primaire ? Si cela devait être le cas, il faudrait la réactualiser en permanence, surtout si l'on veut connaitre l'âge de la personne avec précision : il sera donc préférable de considérer l'âge comme une rubrique calculable à partir de la date du jour et de la date de naissance de la personne.

Une date, bien que n'apparaissant pas dans les documents, peut être considérée comme sous-jacente, et être alors rajoutée au tableau des rubriques pour servir de base au calcul d'un âge.

De nombreux cas similaires existent dès lors que l'information évolue dans le temps comme, par exemple, l'ancienneté dans une entreprise, qui devra être considérée comme une rubrique calculable à partir de la date du jour et de la date d'embauche, ou le stock d'un produit calculable à partir du stock initial et des entrées-sorties.

Rubriques initialisées automatiquement

À l'inverse de ce que nous venons de voir, on pourrait parfois être tenté de considérer certaines rubriques comme calculables alors qu'elles ne le sont pas du tout. Prenons l'exemple de la date de la facture : il est vrai que le futur système informatique va proposer automatiquement la date du jour et qu'il suffira de la valider.

Doit-on pour autant considérer que cette rubrique est calculable ? Pour le savoir, il suffit d'imaginer que la facture est réimprimée quelques jours plus tard : quelle sera alors sa date ? Ce sera le jour de l'impression et non pas le jour de la réalisation de la facture. C'est la raison pour laquelle il est nécessaire de mémoriser cette date de facture qui est donc bien une rubrique primaire, même s'il n'a pas été nécessaire de la saisir.

Le fait que la saisie d'une rubrique soit facilitée par le système ne signifie pas qu'elle ne doive pas être mémorisée ensuite.

Ne pas mettre la charrue avant les bœufs

De même que certaines rubriques peuvent être initialisées par le système informatique, d'autres sont automatiquement déduites lors de l'utilisation du logiciel. Sont-elles calculables pour autant ?

Pour bien comprendre, prenons le cas de la ville du client : ne peut-on pas la considérer comme calculable à partir du code postal ?

NON, et ce pour plusieurs raisons : tout d'abord, un même code postal peut correspondre à plusieurs villes ; mais oublions cela et considérons qu'il s'agit seulement des bureaux distributeurs qui disposent chacun de leur code postal. J'aimerais alors connaitre la formule qui, à partir du code postal, me calcule le nom de la ville ! Comment, à partir du code postal « 31150 », le système peut calculer que j'habite à « Gagnac sur Garonne » ? La réponse à cette question est souvent la même : « *il suffit de faire référence à une table de correspondance entre les codes postaux et les villes* ». D'accord, mais encore faut-il que notre système d'information connaisse les noms des villes pour établir cette fameuse table de correspondance : pour cela, ces noms de villes doivent être mémorisés, et « Ville client » est donc une rubrique primaire.

© Patrick Bergougnoux

De même, lorsque la référence du produit sera saisie, le futur logiciel va automatiquement afficher la désignation et le prix unitaire, mais ces informations auront dues, au préalable, être enregistrées dans le système… à moins que vous ne connaissiez une superbe formule de calcul qui, à partir de « 230 » construise automatiquement le texte « Chaise plastique » !

En pratique, on constate que la très grande majorité des rubriques calculables sont de type numérique.

Pour que le système puisse proposer automatiquement des informations, encore faut-il qu'il les ait au préalable mémorisées dans les rubriques primaires correspondantes.

Il ne faut donc pas mettre la charrue avant les bœufs en considérant comme calculables des informations que le système va certes retrouver seul… mais seulement lorsque son automatisation sera achevée, et à condition qu'il les ait préalablement mémorisées.

L'analyse que nous menons permet de concevoir le système d'information : il ne faut donc pas raisonner comme s'il existait déjà.

Transgresser en connaissance de cause

Dans certaines situations, le concepteur préfèrera mémoriser des rubriques calculables : c'est le cas en particulier quand une information est fréquemment utilisée en consultation et peu souvent modifiée.

Ce qui est important, c'est d'assumer les conséquences liées au choix de mémoriser une rubrique calculable. Comme nous l'avons vu précédemment, le risque se situe au niveau du maintien de la cohérence du système d'information : dès lors qu'une rubrique calculable est mémorisée, il faut s'assurer qu'elle sera bien recalculée à chaque modification des éléments qui contribuent à son calcul. Il faut alors choisir entre la lourdeur d'un recalcul à chaque consultation et la lourdeur d'un recalcul à chaque modification des rubriques impliquées. L'essentiel est d'être conscient de cette nécessité.

Dans cet ouvrage, nous prendrons comme règle de base de ne conserver que les rubriques primaires pour modéliser le système. Vous pouvez transgresser cette règle dans la mesure où vous le faites en connaissance de cause et que vous en assumez les conséquences en vue du maintien de la cohérence du système.

Par ailleurs, la gestion de l'historisation de certaines informations nécessitera parfois de dupliquer des rubriques et/ou de rajouter des rubriques de dates spécifiques. Cette problématique sera traitée plus loin dans cet ouvrage.

Paramètres

Certaines rubriques primaires au sein du tableau que nous avons établi ont une particularité très forte. C'est le cas, par exemple, du nom de l'entreprise : mais qu'est ce que cette rubrique a de si particulier ? La réponse est qu'elle n'a qu'une seule valeur pour tout le système d'information.

Il ne s'agit pas d'une constante : en effet, le nom d'une entreprise peut évoluer, mais toute nouvelle valeur prend la place de l'ancienne, et il n'y a toujours qu'une seule valeur pour le nom de l'entreprise dans tout le système.

Dans notre exemple, d'autres rubriques sont dans le même cas : l'adresse du site web de l'entreprise, et également le taux TVA ; en effet, nous avions précisé que tous les produits de l'entreprise sont soumis au même taux de TVA.

Ces rubriques particulières sont considérées comme des paramètres qui font l'objet d'un traitement spécifique : ils sont mémorisés à part et, comme les rubriques calculables, ne feront pas partie du modèle conceptuel de données que nous allons élaborer par la suite.

Un paramètre est une rubrique primaire qui n'a qu'une seule valeur pour tout le système d'information.

Attention cependant à ne pas commettre l'erreur de voir des paramètres partout !

En effet, après avoir expliqué en cours cette notion, j'ai eu la surprise de voir les étudiants considérer de nombreuses rubriques comme étant des paramètres : le numéro de facture par exemple. Pourquoi ? Et bien, tout simplement parce qu'une facture n'a qu'un seul numéro… Effectivement… sauf que le système d'information comprend de nombreuses factures avec chacune leur numéro !

Ainsi, si on n'y prend pas garde, presque toutes les rubriques deviennent des paramètres : un client n'a qu'un nom, un produit n'a qu'une seule référence, … et la liste pourrait être longue ! Mais ce serait ignorer que le système d'information comprend de nombreux clients, de nombreux produits, …

Les paramètres ne concernent que les renseignements d'ordre général du domaine d'étude : son nom, ses coordonnées, et quelques valeurs spécifiques n'existant qu'en un seul exemplaire dans tout le système d'information.

De nombreux autres éléments comme les libellés, les textes de présentation, les graphiques, … sont présents dans les documents : ce sont soit des constantes, ou au mieux des paramètres, qui doivent donc être exclus du tableau des rubriques.

© Patrick Bergougnoux

Complément du tableau des rubriques

Les deux dernières colonnes permettent donc d'indiquer si les rubriques sont primaires (P) ou calculables (C), et s'il s'agit de paramètres ; pour celles qui sont calculables, on précise les autres rubriques du tableau qui participent au calcul.

On peut alors rayer dans le tableau toutes ces rubriques calculables ainsi que les paramètres :

N°	Libellé	Exemple	P/C	Calcul
~~1~~	~~Nom entreprise~~	Beaux Rivages	P	*Paramètre*
~~2~~	~~Site web entreprise~~	www.beauxrivages.com	P	*Paramètre*
3	Numéro facture	FA19010123	P	
4	Date facture	14/01/2019	P	
~~5~~	~~Montant TTC facture~~	3 375,12 €	C	[16] + [18]
6	Nom client	Hibulaire	P	
7	Prénom client	Pat	P	
8	Adresse client	15 chemin bonne humeur	P	
9	Code postal client	75014	P	
10	Ville client	Paris	P	
11	Réf produit	230	P	
12	Quantité vendue produit	4	P	
13	Désignation produit	Chaise plastique	P	
14	Prix unitaire produit	13,00 €	P	
~~15~~	~~Montant ligne facture~~	52,00 €	C	[12] x [14]
~~16~~	~~Total HT facture~~	2 812,60 €	C	Σ [15]
~~17~~	~~Taux TVA~~	20%	P	*Paramètre*
~~18~~	~~Montant TVA facture~~	562,52 €	C	[16] x [17]
19	Banque client	Big Bank	P	
20	Date paiement	23/01/2019	P	
21	Numéro chèque	1234567	P	
22	Numéro compte client	11111 22222 3333333…	P	
~~23~~	~~CA HT annuel~~	12 345 €	C	Σ [16] d'un client
~~24~~	~~TOTAL HT CA~~	45 677 €	C	Σ [23]

Dictionnaire des données

Le dictionnaire des données est constitué de toutes les rubriques devant être mémorisées dans le système : par conséquent, les rubriques calculables et les paramètres sont exclus pour ne conserver que les rubriques primaires hors paramètres. Voici donc le dictionnaire des données du système d'information de la société Beaux Rivages :

- Numéro facture
- Date facture
- Nom client
- Prénom client
- Adresse client
- Code postal client
- Ville client
- Réf produit
- Quantité vendue produit
- Désignation produit
- Prix unitaire produit
- Banque client
- Date paiement
- Numéro chèque
- Numéro compte client

C'est à partir de ces données que la modélisation va pouvoir s'effectuer. Quelle que soit l'origine des informations (document, texte, …), la constitution de ce dictionnaire est une étape incontournable dans la réalisation du modèle conceptuel de données et de la structure de la base de données correspondante.

Comme nous l'avons déjà vu précédemment, si les données ne sont pas les bonnes, tout ce qui suivra en termes d'informatisation (automatisation de l'information) sera erroné, parce que construit sur de mauvaises bases.

Ce qu'il faut retenir

- L'analyse des flux permet de recenser les informations contenues sur les échanges afin de constituer le tableau des rubriques.
- Lors du recensement, il faut être attentif aux redondances, aux synonymes et aux homonymes.
- Les rubriques calculables et les paramètres sont exclus du tableau pour constituer le dictionnaire des données.

© Patrick Bergougnoux

Chapitre 4
Exercices de recensement

Ce chapitre propose trois exercices avec, pour chacun d'eux, une analyse d'informations effectuée à partir d'un compte-rendu d'interview éventuellement complété de documents présentés en annexe.

Comme dans l'exemple développé dans le chapitre précédent, il faut traiter ces trois exercices en respectant les étapes suivantes :

- Identifier les différents acteurs (internes et externes) du domaine d'étude présenté dans le compte-rendu d'interview, et recenser les flux générés par ce domaine d'étude en réalisant son modèle organisationnel de flux.
- En s'appuyant sur les flux mis en évidence par ce modèle et les documents annexés au compte-rendu d'interview, recenser l'ensemble des informations échangées dans le cadre du domaine d'étude : pour cela, identifier et numéroter les différentes informations présentes sur ces flux, et réaliser le tableau des rubriques en faisant attention aux redondances, aux synonymes et aux homonymes.
- Pour chaque rubrique, déterminer si elle est primaire ou calculable et, pour celles qui sont calculables, préciser les rubriques impliquées dans le calcul.
- Parmi les rubriques ainsi recensées, repérer celles qui n'ont qu'une seule valeur pour tout le système d'information : il s'agit de paramètres qui, comme les rubriques calculables, ne seront pas retenus pour le dictionnaire des données.

Gestion des achats d'une administration

Ce premier exercice est du même type que l'exemple traité jusqu'à présent dans cet ouvrage.

Compte-rendu d'interview

Le service achats d'une administration gère manuellement ses commandes de la façon suivante :

Le service passe une centaine de commandes par mois qui sont déclenchées en cas de besoin de réapprovisionnement sur proposition du secrétariat ; pour cela, celui-ci remplit un bon en trois exemplaires (blanc/vert/bleu). Une commande est datée, numérotée et contient en moyenne dix lignes de produits, une référence produit donnée n'étant proposée que par un seul fournisseur. La liasse est transmise au chef de service pour approbation ; celui-ci peut accepter ou refuser la commande : dans tous les cas, il retourne le bon sous 24 heures avec la mention « Accepté » ou « Rejeté ». En cas de rejet, le secrétariat archive la liasse complète dans un classeur « Rejets commandes » ; sinon, l'exemplaire bleu est mis en attente et les deux autres feuillets sont envoyés au service comptabilité. Celui-ci contrôle alors le budget : si le budget disponible est suffisant pour la commande, il porte à son tour la mention « Accepté », affecte un numéro d'engagement et signe les deux exemplaires ; il met en attente le feuillet vert, et transmet le blanc au fournisseur. Sinon, il porte la mention « Rejeté faute de budget », signe et retourne les deux exemplaires au secrétariat qui les archive avec l'exemplaire bleu dans le classeur « Rejets commandes ».

La livraison de la marchandise est accompagnée de la facture. Le secrétariat vérifie la cohérence de la marchandise, de la facture et de la commande (feuillet bleu) ; au moindre problème, la marchandise est renvoyée intégralement avec une lettre de refus. Sinon, une photocopie de la facture est agrafée à la commande, le tout étant archivé dans le classeur « Commandes soldées ». L'original de la facture est ensuite transmis au service comptabilité avec la mention « Bon à payer ».

Tous les lundis à 9 heures, le service comptabilité traite les factures de la semaine précédente : il met à jour le budget, puis il remplit un mandat administratif par fournisseur regroupant l'ensemble des factures de la semaine, à partir d'informations d'un répertoire « fournisseurs » et des références des commandes (n°, date, n° engagement) et des factures (n°, date, montant). Chaque facture est alors archivée avec le feuillet vert de la commande ainsi qu'une copie du mandat administratif dans un classeur « Commandes payées ».

À 17 heures, tous les mandats administratifs sont envoyés à la Trésorerie Générale de la commune qui assure le paiement des fournisseurs.

© Patrick Bergougnoux

Documents annexes

Administration XYZ
123 rue du chemin
31000 Toulouse
N° Engagement : 12345

Bon de Commande n° 0905 du 15/04/2019

Société DURAND
34 rue du Pont
31700 Blagnac

Référence	*Désignation*	*Quantité*	*Prix Unitaire*	*Montant TTC*
C01	Crayon	10	0,98 €	9,80 €
B04	Bloc Notes	20	3,41 €	68,20 €
G02	Gomme	15	1,20 €	18,00 €
			Total TTC	**96,00 €**

Société DURAND – 34 rue du Pont – 31700 Blagnac 30/04/2019

FACTURE FA19040012

Administration XYZ
123 rue du chemin
31000 Toulouse

Réf : Votre Commande n° 0905 du 15/04/2019

Qté	Code	Désignation	PU HT	Montant HT
10	C01	Crayon	0,82 €	8,20 €
20	B04	Bloc Notes	2,84 €	56,80 €
15	G02	Gomme	1,00 €	15,00 €

Total HT	80,00 €
TVA 20%	16,00 €
Total TTC	**96,00 €**

En votre aimable règlement

Administration XYZ
123 rue du chemin – 31000 Toulouse

Toulouse, le 02/05/2019

Société DURAND
34 rue du Pont
31700 Blagnac

Objet : Refus marchandise

Madame, Monsieur,

La livraison de la marchandise relative à votre facture FA19040012 du 30/04/2019 n'est pas conforme à notre commande n°0905 du 15/04/2019. Vous trouverez donc ci-joint la marchandise en question.

Dans l'attente d'une régularisation rapide, nous vous prions de croire, Madame, Monsieur, en l'expression de nos sincères salutations.

Le secrétariat

Administration XYZ
123 rue du chemin
31000 Toulouse

Mandat n° 4148 du 06/05/2019

Fournisseur : Société DURAND – IBAN : FR76 1234 5678 8901 2345 6789 012

N° Facture	*Date Facture*	*N° Cde*	*Date Cde*	*N° Eng.*	*Montant TTC*
FA19040003	12/04/2019	0902	05/04/2019	12333	57,50 €
FA19040010	26/04/2019	0904	11/04/2019	12342	125,00 €
			À payer		**182,50 €**

© Patrick Bergougnoux

Analyse des flux

Domaine d'étude et acteurs

Le domaine d'étude est composé de trois acteurs internes :

- Le secrétariat qui établit les bons de commande et les transmet aux autres acteurs internes ; il assure également la réception et le contrôle des marchandises et des factures.
- Le chef de service qui accepte ou rejette les bons de commande.
- La comptabilité qui gère l'envoi des commandes acceptées aux fournisseurs et le paiement des factures par mandats administratifs via la trésorerie générale.

Deux acteurs externes sont concernés :

- Les fournisseurs qui envoient la marchandise et les factures correspondantes suite à la réception des bons de commande.
- La trésorerie générale qui assure le paiement des mandats administratifs aux fournisseurs.

Modèle organisationnel de flux

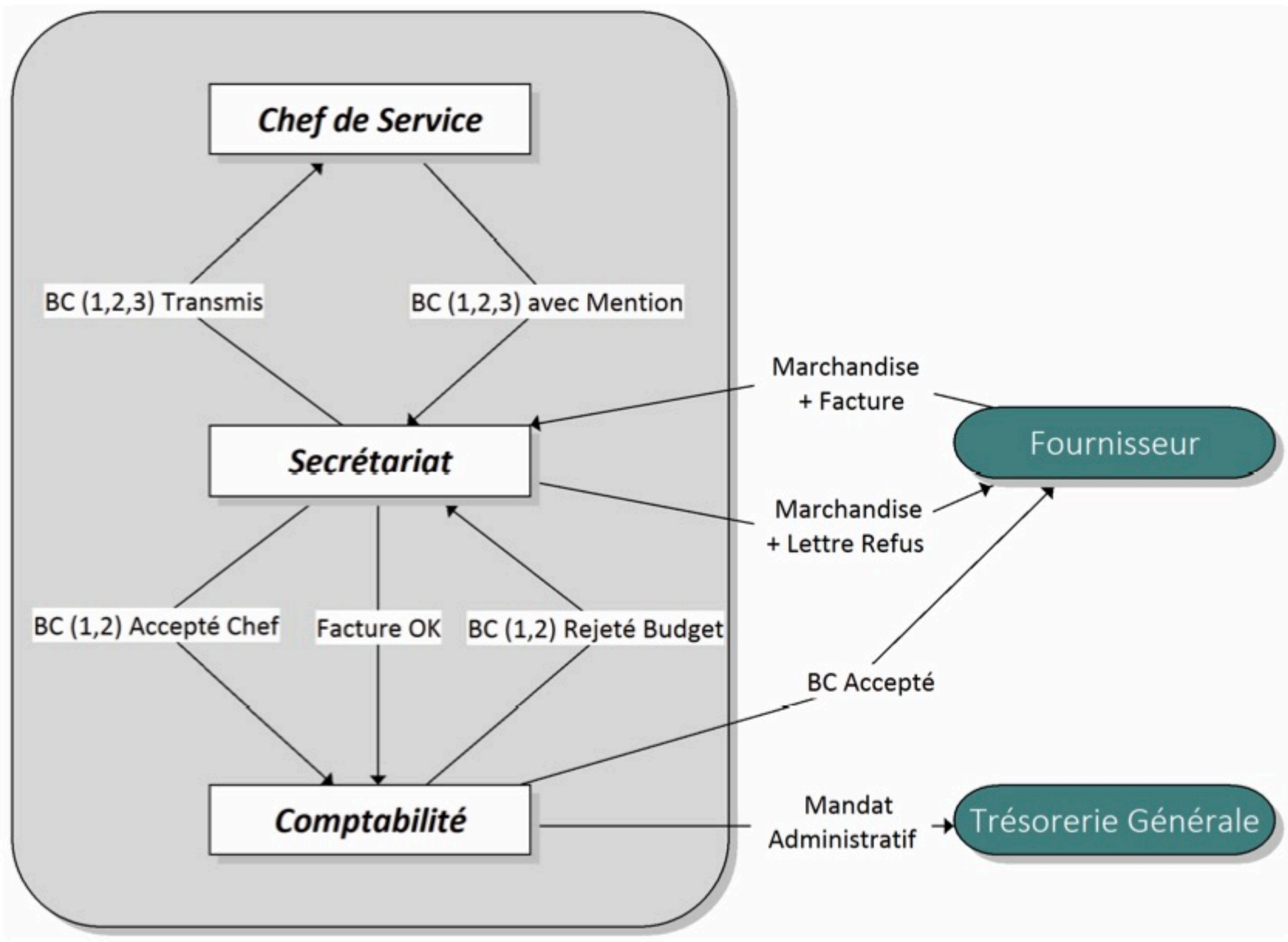


Tableau des rubriques

Après analyse du contenu des flux, toutes les informations échangées dans le cadre du domaine d'étude ont été recensées.

Le tableau des rubriques ci-dessous a été réalisé en faisant attention aux nombreuses redondances, aux synonymes et aux homonymes.

Les rubriques primaires, calculables et les paramètres ont été identifiés.

N°	Libellé	Exemple	P/C	Calcul
~~1~~	~~Nom administration~~	XYZ	P	*Paramètre*
~~2~~	~~Adresse administration~~	123 rue du chemin …	P	*Paramètre*
3	N° commande	0905	P	
4	Date commande	15/04/2019	P	
5	N° engagement	12345	P	
6	Nom fournisseur	Société Durand	P	
7	Adresse fournisseur	34 rue du Pont	P	
8	Code postal fournisseur	31700	P	
9	Ville fournisseur	Blagnac	P	
10	Référence produit	B04	P	
11	Désignation produit	Bloc Notes	P	
12	Quantité produit	20	P	
13	Prix unitaire TTC produit	3,41 €	P	
~~14~~	~~Montant TTC ligne~~	68,20 €	C	[12] x [13]
~~15~~	~~Total TTC cde et facture~~	96,00 €	C	Σ [14]
16	N° facture	FA19040012	P	
17	Date facture	30/04/2019	P	
~~18~~	~~Prix unitaire HT produit~~	2,84 €	C	[13] / (1 + [21])
~~19~~	~~Montant HT ligne~~	56,80 €	C	[12] x [18]
~~20~~	~~Total HT facture~~	80,00 €	C	Σ [19]
~~21~~	~~Taux TVA~~	20,00%	P	*Paramètre*
~~22~~	~~Montant TVA facture~~	16,00 €	C	[20] x [21]
23	Date lettre refus	02/05/2019	P	
24	N° mandat	4148	P	
25	Date mandat	06/05/2019	P	
26	IBAN fournisseur	FR76 1234 5678 8901 …	P	
~~27~~	~~À payer~~	182,50 €	C	Σ [15] d'un [6]

© Patrick Bergougnoux

Dictionnaire des données

Une fois les rubriques calculables et les paramètres écartés, voici les rubriques constituant le dictionnaire des données du système d'information de l'administration XYZ :

- N° commande
- Date commande
- N° engagement
- Nom fournisseur
- Adresse fournisseur
- Code postal fournisseur
- Ville fournisseur
- Référence produit
- Désignation produit
- Quantité produit
- Prix unitaire TTC produit
- N° facture
- Date facture
- Date lettre refus
- N° mandat
- Date mandat
- IBAN fournisseur

Société canine

Cet exercice est également basé sur l'étude des documents échangés. Le modèle organisationnel de flux est très simple, mais l'analyse du contenu de ces flux est intéressante.

Compte-rendu d'interview

La société canine « Botoutou » répertorie les chiens de race et leurs classements aux différents concours auxquels ils ont participé. Il y a une dizaine de concours chaque année auxquels participent plusieurs centaines de chiens. La société gère ainsi des milliers de chiens d'une centaine de races différentes. Les adhérents de Botoutou sont les propriétaires des chiens répertoriés.

Au début de chaque année, le secrétaire de la société envoie aux adhérents les documents suivants :

- *Répertoire des chiens avec nom, âge (an, mois) au 1er janvier dernier, sexe et race.*
- *Liste des différentes races représentées dans la société avec un intitulé, leur origine et un descriptif pour chacune d'elles.*
- *Annuaire des propriétaires avec leur nom, leur adresse complète, le nom de leurs chiens (une personne peut posséder un ou plusieurs chiens), ainsi que la date depuis laquelle ils les ont en leur possession (on ne considère que le dernier propriétaire d'un chien).*

Les adhérents peuvent alors retourner sous huit jours les documents en question avec les éventuelles corrections pour la saison à venir (changement d'adresse, nouveau chien, …).

La liste des concours de l'année à venir avec la ville d'accueil et la date leur est ensuite envoyée.

Après chaque concours, les adhérents téléphonent au secrétaire pour lui indiquer le classement de leurs chiens.

En fin d'année, le secrétaire envoie aux adhérents concernés les documents récapitulatifs suivants :

- *Liste des concours de l'année complétée avec le nombre total de chiens primés et le nombre de participants (tous n'étant pas forcément répertoriés chez Botoutou).*
- *Palmarès de chaque chien comportant la liste des concours auxquels il a participé, le classement obtenu et son âge au moment du concours.*

© Patrick Bergougnoux

Documents annexes

Botoutou

Répertoire des Chiens

Nom	Âge (au 1er janvier)	Sexe	Race
Gibbs	6 ans et 3 mois	M	Bouvier
Noisette	10 ans et 2 mois	F	Teckel
Razmoket	7 mois	M	Bouvier
Raymonde	3 ans et 5 mois	F	Berger
Wouwou	1 an et 2 mois	M	Labrador

Botoutou

Liste des Races

Intitulé	Origine	Descriptif
Berger	Allemagne	Cf. Rintintin
Bouvier	Suisse	Gros toutou à poil long
Labrador	Canada	Chien présidentiel
St Bernard	Suisse	Chien à barrique intégrée
Teckel	Allemagne	Petit et court sur pattes

Botoutou

Annuaire des Propriétaires

Nom	Adresse	Chien(s)	Acquisition
Bertrand	Paris XII	Raymonde	05/10/2016
Dupuy	St Perdou	Gibbs Wouwou	20/10/2014 25/12/2017
Escudié	Toulouse	Razmoket	01/09/2018
Mansard	Nice	Noisette	20/10/2009

Botoutou

Liste des Concours 2019

Ville	Date	Primés	Participants
Muret	20 février	5	57
Arles	4 avril	8	64
Bordeaux	15 juin	3	42
Lille	17 octobre	10	87
Strasbourg	5 décembre	8	59

© Patrick Bergougnoux

Botoutou

Palmarès de Razmoket

Concours	Date	Classement	Âge (au moment du concours)
Bordeaux	15/06/2019	3ème	1 an
Lille	17/10/2019	5ème	1 an et 4 mois
Strasbourg	05/12/2019	2ème	1 an et 6 mois

Nombre de concours : 3 / Meilleure place : 2ème

Analyse des flux

Domaine d'étude et acteurs

Le compte-rendu d'interview ne fait ressortir qu'un seul acteur interne dans le domaine d'étude : le secrétaire, qui s'occupe de tous les échanges avec les adhérents, seuls acteurs externes du système d'information. La représentation des flux se résume donc à un modèle particulièrement simple.

Modèle organisationnel de flux

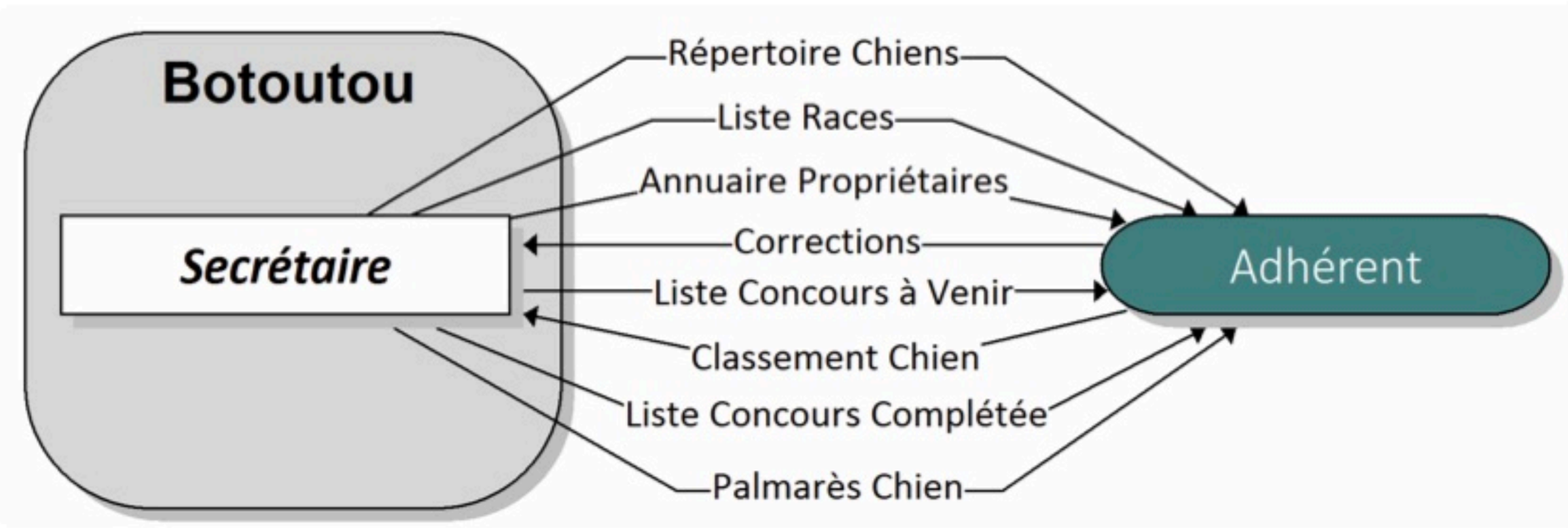


Tableau des rubriques

Le seul paramètre présent dans le compte-rendu d'interview est le nom de la société. Concernant les rubriques calculables, il faut repérer immédiatement celles liées à l'âge des chiens, et les ramener à une rubrique stable dans le temps, à savoir la date de naissance. Il sera alors possible de calculer l'âge du chien au 1er janvier dernier et au moment d'un concours (la différence entre deux dates se calcule en jours et peut ensuite être transformée en années/mois).

Le nombre de concours effectués par un chien, ainsi que sa meilleure place (minimum des classements obtenus) sont également des rubriques calculables.

Par contre, le nombre de chiens primés et le nombre de participants ne peuvent pas être calculés : en effet, comme le précise le compte-rendu d'interview, tous les chiens participant et éventuellement primés à un concours ne sont pas forcément répertoriés chez Botoutou.

Concernant la date du 1er janvier dernier, elle est unique pour tout le système d'information et est donc répertoriée en tant que paramètre.

N°	Libellé	Exemple	P/C	Calcul
~~1~~	~~Entête société~~	Botoutou	P	*Paramètre*
2	Nom chien	Razmoket	P	
~~3~~	~~Âge chien~~	7 mois	C	[19] - [20]
4	Sexe chien	M	P	
5	Intitulé race	Bouvier	P	
6	Origine race	Suisse	P	
7	Descriptif race	Gros toutou à poil long	P	
8	Nom propriétaire	Escudié	P	
9	Adresse propriétaire	Toulouse	P	
10	Date acquisition	01/09/2018	P	
11	Ville concours	Bordeaux	P	
12	Date concours	15/06/2019	P	
13	Nombre chiens primés	3	P	
14	Nombre participants	42	P	
15	Classement	5ème	P	
~~16~~	~~Âge chien lors du concours~~	1 an	C	[12] - [20]
~~17~~	~~Nb concours d'un chien~~	3	C	Nb de [11] d'un [2]
~~18~~	~~Meilleure place d'un chien~~	2ème	C	Min [15] d'un [2]
~~19~~	~~1er janvier dernier~~	01/01/2019	C	*Paramètre*
20	Date naissance chien	01/06/2018	P	

© Patrick Bergougnoux

Dictionnaire des données

Après retrait des paramètres et des rubriques calculables, le dictionnaire des données du système d'information de la société Botoutou est constitué des rubriques suivantes :

- Nom chien
- Sexe chien
- Intitulé race
- Origine race
- Descriptif race
- Nom propriétaire
- Adresse propriétaire
- Date acquisition
- Ville concours
- Date concours
- Nombre chiens primés
- Nombre participants
- Classement
- Date naissance chien

Bibliothèque universitaire

Pour cet exercice, les documents ne sont pas représentés, mais les échanges sont décrits directement dans le texte du compte-rendu d'interview. Les exemples dans le tableau des rubriques devront, pour leur part, être inventés.

Compte-rendu d'interview

Une bibliothèque universitaire gère un service de prêts aux étudiants des ouvrages dont elle dispose. La première fois qu'un étudiant emprunte un livre, il remplit une fiche de renseignements comprenant son nom, son prénom, son adresse et son numéro de carte étudiant, cette fiche étant transmise au responsable de la bibliothèque.

Sachant que la plupart des ouvrages existent en plusieurs exemplaires, le responsable peut en connaitre la disponibilité à partir de listes fournies par les éditeurs sur lesquelles il indique le nombre d'exemplaires disponibles pour chaque ouvrage ; ces listes comprennent le code de l'ouvrage, son titre et ses auteurs (caractérisés par leur nom, prénom et nationalité, un livre pouvant être écrit par un ou plusieurs auteurs).

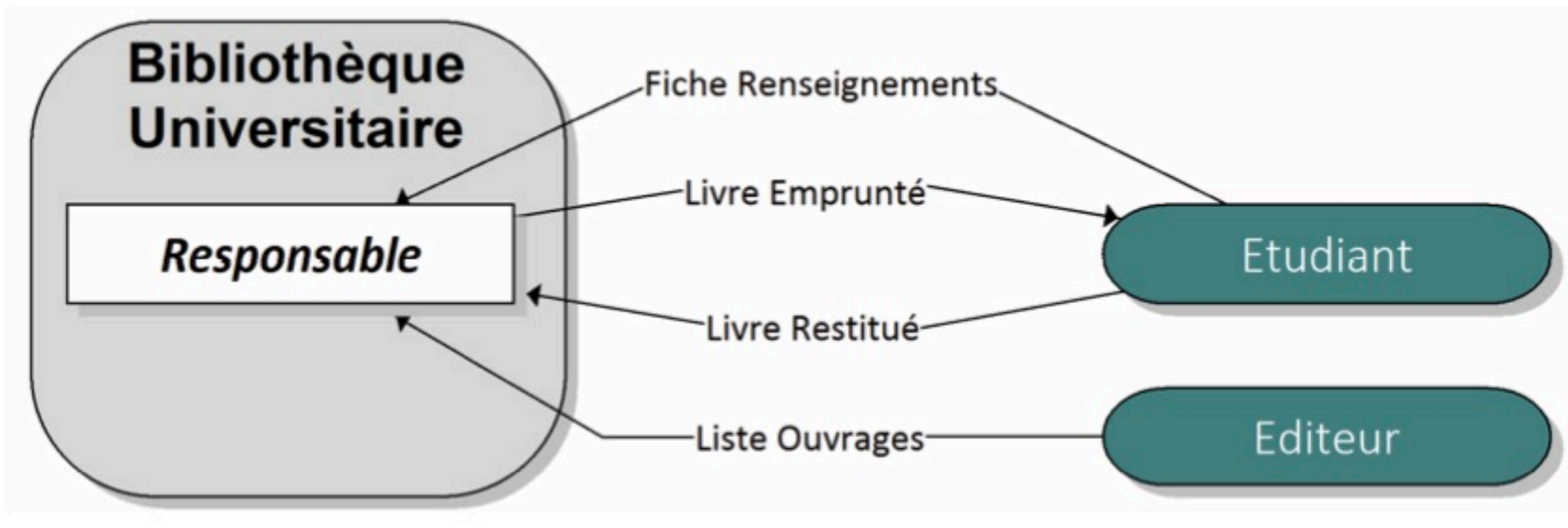
Le responsable gère également l'historique des prêts en notant, lors de chaque emprunt, le code de l'ouvrage, le numéro de carte de l'étudiant ayant pris un exemplaire, la date du prêt ainsi que la date de retour renseignée lorsque l'étudiant a ramené le livre.

Analyse des flux

Domaine d'étude et acteurs

Comme dans l'exercice précédent, le compte-rendu d'interview ne fait ressortir qu'un seul acteur interne dans le domaine d'étude : le responsable de la bibliothèque, qui échange avec les étudiants et les éditeurs.

Modèle organisationnel de flux



© Patrick Bergougnoux

Tableau des rubriques

La seule rubrique calculable est le nombre d'exemplaires disponibles qui évolue sans cesse et représente donc une mauvaise rubrique primaire : cette disponibilité peut être calculée à partir du nombre total d'exemplaires auquel on retire le nombre de livres prêtés, lui-même calculable à partir des dates de retour non encore renseignées ; en effet, une date de retour n'est renseignée que lorsque l'étudiant ramène le livre emprunté.

N°	Libellé	Exemple	P/C	Calcul
1	Nom étudiant	Dupont	P	
2	Prénom étudiant	Alain	P	
3	Adresse étudiant	Rue 3 as – 06000 Nice	P	
4	Numéro carte étudiant	123456	P	
5	Code ouvrage	A42	P	
6	Titre ouvrage	20 cm sur la mer	P	
7	Nom auteur	Vernis	P	
8	Prénom auteur	Jules	P	
9	Nationalité auteur	Français	P	
~~10~~	~~Nb exemplaires dispo~~	10	C	[13] - [14]
11	Date prêt ouvrage	11/02/2019	P	
12	Date retour ouvrage	25/02/2019	P	
13	Nb total exemplaires	25	P	
~~14~~	~~Nb exemplaires prêtés~~	15	C	Nb[12] vide d'un [5]

Dictionnaire des données

Voici les rubriques constituant le dictionnaire des données du système d'information de cette bibliothèque universitaire :

- Nom étudiant
- Prénom étudiant
- Adresse étudiant
- Numéro carte étudiant
- Code ouvrage
- Titre ouvrage
- Nom auteur
- Prénom auteur
- Nationalité auteur
- Date prêt ouvrage
- Date retour ouvrage
- Nb total exemplaires

Notes

Notes

Deuxième partie

Bases de la modélisation

Cette deuxième partie est consacrée à l'étude des éléments de base permettant la réalisation d'un modèle conceptuel de données à partir du dictionnaire des données. Elle est volontairement incomplète et se limite aux seuls concepts indispensables à la mise en œuvre de modèles conceptuels simples, sans se soucier des problèmes techniques liés à leur future informatisation. Des modèles plus complexes seront traités dans les autres parties de cet ouvrage.

L'objectif est de permettre au lecteur de réaliser ses premiers modèles de la manière la plus naturelle possible, sans recours à la théorie sous-jacente.

Chapitre 5
Répartition en classes d'entités

Ce chapitre est probablement le plus important du livre car il va permettre de définir la structure générale du modèle. Afin de bien comprendre la démarche, le même exemple simple, la facturation Beaux Rivages, est développé tout au long de ce chapitre et des deux suivants. Les autres exercices traités au chapitre 4 seront également repris en fin de cette deuxième partie.

Répartition des données

Nécessité d'organiser les données en sous-ensembles

Dans les exemples étudiés dans cet ouvrage, le nombre de rubriques est volontairement limité de manière à ne pas noyer la démarche sous une masse de données. Cependant, dans les cas réels que l'on retrouve en milieu professionnel, le nombre d'informations est souvent bien supérieur et il n'est pas rare d'avoir à gérer plusieurs centaines de rubriques.

Il parait alors difficile d'imaginer que toutes ces rubriques vont faire l'objet d'un grand tableau dans lequel l'ensemble des informations seront regroupées. Au-delà d'un problème de normalisation liée à la redondance des données, se poseraient des problèmes de lisibilité, d'accessibilité et de compréhension de cette masse d'informations.

Il apparait donc nécessaire d'organiser les données de manière à les rendre lisibles et accessibles. Pour cela, l'approche la plus pragmatique consiste à les répartir en différents sous-ensembles qui traitent chacun d'un thème bien identifié.

Trouver des objets significatifs du domaine d'étude

Les thèmes définis à partir des rubriques du dictionnaire des données doivent ressortir clairement du compte-rendu d'interview : ils prennent la forme d'éléments présents dans le domaine d'étude. Il peut s'agir de personnes physiques, d'entreprises, de services d'une organisation, d'objets divers, de documents, …

Le plus important à ce stade de l'analyse est de ne pas trop réfléchir !

En effet, si on étudie le système d'information trop en détail, on trouve des « objets significatifs » partout, et chaque rubrique devient potentiellement un sous-ensemble à elle toute seule…

De plus, il faut baser la réflexion sur le seul dictionnaire des données, et pas sur le modèle organisationnel de flux ; sinon, le risque est de considérer systématiquement que les acteurs et les flux sont les sous-ensembles recherchés : ce sera parfois le cas, mais pas toujours.

Pour déterminer les bons sous-ensembles, il faut se focaliser sur les rubriques du dictionnaire des données et déterminer en quelques secondes les objets qui ressortent de manière évidente.

Alors, bien sûr, il y a un risque que ce premier choix soit erroné, et que certains sous-ensembles soient définis à tort et/ou que d'autres soient oubliés. Mais ce n'est pas grave ; en effet, le respect d'un petit nombre de règles que nous allons étudier par la suite nous ramènera vers le bon résultat et ce, quels que soient les choix faits au départ. Deux types de changements pourront ainsi être effectués :

- Si un sous-ensemble est défini à tort, il se retrouvera sans rubrique ou se verra fusionner avec un autre : il sera alors supprimé.
- Si, après avoir effectué toutes les affectations possibles de rubriques aux différents sous-ensembles existants, certaines rubriques se retrouvent toujours sans affectation, il sera toujours temps de rajouter le sous-ensemble adéquat.

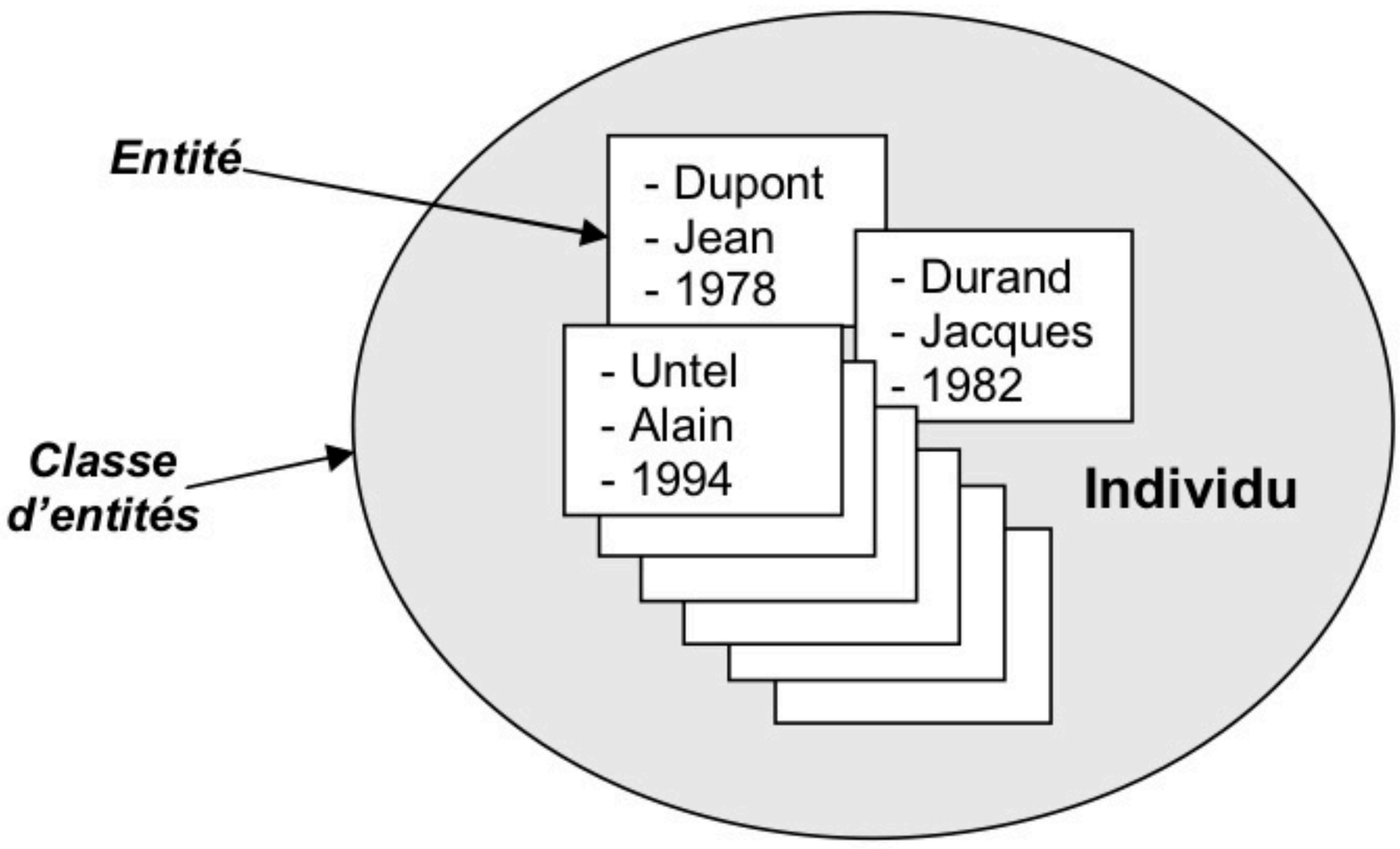
D'une façon générale, c'est moins grave d'oublier des sous-ensembles que d'en trouver trop : avec un faible nombre de sous-ensembles, la répartition des rubriques est plus facile et les éventuels rajouts ne posent pas de problème.

© Patrick Bergougnoux

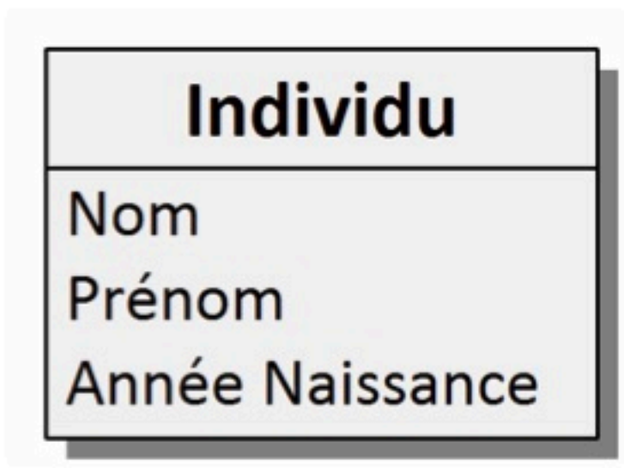
Éléments syntaxiques et représentation

Un sous-ensemble est composé d'objets possédant la même structure (mêmes rubriques) : ces objets sont appelés des « **entités** » et l'ensemble de ces entités composent une « **classe** ».

Imaginons un système d'information avec des individus caractérisés par leur nom, leur prénom et leur année de naissance. La classe d'entités « Individu » peut se schématiser comme suit :



Sa structure est alors représentée de la façon suivante :



Toutes les entités d'une même classe possèdent la même structure et sont donc caractérisées par les mêmes rubriques.

Règle d'affectation à une classe d'entités

Affectation exclusive de toutes les rubriques

Tout d'abord, il est important de s'assurer que toutes les rubriques sont affectées à une classe d'entités, et à une seule ; une rubrique ne peut donc pas être affectée plusieurs fois. Afin de s'assurer du respect de cette règle, le plus sûr est de cocher les rubriques, au fur et à mesure de leur affectation, dans le dictionnaire des données pour s'assurer de n'en oublier aucune et de ne les affecter qu'une seule fois.

Une rubrique ne peut être affectée qu'à une seule classe d'entités.

Principale règle d'affectation

Au moment d'affecter une rubrique à une classe d'entités, il faut s'assurer du respect d'une règle essentielle et incontournable :

Une rubrique n'est affectée à une classe d'entités que s'il n'existe qu'une seule valeur de cette rubrique pour chaque entité de la classe.

Le seul respect de cette règle évitera les plus grosses erreurs : cela ne suffira peut-être pas pour réaliser un modèle parfait et optimisé mais, à défaut d'être parfaitement juste, il ne sera pas tout à fait faux ! Nous reviendrons plus tard sur cette subtilité.

Application à l'exemple

Pour la facturation Beaux Rivages, trois classes d'entités se détachent de manière évidente :

- Client
- Facture
- Produit

D'autres classes, telles que le chèque ou la banque, peuvent éventuellement apparaitre si l'on s'attarde trop longtemps sur l'analyse : nous traiterons ces situations un peu plus loin dans cet ouvrage, et nous verrons que les règles de cohérence des modèles nous ramèneront finalement aux trois classes définies ci-dessus.

© Patrick Bergougnoux

La première étape consiste à affecter les rubriques vraiment évidentes.

Le nom, le prénom, l'adresse, le code postal et la ville du client semblent caractériser de manière évidente la classe d'entités « Client ». Vérifions cependant le respect de la règle d'affectation :

N'existe-t-il qu'une seule valeur de nom client pour chaque client ?
ou
Un client donné n'a-t-il qu'un seul nom ?

La réponse est « **OUI** » : par conséquent la rubrique « Nom client » peut être affectée à la classe d'entités « Client ». Nous verrons plus tard que cette seule règle ne garantit pas que le modèle soit parfaitement optimisé mais, quoiqu'il en soit, ce n'est pas fondamentalement faux.

Il en est de même pour le prénom (on se limite ici au prénom usuel). Concernant l'adresse, il faut s'assurer que chaque client n'a effectivement qu'une seule adresse : considérons dans notre exemple que c'est le cas. Les rubriques « Adresse client », « Code postal client » et « Ville client » caractérisent donc aussi la classe d'entités « Client ».

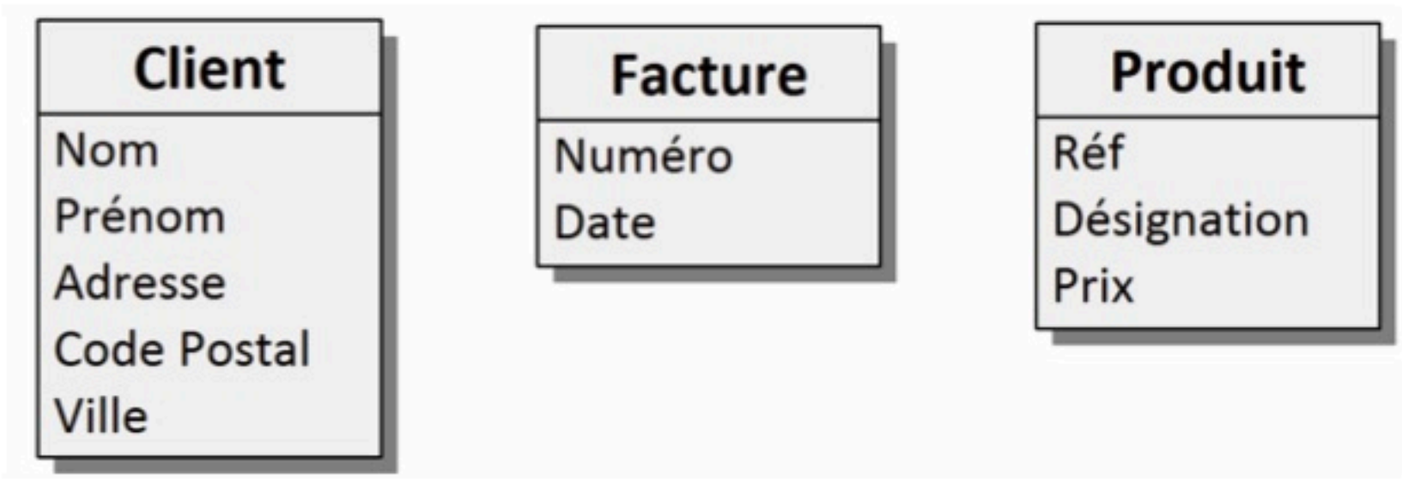
Si les clients possèdent plusieurs adresses, l'adresse ne peut pas appartenir à la classe d'entités « Client » : on ne sait pas alors forcément où la placer, mais on sait qu'on ne peut pas l'affecter à cette classe.

Une facture donnée n'ayant qu'un seul numéro et une seule date, ces deux rubriques peuvent être affectées à la classe d'entités « Facture ».

Un produit donné n'a qu'une seule référence, une seule désignation et un seul prix unitaire : ces trois rubriques peuvent donc être affectées à la classe d'entités « Produit ».

Le nom de la classe d'entités indiquant clairement la nature des informations qu'elle contient, le nom des rubriques peut être simplifié en ne reprenant pas forcément le nom de la classe.

Ces premières affectations nous donnent le début de modèle suivant :



Faisons le point sur les rubriques du dictionnaire des données qui n'ont pas encore été affectées à une classe d'entités :

- Quantité vendue produit
- Banque client
- Date paiement
- Numéro chèque
- Numéro compte client

Pour chacune de ces rubriques dont l'affectation est moins évidente, essayons de les placer dans les classes d'entités existantes. Le compte-rendu d'interview précise que :

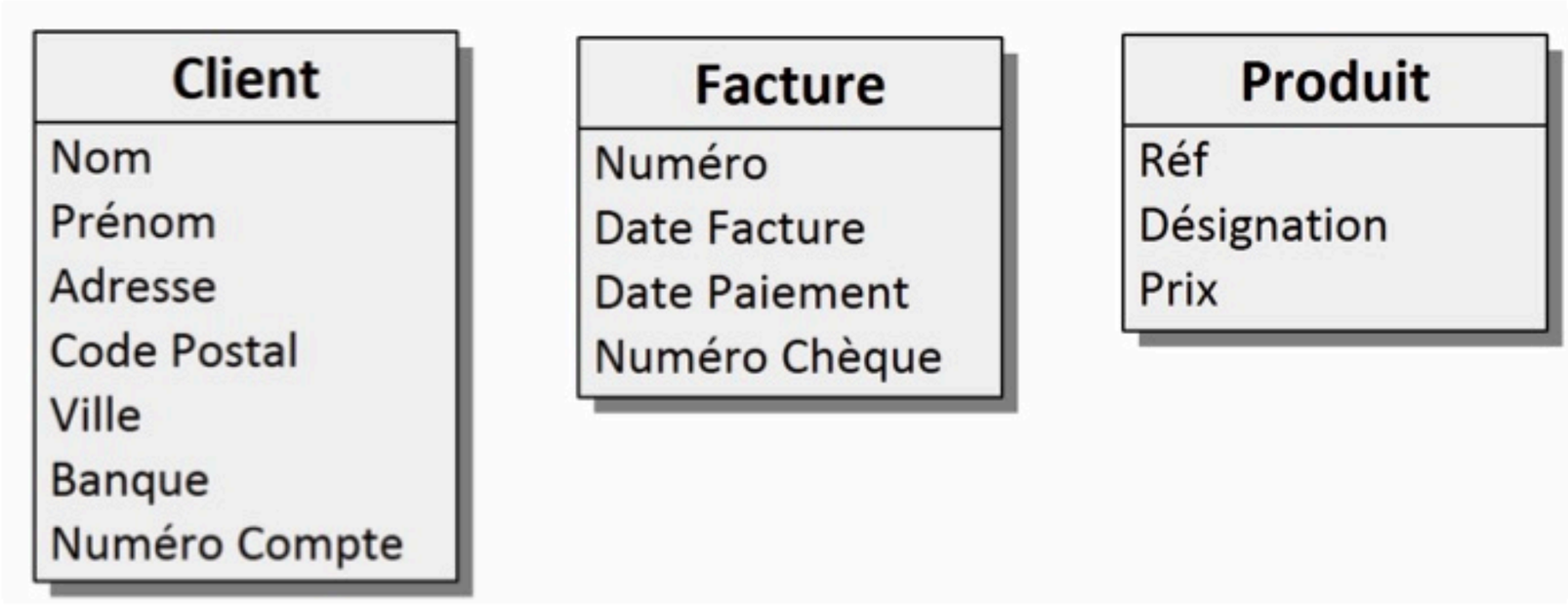
> *Lors de l'arrivée du courrier vers 11 heures, le service comptabilité réceptionne les paiements de clients : pour simplifier, on considère que le paiement d'une facture est effectué par un et un seul chèque (1 facture ➔ 1 chèque).*

Par conséquent, dans notre cas, une facture donnée n'a qu'une seule date de paiement et un seul numéro de chèque : ces deux rubriques peuvent alors être affectées à la classe d'entités « Facture ».

Précisons que, dans la plupart des cas réels, une facture peut être réglée en plusieurs fois : dans ces cas, ces deux rubriques ne pourraient pas être affectées à la classe d'entités « Facture ».

Dans un même souci de simplicité, nous considérons dans cet exercice qu'un client donné n'utilise qu'un seul compte bancaire : les rubriques concernées « Banque client » et « Numéro compte client » peuvent donc être affectées à la classe d'entités « Client ».

Après ces nouvelles affectations, voici le modèle de données :



La classe d'entités « Facture » possédant maintenant deux dates (facture et paiement), il convient de préciser le libellé de chacune des rubriques.

© Patrick Bergougnoux

Reste le cas de la quantité vendue :

- On pourrait être tenté de l'affecter à la classe d'entités « Produit ». Vérifions : des chaises plastiques peuvent être vendues avec une quantité de 4 dans la facture FA19010123 et en quantité de 6 dans la facture FA19050217. La quantité n'a donc pas une seule valeur pour un produit donné : la principale règle d'affectation n'est alors pas respectée ; je ne sais pas où doit aller cette rubrique mais je sais qu'elle ne peut pas aller là !
- Essayons de l'affecter à la classe « Facture » : dans notre exemple, la quantité est égale à 4 pour les chaises plastiques et à 2 pour les tables rectangulaires… La quantité n'a donc pas une seule valeur pour une facture donnée : la principale règle d'affectation n'est alors pas respectée ; je ne sais pas où doit aller cette rubrique mais je sais qu'elle ne peut pas aller là ! **Je sais : je me répète… mais c'est fait exprès pour que vous vous en souveniez ! C'est important !**

Le plus important est de respecter la règle pour ne pas faire faux, et on verra que, à force de ne pas faire faux, on finit par faire juste !

Affectation des rubriques orphelines

Que faire si une rubrique, comme cette quantité vendue, n'a été affectée à aucune des classes existantes et se retrouve donc orpheline de classe ?

Vérifier qu'aucune classe d'entités ne convient

Comme nous avons essayé de le faire ci-dessus pour la quantité vendue, il est vraiment important de s'assurer qu'aucune des classes existantes ne convienne. Pour cela, il suffit de vérifier la principale règle d'affectation pour la rubrique en question et ce, avec toutes les classes d'entités susceptibles de l'accueillir.

Si aucune affectation n'est possible, il faut alors explorer d'autres possibilités.

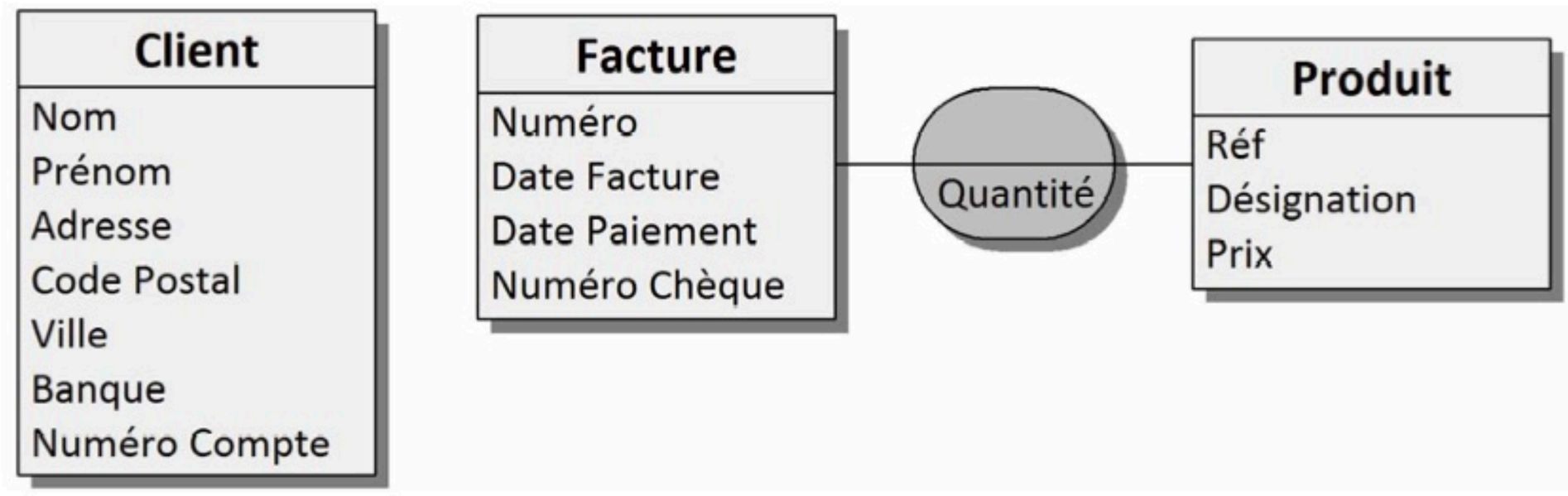
Pas une, mais deux classes concernées

Avant de se précipiter vers une solution radicale consistant à créer une nouvelle classe d'entités, étudions une nouvelle possibilité. Nous avons vu que :

- La quantité ne caractérise pas le produit parce que sa valeur peut varier en fonction des factures sur lesquelles se trouve le produit : pour un produit donné, il n'y a donc pas qu'une seule quantité vendue.
- La quantité ne caractérise pas la facture parce que sa valeur dépend des produits présents sur la facture : pour une facture donnée, il n'y a donc pas qu'une seule quantité vendue.

Mais, pour un produit donné **ET** une facture donnée, il n'y a effectivement qu'une seule quantité vendue. La quantité vendue caractérise donc l'association entre un produit et une facture.

Sur le modèle, nous positionnerons cette dernière rubrique du dictionnaire des données entre les deux classes d'entités concernées :



D'une façon générale, une ou plusieurs rubriques peuvent ainsi être affectées à ce type d'association.

Il ne faut jamais affecter une rubrique à une association avant de s'être assuré avec certitude qu'elle ne va pas dans l'une ou l'autre des classes d'entités associées.

Définition d'une nouvelle classe d'entités

Dans le cas où une rubrique ne peut être affectée ni à une classe d'entités, ni à une association, il convient alors de définir une nouvelle classe pour accueillir la rubrique en question.

Cette possibilité répond à la problématique du début de chapitre où les classes d'entités sont définies de manière rapide et intuitive : si une classe n'est pas détectée au départ, elle le sera forcément au moment d'affecter les rubriques orphelines.

Ce qu'il faut retenir

- Toutes les rubriques du dictionnaire des données doivent être réparties dans différentes classes d'entités.
- Une rubrique ne peut avoir qu'une seule valeur pour chaque entité d'une classe.
- Une rubrique orpheline de classe d'entités peut être affectée à une association entre deux classes.

© Patrick Bergougnoux

Chapitre 6
Association des classes d'entités

Le fait d'associer différentes classes d'entités d'un modèle va au-delà de la nécessité d'affecter les rubriques orphelines de classe. En effet, ces associations ont également un grand intérêt sémantique en permettant de préciser les liens qui existent entre les différentes informations présentes dans le modèle.

Ce chapitre détaille la mise en œuvre complète de ces associations permettant de finaliser les modèles conceptuels de données.

Sémantique des modèles

Nommer les associations existantes

Tout d'abord, il est intéressant de donner un nom aux associations qui ont été créées pour accueillir les rubriques orphelines. Ce nom a pour objectif de rajouter de la sémantique au modèle.

Pour cela, l'utilisation de verbes à l'infinitif est particulièrement adaptée : en effet, la conjugaison du verbe dans les deux sens fait clairement apparaitre le lien sémantique entre les deux classes associées.

Ainsi, dans notre exemple de facturation Beaux Rivages, l'association entre les classes « Facture » et « Produit » peut porter le nom « Contenir » et se conjuguer de la façon suivante :

- Les factures contiennent des produits.
- Les produits sont contenus dans des factures.

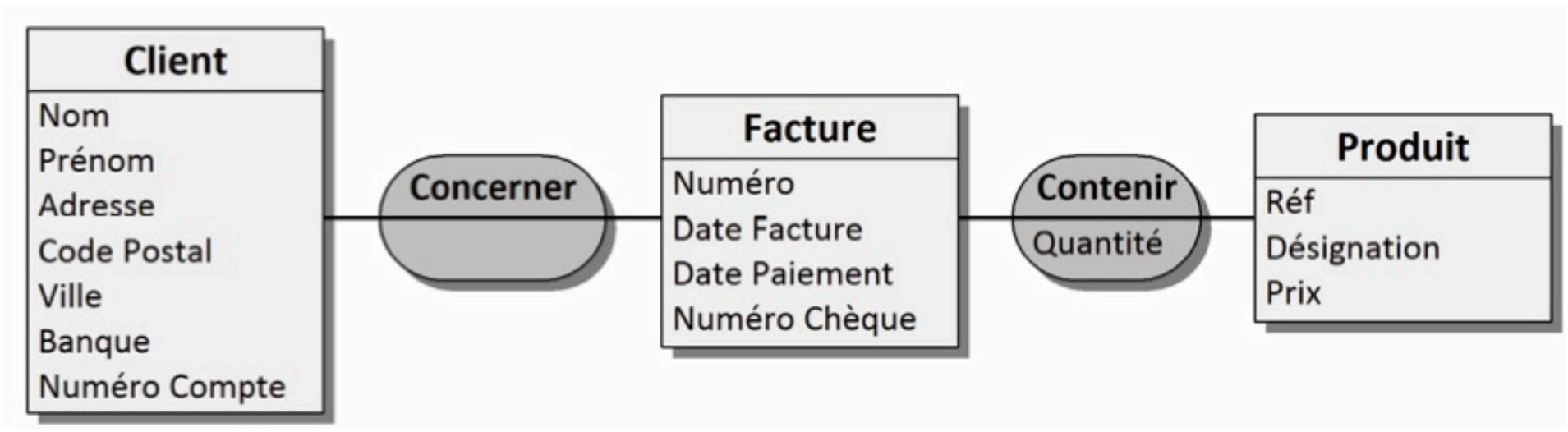
Dans un modèle de données, toutes les associations doivent être nommées.

Rajouter les associations décrites lors de l'analyse

D'autres éléments de sémantique peuvent venir traduire l'exposé du compte-rendu d'interview. Il est alors nécessaire de créer les liens correspondants grâce à de nouvelles associations entre classes d'entités, même si aucune rubrique n'est affectée à ces associations.

Dans notre exemple, il est ainsi judicieux de créer une association entre les classes d'entités « Client » et « Facture ».

Le modèle de données prend donc la forme suivante :



Cardinalités

Étude des liens possibles dans une association

Ces associations créées entre les classes d'entités peuvent également être précisées en termes de dénombrement des liens possibles entre les entités des différentes classes : on parle alors de **cardinalités**.

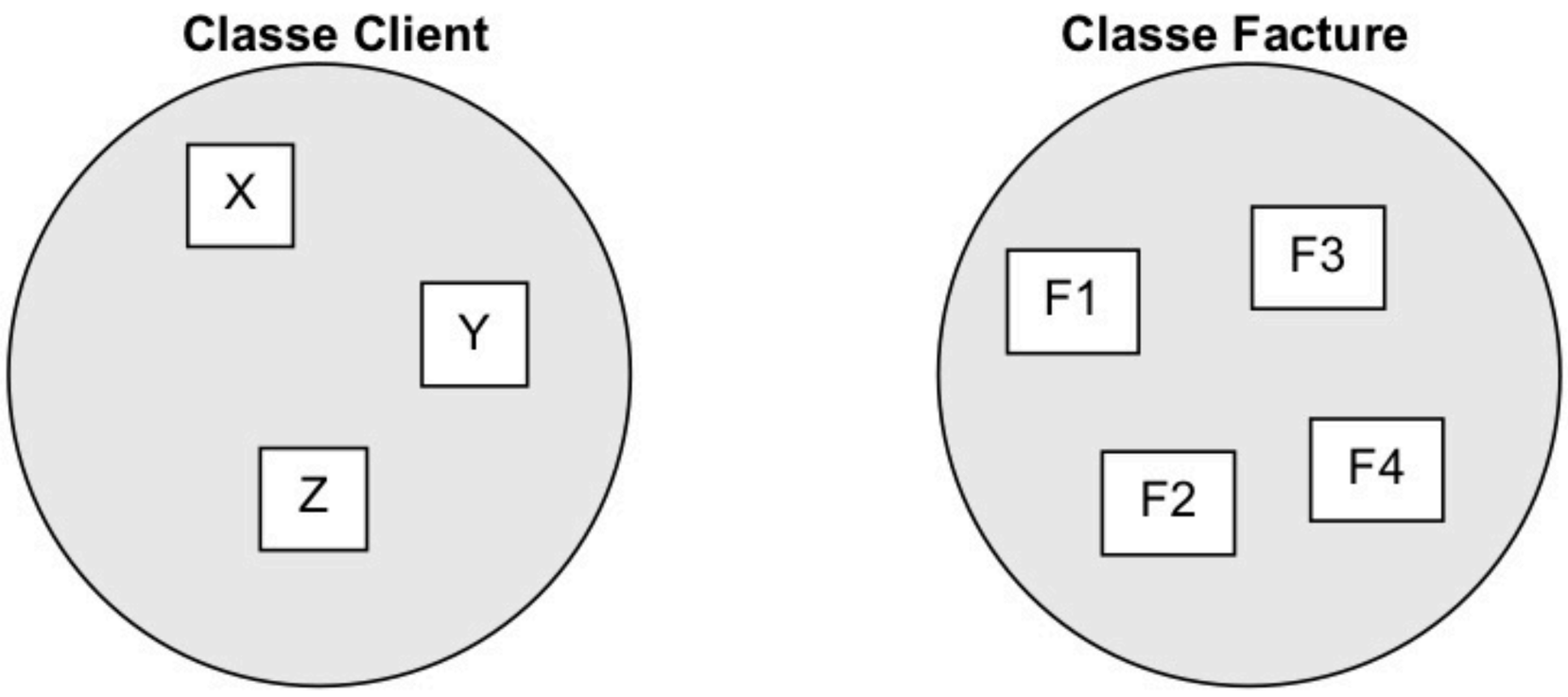
Cette problématique s'appuie sur la théorie des ensembles, avec les notions d'injection, de surjection et de bijection. Bien que cette base mathématique soit la plus appropriée à ce type de problématique, elle présente des difficultés de compréhension que cet ouvrage tente de contourner.

Nous préfèrerons donc ici une approche plus littérale et pragmatique qui nous conduira au même résultat.

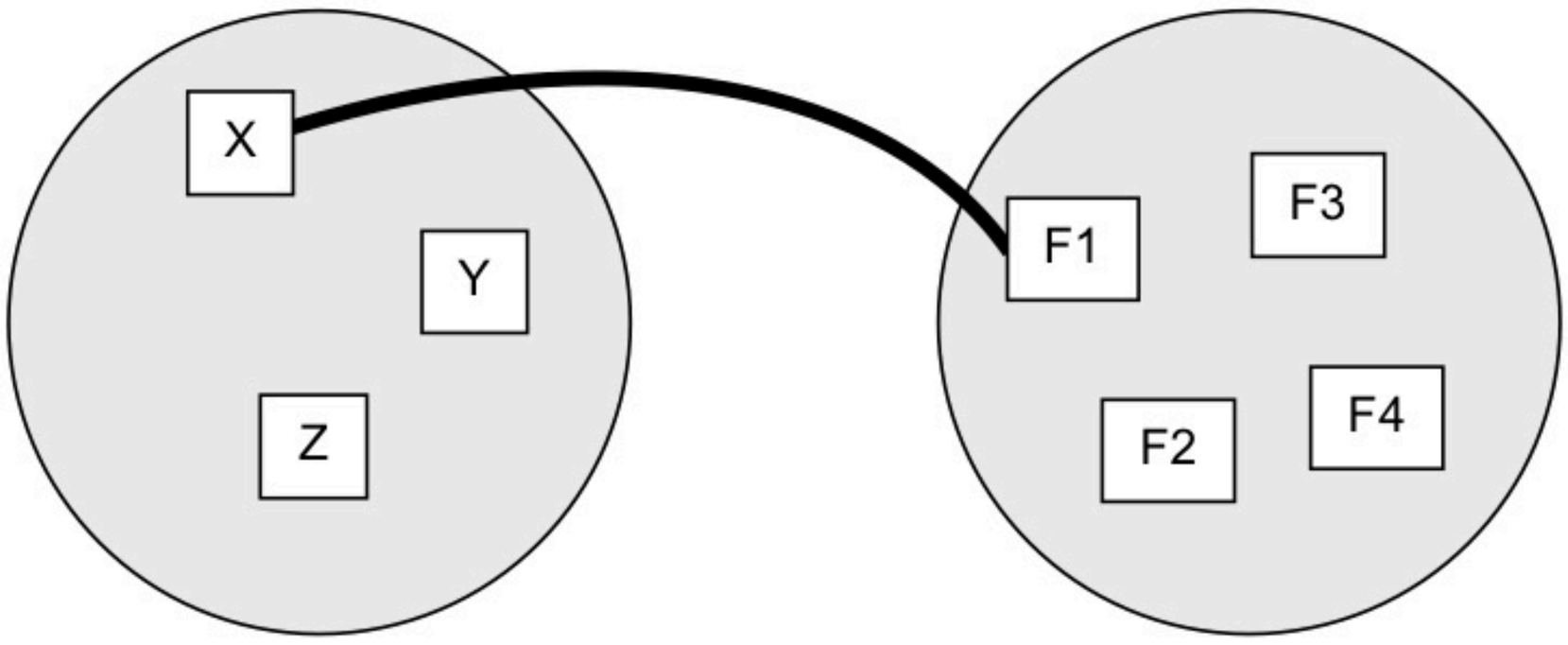
© Patrick Bergougnoux

Pour bien comprendre la signification de ces cardinalités, étudions les liens possibles entre la classe « Client » et la classe « Facture ».

Prenons 3 clients (X, Y et Z) et 4 factures (F1, F2, F3 et F4) :

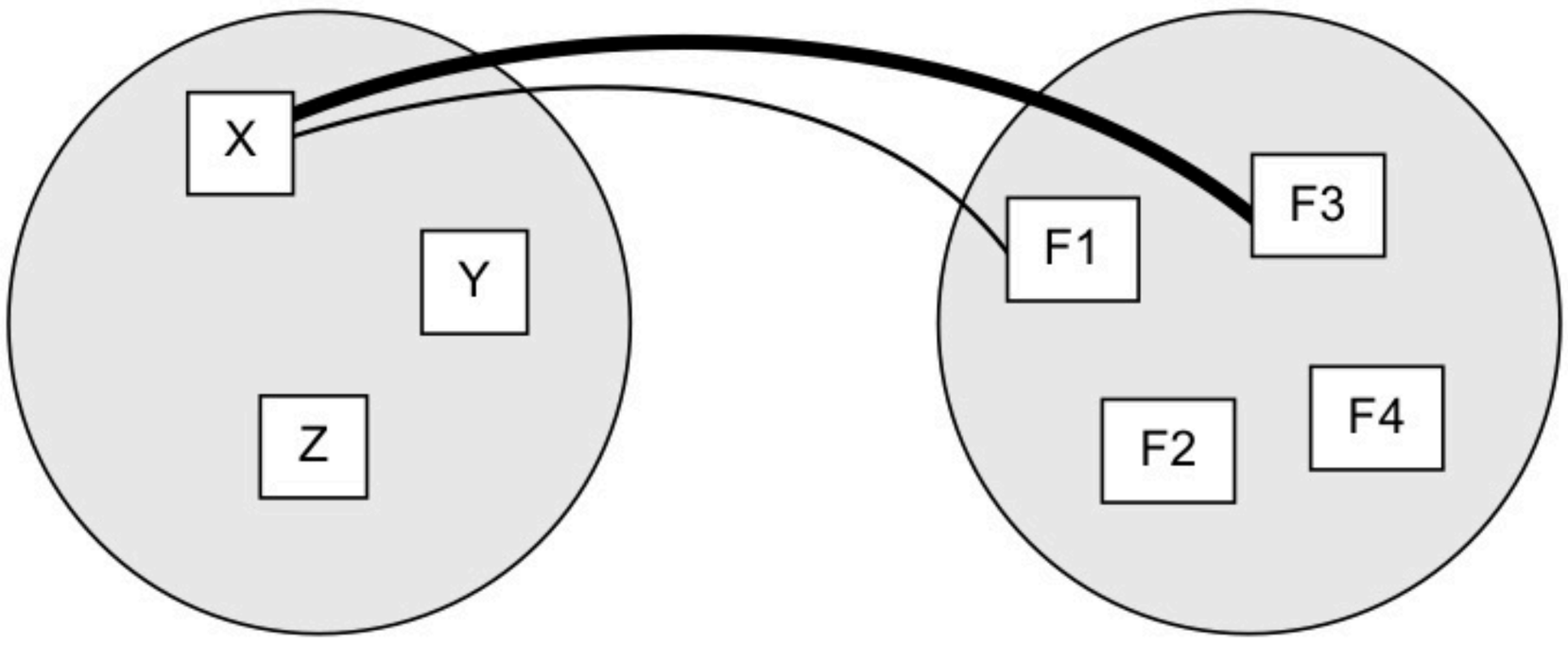


Peut-on faire le lien ci-dessous ?



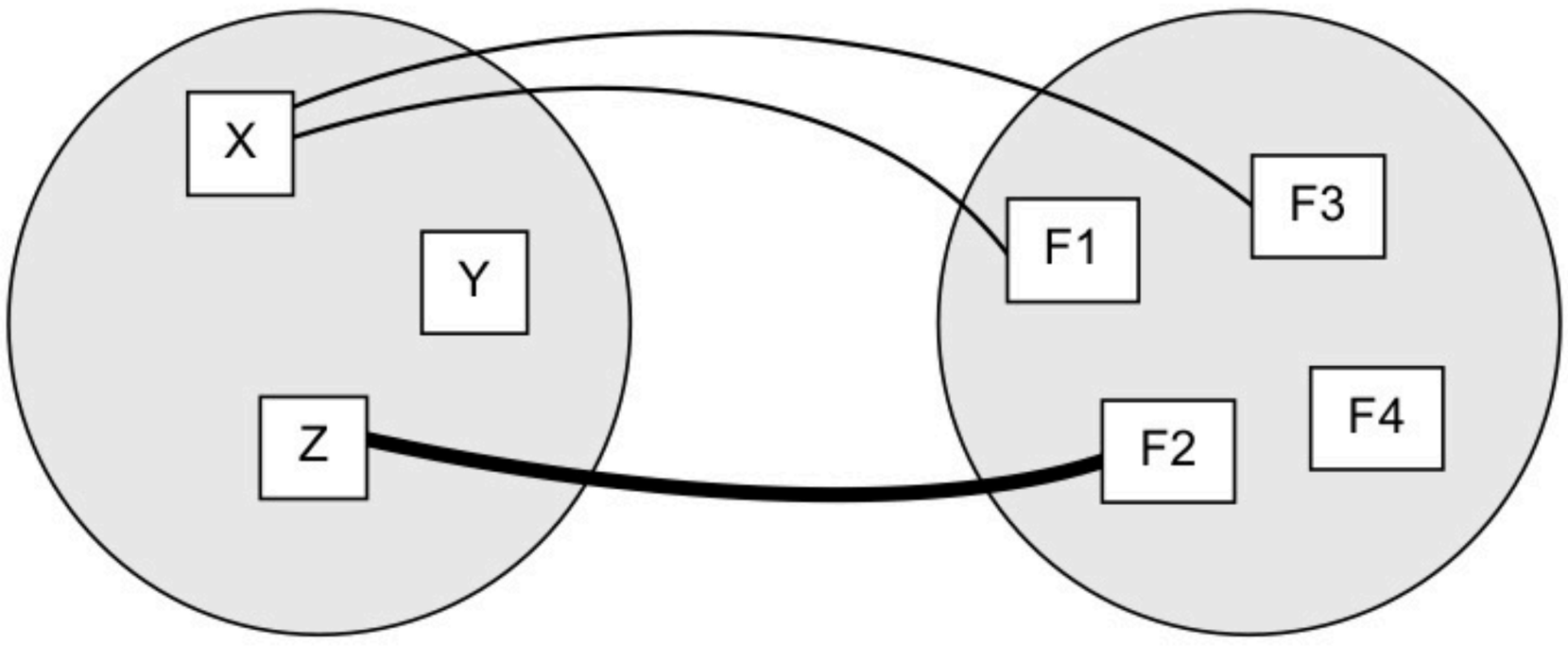
➔ **OUI** : le client X est concerné par la facture F1.

Peut-on rajouter le lien ci-dessous ?



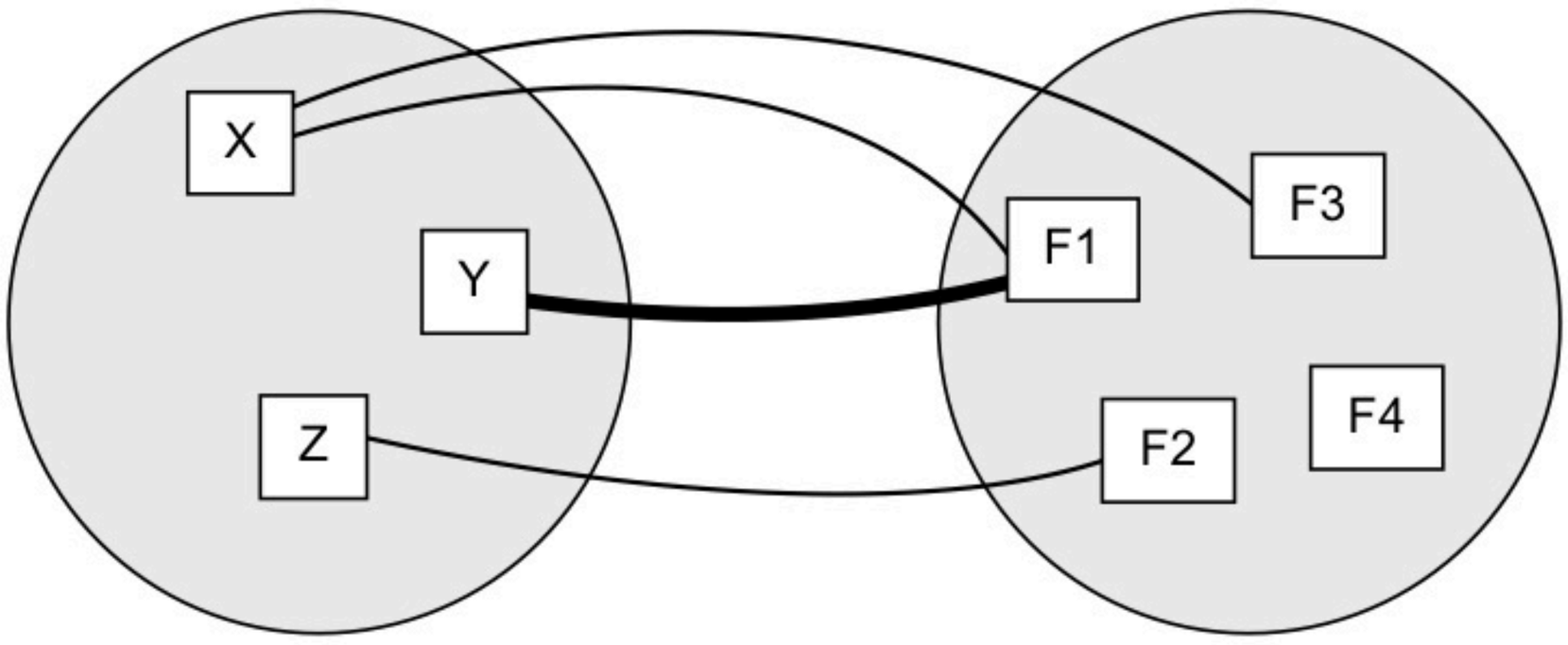
➔ **OUI** : le client X peut être concerné par plusieurs factures (ici : F1 et F3).

Peut-on rajouter le lien ci-dessous ?



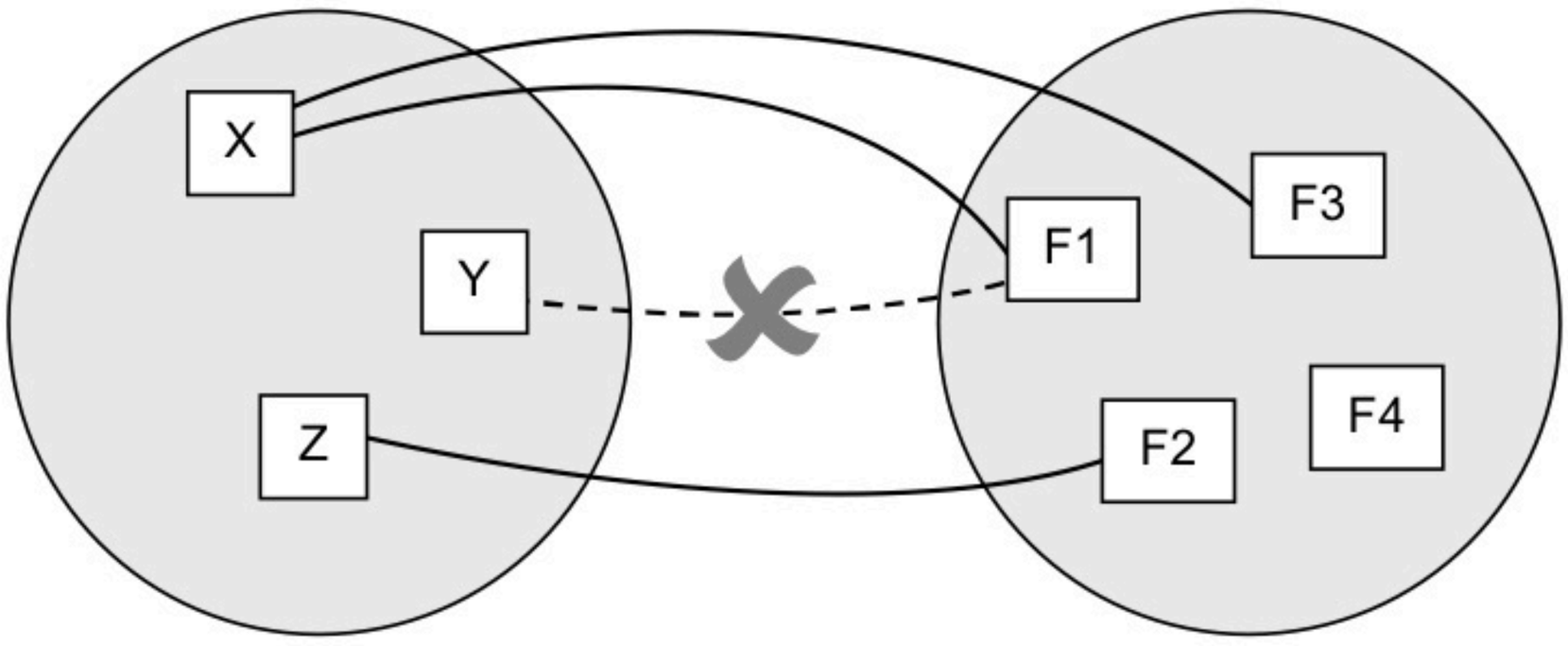
➔ **OUI** : aucun problème, la facture F2 concerne ici le client Z.

Peut-on rajouter le lien ci-dessous ?



➔ **NON !** Une même facture ne peut pas concerner plusieurs clients.

Ce lien doit donc être interdit ; par ailleurs peut-on considérer que le schéma ci-dessous est finalisé ?



➔ **NON !** Une facture doit forcément concerner un client, ce qui n'est pas le cas de F4.

© Patrick Bergougnoux

La question peut également se poser pour le client Y qui se retrouve sans facture : est-il vraiment un client s'il n'a jamais reçu de facture ? Ce genre de questions, dans différents contextes, se pose souvent, sachant que la réponse est liée à l'interprétation qui est faite du système d'information.

Ce sont les possibilités de liens entre deux classes d'entités qui définissent les cardinalités d'une association.

Notation des cardinalités dans le modèle de données

Bien évidemment, les schémas que nous venons de voir peuvent difficilement être joints à un modèle de données pour en décrire les cardinalités.

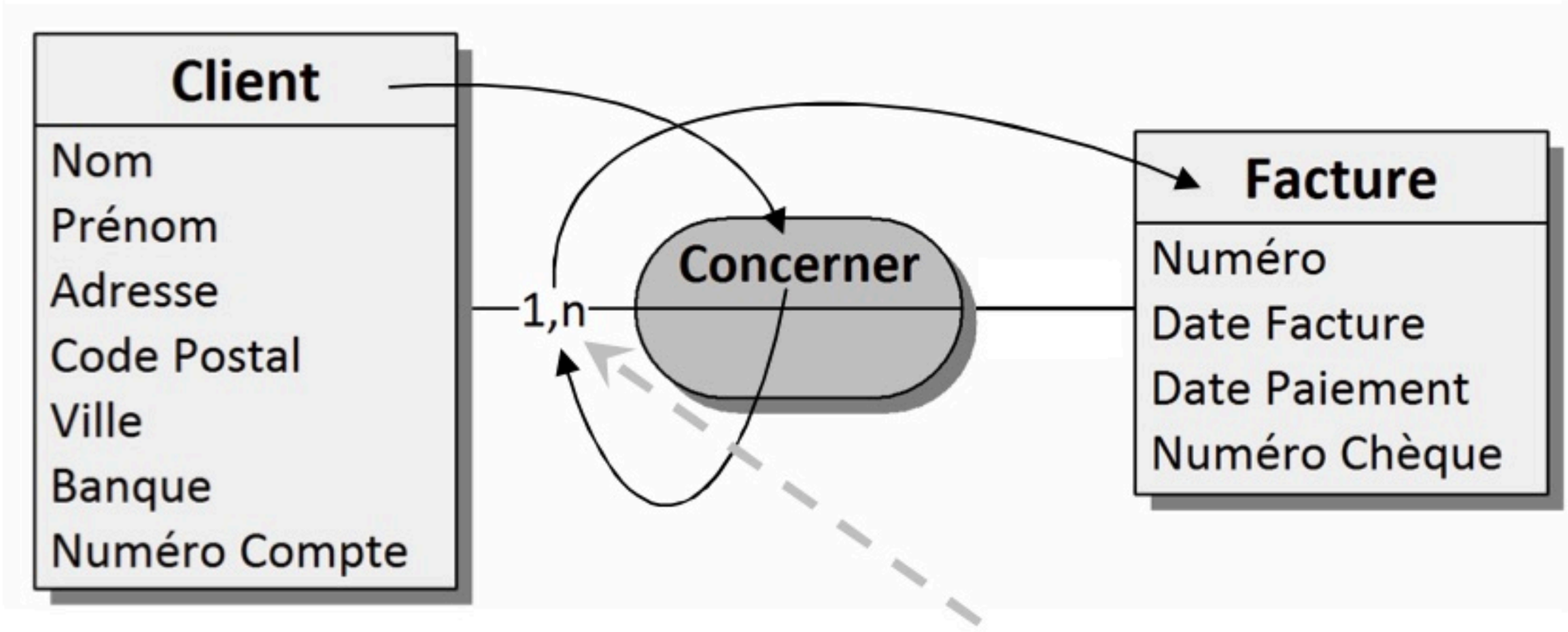
Fort heureusement, les différentes situations peuvent être résumées de manière simple grâce à une notation synthétique directement déduite des associations présentes sur le modèle de données.

Reprenons notre exemple ; en conjuguant le verbe désignant l'association, les liens entre les entités des deux classes peuvent se résumer de la façon suivante :

Chaque CLIENT est concerné par ___ à ___ FACTURE
et
Chaque FACTURE concerne ___ à ___ CLIENT

Il suffit alors de compléter les vides (l'infini est noté « **n** ») et de reporter le résultat dans le modèle de données.

Attention, la position des cardinalités peut surprendre : la lecture s'effectue en réalisant une boucle, une sorte de ***Looping*** !



Chaque CLIENT est concerné par **1** à **n** FACTURE

Procédons ainsi pour l'autre « patte » de l'association :

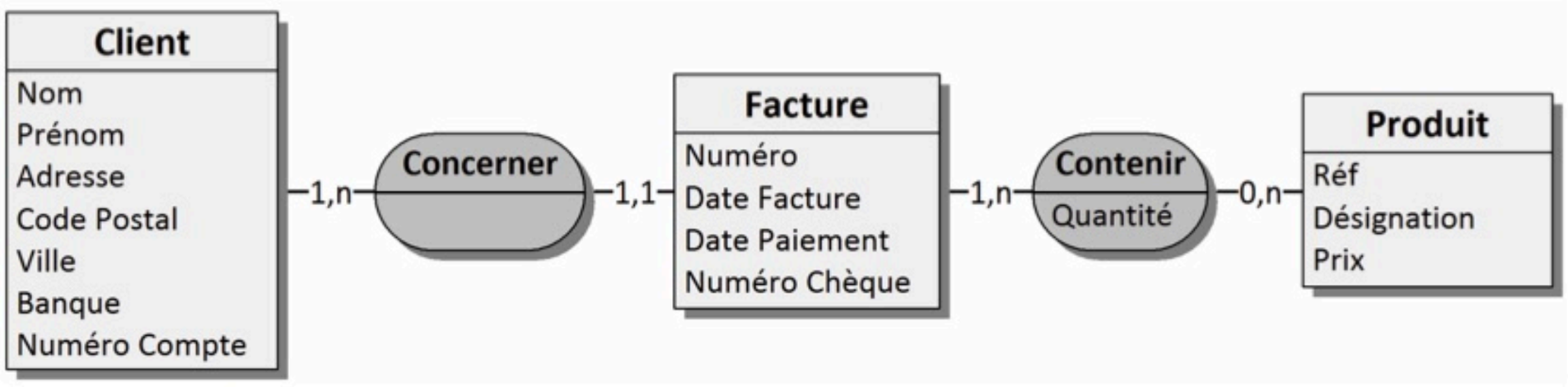
Chaque FACTURE concerne **1** à **1** CLIENT

Concernant l'association « Contenir » entre « Facture » et « Produit », nous avons :

Chaque FACTURE contient **1** à **n** PRODUIT
et
Chaque PRODUIT est contenu dans **0** à **n** FACTURE

En effet, une facture doit contenir au moins un produit ; par contre, un produit peut ne jamais être vendu et n'apparaitre alors dans aucune facture.

Le modèle définitif est donc le suivant :



Les cardinalités les plus fréquemment utilisées sont résumées dans le tableau suivant :

	Simple	Multiple
Obligatoire	**1,1**	**1,n**
Facultative	**0,1**	**0,n**

Pour réaliser une association avec ***Looping***, il suffit de choisir l'outil « Lien » et de relier les deux classes d'entités concernées.

Le modèle est maintenant complet dans sa version conceptuelle. Nous verrons dans la troisième partie de cet ouvrage que certains éléments techniques devront être rajoutés pour permettre la constitution de la base de données correspondante.

Le modèle conceptuel de données, régulièrement désigné avec le sigle **MCD**, représente une étape particulièrement importante ; en effet, il conditionne tout le processus d'informatisation d'un système d'information.

© Patrick Bergougnoux

Choix des verbes

Utiliser des verbes pour désigner les associations est très utile pour bien faire ressortir la sémantique du modèle et déterminer les cardinalités.

Cependant, certains choix de verbes peuvent provoquer des erreurs ; en effet, les verbes d'action peuvent limiter l'association dans le temps et induire le concepteur en erreur au moment de définir les cardinalités.

Par exemple, si le verbe « Envoyer » est utilisé au lieu de « Concerner », on constate déjà que la conjugaison ne fonctionne que dans un sens : en effet une facture est envoyée à un client, mais un client … on est coincé… impossible de conjuguer « Envoyer » dans ce sens.

Au-delà de cet inconvénient pratique, le plus grave est que ce choix peut amener le concepteur à penser que, à un moment donné, on envoie **une** facture à un client, et en déduire donc une cardinalité **1,1** au lieu de **1,n**.

Pire : avec un tel verbe, on peut être tenté de mettre la date de facture dans l'association, vu qu'elle caractérise l'envoi…

Nous verrons dans le chapitre suivant que certaines de ces erreurs pourront être évitées en respectant les règles de cohérence des modèles. Cependant, le plus sûr est d'utiliser systématiquement des verbes d'état tels que « Concerner », « Correspondre », « Être », …

Pour désigner les associations, il faut éviter les verbes d'action et privilégier les verbes d'état.

De plus, pour s'assurer de ne pas limiter l'association dans le temps, une astuce efficace consiste à rajouter, par exemple, « En 50 ans » devant la phrase qui détermine les cardinalités. Nous dirons donc :

En 50 ans, chaque CLIENT est concerné par **1** à **n** FACTURE

En 50 ans, chaque PRODUIT est contenu dans **0** à **n** FACTURE

Beaucoup d'erreurs peuvent ainsi être évitées. Mais attention de ne pas tomber dans l'excès inverse ; en effet, il ne faut pas négliger le mot « **chaque** » et ne pas le remplacer par un pluriel qui pourrait fausser les cardinalités **1,1** :

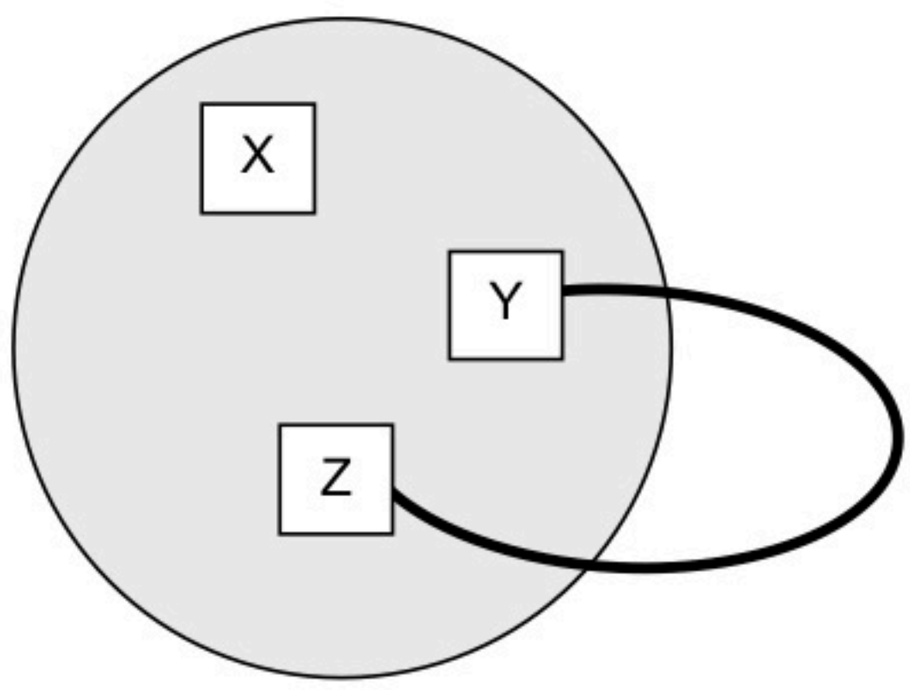
En 50 ans, **chaque** FACTURE concerne **1** à **1** CLIENT

La définition des cardinalités est souvent la bête noire des débutants en modélisation conceptuelle de données. Le respect des recommandations que nous venons de voir doit rendre élémentaire cette étape de la conception.

Associations réflexives

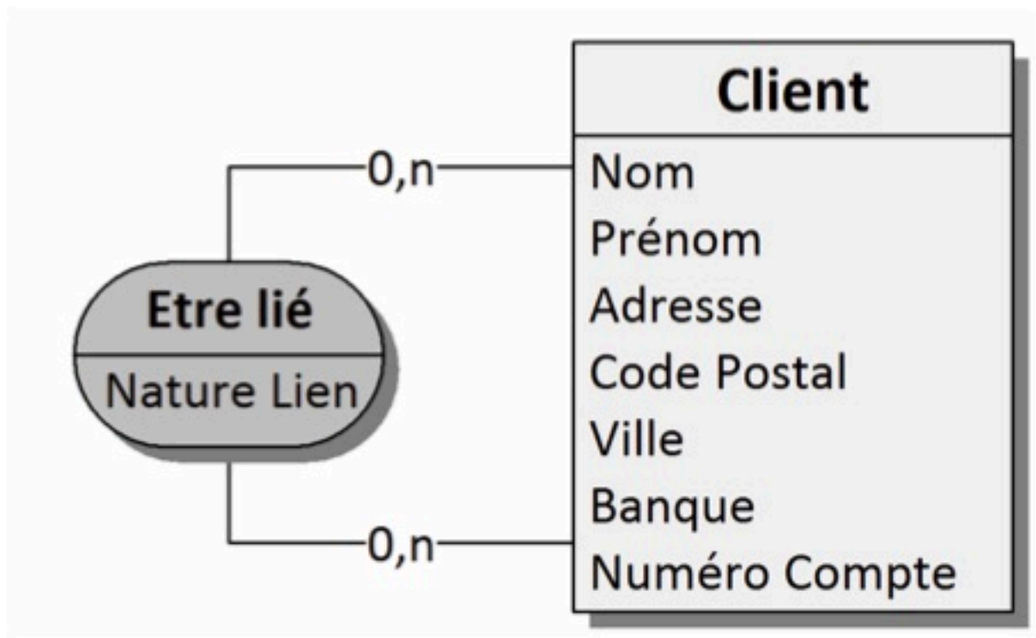
Liens entre entités d'une même classe

Imaginons que nous souhaitions connaitre la nature des éventuels liens commerciaux entre certains clients. Cette situation peut être schématisée de la façon suivante :



Dans l'exemple ci-dessus, les clients Y et Z ont donc des liens commerciaux.

La représentation dans le modèle conceptuel de données reste dans la même logique que précédemment :



La nouvelle rubrique « Nature Lien » ne caractérise pas la classe d'entités « Client » : en effet, un même client peut avoir plusieurs natures de liens différentes en fonction des autres clients auxquels il est lié.

Par contre, pour deux clients associés (liés), il existe bien une seule nature de lien : la rubrique « Nature Lien » doit donc bien être affectée à l'association « Etre lié ».

© Patrick Bergougnoux

Association réflexive symétrique

Dans l'exemple que nous venons de traiter, les cardinalités sont les mêmes sur les deux pattes. Il s'agit dans ce cas d'une association réflexive **symétrique** : en effet, si le client Y est lié au client Z, le client Z est nécessairement lié de la même manière au client Y.

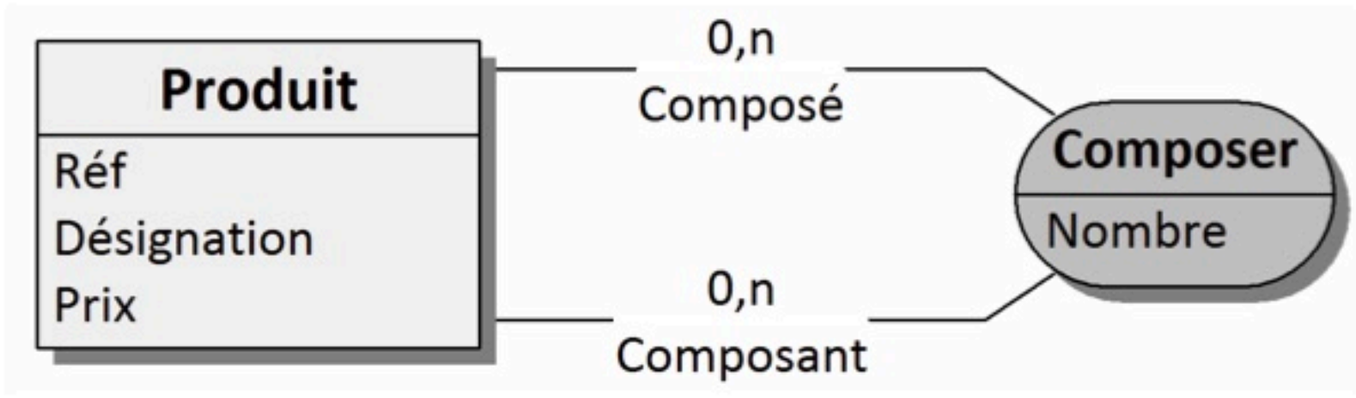
Les cardinalités sont alors déterminées de manière identique dans les deux sens :

Chaque CLIENT est lié avec **0** à **n** CLIENT

Association réflexive asymétrique

Toutes les associations réflexives ne sont pas forcément symétriques. Il est parfois nécessaire de différencier le sens de l'association par l'attribution de rôles à chacune des pattes. On parle alors d'association réflexive **asymétrique**.

Voici un exemple décrivant une nomenclature, certains produits étant composés d'un certain nombre d'autres produits de la même classe :



La définition des rôles permet de mettre en évidence cette asymétrie, ce qui n'est pas utile pour les associations réflexives symétriques.

Par ailleurs, le fait d'avoir les mêmes cardinalités de part et d'autre rend cette association **non hiérarchique** ; en effet, dans notre exemple, tous les produits peuvent être composants et composés avec les mêmes conditions de composition liées aux deux cardinalités **0,n** : il n'y a donc aucune hiérarchie entre eux.

Les cardinalités se lisent de la façon suivante, le rôle précédé du verbe « être » venant se substituer au verbe de l'association afin de bien distinguer le rôle de chaque patte :

Chaque PRODUIT est le **composé** de **0** à **n** PRODUIT (composant)

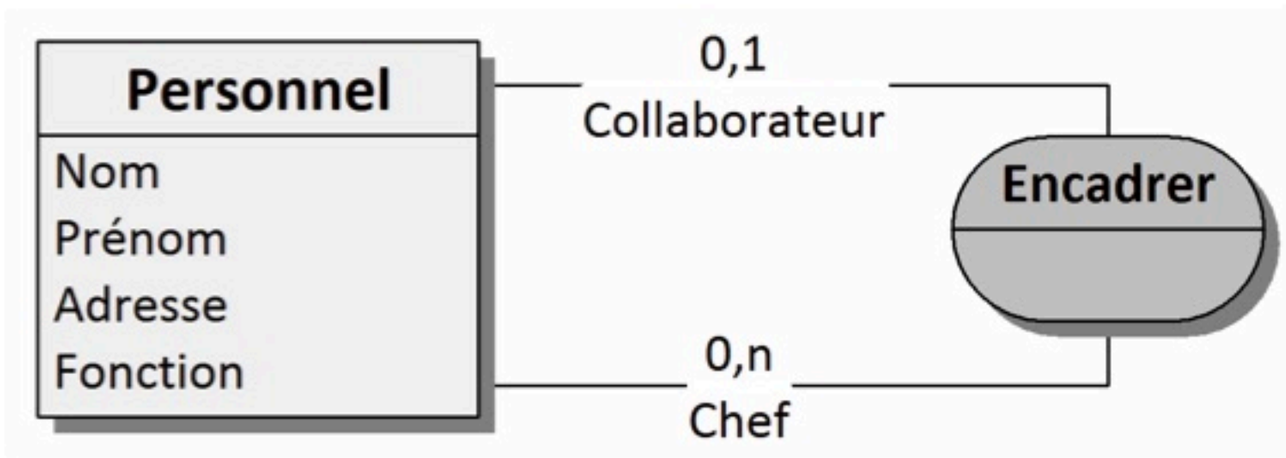
Chaque PRODUIT est le **composant** de **0** à **n** PRODUIT (composé)

De plus, la rubrique « Nombre » indique combien d'exemplaires d'un composant donné sont nécessaires à la composition d'un composé donné.

Association réflexive hiérarchique

En cas de cardinalité simple d'un côté et multiple de l'autre, on parle d'association réflexive **hiérarchique**.

C'est le cas, par exemple, d'une entreprise dans laquelle les salariés (sauf le patron) sont des collaborateurs encadrés par un seul chef, supérieur hiérarchique direct, chaque chef encadrant au moins un salarié (bien évidemment, certains salariés ne sont chefs de personne).



Les associations réflexives hiérarchiques sont forcément asymétriques et doivent donc utiliser les rôles. Comme précédemment, les cardinalités se lisent en remplaçant le verbe de l'association par le rôle précédé du verbe « être » :

Chaque PERSONNEL est le **chef** de **0** à **n** PERSONNEL (collaborateur)

Chaque PERSONNEL est le **collaborateur** de **0** à **1** PERSONNEL (chef)

Les associations réflexives restent relativement rares dans les modèles conceptuels de données, et il faut n'en faire usage que lorsque cela est strictement nécessaire.

Ce qu'il faut retenir

- Les associations entre classes d'entités traduisent, au sein du modèle conceptuel de données, la sémantique du système d'information.
- Les cardinalités dénombrent les liens qui peuvent exister entre les différentes entités des classes associées.
- Les associations réflexives permettent de lier des entités au sein d'une même classe.

© Patrick Bergougnoux

Chapitre 7

Règles de cohérence des modèles

Lors de la répartition des données en classes d'entités, nous avons défini une règle essentielle permettant d'éviter d'affecter une rubrique au mauvais endroit.

Comme nous l'avons précisé, cette règle ne garantit pas à elle seule que le modèle soit parfait et optimisé. Le respect des cinq règles détaillées dans ce chapitre apporte cette garantie.

Règle 1

Une et une seule affectation par rubrique

Nous avons vu au chapitre 5 de cet ouvrage que toutes les rubriques du dictionnaire des données doivent être affectées à une classe d'entités ou à une association, et à une seule.

Il faut donc s'assurer qu'une rubrique ne soit ni oubliée, ni affectée plusieurs fois. Rappelons qu'ont été préalablement exclus les paramètres et les rubriques calculables.

Énoncé de la règle

Chaque rubrique du dictionnaire des données doit être présente une et une seule fois dans un modèle conceptuel de données.

Règle 2

Une seule valeur par rubrique pour une entité ou une association

Cette deuxième règle a également déjà été traitée au chapitre 5 : c'est la principale règle à appliquer systématiquement à chaque affection d'une rubrique à une classe d'entités ou à une association.

Même si cette règle ne garantit pas la parfaite qualité du modèle, elle permet d'éviter les plus grosses erreurs. La règle suivante va venir la compléter de manière exhaustive pour produire des modèles irréprochables.

Énoncé de la règle

Une rubrique affectée à une classe d'entités ou à une association ne peut avoir qu'une seule valeur pour chaque entité de la classe ou chaque association d'entités.

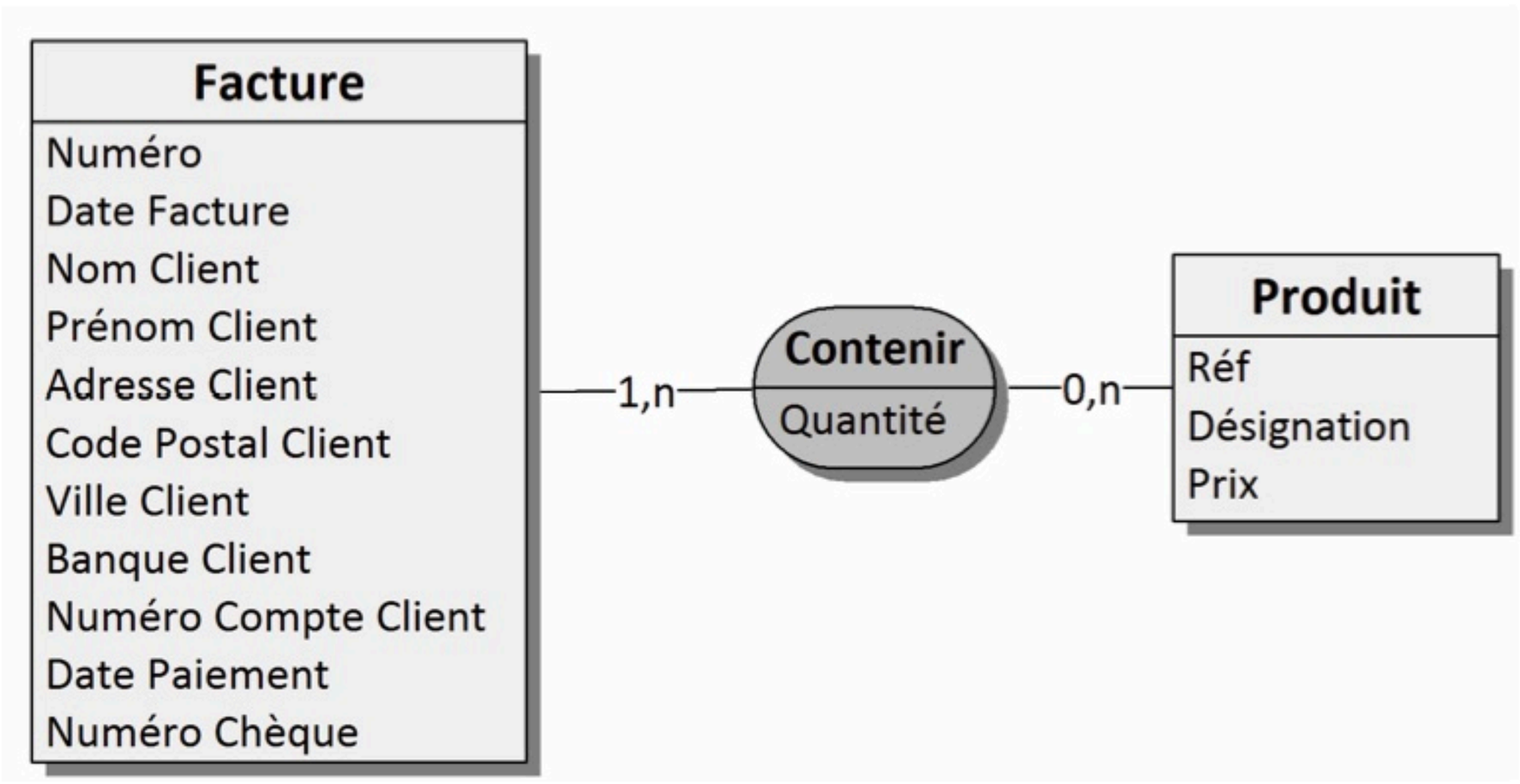
Règle 3

Même valeur de rubrique pour plusieurs entités d'une classe

C'est sans nul doute la règle la plus complexe à appliquer dans la mesure où elle revêt une part non négligeable d'interprétation et la nécessité d'arbitrer de la part du concepteur.

© Patrick Bergougnoux

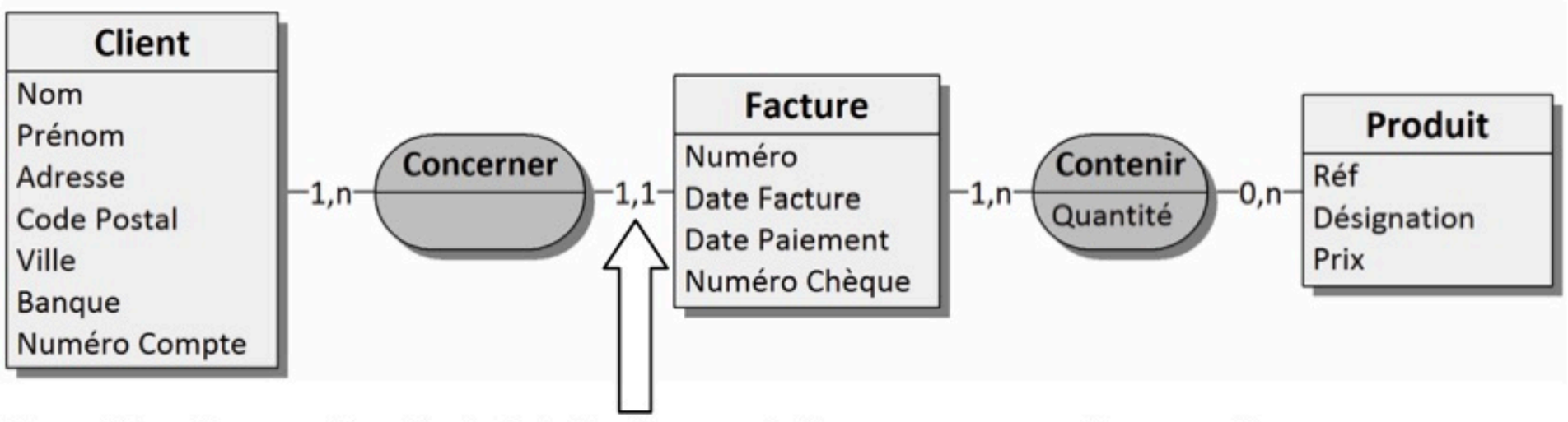
Pour mettre en évidence la problématique, reprenons l'exemple de la facturation Beaux Rivages avec une proposition de modèle différente de la solution présentée jusqu'à présent :



Si nous vérifions les deux premières règles, nous constatons qu'elles sont parfaitement respectées :

- Pour une facture donnée, il n'y a qu'un seul nom de client.
- Pour une facture donnée, il n'y a qu'un seul prénom de client.
- Pour une facture donnée, il n'y a qu'une seule adresse de client.
- …

Bref, toutes les rubriques qui étaient affectées à la classe d'entités « Client » n'ont qu'une valeur pour une facture donnée et peuvent donc, d'après la règle 2, être affectées à la classe « Facture ». Il est d'ailleurs intéressant de noter que notre premier modèle exprimait déjà cela ; reprenons ce modèle :



En effet, la cardinalité **1,1** indique clairement que chaque facture concerne **1** et **1 seul** client.

Mais alors, quel est le problème de ce modèle ?

Pour bien comprendre le problème, il suffit d'imaginer un client auquel Beaux Rivages adresse une centaine de factures : à chaque nouvelle facture, toutes les données relatives au client devront être reprises et restockées, avec le risque qu'elles ne soient pas forcément identiques à chaque fois !

Au-delà de la perte de place dans la future base de données, c'est la cohérence du système qui sera mise à mal par la redondance des informations du client.

De plus, si les coordonnées du client changent et que l'on souhaite qu'elles soient mises à jour dans tout le système d'information, ce sont des centaines de modifications qui devront être faites dans la base de données.

C'est le parfait exemple d'un modèle qui n'est pas fondamentalement faux, mais qui posera un réel problème lors de la mise en œuvre de la base de données correspondante.

Notons néanmoins que le choix de stocker les données du client dans la facture peut être une volonté pour des raisons d'historisation, afin que les anciennes données du client soient conservées en cas de modification. Mais le concepteur devra quand même prévoir une classe « Client » contenant les dernières informations du client pour la saisie des nouvelles factures.

Nous ne traiterons pas pour l'instant cette problématique d'historisation et conserverons donc notre premier modèle.

Il faut alors imaginer une règle permettant d'éviter les redondances dans le système d'information et le modèle conceptuel de données correspondant.

Premier énoncé de la règle et ses limites

Si une rubrique possède la même valeur pour plusieurs entités d'une même classe, il faut que cette rubrique fasse l'objet d'une classe à part.

Mais, pourquoi parle-t-on d'un « premier » énoncé de la règle et de ses limites ?

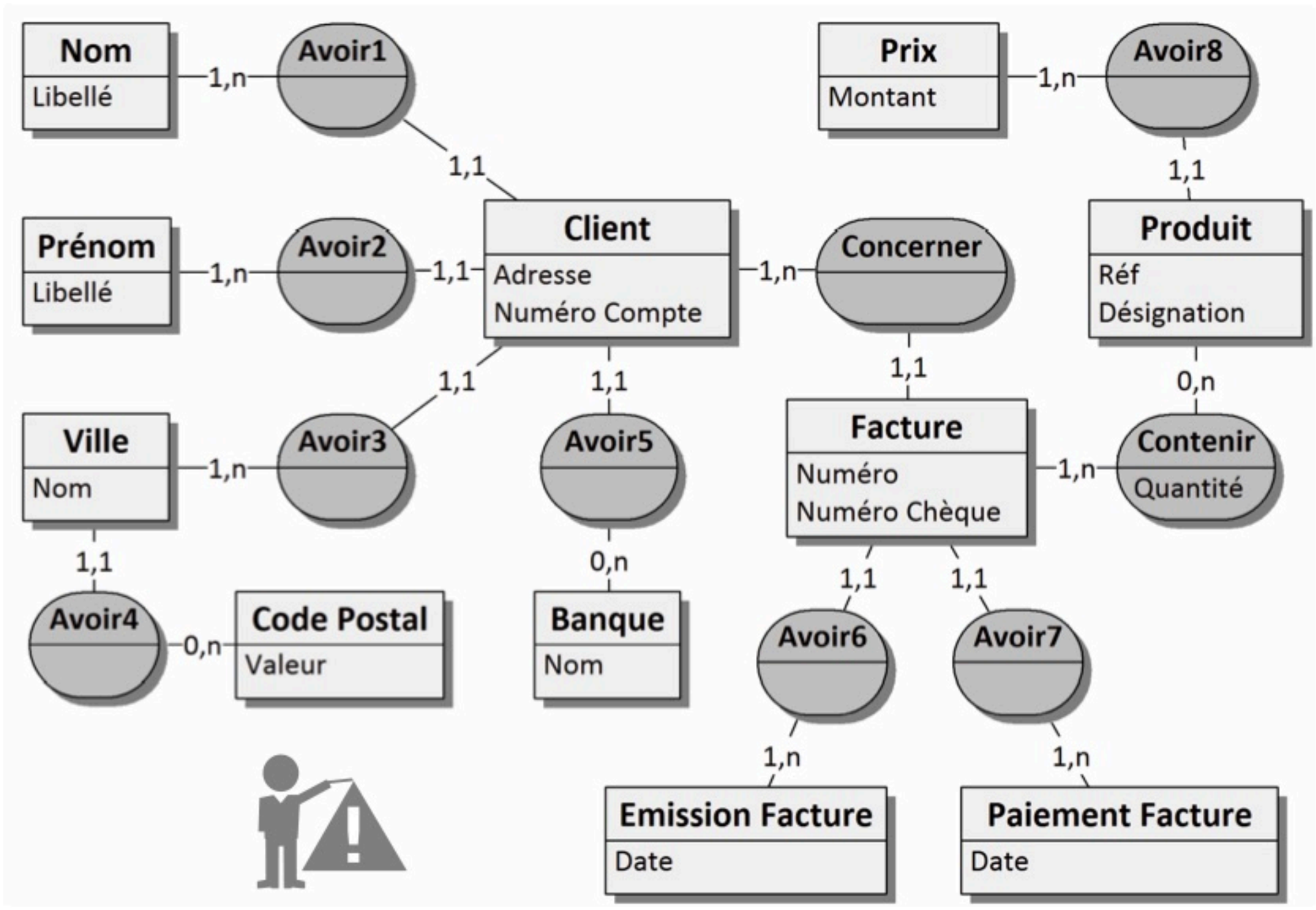
Tout simplement parce qu'une application trop stricte de cette règle nous conduirait à des aberrations !

© Patrick Bergougnoux

En effet :

- Plusieurs clients peuvent avoir le même nom
  ➔ il faut donc créer une classe « Nom » à part.
- Plusieurs clients peuvent avoir le même prénom
  ➔ il faut donc créer une classe « Prénom » à part.
- Plusieurs clients peuvent habiter dans la même ville
  ➔ il faut donc créer une classe « Ville » à part.
- Plusieurs villes peuvent posséder le même code postal
  ➔ il faut donc créer une classe « Code Postal » à part.
- Plusieurs clients peuvent avoir la même banque
  ➔ il faut donc créer une classe « Banque » à part.
- Plusieurs factures peuvent avoir la même date d'émission
  ➔ il faut donc créer une classe « Emission Facture » à part.
- Plusieurs factures peuvent avoir la même date de paiement
  ➔ il faut donc créer une classe « Paiement Facture » à part.
- Plusieurs produits peuvent avoir le même prix
  ➔ il faut donc créer une classe « Prix » à part.

Par conséquent, si nous respections pleinement cette règle 3 avec son énoncé actuel, voici à quoi ressemblerait le modèle :



Il est clair qu'il faut éviter cela ! Mais comment ?

En fait, c'est là qu'un arbitrage du concepteur est nécessaire ; en théorie, la question qu'il faut se poser est la suivante :

La création de ces nouvelles classes d'entités a-t-elle un réel intérêt pour le système d'information ?

Il n'est pas forcément facile de répondre à cette question… et le choix reste subjectif. Il est alors possible d'appliquer un critère formel qui peut apporter une réponse nette à ce dilemme avec un complément d'énoncé pour la règle 3.

Nouvel énoncé de la règle

Si une rubrique possède la même valeur pour plusieurs entités d'une même classe, il faut **envisager** que cette rubrique fasse l'objet d'une classe à part : cette règle est appliquée **si et seulement si plusieurs rubriques** caractérisent la nouvelle classe en question.

En effet, dès lors qu'une classe d'entités possède au moins deux rubriques, nous pouvons considérer que son existence a du sens pour le système d'information.

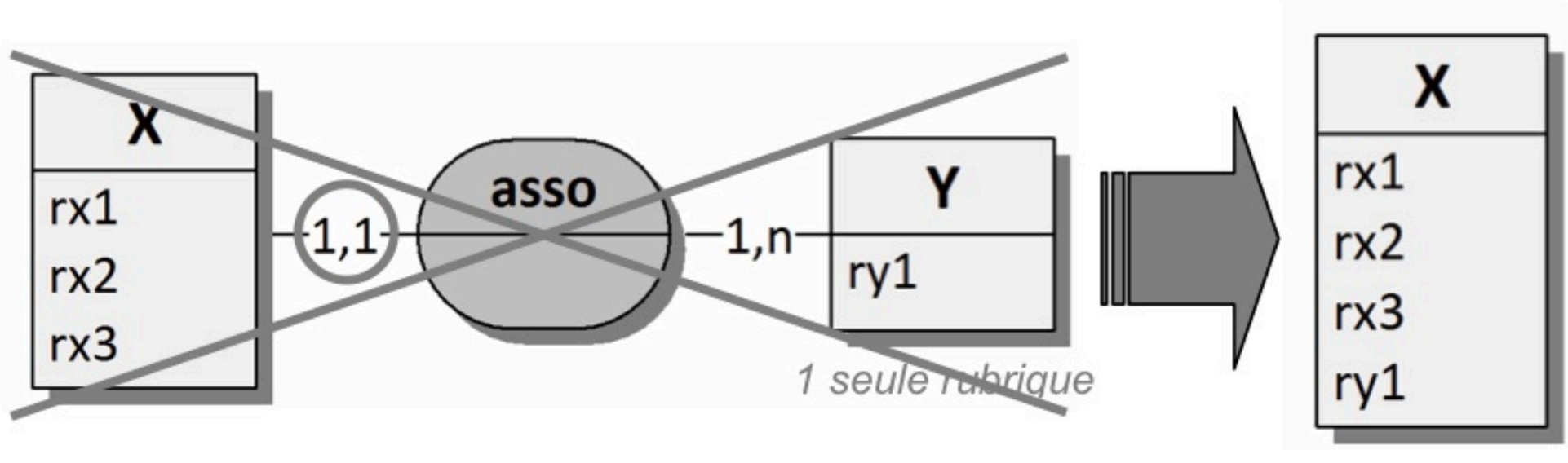
De cette manière, toutes les petites classes « satellites » sans intérêt ne seront plus concernées et ne viendront pas alourdir inutilement le modèle.

Conséquence : règle 3bis

Cette réflexion sur les petites classes satellites ne possédant qu'une seule rubrique nous amène à énoncer une règle directement dérivée de la règle 3 :

Si la classe d'entités Y ne possède qu'une seule rubrique et n'est associée qu'à la seule classe d'entités X, en cas de cardinalité simple côté classe X, la rubrique en question peut être intégrée à la classe X, et la classe Y peut alors être supprimée.

Cette règle 3 bis peut être résumée plus simplement de la façon suivante :



© Patrick Bergougnoux

Tout cela ne signifie pas qu'une classe d'entités avec une seule rubrique est strictement interdite, même si la règle 3bis est vérifiée : le concepteur peut très bien décider de maintenir ce type de classe s'il considère qu'elle a de l'intérêt pour le système d'information.

Règle 4

Jamais de rubrique dans une association avec patte simple

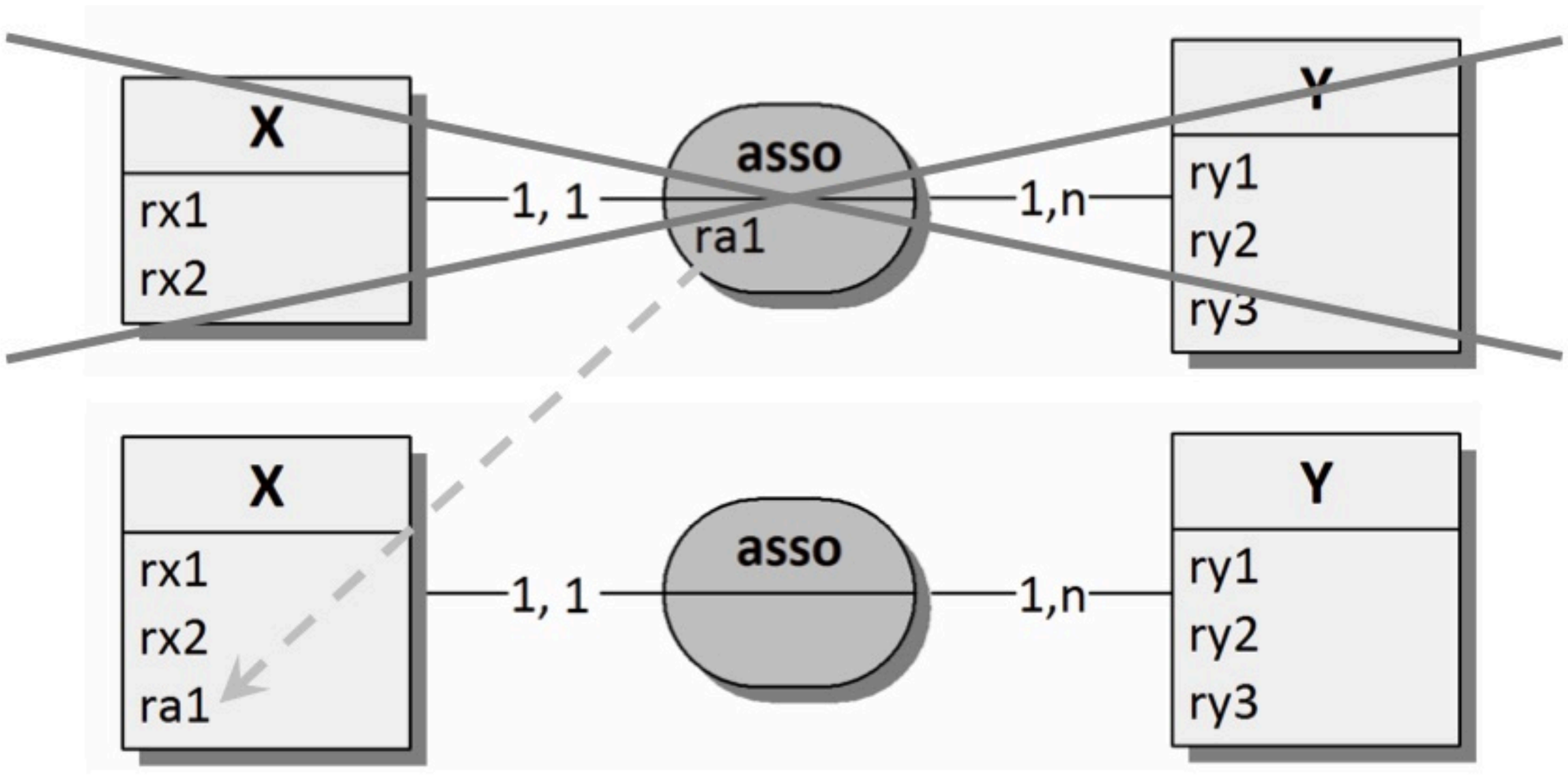
Une mauvaise interprétation des cardinalités, souvent liée à un mauvais choix de verbe, peut conduire à tort à l'affectation d'une rubrique dans une association. Tout d'abord, rappelons qu'il est impératif de vérifier au préalable qu'une telle rubrique ne peut pas être affectée à une des classes associées.

Au-delà de cette vérification toujours nécessaire, il existe une situation bien précise qui interdit l'affectation d'une rubrique dans une association.

Énoncé de la règle et son explication

Une association non multiple de part et d'autre ne peut en aucun cas être porteuse de rubriques.

Si cette situation se présente, **il faut avant tout s'assurer que les cardinalités ont été correctement définies** ; si c'est le cas, les rubriques placées à tort dans l'association doivent alors être déplacées dans la classe d'entités côté patte simple, comme le montre le schéma ci-dessous :

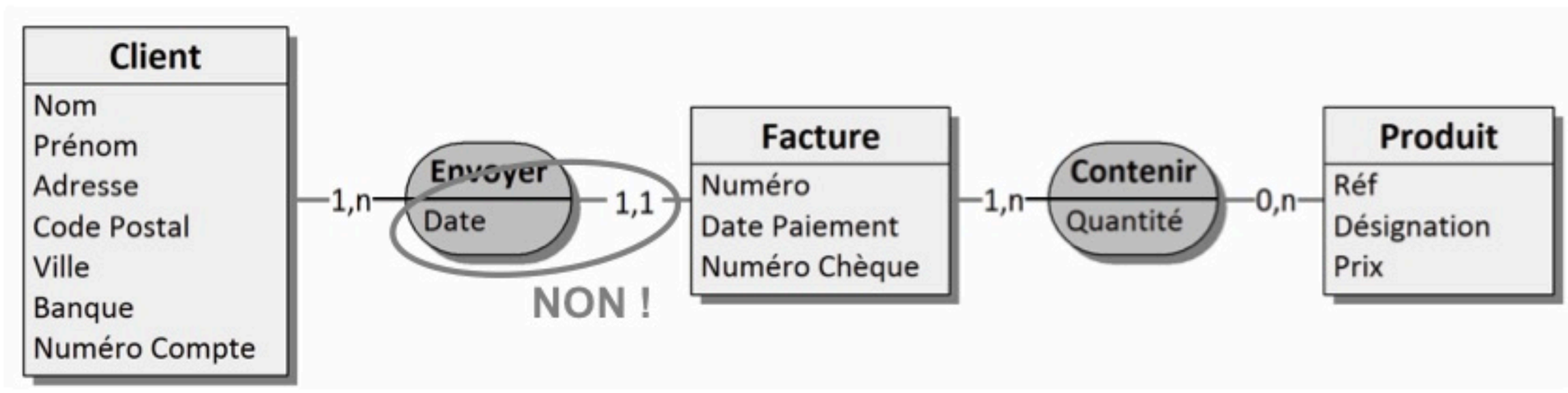


Pourquoi une telle interdiction ?

L'application de la règle est simple ; son explication l'est moins !

Même s'il est possible d'appliquer la règle sans en comprendre le bien-fondé, il est tout de même préférable de la mettre en œuvre en connaissance de cause.

Pour cela, reprenons notre exemple de facturation Beaux Rivages et provoquons une erreur qui transgresse cette règle 4 :



Si l'on n'y prend pas garde, ce modèle ne choque pas ; en effet, la facture est bien envoyée au client à une date donnée : la date caractérise alors l'envoi de la facture au client, et la placer entre les deux classes d'entités peut paraitre logique…

Sauf que, dans le cas présent, nous avons bafoué les règles fondamentales de construction du modèle :

- Mauvais choix de verbe, avec un verbe d'action qui contribue à la mauvaise interprétation de l'association ; en effet, une action se déroule à un moment précis, ce qui peut inciter à lui attribuer une date.
- Ne jamais affecter une rubrique à une association avant de s'être assuré qu'elle ne va pas dans l'une ou l'autre des classes d'entités associées.
- Une facture possède-t-telle une et une seule date ? **OUI !** Alors, il ne faut pas chercher plus loin : il faut affecter cette rubrique « Date » à la classe d'entités « Facture ».

Si nous étudions de près la sémantique du modèle conceptuel de données, tout est écrit. Si la date est placée dans l'association, cela signifie qu'il existe une date pour une facture donnée **ET** un client donné. Or, la cardinalité **1,1** nous indique qu'une facture concerne **1** et **1 seul** client : le client dépend donc directement de la facture.

Il est erroné d'affirmer que la date dépend du client **ET** de la facture sachant que lorsque la facture est connue, il n'y a qu'un seul client possible. La date ne dépend donc que de la facture.

© Patrick Bergougnoux

Il est également possible de présenter la problématique en mettant en évidence les dépendances (A ⟶ B se lit « A donne B »).

Si la date est placée dans l'association entre « Facture » et « Client », on a :

- Facture ⟶ Date
- Client ⟶ Date

La cardinalité **1,1** nous indique que :

- Facture ⟶ Client

Par conséquent :

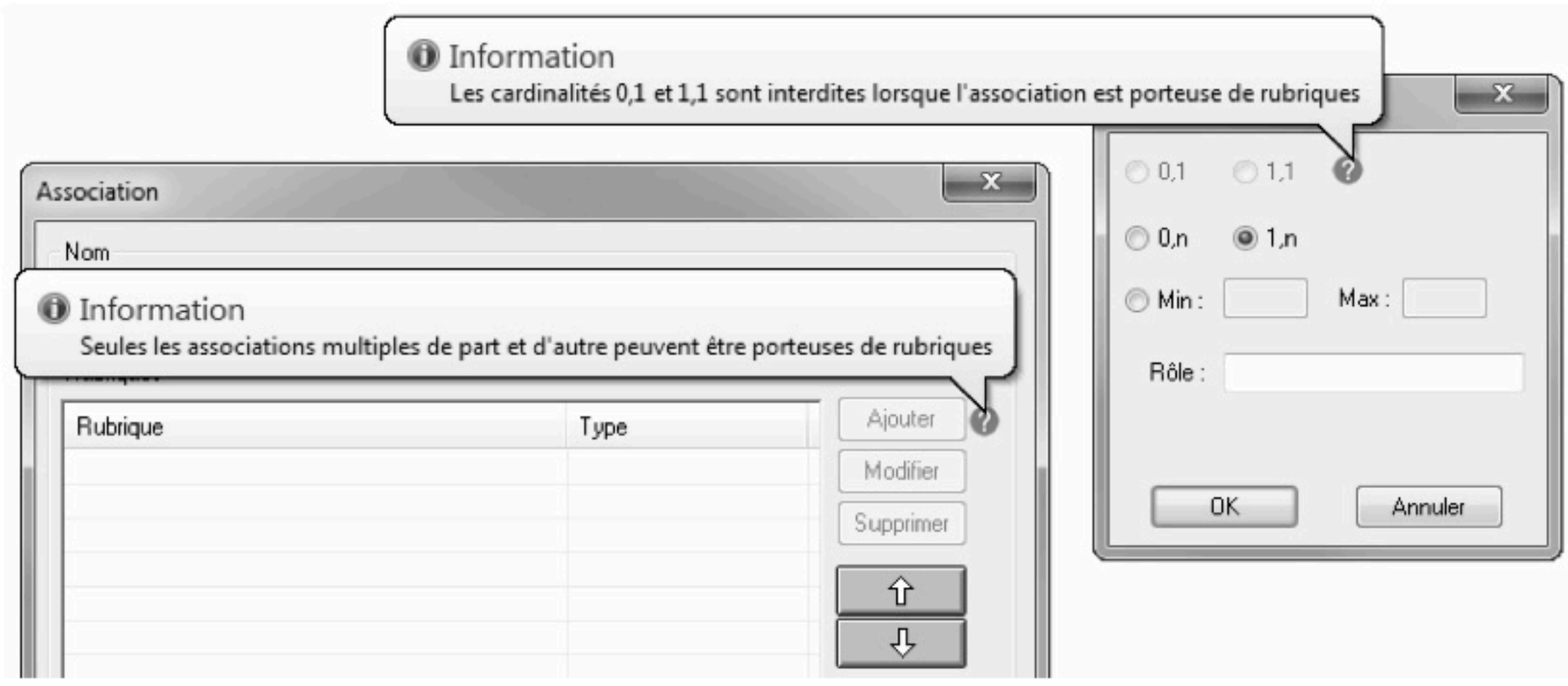
- Facture ⟶ Date
- Facture ⟶ Client ⟶ Date

La date ne dépend donc bien que de la facture.

Afin de respecter la règle 4, vérifions avant tout qu'il n'y a pas d'erreur dans la cardinalité **1,1** : une même facture ne pouvant pas concerner plusieurs clients, cette cardinalité simple est correcte.

La date peut alors être déplacée dans la classe d'entités côté patte simple, c'est à dire dans la classe « Facture ». Nous revenons ainsi à un modèle cohérent, tel qu'il avait été défini au chapitre précédent.

Looping prend en charge cette règle 4 en interdisant l'utilisation des cardinalités simples en cas de présence de rubriques dans l'association, ainsi que la saisie de rubriques dans une association ne possédant pas des cardinalités multiples de part et d'autre.



L'application de cette règle pour la cardinalité **0,1** peut faire débat chez certains puristes des bases de données. Néanmoins, nous justifierons ce choix un peu plus loin dans cet ouvrage.

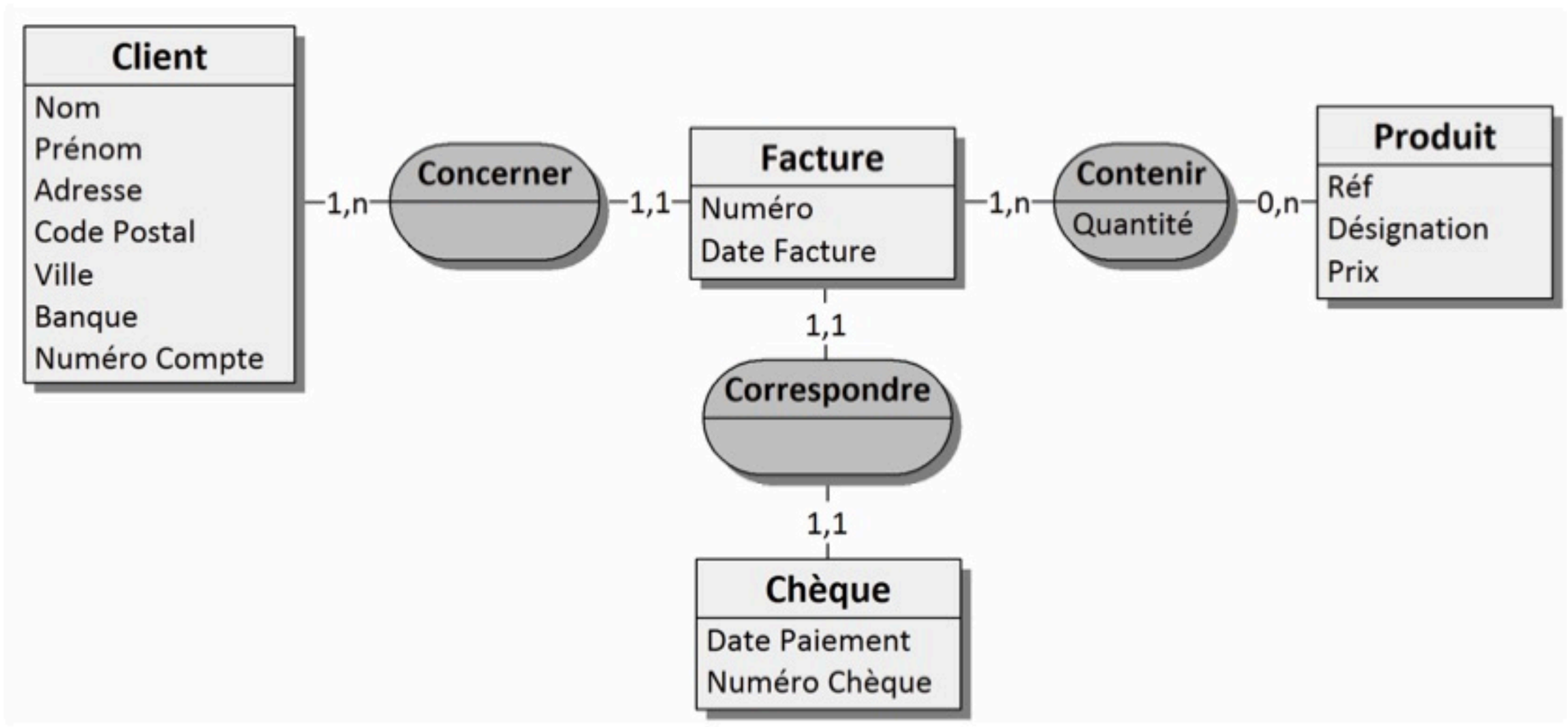
Règle 5

Jamais deux cardinalités 1,1 pour une même association

Lorsque le dictionnaire des données est achevé, le concepteur doit définir les classes d'entités qui semblent ressortir de manière évidente de l'analyse.

Comme nous l'avons déjà vu, il se peut que certaines classes soient oubliées ou/et que d'autres soient créées à tort.

Par exemple, lors du cours où je traite la facturation Beaux Rivages, la réflexion des étudiants les amène souvent à proposer une classe d'entités « Chèque ». Une fois que le processus de construction du modèle conceptuel de données est achevé, voici le résultat obtenu :



Ce modèle n'est pas faux et il respecte l'ensemble des règles que nous avons vues jusqu'à présent. En effet, d'après le compte-rendu d'interview qui présente cette facturation simplifiée, le paiement d'une facture est effectué par un seul chèque et un chèque ne paie qu'une seule facture : c'est bien ce qui est exprimé dans le modèle avec les cardinalités **1,1** de part et d'autre de l'association « Correspondre » : le modèle représente donc bien le système d'information.

Mais alors, quel est le problème ?

En fait, il s'agit d'un problème d'**optimisation** : en modélisation, il est souvent préférable de privilégier la simplicité de manière à ce que la future base de données soit plus aisée à manipuler.

En résumé, moins il y a de classes dans le modèle, plus simple sera la structure de la base de données, et son utilisation sera plus optimale.

© Patrick Bergougnoux

Dans l'exemple que nous venons de voir, il y a en effet plus simple :

- Pour une facture donnée, il n'y a qu'une seule date de paiement et qu'un seul numéro de chèque.
- Pour un chèque donné, il n'y a qu'un seul numéro de facture et qu'une seule date de facture.

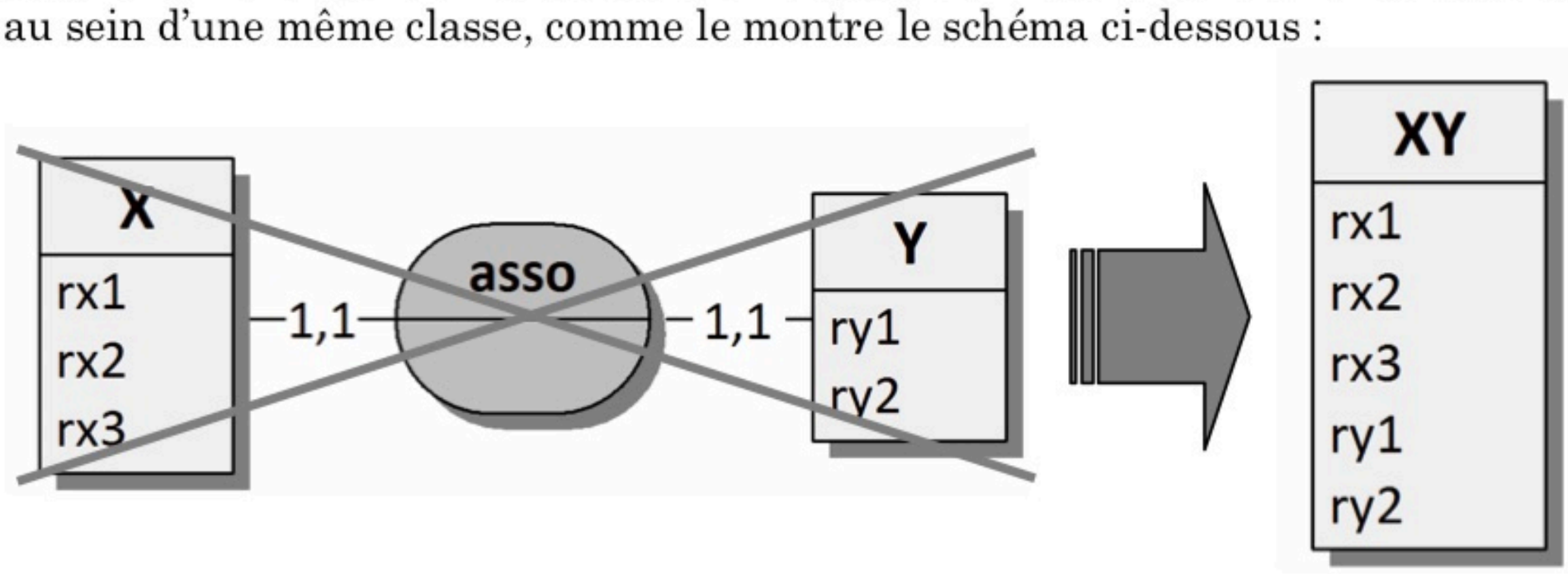
Chaque rubrique d'une classe peut alors être affectée à l'autre classe ; par conséquent, toutes ces rubriques peuvent appartenir à la même classe, et ce sont les deux cardinalités **1,1** de part et d'autre de l'association qui nous dictent cela.

Au-delà des problèmes liés à leur interdépendance, la fusion des deux classes s'impose donc afin d'optimiser le modèle.

Énoncé de la règle

Deux classes d'entités ne peuvent pas être associées par une association possédant des cardinalités **1,1** de part et d'autre.

Comme pour la règle 4, si cette situation se présente, **il faut avant tout s'assurer que les cardinalités ont été correctement définies** ; si c'est le cas, l'ensemble des rubriques des deux classes d'entités doivent être regroupées au sein d'une même classe, comme le montre le schéma ci-dessous :



Les cas où ces deux cardinalités **1,1** trouvent une réelle justification dans l'analyse du système d'information sont assez rares : il s'agit très souvent d'erreurs de cardinalités liées à un mauvais choix de verbe et/ou à une limitation de l'association dans le temps.

Looping prend en charge cette règle 5 en interdisant l'utilisation des cardinalités **1,1** de part et d'autre d'une association.

Ce qu'il faut retenir

- **Règle 1** : Chaque rubrique du dictionnaire des données doit être présente une et une seule fois dans un modèle conceptuel de données.
- **Règle 2** : Une rubrique affectée à une classe d'entités ou à une association ne peut avoir qu'une seule valeur pour chaque entité de la classe ou chaque association d'entités.
- **Règle 3** : Si une rubrique possède la même valeur pour plusieurs entités d'une même classe, il faut envisager que cette rubrique fasse l'objet d'une classe à part : cette règle est appliquée si et seulement si plusieurs rubriques caractérisent la nouvelle classe en question.
- **Règle 3bis** : Si la classe d'entités Y ne possède qu'une seule rubrique et n'est associée qu'à la seule classe d'entités X, en cas de cardinalité simple côté classe X, la rubrique en question peut être intégrée à la classe X, et la classe Y peut alors être supprimée.
- **Règle 4** : Une association non multiple de part et d'autre ne peut en aucun cas être porteuse de rubriques.
- **Règle 5** : Deux classes d'entités ne peuvent pas être associées par une association possédant des cardinalités **1,1** de part et d'autre.

© *Patrick Bergougnoux*

Chapitre 8

Exercices de modélisation

Dans ce chapitre, les trois exercices initiés au chapitre 4 sont repris pour être finalisés jusqu'à la réalisation du modèle conceptuel de données.

Cinq nouveaux exercices complets sont ensuite proposés et traités en respectant l'ensemble de la démarche étudiée jusqu'à présent.

Une fois que le dictionnaire des données est établi, les rubriques sont réparties entre les classes d'entités apparues de manière rapide et évidente à la lecture du dictionnaire. Chaque rubrique restante est étudiée de manière plus approfondie pour en définir l'affectation au sein d'une classe d'entités existante, d'une association, ou d'une nouvelle classe créée pour l'occasion.

Le modèle est alors complété avec les associations et leurs cardinalités en respectant la sémantique du système d'information. Il est très important de passer ensuite à la phase de vérification du modèle avec le respect des règles de cohérence que nous venons de voir au chapitre précédent.

Chaque système d'information met en évidence des problématiques spécifiques rappelées en fin de chaque exercice.

Des erreurs volontaires sont commises (et corrigées ensuite !) sur certaines solutions afin de bien mettre en évidence le respect des différentes règles. Ces erreurs sont celles que j'ai le plus souvent constatées chez les étudiants en plus de vingt ans d'enseignement.

Gestion des achats d'une administration

Rappel du dictionnaire des données

- N° commande
- Date commande
- N° engagement
- Nom fournisseur
- Adresse fournisseur
- Code postal fournisseur
- Ville fournisseur
- Référence produit
- Désignation produit
- Quantité produit
- Prix unitaire TTC produit
- N° facture
- Date facture
- Date lettre refus
- N° mandat
- Date mandat
- IBAN fournisseur

Répartition des données dans les classes d'entités

Première détermination des classes d'entités

Voici les classes d'entités qui apparaissent immédiatement à la lecture du dictionnaire des données :

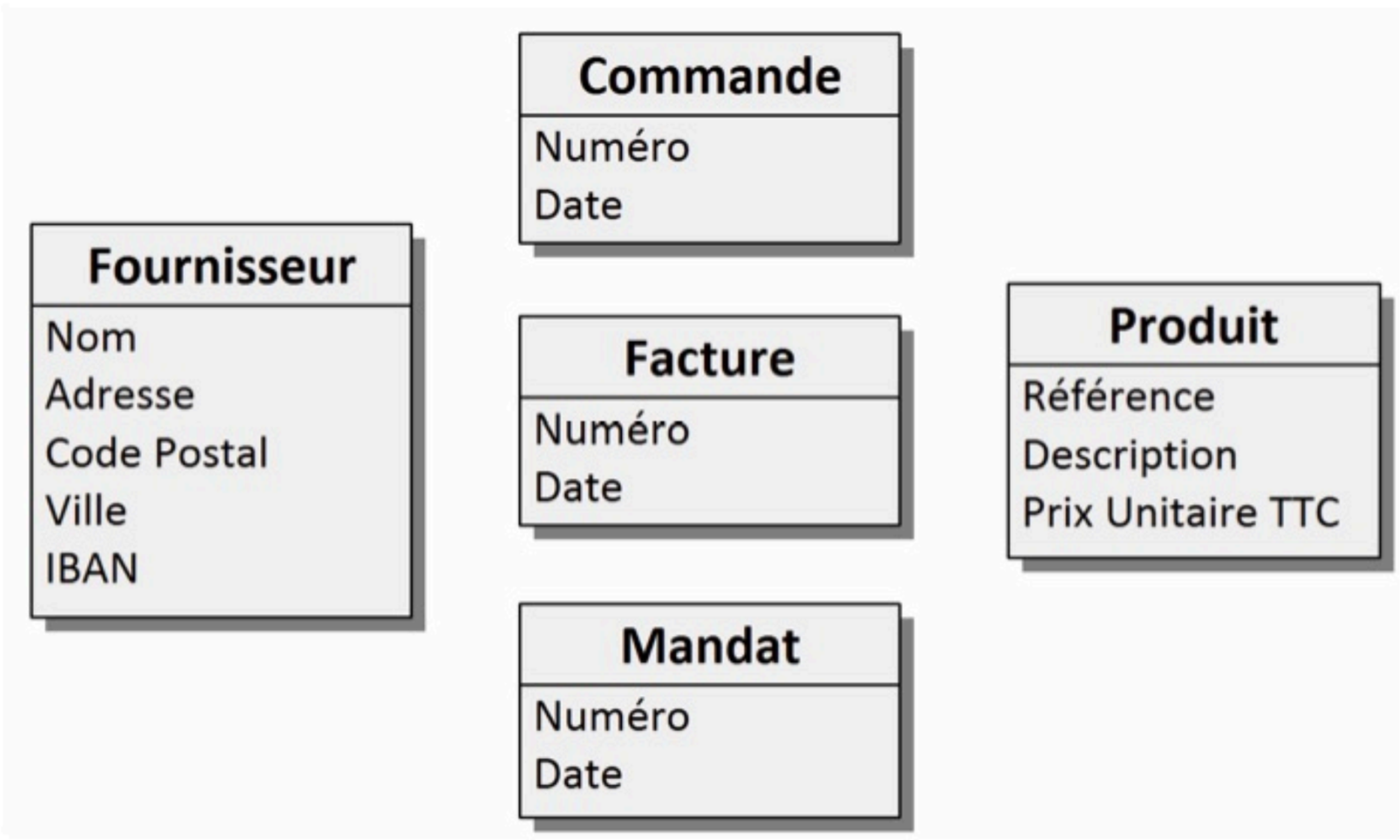
- Commande
- Fournisseur
- Produit
- Facture
- Mandat

Cette première estimation ne correspond pas forcément à la réalité, mais cela représente un point de départ de la modélisation, sachant que les règles que nous venons d'étudier nous ramèneront forcément à la bonne solution.

© Patrick Bergougnoux

Affectation des rubriques les plus évidentes

Le plus important maintenant, quelles que soient les classes en présence, est de respecter les règles 1 et 2 afin de ne pas faire faux. Les autres règles nous permettront éventuellement ensuite de contrôler et d'optimiser... afin de faire juste.



Pour chacune de ces affectations, nous avons bien vérifié que la rubrique n'a qu'une seule valeur pour chaque entité de la classe. Une fois les affectations ci-dessus réalisées (les plus évidentes), il nous reste donc :

- N° engagement
- Quantité produit
- Date lettre refus

Affectation des rubriques restantes

Après analyse des documents, on voit qu'il y a un seul numéro d'engagement par commande. D'autre part, la lettre de refus ne concernant qu'une facture donnée, il ne peut y avoir qu'une seule date de lettre de refus par facture.

Le numéro d'engagement est donc affecté à la commande et la date de la lettre de refus est affectée à la facture.

Concernant la quantité de produits, comme nous l'avons vu pour la facturation Beaux Rivages, elle ne caractérise pas le produit parce que sa valeur peut varier en fonction des commandes sur lesquelles se trouve le produit, et elle ne caractérise pas la commande parce que sa valeur dépend des produits présents sur la commande : pour un produit donné, il n'y a donc pas une seule quantité et, pour une commande donnée, il n'y a pas non plus une seule quantité…

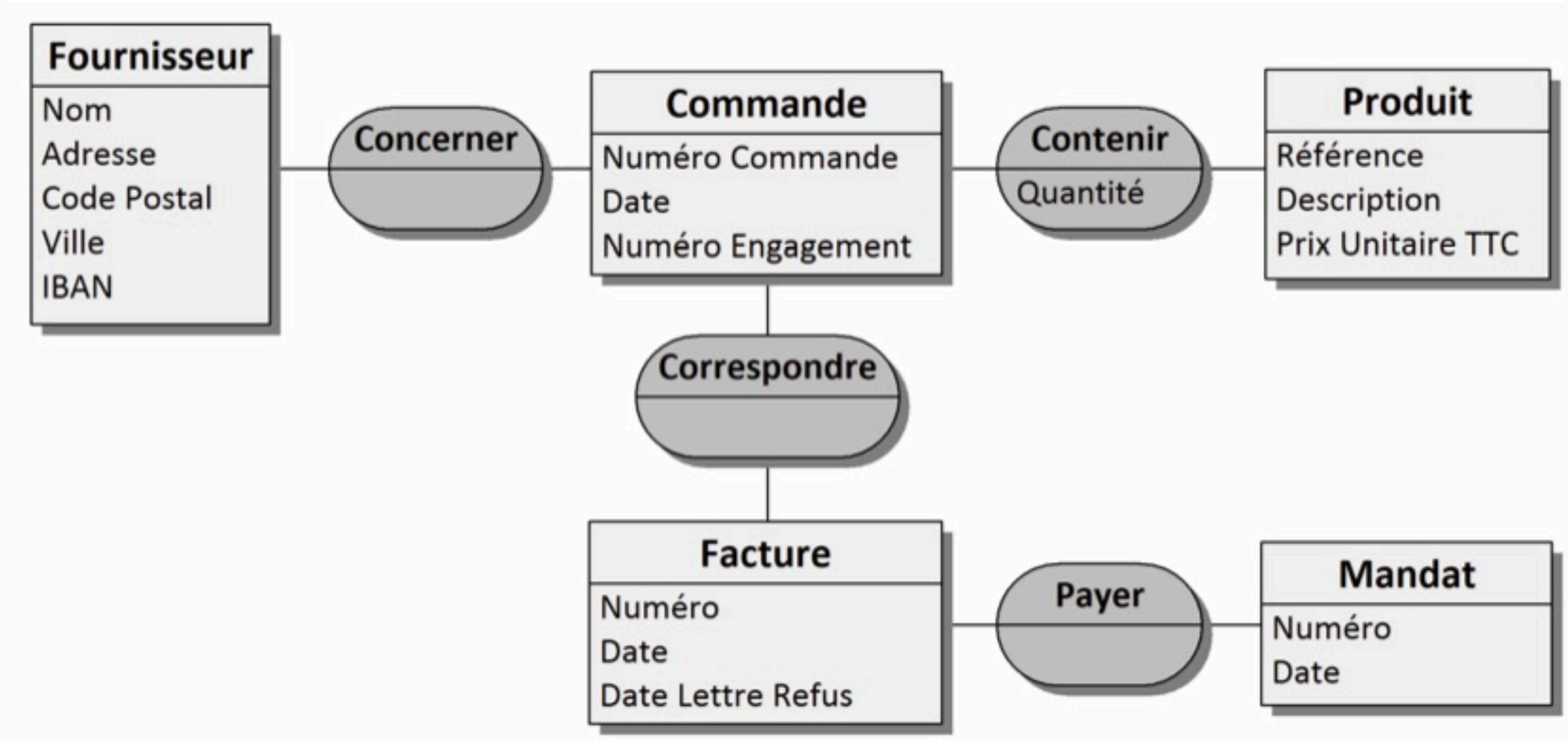
Mais, pour un produit donné **ET** une commande donnée, il n'y a effectivement qu'une seule quantité commandée. Par conséquent, cette quantité caractérise l'association entre un produit et une commande.

Sur le modèle, nous positionnerons donc cette dernière rubrique du dictionnaire des données entre les deux classes d'entités concernées.

Sémantique : mise en œuvre des associations

Une association apparait alors entre les classes d'entités « Commande » et « Produit ».

D'autres associations sans rubrique doivent également venir compléter le modèle pour décrire les liens entre les différentes classes d'entités. Voici les associations qui apparaissent clairement à la lecture du compte-rendu d'interview :



Modèle conceptuel de données

Détermination des cardinalités

Il ne reste plus qu'à définir les cardinalités et à vérifier les règles afin de finaliser le modèle conceptuel de données :

- Chaque fournisseur est concerné par **0** à **n** commandes (on peut en effet considérer qu'un fournisseur propose des produits mais que l'administration ne fait jamais appel à lui).
- Chaque commande concerne **1** et **1** seul fournisseur.
- Chaque commande contient **1** à **n** produits (une commande ne peut pas contenir **0** produit).

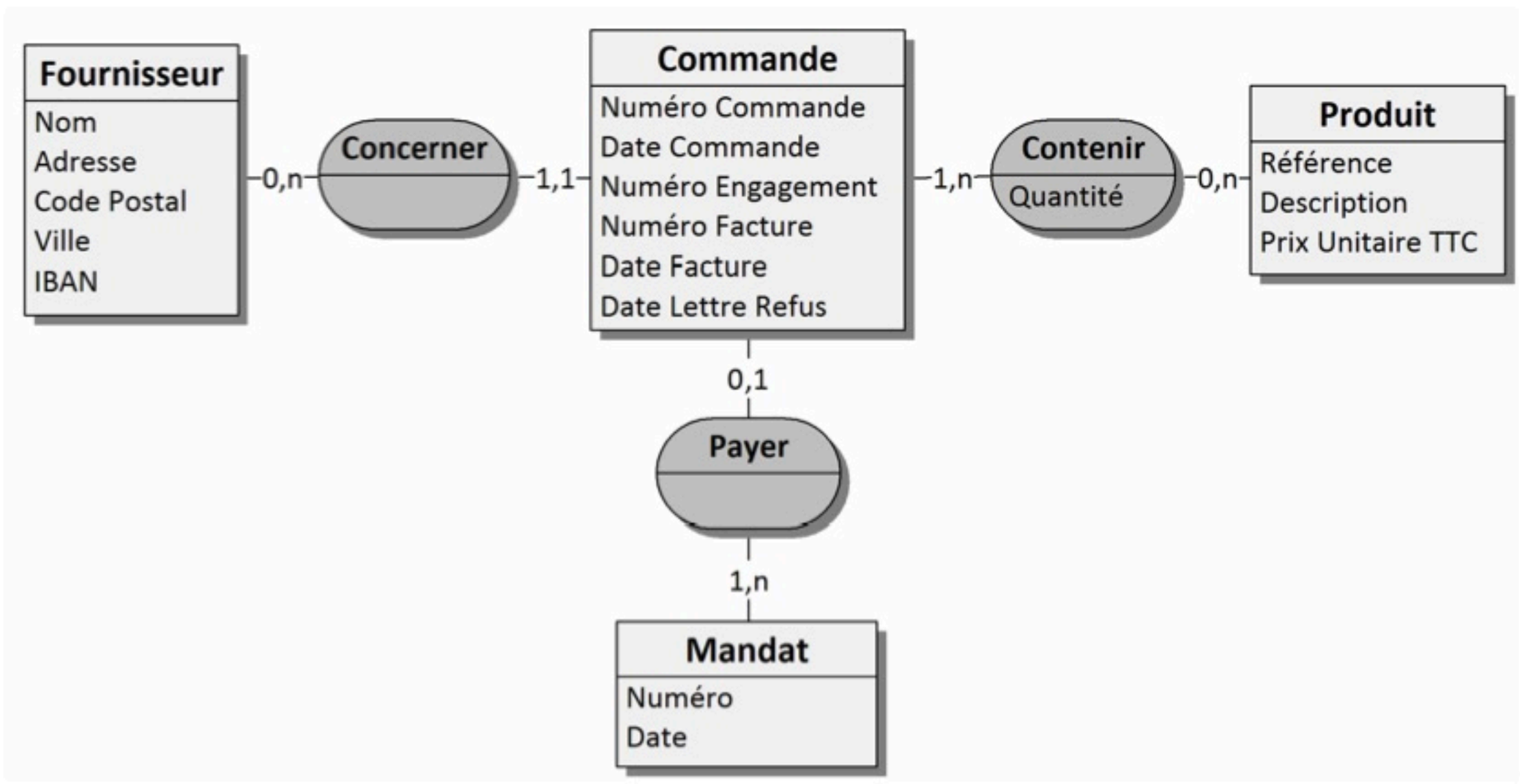
© Patrick Bergougnoux

- Chaque produit est contenu dans **0** à **n** commandes (un produit proposé peut ne jamais être commandé).
- Chaque commande correspond à **1** et **1** seule facture.
- Chaque facture correspond à **1** et **1** seule commande.

L'association « Correspondre » possède des cardinalités **1,1** de part et d'autre ➔ **Règle 5** : les classes d'entités « Commande » et « Facture » doivent donc être regroupées.

- Chaque facture est payée par **0** ou **1** mandat (une facture peut ne pas être mandatée si elle est refusée).
- Chaque mandat paye **1** à **n** factures.

Après mise en œuvre des cardinalités et application de la règle 5, voici le modèle conceptuel de données correspondant à notre analyse :



Vérification du modèle

Pour bien vérifier un modèle, il faut tout d'abord s'assurer que les règles de cohérence soient bien respectées.

Il faut ensuite imaginer le fonctionnement du modèle dans une situation réelle afin d'être sûr qu'il répond correctement aux besoins du système d'information.

Le modèle est-il correct ?

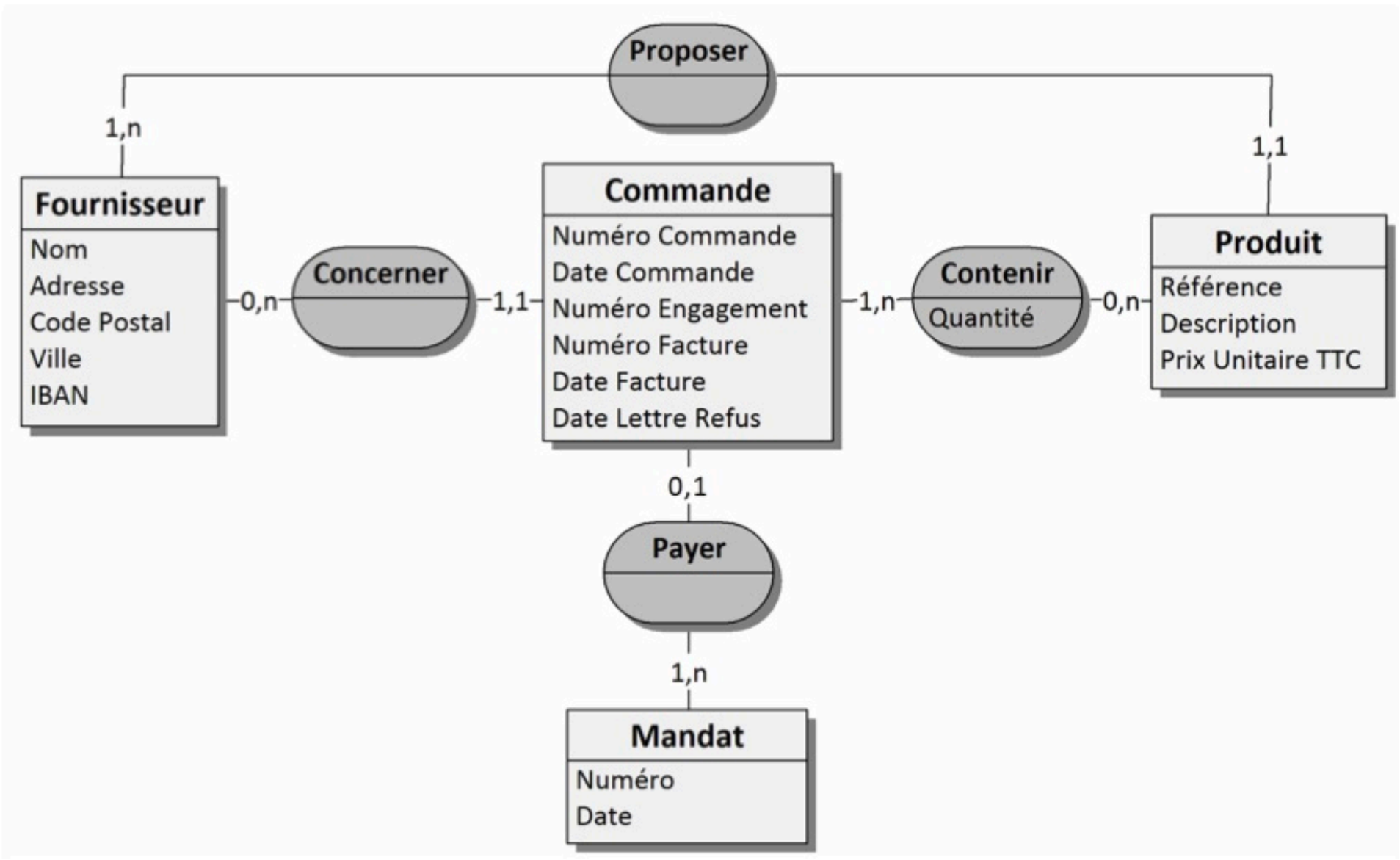
Et bien **NON** !

Mais pourquoi ?

Imaginons la situation réelle : l'administration XYZ souhaite commander des produits chez le fournisseur Durand, comme nous le montre le premier document en annexe du compte-rendu d'interview. Comment le système d'information connait-il les produits proposés par ce fournisseur ? D'après notre modèle, les produits ne sont associés aux fournisseurs qu'à travers les commandes qui lui ont été faites… Autrement dit, il faut que la commande existe pour pouvoir passer commande !

La sémantique du modèle n'est donc pas complète : il est en effet nécessaire d'associer les fournisseurs aux produits qu'ils proposent afin de pouvoir leur passer commande.

D'après le compte-rendu d'interview, une référence de produit donnée n'est proposée que par un seul fournisseur ; le modèle conceptuel de données complet est alors le suivant :



Variante : un même produit proposé par différents fournisseurs

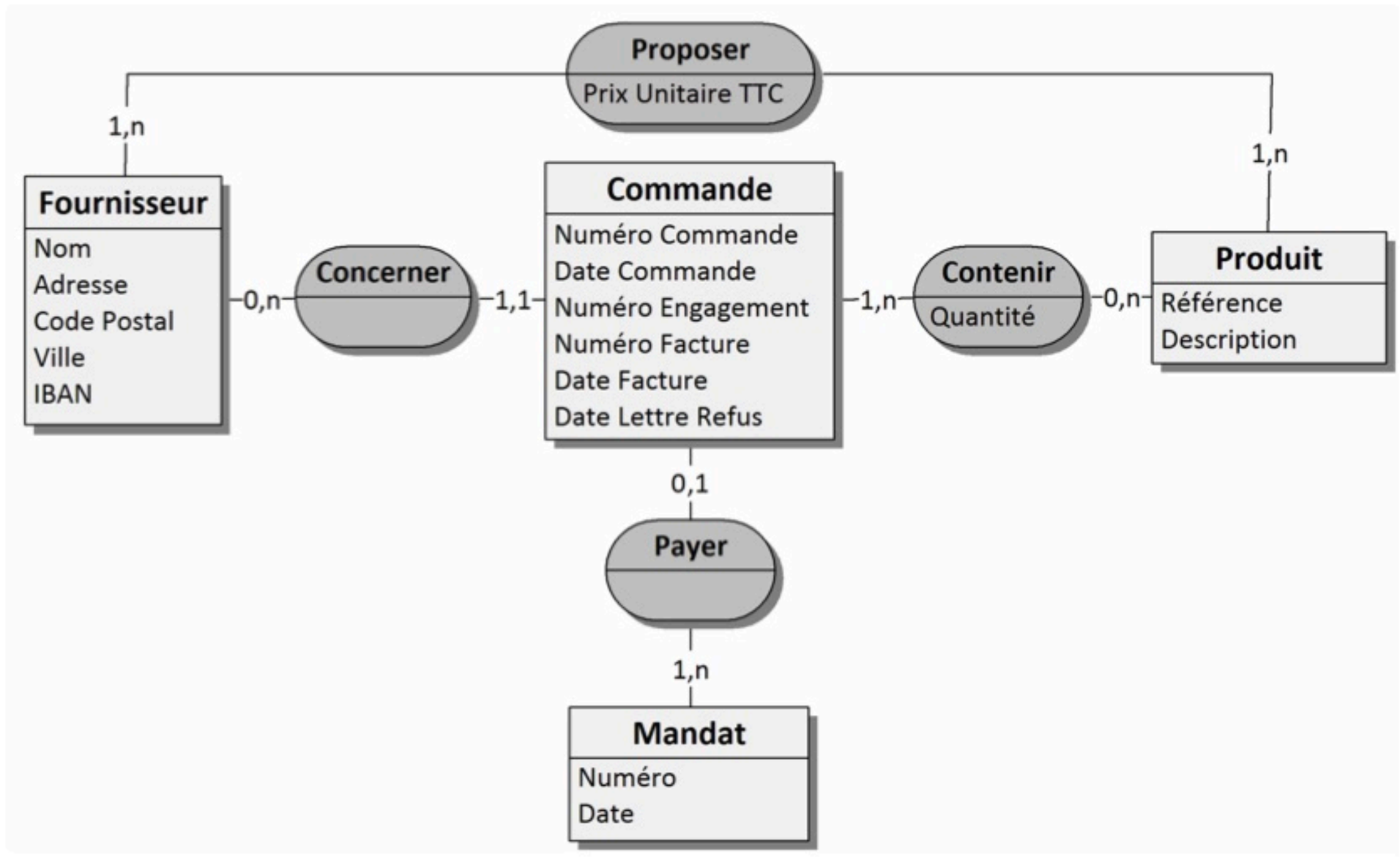
Une évolution du système d'information pourrait demander à ce que plusieurs fournisseurs puissent proposer une même référence de produit, chaque fournisseur pouvant alors pratiquer son propre prix de vente. L'administration XYZ pourrait ainsi choisir le fournisseur qui fait la meilleure offre.

© Patrick Bergougnoux

La première évolution se situe au niveau des cardinalités ; en effet :

- Chaque fournisseur peut proposer **1** à **n** produits.
- Chaque produit est proposé par **1** à **n** fournisseurs.

Par ailleurs, chaque fournisseur peut définir son propre prix pour chaque produit qu'il propose : par conséquent, le prix ne caractérise plus le produit puisqu'il peut varier en fonction des fournisseurs qui le proposent, et il ne caractérise toujours pas la commande qui comporte des prix différents en fonction des produits présents sur la commande. Mais, pour un fournisseur donné **ET** un produit donné, il n'y a effectivement qu'un seul prix de vente. Le prix unitaire TTC caractérise donc l'association entre un fournisseur et un produit. Voici le modèle correspondant à cette variante :



Une rubrique est maintenant présente dans l'association « Proposer » : il faut alors s'assurer du respect de la règle 4, à savoir que les cardinalités de cette association sont bien multiples de part et d'autre. C'est le cas : notre modèle conceptuel de données est terminé.

Ce que le modèle met en évidence

- Application de la règle 5 avec la fusion des deux classes d'entités « Commande » et « Facture ».
- Respect de la sémantique du modèle en vérifiant le bon fonctionnement du modèle en situation réelle.

Société canine

Rappel du dictionnaire des données

- Nom chien
- Sexe chien
- Intitulé race
- Origine race
- Descriptif race
- Nom propriétaire
- Adresse propriétaire
- Date acquisition
- Ville concours
- Date concours
- Nombre chiens primés
- Nombre participants
- Classement
- Date naissance chien

Répartition des données dans les classes d'entités

Première détermination des classes d'entités

Trois classes d'entités apparaissent naturellement dans le dictionnaire des données :

- Chien
- Propriétaire
- Concours

Rappelons qu'il s'agit d'un point de départ qui pourra ensuite évoluer en fonction des règles de cohérence du modèle.

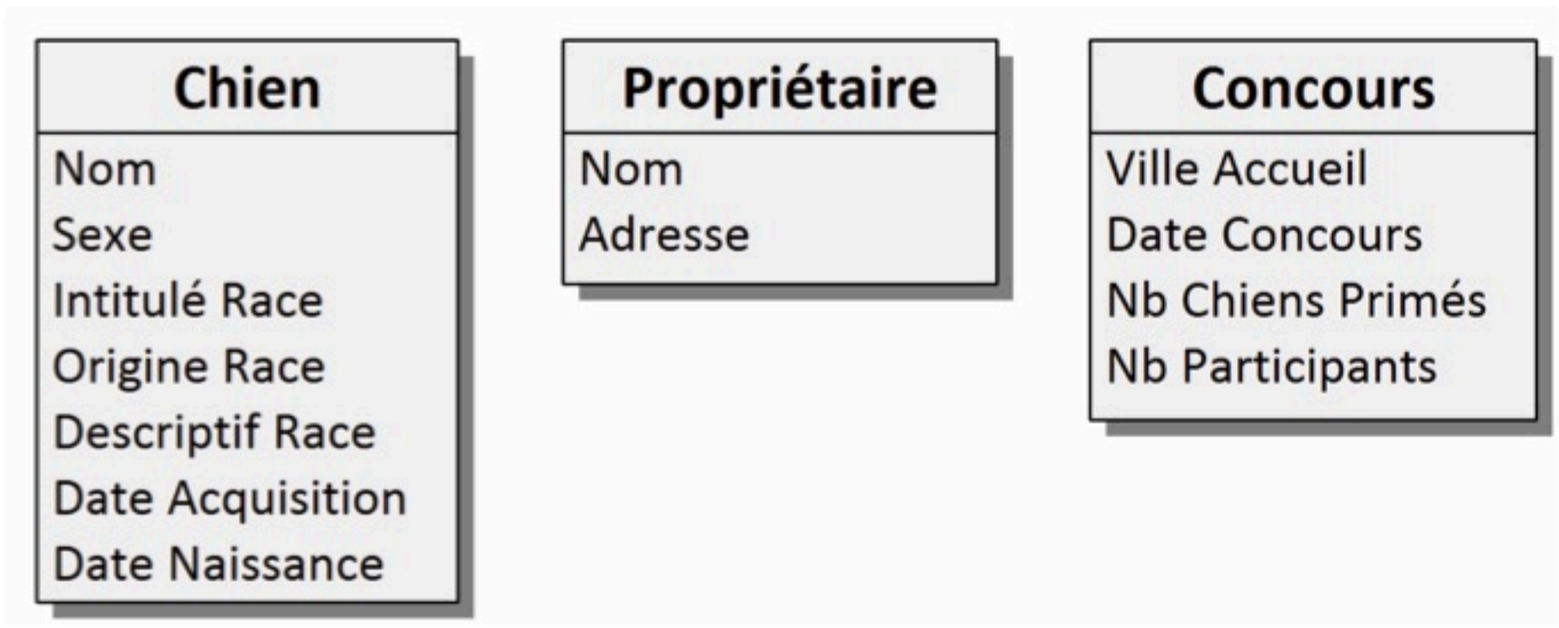
Affectation des rubriques les plus évidentes

Les affectations proposées ci-dessous ne posent pas de problème, mais il convient cependant de prendre quelques secondes pour bien vérifier, pour chaque rubrique, la règle 2.

© Patrick Bergougnoux

Si l'on n'y prend pas garde, on peut être conduit à affecter la date d'acquisition au propriétaire : en effet, c'est lui qui fait l'acquisition… mais s'il possède plusieurs chiens, il aura plusieurs dates d'acquisition.

Par ailleurs, le compte-rendu spécifie que l'on ne considère que le dernier propriétaire d'un chien ; il n'y a alors qu'une date d'acquisition par chien : cette date caractérise donc la classe « Chien ».



Pour chaque affectation, les rubriques n'ont qu'une seule valeur pour chaque entité de la classe. Une fois ces affectations effectuées, il ne reste plus qu'une rubrique du dictionnaire des données à affecter.

Affectation des rubriques restantes

Cette rubrique, « Classement », ne se place pas de manière évidente dans l'une de ces trois classes d'entités.

Essayons tout de même de l'affecter aux classes existantes :

- Dans « Chien » : cela signifierait qu'un chien a toujours le même classement, quels que soient les concours auxquels il participe.
- Dans « Propriétaire » : c'est pire ! Tous les chiens d'un même propriétaire auraient tous le même classement à tous les concours.
- Dans « Concours » : cela signifierait que tous les chiens participant à un concours ont le même classement.

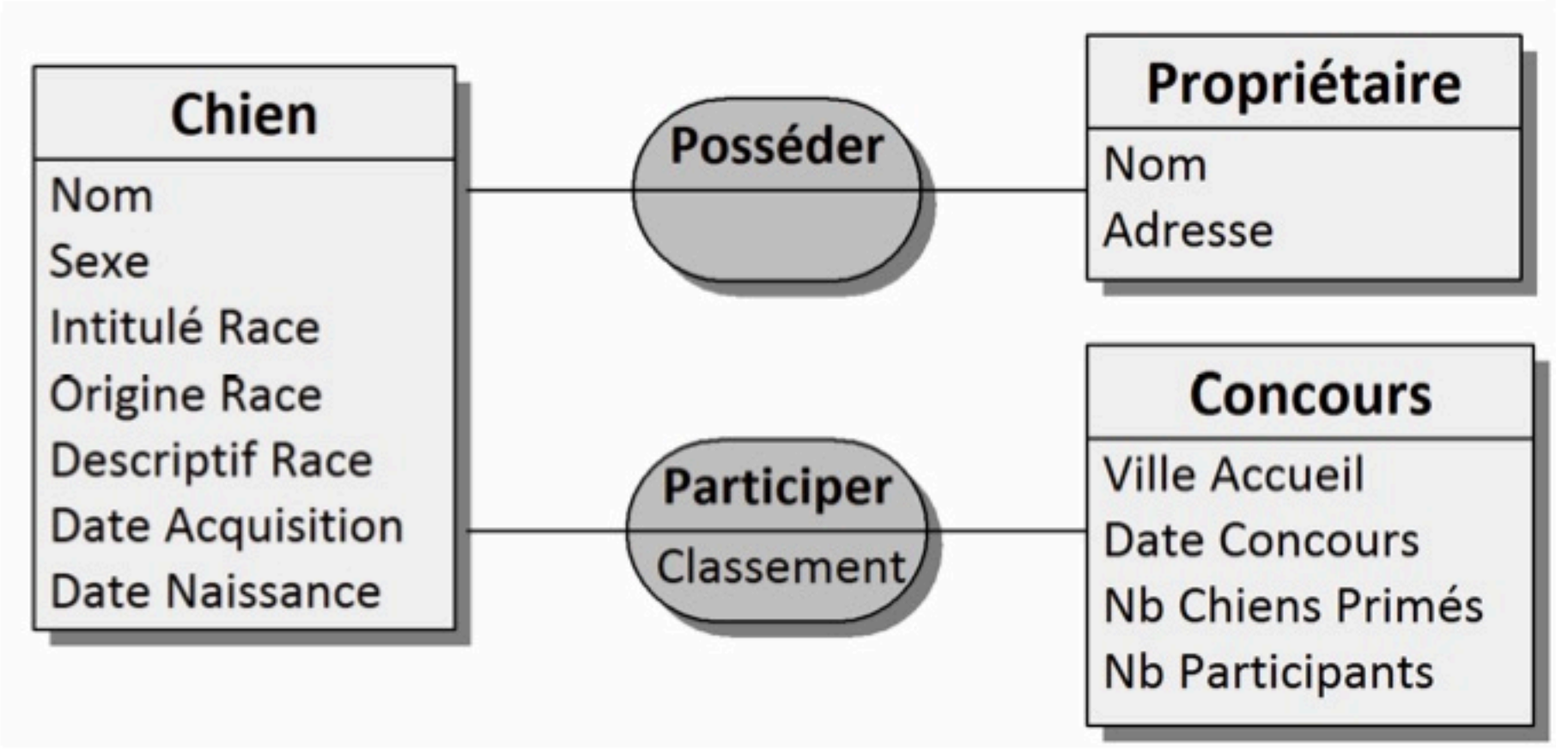
Par conséquent, aucune classe existante ne convient : vérifions maintenant si le classement caractérise une association entre classes.

C'est effectivement le cas : pour une participation d'un chien donné à un concours donné, il y a bien un et un seul classement.

Sémantique : mise en œuvre des associations

Sur le modèle, la rubrique « Classement » est donc affectée à l'association « Participer » qui va relier les classes « Chien » et « Concours ».

Une autre association, sans rubrique, semble également évidente entre les classes d'entités « Chien » et « Propriétaire » pour finaliser la sémantique de ce modèle.



Modèle conceptuel de données

Détermination des cardinalités

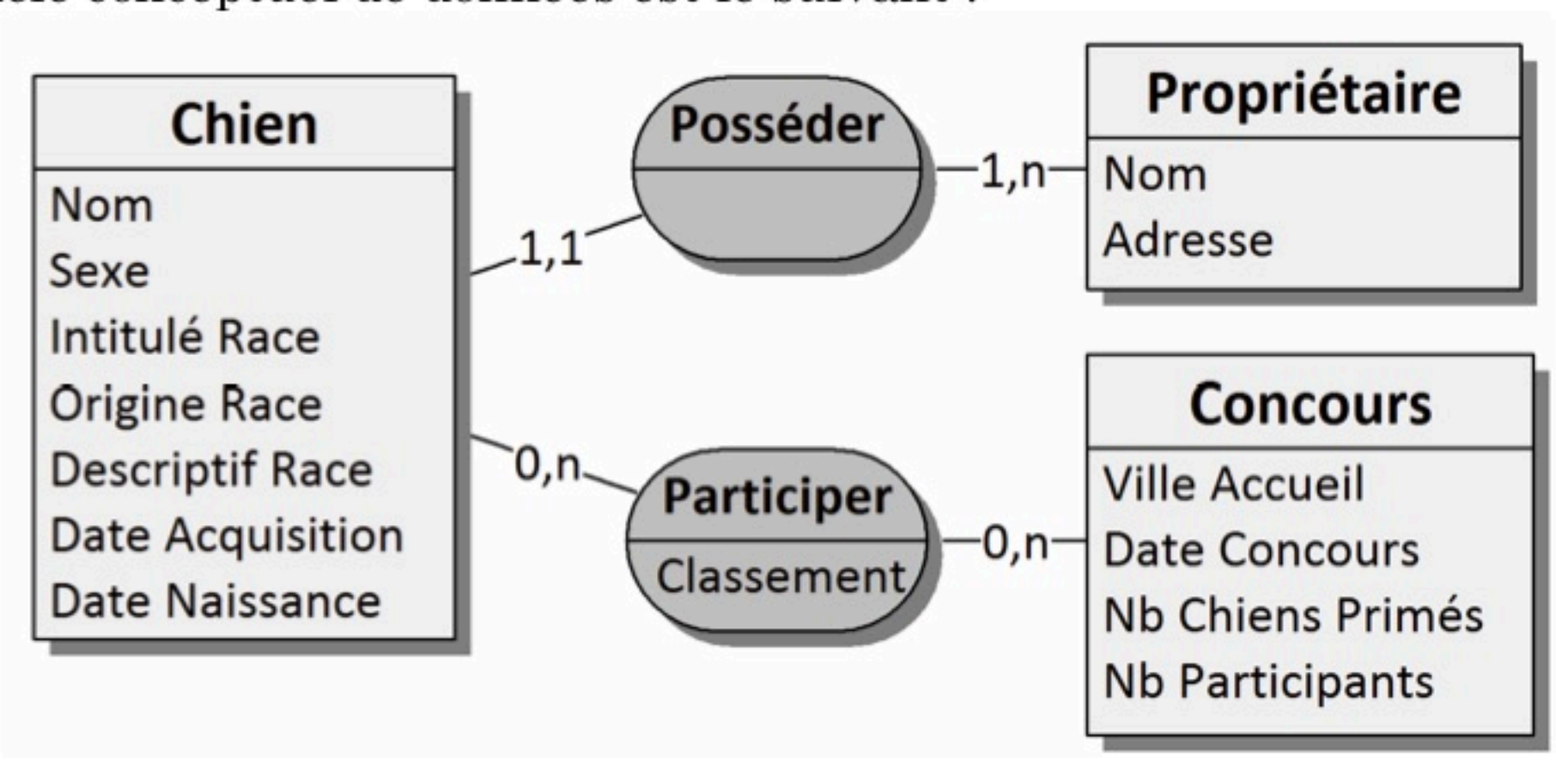
Les cardinalités de ce modèle ne posent pas de problème, mais il convient cependant de rester attentif à certains détails :

- Chaque chien est possédé par **1** et **1** seul propriétaire (dans la mesure où l'on ne considère que le dernier propriétaire d'un chien).
- Chaque propriétaire possède **1** à **n** chiens.
- Chaque chien participe à **0** à **n** concours (un chien peut ne pas faire de concours).
- Chaque concours voit participer **0** à **n** chiens.

On pourrait penser que, pour faire un concours, il faut au moins 2 chiens ! Mais, attention, tous les participants n'appartiennent pas forcément à la société Botoutou, et des concours peuvent avoir lieu sans qu'aucun des chiens de Botoutou n'y participe.

© Patrick Bergougnoux

Le modèle conceptuel de données est le suivant :



Vérification du modèle

Les règles de cohérence 1 et 2 sont bien respectées. L'association « Participer » qui contient une rubrique a bien des cardinalités multiples de part et d'autre : donc, pas de problème avec la règle 4, ni avec la 5 puisqu'il n'y a pas d'association **1,1/1,1**. Reste à vérifier la règle 3.

Existe-t-il, dans une classe d'entités de ce modèle, un ensemble de rubriques qui possèdent les mêmes valeurs pour plusieurs entités de la classe ?

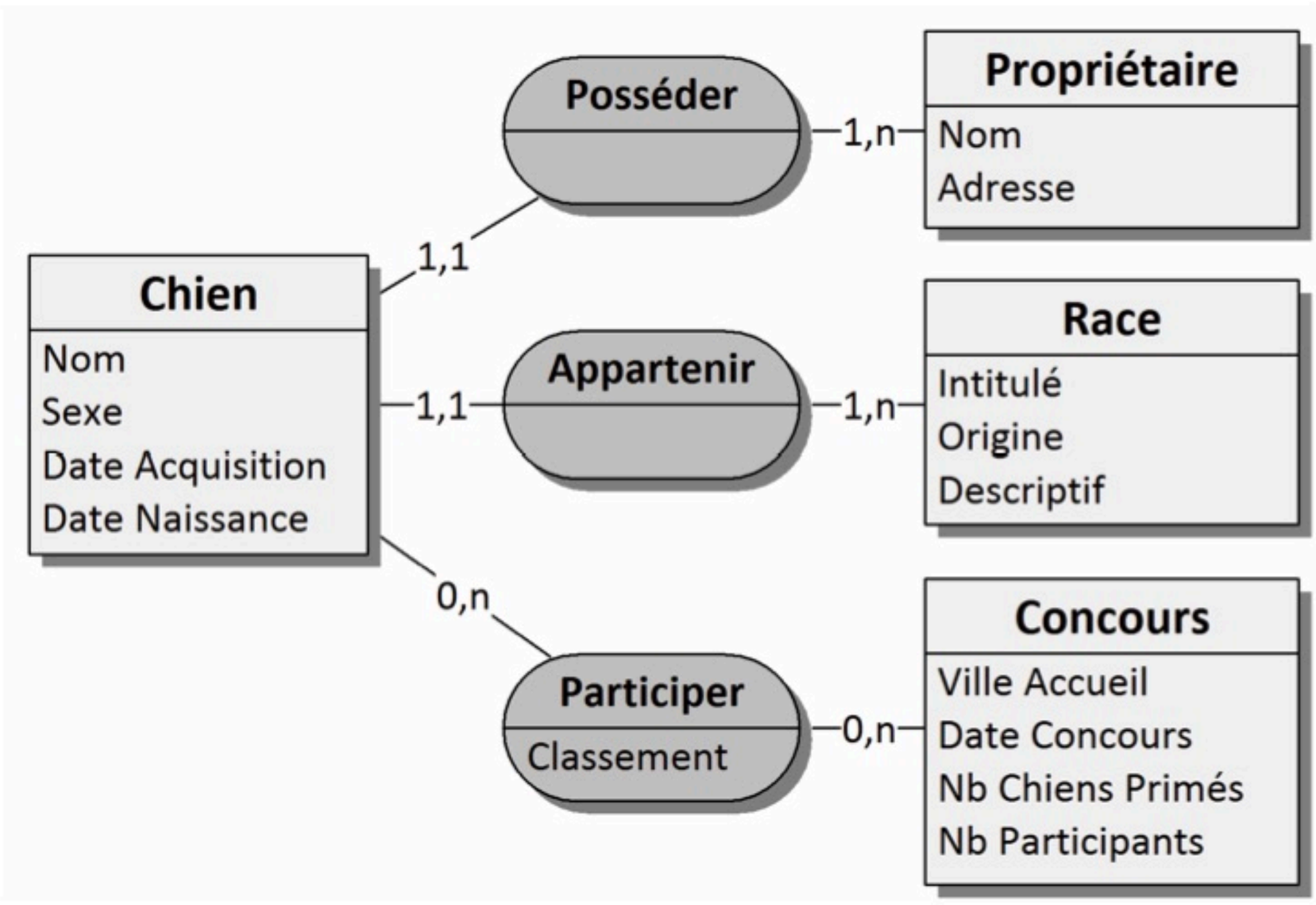
C'est effectivement le cas des rubriques « Intitulé Race », « Origine Race » et « Descriptif Race » qui, comme leur nom l'indique, ont des valeurs communes pour tous les chiens d'une même race. Plusieurs rubriques étant concernées, elles peuvent alors faire l'objet d'une classe à part.

Pour bien comprendre l'application de cette règle, prenons un cas réel : le descriptif d'une race de chien intègre ses caractéristiques morphologiques, les grandes lignes de son caractère, son histoire, … Tel que notre système est modélisé, toutes ces informations relatives à la race seront répétées et mémorisées à l'identique pour chaque chien d'une même race.

Imaginons que la société Botoutou répertorie une centaine de bergers allemands : les informations de cette race seront donc dupliquées une centaine de fois, avec tous les problèmes que cela va poser en termes de perte de place et surtout de cohérence du système d'information.

En effet, si de nouvelles informations viennent compléter le descriptif des bergers allemands, les modifications devront être effectuées sur la centaine de chiens de cette race.

Le rajout d'une classe d'entités « Race » évite tous ces inconvénients et s'avère donc très utile au modèle :



Variante : gestion des propriétaires successifs d'un chien

Au lieu de ne conserver que le dernier propriétaire d'un chien, faisons maintenant évoluer le modèle pour connaitre tous les propriétaires qui ont successivement fait l'acquisition du chien.

Tout d'abord, faisons évoluer les cardinalités :

- (En 50 ans) chaque chien est possédé par **1** à **n** propriétaires.
- (En 50 ans) chaque propriétaire possède **1** à **n** chiens.

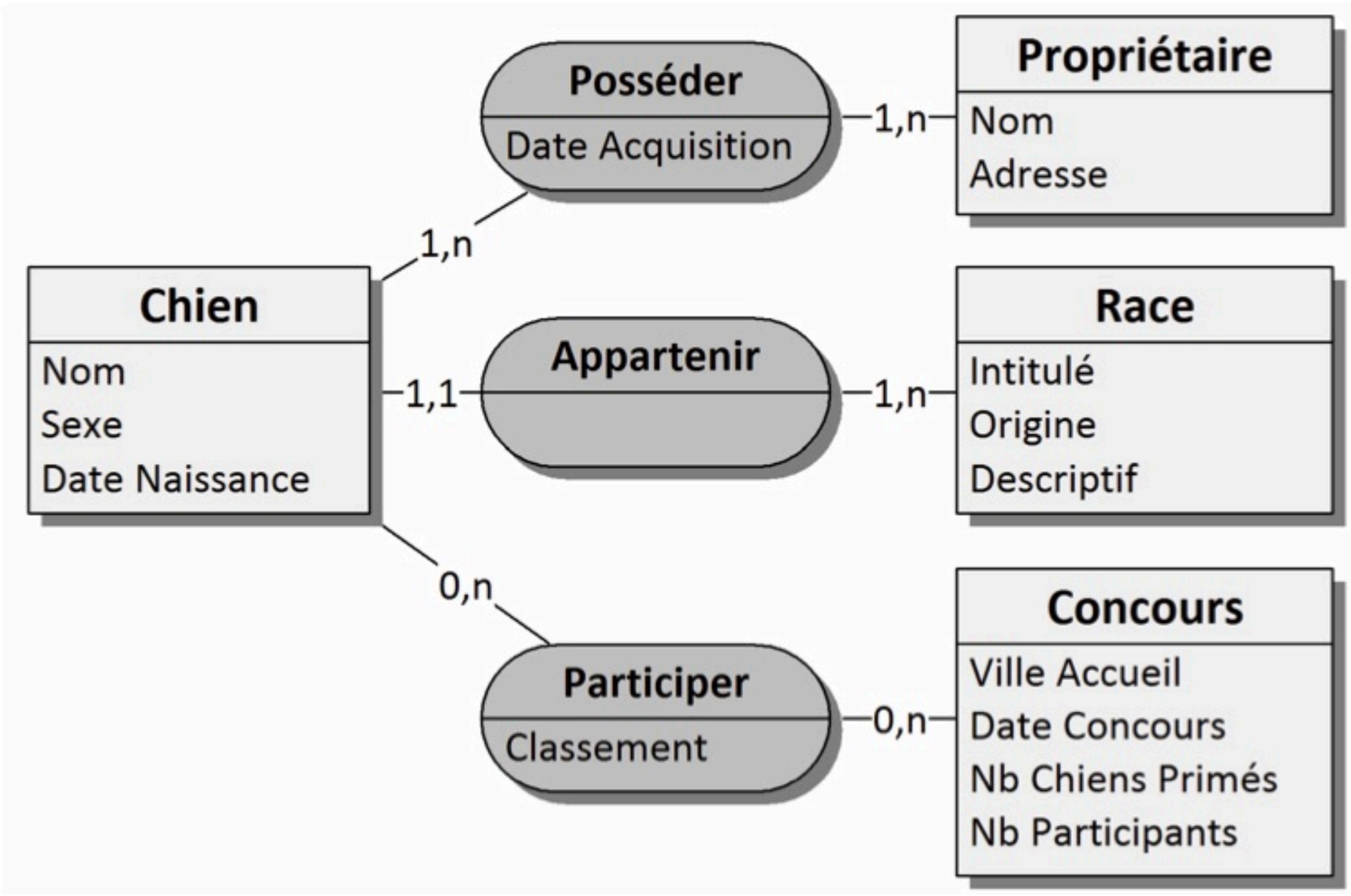
On remarque que, dans ce cas là, il est important de ne surtout pas figer l'association dans le temps (d'où l'utilisation de « En 50 ans »).

Un chien acquis successivement par plusieurs propriétaires aura donc plusieurs dates d'acquisition : cette rubrique ne peut alors plus rester dans la classe d'entités « Chien ». Elle ne peut toujours pas aller dans la classe « Propriétaire », un propriétaire ayant pu faire l'acquisition de plusieurs chiens à des dates différentes.

Mais, pour un chien donné **ET** un propriétaire donné, il n'y a effectivement qu'une seule date d'acquisition. Cette date caractérise donc l'association entre un chien et un propriétaire.

© Patrick Bergougnoux

Voici le modèle correspondant à cette variante :



La rubrique « Date Acquisition » est maintenant présente dans l'association « Posséder » ce qui est conforme à la règle 4 puisque les cardinalités de cette association sont bien multiples de part et d'autre.

Le modèle possède tout de même une limite : laquelle ?

Le fait que, pour un chien donné et un propriétaire donné, il n'y ait qu'une **SEULE** date d'acquisition provoque un des deux problèmes suivants :

- Soit un propriétaire ne peut pas racheter un chien qu'il a déjà acquis précédemment.
- Soit la nouvelle date d'acquisition prend la place de la précédente.

Dans les deux cas, l'historique sera incomplet. Nous verrons plus loin dans cet ouvrage comment traiter cette limite.

Ce que le modèle met en évidence

- Bonne affectation de la rubrique « Date Acquisition » dans le respect des règles 2 et 4.
- Application de la règle 3 avec la création de la classe « Race » qui n'avait pas été définie au départ.

Bibliothèque universitaire

Rappel du dictionnaire des données

- Nom étudiant
- Prénom étudiant
- Adresse étudiant
- Numéro carte étudiant
- Code ouvrage
- Titre ouvrage
- Nom auteur
- Prénom auteur
- Nationalité auteur
- Date prêt ouvrage
- Date retour ouvrage
- Nb total exemplaires

Répartition des données dans les classes d'entités

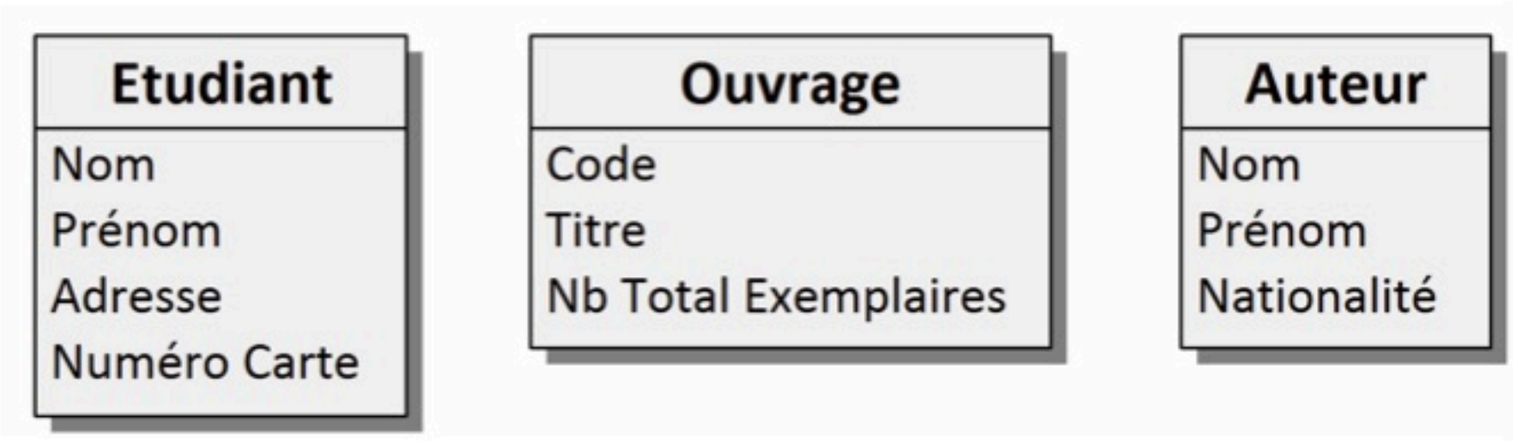
Première détermination des classes d'entités

Compte-tenu du faible nombre de rubriques, les classes d'entités apparaissent très rapidement :

- Etudiant
- Ouvrage
- Auteur

Affectation des rubriques les plus évidentes

Les premières affectations se déduisent de manière vraiment immédiate :



© Patrick Bergougnoux

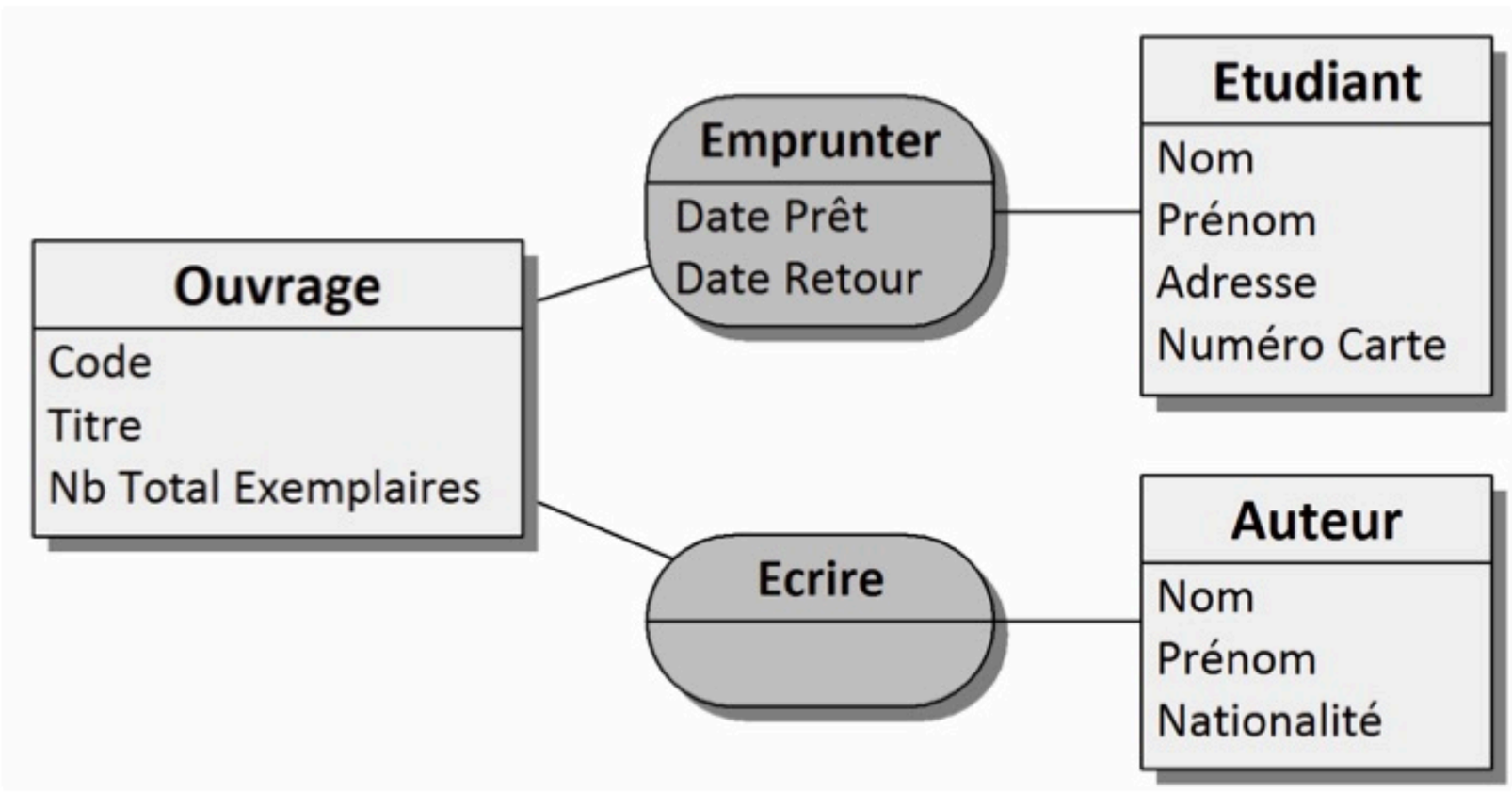
Pour chaque affectation, la règle 2 a été vérifiée et chaque rubrique n'a bien qu'une seule valeur pour chaque entité de la classe. Pour les deux rubriques restantes, « Date prêt ouvrage » et « Date retour ouvrage », l'affectation est moins évidente.

Affectation des rubriques restantes

Les dates de prêt et de retour des ouvrages ne caractérisent aucune des classes d'entités définies. Par contre, on peut voir sans problème qu'elles caractérisent l'association entre « Etudiant » et « Ouvrage ». Sur le modèle, ces deux dates seront donc placées entre les deux classes d'entités concernées.

Sémantique : mise en œuvre des associations

Une association contenant ces deux rubriques existe donc entre les classes d'entités « Etudiant » et « Ouvrage ». Une autre association, bien que ne contenant aucune rubrique, relie de manière naturelle les ouvrages et les auteurs :



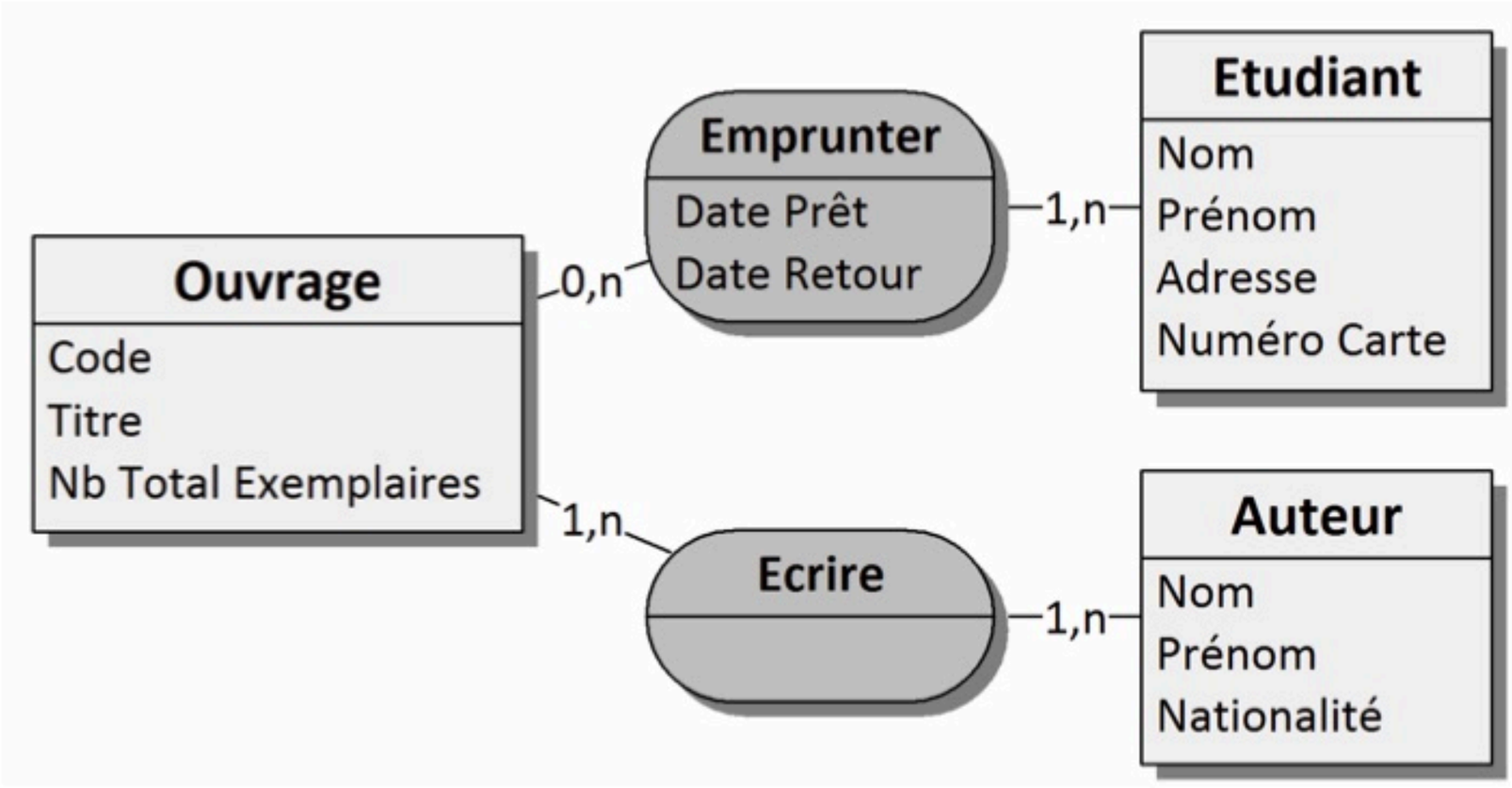
Modèle conceptuel de données

Détermination des cardinalités

Les cardinalités de ce modèle se déterminent sans difficulté particulière :

- Chaque ouvrage est emprunté par **0** à **n** étudiants (certains ouvrages peuvent ne jamais être empruntés).
- Chaque étudiant emprunte **1** à **n** ouvrage (la fiche étudiant est créée lors du premier emprunt).
- Chaque ouvrage est écrit par **1** à **n** auteurs.
- Chaque auteur écrit **1** à **n** ouvrages.

On rappelle que l'ouvrage ne représente pas un livre physique mais une œuvre pouvant exister en plusieurs exemplaires (rubrique « Nb Total Exemplaires »). Voici le modèle correspondant à cette gestion de prêt :



L'association « Ecrire » possède des cardinalités multiples de part et d'autre sans porter de rubrique : **cela ne pose aucun problème**, la règle 4 interdisant uniquement aux associations de porter des rubriques si elles ne sont pas multiples de part et d'autre.

Vérification du modèle

Après vérification de toutes les règles de cohérence, le modèle semble correct.

Comme dans l'exercice précédent, et comme souvent quand une date est présente dans une association, ce modèle possède une limite : laquelle ?

Même type de conséquence que pour la société canine : un étudiant ne peut pas emprunter un ouvrage qu'il a déjà emprunté précédemment, ou la nouvelle date de prêt prend la place de la précédente. En effet, pour un étudiant donné et un ouvrage donné, il n'y a qu'**une seule** date de prêt.

Nous proposerons une solution à cette problématique en modélisation avancée *(cf. chapitre 13)*.

Ce que le modèle met en évidence

- Une association peut être multiple de part et d'autre sans forcément être porteuse de rubriques.
- Limite liée à la mémorisation de l'historique d'associations datées.

© Patrick Bergougnoux

Guide gastronomique

Pour les nouveaux exercices, nous nous focaliserons sur la technique de modélisation : les comptes-rendus d'interview décriront donc directement les informations du domaine d'étude et la réalisation des modèles organisationnels de flux ne sera alors pas nécessaire.

Compte-rendu d'interview

Le syndicat d'initiative d'une grande ville décide de constituer un guide gastronomique des restaurants de la ville, consultable sur le Web. Ce guide devra indiquer les différents plats et vins proposés dans les restaurants.

Trois catalogues sont actuellement distribués au public :

- *Le catalogue des restaurants de la ville avec leur nom, leur adresse, la liste des plats mis à la carte (certains d'entre eux pouvant être notés comme une spécialité du restaurant), et la liste des vins proposés.*
- *Le catalogue des plats que l'on peut déguster en ville : chaque plat est caractérisé par son nom, sa place dans le repas (entrée, dessert, ...), la liste des prix dans les restaurants qui le mettent à leur carte, et le vin éventuellement recommandé avec ce plat.*
- *Le catalogue des vins que l'on peut trouver en ville avec, pour chacun d'eux, le nom, l'origine (appellation et vignoble), la couleur, le millésime, et la liste des prix (bouteille et verre) variables en fonction des restaurants qui le proposent. On notera que plusieurs vins peuvent bien évidemment avoir la même origine.*

Tableau des rubriques

L'analyse de ce compte-rendu ne fait apparaitre aucune rubrique calculable, ni aucun paramètre. Cependant, il convient d'être attentif à deux éléments importants qui justifient l'utilisation des exemples afin d'éviter toute erreur :

- Tout d'abord, il faut faire attention aux listes indiquées dans le compte-rendu d'interview (liste de plats, liste de prix, ...) : ce ne sont pas des rubriques, mais des ensembles de rubriques ; en effet, si l'on devait donner un exemple pour une liste, ça pourrait bien être long… et ne pas rentrer dans la case du tableau ! En fait, un exemple doit correspondre à un ou quelques mots maximum, susceptibles d'être saisis dans une zone d'un formulaire.
- La rubrique « Spécialité » peut également entrainer une erreur : si l'on choisit trop rapidement un exemple de spécialité, on peut penser à « Steak frites », « Paëlla », « Couscous », … Le problème, c'est qu'il s'agit d'exemples pour la rubrique « Nom plat » qui existe déjà dans le tableau.

En fait, le bon exemple pour « Spécialité » est « Oui » ou « Non » (rubrique booléenne) ; pour bien le signaler, l'indication « O/N » peut être rajoutée au nom de la rubrique.

Les documents n'étant pas présentés en annexe, les exemples doivent donc être imaginés pour constituer correctement le tableau des rubriques ci-dessous :

N°	Libellé	Exemple	P/C	Calcul
1	Nom restaurant	Chez Maxime	P	
2	Adresse restaurant	Grande rue – 13080 Aix	P	
3	Nom plat	Paëlla	P	
4	Spécialité (O/N)	Oui	P	
5	Nom vin	Château Latour	P	
6	Place plat	Principal	P	
7	Prix plat	17,80 €	P	
8	Appellation vin	Saint-Émilion	P	
9	Vignoble	Bordelais	P	
10	Couleur vin	Rouge	P	
11	Millésime vin	2016	P	
12	Prix bouteille	78,00 €	P	
13	Prix verre	15,00 €	P	

Dictionnaire des données

Pour cet exercice, le dictionnaire est équivalent au tableau des rubriques :

- Nom restaurant
- Adresse restaurant
- Nom plat
- Spécialité (O/N)
- Nom vin
- Place plat
- Prix plat
- Appellation vin
- Vignoble
- Couleur vin
- Millésime vin
- Prix bouteille
- Prix verre

© Patrick Bergougnoux

Répartition des données dans les classes d'entités

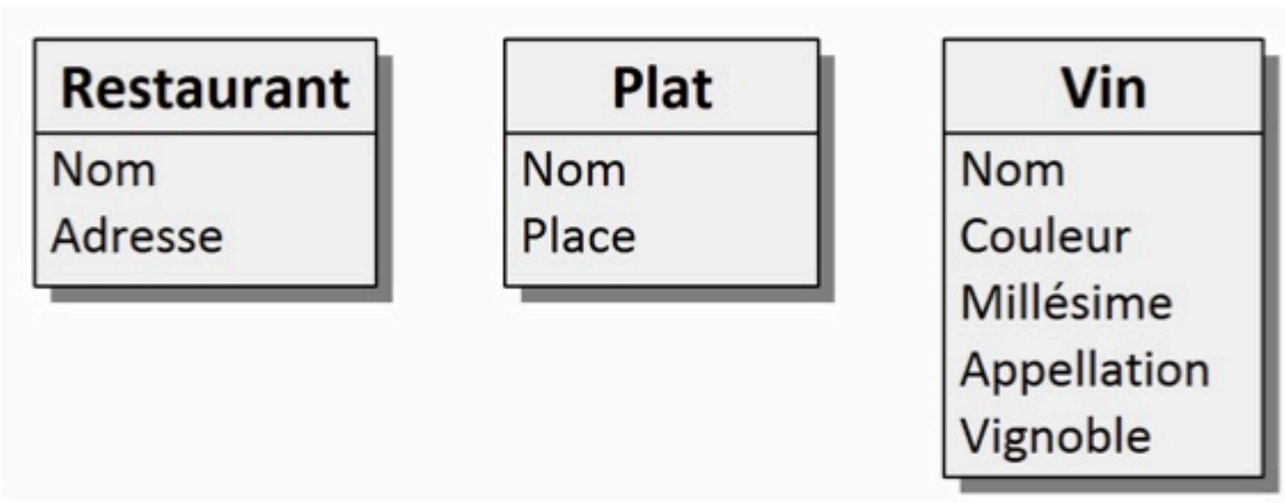
Première détermination des classes d'entités

Trois classes d'entités ressortent d'une analyse rapide du dictionnaire des données :

- Restaurant
- Plat
- Vin

Affectation des rubriques les plus évidentes

Neuf rubriques du dictionnaire des données trouvent naturellement leur place dans ces classes d'entités :



Pour chaque affectation, la règle 2 a été vérifiée et chaque rubrique n'a bien qu'une seule valeur pour chaque entité de la classe. Ce ne serait pas le cas des quatre rubriques restantes ci-dessous si l'on cherchait à les affecter à l'une de ces trois classes d'entités :

- Spécialité (O/N)
- Prix plat
- Prix bouteille
- Prix verre

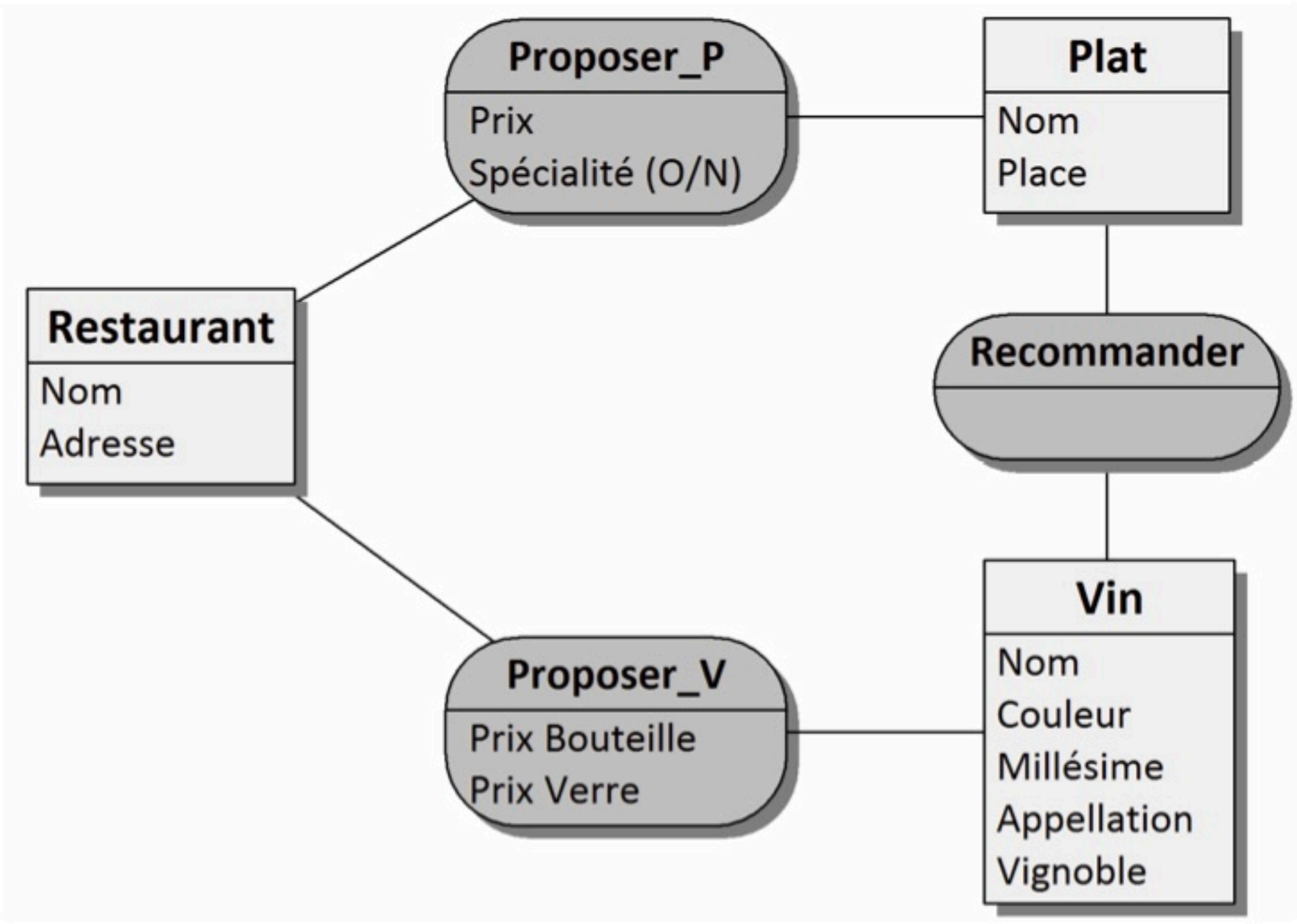
Affectation des rubriques restantes

Les prix des plats et des vins varient en fonction des restaurants dans lesquels ils sont proposés ; la rubrique « Prix plat » concerne donc l'association entre les classes d'entités « Plat » et « Restaurant », et les rubriques « Prix bouteille » et « Prix verre » caractérisent l'association entre les classes d'entités « Vin » et « Restaurant ».

De même, l'indication de spécialité concerne un plat donné dans un restaurant donné et doit être également placée dans l'association entre « Plat » et « Restaurant ».

Sémantique : mise en œuvre des associations

Les rubriques restantes que nous venons de traiter entrainent naturellement la mise en œuvre des associations dans lesquelles elles se trouvent. Il ne reste alors plus qu'à rajouter le lien permettant de spécifier la recommandation d'un vin avec un plat.



Modèle conceptuel de données

Détermination des cardinalités

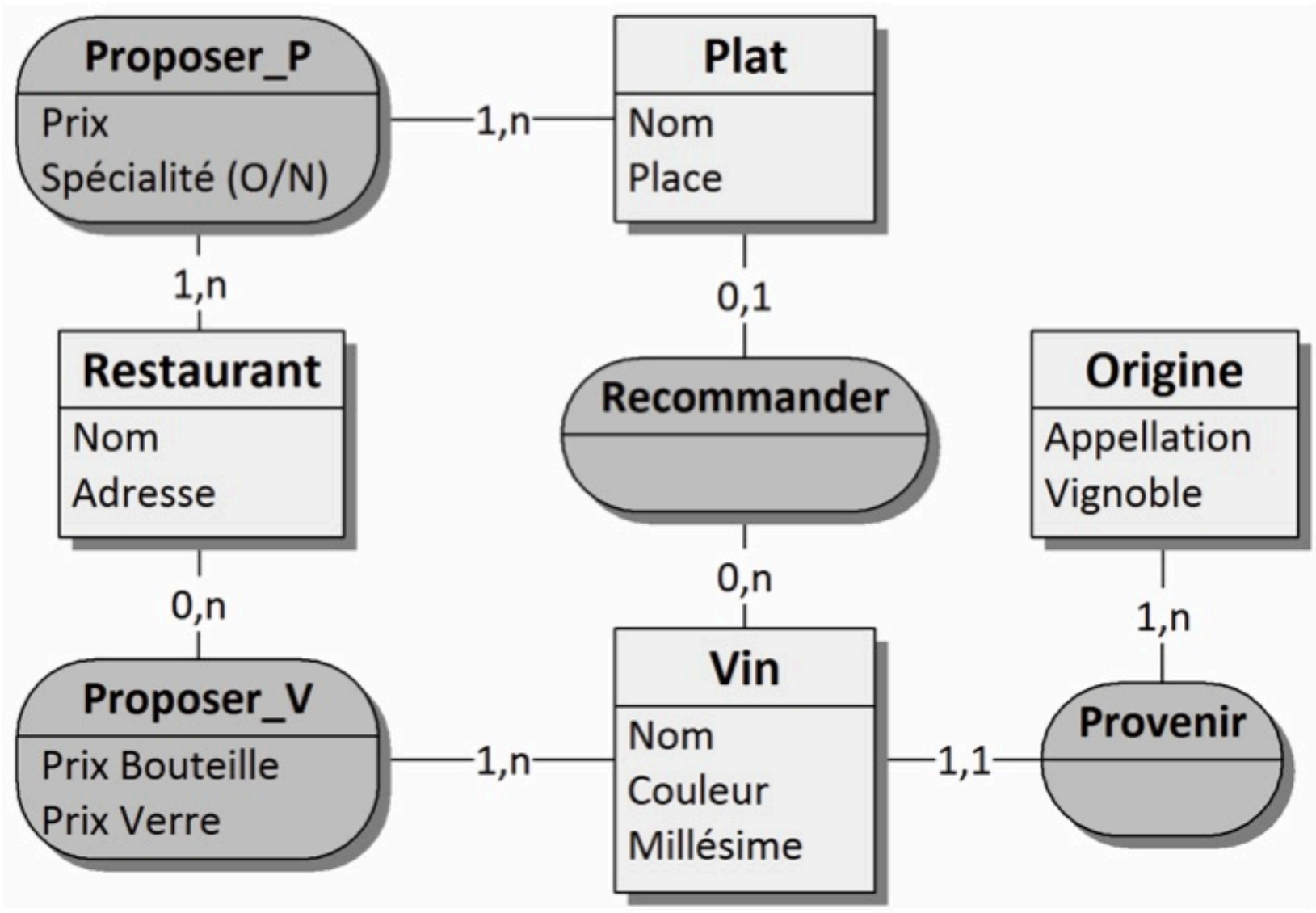
Les cardinalités des associations « Proposer_P » et « Proposer_V » se définissent de manière naturelle, avec des cardinalités forcément multiples de part et d'autre compte-tenu de la présence de rubriques ; précisons juste qu'un restaurant propose forcément au moins un plat, mais peut ne pas proposer de vin.

Concernant la recommandation des vins, le compte-rendu d'interview indique « *… **le** vin **éventuellement** recommandé avec ce plat.* » : autrement dit, un plat est recommandé avec **0** ou **1** vin ; par ailleurs, un vin peut être recommandé avec de nombreux plats… ou avec aucun.

On constate également que l'appellation et le vignoble sont deux rubriques qui caractérisent l'origine du vin : elles peuvent donc faire l'objet d'une classe « Origine » à part dans le respect de la règle de cohérence 3, avec une association **1,1/1,n**.

© Patrick Bergougnoux

Cela nous donne le modèle suivant :



Vérification du modèle

Les règles de cohérence, en particulier les règles 3 et 4, ont bien été prises en compte lors de la détermination des cardinalités.

La spécialité a été traitée avec une rubrique booléenne : une autre solution aurait consisté à rajouter une association **0,n/0,n** supplémentaire entre « Plat » et « Restaurant » pour associer les plats aux restaurants qui les proposent en tant que spécialités.

Cette possibilité sera souvent une alternative aux rubriques booléennes présentes dans les associations, mais ne sera pas forcément privilégiée compte-tenu de l'alourdissement qu'elle génère au niveau de la structure de la base de données correspondante.

Ce que le modèle met en évidence

- Il est important d'imaginer des exemples significatifs afin d'éviter des confusions telles que celles rencontrées pour « Nom plat » et « Spécialité ». Attention également aux listes qui ne sont pas des rubriques, mais des ensembles de rubriques.
- L'association **0,1/0,n** entre les classes d'entités « Plat » et « Vin » est intéressante à analyser, ainsi que l'application de la règle 3 pour l'origine des vins.

Représentations théâtrales

Compte-rendu d'interview

Pour tenir à jour un registre de ses représentations, le théâtre « L'Atout Jouer » décide de rassembler les informations issues de ses spectacles depuis sa création. Ces informations concernent les acteurs et les représentations qu'ils ont effectuées, une représentation impliquant bien sûr un ou plusieurs acteurs. La direction du théâtre dispose actuellement des éléments suivants :

- *La liste des acteurs (nom, prénom, adresse et âge le jour de la demande) qui ont participé à des représentations avec leur rôle dans chacune de ces représentations.*
- *Le nom, le prénom et la nationalité de l'organisateur qui a mis en œuvre une représentation donnée, une représentation n'étant confiée qu'à un seul organisateur.*
- *Toutes les informations relatives aux représentations auxquelles a participé un acteur donné : description, date, nombre de spectateurs, et la pièce jouée, sachant qu'une pièce peut donner lieu à plusieurs représentations avec des acteurs différents, et que ces acteurs n'ont pas forcément le même rôle à chaque représentation d'une même pièce.*
- *La liste des pièces jouées à « L'Atout Jouer » avec pour chacune : son titre, sa durée, sa date de création ainsi que le nom, le prénom et l'âge de ses auteurs au moment de la création de la pièce ; une pièce peut avoir été coécrite par plusieurs auteurs : dans ce cas, l'un d'entre eux peut éventuellement être noté comme l'auteur principal ayant dirigé l'écriture de la pièce en question.*

Tableau des rubriques

Les rubriques apparaissent clairement dans le compte-rendu et il suffit donc de les reporter dans le tableau au fur et à mesure de leur énoncé. Le nom du théâtre, unique pour tout le système d'information, est bien sûr un paramètre.

À la première lecture, on pourrait penser qu'il n'y a pas de rubriques calculables. Or, à deux reprises, des âges (acteur et auteur) sont répertoriés : ce sont de « mauvaises » rubriques primaires dans la mesure où elles évoluent sans cesse dans le temps ; il faut alors les ramener à des rubriques pérennes, à savoir des dates de naissance qui permettront de calculer à tout moment l'âge des acteurs et des auteurs.

Comme dans l'exercice précédent, les documents ne sont pas présentés en annexe et les exemples sont donc inventés pour compléter le tableau des rubriques :

© Patrick Bergougnoux

N°	Libellé	Exemple	P/C	Calcul
~~1~~	~~Nom théâtre~~	L'Atout Jouer	P	*Paramètre*
2	Nom acteur	Faitro	P	
3	Prénom acteur	Jean	P	
4	Adresse acteur	12 rue Bis, 75000 Paris	P	
~~5~~	~~Âge acteur~~	35 ans	C	[21] - [20]
6	Rôle	Cyrano	P	
7	Nom organisateur	Proviste	P	
8	Prénom organisateur	Alain	P	
9	Nationalité organisateur	Belge	P	
10	Description représentation	Première de la pièce	P	
11	Date représentation	14/02/2019	P	
12	Nombre spectateurs	168	P	
13	Titre pièce	Il était à Bergerac, une fois	P	
14	Durée pièce	1h35	P	
15	Date création pièce	20/07/2018	P	
16	Nom auteur	Pagatha	P	
17	Prénom auteur	Christy	P	
18	Auteur principal (O/N)	Non	P	
~~19~~	~~Âge auteur~~	51 ans	C	[15] - [22]
20	Date naissance acteur	18/08/1983	P	
~~21~~	~~Date du jour~~	25/05/2019	P	*Paramètre*
22	Date naissance auteur	26/03/1967	P	

Dictionnaire des données

- Nom acteur
- Prénom acteur
- Adresse acteur
- Rôle
- Nom organisateur
- Prénom organisateur
- Nationalité organisateur
- Description représentation
- Date représentation
- Nombre spectateurs
- Titre pièce
- Durée pièce
- Date création pièce
- Nom auteur
- Prénom auteur
- Auteur principal (O/N)
- Date naissance acteur
- Date naissance auteur

Répartition des données dans les classes d'entités

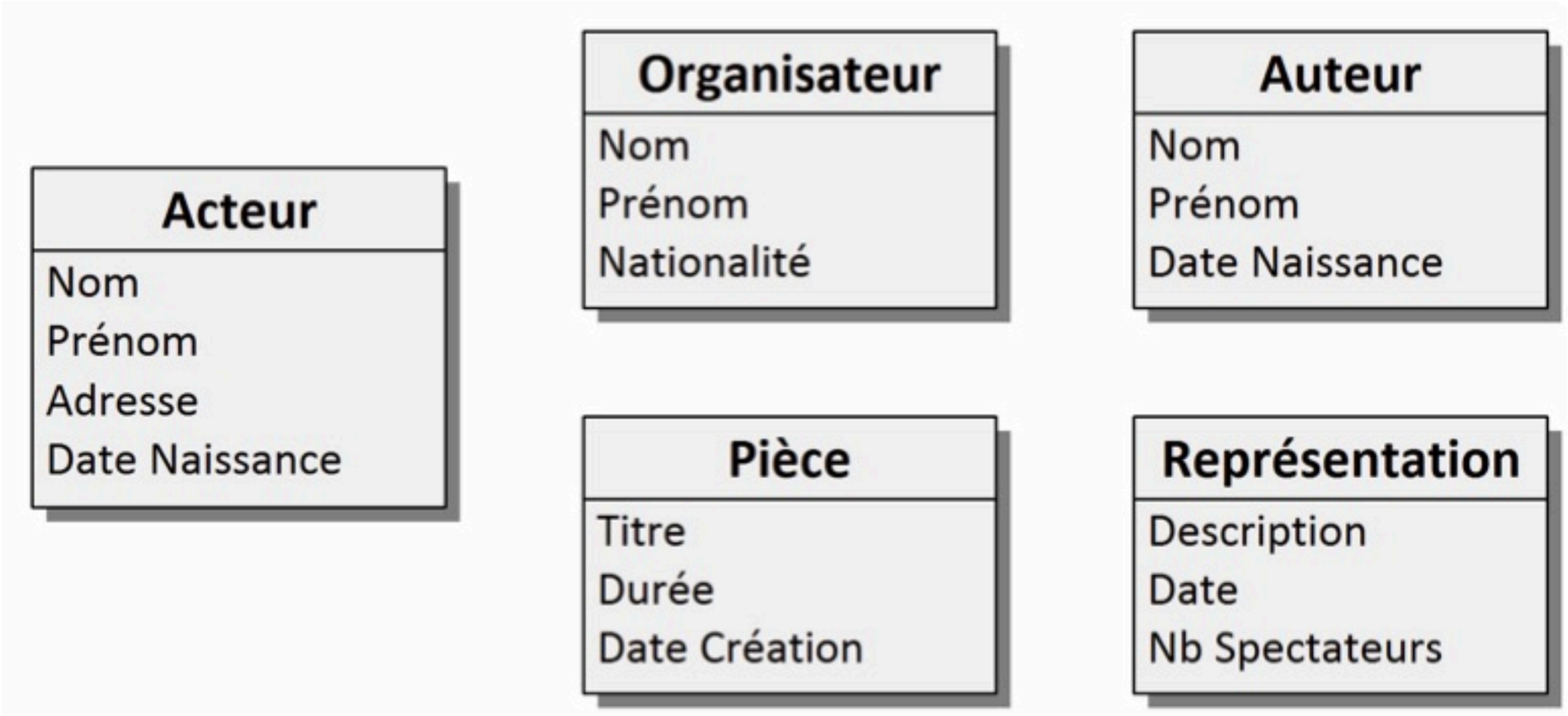
Première détermination des classes d'entités

Cinq classes d'entités ressortent immédiatement de l'analyse rapide du dictionnaire des données :

- Acteur
- Organisateur
- Représentation
- Pièce
- Auteur

Affectation des rubriques les plus évidentes

Le compte-rendu d'interview indique clairement l'affectation de la plupart des rubriques du dictionnaire des données :



Affectation des rubriques restantes

Il ne reste que deux rubriques à affecter : « Rôle » et « Auteur principal (O/N) » :

- Un acteur peut jouer des rôles différents, et jouer une pièce nécessite bien sûr plusieurs rôles. On pourrait alors être tenté de placer le rôle dans une association entre les classes d'entités « Acteur » et « Pièce » : dans ce cas, un acteur donné jouerait un et un seul rôle pour toutes les représentations d'une pièce donnée. Or le compte-rendu d'interview précise que les acteurs n'ont pas forcément le même rôle à chaque représentation d'une même pièce. Une association porteuse de la rubrique « Rôle » doit donc être mise en œuvre entre les classes « Acteur » et « Représentation ».

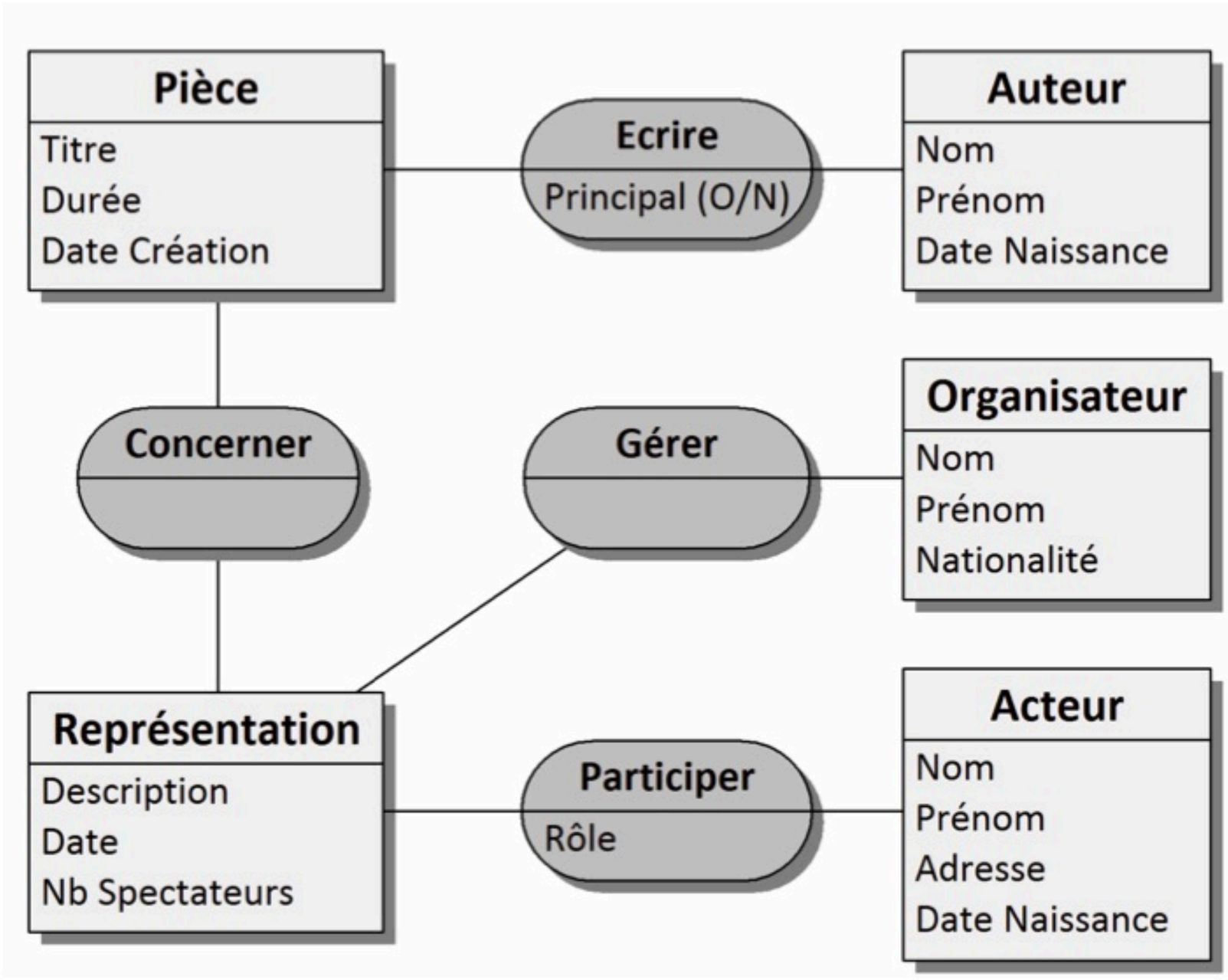
© Patrick Bergougnoux

- Pour la rubrique concernant l'auteur principal d'une pièce, un auteur peut être principal que sur aucune ou sur certaines des pièces qu'il a coécrites : par conséquent, cette rubrique booléenne doit se positionner dans une association reliant les classes d'entités « Pièce » et « Auteur ».

Sémantique : mise en œuvre des associations

Au-delà de ces deux associations, le compte-rendu d'interview fait ressortir d'autres liens entre les différentes classes du système d'information.

On note ainsi les rapports entre une pièce et ses représentations, elles-mêmes liées aux organisateurs et, comme vu précédemment, aux acteurs.



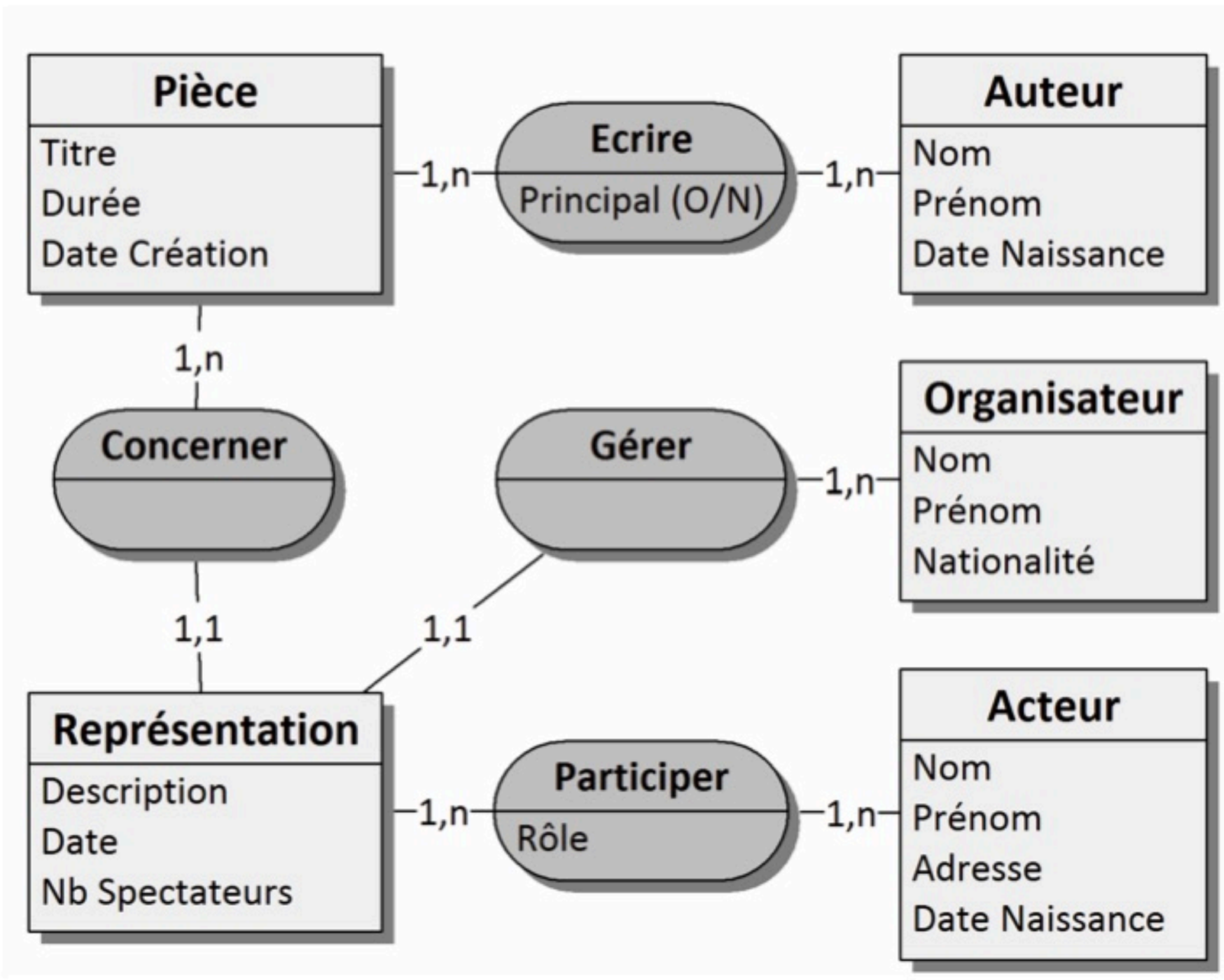
Modèle conceptuel de données

Détermination des cardinalités

On détermine aisément que les associations « Ecrire » et « Participer », porteuses de rubriques, possèdent des cardinalités multiples **1,n/1,n**.

Par ailleurs, une représentation ne concerne qu'une seule pièce et n'a qu'un seul organisateur : les deux associations correspondantes ont donc des cardinalités **1,1/1,n**.

Nous obtenons alors le modèle conceptuel de données suivant :



Vérification du modèle

Les deux associations porteuses de rubriques ont bien des cardinalités multiples de part et d'autre : la règle 4 est donc respectée, ainsi que les autres règles qui, dans ce modèle, ne posent aucun problème.

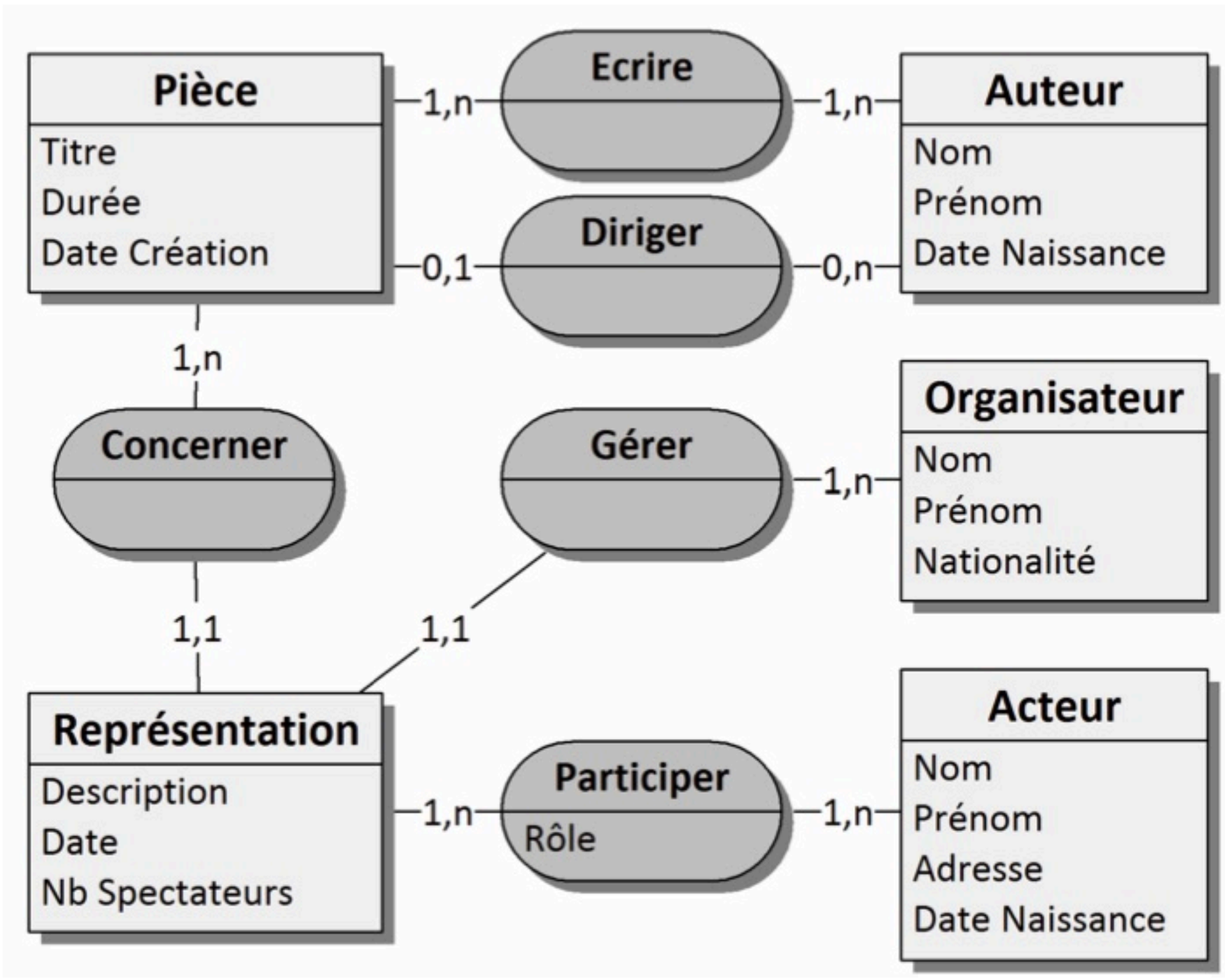
Mais ce modèle n'est pas parfait ! En effet, pour l'auteur principal dirigeant l'écriture de la pièce, la question de l'utilisation de la rubrique booléenne « Principal (O/N) » au sein de l'association « Ecrire » ne répond pas parfaitement au compte-rendu d'interview.

Le problème réside dans la sémantique et plus particulièrement dans les cardinalités : en effet, la valeur de cette rubrique peut très bien être « Oui » pour plusieurs coauteurs associés à une même pièce, ce qui va à l'encontre du compte-rendu d'interview qui spécifie : « … *l'**un** d'entre eux peut éventuellement être noté comme l'auteur principal* … ».

Il faut donc faire en sorte qu'un seul auteur (ou aucun) puisse diriger l'écriture d'une pièce, sachant qu'un auteur peut en diriger plusieurs (ou aucune).

© Patrick Bergougnoux

Pour modéliser cette contrainte, il est nécessaire de rajouter une association **0,1/0,n**. La rubrique « Principal (O/N) » n'est alors plus utile, la cardinalité **0,1** permettant de spécifier la présence ou non de l'auteur principal.



Ce modèle répond ainsi aux exigences du système d'information. On notera cependant qu'un auteur peut diriger une pièce qu'il n'a pas écrite : cette contrainte inter-associations sera traitée plus loin dans cet ouvrage en modélisation avancée *(cf. chapitre 15)*.

Ce que le modèle met en évidence

- Les rubriques concernant les âges doivent être calculées à partir des dates de naissance ; la problématique est la même pour toute rubrique en constante évolution dans le temps comme, par exemple, l'ancienneté dans une entreprise.
- L'alternative entre booléen et association supplémentaire est également mise en évidence ; c'est la contrainte liée aux cardinalités de l'association qui permet de décider de remplacer ou pas la rubrique booléenne par une nouvelle association.

Location de logements

Compte-rendu d'interview

Un organisme de vacances gère la location de logements et désire exploiter les informations recueillies durant l'année.

Chaque logement est identifié par son numéro et caractérisé par sa nature (villa, appartement ou gite rural). On connait également son niveau de confort et le nombre de personnes qui peuvent l'occuper simultanément. Il est situé dans une commune dont on connait le nom, la situation (mer, montagne ou campagne) et la date d'adhésion à l'organisme de location.

Chaque groupe d'occupants est caractérisé par le nom du contact dans le groupe, le nombre d'adultes et le nombre d'enfants qui le constituent, ainsi que par la région dont il est originaire. La location d'un logement à un groupe est définie par une date de début, un nombre de semaines et un loyer total calculé à partir d'un loyer hebdomadaire variable en fonction de nombreux paramètres non définis dans cette analyse.

Tableau des rubriques

Nous sommes à nouveau face à un compte-rendu d'interview dans lequel les rubriques sont indiquées de manière explicite et sont quasiment toutes primaires.

En effet, seul le loyer total est calculable à partir du loyer hebdomadaire et du nombre de semaines. Par ailleurs, le fait que le loyer hebdomadaire varie en fonction de paramètres non précisés dans le système d'information rend cette rubrique primaire, car non calculable à partir des éléments mis à notre disposition.

Même si aucun document n'est présenté en annexe, plusieurs exemples apparaissent dans le compte-rendu d'interview.

Attention à ne pas confondre un contenant avec un contenu, c'est-à-dire une rubrique avec une valeur que peut prendre cette rubrique.

Ainsi, « villa », « appartement » et « gite rural » ne sont pas des rubriques, mais des exemples de valeurs pour la rubrique « Nature logement ».

De même, « mer », « montagne » et « campagne » sont des exemples de « Situation commune ».

Le tableau des rubriques est donc le suivant :

© Patrick Bergougnoux

N°	Libellé	Exemple	P/C	Calcul
1	Numéro logement	42	P	
2	Nature logement	Villa	P	
3	Niveau confort logement	Excellent	P	
4	Capacité logement	10	P	
5	Nom commune	Narbonne-Plage	P	
6	Situation commune	Mer	P	
7	Date adhésion commune	14/01/2002	P	
8	Nom contact groupe	Untel	P	
9	Nombre adultes groupe	2	P	
10	Nombre enfants groupe	4	P	
11	Région origine groupe	Occitanie	P	
12	Date début location	07/07/2018	P	
13	Nombre semaines location	2	P	
~~14~~	~~Loyer total location~~	2 400 €	C	[13] x [15]
15	Loyer hebdomadaire location	1 200 €	P	

Dictionnaire des données

Il n'y a que la rubrique calculable « Loyer total location » à retirer pour obtenir le dictionnaire des données :

- Numéro logement
- Nature logement
- Niveau confort logement
- Capacité logement
- Nom commune
- Situation commune
- Date adhésion commune
- Nom contact groupe
- Nombre adultes groupe
- Nombre enfants groupe
- Région origine groupe
- Date début location
- Nombre semaines location
- Loyer hebdomadaire location

Répartition des données dans les classes d'entités

Première détermination des classes d'entités

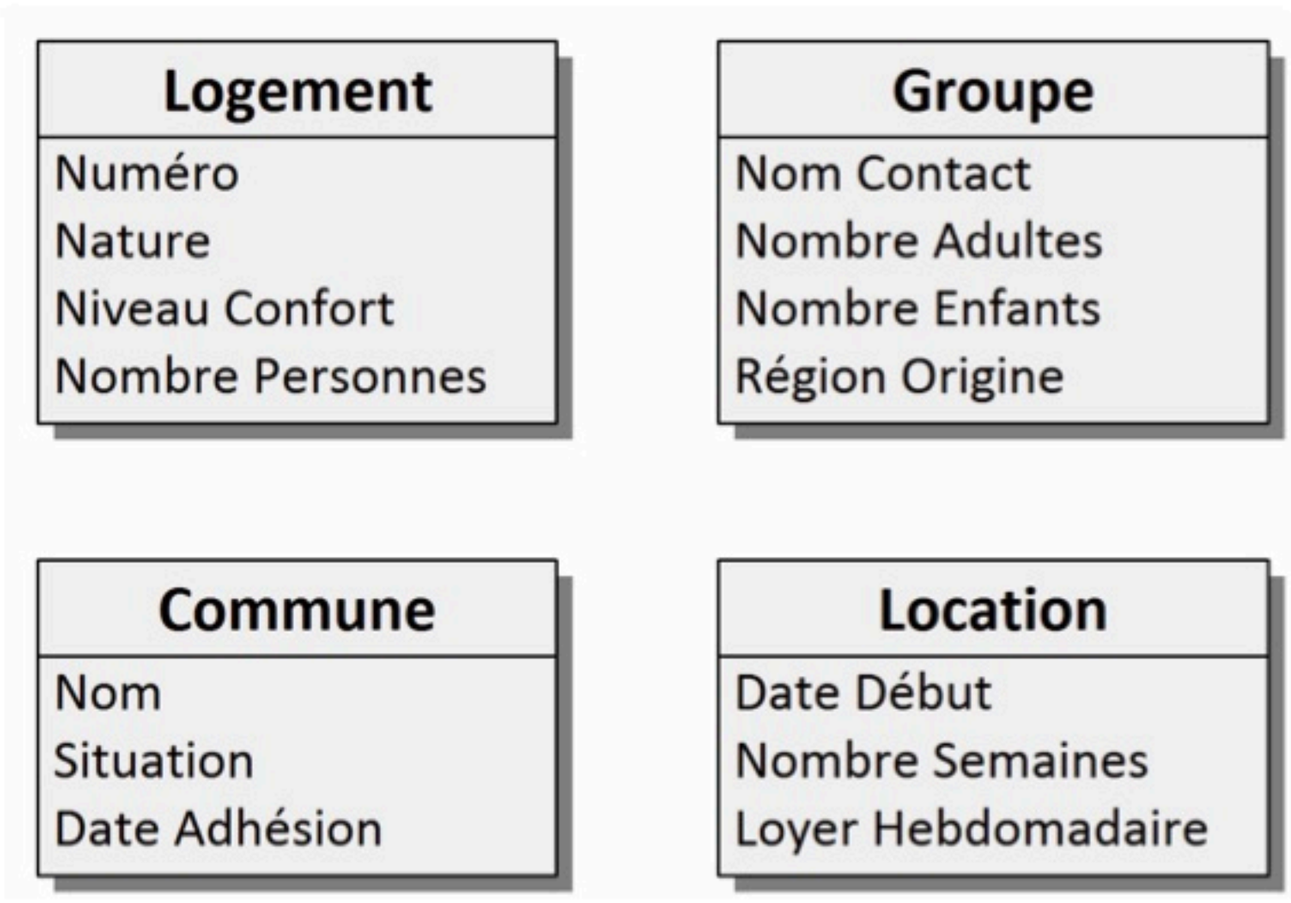
Suite à l'analyse du dictionnaire des données, quatre classes d'entités peuvent être définies :

- Logement
- Commune
- Groupe
- Location

Cette dernière classe d'entités peut ne pas être repérée au premier coup d'œil : nous verrons en fin d'exercice comment les informations qui la concernent peuvent alors être traitées.

Affectation des rubriques les plus évidentes

Dans ce système d'information, toutes les rubriques du dictionnaire trouvent naturellement leur place dans les quatre différentes classes d'entités proposées :



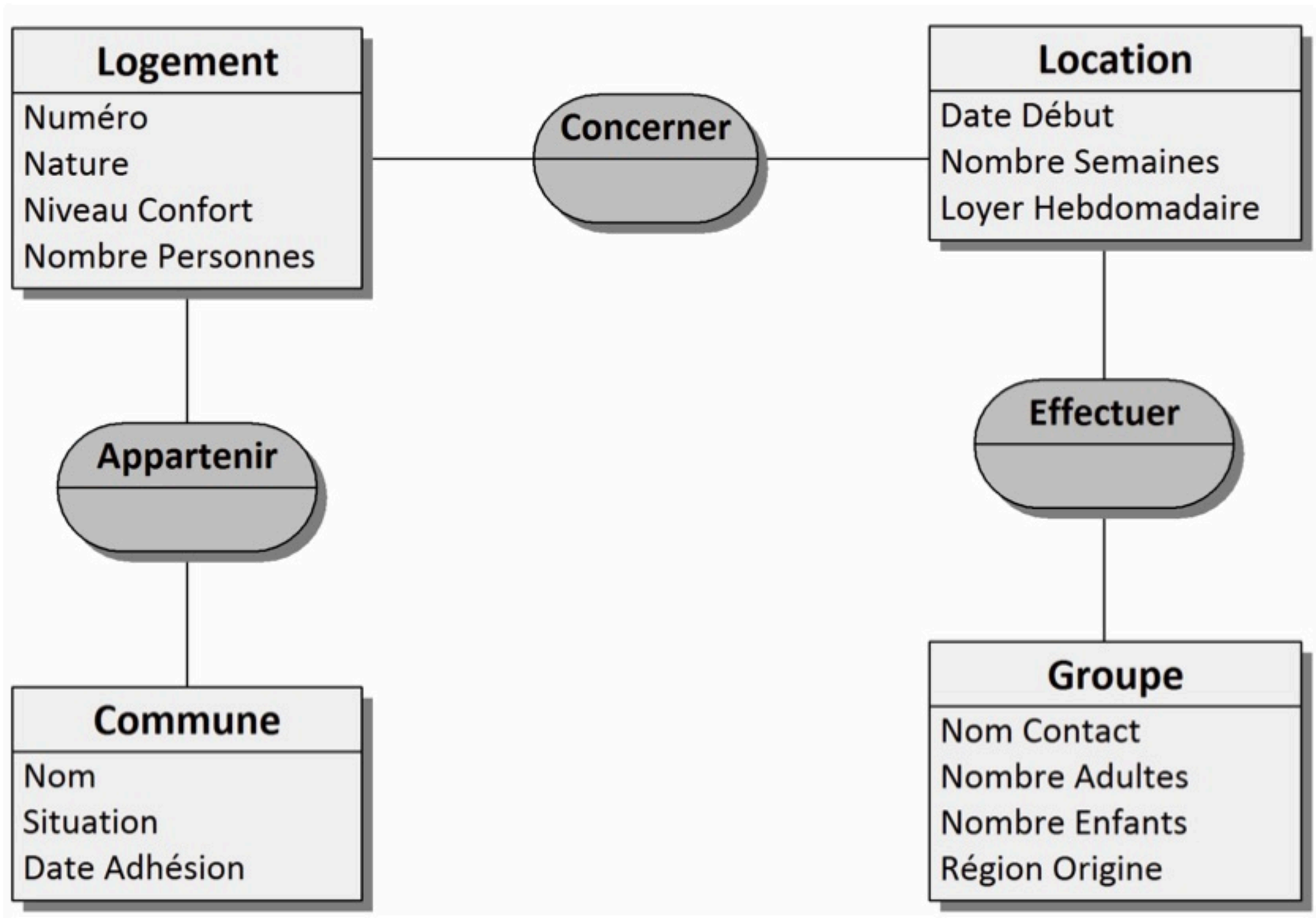
Sémantique : mise en œuvre des associations

Aucune rubrique ne restant à affecter, aucune association n'a été créée pour recevoir des rubriques orphelines.

© Patrick Bergougnoux

Toutes les associations vont donc devoir être rajoutées pour représenter la sémantique liée au fonctionnement du système d'information.

Ces associations sont directement déduites du fait qu'une location est liée à un groupe et à un logement, ce dernier étant rattaché à une commune.



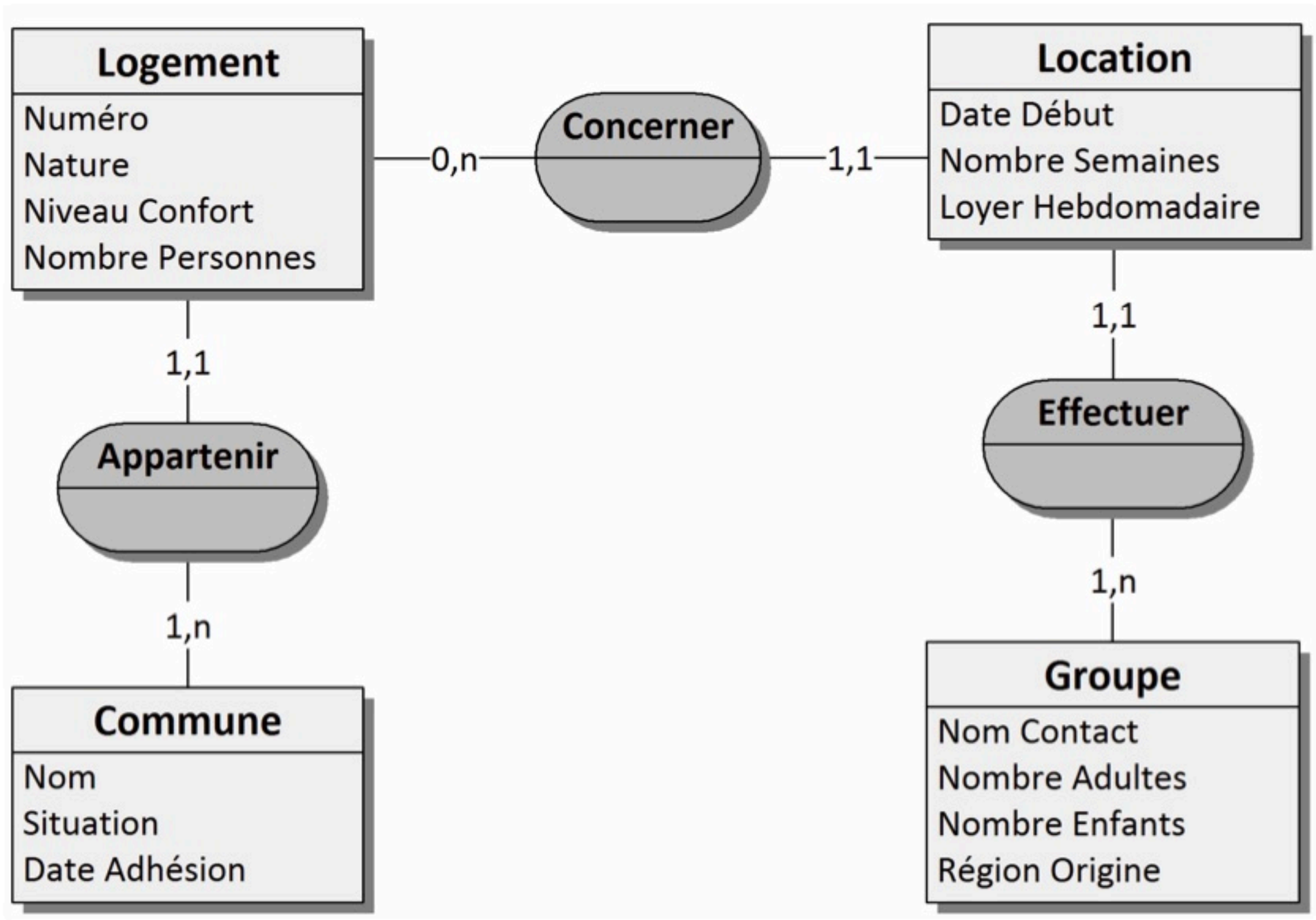
Modèle conceptuel de données

Détermination des cardinalités

Les cardinalités des trois associations de ce modèle se déterminent aisément :

- Chaque commune possède **1** à **n** logements.
- Chaque logement appartient à **1** et **1** seule commune.
- Chaque logement est concerné par **0** à **n** locations (un logement peut ne jamais être loué).
- Chaque location concerne **1** et **1** seul logement.
- Chaque location est effectuée par **1** et **1** seul groupe.
- Chaque groupe effectue **1** à **n** locations.

Voici le modèle conceptuel de données correspondant à ces cardinalités :



Vérification du modèle

Ce modèle conceptuel ne possédant que des associations avec une patte simple, il faut s'assurer du respect de la règle 4, à savoir l'absence de rubriques dans ces associations : c'est effectivement le cas, et il n'y donc pas de problème avec cette règle.

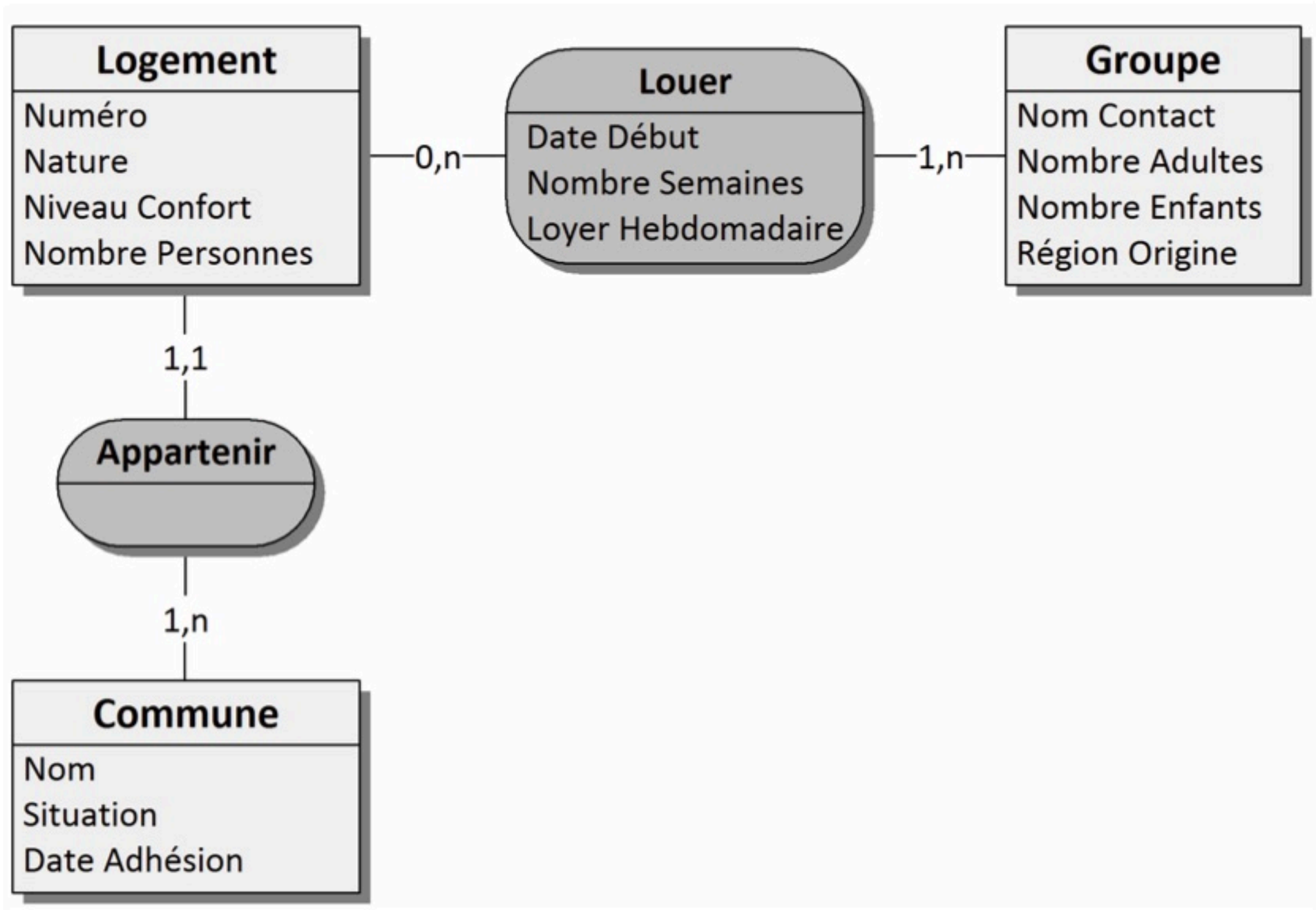
Cette règle 4 est vraiment importante à vérifier dans tous les modèles, en particulier ceux comportant des dates ; en effet, dans celui-ci, certains concepteurs auraient pu être tentés de placer la date de début de location dans l'association « Concerner » entre les classes d'entités « Logement » et « Location », considérant qu'une location concerne un logement à une date donnée… ce qui aurait été une erreur de conception *(cf. chapitre précédent)*.

Variante : la classe « Location » n'est pas vue dès le départ

Lors de la mise en œuvre de cet exercice, il arrive assez fréquemment que l'on ne voit pas « Location » comme une classe d'entités évidente, considérant que la location est plutôt ce qui lie un logement à un groupe.

© Patrick Bergougnoux

Cette approche est tout à fait valable et aboutit alors au modèle conceptuel suivant :



D'une façon générale, cette alternative sera présente chaque fois que toutes les pattes rattachées à une classe d'entités portent la cardinalité **1,1**.

Cette dernière version peut paraitre plus intéressante parce que plus compacte, mais nous verrons un peu plus loin dans cet ouvrage qu'elle posera des problèmes d'historisation des locations, et que nous reviendrons alors indirectement à une version basée sur la modélisation précédente.

Ce que le modèle met en évidence

- Au moment du recensement des rubriques, il ne faut pas confondre contenant (libellé de la rubrique) et contenu (exemple de valeur).
- Il ne faut pas mettre une rubrique dans une association avant d'avoir essayé de la placer dans les classes d'entités concernées.
- Une classe d'entités entourée uniquement de pattes **1,1** peut également se modéliser avec une association, mais ce n'est en aucun cas une obligation.

Agence de travail temporaire

Compte-rendu d'interview

Une entreprise est spécialisée dans la mise à disposition de personnel pour le compte de ses clients. Pour chaque intervention demandée, le client (caractérisé par sa raison sociale et son adresse) précise la description succincte de l'intervention, sa date de début, la qualification précise des intervenants nécessaires (il existe une vingtaine de qualifications standards avec un libellé et un tarif journalier spécifique à chacune d'elles), et le nombre de jours-homme prévu par qualification.

Chaque personnel pouvant être mis à disposition est caractérisé par sa civilité, son nom, son adresse, son âge et son ancienneté dans l'entreprise. D'autre part, l'entreprise s'accorde en interne une certaine souplesse dans la détermination du personnel qualifié pour une intervention en attribuant, à chacun, une qualification de base et, éventuellement, une ou deux qualifications secondaires.

Tableau des rubriques

Ce compte-rendu d'interview très condensé qui nous est proposé se limite quasiment à une énumération des différentes rubriques concernées par le système d'information.

Il suffit alors de rependre ces informations pour les placer dans le tableau des rubriques. Il ne faut pas pour autant en oublier les bases, à commencer par les « mauvaises rubriques primaires » que sont l'âge et l'ancienneté ; en effet, ces rubriques évoluent sans cesse dans le temps et doivent donc être ramenées à une information pérenne : leur calcul pourra être effectué à partir de la date du jour et des dates de naissance et d'embauche. Ces dates devront alors être rajoutées au tableau des rubriques, même si elles ne figurent pas de manière explicite dans le compte-rendu d'interview.

Un autre point doit attirer notre attention : il s'agit des qualifications des intervenants, et plus précisément le fait qu'elles soient de base ou secondaires. Une erreur peut en effet être commise en considérant deux rubriques différentes pour gérer cet aspect.

Une même qualification peut être de base pour certains intervenants et secondaire pour d'autres.

Le libellé de la qualification ne fait donc l'objet que d'une seule rubrique, et le fait que cette qualification soit de base ou secondaire représente une information à part, que nous traiterons dans un premier temps avec une nouvelle rubrique… avant de nous orienter vers une autre solution en fin d'exercice.

© Patrick Bergougnoux

N°	Libellé	Exemple	P/C	Calcul
1	Raison sociale client	Kilourien	P	
2	Adresse client	34 rue Ade - 31000 Tlse	P	
3	Description intervention	Carrelage sol entrepôts	P	
4	Date début intervention	04/03/2019	P	
5	Libellé qualification	Carreleur	P	
6	Tarif journalier qualification	185 €	P	
7	Nombre jours-homme	8	P	
8	Civilité personnel	Monsieur	P	
9	Nom personnel	Quelcun	P	
10	Adresse personnel	7 rue Stine - 31280 Mons	P	
~~11~~	~~Âge personnel~~	42 ans	C	[14] - [15]
~~12~~	~~Ancienneté personne~~	14 ans	C	[14] - [16]
13	Niveau qualification personnel	Secondaire	P	
~~14~~	~~Date du jour~~	25/05/2019	C	*Paramètre*
15	Date naissance personnel	15/02/1977	P	
16	Date embauche personnel	01/04/2005	P	

Dictionnaire des données

Une fois le paramètre de la date du jour et les deux rubriques calculables écartés, on obtient le dictionnaire des données suivant :

- Raison sociale client
- Adresse client
- Description intervention
- Date début intervention
- Libellé qualification
- Tarif journalier qualification
- Nombre jours-homme
- Civilité personnel
- Nom personnel
- Adresse personnel
- Niveau qualification personnel
- Date naissance personnel
- Date embauche personnel

Répartition des données dans les classes d'entités

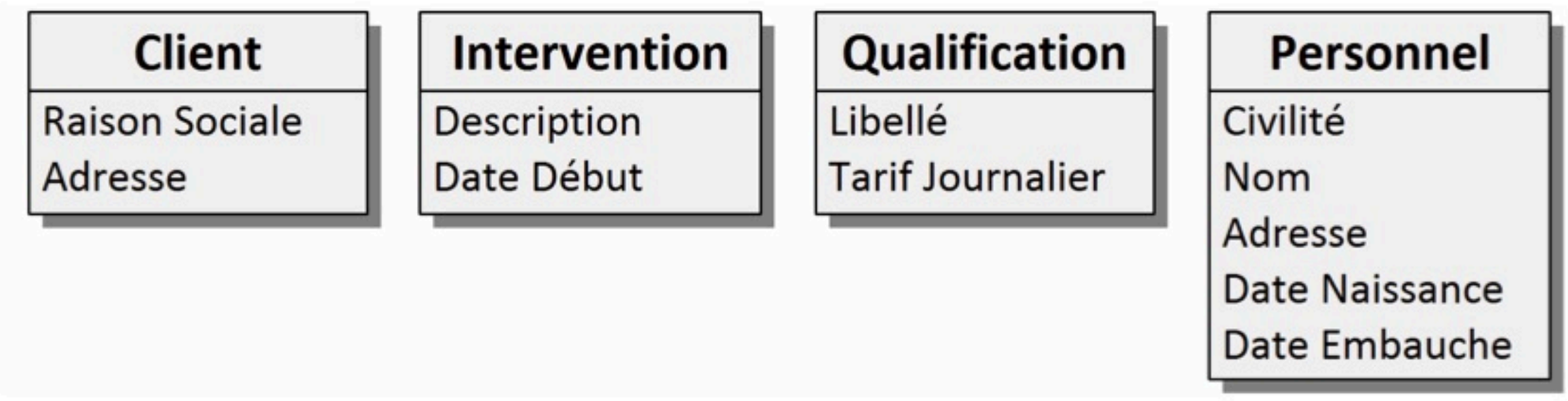
Première détermination des classes d'entités

Quatre classes d'entités ressortent de l'analyse du dictionnaire des données :

- Client
- Intervention
- Qualification
- Personnel

Affectation des rubriques les plus évidentes

Bon nombre de rubriques trouvent leur place dans ces quatre classes :



Affectation des rubriques restantes

Les rubriques qu'il reste à affecter sont :

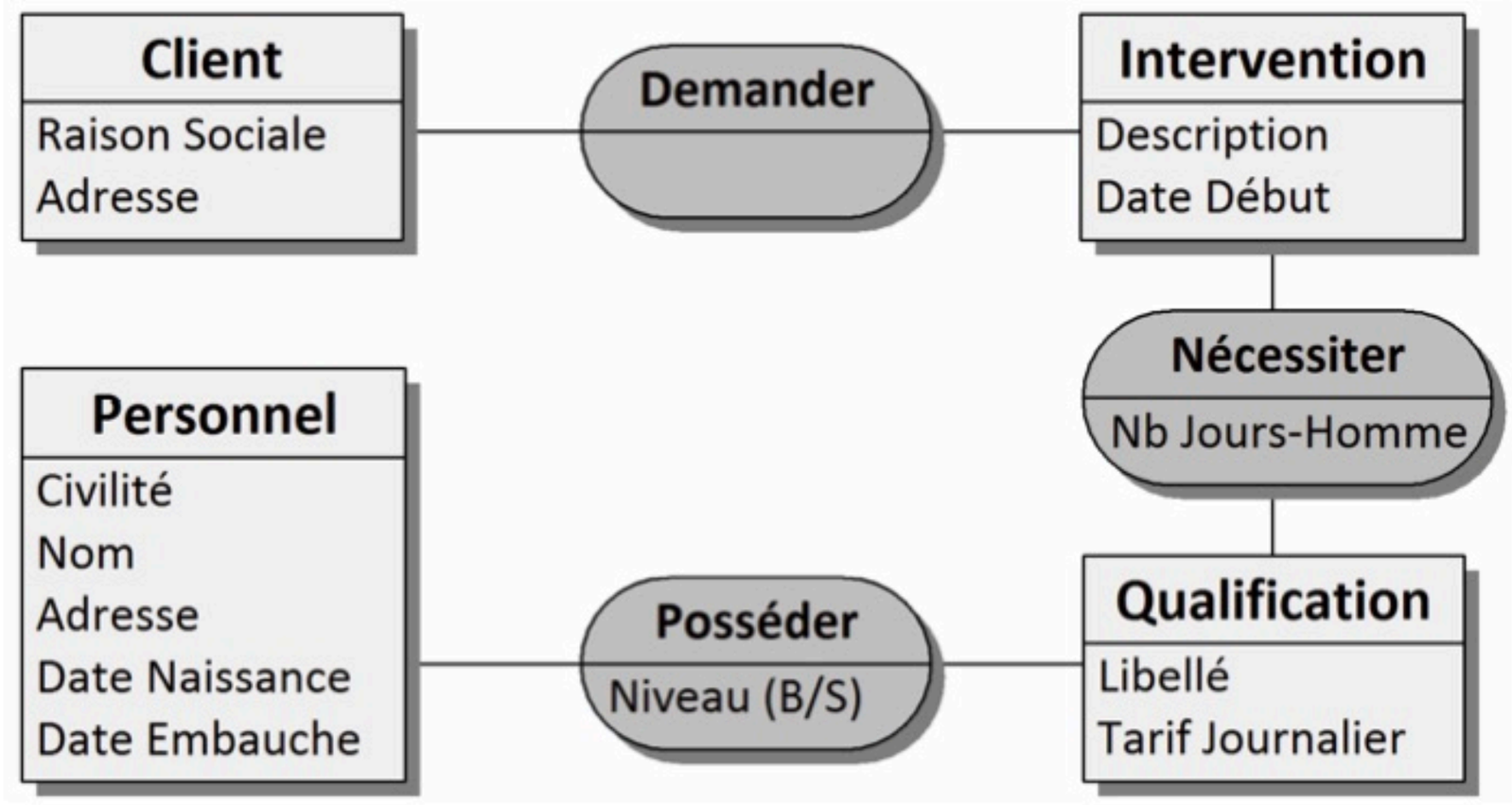
- Nombre jours-homme : une intervention nécessite diverses qualifications avec un certain nombre de jours pour chacune d'elles sachant que 3 jours à 2 personnes correspondant à 6 jours-homme ; par exemple, une intervention peut nécessiter 5 jours-homme de carrelage et 2 jours-homme de peinture. Cette rubrique caractérise une qualification donnée dans le cadre d'une intervention donnée : elle doit donc être affectée à une association reliant les classes d'entités « Intervention » et « Qualification ».
- Niveau qualification personnel : comme son nom l'indique, cette rubrique se place naturellement entre les classes d'entités « Qualification » et « Personnel » pour indiquer si une qualification d'un personnel donné est de base ou secondaire.

Sémantique : mise en œuvre des associations

Les deux premières associations sont mises en évidence grâce au placement des deux rubriques ci-dessus.

Une autre association s'impose également à la lecture du compte-rendu d'interview : en effet, ce sont les clients qui demandent les interventions, d'où l'association entre les classes d'entités « Client » et « Intervention ».

© Patrick Bergougnoux

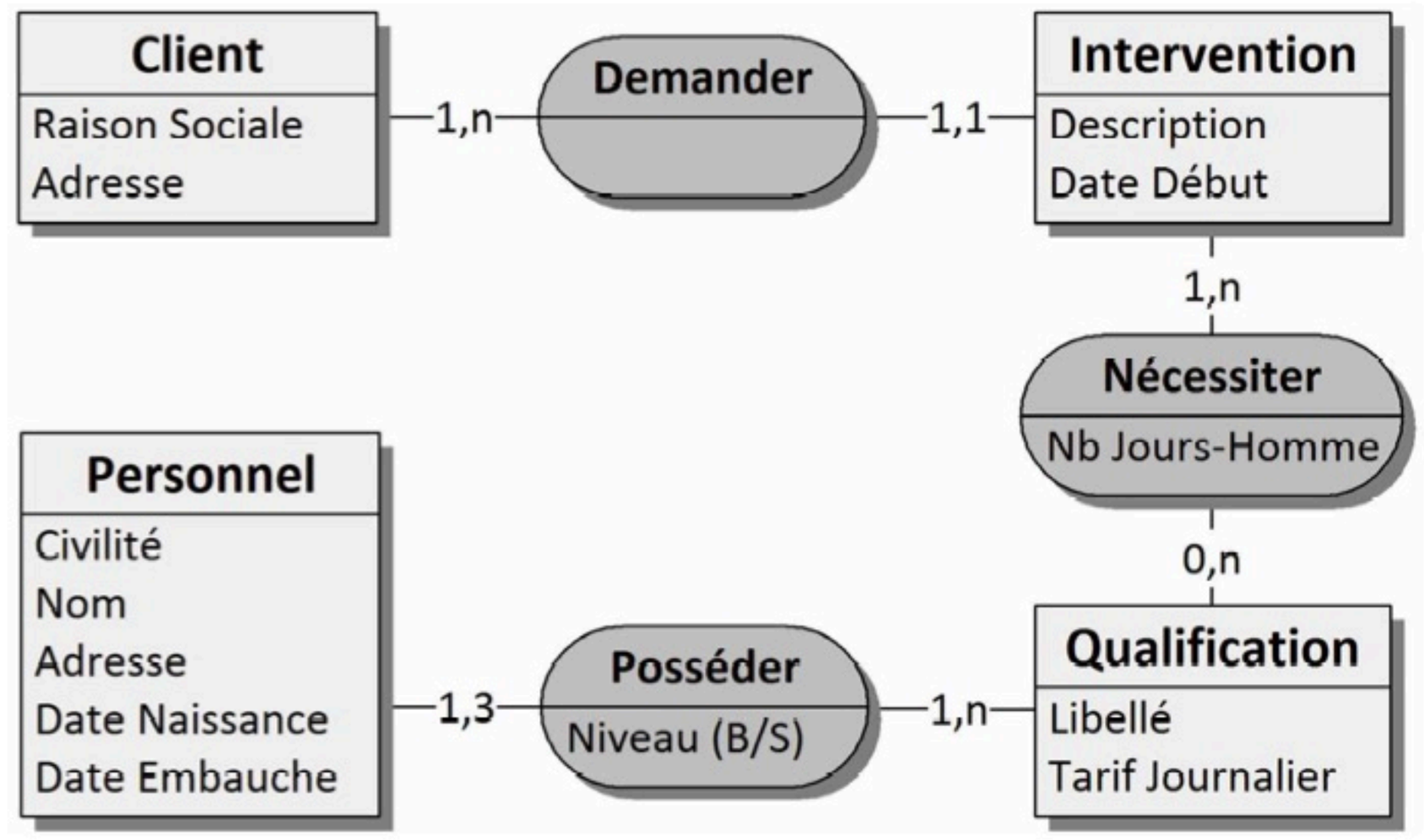


Modèle conceptuel de données

Détermination des cardinalités

Les cardinalités de ce modèle se définissent de la façon suivante :

- Chaque client demande **1** à **n** interventions.
- Chaque intervention est demandée par **1** et **1** seul client.
- Chaque intervention nécessite **1** à **n** qualifications.
- Chaque qualification est nécessitée par **0** à **n** interventions (une qualification peut ne jamais être nécessaire).
- Chaque qualification est possédée par **1** à **n** personnel.
- Chaque personnel possède **1** à **3** qualifications ➔ **1** de base et, éventuellement, **1** ou **2** secondaires (c'est-à-dire **0** à **2** secondaires).

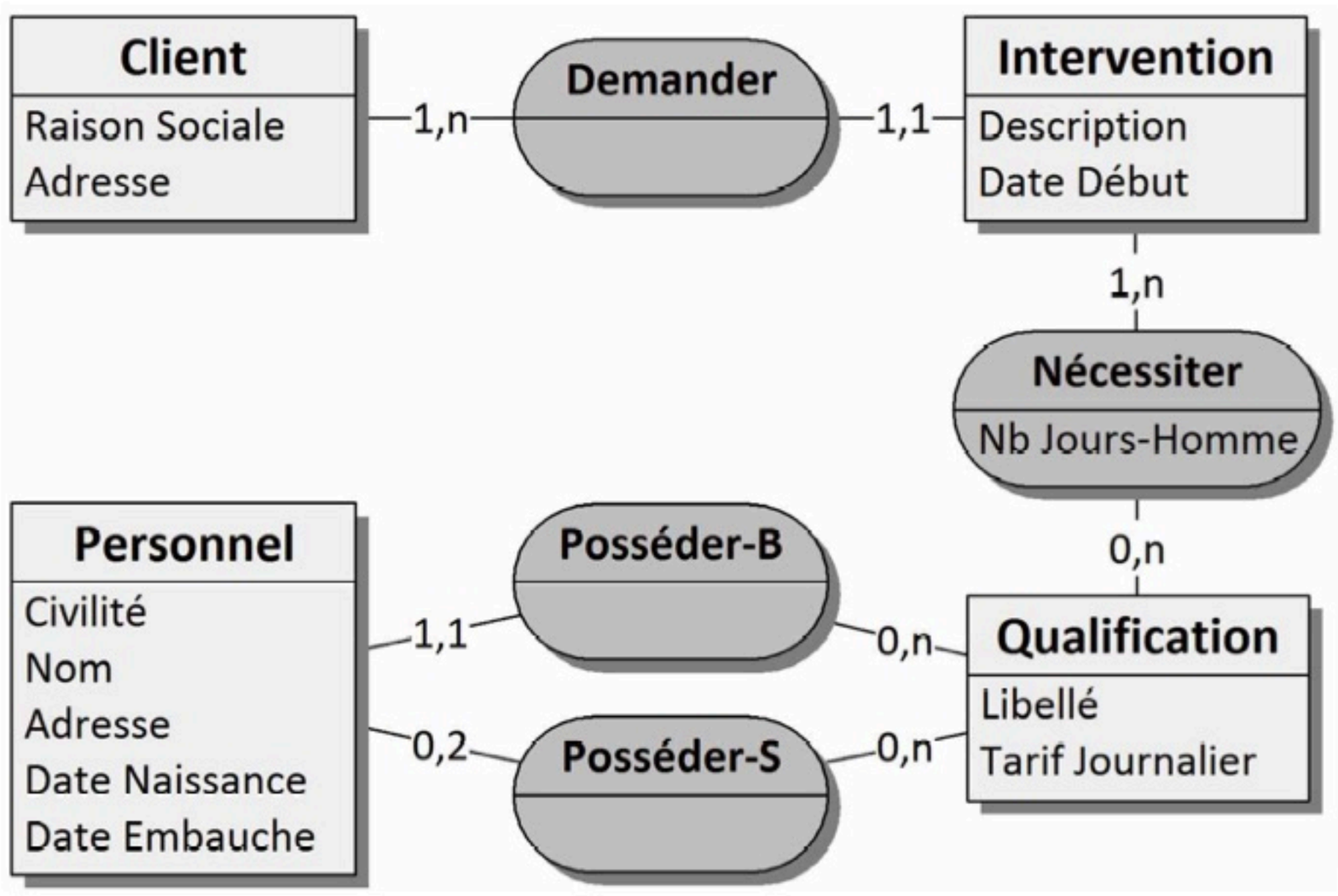


Vérification du modèle

L'ensemble des règles de cohérence sont respectées, en particulier la règle 2 avec des cardinalités multiples de part et d'autre pour les associations porteuses de rubriques.

La cardinalité **1,3** est bien multiple ; en fait, seules les cardinalités **1,1** et **0,1** sont considérées comme simples.

Cependant, ce modèle ne respecte pas entièrement les contraintes imposées par le compte-rendu d'interview. En effet, avec cette modélisation, un intervenant peut n'avoir aucune qualification de base et avoir jusqu'à trois qualifications secondaires ; il peut aussi posséder deux ou trois qualifications de base malgré qu'il soit bien précisé que chaque personnel possède *« une qualification de base et, éventuellement, une ou deux qualifications secondaires »*.
Il est donc nécessaire de créer deux associations avec des cardinalités spécifiques, chacune faisant respecter une partie de cette contrainte. La rubrique « Niveau (B/S) » n'est alors plus nécessaire :



Ce que le modèle met en évidence

- L'âge et l'ancienneté se calculent à partir de dates de référence.
- Une cardinalité multiple peut avoir des valeurs fixes (ex : **1,3** ou **0,2**).
- Le niveau de qualification s'avère être un élément sémantique traité, non pas avec une rubrique, mais avec deux associations reliant les mêmes classes d'entités avec des cardinalités appropriées.

© Patrick Bergougnoux

Gendarmes et voleurs

Compte-rendu d'interview

Pour lutter efficacement contre le grand banditisme, la police décide de rassembler les informations qu'elle a recueillies pendant ces trente dernières années. Ces informations concernent des bandits, leurs amis (eux-mêmes bandits) et les casses qu'ils ont effectués, un casse pouvant être réalisé par un ou plusieurs bandits. La police dispose actuellement des éléments suivants :

- *La liste des bandits avec leur nom, prénom, date de naissance et dernière adresse connue, qui ont participé à des casses avec leur rôle pour chacun de ces casses.*
- *Le nom et le matricule du commissaire responsable de l'enquête sur un casse donné, une enquête n'étant confiée qu'à un seul commissaire.*
- *Toutes les informations relatives aux casses commis par les bandits : description, ville, date et type du casse (braquage banque, attaque fourgon, vol à l'étalage, ...).*
- *La liste des types de casse avec leur libellé (cf. exemples ci-dessus) et la peine encourue (montant de l'amende et nombre d'années de prison).*
- *Les bandits amis d'un bandit donné : en effet, cette information est essentielle car, par exemple, si on sait où est Averell Dalton, Joe n'est surement pas loin !*

Tableau des rubriques

Après la lecture de ce compte-rendu d'interview, les rubriques composant le système d'information sont certes faciles à recenser, mais il convient tout de même de rester attentif à certains points :

- Ne jamais considérer une liste comme étant une rubrique : une liste, comme tout document, contient des rubriques.
- De même, la peine encourue n'est pas une rubrique mais est composée de deux rubriques : montant amende et nombre d'années de prison.

Par ailleurs, « Ami » peut être considéré au premier abord comme une rubrique.

Mais alors, quel exemple d'ami faut-il mettre dans le tableau des rubriques ?

On pourrait être tenté de donner comme exemple un nom de bandit… mais cela ferait de « Ami » un synonyme de « Nom bandit », rubrique déjà existante.

Une rubrique booléenne parait alors plus appropriée comme cela était le cas pour la spécialité dans le modèle du guide gastronomique.

Il y a également une autre solution qui consiste à considérer l'amitié, non pas comme une rubrique, mais plutôt comme un lien sémantique entre bandits : c'est cette approche que nous privilégierons pour ce modèle.

N°	Libellé	Exemple	P/C	Calcul
1	Nom bandit	Dalton	P	
2	Prénom bandit	Averell	P	
3	Date naissance bandit	01/04/1964	P	
4	Dernière adresse bandit	Oklahoma	P	
5	Rôle bandit	Ouvreur coffre	P	
6	Nom commissaire	Luke	P	
7	Matricule commissaire	777	P	
8	Description casse	Attaque de deux banques	P	
9	Ville casse	Coffeyville	P	
10	Date casse	23/05/1998	P	
11	Libellé type de casse	Braquage banque	P	
12	Montant amende	20 000 $	P	
13	Nombre années de prison	14	P	

Dictionnaire des données

Le tableau des rubriques ne contenant que des rubriques primaires, il est repris à l'identique pour constituer le dictionnaire des données :

- Nom bandit
- Prénom bandit
- Date naissance bandit
- Dernière adresse bandit
- Rôle bandit
- Nom commissaire
- Matricule commissaire
- Description casse
- Ville casse
- Date casse
- Libellé type de casse
- Montant amende
- Nombre années de prison

© Patrick Bergougnoux

Répartition des données dans les classes d'entités

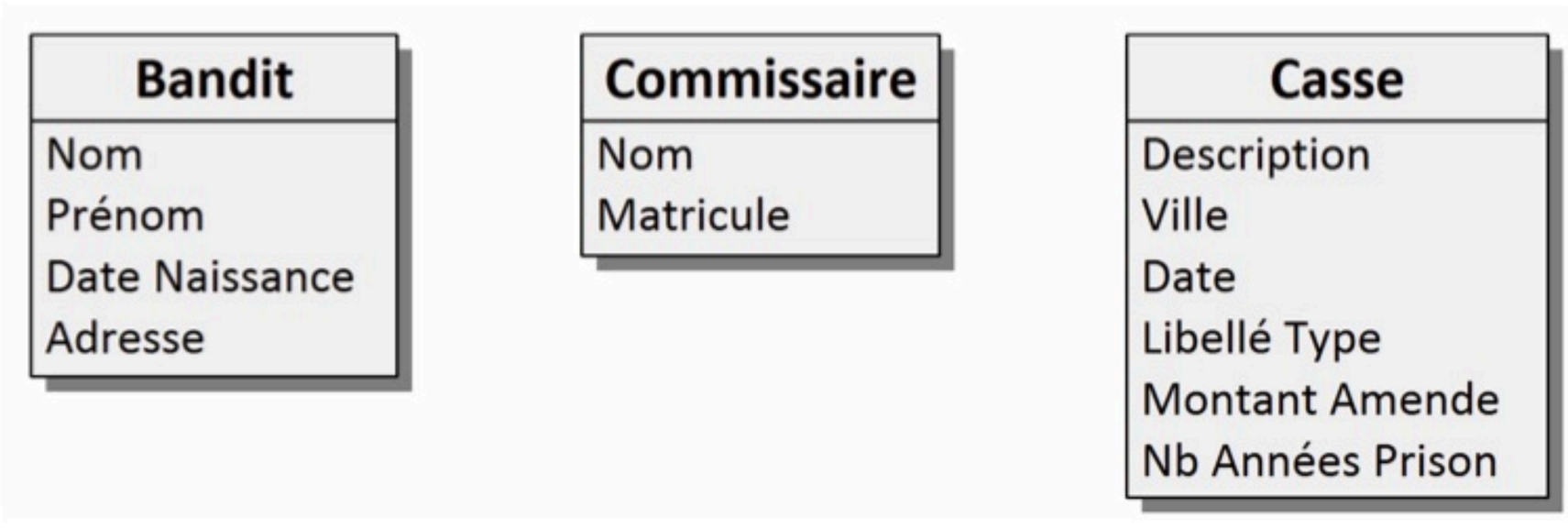
Première détermination des classes d'entités

Le dictionnaire des données met immédiatement en évidence trois classes d'entités :

- Bandit
- Commissaire
- Casse

Affectation des rubriques les plus évidentes

La plupart des rubriques du dictionnaire trouvent leur place dans les classes d'entités que nous venons de définir :



Les rubriques n'ont qu'une seule valeur pour chaque entité de ces trois classes : la règle 2 est donc bien respectée. Une fois ces affectations effectuées, il ne reste plus qu'une seule rubrique du dictionnaire des données à affecter.

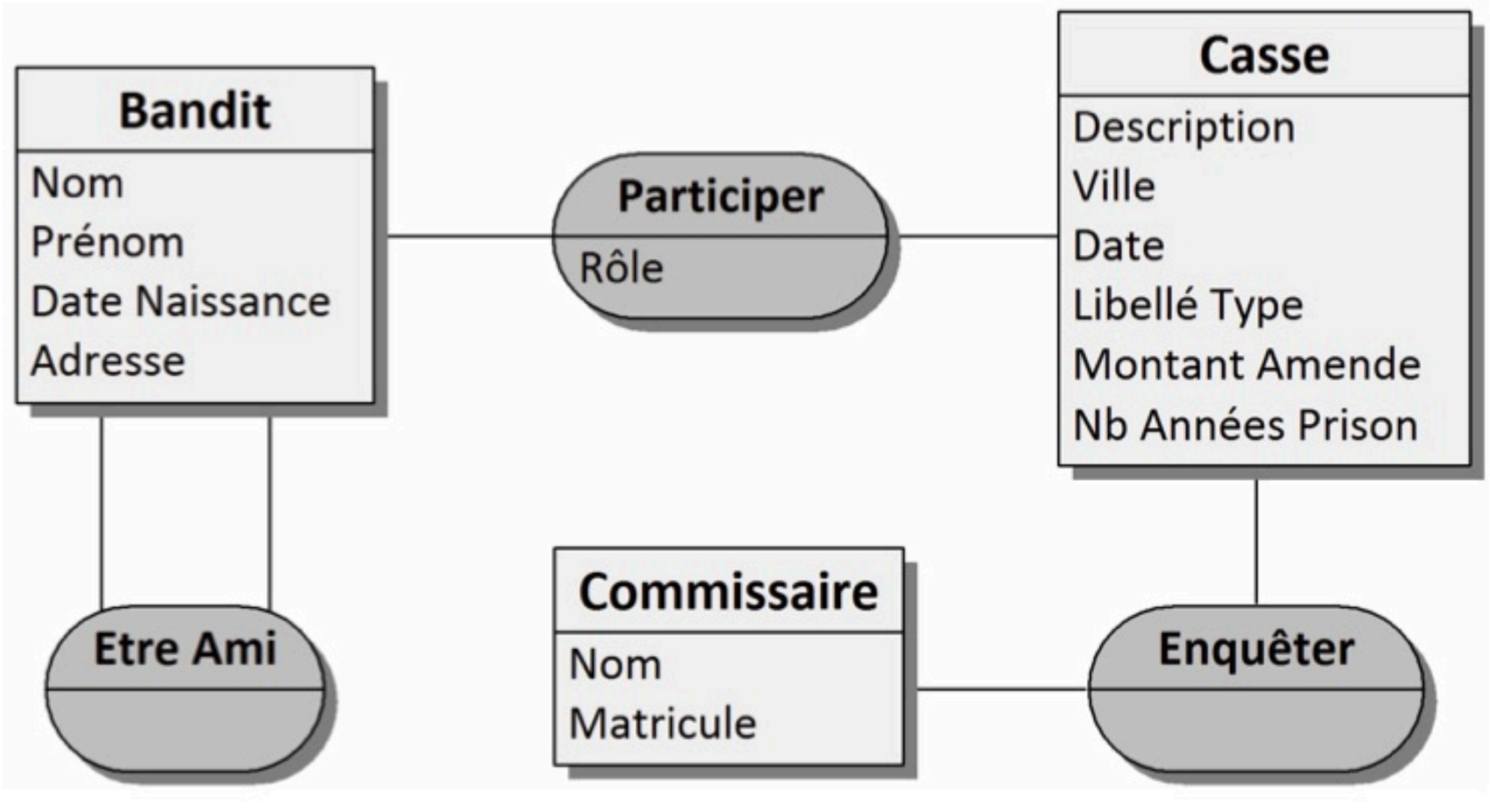
Affectation des rubriques restantes

La rubrique « Rôle » ne peut être affectée à un bandit, ce rôle pouvant varier en fonction des casses ; de même, différents rôles peuvent être nécessaires pour un même casse. Par contre, pour un casse donné et un bandit donné, il n'existe effectivement qu'un seul rôle. Par conséquent, la rubrique « Rôle » se trouve dans l'association entre les classes « Bandit » et « Casse ».

Sémantique : mise en œuvre des associations

En plus de cette première association, le lien entre les classes d'entités « Casse » et « Commissaire » ressort clairement de la lecture du compte-rendu d'interview.

Par ailleurs, nous avons vu précédemment que la notion d'ami n'était pas traitée par l'intermédiaire d'une rubrique, mais par un lien entre entités de la classe « Bandit » : il s'agit donc là d'une **association réflexive**.



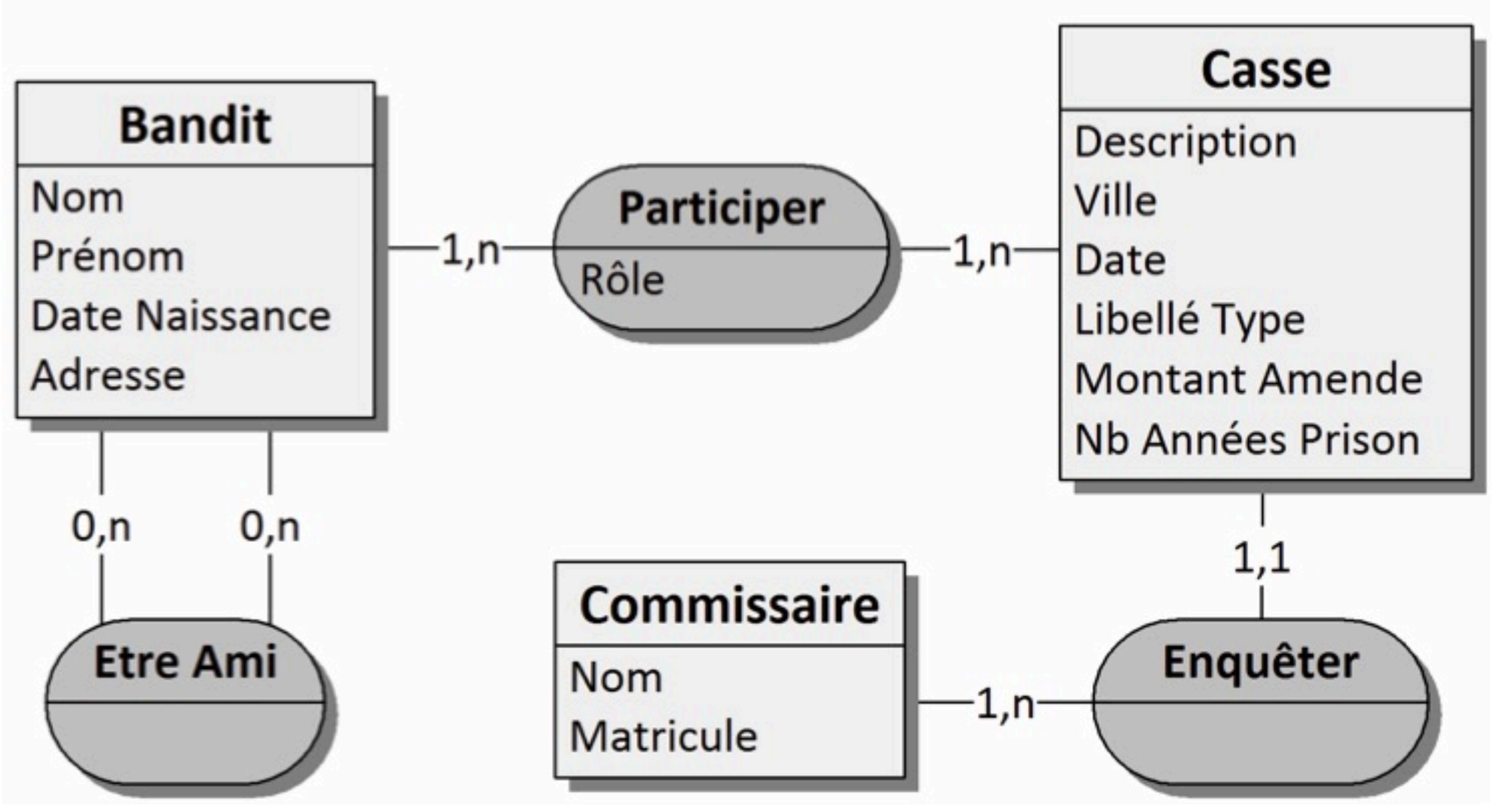
Modèle conceptuel de données

Détermination des cardinalités

Les cardinalités des deux associations entre classes sont les suivantes :

- Chaque bandit participe à **1** à **n** casses.
- Chaque casse voit participer **1** à **n** bandits.
- Chaque casse est enquêté par **1** et **1** seul commissaire.
- Chaque commissaire enquête sur **1** à **n** casses.

Concernant l'association « Etre Ami », chaque bandit peut être ami avec **0** à **n** autres bandits : il s'agit d'une **association réflexive symétrique** qui n'a donc pas besoin d'être complétée par des rôles au niveau de ses pattes.



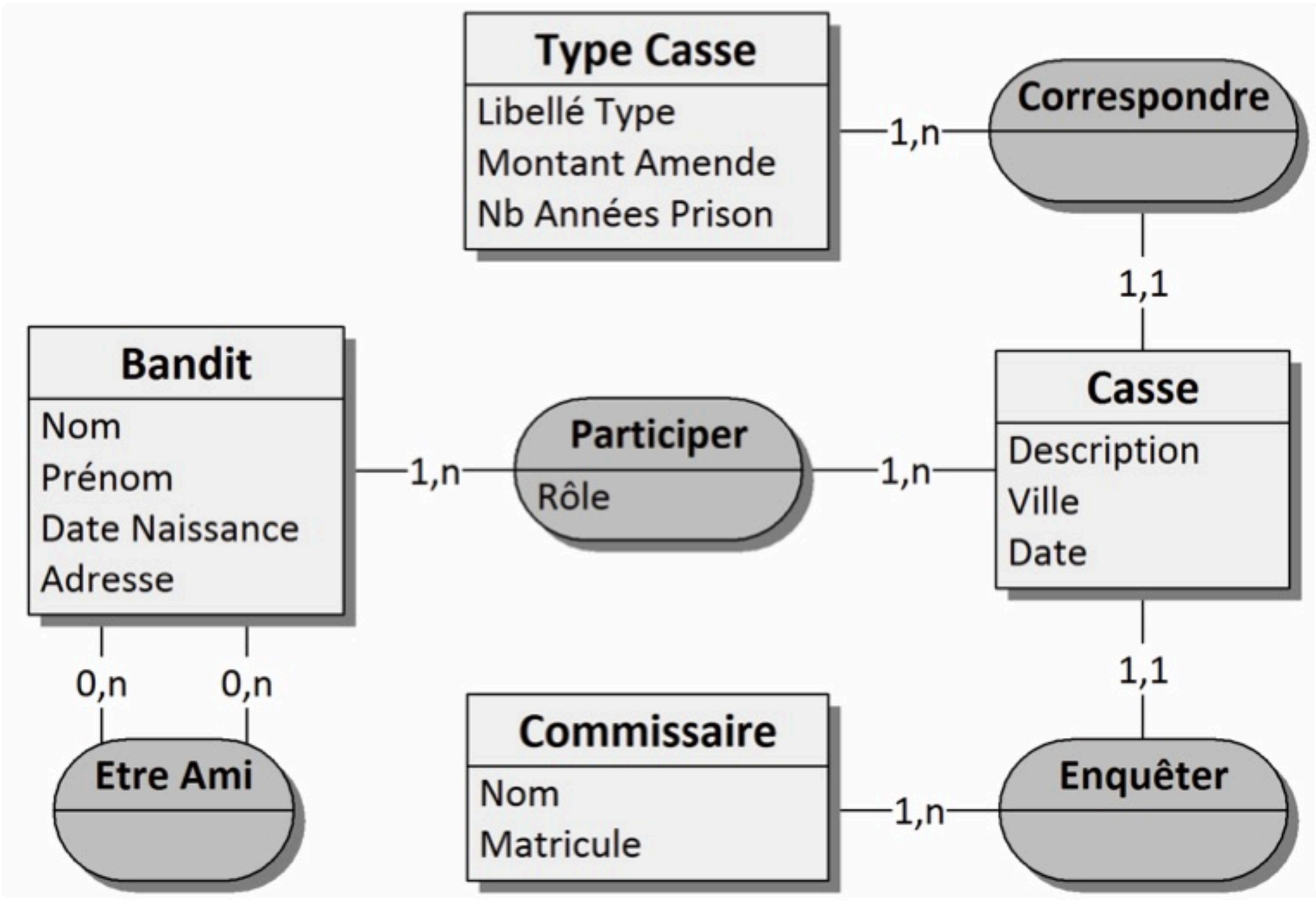
© Patrick Bergougnoux

Vérification du modèle

L'association « Participer » porteuse d'une rubrique a bien des cardinalités multiples de part et d'autre : la règle 4 est donc respectée.

La règle 3 doit, pour sa part, être vérifiée concernant la classe d'entités « Casse » ; en effet, plusieurs rubriques caractérisent le type du casse sachant que plusieurs casses peuvent être de même type. Dans ce cas, ces rubriques possèdent la même valeur pour plusieurs entités de la même classe et peuvent donc faire l'objet d'une classe à part.

On obtient alors le modèle conceptuel de données suivant :



Ce que le modèle met en évidence

- En voulant choisir un exemple d'ami, on donne un nom de bandit (rubrique déjà existante) : « Ami » n'est donc pas une nouvelle rubrique.
- Les informations relatives aux amis n'ont pas conduit au rajout d'une rubrique, mais à la mise en œuvre d'une association réflexive symétrique liant entre elles des entités de la classe « Bandit ».
- La classe d'entités « Type casse » qui n'avait pas été intégrée au modèle de départ a été rajoutée conformément à la règle 3.

Notes

Notes

Troisième partie

Informatisation des modèles

Cette troisième partie va nous réconcilier avec les spécialistes des bases de données ! Non pas que nous soyons fâchés, mais beaucoup se demandent surement comment des modèles conceptuels de données peuvent être produits sans jamais aborder les problèmes techniques liés à leur mise en œuvre pratique à travers un système de gestion de bases de données, et en particulier sans avoir défini d'identifiants pour les classes d'entités…

Notre approche conceptuelle, directement issue de l'analyse du système d'information, ne l'a jamais exigé et les modèles que nous avons produits n'ont pas souffert de ce manque. Il va falloir néanmoins nous en préoccuper à présent, avec les contraintes liées à l'automatisation de l'information contenue dans ces modèles.

Chapitre 9
Mise en relation des tables

Les modèles conceptuels de données mettent bien en évidence l'organisation des données du système d'information. Nous allons maintenant voir dans ce chapitre comment les données réelles sont physiquement stockées au sein d'un système informatique et comment les entités des différentes classes sont mises en relation les unes avec les autres.

Représentation des données

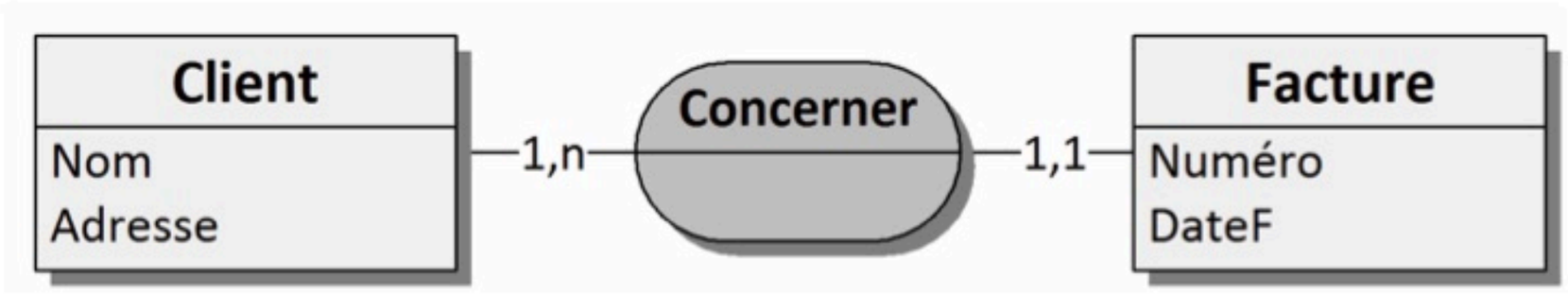
Format tabulaire

Le plus simple et le plus représentatif pour visualiser les données d'un système est de les exprimer sous forme de tableaux, avec un tableau par classe d'entités. Précisons qu'il s'agit là d'une simple visualisation des données qui ne présume en rien leur implémentation physique au sein d'une base de données.

On peut ainsi imaginer un fonctionnement de type tableur, où chaque classe fait l'objet d'une feuille de calcul (que nous appellerons « **table** ») au sein d'un classeur qui représente le système d'information.

Pour visualiser cette représentation, reprenons notre modèle de base, la facturation Beaux Rivages, en nous focalisant, dans un premier temps, sur une version simplifiée des classes d'entités « Client » et « Facture ».

Voici donc l'extrait du modèle conceptuel de données dans une version la plus épurée possible :



Prenons un petit jeu d'essai de données pour illustrer le contenu de ces deux classes d'entités stockées dans deux tables :

Table Client

Nom	Adresse
DUPONT	Toulouse
LOMBEC	Paris
MARTIN	Toulouse

Table Facture

Numéro	DateF
123	12/02/2019
234	14/02/2019
345	14/02/2019
456	20/02/2019
567	25/02/2019
678	25/02/2019
789	28/02/2019

Il est ainsi possible de stocker autant de données de clients et de factures que le système d'information le nécessite.

Que manque-t-il alors au système pour être complet ?

Les tables « Client » et « Facture » contiennent bien les bonnes informations, mais le lien entre les deux n'est pas précisé. En effet, si l'on considère par exemple la facture n°123, rien ne nous indique le client concerné par cette facture.

En fait, à ce stade, les tables représentent les données des différentes classes entités, mais n'intègrent pas les associations entre ces classes.

© Patrick Bergougnoux

Représentation graphique des liens

Comme nous l'indiquent les cardinalités du MCD, chaque facture doit être associée à un seul client, un client pouvant être concerné par une ou plusieurs factures. Cette association peut être représentée graphiquement de la façon suivante :

Table Client

Nom	Adresse
DUPONT	Toulouse
LOMBEC	Paris
MARTIN	Toulouse

Table Facture

Numéro	DateF
123	12/02/2019
234	14/02/2019
345	14/02/2019
456	20/02/2019
567	25/02/2019
678	25/02/2019
789	28/02/2019

Le problème est la représentation de ces liens dans un contexte de données informatisées : il n'est effectivement pas question de tirer des bouts de ficelle entre les différents éléments des fichiers composant la base de données !

De plus, si la représentation ci-dessus est à peu près lisible avec trois clients et sept factures, il n'en serait bien évidemment pas de même avec plusieurs centaines ou milliers de clients et de factures.

Référencement d'une entité au sein d'une autre classe

Création de nouvelles colonnes de liaison

La seule solution technique pour lier une entité d'une classe à une entité d'une autre classe consiste à spécifier, pour chaque élément d'une table, par quel élément il est concerné dans la table associée.

Pour cela, il est nécessaire de rajouter une colonne dans une table grâce à laquelle chaque entité pourra faire référence à une entité située dans la table associée.

Peut-on rajouter une colonne à la table « Client » pour référencer les factures concernées par chaque client ?

Sachant que chaque colonne correspond à une rubrique et qu'une rubrique ne peut contenir qu'une seule référence à la table « Facture », le problème est le nombre de colonnes qu'il faudrait rajouter à la table « Client » : d'après le MCD, il faudrait en rajouter entre **1** et … **n** ! En effet, chaque client est concerné par **1** à **n** factures, et « **n** » n'ayant pas de limite, autant dire que cela n'est pas envisageable.

Par contre, ce que nous dit également le MCD, c'est que chaque facture ne concerne que **1** et **1** seul client.

Par conséquent, en rajoutant à la table « Facture » une colonne faisant référence au client concerné par chaque facture, le lien pourra être parfaitement assuré.

Mais que mettre dans cette colonne de référencement ?

Table Client

Nom	Adresse
DUPONT	Toulouse
LOMBEC	Paris
MARTIN	Toulouse

Table Facture

Numéro	DateF	*Réf Client*
123	12/02/2019	?
234	14/02/2019	
345	14/02/2019	
456	20/02/2019	
567	25/02/2019	
678	25/02/2019	
789	28/02/2019	

- Le nom du client ?
  ➔ NON : plusieurs clients peuvent avoir le même nom.
- L'adresse du client ?
  ➔ Pourquoi pas si l'on considère que plusieurs clients n'habitent pas à la même adresse ; mais nous allons voir que ce n'est pas forcément une bonne idée…

Une entité étant susceptible d'être référencée dans une autre classe que la sienne, il est nécessaire de pouvoir identifier formellement chaque entité au sein de sa propre classe.

© Patrick Bergougnoux

Propriétés nécessaires aux identifiants des classes d'entités

Pour identifier une entité au sein de sa classe, l'identifiant en question doit posséder plusieurs propriétés. Certaines sont bien connues et évidentes, d'autres le sont moins.

Unicité

C'est bien évidemment la propriété la plus connue et celle dont on se contente bien souvent. Il faudra certes s'assurer en premier lieu du respect de cette propriété, plusieurs entités d'une même classe ne pouvant pas posséder le même identifiant.

Mais cette propriété d'unicité n'est pas la seule que doit respecter un identifiant.

Formalisme

Pour comprendre cette propriété, le plus simple est de faire une petite expérience : demandez à une dizaine de personnes d'écrire une adresse en la dictant oralement ; par exemple : « *123 avenue du Général de Gaulle, appartement B45 – 31234 Toulouse* ».

Vous aurez très probablement des résultats bien différents d'une personne à l'autre ; pour ma part, lorsque je réalise cette petite expérience en cours, j'obtiens des réponses du genre :

- 123 av Gén. De Gaulle, apt B45 31234 Tse
- 123 avenue De Gaulle – appart. B45 – 31234 Toulouse
- 123, AV. G. de Gaule, APPT B45, 31234 TOULOUSE

Sans parler des « *123 av. 2 goals…* » (pour les fans de football surement !), entre autres surprises orthographiques ! Ce qui est constant, c'est qu'en plus de vingt ans, je n'ai jamais obtenu deux résultats strictement identiques. De plus, un même étudiant écrit rarement la même chose au début et à la fin du cours !

Alors, comment pourrait-on envisager d'utiliser comme identifiant une rubrique qui ne soit pas écrite de manière identique par tous les utilisateurs, voire par un même utilisateur à des moments différents ?

Lorsque l'on parle de formalisme pour un identifiant, cela signifie que la rubrique candidate est écrite de la même manière par n'importe qui et à n'importe quel moment.

Toutes les rubriques composées de plusieurs mots, de noms trop longs, d'espaces, … sont généralement à proscrire en tant qu'identifiant.

On privilégiera donc essentiellement des codes, des références, des numéros, … qui peuvent sans problème être considérés comme des rubriques respectant un formalisme précis. D'ailleurs, les seuls éléments qui étaient identiques sur les adresses écrites par mes étudiants, étaient le numéro de rue (123), l'appartement (B45) et le code postal (31234).

Pérennité

Afin de garantir l'accès à une entité d'une table de manière pérenne, encore faut-il que l'identifiant soit stable dans le temps. En effet, si l'identifiant d'une entité évolue dans le temps, les liens établis avec lui depuis les autres tables seront mis à mal.

Un des exemples les plus significatifs est le numéro de téléphone pour identifier une personne : ce numéro est bien unique, formel (si l'on convient de ne pas y intégrer les espaces, les points ou les parenthèses), mais il est susceptible de changer. Il n'est donc pas pérenne et ne constitue pas, de ce fait, un bon identifiant.

Par contre, le numéro d'une facture et la référence d'un produit n'évolueront jamais et sont alors de bons identifiants pour leur classe respective.

Légalité

Il convient bien sûr de s'assurer que l'utilisation d'une rubrique en tant qu'identifiant est autorisée par la Loi. Cet aspect est particulièrement sensible lorsque cela concerne l'identification de personnes.

Alors que le numéro de sécurité sociale (NIR : Numéro d'Inscription au Répertoire) possède toutes les qualités vues précédemment pour un identifiant, peut-on pour autant l'utiliser pour identifier des employés au sein d'une entreprise ?

NON ! En effet, les règles de protection des données personnelles imposent que les informations enregistrées dans un fichier soient pertinentes, adéquates et non excessives au regard de l'utilisation qui en sera faite ; et, pour les fichiers concernant la gestion des ressources humaines, la CNIL (Commission Nationale de l'Informatique et des Libertés) considère que l'utilisation du numéro de sécurité sociale n'est pas justifiée et qu'elle ne répond à aucun impératif d'intérêt général.

Dans son avis, la CNIL rappelle « l'importance que l'utilisation du NIR, en tant qu'identifiant, soit cantonnée à la sphère sanitaire et médico-sociale et sera particulièrement vigilante à ce qu'aucun élargissement de cet usage n'intervienne à l'avenir ».

© Patrick Bergougnoux

En effet, le numéro de sécurité sociale ne fait pas partie de la liste des informations qui doivent figurer dans le registre unique de personnel, fixée par les articles L.620-3 et R.620-3 du Code du travail, et ne doit donc pas être enregistré dans ce cadre. Par conséquent, le numéro de sécurité sociale d'un employé ne peut pas être utilisé comme numéro unique pour l'identifier dans tous les fichiers de gestion de ressources humaines de son entreprise ou de son administration.

C'est la raison pour laquelle une rubrique spécifique du type « Matricule », interne à l'organisation, est systématiquement utilisée pour l'identification du personnel au sein des entreprises ou des administrations.

Cet exemple du numéro de sécurité sociale est le plus répandu en termes de respect de la légalité concernant la définition d'identifiants ; d'autres cas similaires peuvent se présenter dans des contextes spécifiques avec des règles définies dans le cadre de statuts ou de règlements intérieurs applicables au sein du domaine d'étude.

Irréductibilité (ou minimalité)

Il faut enfin s'assurer que l'identifiant choisi soit le plus simple possible, c'est-à-dire qu'il n'existe pas une alternative plus réduite à cet identifiant.

Cette dernière propriété sera étudiée de manière approfondie en modélisation avancée lorsqu'il sera question de s'assurer du respect de la notion de « clé minimale ».

Choix de l'identifiant

Un bon choix de rubriques permettant d'identifier sans ambigüité, en respectant les propriétés ci-dessus, chaque entité au sein de sa classe, est donc essentiel au bon fonctionnement de la future base de données.

Le processus de choix de ces identifiants se déroule en 3 étapes tout en respectant les priorités suivantes pour chacune des classes d'entités du modèle :

1. Existe-t-il une rubrique possédant toutes les propriétés nécessaires pour jouer le rôle d'identifiant ?

Si c'est le cas, on la sélectionne en la soulignant dans le MCD.

Si plusieurs choix sont possibles (ce qui n'est pas très fréquent), la meilleure option est toujours de choisir comme identifiant la rubrique possédant la structure la plus simple. Les autres rubriques candidates sont alors dotées de la propriété d'unicité, sans être définies comme identifiant.

2. *Si aucune rubrique ne possède à elle seule les qualités d'un identifiant, la concaténation de plusieurs rubriques peut-elle convenir ?*

Il s'agit alors de sélectionner plusieurs rubriques et de les concaténer avec, comme principal objectif, d'obtenir la première propriété d'un identifiant, à savoir **l'unicité**.

Le problème qui se pose souvent dans ce cas, c'est que plus on rajoute de rubriques, plus on tend vers l'unicité, mais plus on prend le risque d'un manque de formalisme.

Les rubriques qui composent un identifiant doivent impérativement posséder individuellement les qualités de formalisme, de pérennité et de légalité : leur concaténation peut alors permettre d'atteindre l'unicité.

Concernant l'irréductibilité, il faut bien sûr s'assurer que les rubriques composant l'identifiant soient bien toutes nécessaires.

Si une concaténation de rubriques peut ainsi assurer correctement le rôle d'identifiant, les rubriques concernées sont toutes soulignées dans le MCD.

3. *Si aucune rubrique ou groupe de rubriques ne peut faire office d'identifiant, que faire ?*

C'est une situation qui reste assez fréquente ; en effet, les rubriques qui ressortent naturellement du système d'information suite à l'analyse de l'existant, ne sont pas forcément prévues pour supporter les problématiques techniques liées à l'informatisation des données.

Compte-tenu de la nécessité absolue d'identifier chaque entité au sein de sa classe, il est alors indispensable de rajouter une rubrique capable de jouer ce rôle.

Cette nouvelle rubrique sera donc créée pour des raisons purement techniques, et n'appartient pas aux informations du domaine d'étude. Il est alors important de bien signaler la nature de cette rubrique, et nous verrons dans le prochain chapitre que le type de données attribué à cet identifiant sera très spécifique.

Le nom donné à un identifiant créé pour des raisons techniques doit clairement montrer qu'il s'agit d'une donnée spéciale ; nous proposons dans cet ouvrage de le désigner systématiquement de la façon suivante : « **Id_***NomClasse* ».

La création d'un identifiant spécifique de ce type ne doit en aucun cas être systématique ; en effet, cette donnée n'appartient pas au domaine d'étude et ne doit être rajoutée que lorsqu'aucun autre choix n'est possible.

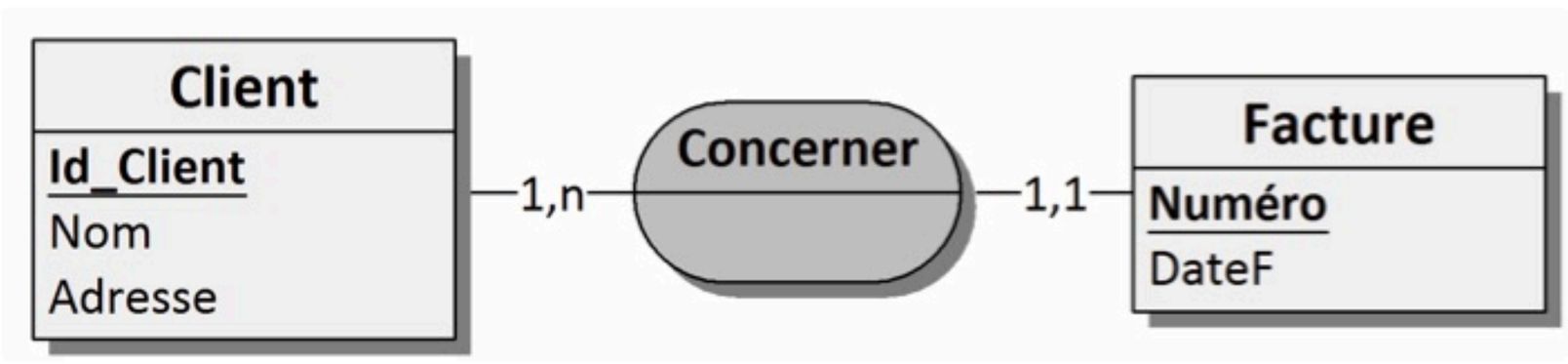
© Patrick Bergougnoux

Modèle conceptuel de données informatisé

Reprenons notre exemple du début de chapitre, et déterminons les identifiants des deux classes d'entités présentes :

- Pour la classe « Client », nous avons vu que ni le nom, ni l'adresse ne possèdent les qualités nécessaires à un identifiant : problème d'unicité, mais aussi de formalisme. Concaténer les deux ne fait qu'empirer le problème de formalisme ; par conséquent, il est nécessaire de créer une nouvelle rubrique spécifique pour jouer le rôle d'identifiant : pour bien en repérer la nature, nous lui affectons le libellé « **Id_Client** ».
- Pour la classe « Facture », c'est plus simple ; en effet, le numéro de facture possède toutes les propriétés d'un bon identifiant : il est donc inutile de chercher plus loin.

Nous obtenons alors le modèle suivant :



Afin de distinguer les MCD de base tels que nous les avons étudiés jusqu'à présent, et ceux complétés par les identifiants en vue de l'informatisation, nous parlerons de modèles conceptuels de données informatisés : **MCDi**.

La présence d'un identifiant dans chaque classe d'entités va ainsi permettre d'assurer les liaisons entre les tables.

Dès lors que l'on se situe au niveau physique de la base de données, où les tables viennent représenter les classes d'entités et les associations, de nouveaux termes sont également utilisés pour désigner les identifiants :

- Dans sa table d'origine, l'identifiant est appelé « **clé primaire** ».
- Lorsqu'il est référencé dans une autre table que la sienne, il est désigné sous le terme « **clé étrangère** » ; afin de mieux mettre en évidence ces clés étrangères, leur nom est précédé d'un « **#** ».

Ce « ***#*** » est au centre d'une anecdote amusante : en effet, cette notation est utilisée depuis plusieurs décennies pour les clés étrangères et est prononcée en cours : « ***dièse*** ».
Il y a quelques années, un étudiant corrige un exercice et m'annonce pour désigner la clé étrangère : « ***hashtag Id_Client*** » !
Après avoir révisé mes classiques concernant Facebook ou Instagram, j'ai constaté que nous avions, en fait, été des précurseurs avec notre historique « ***dièse Id_Client*** » qui, comme le « ***hashtag Id_Client*** » de nos djeuns, propose bien une référence à « **Id_Client** » !

Voici les tables complètes correspondant à notre exemple :

Table Client

Id_Client	Nom	Adresse
1	DUPONT	Toulouse
2	LOMBEC	Paris
3	MARTIN	Toulouse

Table Facture

Numéro	DateF	*#Id_Client*
123	12/02/2019	*1*
234	14/02/2019	*2*
345	14/02/2019	*1*
456	20/02/2019	*1*
567	25/02/2019	*3*
678	25/02/2019	*2*
789	28/02/2019	*3*

La connexion entre les tables est entièrement assurée par la relation qui existe entre les clés primaires et étrangères. C'est la base fondamentale de fonctionnement de tous les systèmes de gestion de bases de données relationnelles (SGBDR).

Une clé primaire est donc présente dans les tables associées sous forme de clés étrangères : son nom doit alors être représentatif et unique, non seulement au sein de sa classe d'origine, mais aussi pour l'ensemble du système d'information.

Les clés étrangères, bien que présentes dans les tables, ne doivent pas être inscrites dans les MCDi ; en effet, la représentation des relations entre les classes d'entités est assurée dans les modèles conceptuels par les associations et leurs cardinalités.

Table de correspondance

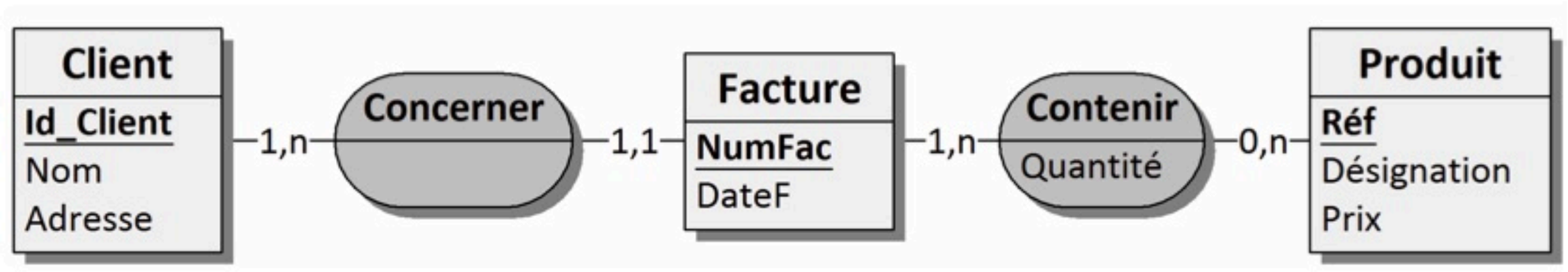
Limites du référencement direct d'une table vers une autre

Reprenons notre exemple simplifié de facturation en y rajoutant la table « Produit » pour laquelle la rubrique « Réf » représente un excellent identifiant.

Par ailleurs, afin de respecter la représentativité du nom des clés primaires au sein de l'ensemble du système d'information, le libellé du numéro de facture « Numéro », trop vague, sera remplacé par un libellé plus significatif : « **NumFac** ».

© Patrick Bergougnoux

Voici le modèle correspondant :



Complétons maintenant le jeu d'essai avec une représentation des tables « Facture » et « Produit » :

Table Facture

NumFac	DateF	#Id_Client
123	12/02/2019	*1*
234	14/02/2019	*2*
345	14/02/2019	*1*
456	20/02/2019	*1*
567	25/02/2019	*3*
678	25/02/2019	*2*
789	28/02/2019	*3*

Table Produit

Réf	Désignation	Prix
A12	Fauteuil	95,00
A45	Banquette	45,00
B23	Chaise	15,80
B34	Tabouret	13,50
B43	Banc	58,40
C55	Transat	79,90
C77	Coussin	12,50
D89	Bar roulant	85,00

La problématique pour associer ces deux tables est la même que précédemment, sauf que deux problèmes se posent :

- L'association « Contenir » ne possèdent que des cardinalités multiples de part et d'autre : **1,n** et **0,n**.
- La rubrique « Quantité » n'est pas représentée dans les tables.

Intéressons-nous au premier problème : rajouter une colonne dans une des deux tables est impossible ; en effet, que ce soit dans la table « Facture » (où chaque facture peut contenir **1** à **n** produits), ou dans la table « Produit » (où chaque produit peut être contenu dans **0** à **n** factures), il faudrait rajouter **n** colonnes…

Le référencement direct d'une table vers une autre n'est pas envisageable si l'association qui les lie possède des cardinalités multiples de part et d'autre.

Nécessité de rajouter une table de correspondance

La difficulté est de pouvoir associer un élément d'une table avec plusieurs éléments d'une autre table… et réciproquement.

Dans notre exemple, voici ce que cela pourrait donner graphiquement :

Table Facture

NumFac	DateF	*#Id_Client*
123	12/02/2019	*1*
234	14/02/2019	*2*
345	14/02/2019	*1*
456	20/02/2019	*1*
567	25/02/2019	*3*
678	25/02/2019	*2*
789	28/02/2019	*3*

Table Produit

Réf	Désignation	Prix
A12	Fauteuil	95,00
A45	Banquette	45,00
B23	Chaise	15,80
B34	Tabouret	13,50
B43	Banc	58,40
C55	Transat	79,90
C77	Coussin	12,50
D89	Bar roulant	85,00

Chaque lien devra également porter la valeur de la rubrique « Quantité ».

La seule solution consiste à créer une table représentant ces liens : cette table de correspondance devra référencer chacune des deux tables associées grâce à leur clé primaire, et sera complétée par les éventuelles rubriques portées par l'association.

Les clés primaires ainsi référencées seront donc des clés étrangères dans la table d'association. Cette table portera le nom de l'association dans le MCDi (« Contenir » dans notre exemple) et fonctionnera exactement de la même manière que les tables déduites des classes d'entités.

Comme toutes les tables de la base de données, une table de correspondance doit posséder sa clé primaire : quelles sont les rubriques pouvant rentrer dans sa composition ?

Tout d'abord, de quelles rubriques disposons-nous ?

- La clé étrangère « *#NumFac* » issue de la table « Facture ».
- La clé étrangère « *#Réf* » issue de la table « Produit ».
- La rubrique « Quantité » présente dans l'association.

Isolée, aucune de ces rubriques ne garantit l'unicité. Par contre, en concaténant les deux clés étrangères, on obtient bien cette propriété d'unicité.

© Patrick Bergougnoux

De plus, les clés étrangères font référence à des clés primaires et ont alors forcément les autres qualités nécessaires : formalisme, pérennité et légalité. En général, l'irréductibilité est également assurée, mais cet aspect sera étudié de manière plus approfondie dans la quatrième partie de cet ouvrage.

Nous obtenons donc la table de correspondance suivante :

Table Contenir

#NumFac	#Réf	Quantité
123	A12	1
123	B23	4
234	A12	8
234	A45	2
234	C55	4
345	B34	5
456	B43	2
456	C55	3
567	B23	12
567	D89	1
678	D89	2
789	C55	4

Toutes les associations à cardinalités multiples de part et d'autre, **qu'elles possèdent ou non des rubriques**, font l'objet d'une table de correspondance dont la clé primaire est la concaténation des clés étrangères issues des tables associées. **Une cardinalité est considérée comme multiple dès qu'elle est différente de 0,1 ou 1,1** (une cardinalité telle que **1,4** est donc considérée comme multiple).

Ainsi, toutes les associations définies dans un MCDi peuvent être traduites au niveau physique grâce aux liens entre les clés primaires de chaque table et leur référencement par les clés étrangères présentes dans les tables associées, soit directement, soit par l'intermédiaire de tables de correspondance.

Le dilemme de la cardinalité 0,1

En cas de cardinalité **0,1** d'un côté et multiple de l'autre, le principe de la clé étrangère dans la table rattachée à la patte simple fonctionne ; mais, dans le cas du « **0** », sa valeur ne sera pas renseignée… et ça, certains puristes en bases de données ne le supportent pas ! Il est vrai qu'elle met à mal la logique binaire en engendrant une logique à trois états (*true*, *false*, *null*) qui peut mettre en difficulté certains systèmes de gestion de bases de données relationnelles.

Pour contourner le problème, une table de correspondance peut être créée pour assurer la liaison entre les tables seulement lorsque cela s'avère nécessaire.

La valeur nulle n'est cependant pas une « malédiction », et il est également légitime de la préférer au rajout d'une table alourdissant la structure de données et les requêtes qui vont avec. Nous privilégierons donc cette solution.

En cas de cardinalités **0,1/1,1** la clé étrangère est placée dans la table rattachée à la patte **1,1**. Dans le cas d'une association **0,1/0,1** il suffit de choisir une des deux tables pour y placer la clé étrangère référençant l'autre table.

Le doute sur les cardinalités 1,1 et 1,n

En pratique, au sein de la base de données implémentant notre exemple de facturation, un client peut être créé juste avant d'être associé à sa première facture ; de même, l'entité facture sera physiquement créée avant que les produits qu'elle contient ne lui soient affectés, ou même que le client soit sélectionné. Ce type de réflexion sur la future implémentation d'une base de données et de l'application associée peut conduire le concepteur à choisir des cardinalités **0,n** au lieu de **1,n** et même des cardinalités **0,1** au lieu de **1,1**.

Si techniquement cela peut se comprendre, conceptuellement c'est une erreur ! En effet, les états transitoires et temporaires de la base de données ne doivent pas influencer le modèle conceptuel qui représente, lui, l'état stable du système d'information.

Ces situations où la base de données est momentanément dans un état incohérent doivent se régler techniquement au moment de l'implémentation dans le SGBD grâce à la gestion des contraintes et à l'utilisation de triggers.

Ce qu'il faut retenir

- Dans un MCDi, chaque classe d'entités doit posséder un identifiant permettant d'identifier chaque entité au sein de sa classe de manière unique, formelle, pérenne, légale et irréductible.
- Au niveau physique, les classes d'entités font l'objet de tables ; leur identifiant est appelé clé primaire.
- Une association avec au moins une cardinalité simple est assurée physiquement par une clé étrangère référençant, au sein de la table rattachée à la patte simple, la clé primaire de la table associée.
- Pour une association à cardinalités multiples de part et d'autre, une table de correspondance est créée avec un référencement des clés primaires des tables associées ; cette table est complétée par les éventuelles rubriques portées par l'association, et sa clé primaire est la concaténation des clés étrangères issues des tables associées.

© Patrick Bergougnoux

Chapitre 10
Modèle logique de données

Pour la description technique du modèle conceptuel de données, la représentation tabulaire que nous venons de voir n'est bien évidemment pas envisageable : seule la structure de données doit être présentée, sans les exemples.

On parle alors de schéma relationnel de base de données ou de modèle logique de données (MLD).

Représentation de la structure de données

Modèle graphique

La représentation de ce modèle logique de données peut prendre plusieurs formes dont la plus fréquemment rencontrée est un modèle graphique proche du MCDi.

Dans cette représentation, les tables prennent la place des classes d'entités, et les associations à cardinalités multiples de part et d'autre sont remplacées par des tables de correspondance.

Les clés étrangères, qui ne sont jamais représentées au niveau MCDi, font leur apparition dans le MLD.

Les tables sont graphiquement reliées entre elles au sein du MLD par des flèches indiquant les dépendances fonctionnelles, et les clés étrangères sont explicitement inscrites dans les tables concernées.

De nombreux logiciels ont opté pour ce type de représentation graphique ; c'est le cas, par exemple, de Win'Design qui, pour notre exemple de facturation Beaux Rivages, génère le MLD suivant :

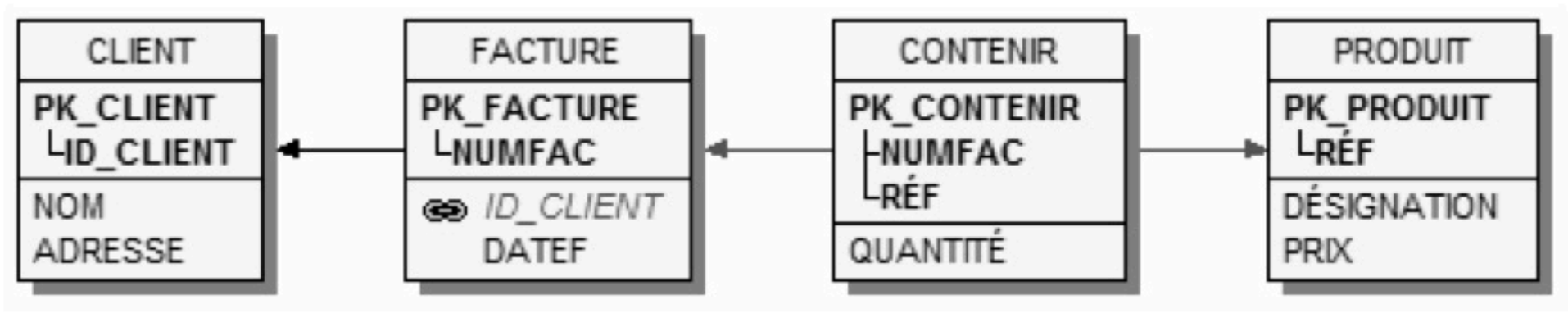


Schéma relationnel textuel

Dans cet ouvrage, nous privilégierons une approche peut-être moins esthétique, mais bien plus proche de la réalité. En effet, les liens graphiques utilisés n'existent pas physiquement : seuls les liens logiques entre clés primaires et clés étrangères assurent les associations entre les différentes tables de la base de données.

C'est pourquoi une approche textuelle parait plus appropriée. Cette notation consiste alors à reprendre la structure des tables en mettant en évidence les clés étrangères avec le préfixe « # » comme nous l'avons déjà fait dans la représentation tabulaire. En fait, il suffit de reprendre les en-têtes des tables définies au chapitre précédent en remplaçant les caractères spéciaux et les blancs par le caractère « _ ».

Ordre de déclaration des tables

Afin d'être tout à fait cohérent par rapport au chapitre suivant qui utilisera ce MLD pour générer les instructions de création des tables de la base de données, il convient de respecter un certain ordre dans la déclaration des tables.

Il ne faut pas faire référence, via des clés étrangères, à une clé primaire appartenant à une table qui n'a pas encore été déclarée.

Dans notre exemple, la table « Client » doit être déclarée avant la table « Facture » qui fait référence à « Id_Client ». De même, la table de correspondance « Contenir » doit être déclarée après les tables associées « Facture » et « Produit ».

© Patrick Bergougnoux

Voici donc le modèle logique de données sous forme de schéma relationnel textuel que nous obtenons pour la version simplifiée de Beaux Rivages :

Client = (**Id_Client**, Nom, Adresse)
Facture = (**NumFac**, DateF, #*Id_Client*)
Produit = (**Réf**, Désignation, Prix)
Contenir = (***#NumFac, #Réf***, Quantité)

Utilisation visuelle du « # »

Le préfixe « # » est utilisé pour bien repérer les clés étrangères dans le MLD. Ce signe sera ensuite supprimé au moment de générer la structure physique de la base de données. La clé primaire et les clés étrangères associées auront alors exactement le même libellé.

Remarque concernant le libellé des clés étrangères

Cette notation confirme l'importance d'avoir un libellé de clé représentatif et unique au sein du système d'information : en effet, un lien entre tables devient évident dès lors que le libellé de la clé étrangère correspond au libellé de la clé primaire à laquelle elle fait référence ; et ce libellé doit avoir du sens quelle que soit la table dans laquelle il se trouve.

Par contre, on constate que beaucoup de concepteurs, au moment de la définition du schéma relationnel de leur base de données, utilisent pour une clé étrangère un libellé différent de celui de la clé primaire associée… comme si la rubrique devait changer de nom en fonction de l'endroit où elle se trouve…

Cette pratique ne parait pas appropriée : en effet, pourquoi une rubrique devrait-elle changer de nom en fonction de l'endroit où elle se trouve ? Changez-vous de nom lorsque vous allez au travail ou en vacances par rapport au nom que vous portez chez vous ?

Une clé étrangère fait référence à la même entité que la clé primaire associée et contient donc la même valeur : il est alors naturel que son nom soit le même ! **Il est néanmoins possible de rajouter des rôles sur les pattes des associations pour compléter ce nom.**

Bien souvent, les schémas relationnels dans lesquels les clés étrangères ont des noms différents des clés primaires associées, ont été définis directement, sans passer par l'étape conceptuelle.

Un compromis acceptable peut consister à rajouter aux libellés des clés étrangères un préfixe qui viendra se mettre à la place du « # » dans le schéma relationnel de la base de données (par exemple : « *FK_NumFac* »)… mais nous ne le ferons pas dans cet ouvrage !

Introduction des types de données

La mise en œuvre du MLD est une étape technique qui doit permettre la génération de la structure de la base de données correspondant au MCDi. Il convient donc, à ce niveau, de définir avec précision la nature des données utilisées. Pour cela, leur type doit être précisé dès le MCDi, de manière à ce que les étapes suivantes, concernant le MLD et les requêtes SQL de génération du schéma relationnel, puissent être entièrement automatiques.

Des types variés et différents en fonction des SGBD

De nombreux types de données sont proposés par les différents systèmes de gestion de bases de données du marché. Le tableau ci-dessous propose une synthèse non exhaustive des types les plus utilisés, sur la base de ceux proposés dans « MS-Access » :

TEXTE	VARCHAR(*n*)	Chaine alphanumérique à longueur variable de 1 à 255 caractères
	CHAR(*n*)	Chaine alphanumérique à longueur fixe de 1 à 255 caractères (complétée par des blancs si inférieure à la taille maximale)
	TEXT	Zone de texte jusqu'à 65 536 caractères
NUMÉRIQUE	BYTE	Entier codé sur 1 octet (0 à 255)
	SMALLINT	Entier codé sur 2 octets (-32 768 à +32 767)
	INT	Entier codé sur 4 octets (-2 147 483 648 à +2 147 483 647)
	DECIMAL(*n*,*d*)	Nombre décimal de « *n* » chiffres dont « *d* » après la virgule
	REAL	Nombre réel codé sur 4 octets (1 bit de signe, 8 bits pour l'exposant et 23 bits pour la mantisse)
	DOUBLE	Nombre réel codé sur 8 octets (1 bit de signe, 11 bits pour l'exposant et 52 bits pour la mantisse)
	CURRENCY	Valeur monétaire codée sur 8 octets comprenant 15 chiffres + 4 décimales
	COUNTER	Type INT avec une auto-incrémentation à chaque nouvelle entité
DATE/HEURE	DATE	Date codée sur 4 octets
	TIME	Heure codée sur 8 octets
	DATETIME	Date et heure codées sur 12 octets
BOOLÉEN	LOGICAL	Valeur « Vrai » ou « Faux » codée sur 1 octet

© Patrick Bergougnoux

Ces types peuvent varier de manière assez significative en fonction des SGBD, tant par leur dénomination que par la façon de coder l'information. Il convient donc d'être vigilant au moment de définir les types de données, le plus sûr étant de rester le plus standard possible.

Voici un tableau de correspondance de cette sélection de types pour d'autres SGBD du marché :

Access	MySQL	Oracle	PostgreSQL
Varchar(*n*)	Varchar(*n*)	Varchar2(*n*)	Varchar(*n*)
Char(*n*)	Char(*n*)	Char(*n*)	Char(*n*)
Text	Text	Clob	Text
Byte	Tinyint	Number(3)	Smallint
Smallint	Smallint	Number(5)	Smallint
Int	Int	Number(10)	Integer
Decimal(*n*,*d*)	Decimal(*n*,*d*)	Number(*n*,*d*)	Numeric(*n*,*d*)
Real	Float	Binary_Float	Real
Double	Double	Binary_Double	Double Precision
Currency	Decimal(19,4)	Number(19,4)	Money
Counter	Int Auto_Increment	Number(10)	Serial
Date	Date	Date	Date
Time	Time	Date	Time
Datetime	Datetime	Date	Timestamp
Logical	Tinyint(1)	Number(1)	Boolean

Comme on peut le constater dans ce tableau, « Access » propose une palette assez précise et complète des principaux types de données : ce sont donc ces types « Access » que nous utiliserons dans la suite de cet ouvrage.

Le logiciel ***Looping*** propose la génération des tables de la base de données pour les différents SGBD présents dans le tableau ci-dessus. Il suffit pour cela de sélectionner le SGBD souhaité dans les options générales des propriétés du modèle. Par ailleurs, un type « Libre » permet de spécifier n'importe quel autre type de données.

Usage spécifique du type « COUNTER »

Comme nous l'avons vu précédemment, il est parfois nécessaire de rajouter une rubrique spécifique pour assurer le rôle d'identifiant au sein de certaines classes d'entités lorsqu'aucune autre rubrique ou groupe de rubriques n'a les qualités pour le faire. C'est le cas, dans notre exemple, de la classe « Client » pour laquelle nous avons dû rajouter un identifiant « Id_Client ».

La particularité de cette rubrique, au-delà de ses propriétés d'identifiant, est qu'elle n'appartient pas au système d'information initial et qu'elle a dû être rajoutée au système informatique pour des raisons purement techniques.

Vu que cet identifiant a été rajouté pour des raisons techniques, il parait légitime que la technique se charge de le gérer !

C'est pourquoi, le type préconisé pour ces identifiants est « COUNTER », qui est en fait un type « INT » (nombre entier codé sur 4 octets) ayant la particularité de s'incrémenter automatiquement d'une unité à chaque nouvelle création d'entités dans la table où il se trouve (« COUNTER » est le type utilisé par « Access » : voir le tableau de la page précédente pour les autres SGBD).

Au niveau de l'interface avec l'utilisateur, il est alors recommandé de laisser le SGBD gérer cette incrémentation de manière masquée, en n'affichant pas cette donnée qui n'a rien à voir avec le système d'information naturel.

Attention au type attribué aux clés étrangères associées à une clé primaire « COUNTER » : il doit être compatible (« INT »), mais ne doit en aucun cas être aussi de type « COUNTER » ; sinon, le lien ne serait pas assuré par l'utilisateur, mais quasi-aléatoirement par le SGBD.

Interdiction de la valeur « NULL »

Il est possible d'imposer qu'une rubrique possède obligatoirement une valeur non nulle (le nom du client par exemple). C'est en particulier systématiquement le cas pour les clés primaires qui doivent, par définition, posséder une valeur pour identifier chaque entité au sein de leur classe ; **une clé primaire étant forcément non nulle, il est inutile de le préciser**.

Pour les clés étrangères, plusieurs cas se présentent :

- Lorsque l'association possède une cardinalité **1,1** la clé étrangère doit référencer obligatoirement la clé primaire de la table associée : dans ce cas, une valeur non nulle est exigée.
- En cas de cardinalité **0,1** le lien n'est pas obligatoire et la clé étrangère peut donc être non renseignée (contenu = NULL) : elle est alors marquée d'un astérisque * au niveau du MLD pour bien identifier ce cas précis.

© Patrick Bergougnoux

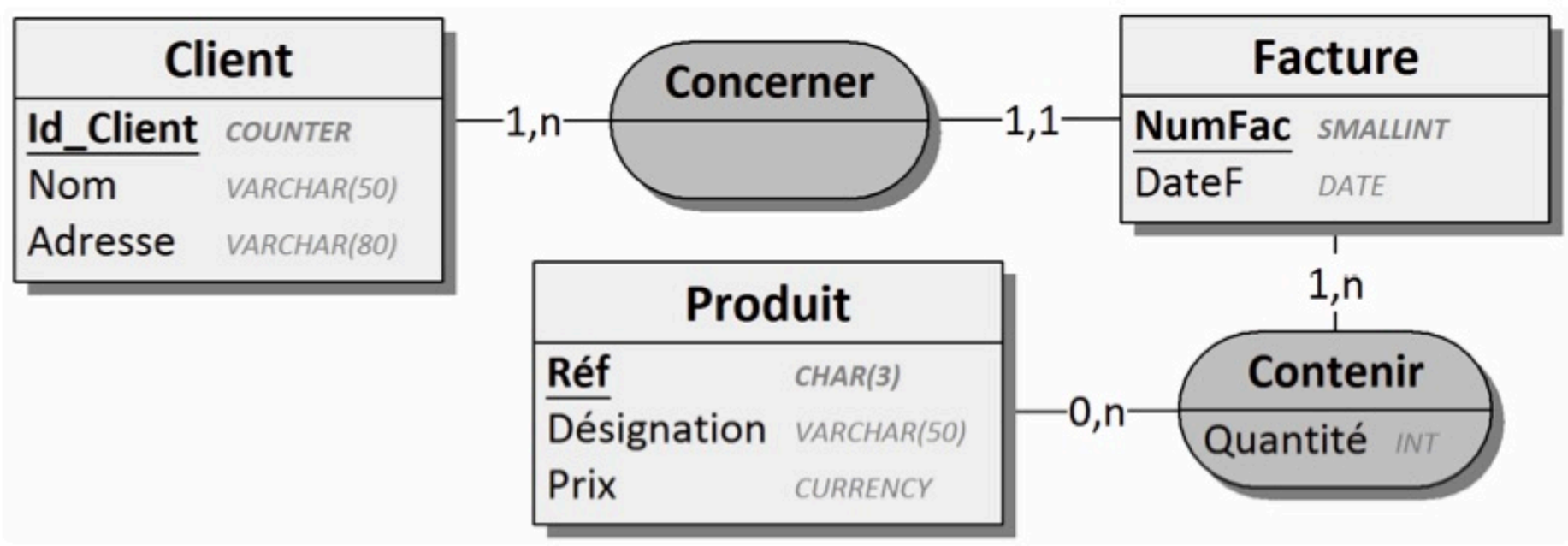
- Pour une association avec des cardinalités multiples de part et d'autre, les liens ne sont créés dans la table de correspondance que si l'association entre deux entités est effective. Les clés étrangères participent alors à la clé primaire et sont donc forcément renseignées, c'est à dire non nulles.

Il ne faut pas confondre « NULL » avec une chaine de caractères vide ou un zéro qui sont des valeurs comme les autres. « NULL » signifie que la rubrique ne contient aucune valeur.

Affichage des types dès le MCDi

En définissant les types de données dès le MCDi, ce modèle devient central et complet en répondant à toutes les problématiques du système d'information et de son automatisation.

Avec ***Looping***, les types de données s'affichent automatiquement dans les MCDi en cochant simplement la case « Types données ». Pour les MLD, l'option est proposée dans les propriétés générales.



Client = (**Id_Client** *COUNTER*, Nom *VARCHAR(50)*, Adresse *VARCHAR(80)*)
Facture = (**NumFac** *SMALLINT*, DateF *DATE*, *#Id_Client*)
Produit = (**Réf** *CHAR(3)*, Désignation *VARCHAR(50)*, Prix *CURRENCY*)
Contenir = (***#NumFac, #Réf***, Quantité *INT*)

Le type d'une clé étrangère est bien sûr le même que celui de la clé primaire correspondante (sauf, comme vu précédemment, pour « COUNTER ») : il est alors inutile de repréciser le type des clés étrangères dans le MLD.

On remarque que, quelles que soient les valeurs des cardinalités multiples (**0,n** ou **1,n**), les clés étrangères sont générées de la même manière dans le schéma relationnel décrit par le MLD. Les contraintes liées aux bornes (y compris pour des cardinalités de type **1,4**) sont traitées par la gestion de contraintes et l'utilisation de déclencheurs (triggers) spécifiques à chaque SGBD et au langage qui lui est adossé. Cet aspect ne sera pas développé dans cet ouvrage.

Génération automatique du MLD

Un passage du MCDi au MLD facilement automatisable

Il est bien sûr préférable de bien maitriser tous les concepts décrits dans ce chapitre. Néanmoins, même si certaines notions ont pu vous échapper, il existe une méthode qui permet de déduire automatiquement un MLD du MCDi correspondant. C'est d'ailleurs cet algorithme qu'utilise ***Looping***.

Un processus en trois étapes simples

Pour une plus grande clarté, les types de données ne sont pas repris :

1. Recopier toutes les rubriques des classes d'entités du MCDi sans fermer la parenthèse (des clés étrangères pouvant venir ensuite compléter les tables) ; les caractères spéciaux et les blancs doivent être remplacés par des « _ » :

   Client = (**Id_Client**, Nom, Adresse

   Facture = (**NumFac**, DateF

   Produit = (**Réf**, Désignation, Prix

2. Rajouter une clé étrangère pour chaque patte simple **1,1** et éventuellement **0,1** raccrochée à une classe d'entités, puis fermer la parenthèse :

   Client = (**Id_Client**, Nom, Adresse) *pas de patte simple*

   Facture = (**NumFac**, DateF, *#Id_Client*) *1 patte simple ➔ 1 clé étrangère*

   Produit = (**Réf**, Désignation, Prix) *pas de patte simple*

3. Créer une table de correspondance pour toute association multiple de part et d'autre, qu'elle soit ou non porteuse de rubriques, avec pour clé primaire la concaténation des clés étrangères issues des classes d'entités associées ; rajouter ensuite les éventuelles rubriques portées par l'association :

   Contenir = (***#NumFac, #Réf***, Quantité)

Toutes les tables du MLD sont ainsi définies de manière systématique.

Ce qu'il faut retenir

- Le modèle logique de données (MLD) reprend la structure des tables issues du MCDi complété par la définition des types de données.
- Les identifiants rajoutés pour des problèmes techniques sont gérés par le SGBD avec un type entier auto-incrémenté.
- Une clé étrangère possède le même type que la clé primaire associée.

© Patrick Bergougnoux

Chapitre 11

Requêtes SQL de création de tables

Le modèle logique de données sous forme de schéma relationnel textuel contient toutes les informations nécessaires à la construction de la structure de la base de données : libellés des tables et des rubriques, clés primaires, clés étrangères, types des données.

Utilisation de SQL

Un langage normalisé

SQL (Structured Query Language) est, comme son nom l'indique, un langage de requête structuré ; il est utilisé pour exploiter des bases de données relationnelles. Créé en 1974 et normalisé depuis 1986, SQL est aujourd'hui reconnu par tous les systèmes de gestion de bases de données relationnelles du marché.

La majeure partie du langage concerne la manipulation des données de la base : il offre ainsi la possibilité de rechercher, d'ajouter, de modifier ou de supprimer des données dans les bases de données relationnelles ; il intègre également des aspects de contrôle de transactions et d'administration d'utilisateurs pour la gestion de l'accès aux données.

Un langage d'interrogation structuré

Bien que doté de très nombreuses fonctions, SQL est surtout connu pour ses requêtes de recherche d'information, au sein d'une base déjà construite, avec son fameux « SELECT ».

Par exemple, pour afficher le nom des clients avec les produits qu'il a achetés en 2019, la requête suivante sera utilisée :

```
SELECT Client.Nom,  Produit.Réf,  Facture.DateF,  Contenir.Quantité
FROM   Client,  Facture,  Contenir,  Produit
WHERE  Client.Id_Client    =  Facture.Id_Client
AND    Facture.NumFac  =  Contenir.NumFac
AND    Contenir.Réf       =  Produit.Réf
AND    Facture.DateF  LIKE  "*2019" ;
```

Mais nous ne détaillerons pas davantage cet aspect de SQL : de nombreux ouvrages le font déjà très bien.

Notons simplement la façon dont les différentes tables sont associées à travers les jointures entre clés primaires et clés étrangères. Des spécificités pour assurer ces jointures peuvent également apparaitre en fonction des SGBD.

On note également la disparition du « # » dans le libellé des clés étrangères.

Création du schéma relationnel

Requêtes de création de tables

Au-delà de l'aspect manipulation des données, SQL permet aussi la définition de la structure des tables constituant la base de données. Pour générer cette structure, des requêtes SQL doivent donc également être mises en œuvre.

Pour cette partie définition, en plus des différences de types entre SGBD constatées dans le chapitre précédent, des spécificités peuvent être proposées par certains SGBD. Ces variantes restent néanmoins à la marge et nous tacherons, dans cet ouvrage, d'utiliser des instructions basiques compatibles avec tous les systèmes.

L'instruction « CREATE TABLE »

Chaque table du MLD doit faire l'objet d'une requête de création. Pour cela, c'est l'instruction « CREATE TABLE » qui est utilisée avec, dans un premier temps, une recopie quasi à l'identique du MLD, en précisant l'instruction « NOT NULL » pour les rubriques devant être forcément renseignées, en particulier les clés étrangères issues d'une association **1,1**.

© Patrick Bergougnoux

Les composantes d'une clé primaire ne peuvent en aucun cas être nulles : il est donc inutile de préciser l'instruction « NOT NULL » pour ces rubriques.

Il suffit ensuite d'indiquer quelles sont, parmi ces rubriques, celles qui ont un rôle de clé, primaire ou étrangère. Pour une clé étrangère, il convient également de préciser la clé primaire à laquelle elle fait référence.

Pour notre exemple simplifié de facturation Beaux Rivages, voici les instructions correspondant à la création du schéma relationnel avec les types de données « Access » :

```
CREATE TABLE Client(
        Id_Client COUNTER,
        Nom VARCHAR(50),
        Adresse VARCHAR(80),
        PRIMARY KEY(Id_Client)
);

CREATE TABLE Facture(
        NumFac SMALLINT,
        DateF DATE,
        Id_Client INT NOT NULL,
        PRIMARY KEY(NumFac),
        FOREIGN KEY(Id_Client) REFERENCES Client(Id_Client)
);

CREATE TABLE Produit(
        Réf CHAR(3),
        Désignation VARCHAR(50),
        Prix CURRENCY,
        PRIMARY KEY(Réf)
);

CREATE TABLE Contenir(
        NumFac SMALLINT,
        Réf CHAR(3),
        Quantité INT,
        PRIMARY KEY(NumFac, Réf),
        FOREIGN KEY(NumFac) REFERENCES Facture(NumFac),
        FOREIGN KEY(Réf) REFERENCES Produit(Réf)
);
```

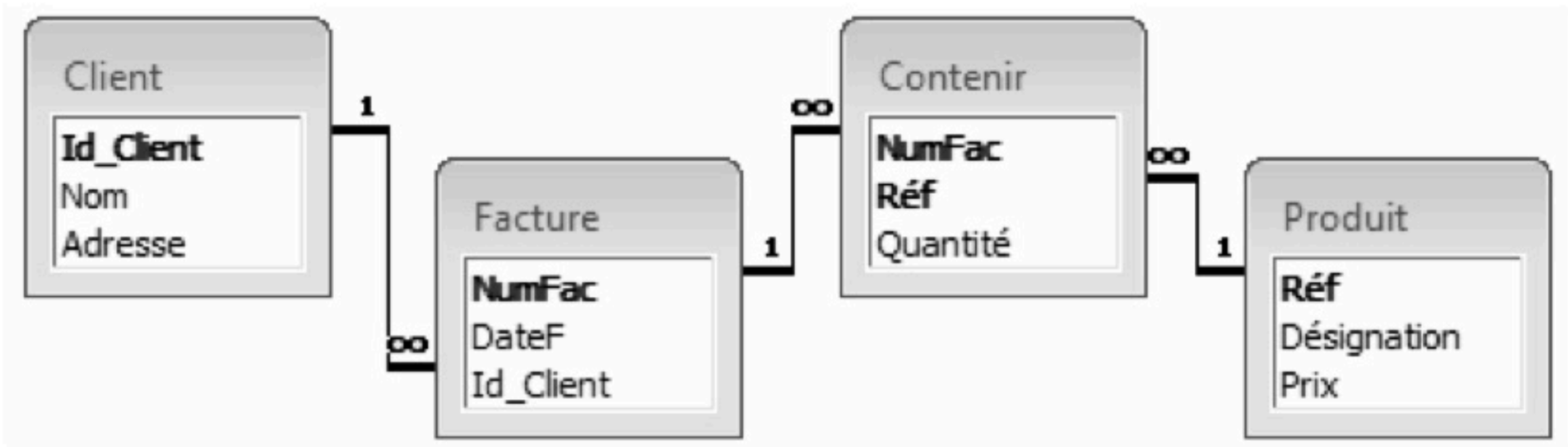
L'utilisation des majuscules ou des minuscules dans les instructions SQL est totalement libre, quels que soient les mots concernés. Néanmoins, afin de rendre la requête la plus lisible possible, il est préférable de mettre les mots clés SQL en majuscule, et conserver la même écriture que dans le MCDi pour le libellé des rubriques.

Comme pour le MLD, l'ordre dans lequel les tables sont définies est important : en effet, on ne peut pas faire référence à une table qui n'a pas encore été créée. En cas de référence circulaire, il est possible de rajouter une clé étrangère à une table après sa création avec l'instruction « ALTER TABLE ».

Looping propose l'affichage temps-réel du MLD et des requêtes SQL correspondantes au fur et à mesure de la construction du MCDi. De plus, une fonction d'exportation permet de générer un script d'exécution de requêtes SQL pour une construction automatique du schéma relationnel en fonction du SGBD sélectionné.

Exécution des requêtes

Le lancement de ces requêtes dans le SGBD permet de créer les tables et les liens correspondant aux associations. Voici le schéma relationnel obtenu après le lancement dans « Access » du script généré par ***Looping*** :



La base de données est alors prête à être utilisée.

Ce qu'il faut retenir

- SQL est un langage de requête structuré utilisé pour exploiter les bases de données relationnelles.
- Chaque table du MLD fait l'objet d'une requête SQL avec l'instruction « CREATE TABLE » permettant de générer le schéma relationnel.
- L'ordre de création des tables doit respecter le fait que l'on ne peut faire référence à une autre table que si elle existe déjà.

© Patrick Bergougnoux

Chapitre 12

Reprise exercices avec MLD et SQL

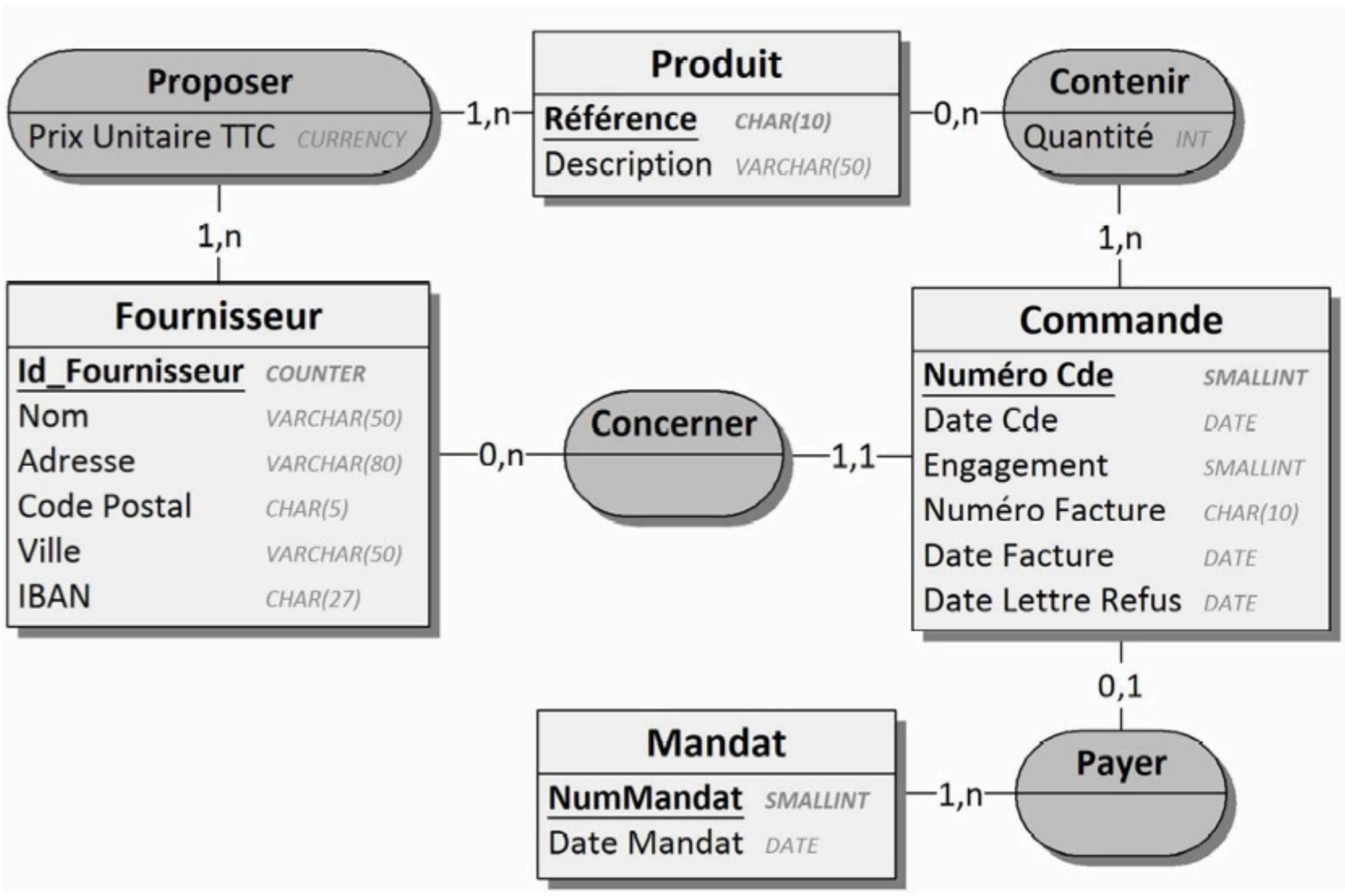
Dans ce dernier chapitre sur l'informatisation des modèles, nous reprenons l'ensemble des exercices réalisés dans la partie précédente *(cf. chapitre 8)*.

Pour chaque exercice, nous poursuivons la démarche en réalisant les étapes vues dans cette troisième partie, à savoir :

- Le MCDi : sur la base du MCD, un identifiant est défini pour chaque classe d'entités en s'assurant que son libellé est significatif pour l'ensemble du système d'information ; les types de données sont également précisés, et les libellés des rubriques sont parfois légèrement modifiés pour une meilleure lisibilité.
- Le MLD : directement dérivé du MCDi, il met en évidence sous une forme textuelle le schéma relationnel de la base de données avec, en particulier, les clés étrangères.
- Les requêtes SQL de création des tables : chaque table du MLD fait l'objet d'une requête avec l'instruction « CREATE TABLE » ; le SGBD choisi est « Access », sachant que seuls les types de données diffèrent par rapport à d'autres SGBD.
- La génération du schéma relationnel complet : une représentation de ce schéma relationnel suite à l'exécution des requêtes SQL sous « Access » est présentée.

Gestion des achats d'une administration

MCDi



MLD

Fournisseur = (**Id_Fournisseur** *COUNTER*, Nom *VARCHAR(50)*, Adresse *VARCHAR(50)*, Code_Postal *CHAR(5)*, Ville *VARCHAR(50)*, IBAN *CHAR(27)*)

Produit = (**Référence** *CHAR(10)*, Description *VARCHAR(50)*)

Mandat = (**NumMandat** *SMALLINT*, Date_Mandat *DATE*)

Commande = (**Numéro_Cde** *SMALLINT*, Date_Cde *DATE*, Engagement *SMALLINT*, Numéro_Facture *CHAR(10)*, Date_Facture *DATE*, Date_Lettre_Refus *DATE*, *#NumMandat**, *#Id_Fournisseur*)

Contenir = (***#Référence, #Numéro_Cde***, Quantité *INT*)

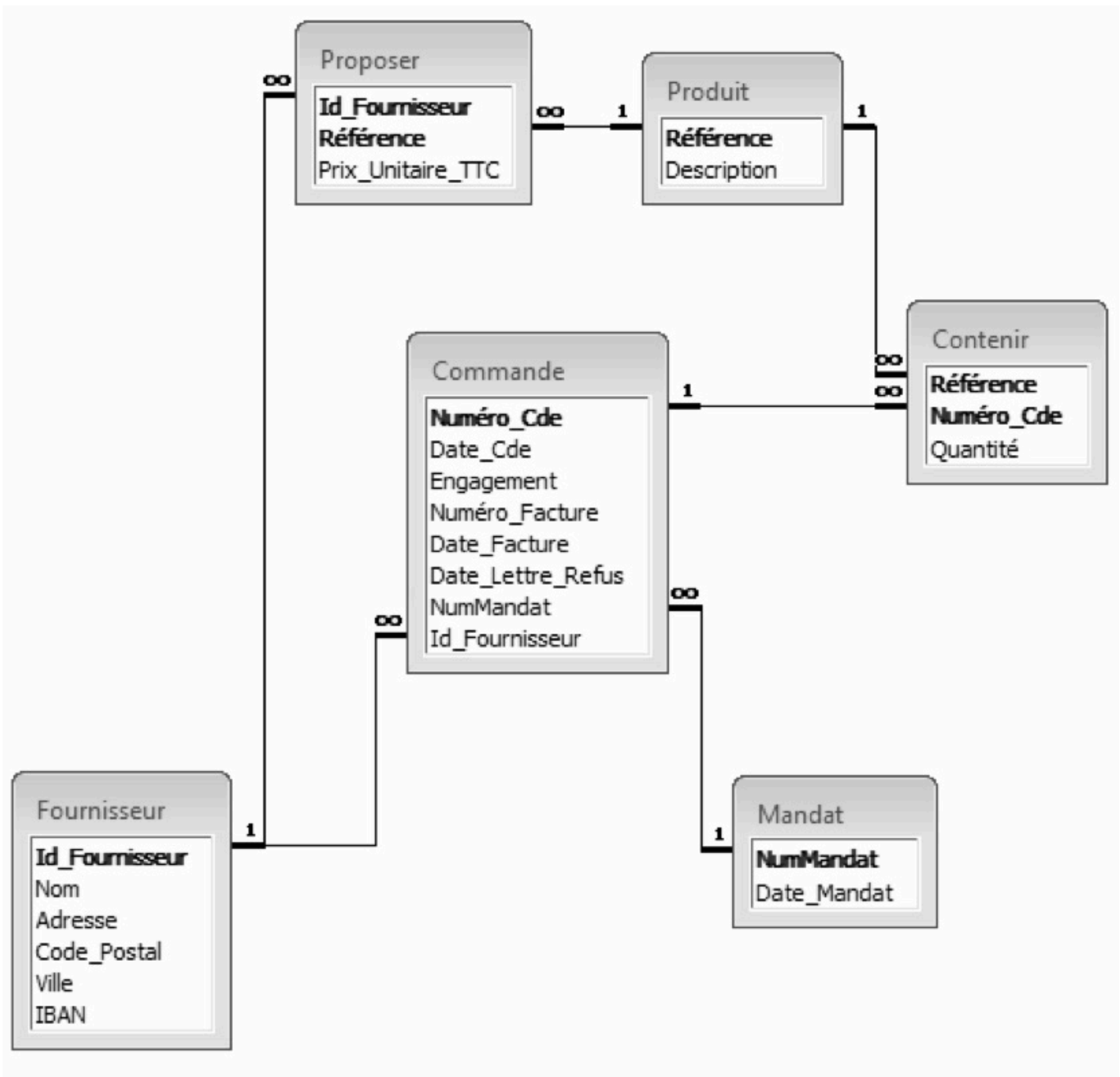
Proposer = (***#Id_Fournisseur, #Référence***, Prix_Unitaire_TTC *CURRENCY*)

© Patrick Bergougnoux

Requêtes SQL

```
CREATE TABLE Fournisseur(
	Id_Fournisseur COUNTER,
	Nom VARCHAR(50),
	Adresse VARCHAR(50),
	Code_Postal CHAR(5),
	Ville VARCHAR(50),
	IBAN CHAR(27),
	PRIMARY KEY(Id_Fournisseur)
);
CREATE TABLE Produit(
	Référence CHAR(10),
	Description VARCHAR(50),
	PRIMARY KEY(Référence)
);
CREATE TABLE Mandat(
	NumMandat SMALLINT,
	Date_Mandat DATE,
	PRIMARY KEY(NumMandat)
);
CREATE TABLE Commande(
	Numéro_Cde SMALLINT,
	Date_Cde DATE,
	Engagement SMALLINT,
	Numéro_Facture CHAR(10),
	Date_Facture DATE,
	Date_Lettre_Refus DATE,
	NumMandat SMALLINT,
	Id_Fournisseur INT NOT NULL,
	PRIMARY KEY(Numéro_Cde),
	FOREIGN KEY(NumMandat) REFERENCES Mandat(NumMandat),
	FOREIGN KEY(Id_Fournisseur) REFERENCES Fournisseur(Id_Fournisseur)
);
CREATE TABLE Contenir(
	Référence CHAR(10),
	Numéro_Cde SMALLINT,
	Quantité INT,
	PRIMARY KEY(Numéro_Cde, Référence),
	FOREIGN KEY(Référence) REFERENCES Produit(Référence),
	FOREIGN KEY(Numéro_Cde) REFERENCES Commande(Numéro_Cde)
);
CREATE TABLE Proposer(
	Id_Fournisseur INT,
	Référence CHAR(10),
	Prix_Unitaire_TTC CURRENCY,
	PRIMARY KEY(Id_Fournisseur, Référence),
	FOREIGN KEY(Id_Fournisseur) REFERENCES Fournisseur(Id_Fournisseur),
	FOREIGN KEY(Référence) REFERENCES Produit(Référence)
);
```

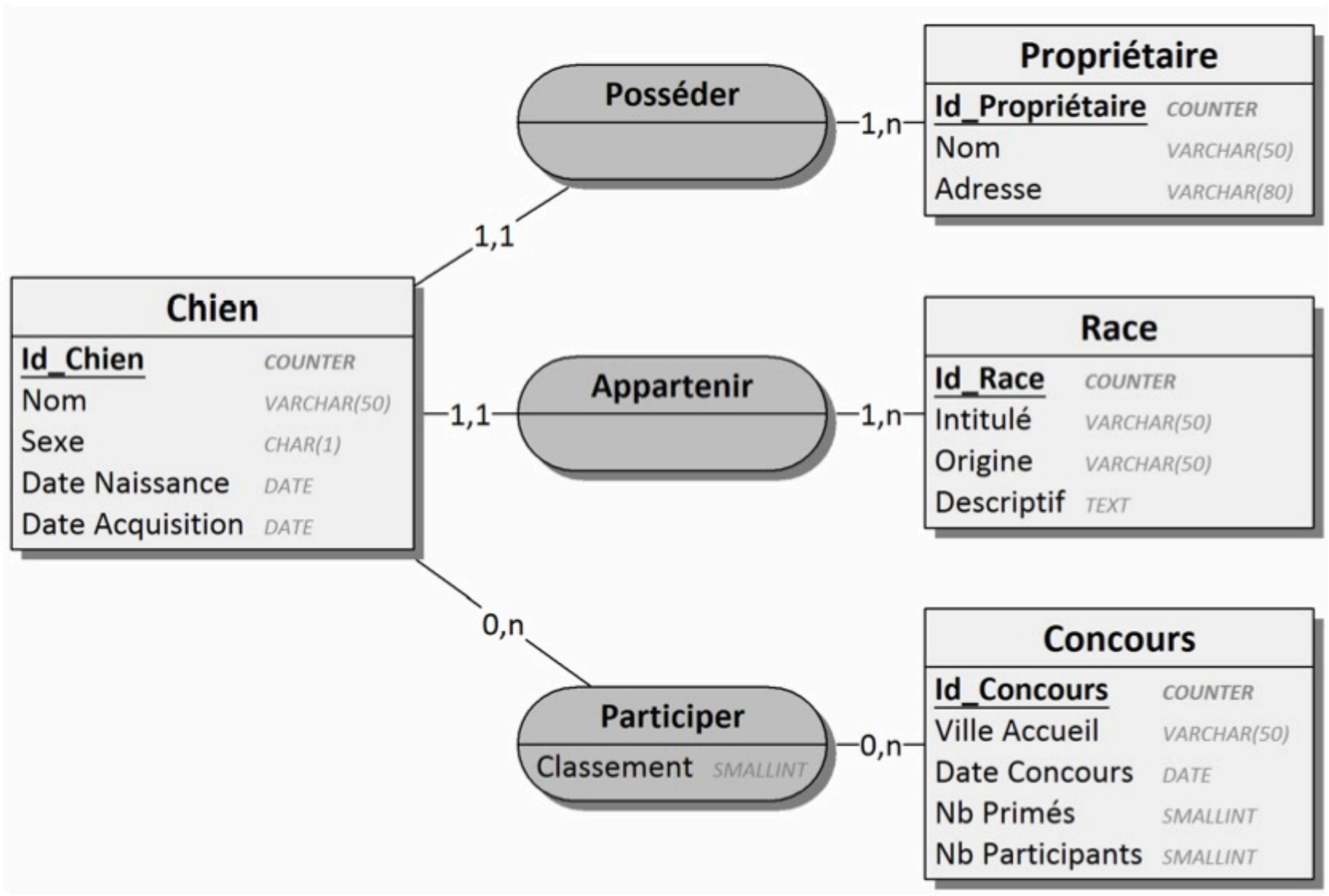
Schéma relationnel Access



© Patrick Bergougnoux

Société canine

MCDi



MLD

Propriétaire = (**Id_Propriétaire** *COUNTER*, Nom *VARCHAR(50)*, Adresse *VARCHAR(80)*)

Race = (**Id_Race** *COUNTER*, Intitulé *VARCHAR(50)*, Origine *VARCHAR(50)*, Descriptif *TEXT*)

Concours = (**Id_Concours** *COUNTER*, Ville_Accueil *VARCHAR(50)*, Date_Concours *DATE*, Nb_Primés *SMALLINT*, Nb_Participants *SMALLINT*)

Chien = (**Id_Chien** *COUNTER*, Nom *VARCHAR(50)*, Sexe *CHAR(1)*, Date_Naissance *DATE*, Date_Acquisition *DATE*, *#Id_Propriétaire*, *#Id_Race*)

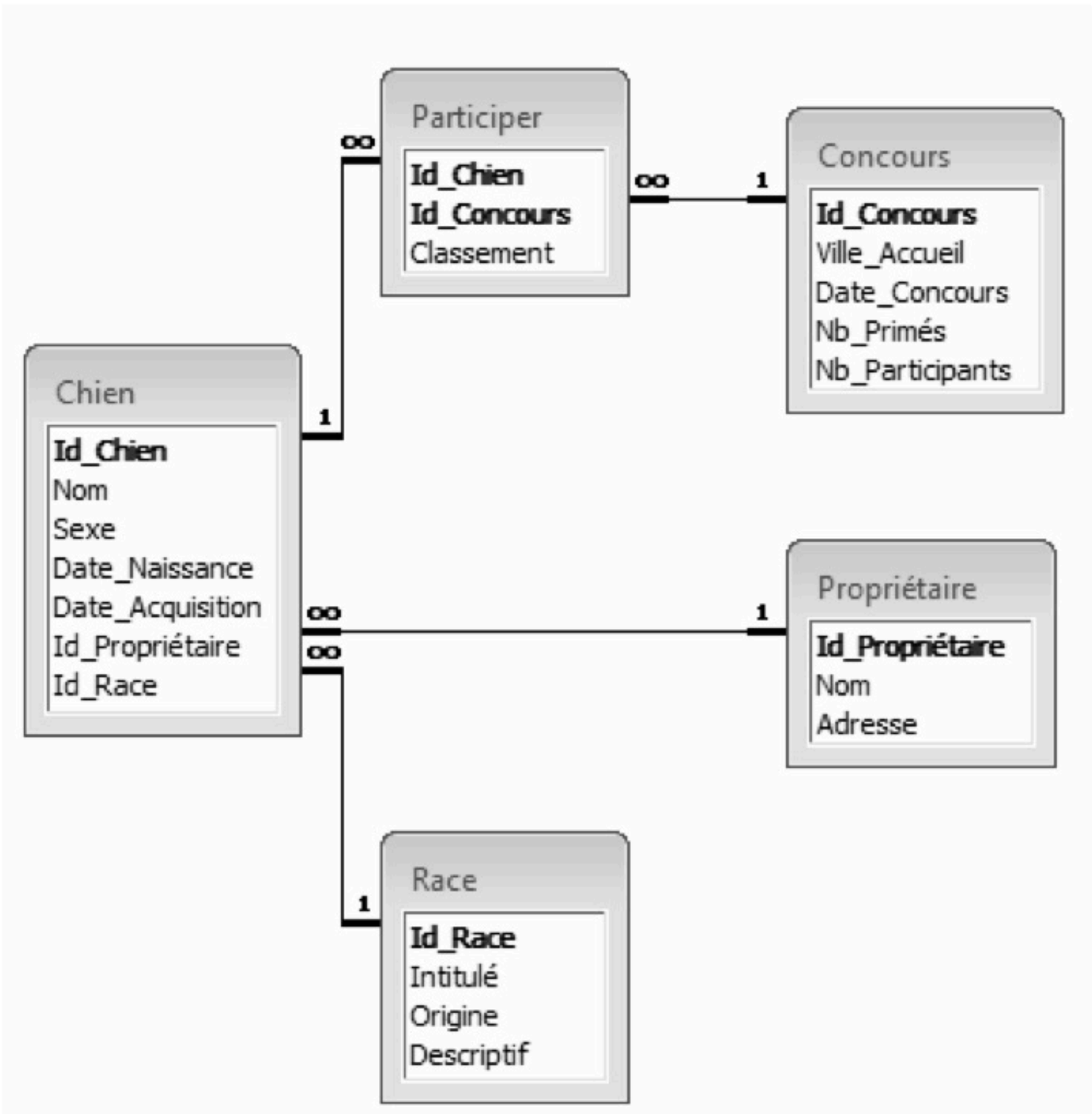
Participer = (***#Id_Chien, #Id_Concours***, Classement *SMALLINT*)

Requêtes SQL

```
CREATE TABLE Propriétaire(
	Id_Propriétaire COUNTER,
	Nom VARCHAR(50),
	Adresse VARCHAR(80),
	PRIMARY KEY(Id_Propriétaire)
);

CREATE TABLE Race(
	Id_Race COUNTER,
	Intitulé VARCHAR(50),
	Origine VARCHAR(50),
	Descriptif TEXT,
	PRIMARY KEY(Id_Race)
);

CREATE TABLE Concours(
	Id_Concours COUNTER,
	Ville_Accueil VARCHAR(50),
	Date_Concours DATE,
	Nb_Primés SMALLINT,
	Nb_Participants SMALLINT,
	PRIMARY KEY(Id_Concours)
);

CREATE TABLE Chien(
	Id_Chien COUNTER,
	Nom VARCHAR(50),
	Sexe CHAR(1),
	Date_Naissance DATE,
	Date_Acquisition DATE,
	Id_Propriétaire INT NOT NULL,
	Id_Race INT NOT NULL,
	PRIMARY KEY(Id_Chien),
	FOREIGN KEY(Id_Propriétaire) REFERENCES Propriétaire(Id_Propriétaire),
	FOREIGN KEY(Id_Race) REFERENCES Race(Id_Race)
);

CREATE TABLE Participer(
	Id_Chien INT,
	Id_Concours INT,
	Classement SMALLINT,
	PRIMARY KEY(Id_Chien, Id_Concours),
	FOREIGN KEY(Id_Chien) REFERENCES Chien(Id_Chien),
	FOREIGN KEY(Id_Concours) REFERENCES Concours(Id_Concours)
);
```

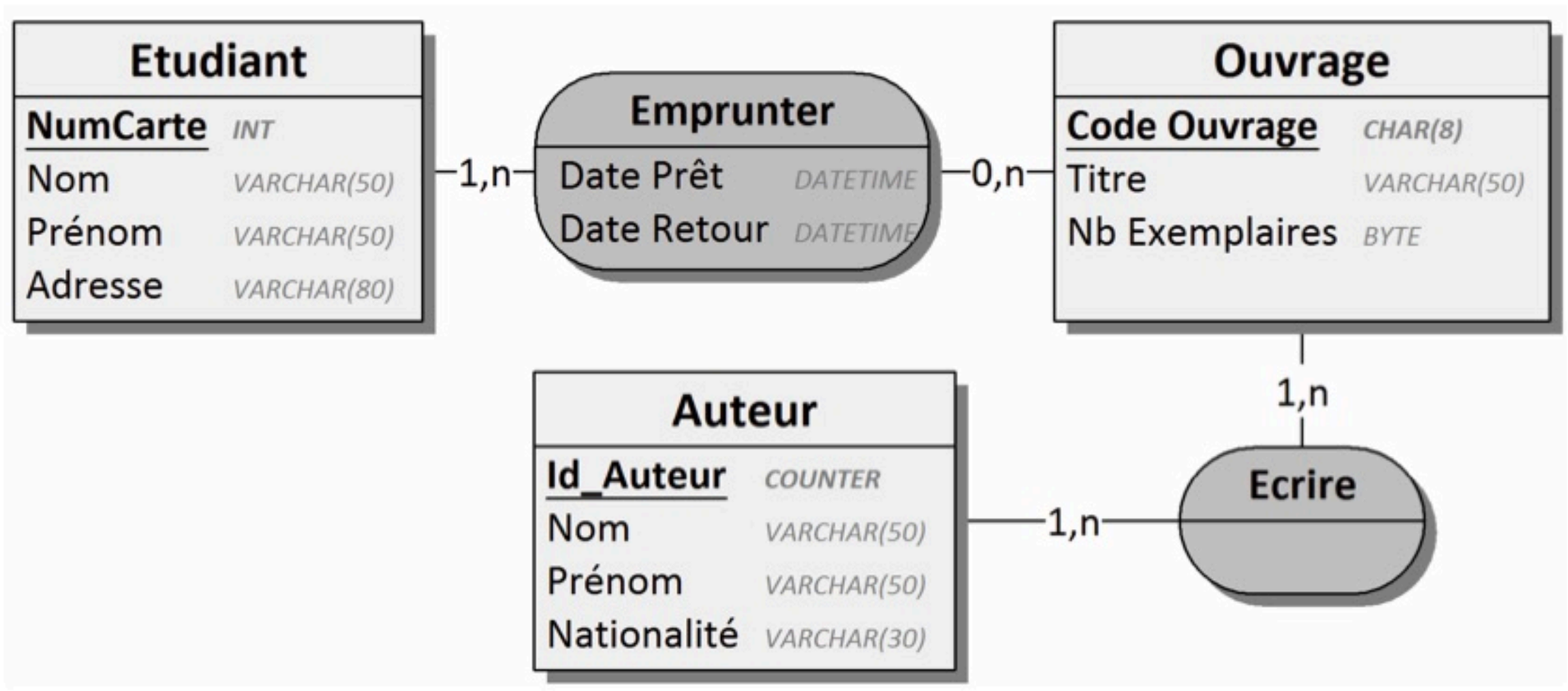

© Patrick Bergougnoux

Schéma relationnel Access



Bibliothèque universitaire

MCDi



MLD

Etudiant = (**NumCarte** *INT*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Adresse *VARCHAR(80)*)

Ouvrage = (**Code_Ouvrage** *CHAR(8)*, Titre *VARCHAR(50)*, Nb_Exemplaires *BYTE*)

Auteur = (**Id_Auteur** *COUNTER*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Nationalité *VARCHAR(30)*)

Emprunter = (***#NumCarte, #Code_Ouvrage***, Date_Prêt *DATETIME*, Date_Retour *DATETIME*)

Ecrire = (***#Code_Ouvrage, #Id_Auteur***)

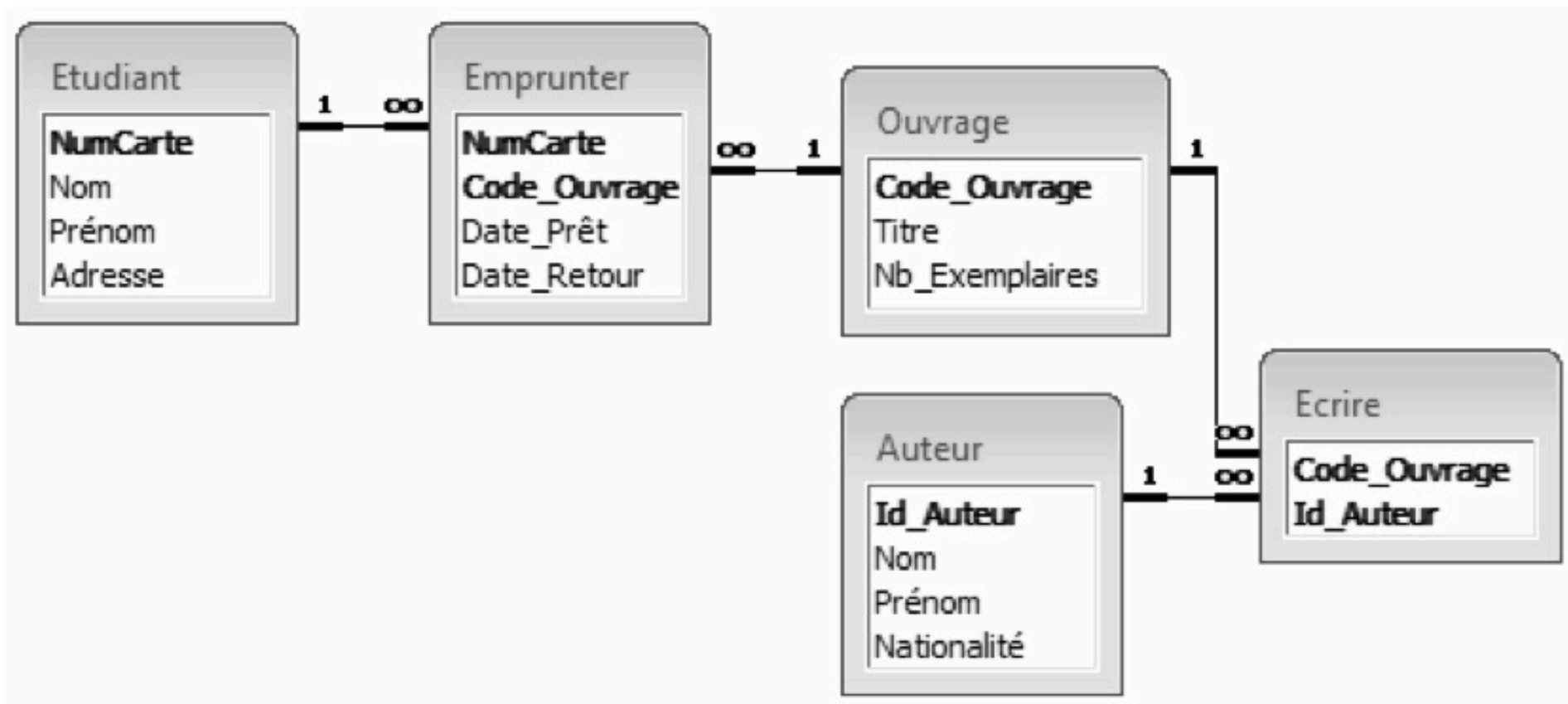
Requêtes SQL

```
CREATE TABLE Etudiant(
        NumCarte INT,
        Nom VARCHAR(50),
        Prénom VARCHAR(50),
        Adresse VARCHAR(80),
        PRIMARY KEY(NumCarte)
);
```


© Patrick Bergougnoux

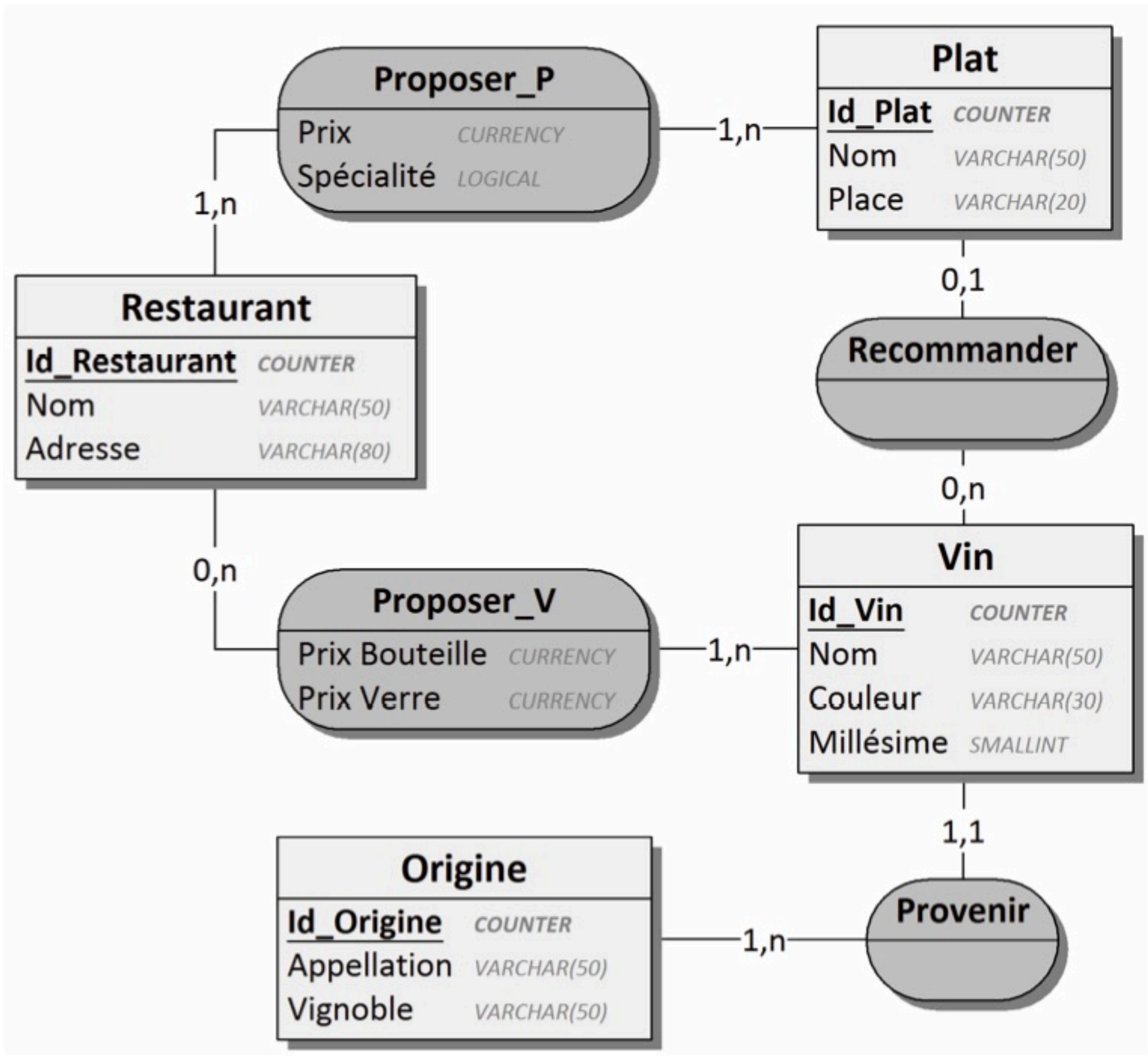
```
CREATE TABLE Ouvrage(
	Code_Ouvrage CHAR(8),
	Titre VARCHAR(50),
	Nb_Exemplaires BYTE,
	PRIMARY KEY(Code_Ouvrage)
);

CREATE TABLE Auteur(
	Id_Auteur COUNTER,
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	Nationalité VARCHAR(30),
	PRIMARY KEY(Id_Auteur)
);

CREATE TABLE Emprunter(
	NumCarte INT,
	Code_Ouvrage CHAR(8),
	Date_Prêt DATETIME,
	Date_Retour DATETIME,
	PRIMARY KEY(NumCarte, Code_Ouvrage),
	FOREIGN KEY(NumCarte) REFERENCES Etudiant(NumCarte),
	FOREIGN KEY(Code_Ouvrage) REFERENCES Ouvrage(Code_Ouvrage)
);

CREATE TABLE Ecrire(
	Code_Ouvrage CHAR(8),
	Id_Auteur INT,
	PRIMARY KEY(Code_Ouvrage, Id_Auteur),
	FOREIGN KEY(Code_Ouvrage) REFERENCES Ouvrage(Code_Ouvrage),
	FOREIGN KEY(Id_Auteur) REFERENCES Auteur(Id_Auteur)
);
```

Schéma relationnel Access



Guide gastronomique

MCDi



MLD

Restaurant = (**Id_Restaurant** *COUNTER*, Nom *VARCHAR(50)*, Adresse *VARCHAR(80)*)

Origine = (**Id_Origine** *COUNTER*, Appellation *VARCHAR(50)*, Vignoble *VARCHAR(50)*)

Vin = (**Id_Vin** *COUNTER*, Nom *VARCHAR(50)*, Couleur *VARCHAR(30)*, Millésime *SMALLINT*, *#Id_Origine*)

Plat = (**Id_Plat** *COUNTER*, Nom *VARCHAR(50)*, Place *VARCHAR(20)*, *#Id_Vin**)

Proposer_P = (***#Id_Restaurant, #Id_Plat***, Prix *CURRENCY*, Spécialité *LOGICAL*)

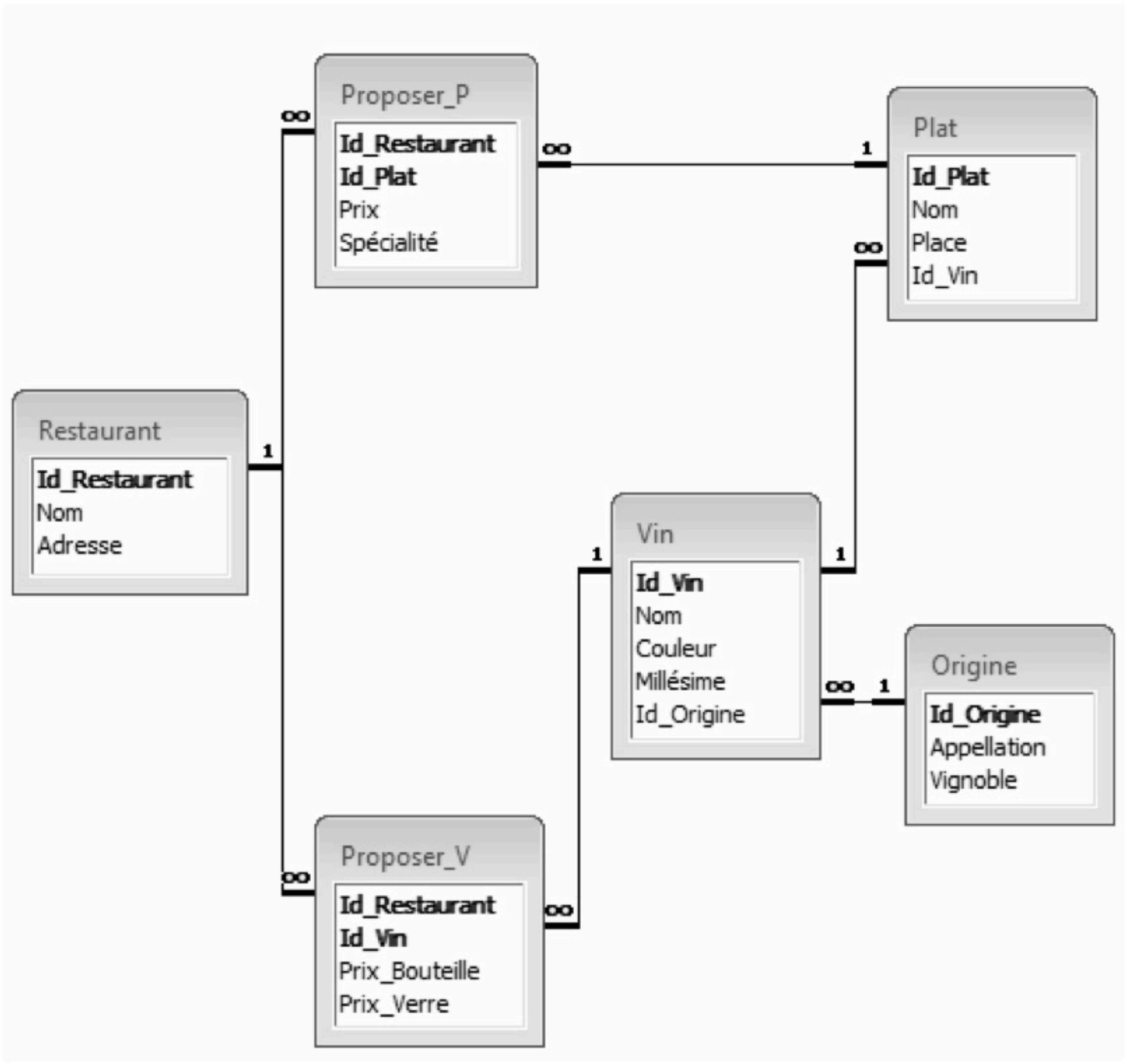
Proposer_V = (***#Id_Restaurant, #Id_Vin***, Prix_Bouteille *CURRENCY*, Prix_Verre *CURRENCY*)

© Patrick Bergougnoux

Requêtes SQL

```
CREATE TABLE Restaurant(
	Id_Restaurant COUNTER,
	Nom VARCHAR(50),
	Adresse VARCHAR(80),
	PRIMARY KEY(Id_Restaurant)
);
CREATE TABLE Origine(
	Id_Origine COUNTER,
	Appellation VARCHAR(50),
	Vignoble VARCHAR(50),
	PRIMARY KEY(Id_Origine)
);
CREATE TABLE Vin(
	Id_Vin COUNTER,
	Nom VARCHAR(50),
	Couleur VARCHAR(30),
	Millésime SMALLINT,
	Id_Origine INT NOT NULL,
	PRIMARY KEY(Id_Vin),
	FOREIGN KEY(Id_Origine) REFERENCES Origine(Id_Origine)
);
CREATE TABLE Plat(
	Id_Plat COUNTER,
	Nom VARCHAR(50),
	Place VARCHAR(20),
	Id_Vin INT,
	PRIMARY KEY(Id_Plat),
	FOREIGN KEY(Id_Vin) REFERENCES Vin(Id_Vin)
);
CREATE TABLE Proposer_P(
	Id_Restaurant INT,
	Id_Plat INT,
	Prix CURRENCY,
	Spécialité LOGICAL,
	PRIMARY KEY(Id_Restaurant, Id_Plat),
	FOREIGN KEY(Id_Restaurant) REFERENCES Restaurant(Id_Restaurant),
	FOREIGN KEY(Id_Plat) REFERENCES Plat(Id_Plat)
);
CREATE TABLE Proposer_V(
	Id_Restaurant INT,
	Id_Vin INT,
	Prix_Bouteille CURRENCY,
	Prix_Verre CURRENCY,
	PRIMARY KEY(Id_Restaurant, Id_Vin),
	FOREIGN KEY(Id_Restaurant) REFERENCES Restaurant(Id_Restaurant),
	FOREIGN KEY(Id_Vin) REFERENCES Vin(Id_Vin)
);
```

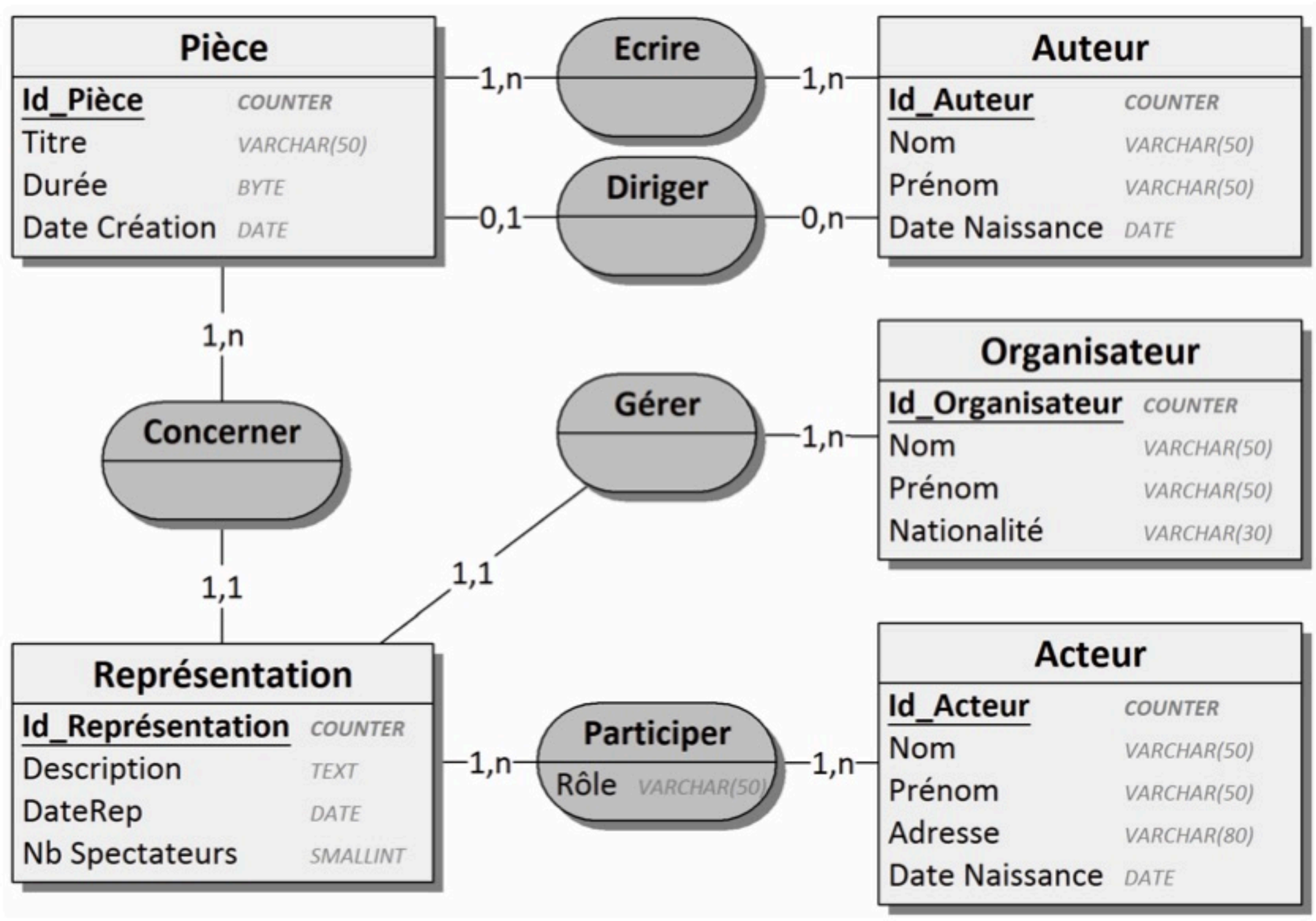
Schéma relationnel Access



© Patrick Bergougnoux

Représentations théâtrales

MCDi



MLD

Organisateur = (**Id_Organisateur** *COUNTER*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Nationalité *VARCHAR(30)*)

Acteur = (**Id_Acteur** *COUNTER*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Adresse *VARCHAR(80)*, Date_Naissance *DATE*)

Auteur = (**Id_Auteur** *COUNTER*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Date_Naissance *DATE*)

Pièce = (**Id_Pièce** *COUNTER*, Titre *VARCHAR(50)*, Durée *BYTE*, Date_Création *DATE*, *#Id_Auteur**)

Représentation = (**Id_Représentation** *COUNTER*, Description *TEXT*, DateRep *DATE*, Nb_Spectateurs *SMALLINT*, *#Id_Pièce, #Id_Organisateur*)

Participer = (***#Id_Représentation, #Id_Acteur***, Rôle *VARCHAR(50)*)

Ecrire = (***#Id_Pièce, #Id_Auteur***)

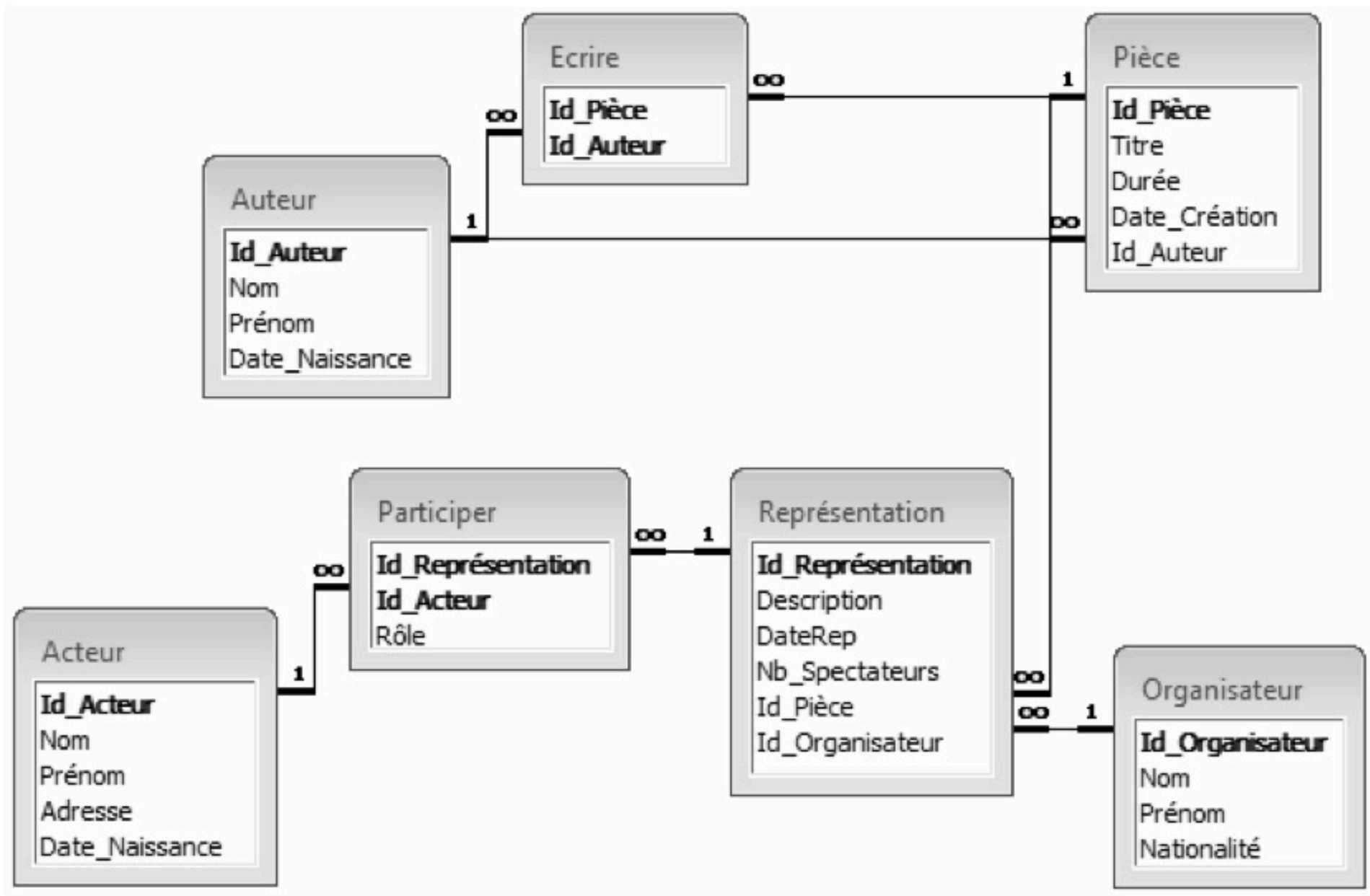
Requêtes SQL

```
CREATE TABLE Organisateur(
	Id_Organisateur COUNTER,
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	Nationalité VARCHAR(30),
	PRIMARY KEY(Id_Organisateur)
);

CREATE TABLE Acteur(
	Id_Acteur COUNTER,
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	Adresse VARCHAR(80),
	Date_Naissance DATE,
	PRIMARY KEY(Id_Acteur)
);

CREATE TABLE Auteur(
	Id_Auteur COUNTER,
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	Date_Naissance DATE,
	PRIMARY KEY(Id_Auteur)
);

CREATE TABLE Pièce(
	Id_Pièce COUNTER,
	Titre VARCHAR(50),
	Durée BYTE,
	Date_Création DATE,
	Id_Auteur INT,
	PRIMARY KEY(Id_Pièce),
	FOREIGN KEY(Id_Auteur) REFERENCES Auteur(Id_Auteur)
);

CREATE TABLE Représentation(
	Id_Représentation COUNTER,
	Description TEXT,
	DateRep DATE,
	Nb_Spectateurs SMALLINT,
	Id_Pièce INT NOT NULL,
	Id_Organisateur INT NOT NULL,
	PRIMARY KEY(Id_Représentation),
	FOREIGN KEY(Id_Pièce) REFERENCES Pièce(Id_Pièce),
	FOREIGN KEY(Id_Organisateur) REFERENCES Organisateur(Id_Organisateur)
);
```


© Patrick Bergougnoux

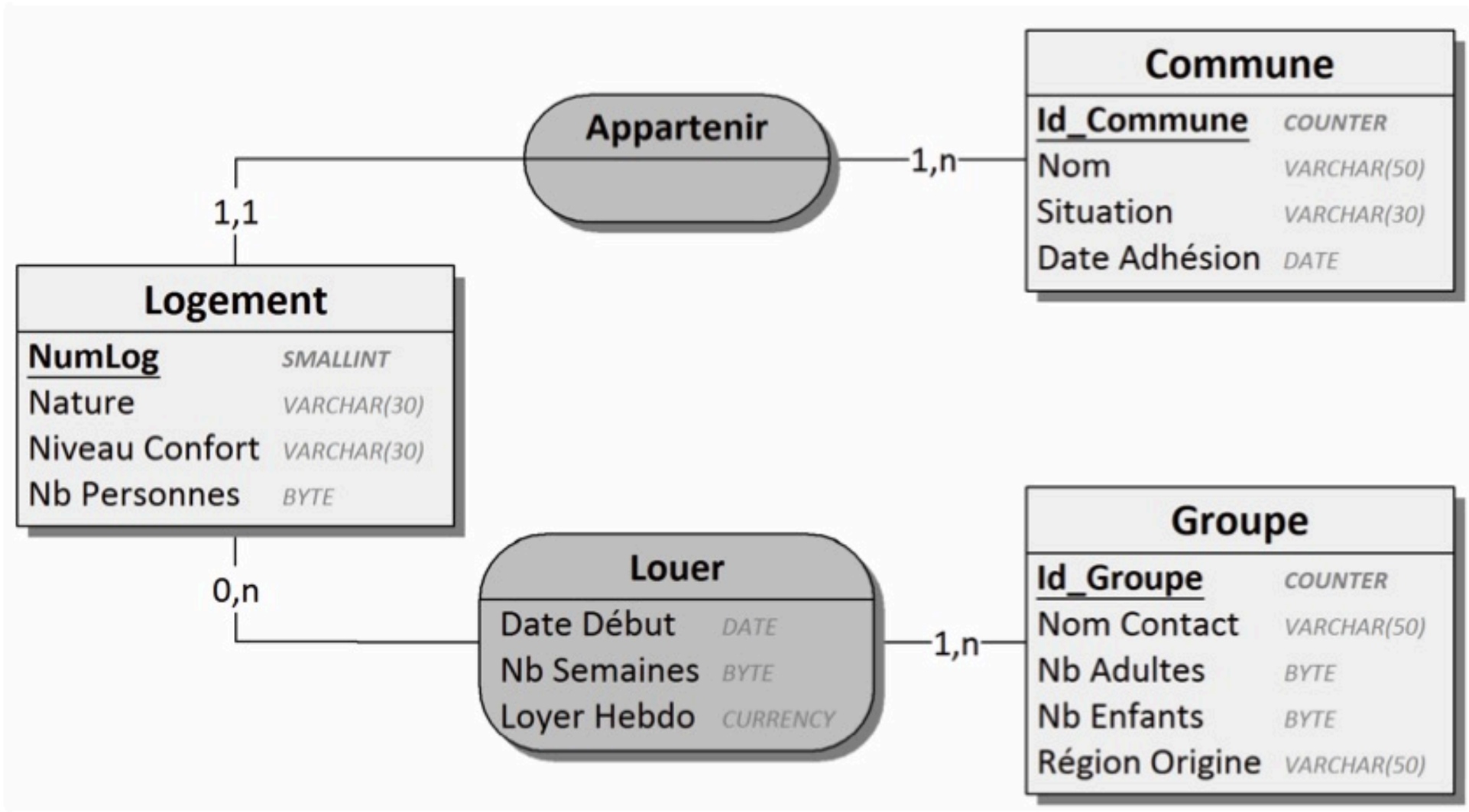
```
CREATE TABLE Participer(
        Id_Représentation INT,
        Id_Acteur INT,
        Rôle VARCHAR(50),
        PRIMARY KEY(Id_Représentation, Id_Acteur),
        FOREIGN KEY(Id_Représentation) REFERENCES
                                        Représentation(Id_Représentation),
        FOREIGN KEY(Id_Acteur) REFERENCES Acteur(Id_Acteur)
);

CREATE TABLE Ecrire(
        Id_Pièce INT,
        Id_Auteur INT,
        PRIMARY KEY(Id_Pièce, Id_Auteur),
        FOREIGN KEY(Id_Pièce) REFERENCES Pièce(Id_Pièce),
        FOREIGN KEY(Id_Auteur) REFERENCES Auteur(Id_Auteur)
);
```

Schéma relationnel Access



Location de logements

MCDi



MLD

Commune = (**Id_Commune** *COUNTER*, Nom *VARCHAR(50)*, Situation *VARCHAR(30)*, Date_Adhésion *DATE*)

Logement = (**NumLog** *SMALLINT*, Nature *VARCHAR(30)*, Niveau_Confort *VARCHAR(30)*, Nb_Personnes *BYTE*, *#Id_Commune*)

Groupe = (**Id_Groupe** *COUNTER*, Nom_Contact *VARCHAR(50)*, Nb_Adultes *BYTE*, Nb_Enfants *BYTE*, Région_Origine *VARCHAR(50)*)

Louer = (***#NumLog, #Id_Groupe***, Date_Début *DATE*, Nb_Semaines *BYTE*, Loyer_Hebdo *CURRENCY*)

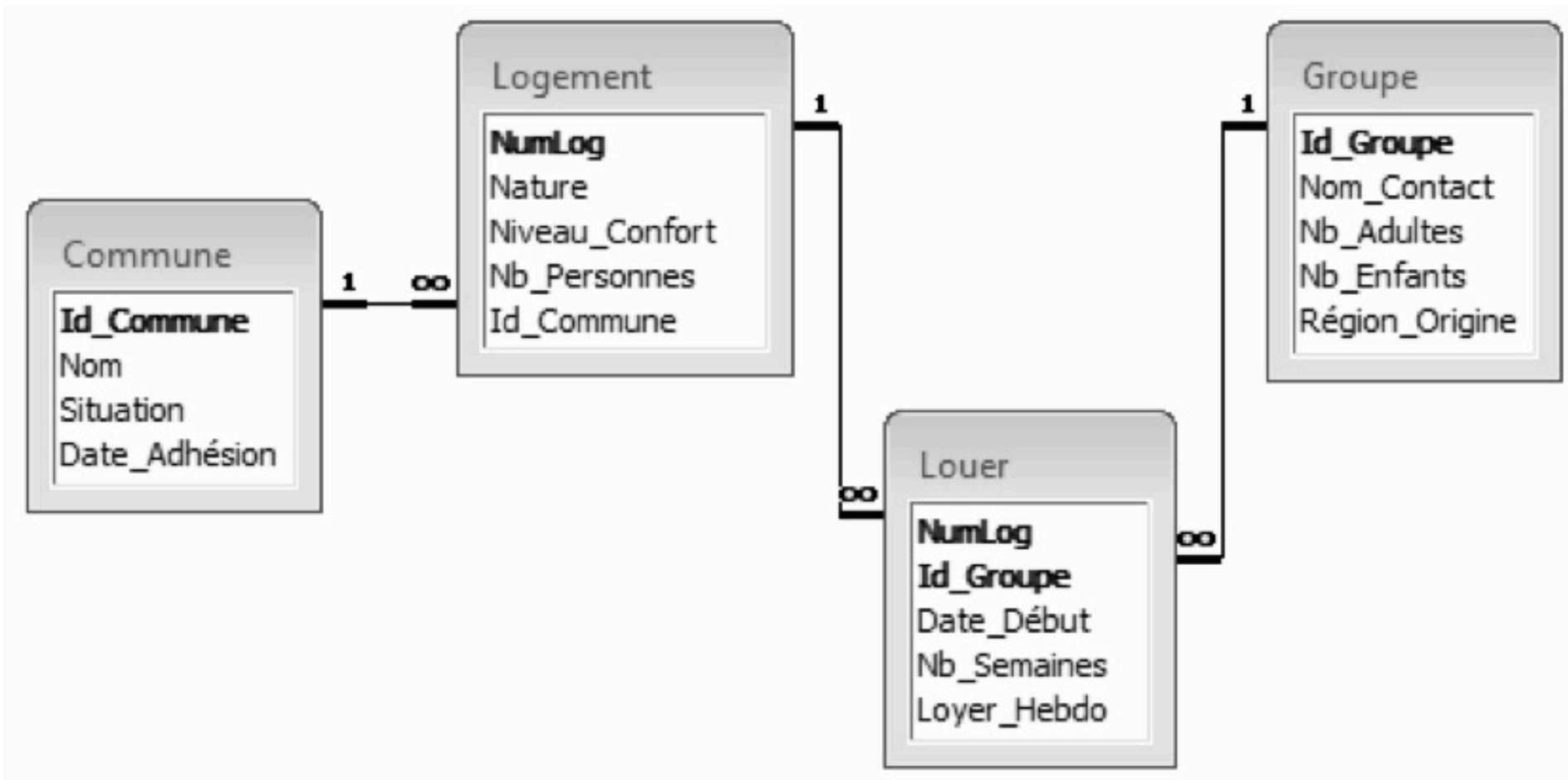
Requêtes SQL

```
CREATE TABLE Commune(
        Id_Commune COUNTER,
        Nom VARCHAR(50),
        Situation VARCHAR(30),
        Date_Adhésion DATE,
        PRIMARY KEY(Id_Commune)
);
```


© Patrick Bergougnoux

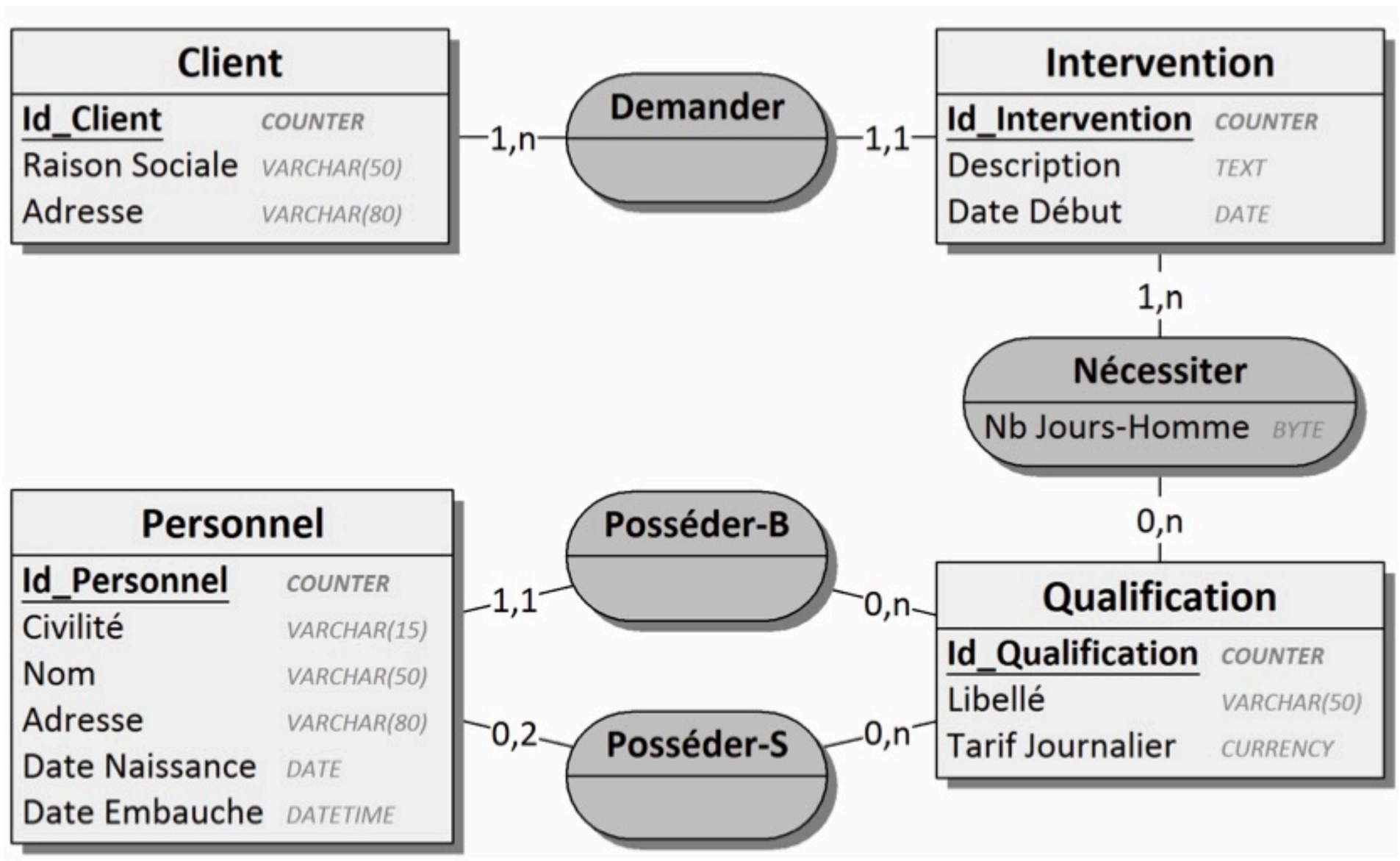
```
CREATE TABLE Logement(
	NumLog SMALLINT,
	Nature VARCHAR(30),
	Niveau_Confort VARCHAR(30),
	Nb_Personnes BYTE,
	Id_Commune INT NOT NULL,
	PRIMARY KEY(NumLog),
	FOREIGN KEY(Id_Commune) REFERENCES Commune(Id_Commune)
);

CREATE TABLE Groupe(
	Id_Groupe COUNTER,
	Nom_Contact VARCHAR(50),
	Nb_Adultes BYTE,
	Nb_Enfants BYTE,
	Région_Origine VARCHAR(50),
	PRIMARY KEY(Id_Groupe)
);

CREATE TABLE Louer(
	NumLog SMALLINT,
	Id_Groupe INT,
	Date_Début DATE,
	Nb_Semaines BYTE,
	Loyer_Hebdo CURRENCY,
	PRIMARY KEY(NumLog, Id_Groupe),
	FOREIGN KEY(NumLog) REFERENCES Logement(NumLog),
	FOREIGN KEY(Id_Groupe) REFERENCES Groupe(Id_Groupe)
);
```

Schéma relationnel Access



Agence de travail temporaire

MCDi



MLD

Client = (**Id_Client** *COUNTER*, Raison_Sociale *VARCHAR(50)*, Adresse *VARCHAR(80)*)

Intervention = (**Id_Intervention** *COUNTER*, Description *TEXT*, Date_Début *DATE*, *#Id_Client*)

Qualification = (**Id_Qualification** *COUNTER*, Libellé *VARCHAR(50)*, Tarif_Journalier *CURRENCY*)

Personnel = (**Id_Personnel** *COUNTER*, Civilité *VARCHAR(15)*, Nom *VARCHAR(50)*, Adresse *VARCHAR(80)*, Date_Naissance *DATE*, Date_Embauche *DATETIME*, *#Id_Qualification*)

Nécessiter = (***#Id_Intervention, #Id_Qualification***, Nb_Jours_Homme *BYTE*)

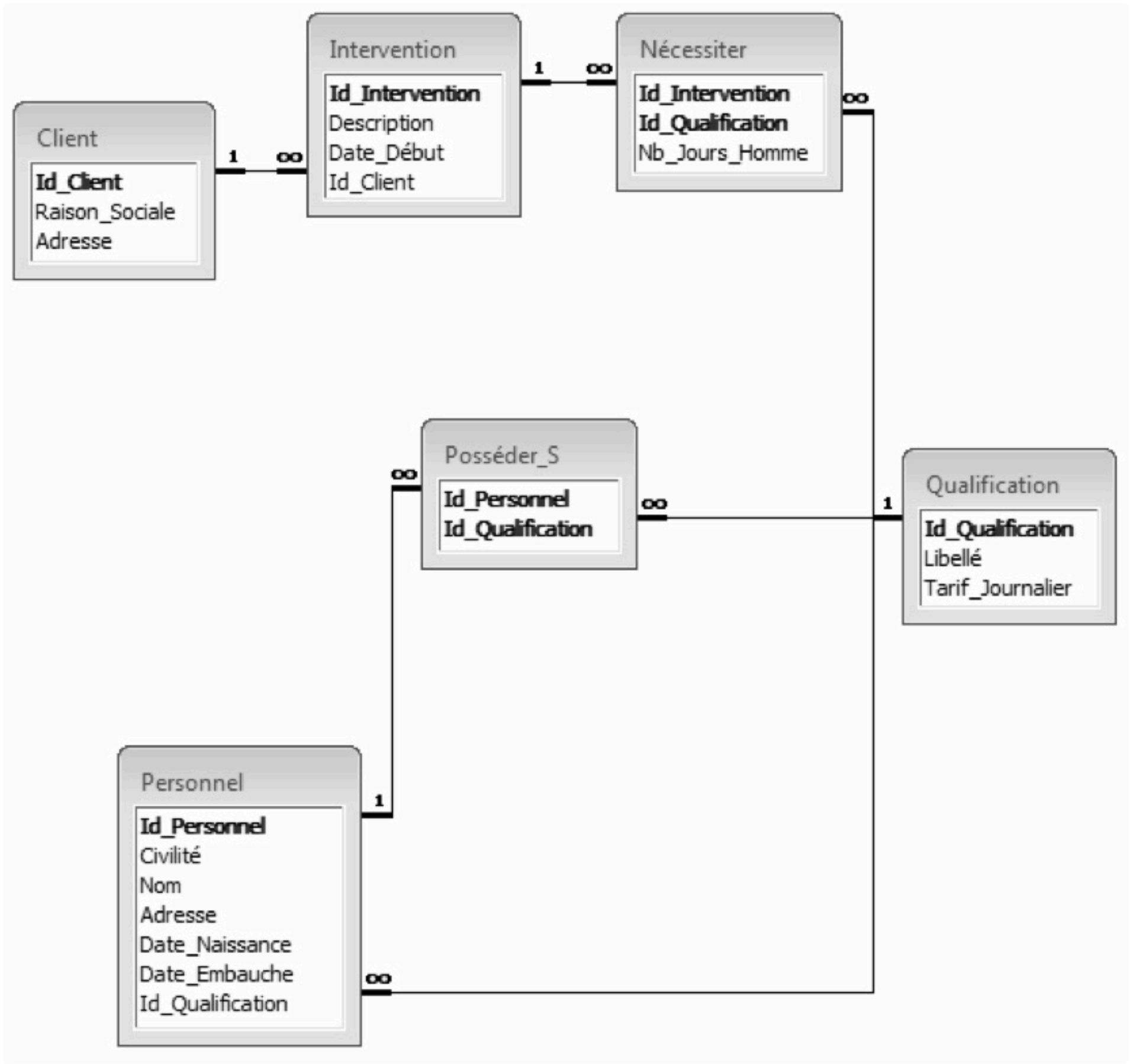
Posséder_S = (***#Id_Personnel, #Id_Qualification***)

© Patrick Bergougnoux

Requêtes SQL

```
CREATE TABLE Client(
	Id_Client COUNTER,
	Raison_Sociale VARCHAR(50),
	Adresse VARCHAR(80),
	PRIMARY KEY(Id_Client)
);

CREATE TABLE Intervention(
	Id_Intervention COUNTER,
	Description TEXT,
	Date_Début DATE,
	Id_Client INT NOT NULL,
	PRIMARY KEY(Id_Intervention),
	FOREIGN KEY(Id_Client) REFERENCES Client(Id_Client)
);

CREATE TABLE Qualification(
	Id_Qualification COUNTER,
	Libellé VARCHAR(50),
	Tarif_Journalier CURRENCY,
	PRIMARY KEY(Id_Qualification)
);

CREATE TABLE Personnel(
	Id_Personnel COUNTER,
	Civilité VARCHAR(15),
	Nom VARCHAR(50),
	Adresse VARCHAR(80),
	Date_Naissance DATE,
	Date_Embauche DATETIME,
	Id_Qualification INT NOT NULL,
	PRIMARY KEY(Id_Personnel),
	FOREIGN KEY(Id_Qualification) REFERENCES Qualification(Id_Qualification)
);

CREATE TABLE Nécessiter(
	Id_Intervention INT,
	Id_Qualification INT,
	Nb_Jours_Homme BYTE,
	PRIMARY KEY(Id_Intervention, Id_Qualification),
	FOREIGN KEY(Id_Intervention) REFERENCES Intervention(Id_Intervention),
	FOREIGN KEY(Id_Qualification) REFERENCES Qualification(Id_Qualification)
);

CREATE TABLE Posséder_S(
	Id_Personnel INT,
	Id_Qualification INT,
	PRIMARY KEY(Id_Personnel, Id_Qualification),
	FOREIGN KEY(Id_Personnel) REFERENCES Personnel(Id_Personnel),
	FOREIGN KEY(Id_Qualification) REFERENCES Qualification(Id_Qualification)
);
```

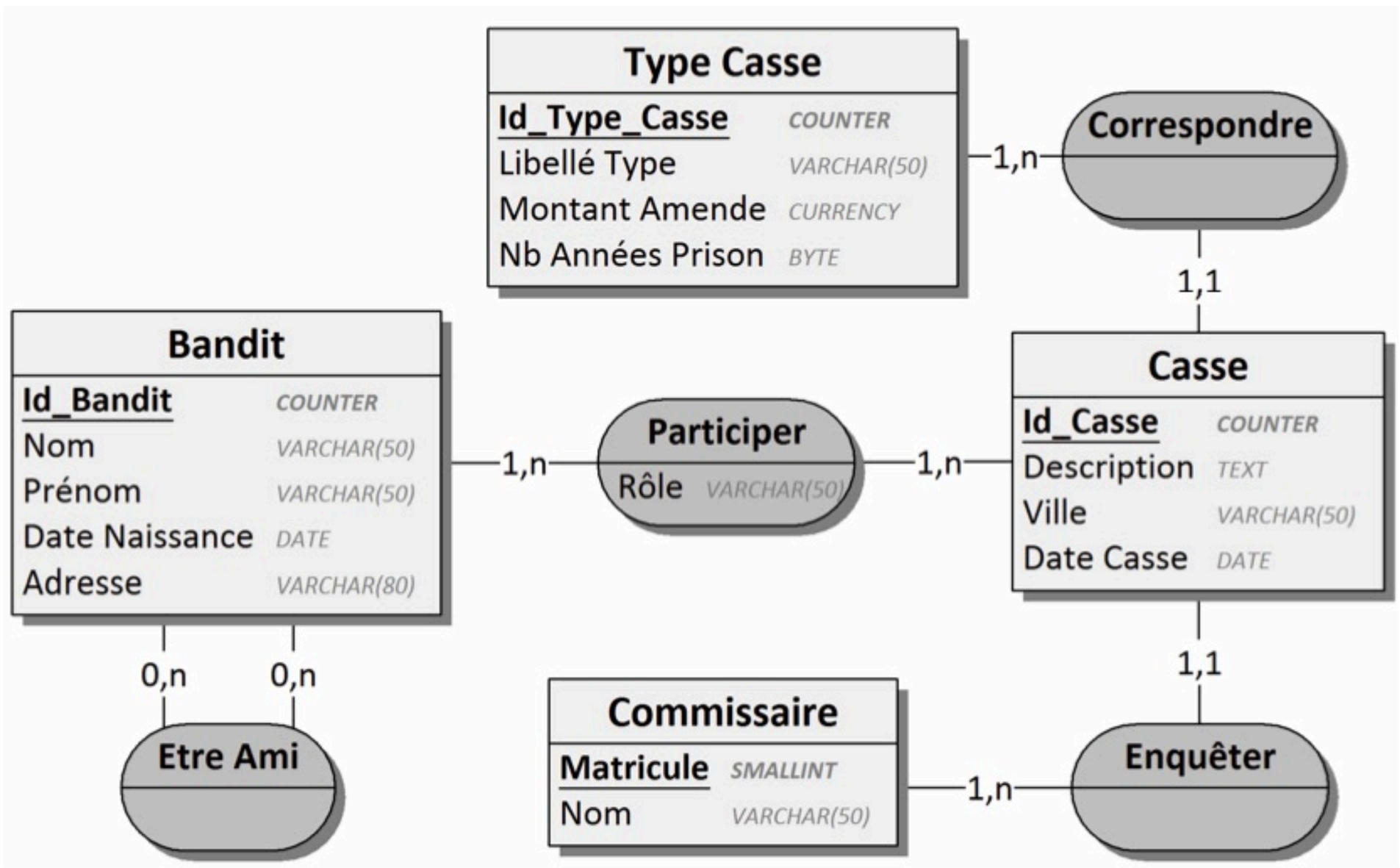
Schéma relationnel Access



© Patrick Bergougnoux

Gendarmes et voleurs

MCDi



MLD

Bandit = (**Id_Bandit** *COUNTER*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Date_Naissance *DATE*, Adresse *VARCHAR(80)*)

Type_Casse = (**Id_Type_Casse** *COUNTER*, Libellé_Type *VARCHAR(50)*, Montant_Amende *CURRENCY*, Nb_Années_Prison *BYTE*)

Commissaire = (**Matricule** *SMALLINT*, Nom *VARCHAR(50)*)

Casse = (**Id_Casse** *COUNTER*, Description *TEXT*, Ville *VARCHAR(50)*, Date_Casse *DATE*, *#Matricule, #Id_Type_Casse*)

Participer = (***#Id_Bandit, #Id_Casse***, Rôle *VARCHAR(50)*)

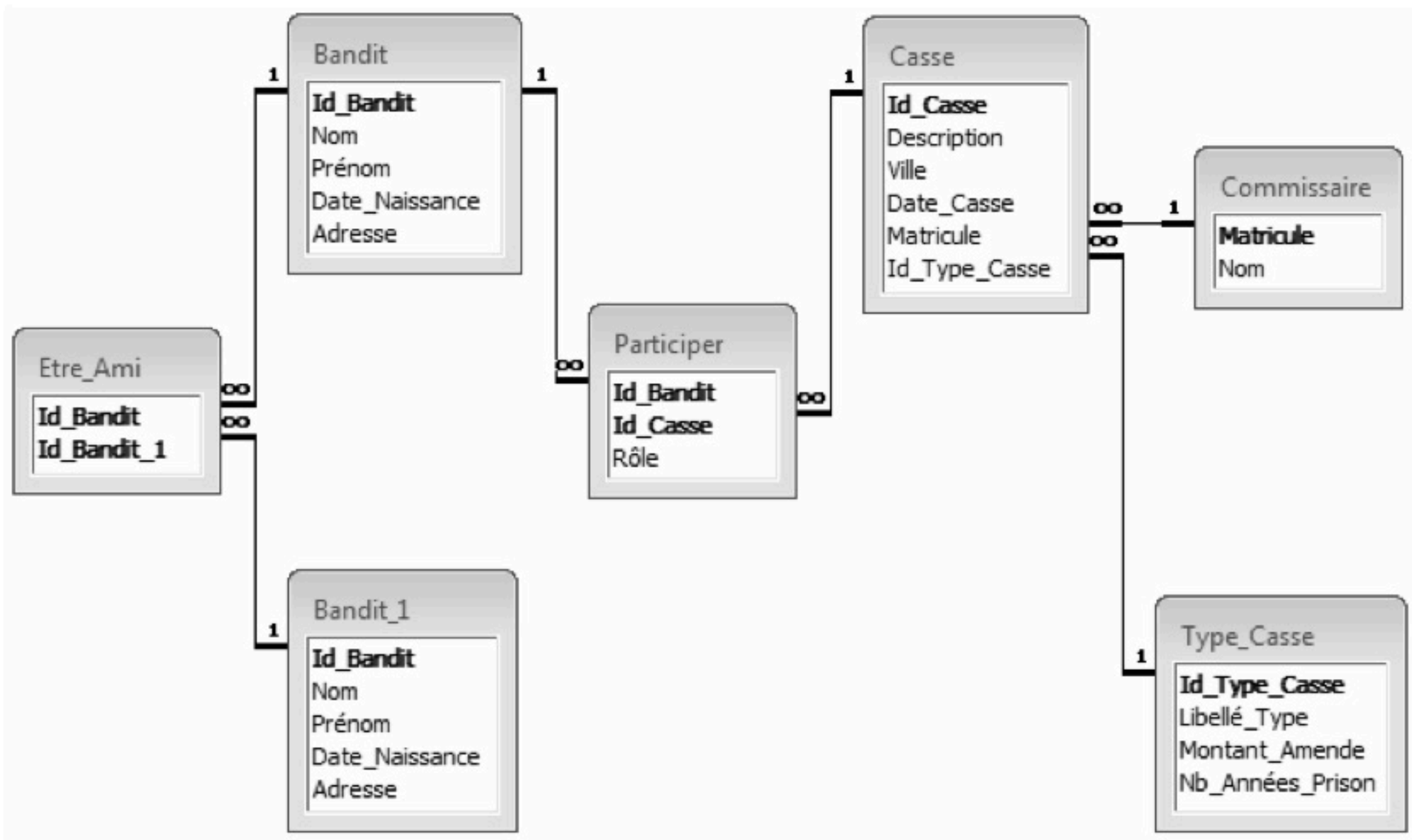
Etre_Ami = (***#Id_Bandit, #Id_Bandit_1***)

Requêtes SQL

```
CREATE TABLE Bandit(
	Id_Bandit COUNTER,
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	Date_Naissance DATE,
	Adresse VARCHAR(80),
	PRIMARY KEY(Id_Bandit)
);

CREATE TABLE Type_Casse(
	Id_Type_Casse COUNTER,
	Libellé_Type VARCHAR(50),
	Montant_Amende CURRENCY,
	Nb_Années_Prison BYTE,
	PRIMARY KEY(Id_Type_Casse)
);

CREATE TABLE Commissaire(
	Matricule SMALLINT,
	Nom VARCHAR(50),
	PRIMARY KEY(Matricule)
);

CREATE TABLE Casse(
	Id_Casse COUNTER,
	Description TEXT,
	Ville VARCHAR(50),
	Date_Casse DATE,
	Matricule SMALLINT NOT NULL,
	Id_Type_Casse INT NOT NULL,
	PRIMARY KEY(Id_Casse),
	FOREIGN KEY(Matricule) REFERENCES Commissaire(Matricule),
	FOREIGN KEY(Id_Type_Casse) REFERENCES Type_Casse(Id_Type_Casse)
);

CREATE TABLE Participer(
	Id_Bandit INT,
	Id_Casse INT,
	Rôle VARCHAR(50),
	PRIMARY KEY(Id_Bandit, Id_Casse),
	FOREIGN KEY(Id_Bandit) REFERENCES Bandit(Id_Bandit),
	FOREIGN KEY(Id_Casse) REFERENCES Casse(Id_Casse)
);

CREATE TABLE Etre_Ami(
	Id_Bandit INT,
	Id_Bandit_1 INT,
	PRIMARY KEY(Id_Bandit, Id_Bandit_1),
	FOREIGN KEY(Id_Bandit) REFERENCES Bandit(Id_Bandit),
	FOREIGN KEY(Id_Bandit_1) REFERENCES Bandit(Id_Bandit)
);
```


© Patrick Bergougnoux

Schéma relationnel Access



On remarque, dans ce dernier exercice, la façon dont Access représente les associations réflexives en dupliquant (seulement visuellement) la table « Bandit ».

Notes

Notes

Quatrième partie

Modélisation avancée

Le début de ce livre a fait état de quelques limites par rapport à certaines situations de modélisation. Cette quatrième partie vient apporter des solutions techniques permettant de régler de manière très précise tous les problèmes que l'on peut rencontrer en termes de modélisation conceptuelle de données.

Par ailleurs, jusqu'à présent, tous les modèles ont été mis à œuvre à travers le formalisme entité-association ; le dernier chapitre de cette partie introduit un autre formalisme très utilisé, le diagramme de classes UML, avec lequel nous reprendrons l'ensemble des exercices traités dans cet ouvrage.

Chapitre 13

Identifiant relatif

Le premier chapitre de cette partie va nous montrer comment faire usage de tous les éléments mis à notre disposition pour faire le meilleur choix de modélisation.

Utilisation poussée des clés étrangères

Mise en évidence de la problématique avec un exemple simple

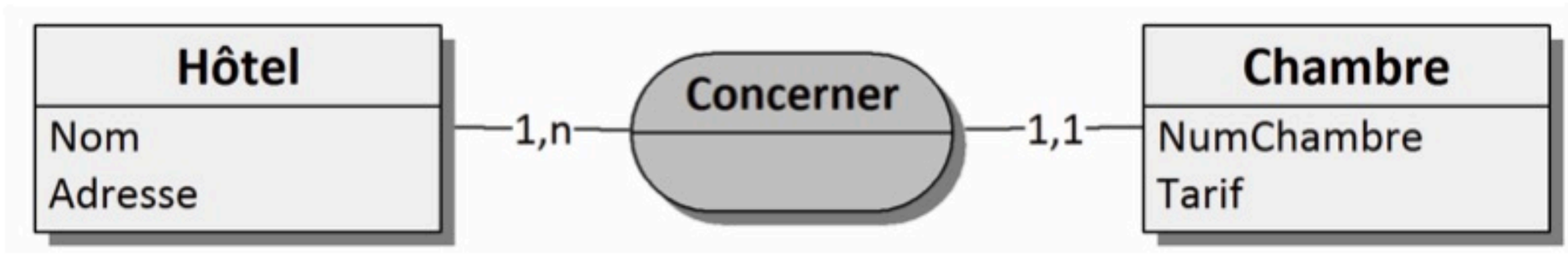
Pour illustrer une première situation, prenons un nouvel exemple avec un petit système d'information très simple ; voici le compte-rendu d'interview :

« Un office de tourisme veut répertorier l'offre de la commune en termes de chambres d'hôtels ; pour cela, il souhaite gérer seulement les informations essentielles : le nom et l'adresse des hôtels, avec le numéro et le tarif des chambres qu'ils proposent. »

Nous obtenons donc sans difficulté le dictionnaire des données suivant :

- Nom Hôtel
- Adresse Hôtel
- Numéro Chambre
- Tarif Chambre

Poursuivons la démarche en réalisant le MCD correspondant :



Avec tout ce que nous avons vu jusqu'à présent, j'espère que ce petit modèle ne pose pas de problème !

Passons maintenant à l'étape suivante avec le MCDi, et en particulier la définition des identifiants :

- Pour la classe d'entités « Hôtel », aucune rubrique ou groupe de rubriques ne convient : un « Id_Hôtel » s'impose.
- Pour la classe d'entités « Chambre », le numéro de chambre n'étant pas unique (plusieurs hôtels de la commune auront, par exemple, la chambre n°8), la procédure nous amènerait alors naturellement à définir un « Id_Chambre ».

Mais regardons de plus près ce que cela donne au niveau technique en anticipant le MLD :

Hôtel =(**Id_Hôtel** *COUNTER*, Nom *VARCHAR(50)*, Adresse *VARCHAR(80)*)

Chambre =(**Id_Chambre** *COUNTER*, NumChambre *SMALLINT*, Tarif *CURRENCY*, *#Id_Hôtel*)

N'y avait-il pas une autre solution avant d'en arriver à rajouter « Id_Chambre » ?

S'il est vrai qu'aucune rubrique de la table « Chambre » ne peut assurer seule la fonction de clé primaire, on voit clairement que la concaténation de « *#Id_Hôtel* » et « NumChambre » fait parfaitement l'affaire. En effet, cette concaténation assure l'unicité avec des rubriques qui, individuellement, possèdent les autres qualités requises pour des identifiants.

Au niveau du MLD, cela donnerait donc :

Hôtel =(**Id_Hôtel** *COUNTER*, Nom *VARCHAR(50)*, Adresse *VARCHAR(80)*)

Chambre =(***#Id_Hôtel*, NumChambre** *SMALLINT*, Tarif *CURRENCY*)

Cette solution est parfaitement cohérente et évite le rajout d'une rubrique « Id_Chambre » ne faisant pas partie du système d'information initial. De plus, et contrairement à la solution précédente, le numéro de chambre est alors forcément unique pour chaque hôtel, ce qui est parfaitement cohérent avec le système d'information.

© Patrick Bergougnoux

Mise en œuvre au sein du MCDi

Si les choses sont claires pour le MLD, un problème va se poser au niveau du MCDi : en effet, les clés étrangères ne sont pas représentées dans les MCDi puisque les associations expriment clairement les liens entre classes d'entités.

Comment exprimer dans un MCDi le fait qu'une clé étrangère participe à la clé primaire de la table dans laquelle elle se trouve ?

Rappelons tout d'abord que la participation d'une clé étrangère à une clé primaire n'est pas une nouveauté : en effet, c'est systématiquement le cas pour les tables de correspondance issues d'associations à cardinalités multiples de part et d'autre. Dans ce cas, d'une certaine façon, ce sont les cardinalités qui expriment dans le MCDi cette utilisation.

Et bien, de la même manière, ce sont les cardinalités qui devront, dans notre exemple de gestion de chambres d'hôtels, exprimer la participation de la clé étrangère « *#Id_Hôtel* » dans la clé primaire de la table « Chambre ».

Nous savons que cette situation se produit parce qu'une clé étrangère est présente dans la table associée, et donc qu'une cardinalité simple (**1,1** ou **0,1**) est présente sur une patte de l'association.

Dans le cas d'une cardinalité **0,1** la clé étrangère peut ne pas être renseignée : elle ne peut alors pas faire partie d'une clé primaire composée, celle-ci ne pouvant pas contenir un composant non renseigné.

En dehors des tables de correspondance, la seule situation où une clé étrangère peut participer à une clé primaire, est conditionnée par la présence d'une cardinalité **1,1**.

C'est donc au niveau de cette cardinalité qu'il faut indiquer si la clé étrangère présente dans la table associée doit ou non participer à la clé primaire, c'est à dire faire partie ou pas de l'identifiant.

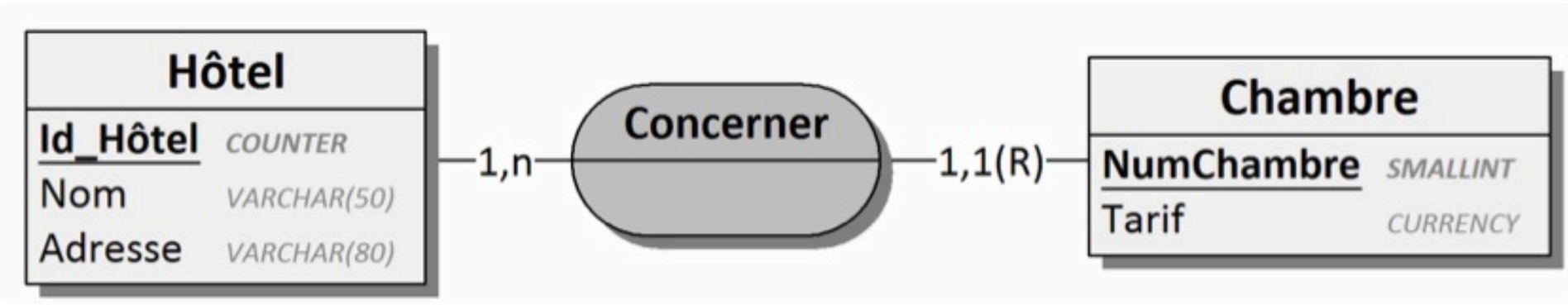
Considérant que cette composante de l'identifiant est relative à la classe d'entités associée, on parle alors d'**identifiant relatif**.

Le formalisme entité-association propose deux façons d'indiquer la présence de ces identifiants relatifs :

- Soit la cardinalité est mise entre parenthèses ➔ **(1,1)**
- Soit la cardinalité est suivie de l'indicateur « R » ➔ **1,1 (R)**

Dans cet ouvrage, nous opterons pour la deuxième solution qui met davantage en évidence l'identification relative.

Voici le MCDi correspondant pour notre exemple :



Requêtes SQL

Concernant les requêtes SQL de création des tables, cela reste tout à fait classique ; on note cependant que, la clé étrangère étant une composante de la clé primaire, il est inutile de lui rajouter la contrainte « NOT NULL » :

```
CREATE TABLE Hôtel(
        Id_Hôtel COUNTER,
        Nom VARCHAR(50),
        Adresse VARCHAR(80),
        PRIMARY KEY(Id_Hôtel)
);

CREATE TABLE Chambre(
        Id_Hôtel INT,
        NumChambre SMALLINT,
        Tarif CURRENCY,
        PRIMARY KEY(Id_Hôtel, NumChambre),
        FOREIGN KEY(Id_Hôtel) REFERENCES Hôtel(Id_Hôtel)
);
```

Un usage très spécifique

Important : cette possibilité reste très exceptionnelle ! Le principal danger après avoir étudié les identifiants relatifs est de vouloir en mettre partout ; or leur usage est très limité.

D'ailleurs, dans les huit exercices réalisés au chapitre précédent, aucun n'était susceptible d'utiliser des identifiants relatifs.

Seul l'exercice sur la location de logements pourrait être concerné si l'on considérait que le numéro de logement était propre à une commune : un même numéro pourrait alors se retrouver dans plusieurs communes, et l'identifiant de la classe d'entités « Commune » devrait être rajouté en tant qu'identifiant relatif de la classe « Logement ».

© Patrick Bergougnoux

Décomposition d'une association

Problèmes d'historisation

Nous avons vu dans l'exercice sur la bibliothèque universitaire au chapitre 8, qu'un étudiant ne pouvait pas emprunter un ouvrage qu'il avait déjà emprunté précédemment. En effet, comme le confirme la clé primaire de la table de correspondance « Emprunter » dans le MLD, pour un étudiant donné et un ouvrage donné, il n'y a qu'une seule date de prêt et une seule date de retour :

Emprunter = (***#NumCarte, #Code_Ouvrage***, Date_Prêt *DATETIME*, Date_Retour *DATETIME*)

Par conséquent, si un étudiant souhaite emprunter un ouvrage qu'il a déjà emprunté, la seule possibilité consiste à remplacer l'ancien prêt par le nouveau, ce qui est gênant, et totalement inenvisageable si le précédent emprunt n'a pas été rendu.

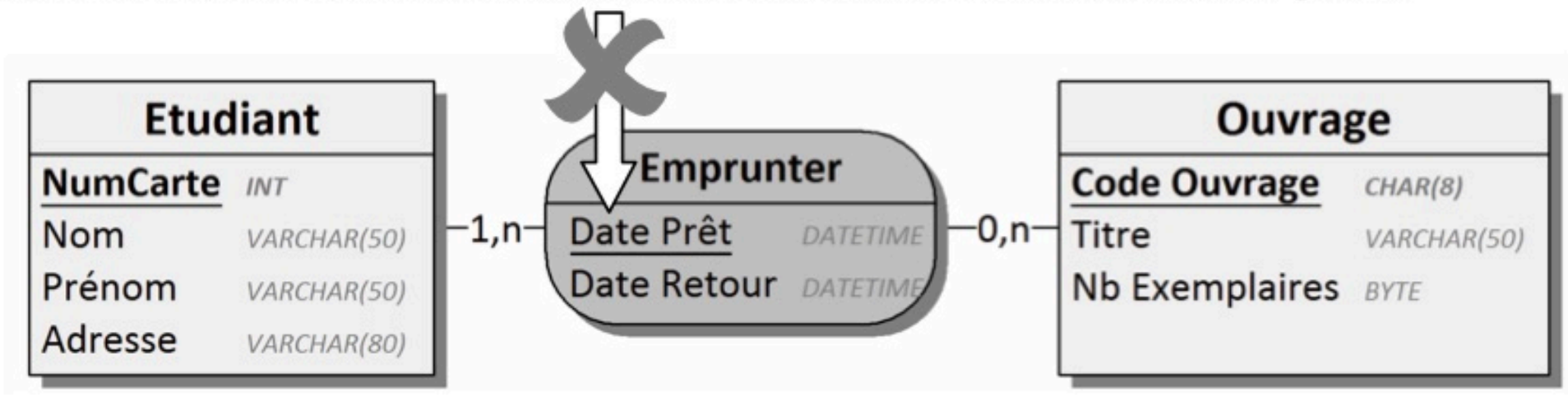
Comment faire pour permettre la gestion de cet historique des prêts ?

Si, comme précédemment, nous raisonnons directement au niveau du MLD, la solution consiste à ajouter la date de prêt dans la composition de la clé primaire. Ainsi, un même étudiant pourra emprunter plusieurs fois le même ouvrage à des dates différentes (en rajoutant l'heure, c'est encore plus précis). Le MLD devient alors :

Emprunter = (***#NumCarte, #Code_Ouvrage*, Date_Prêt** *DATETIME*, Date_Retour *DATETIME*)

La question est la même que précédemment : comment faire apparaitre cette solution dans le MCDi ?

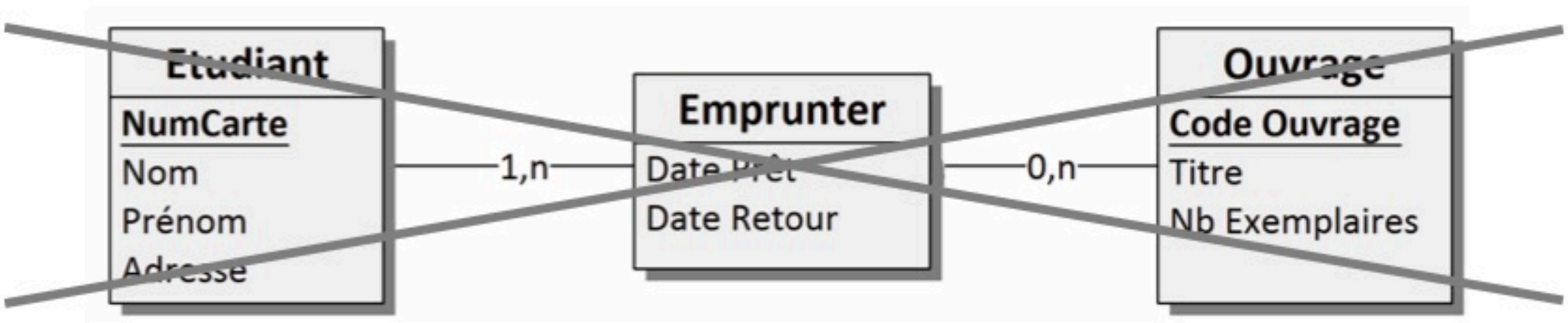
Une solution consisterait à souligner la date de prêt dans l'association du MCDi… **mais le formalisme entité-association ne l'autorise pas…**



En effet, cela donnerait l'impression que la rubrique « Date Prêt » est un identifiant à elle toute seule, alors qu'elle n'en est qu'une simple composante.

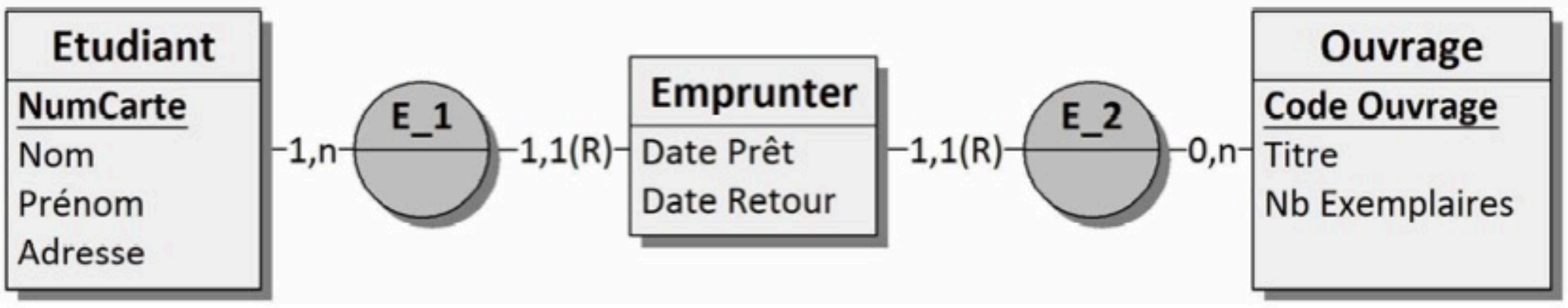
Transformation d'une association en classe d'entités

Une solution consiste alors à décomposer l'association pour générer une classe d'entités dans laquelle il sera possible de désigner la date de prêt comme une composante de la clé primaire. **Pour clarifier les modèles, les types de données sont masqués dans les MCDi et les MLD suivants** :

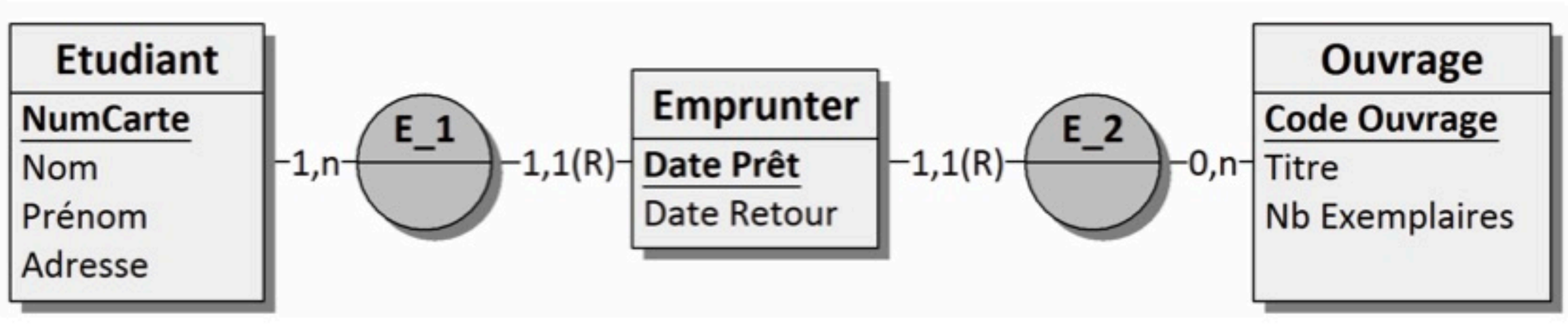


Cette solution n'est pas non plus conforme au formalisme : deux classes d'entités ne peuvent être reliées qu'à travers une association. Il faut donc rajouter des associations de part et d'autre de cette nouvelle classe d'entités.

Concernant les cardinalités de ces associations, l'utilisation de cardinalités **1,1** avec identifiants relatifs va nous permettre d'obtenir un MCDi conforme au MLD recherché :



Le modèle obtenu donne un MLD strictement équivalent à celui du MCDi avant décomposition. Bien que graphiquement plus lourd, il va permettre d'aller plus loin dans la modélisation, en particulier pour la gestion de l'historisation. Dans notre exemple, nous pouvons ainsi rajouter la rubrique « Date Prêt » à l'identifiant de la classe « Emprunter » :

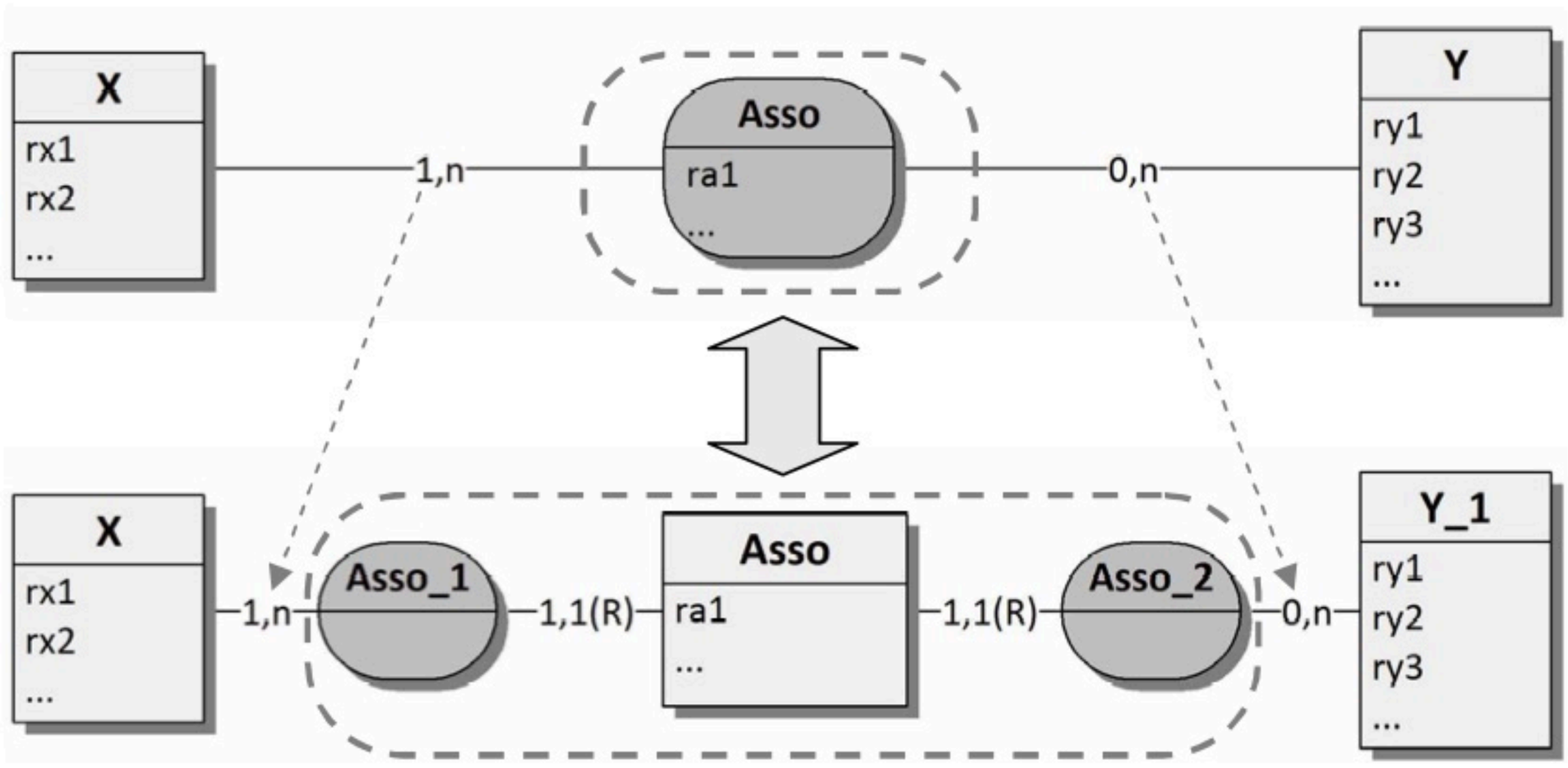


Nous obtenons alors un MCDi permettant la gestion de l'historique des emprunts d'un étudiant, y compris lorsqu'il emprunte plusieurs fois le même ouvrage.

© Patrick Bergougnoux

Équivalence stricte de la décomposition

L'exemple que nous venons de traiter peut bien évidemment être généralisé. Ainsi, les deux modélisations suivantes sont toujours strictement équivalentes, génèrent donc le même MLD et, par conséquent, le même schéma relationnel de la base de données.



Les étapes de cette transformation sont simples :

- Remplacement de l'association par une classe d'entités en conservant le même nom et les mêmes rubriques éventuelles.
- Rajout des deux associations de part et d'autre de la nouvelle classe d'entités.
- Reprise des cardinalités existantes sur les pattes du côté des classes d'entités initiales.
- Rajout des cardinalités **1,1 (R)** sur les deux pattes du côté de la nouvelle classe d'entités.

L'utilisation des identifiants relatifs permet d'intégrer les clés étrangères issues des classes associées dans la clé primaire de la table « Asso », comme c'était déjà le cas lorsque « Asso » était une association à cardinalités multiples de part et d'autre.

Looping propose une fonction de décomposition automatique des associations à cardinalités multiples de part et d'autre. Pour cela, dans la fenêtre de définition des associations, il suffit d'utiliser le bouton

.

Associer une classe d'entités à une association

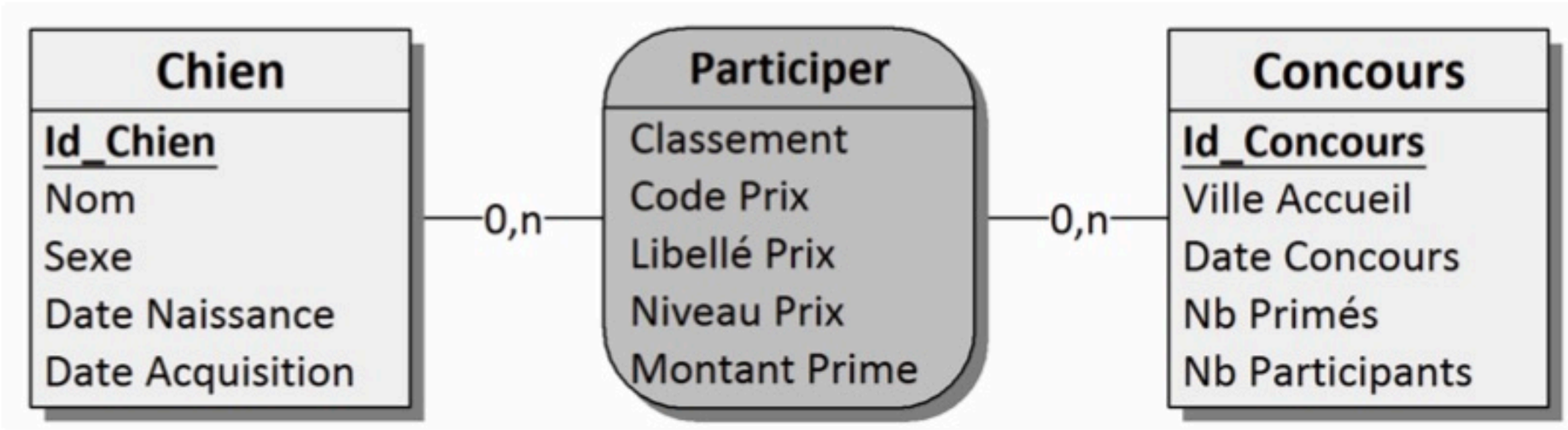
La décomposition va permettre de régler d'autres situations et ainsi apporter une solution à tout problème de modélisation conceptuelle des données d'un système d'information.

Pour illustrer ces nouvelles problématiques et les solutions qui vont avec, reprenons l'exercice sur la société canine en rajoutant un complément au compte-rendu d'interview :

« Tout chien primé à un concours reçoit un prix caractérisé par un code, un libellé (médaille, accessit, …), un niveau (or, argent, …) et le montant de la prime attribuée à son propriétaire (par exemple : 500 euros pour une médaille d'or). »

Le prix est composé des trois nouvelles rubriques indiquées dans le compte-rendu d'interview : le libellé du prix, son niveau et la prime correspondante.

Une première idée de modélisation pourrait être la suivante (on se limite dans cette démarche aux classes « Chien » et « Concours », les classes « Propriétaire » et « Race » n'étant pas concernées par l'évolution) :



Mais, après vérification des règles de cohérence des modèles étudiés au chapitre 7, on se rend compte que la règle 3 doit être appliquée : « *Si une rubrique possède la même valeur pour plusieurs entités d'une même classe, il faut envisager que cette rubrique fasse l'objet d'une classe à part : cette règle est appliquée si et seulement si plusieurs rubriques caractérisent la nouvelle classe en question* ».

Or, de nombreuses participations (associations entre tous les chiens participants et les concours) font l'objet d'un même prix, avec un même code, un même libellé, un même niveau et une même prime. Les rubriques qui caractérisent le prix sont donc au nombre de quatre, et doivent alors faire l'objet d'une classe à part.

Cette règle 3 que nous avons l'habitude d'appliquer aux classes d'entités doit s'appliquer de la même manière pour les associations.

© Patrick Bergougnoux

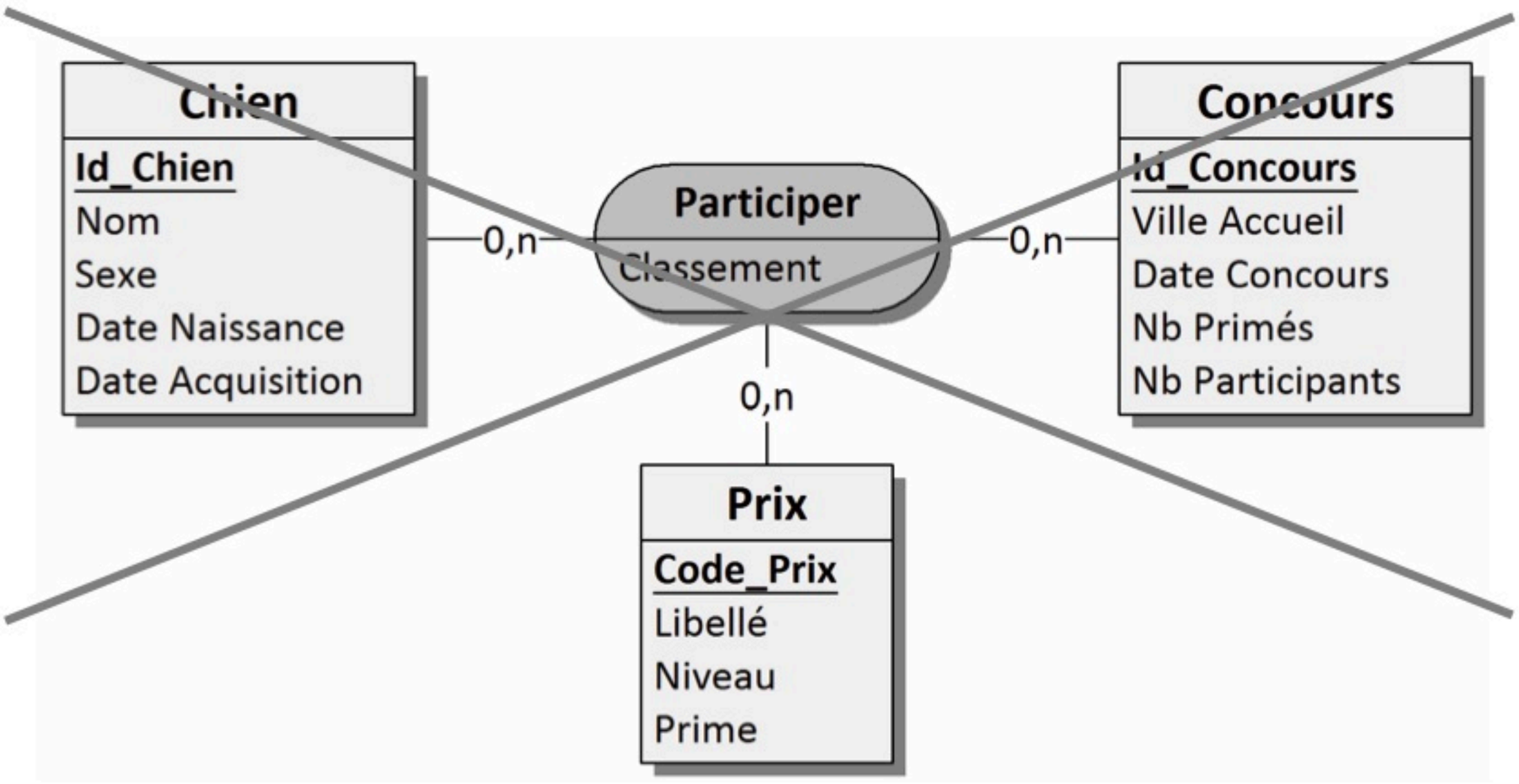
Une nouvelle classe d'entités « Prix » doit donc être rajoutée au modèle : se pose alors la question de son association avec les autres composantes du MCDi.

Cette classe « Prix » ne peut pas être directement associée à la classe « Chien » ; en effet, en fonction de ses participations à différents concours, un chien obtiendra des prix différents, ou pas de prix du tout.

Le prix dépend donc également du concours, mais la classe « Prix » ne peut pas non plus être associée directement à la classe « Concours » ; en effet, les prix accordés dans le cadre d'un concours dépendent bien sûr aussi du chien.

Par conséquent, un prix dépend de la participation d'un chien donné à un concours donné : c'est d'ailleurs ce que l'on perçoit déjà sur la première idée de modélisation qui positionne les quatre nouvelles rubriques dans l'association « Participer ». Pour continuer à respecter cela, il nous faut donc associer la classe d'entités « Prix »… à l'association « Participer »…

Et voilà ce qui peut alors émerger comme idée :



Le problème est que ce modèle est totalement faux ! L'association de plus de deux classes d'entités est certes possible, mais seulement dans des cas bien spécifiques que nous traiterons plus tard.

Pour comprendre l'erreur, projetons-nous au niveau du MLD :

Participer = (***#Id_Chien, #Id_Concours, #Code_Prix,*** Classement *SMALLINT*)

Avec ce MLD, on se rend compte que le classement d'un chien à un concours dépend également du prix… qu'il recevra peut-être… ou peut-être pas…

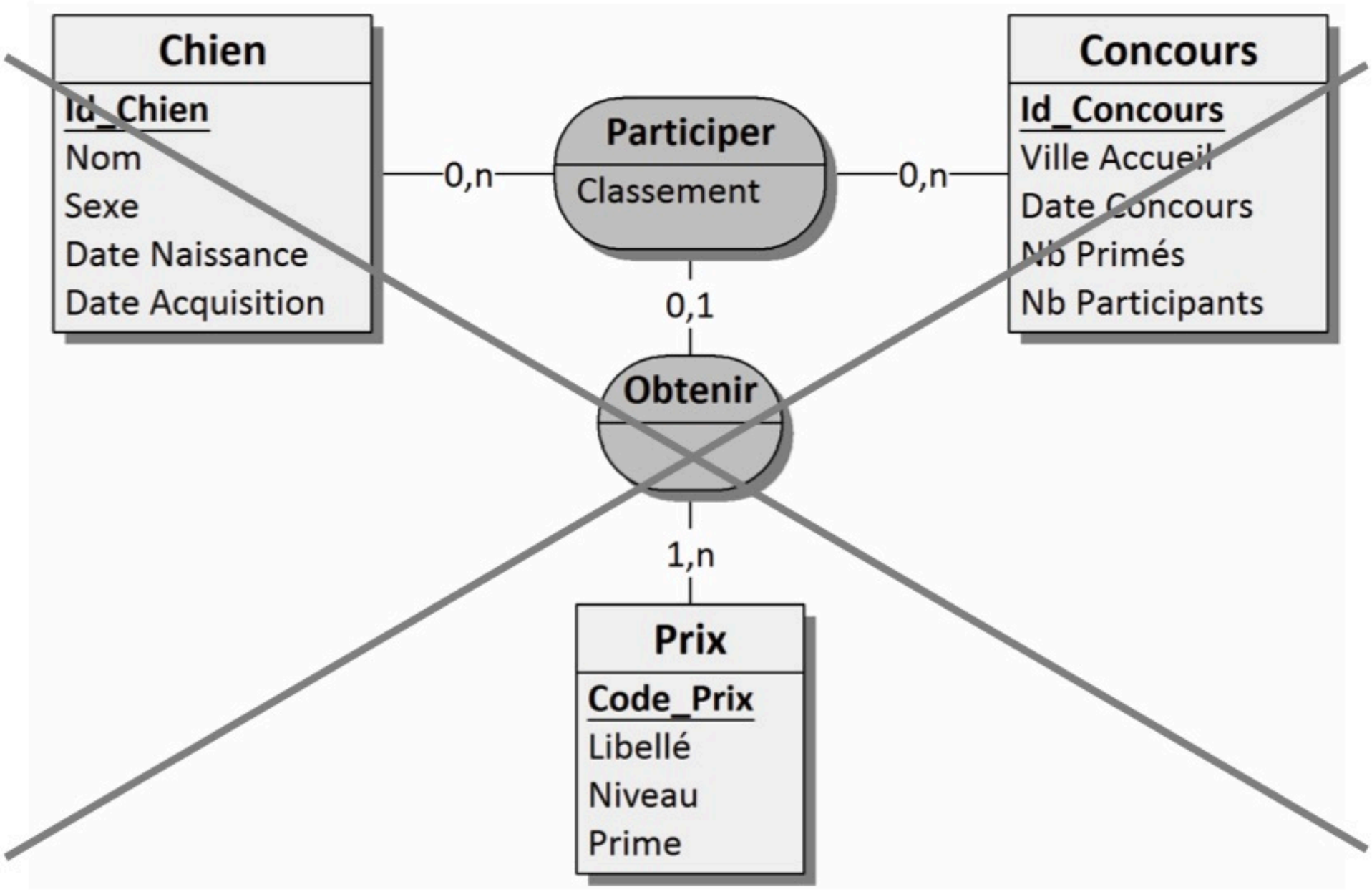
Or, c'est la participation qui doit déterminer le prix, et celui-ci ne doit bien évidemment pas être nécessaire à la détermination du classement.

Ce type d'association ternaire, appelée également « tripatte » (ou plus généralement n-aire), associant plus de deux classes d'entités, doit faire l'objet de la plus grande méfiance ; leur usage, même s'il n'est pas interdit, est très délicat et n'a d'intérêt que dans de rares situations. Nous étudierons ces associations n-aires au chapitre suivant.

Pour notre exemple, la solution est ailleurs : il faut pouvoir associer **0** ou **1** prix à une participation ; le MLD devra donc proposer une table « Participer » avec la structure suivante :

Participer = (*#Id_Chien, #Id_Concours*, Classement, *#Code_Prix)**

Le modèle qui pourrait répondre au mieux à cette spécification serait le suivant :

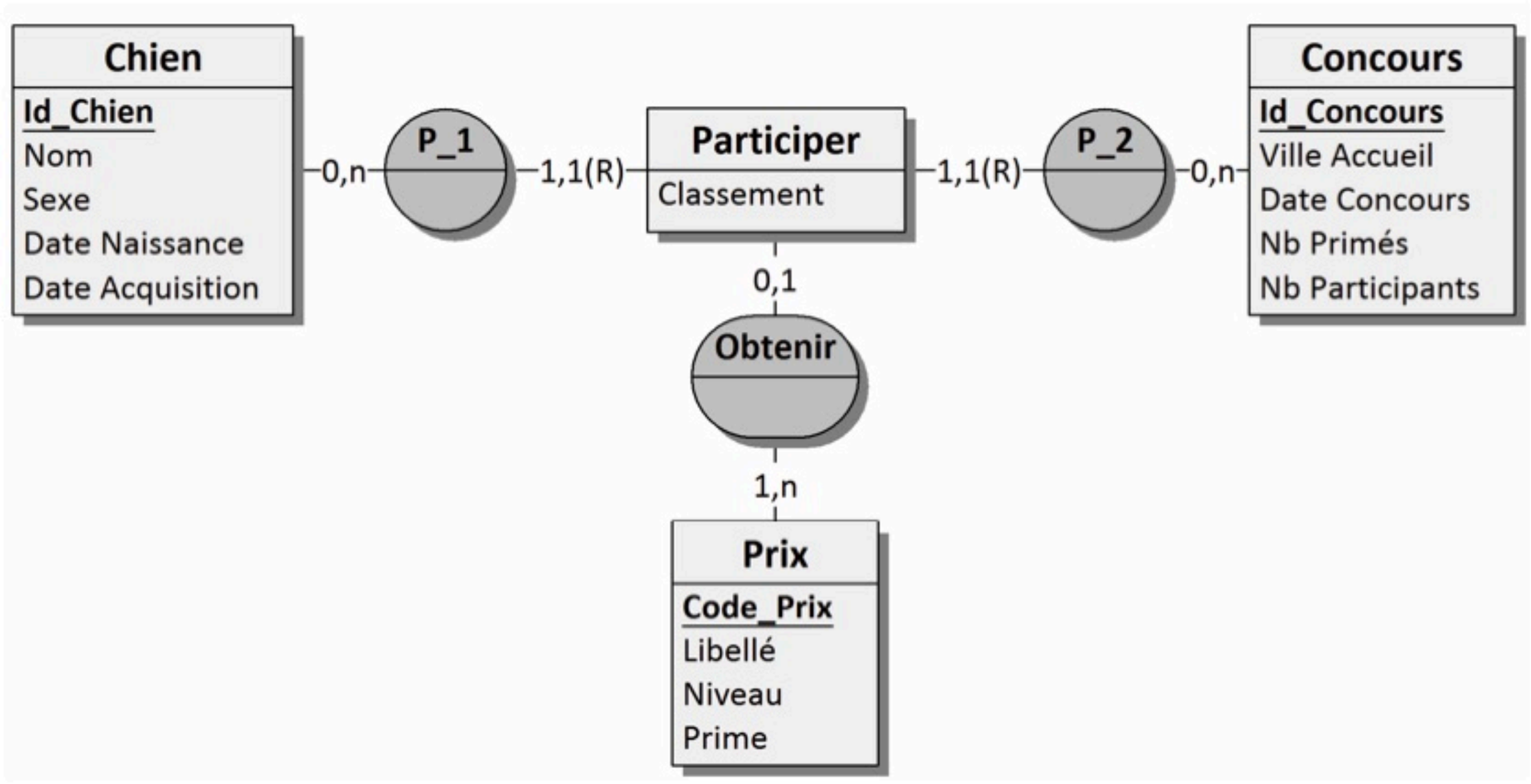


Conceptuellement, cela répond à la problématique du système d'information… **mais le formalisme entité-association ne l'autorise pas**… et il faut reconnaitre que cela génèrerait une certaine confusion dans la compréhension du modèle.

Comme précédemment, la solution passe par une décomposition de l'association « Participer », ce qui permettra ensuite d'assurer correctement l'association de la classe d'entités « Prix » avec la nouvelle classe générée par la décomposition, tout en respectant le formalisme entité-association.

© Patrick Bergougnoux

Voici le MCDi obtenu après décomposition :



L'association de la classe « Prix » avec la nouvelle classe « Participer » générée par la décomposition respecte alors parfaitement le formalisme entité-association. Le MLD correspondant est conforme aux exigences du système d'information. Les requêtes SQL permettant la génération du schéma relationnel n'ont rien de spécifique.

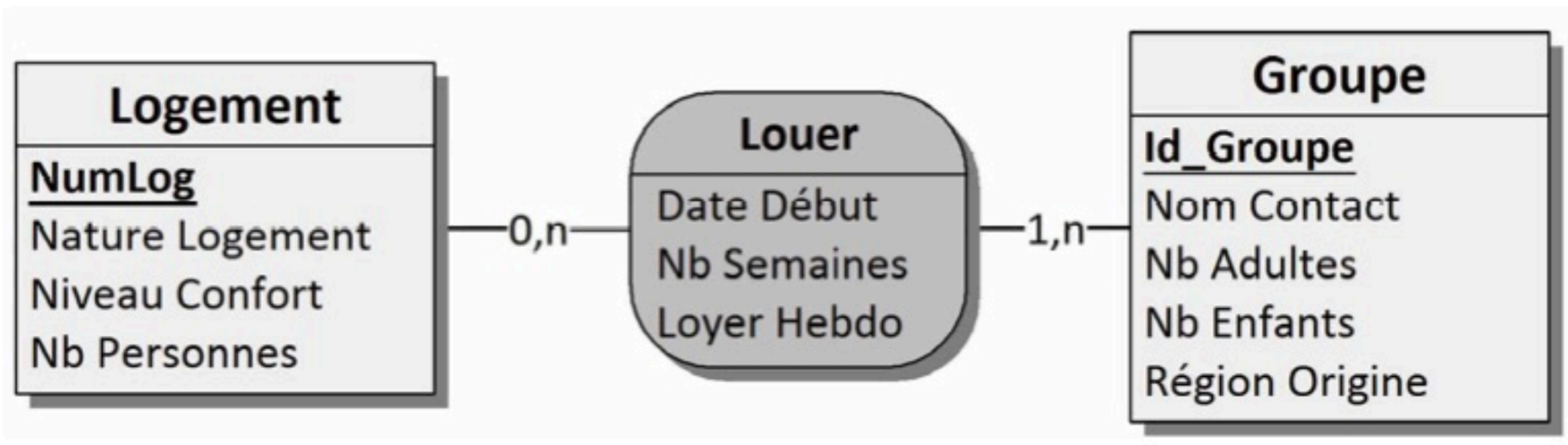
Gestion de l'irréductibilité

Le fait de rajouter des rubriques à l'identifiant pose la question de l'irréductibilité de l'identifiant composé obtenu.

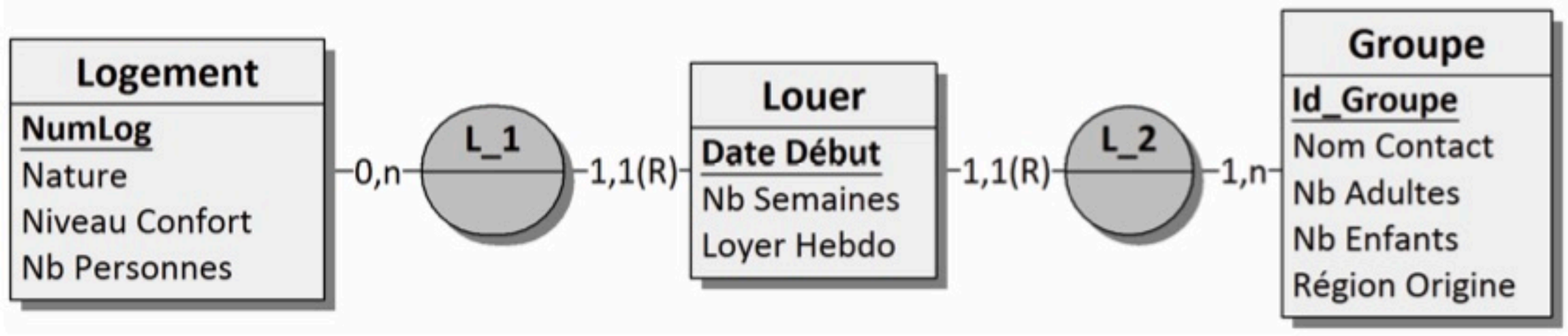
Au niveau du MLD se pose la question : **la clé primaire est-elle minimale** ?

Réduction éventuelle d'une clé primaire

Pour illustrer cette problématique, étudions le cas de la location des logements traité précédemment en reprenant un extrait du MCDi :



Avec cette version du MCDi, se pose le problème de l'historisation des locations ; en effet, le problème est identique à celui de la société canine : pour un groupe donné et un logement donné, il ne peut y avoir qu'une seule location (une seule date de début, un seul nombre de semaines et un seul loyer hebdomadaire). Le MCDi ne permet donc pas de gérer l'historique des locations si un groupe veut louer plusieurs fois le même logement : pour cela, il faut décomposer l'association « Louer » et rajouter « Date Début » à l'identifiant :



Le MLD correspondant à la table d'association est le suivant :

Louer = (***#NumLog, #Id_Groupe***, **Date_Début**, Nb_Semaines, Loyer_Hebdo)

Le problème d'historisation est ainsi réglé, mais cette clé composée est-elle minimale, et l'irréductibilité de l'identifiant est-elle donc respectée ?

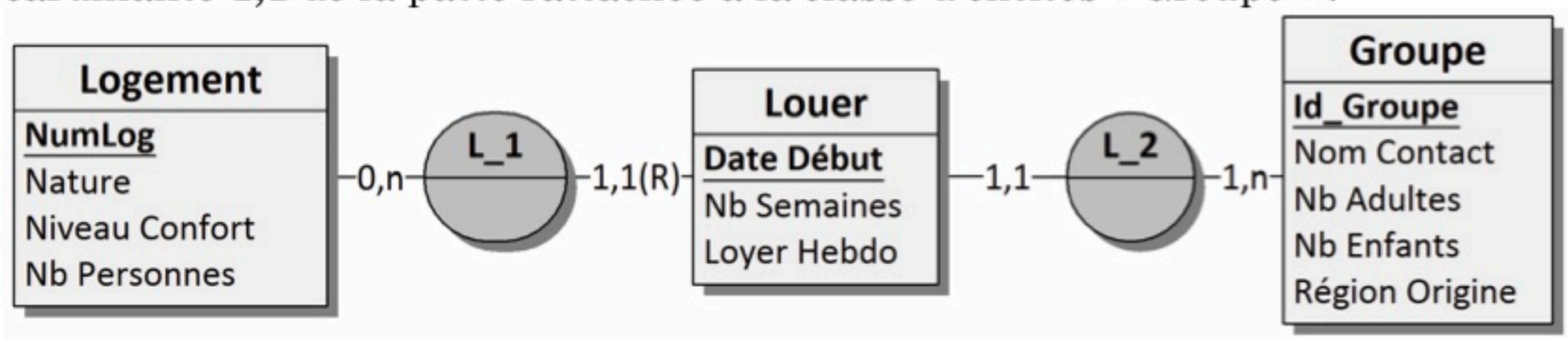
Pour le vérifier, il suffit d'enlever, chacune à leur tour, une des rubriques composant la clé et de s'assurer que l'unicité est toujours respectée :

- **Date_Début** : c'est la rubrique que l'on vient de rajouter pour assurer la gestion de l'historisation des locations ➔ ***à conserver***.
- ***#NumLog*** : si on le retire de l'identifiant, un même groupe ne pourra pas louer plusieurs logements en même temps, ce qui peut être gênant surtout pour un groupe important ➔ ***à conserver***.
- ***#Id_Groupe*** : sans cette rubrique dans l'identifiant, plusieurs groupes ne pourront pas louer un même logement à la même date... ce qui est effectivement préférable ! Par conséquent, cette rubrique peut (et doit) être retirée de l'identifiant.

Au niveau du MLD, on obtient alors :

Louer = (***#NumLog*, Date_Début**, Nb_Semaines, Loyer_Hebdo, *#Id_Groupe*)

Pour le MCDi, il suffit de retirer la propriété d'identifiant relatif sur la cardinalité **1,1** de la patte rattachée à la classe d'entités « Groupe » :



© Patrick Bergougnoux

Clé primaire et dépendance fonctionnelle

Lorsqu'une rubrique participe à la clé primaire, cela signifie que l'on doit connaitre sa valeur pour obtenir la valeur des autres rubriques de la table.

Chaque valeur de la clé primaire détermine une et une seule valeur pour toutes les autres rubriques de la table qui dépendent alors fonctionnellement de cette clé primaire.

Pour notre exemple, nous avons donc les dépendances fonctionnelles suivantes :

- **NumLog** ⟶ Nature, Niveau_Confort, Nb_personnes
- **Id_Groupe** ⟶ Nom_Contact, Nb_Adultes, Nb_Enfants, Région_Origine
- ***#NumLog*, Date_Début** ⟶ Nb_Semaines, Loyer_Hebdo, *#Id_Groupe*

Pour la table « Louer », cela signifie qu'en connaissant le numéro de logement et la date de début, on peut obtenir les autres informations de la table, y compris l'identifiant du groupe occupant.

Si la clé minimale n'est pas respectée et que « *#Id_Groupe* » fait partie de la clé primaire, il faudra connaitre ce groupe pour obtenir les autres informations, ce qui est inutile et, par conséquent, erroné.

En fait, ne pas respecter une clé minimale, c'est demander au système de solliciter une information qu'il est censé fournir.

Cette vérification de la clé minimale est donc très importante et doit être effectuée systématiquement en cas de clé composée.

Récapitulatif sur l'usage des identifiants relatifs

Une solution répondant à tout type de modélisation

D'une façon générale, les identifiants relatifs, qu'ils soient issus d'une décomposition ou non, offrent des possibilités supplémentaires de construction d'identifiants composés. Ces possibilités permettent ainsi de ne pas avoir recours au rajout d'une rubrique n'appartenant pas au système d'information.

De plus, leur utilisation dans le cadre d'une décomposition d'association permet, si on le souhaite, d'éviter les associations n-aires qui, comme nous le verrons au chapitre suivant, sont souvent sujettes à problèmes.

Enfin, toujours suite à une décomposition, les identifiants relatifs associés à une date, offrent de bonnes possibilités pour la gestion des historisations.

Rappel : l'utilisation des identifiants relatifs reste spécifique et très limitée ; attention à ne pas vouloir les utiliser sans une bonne raison.

Transgresser en connaissance de cause

Il peut arriver qu'un identifiant composé de nombreuses rubriques atteigne un tel niveau de complexité qu'il devient alors préférable de créer un nouvel identifiant simple de type « COUNTER ». Ce niveau de complexité reste à l'appréciation du concepteur.

La règle de base consiste à tout faire pour éviter le rajout de ces identifiants inconnus du système d'information initial… mais il est tout de même possible de s'autoriser à transgresser cette règle en connaissance de cause, comme nous l'avons déjà vu pour la conservation de rubriques calculables *(cf. chapitre 3)*.

Il faut donc choisir entre une clé composée qui peut être lourde à gérer et le rajout d'un identifiant, certes plus simple, mais créé pour des raisons purement techniques. L'essentiel est d'avoir conscience de l'existence de cette alternative.

Dans la suite de cet ouvrage, nous respecterons la règle de base et nous éviterons, chaque fois que cela sera possible, le rajout de rubriques non issues du système d'information initial.

Ce qu'il faut retenir

- Les identifiants relatifs permettent l'utilisation des clés étrangères pour composer la clé primaire d'une table.
- La transformation d'une association en classes d'entités avec identifiants relatifs offre de larges possibilités de modélisation, en particulier pour la gestion des historisations.
- À chaque rajout d'une rubrique à un identifiant composé, il est essentiel de s'assurer de l'irréductibilité de l'identifiant obtenu, et donc du respect de la clé minimale.
- Lorsque la clé composée devient trop complexe, la création d'un nouvel identifiant peut être envisagée ; la règle de base privilégiant l'utilisation des rubriques du système d'information initial peut alors être transgressée, mais en connaissance de cause.

© Patrick Bergougnoux

Chapitre 14
Association n-aire

Jusqu'à présent, cet ouvrage a seulement fait état d'associations reliant deux classes d'entités. Ces associations binaires représentent une très large majorité des associations présentes dans les modèles de données.

Il est cependant possible d'associer plus de deux classes d'entités mais, même s'il n'y a pas, en théorie, de limite sur l'arité d'une association, dans la pratique on ne va que très rarement au-delà de trois. Les associations de degré supérieur à deux sont d'ailleurs plus difficiles à manipuler et à interpréter, notamment au niveau des cardinalités.

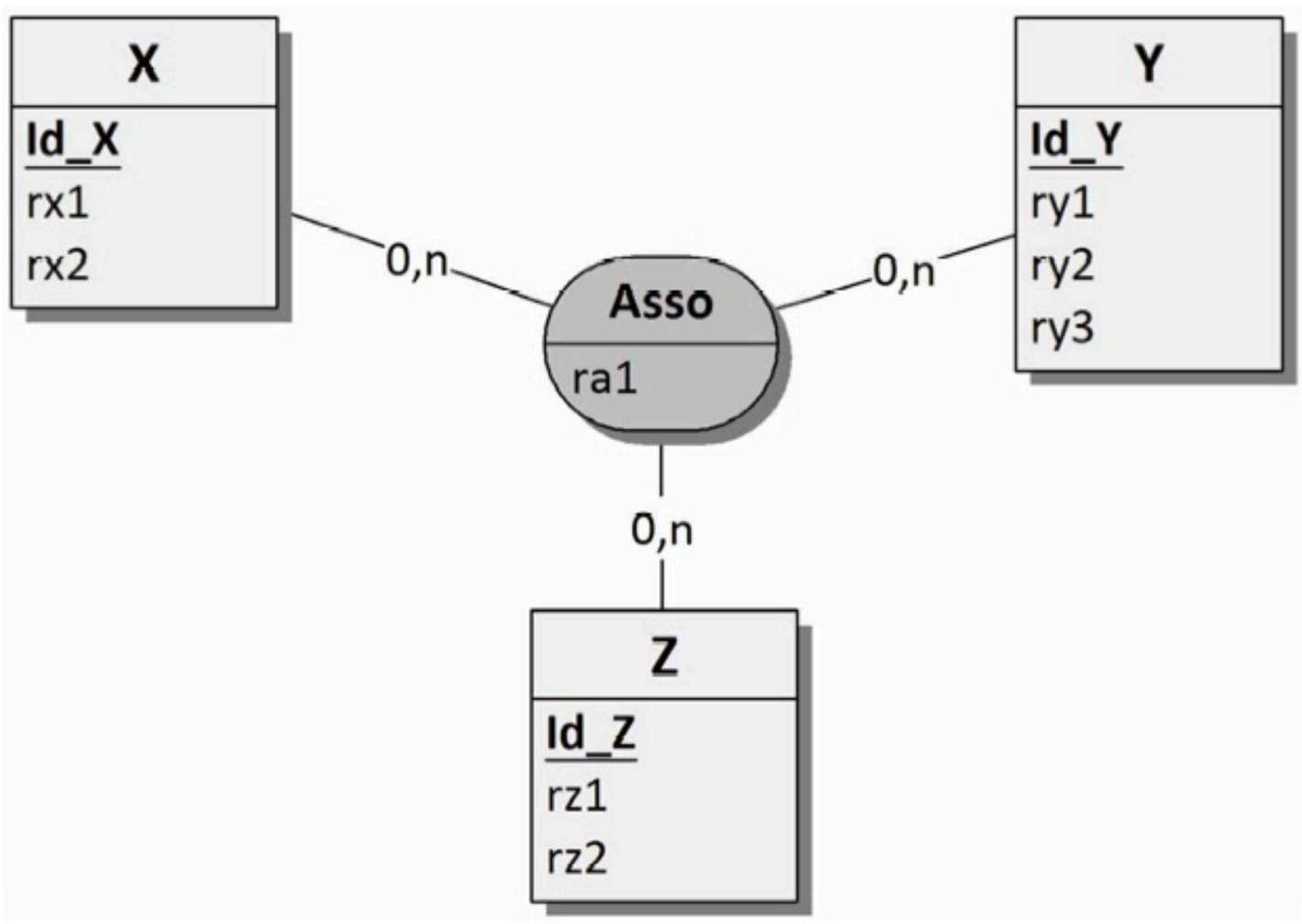
De plus, comme nous l'avons déjà abordé au chapitre précédent, nous verrons que la décomposition des associations ainsi que l'utilisation des identifiants relatifs permettent de réaliser la totalité des modèles avec des associations binaires.

Ce chapitre présente donc les possibilités de modélisation avec des associations n-aires (nous nous contenterons d'associations ternaires : les fameuses « tripattes »), tout en mettant en garde le concepteur sur leur utilisation ; en effet, les conséquences sont souvent importantes et leur usage est rarement conforme aux exigences techniques du modèle relationnel, en particulier en termes de respect des dépendances fonctionnelles et des clés minimales.

Les risques d'utilisation d'associations n-aires

Représentation graphique

La représentation graphique d'une association n-aire est classique :



On constate immédiatement que la lisibilité n'est pas évidente : en effet, la cardinalité portée par la patte rattachée à une classe d'entités définit forcément des liens identiques entre cette classe et **toutes** les autres classes associées. Cela entraine de nombreuses limites mais, surtout, un risque important de mauvaise conception du modèle.

MLD d'une association ternaire

Le MLD correspondant à l'exemple ci-dessus est le suivant :

X = (**Id_X**, rx1, rx2)

Y = (**Id_Y**, ry1, ry2, ry3)

Z = (**Id_Z**, rz1, rz2)

Asso = (***#Id_X, #Id_Y, #Id_Z***, ra1)

L'ensemble des clés étrangères issues des tables associées composent la clé primaire de la table de correspondance : le principe est exactement le même que pour les associations binaires.

Compte-tenu du nombre de composantes de cette clé primaire, le premier réflexe doit être de vérifier qu'il s'agit bien d'une clé minimale.

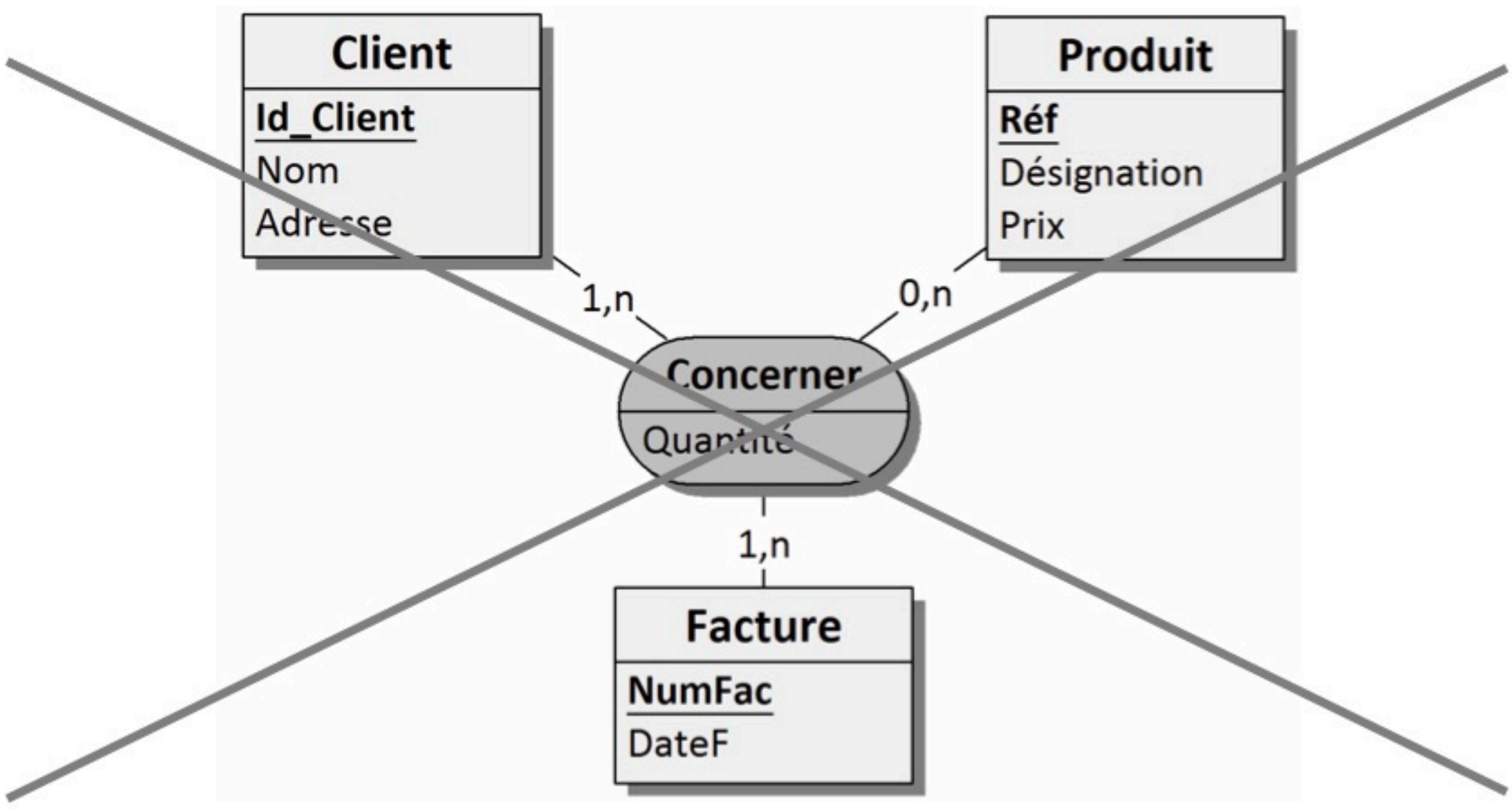
© Patrick Bergougnoux

Exemples de mauvaise utilisation d'associations n-aires

Le problème des associations n-aires est qu'elles peuvent donner l'impression, au niveau conceptuel, d'être la solution à tous les problèmes : on prend les classes d'entités, on relie tout ça ensemble autour d'une association et le tour est joué !

Le danger est qu'une lecture de tels modèles par un non-spécialiste ne le choquera pas forcément.

Prenons quelques exemples de mauvaises utilisations de tripattes, en commençant par notre modèle de base, la facturation Beaux Rivages :



Montrez ce modèle à une personne n'ayant pas de bonnes notions de modélisation de données : ça peut passer !

Par contre, maintenant, projetons-nous au niveau du MLD correspondant :

Client = (**Id_Client**, Nom, Adresse)
Facture = (**NumFac**, DateF)
Produit = (**Réf**, Désignation, Prix)
Concerner = (***#Id_Client, #NumFac, #Réf***, Quantité)

Et prenons un tout petit jeu d'essai pour la table « Concerner » :

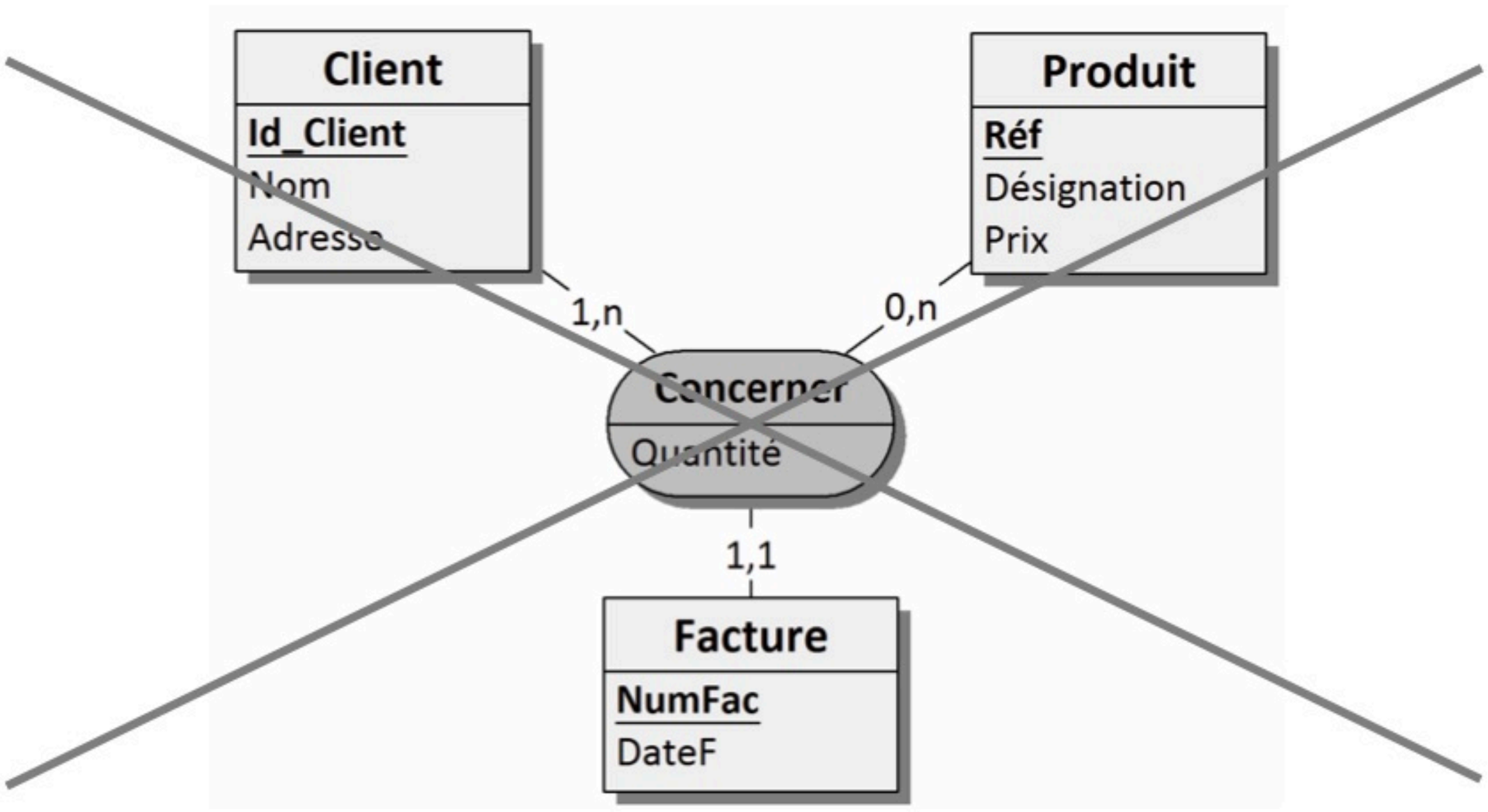
- (**1, 123, A12**, 15)
- (**2, 123, B13**, 10)

Stop ! On vient déjà de souligner une erreur : en effet, alors que l'unicité de la clé composée a bien été respectée, la facture 123 concerne à la fois le client 1 et le client 2 !

Le modèle est bien évidemment faux. Une bonne lecture du MCDi aurait déjà dû permettre de constater l'erreur : « *Chaque facture concerne **1** à **n** clients* » !

En fait, le problème de ces associations n-aires est qu'il est souvent très difficile d'appréhender correctement les cardinalités.

En effet, pour régler le problème précédent, on pourrait écrire :

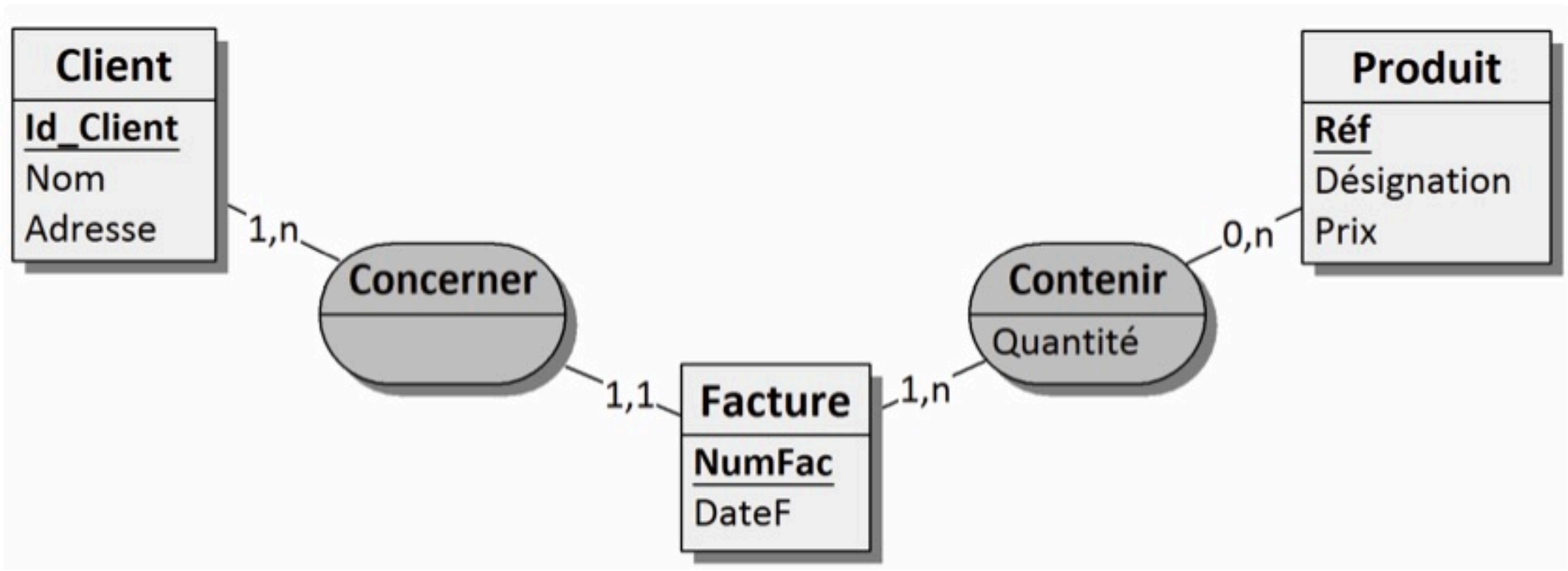


Mais dans ce cas : « *Chaque facture concerne **1** et **1 seul** produit* »…

On voit bien que la cardinalité sur la patte rattachée à la classe d'entités « Facture » doit être différente en fonction des deux autres classes associées :

- **1,1** vers « Client »
- **1,n** vers « Produit »

Il faut donc bien deux associations binaires différentes ; nous retrouvons alors la solution proposée au début de cet ouvrage :

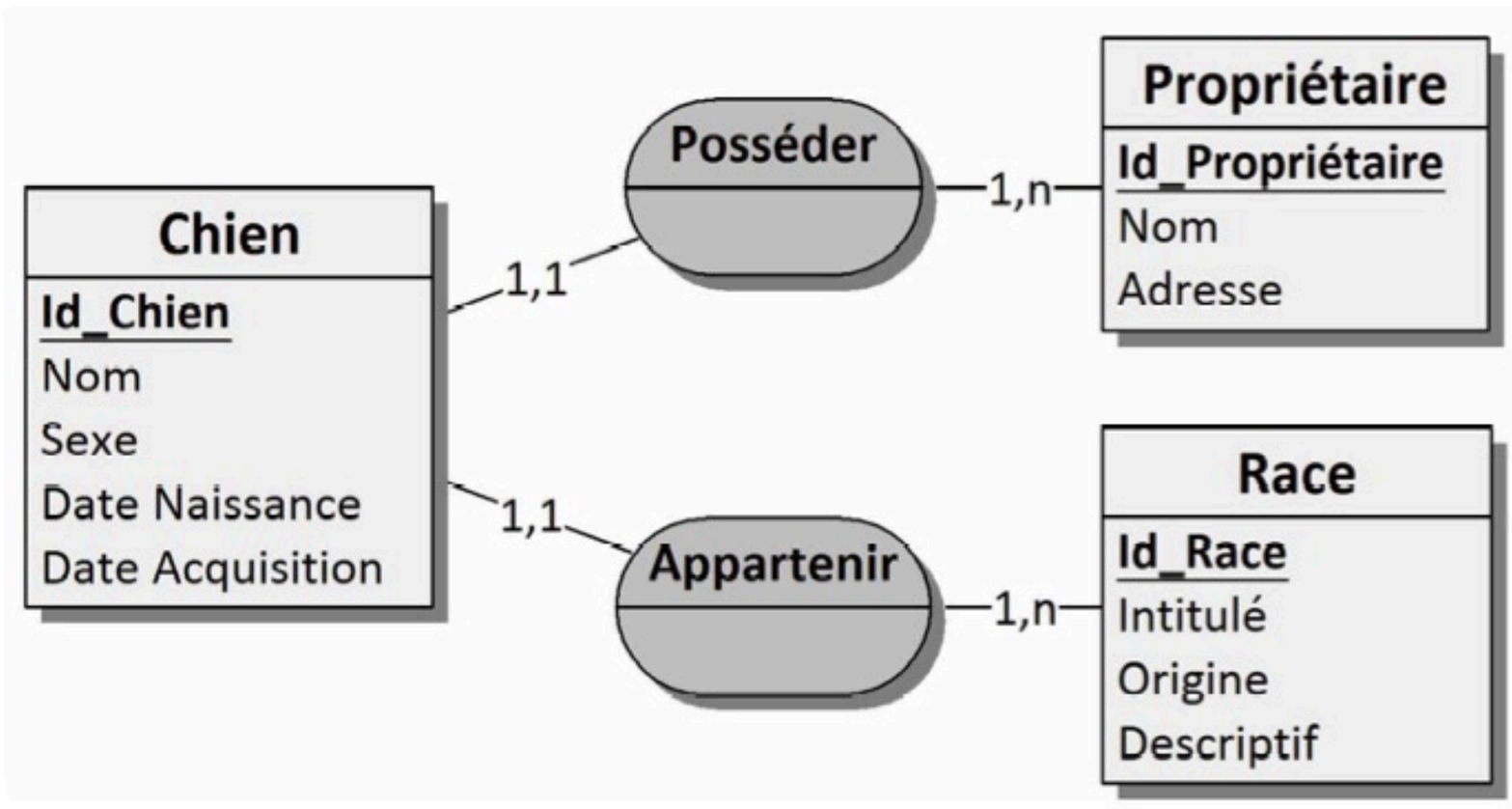


© Patrick Bergougnoux

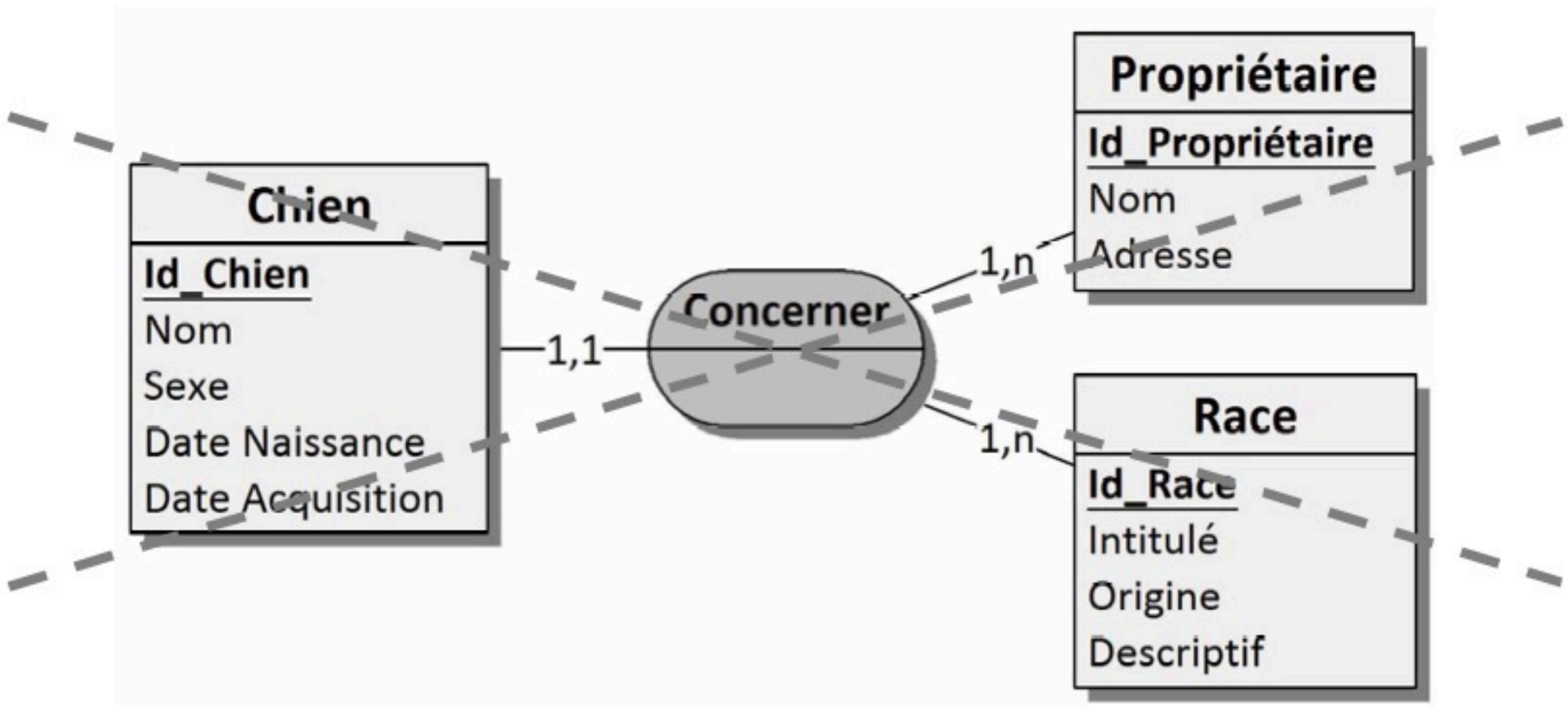
Présence d'une cardinalité simple dans une association n-aire

Au-delà de l'erreur de modélisation que nous venons de voir, le problème du mélange de cardinalités simples et multiples autour d'une même association regroupant plus de deux classes d'entités peut être réglé de manière très simple en créant plusieurs associations.

Pour illustrer cette situation, reprenons une partie du MCDi de la société canine :



Compte-tenu que les cardinalités des pattes rattachées à la classe d'entités « Chien » sont **1,1** pour les deux associations allant vers les classes « Race » et « Propriétaire », la modélisation suivante pourrait être acceptable et produirait le même MLD que le modèle précédent :



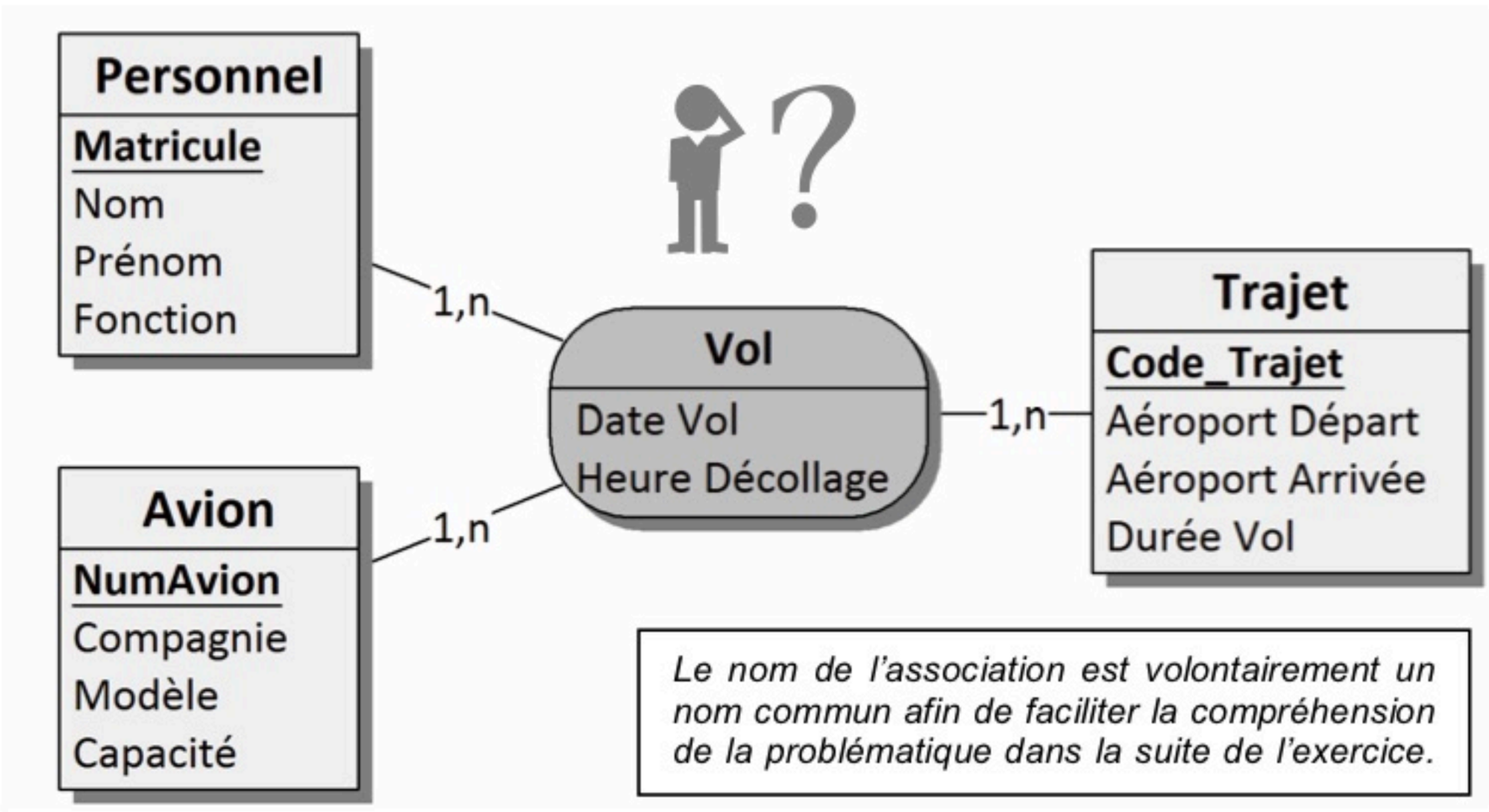
Mais, franchement, à quoi bon prendre le risque d'introduire une ambigüité alors qu'il est bien plus clair sur un plan sémantique de séparer les associations.

Décomposition systématique des associations n-aires

Des problèmes de cardinalités difficiles à appréhender

Dans les exemples que nous avons vus jusqu'à présent, les problèmes liés à l'utilisation d'associations n-aires sont relativement faciles à appréhender. Cela n'est pas toujours le cas, et certains modèles donnent vraiment l'impression d'être justes, parfois même pour un concepteur averti.

Prenons le cas d'une gestion très simplifiée d'une compagnie aérienne souhaitant gérer ses vols ; voici le modèle qui nous est proposé :



Une première analyse sémantique du modèle nous indiquent que :

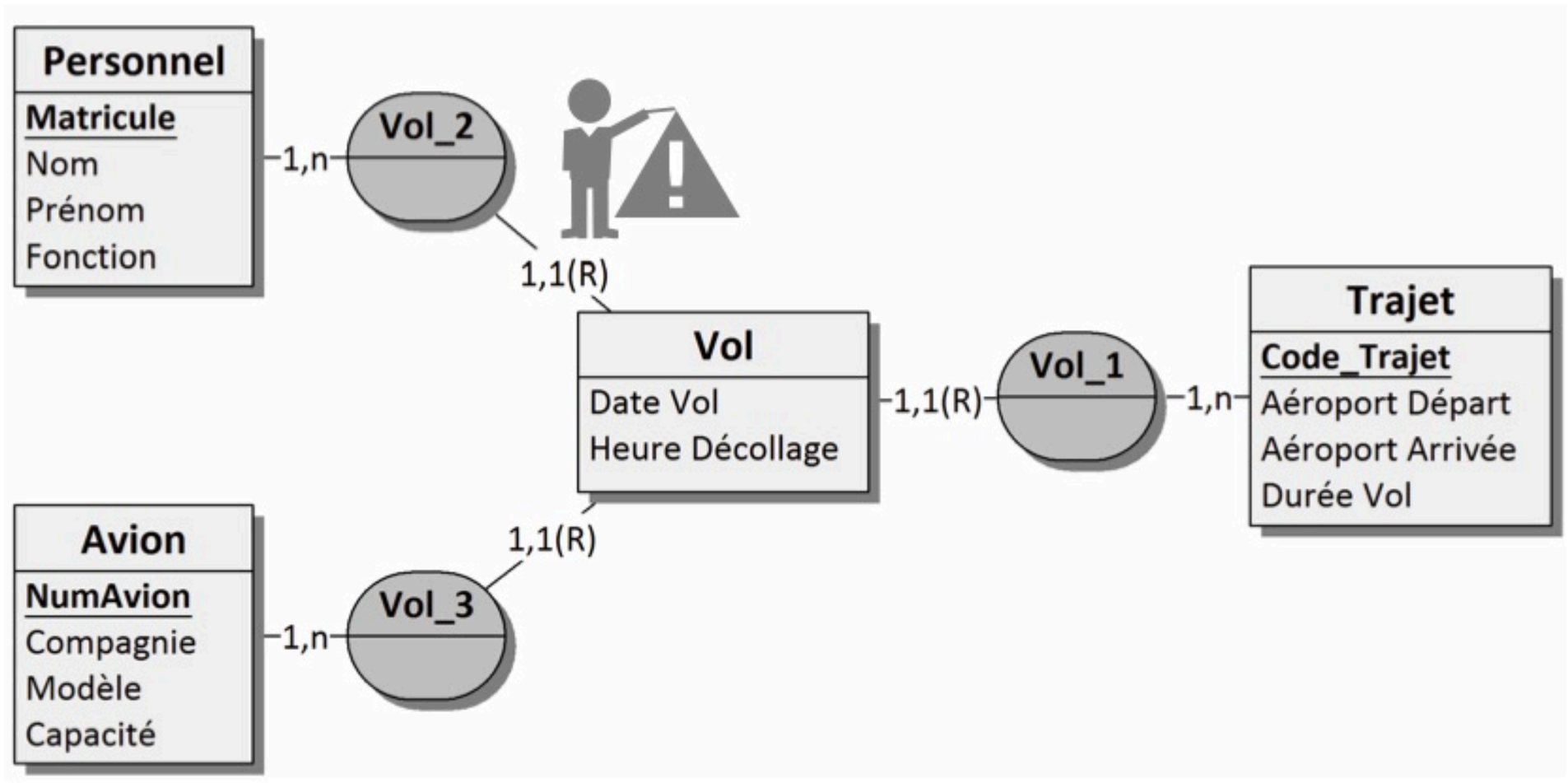
- (En 50 ans) chaque avion fait l'objet de vols concernant **1** à **n** trajets.
- (En 50 ans) chaque avion fait l'objet de vols concernant **1** à **n** personnels.
- (En 50 ans) chaque personnel fait l'objet de vols concernant **1** à **n** trajets.
- (En 50 ans) chaque personnel fait l'objet de vols concernant **1** à **n** avions.
- (En 50 ans) chaque trajet fait l'objet de vols concernant **1** à **n** avions.
- (En 50 ans) chaque trajet fait l'objet de vols concernant **1** à **n** personnels.
- Chaque vol est caractérisé par sa date et son heure de décollage.

Tout cela a l'air parfaitement correct.

Pourtant, je ne vous recommande pas de prendre un vol géré par un système d'information issu de ce modèle !

© Patrick Bergougnoux

Pour comprendre le problème, la meilleure solution reste de décomposer l'association comme nous l'avons fait au chapitre précédent ; et voici ce que nous obtenons :

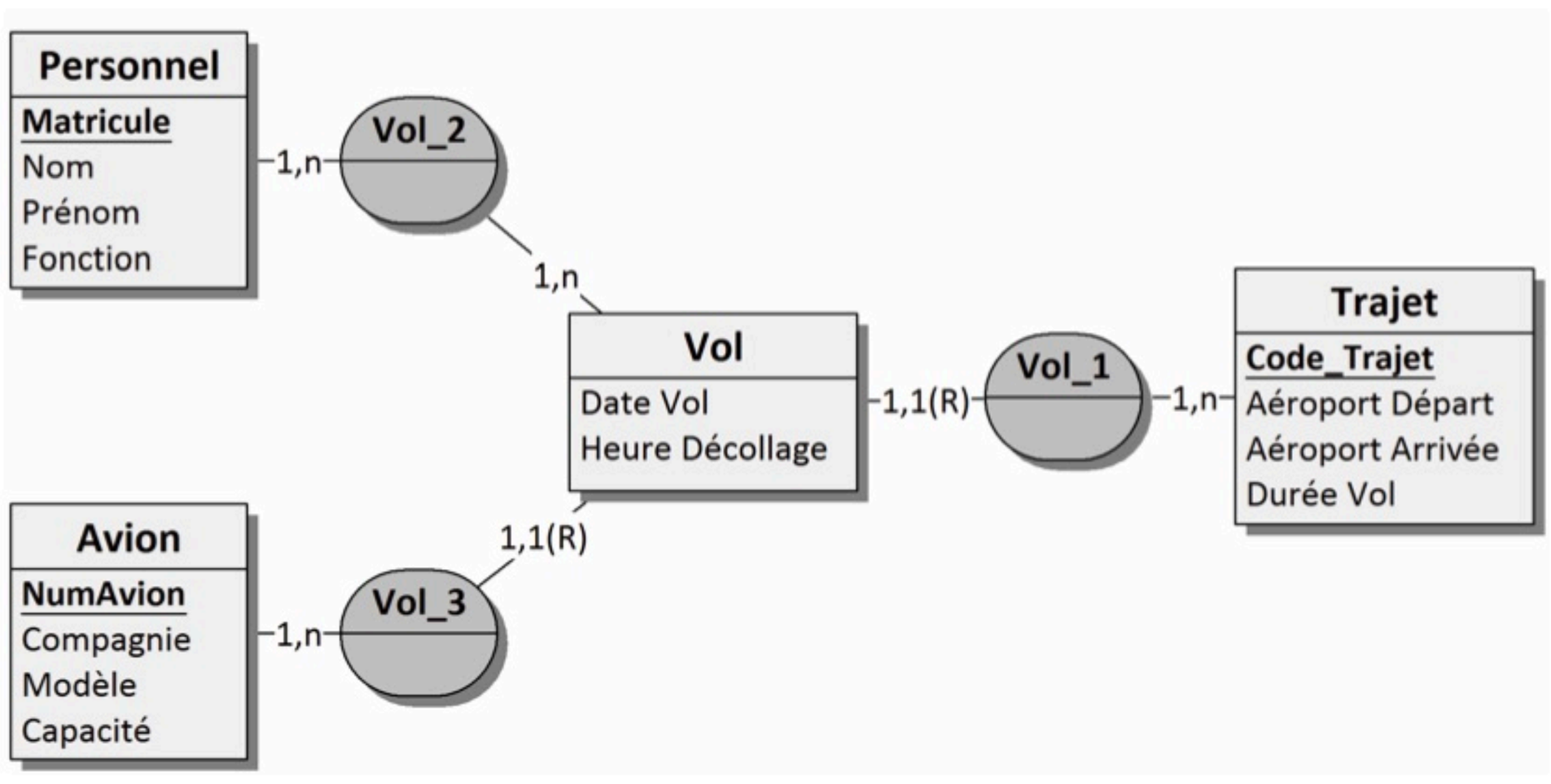


Rappelons que ce modèle décomposé est **strictement** équivalent au modèle précédent. Et que lisons-nous ?

➔ Un vol est assuré par 1 et 1 seul personnel !

À moins que la compagnie ne gère que de tout petits avions, on a du mal à imaginer un tel fonctionnement.

On voit ici clairement l'avantage d'avoir décomposé l'association ; en effet, la correction s'avère ensuite très simple, en modifiant juste une cardinalité de manière à ce qu'un vol puisse concerner plusieurs membres d'équipage :

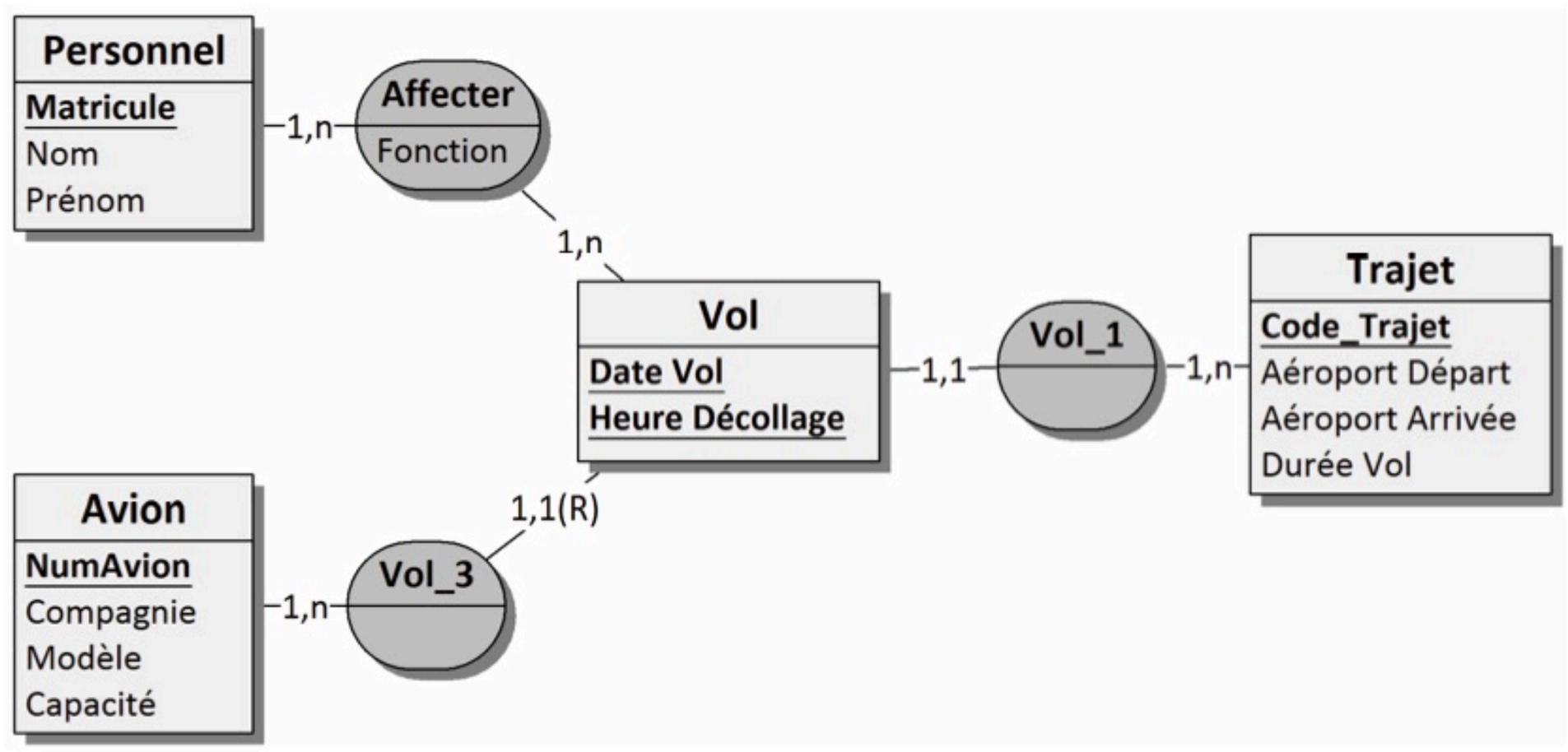


Par contre, « *#Matricule* » n'étant plus identifiant relatif, la classe d'entités « Vol » se retrouve avec uniquement les identifiants relatifs « *#NumAvion* » et « *#Code_Trajet* » pour composer sa clé primaire… Et à eux deux, ils n'assurent pas l'unicité puisqu'un avion peut effectuer plusieurs fois le même trajet : comme dans les exercices précédents, il convient alors de compléter l'identifiant avec la date du vol, voire son heure de décollage, ce qui ne pose aucun problème compte-tenu de la décomposition déjà effectuée.

Mais, après le rajout de la date et de l'heure du vol dans sa composition, la clé n'est plus minimale ! En effet, pour un avion donné et un horaire donné, il ne peut y avoir qu'un seul trajet : l'identifiant relatif « *#Code_Trajet* » n'est alors pas nécessaire et ne doit pas faire partie de la clé primaire de la table « Vol ».

Par ailleurs, l'association « Vol_2 » ayant maintenant des cardinalités multiples de part et d'autre, elle donnera lieu à une table de correspondance lors de la génération du MLD : il convient donc de lui attribuer un nom plus significatif.

Enfin, en intégrant le fait que la fonction du personnel puisse changer selon les vols, voici le bon MCDi pour cette compagnie aérienne :



Et voici le MLD correspondant :

Personnel = (**Matricule**, Nom, Prénom)

Avion = (**NumAvion**, Compagnie, Modèle, Capacité)

Trajet = (**Code_Trajet**, Aéroport_Départ, Aéroport_Arrivée, Durée_Vol)

Vol = (***#NumAvion, Date_Vol, Heure_Décollage***, *#Code_Trajet*)

Affecter = (***#(#NumAvion, Date_Vol, Heure_Décollage),#Matricule***, Fonction)

Compte tenu de la clé composée assez complexe de la table « Vol » et de ses conséquences sur la clé de la table « Affecter », créer un nouvel identifiant tel que le numéro de vol peut s'avérer pertinent.

© Patrick Bergougnoux

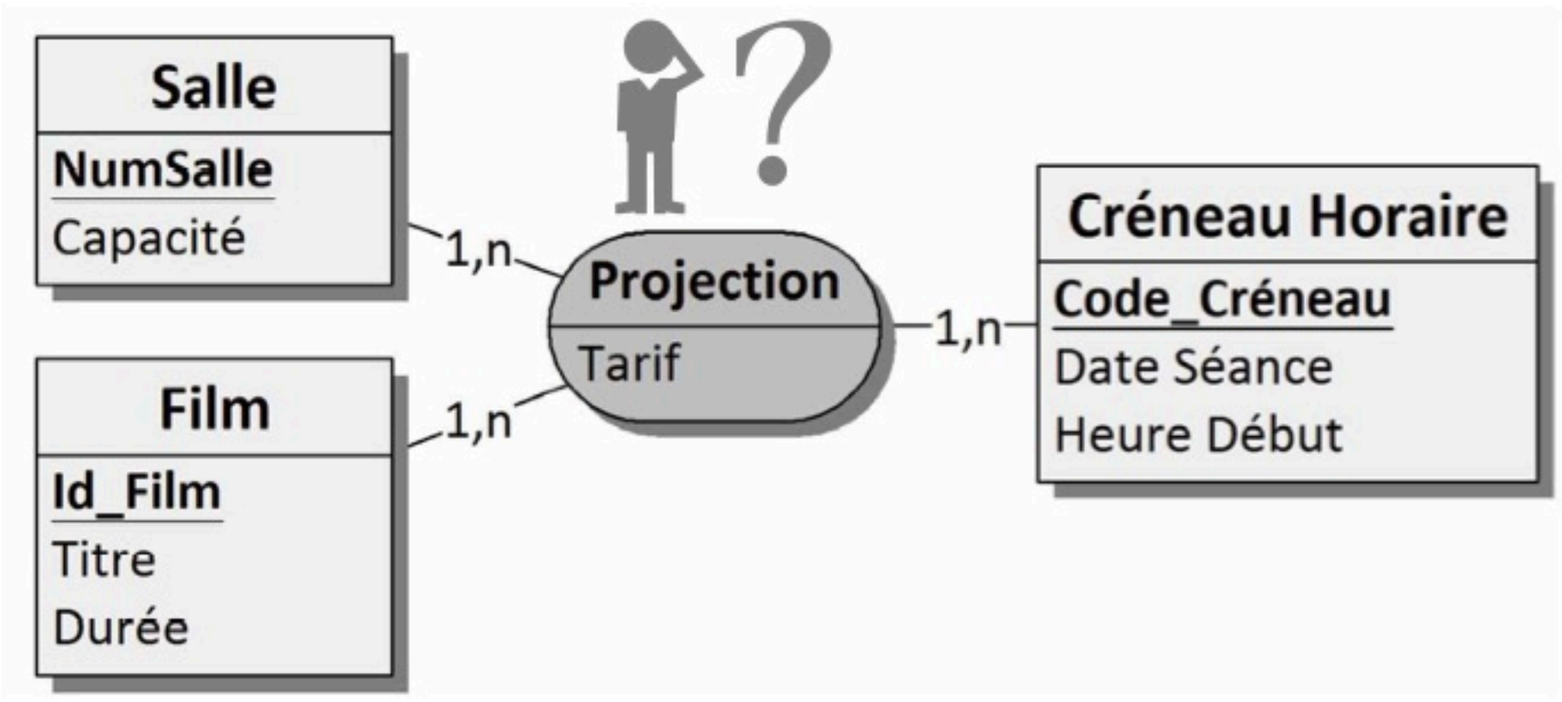
Dorénavant dans cet ouvrage, nous choisirons **des noms communs pour les associations à cardinalités multiples de part et d'autre**. En effet, ces associations étant susceptibles d'être décomposées et des tables de correspondance étant générées au niveau du MLD, des noms communs sont préférables pour les futures instructions SQL.

Détection des problèmes de clé minimale

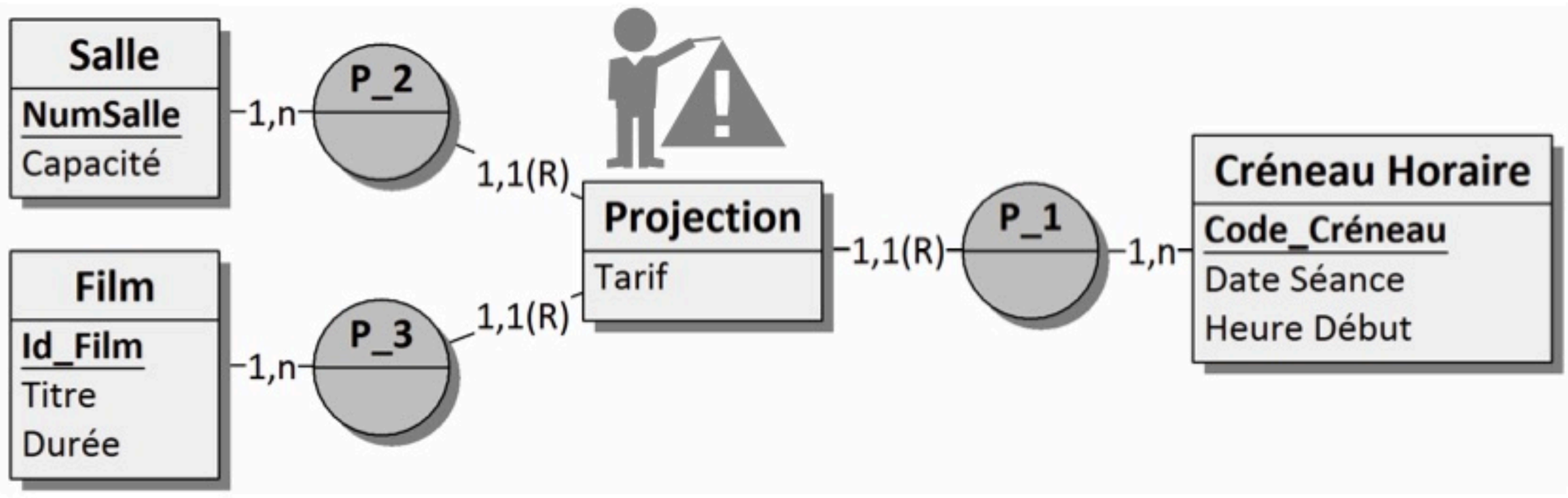
Nous venons de voir que la décomposition permet de corriger des problèmes de cardinalités et, par la même occasion, de mettre en évidence des soucis de clés minimales.

D'une façon générale, même s'il n'y a pas d'erreur de cardinalités, la décomposition va s'avérer très utile pour détecter d'éventuels problèmes concernant l'irréductibilité des identifiants.

Pour illustrer cela, prenons un nouvel exemple avec la modélisation de la projection de films dans un cinéma multisalles. Voici le MCDi que l'on pourrait proposer pour un tel système :



Comme dans l'exercice précédent, ce modèle ne semble pas poser de problème de conception. Effectuons tout de même la décomposition :



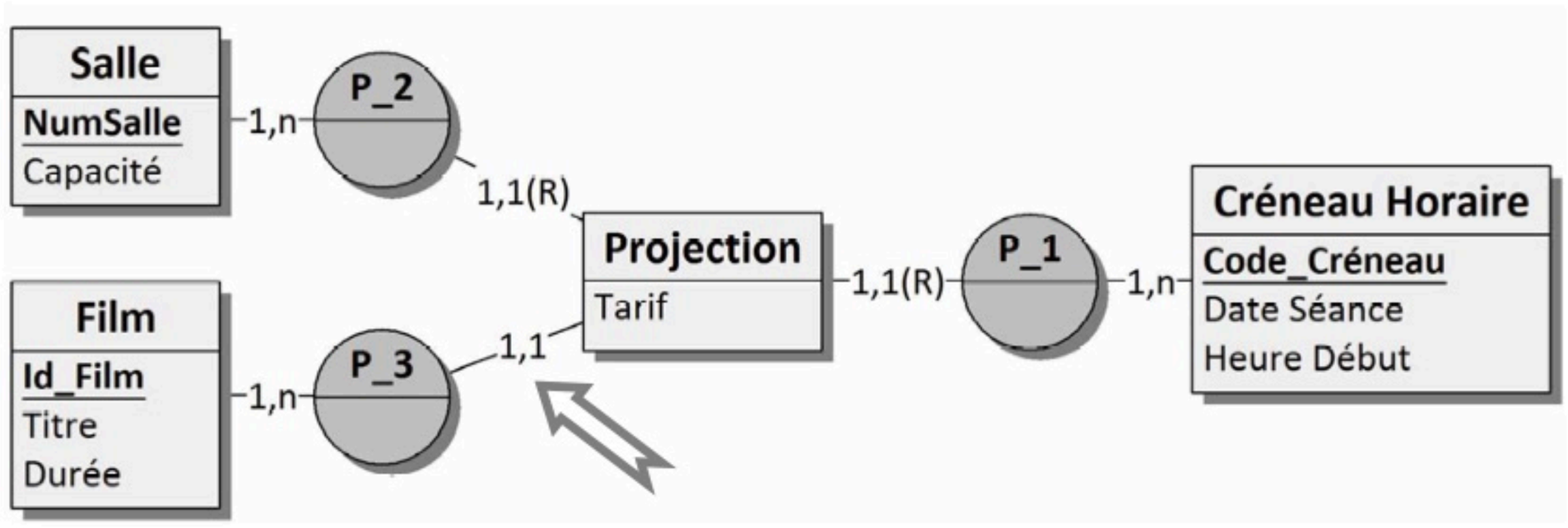
L'identifiant de la classe « Projection » est alors composé des trois identifiants relatifs issus des classes associées.

Nous devons toujours nous poser la même question : la règle d'irréductibilité est-elle respectée ?

Pour s'en assurer, retirons successivement chacun des trois identifiants relatifs et vérifions si l'unicité est toujours respectée :

- ***#Code_Créneau*** : en ne conservant que les identifiants du film et de la salle, un même film ne pourrait pas être projeté dans la même salle à différents créneaux ➔ ***à conserver***.
- ***#NumSalle*** : si on retire le numéro de la salle de l'identifiant, un même film ne pourra pas se jouer en même temps dans deux salles différentes, ce qui peut poser problème pour les films à succès ➔ ***à conserver***.
- ***#Id_Film*** : sans cette rubrique dans l'identifiant, plusieurs films ne pourront pas être projetés dans la même salle en même temps... il vaudrait mieux ! Par conséquent, l'identifiant relatif « Id_Film » doit être retiré de la composition de l'identifiant de la classe « Projection ».

Le MCDi suivant prend donc en compte cette contrainte, ce que le modèle initial avec l'association ternaire ne faisait pas :



Le MLD correspondant montre bien que le film projeté dépend de la salle et du créneau horaire :

Salle = (**NumSalle**, Capacité)

Film = (**Id_Film**, Titre, Durée)

Créneau_Horaire = (**Code_Créneau**, Date_Séance, Heure_Début)

Projection = (***#NumSalle, #Code_Créneau***, Tarif, *#Id_Film*)

« *#Id_Film* » devient alors une clé étrangère classique qui ne participe pas à la clé primaire de la table « Projection », et qui dépend fonctionnellement de « *#NumSalle* » et « *#Code_Créneau* ». Nous verrons à la fin de ce chapitre une autre manière de représenter, dans le MCDi, cette même contrainte.

© Patrick Bergougnoux

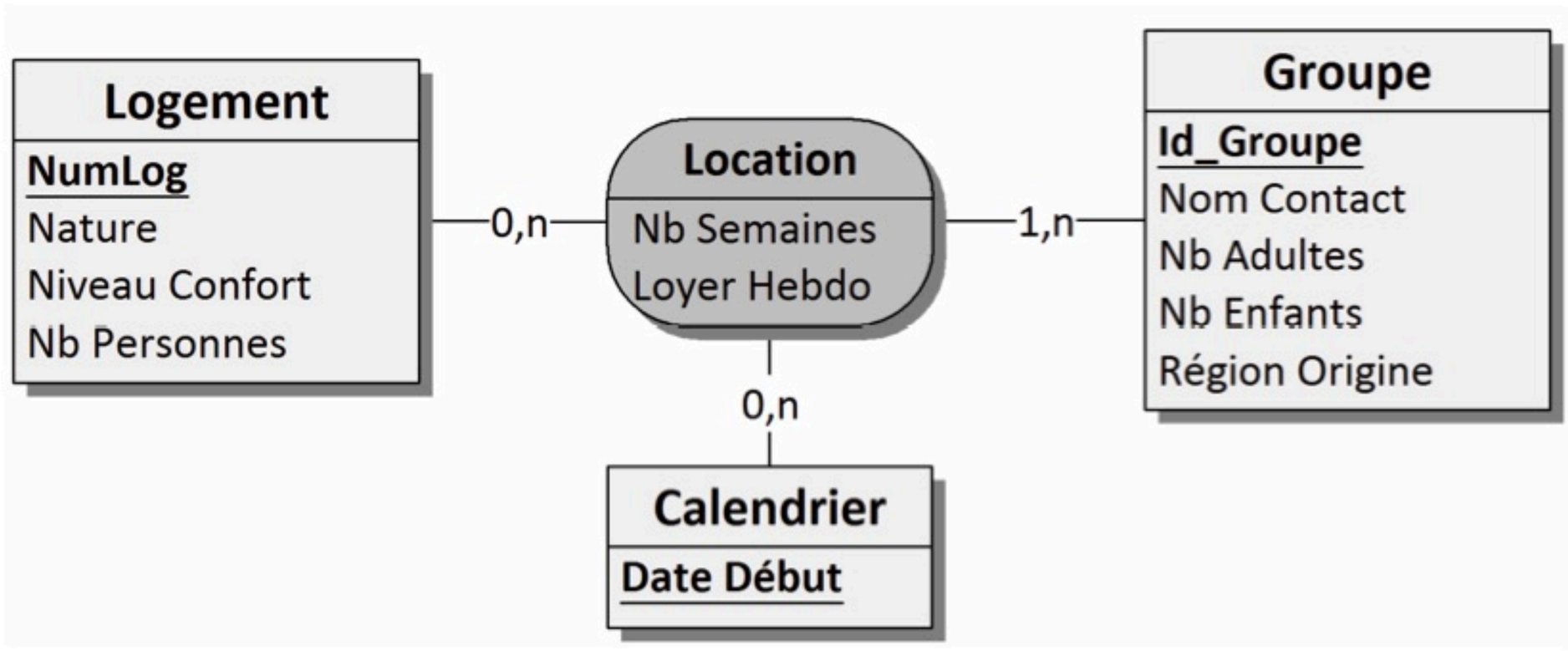
L'historisation et la tentation des tripattes

Utilisation d'une classe d'entités de type Calendrier

Les associations ternaires sont souvent utilisées dès lors qu'un problème d'historisation se pose : une classe d'entités ayant la date pour seule rubrique est alors rajoutée à l'association concernée, cette date faisant office d'identifiant.

Ce type de modélisation fonctionne très bien… mais encore faut-il vérifier que la clé primaire de la table de correspondance issue de l'association soit bien minimale.

Reprenons l'exemple de la location des logements où se posait le problème de la location à plusieurs reprises d'un même logement à un même groupe ; la solution pourrait être :



Le MLD correspondant est le suivant :

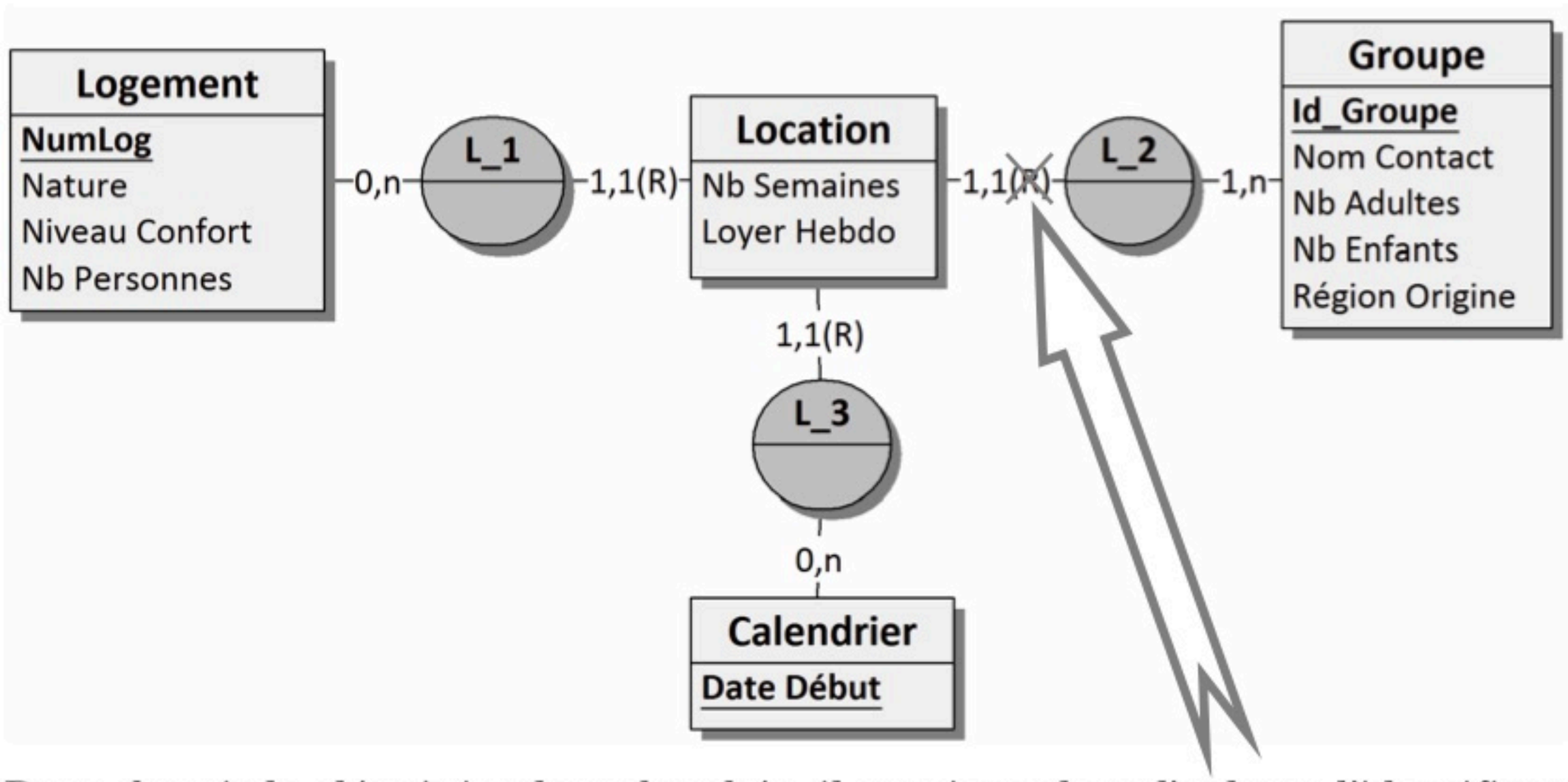
Location = (*#NumLog, #Id_Goupe, #Date_Début*, Nb_Semaines, Loyer_Hebdo)

Lorsque nous avons traité ce même exemple au chapitre précédent dans le cadre des identifiants relatifs, nous avons décomposé l'association pour, là aussi, rajouter « Date_Début » à la clé primaire de la table correspondante, et nous avons obtenu le même MLD.

Mais nous avons également constaté que la clé primaire de cette table n'était pas minimale et qu'il convenait de retirer « *#Id_Groupe* » de cette clé pour obtenir le MLD suivant :

Location = (*#NumLog, #Date_Début*, Nb_Semaines, Loyer_Hebdo, *#Id_Goupe*)

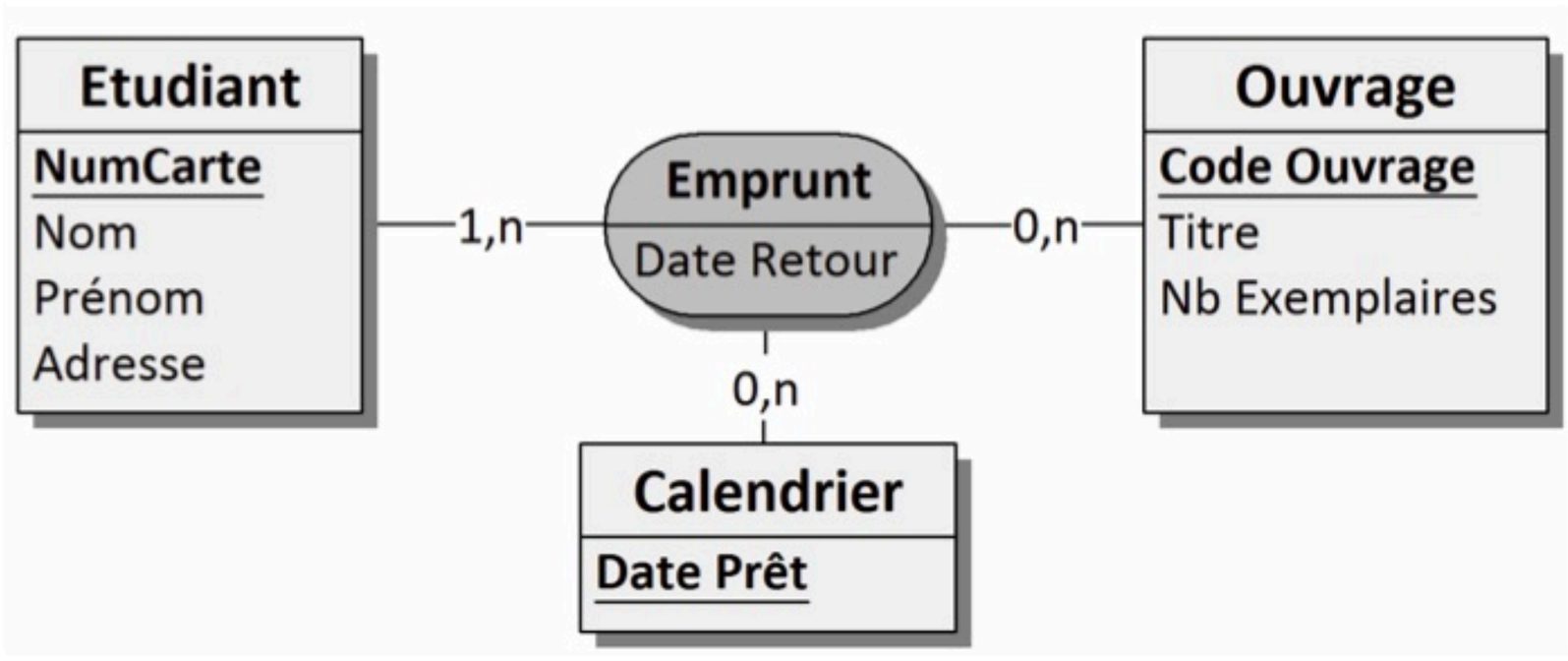
Décomposons cette association tripatte pour visualiser le résultat obtenu et l'adapter afin de respecter la règle d'irréductibilité de l'identifiant :



Pour obtenir la clé minimale recherchée, il convient alors d'enlever l'identifiant relatif sur la patte qui relie la classe « Groupe ».

On se rend ainsi compte que la modélisation proposée avec l'association ternaire n'était pas correcte.

Reprenons également l'exemple de l'historisation des emprunts pour la bibliothèque universitaire ; voici le MCDi correspondant avec une association ternaire faisant appel à une classe d'entités « Calendrier » :



Le MLD correspondant est le suivant :

Emprunt = (*#NumCarte, #Code_Ouvrage, #Date_Prêt*, Date_Retour)

Dans ce cas, la clé obtenue est bien minimale : **le modèle est donc correct**. Néanmoins, la version avec décomposition l'est tout autant et évite l'usage d'associations n-aires.

© Patrick Bergougnoux

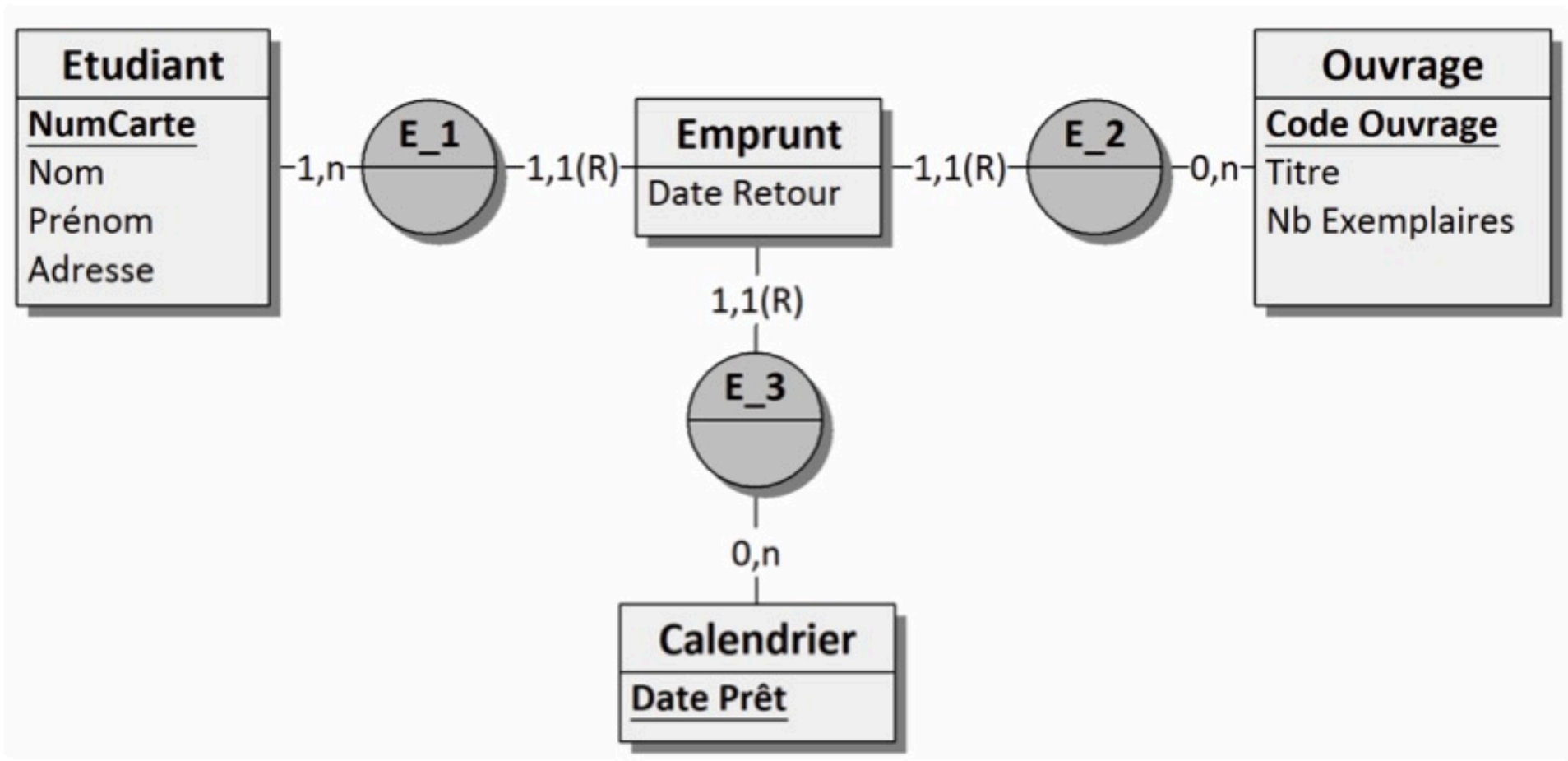
Classe d'entités fictive

Au-delà de son usage dans des associations n-aires, le rajout de la classe d'entités « Calendrier » provoque forcément l'apparition, dans le MLD, de la table correspondante :

- Pour la location des logements : **Calendrier = (Date_Début)**
- Pour la bibliothèque universitaire : **Calendrier = (Date_Prêt)**

À chaque fois, la date indiquée dans la classe « Calendrier » est la seule rubrique et sert donc d'identifiant : on la retrouve alors forcément en tant que clé étrangère dans les tables associées. Par conséquent, dans le MLD et la base de données correspondante, la table « Calendrier » n'apporte aucune nouvelle information.

Pour bien comprendre, décomposons l'association « Emprunt » :

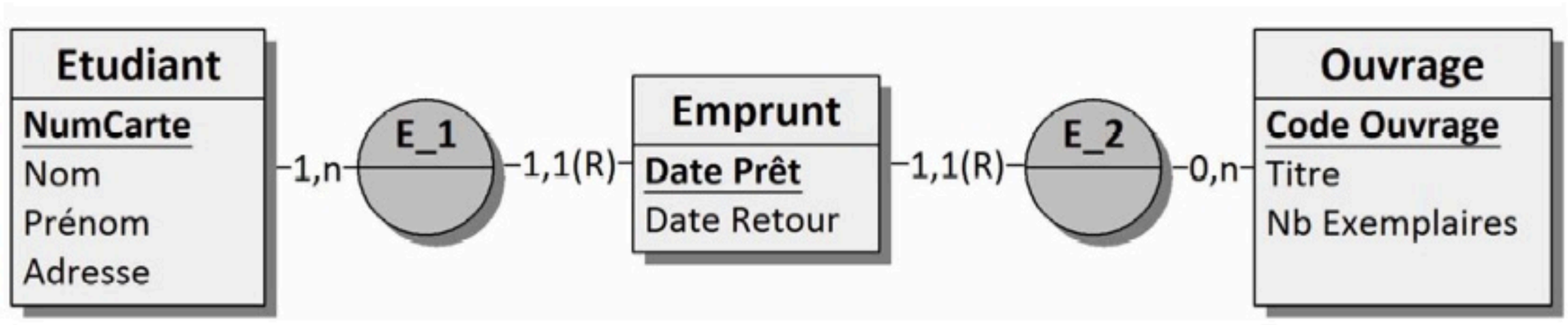


Maintenant, rappelons la règle 3bis vue au chapitre 7 concernant la cohérence des modèles conceptuels de données : « *Si la classe d'entités Y ne possède qu'une seule rubrique et n'est associée qu'à la seule classe d'entités X, en cas de cardinalité simple côté classe X, la rubrique en question peut être intégrée à la classe X, et la classe Y peut alors être supprimée* ».

C'est bien le cas de la classe « Calendrier » qui ne possède qu'une seule rubrique (« Date_Prêt ») et n'est associée qu'à la classe « Emprunt » avec une cardinalité simple côté classe « Emprunt ».

La rubrique « Date_Prêt » peut donc être déplacée dans la classe « Emprunt », et la classe « Calendrier » peut alors être supprimée.

Nous obtenons le MCDi suivant :

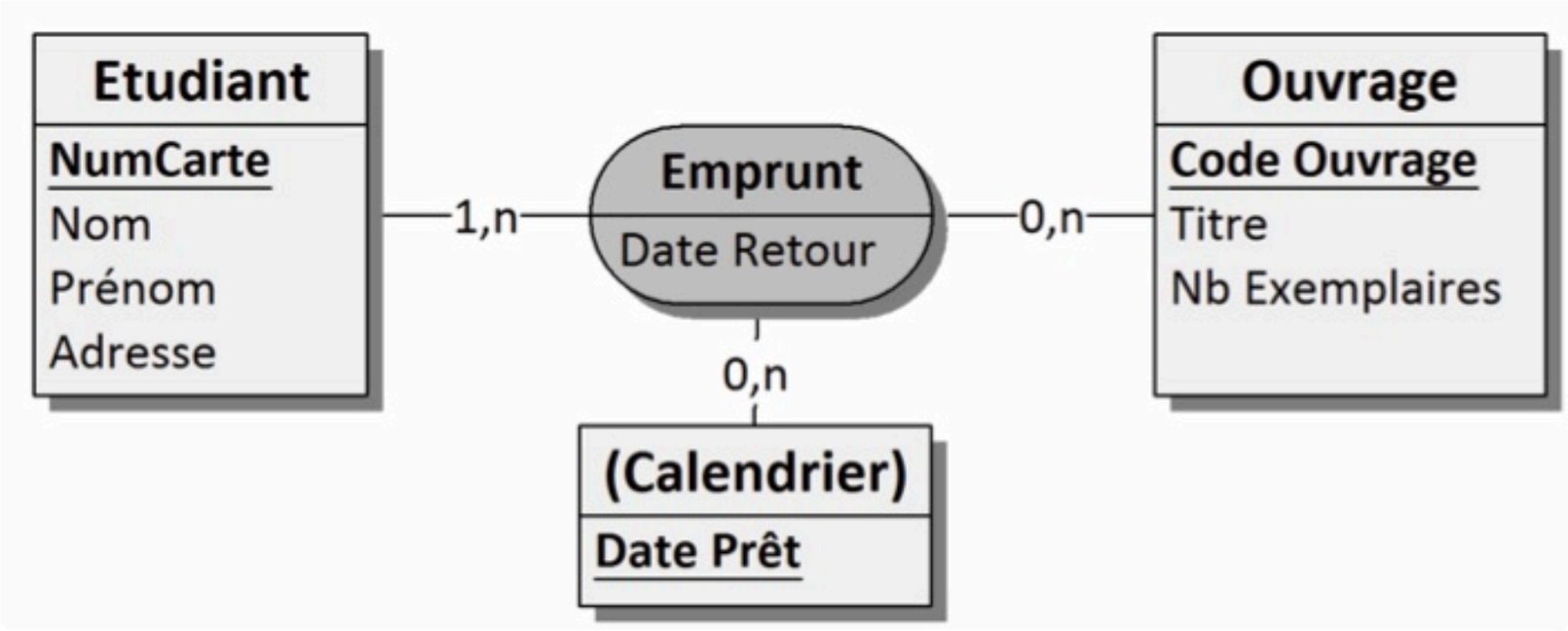


Ça ne vous rappelle rien ? C'est **exactement** le même modèle que celui que nous avons réalisé au chapitre précédent… nous ne connaissions pourtant alors que les associations binaires ! Cela démontre une nouvelle fois que l'on peut très bien se débrouiller sans faire usage des associations n-aires.

Le même résultat peut être obtenu avec l'exercice sur la location des logements.

Si l'on souhaite néanmoins conserver, dans le MCDi, une association dans laquelle une classe de type « Calendrier » est présente sans qu'elle soit reprise sous forme de table dans le MLD, il suffit de la déclarer comme « **classe d'entités fictive** ».

D'une façon générale, si une classe d'entités n'est constituée que des rubriques composant son identifiant, ses informations se retrouveront forcément dans les autres tables du MLD en tant que clés étrangères : la classe d'entités peut alors être déclarée **fictive** et son nom est écrit entre parenthèses dans le MCDi :



Dans le MLD correspondant à cette modélisation, la classe d'entités fictive « Calendrier » n'apparait donc pas, mais la rubrique « Date_Prêt » est bien reprise pour composer l'identifiant de la table « Emprunt ».

Emprunt = (*#NumCarte, #Code_Ouvrage*, Date_Prêt, Date_Retour**)**

Dès qu'une classe d'entités ne possède que des rubriques composant l'identifiant, ***Looping*** propose l'option « Classe d'entités fictive » permettant de ne pas générer de table dans le MLD pour cette classe.

© Patrick Bergougnoux

À utiliser avec précaution

Vous l'aurez compris, je ne suis pas un farouche partisan des associations n-aires ! Je pars du principe que tous les modèles peuvent être réalisés sans y avoir recours, en particulier grâce aux identifiants relatifs.

Cependant, bien utilisées et après avoir été soigneusement vérifiées et décomposées, elles peuvent bien sûr avoir leur place dans vos modèles conceptuels de données… **Mais surtout, vérifiez bien les clés minimales !**

Un bon usage des associations n-aires peut parfois rendre un modèle visuellement un peu moins lourd, mais pas forcément plus lisible surtout lorsqu'il s'agit de s'assurer de l'irréductibilité des identifiants.

Les contraintes d'intégrité fonctionnelle

Une alternative à la décomposition

Si un des identifiants des classes d'entités rattachées à une association ne doit pas participer à la clé primaire de la table de correspondance, le seul moyen que nous avons étudié jusqu'à présent pour représenter le MCDi consiste à décomposer l'association en question, et à retirer l'identifiant relatif adéquat.

Il existe cependant une autre solution pour indiquer la non-participation d'une clé étrangère à une table de correspondance.

Pour cela, il faut signaler qu'une des tables associées dépend fonctionnellement des autres : on parle alors de contrainte d'intégrité fonctionnelle (**CIF**), concept lié à la notion de **dépendance fonctionnelle** étudiée au chapitre précédent.

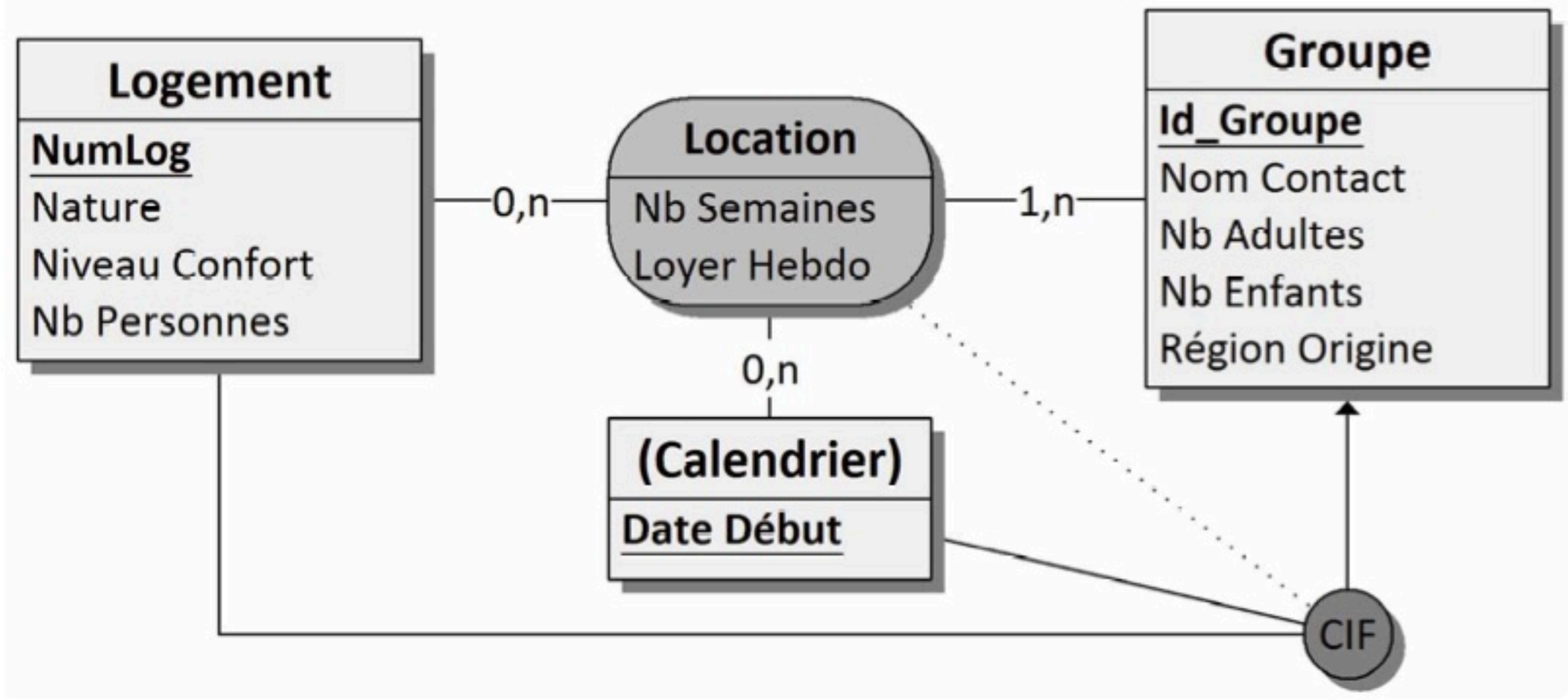
La clé étrangère issue d'une classe d'entités visée par une CIF n'entre pas dans la composition de la clé primaire de la table de correspondance déduite de l'association.

La CIF est représentée par un rond auquel sont rattachées l'association et les classes d'entités concernées, la classe cible étant signalée par une flèche. Précisons que cette solution n'apporte rien de plus comparée à la décomposition de l'association suivie du retrait d'un identifiant relatif. Elle a le mérite d'offrir une représentation graphique qui préserve l'usage des associations n-aires tout en permettant la gestion de l'irréductibilité des identifiants.

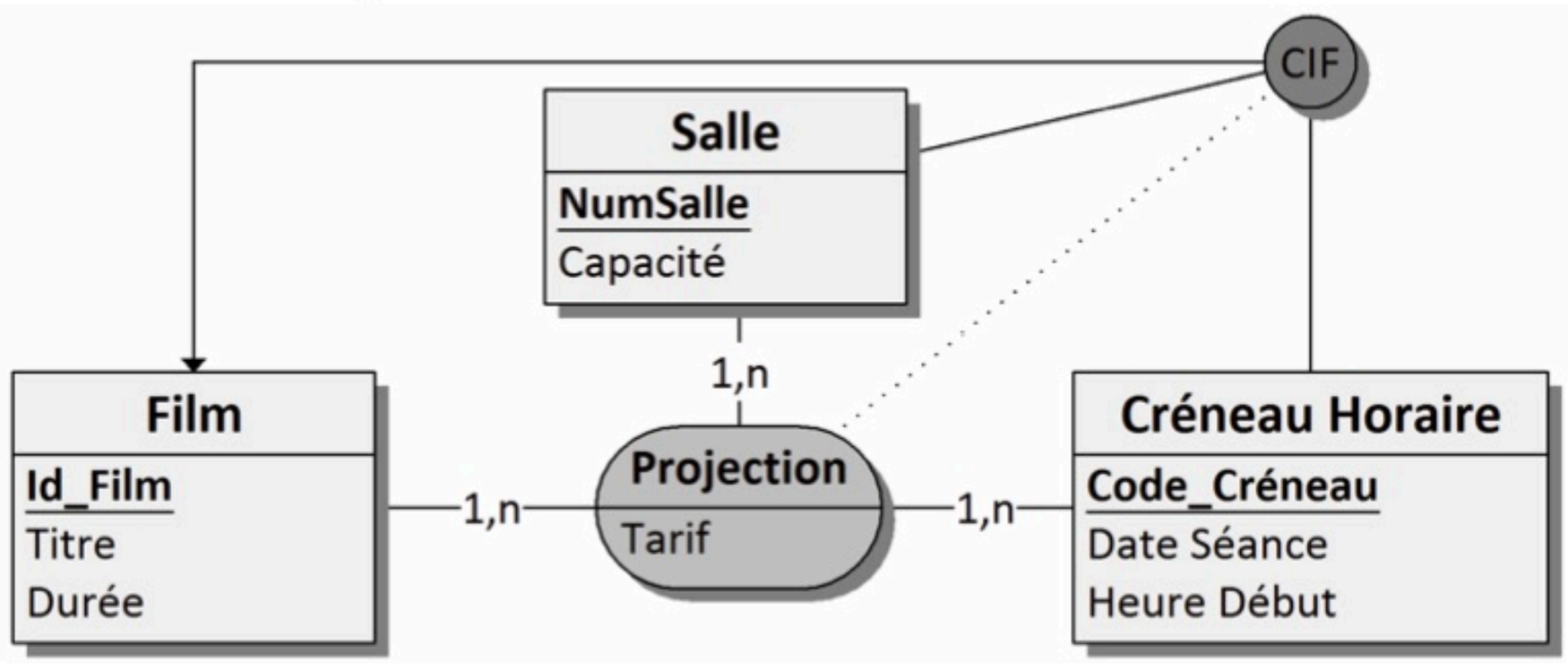
Exemples d'utilisation de CIF

Pour illustrer ce type de modélisation, reprenons les deux exemples qui se prêtent à cette situation.

Tout d'abord, revenons sur la location des logements pour laquelle le groupe dépend du numéro de logement et de la date de début de location :



Reprenons maintenant la gestion des projections de films dans les salles d'un cinéma où le film dépend de la salle et du créneau horaire :



Les MLD correspondants sont exactement les mêmes que ceux obtenus avec la décomposition des associations et l'utilisation sélective des identifiants relatifs.

Ce qu'il faut retenir

- Les associations n-aires peuvent être utilisées dans des cas bien spécifiques, mais devront être systématiquement contrôlées, en particulier concernant le respect des clés minimales.
- Toutes les associations n-aires peuvent faire l'objet d'une décomposition et être ramenées à des associations binaires avec un usage sélectif d'identifiants relatifs.
- Les CIF permettent aussi d'assurer l'irréductibilité des identifiants.

© Patrick Bergougnoux

Chapitre 15
Compléments de modélisation

Ce chapitre apporte quelques éléments de modélisation complémentaires, certes beaucoup moins utilisés que ceux étudiés précédemment, mais qui permettent d'affiner certains modèles.

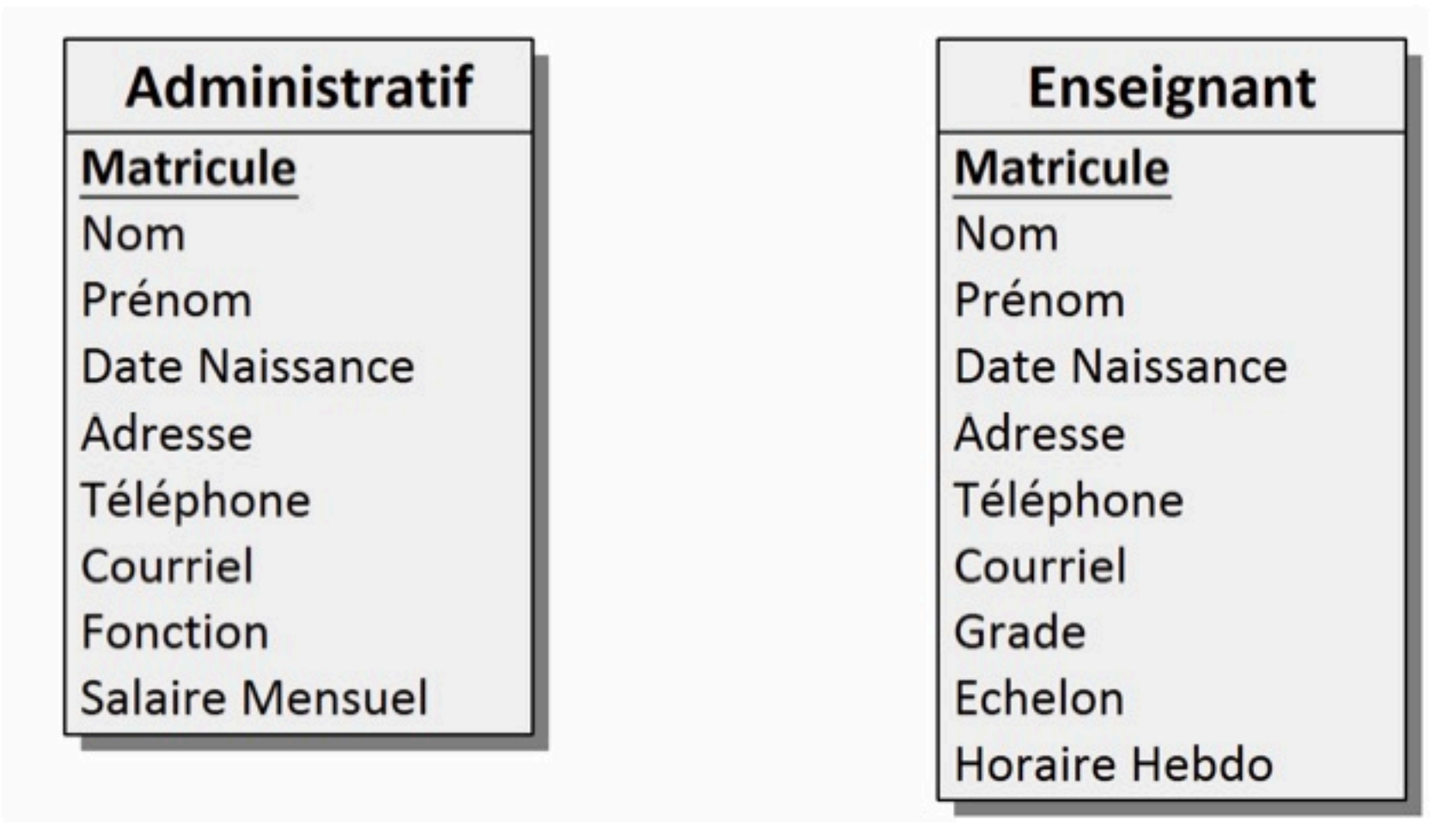
Héritage : spécialisation et généralisation

Rubriques communes à plusieurs classes d'entités

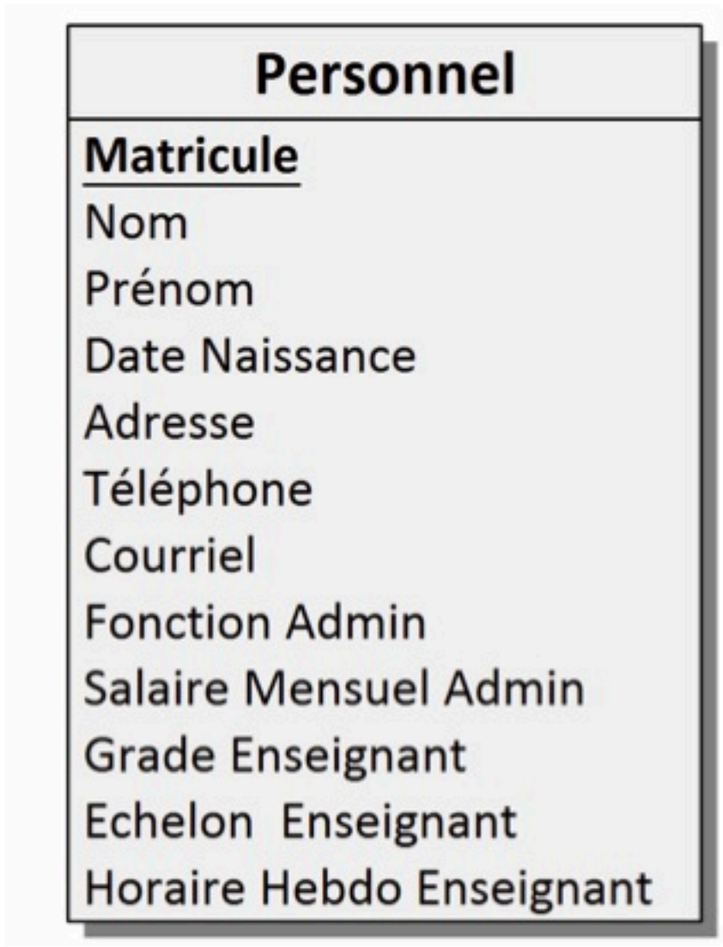
Lors de la définition des classes d'entités, il peut arriver que plusieurs classes aient un certain nombre de rubriques en commun. Se pose alors la question de maintenir des classes d'entités différentes ou de regrouper toutes les rubriques au sein d'une même classe, quitte à ce que certaines rubriques ne soient pas renseignées pour certaines entités.

Prenons l'exemple d'une classe d'entités regroupant le personnel d'un établissement scolaire. Certaines rubriques telles que le matricule, le nom, le prénom, la date de naissance, l'adresse, le téléphone, le courriel sont communes ; d'autres sont spécifiques à certaines catégories de personnel : par exemple, la fonction et le salaire mensuel pour le personnel administratif, et le grade, l'échelon et l'horaire hebdomadaire pour les enseignants.

Une première approche consiste à créer deux classes d'entités différentes pour le personnel administratif et pour les enseignants :



À l'opposé, une deuxième solution consiste à regrouper toutes les rubriques au sein d'une même classe sachant que, dans ce cas là, certaines rubriques ne seront pas renseignées pour certaines entités de la classe en question :



La première solution est satisfaisante si le personnel administratif et les enseignants n'ont pas d'autres liens que ces rubriques communes. Mais, s'il existe aussi des associations avec le reste du système d'information qui sont communes aux deux classes d'entités, la création d'une classe d'entités « Personnel » prend tout son sens.

De la même manière, si le personnel administratif et les enseignants ont, en plus de leurs rubriques spécifiques, leurs propres associations avec le reste du système d'information, la création d'une classe d'entités pour chacun d'entre eux prend aussi tout son sens.

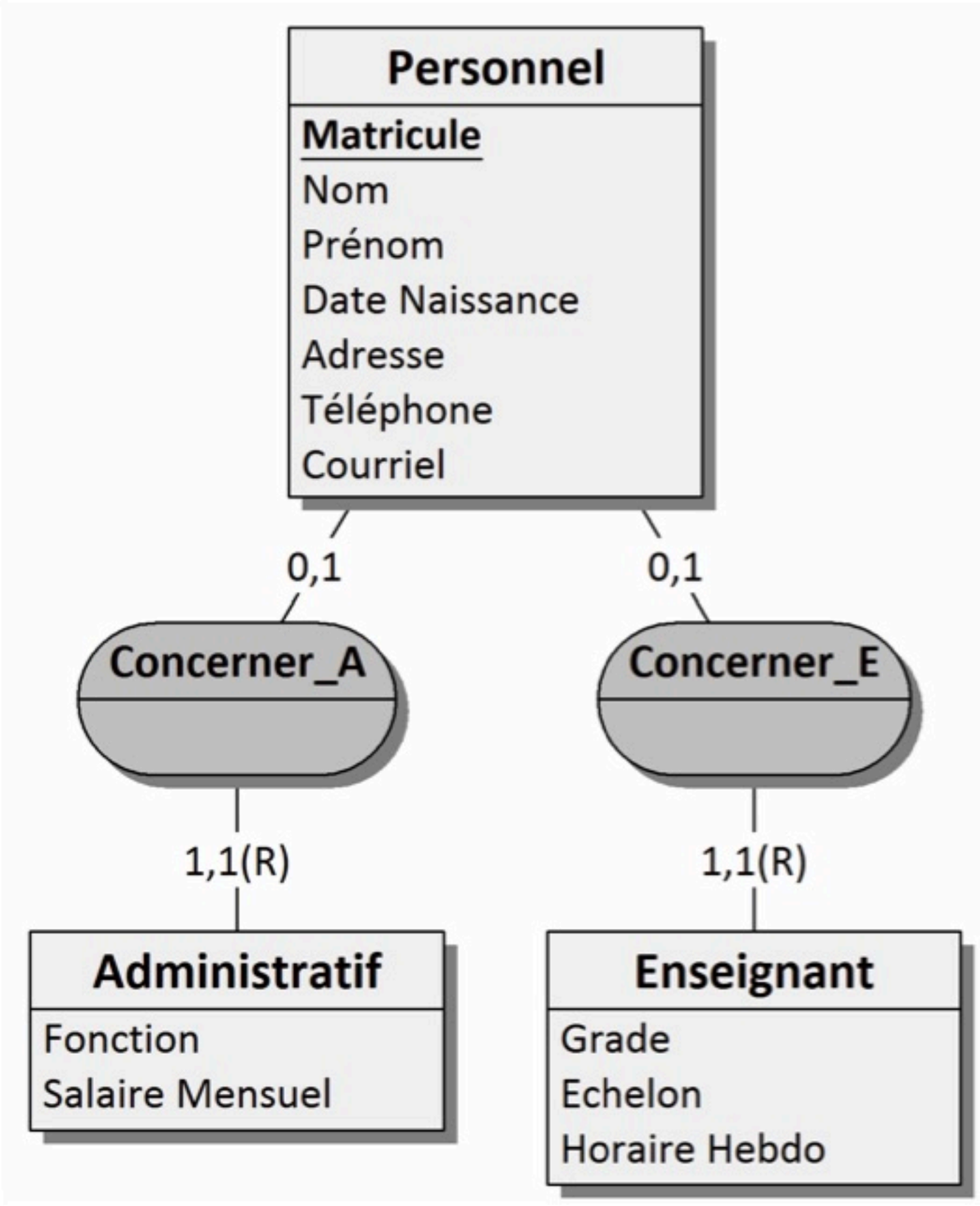
© Patrick Bergougnoux

Sur-type et sous-type

Il arrive donc parfois que la classe d'entités contenant les rubriques communes et celles contenant les rubriques spécifiques aient toutes un réel intérêt pour le système d'information.

La solution consiste à définir une classe d'entités avec la partie commune, appelée « **sur-type** », et des classes d'entités avec les parties spécifiques appelées « **sous-types** ».

Cette situation peut être modélisée de la façon suivante avec les éléments de formalisme que nous avons déjà étudiés :



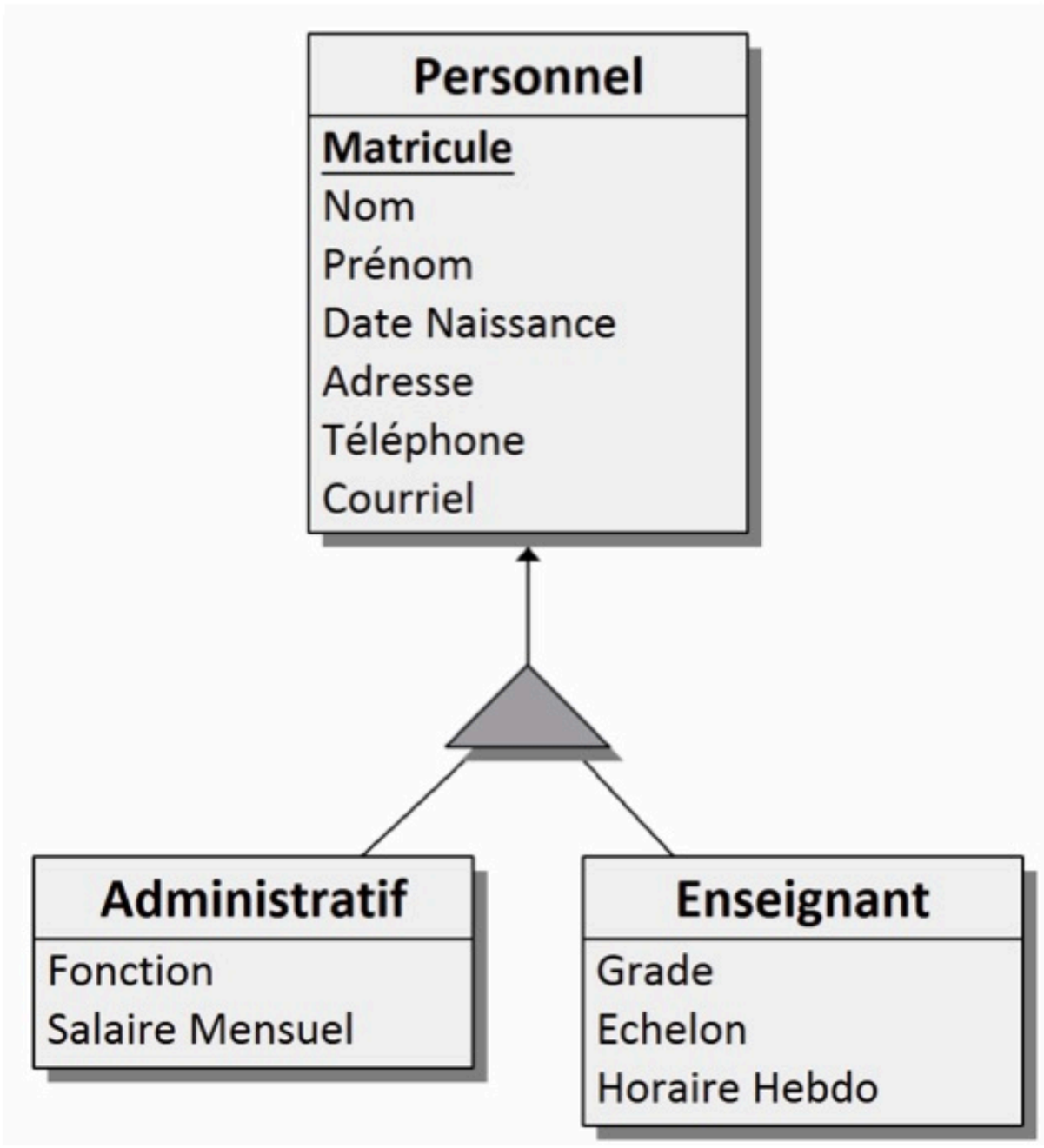
Le MLD correspondant est le suivant :

Personnel = (**Matricule**, Nom, Prénom, Date_Naissance, Adresse, Téléphone, Courrier)

Administratif = (***#Matricule***, Fonction, Salaire_Mensuel)

Enseignant = (***#Matricule***, Grade, Echelon, Horaire_Hebdo)

Le formalisme entité-association propose une notation spécifique plus concise avec la notion d'**héritage** :



Il n'existe qu'un seul sur-type, désigné par la flèche, et le nombre de sous-types est entièrement libre.

Le MLD correspondant à cette notation est bien sûr exactement le même qu'avec la modélisation précédente.

Spécialisation ou généralisation

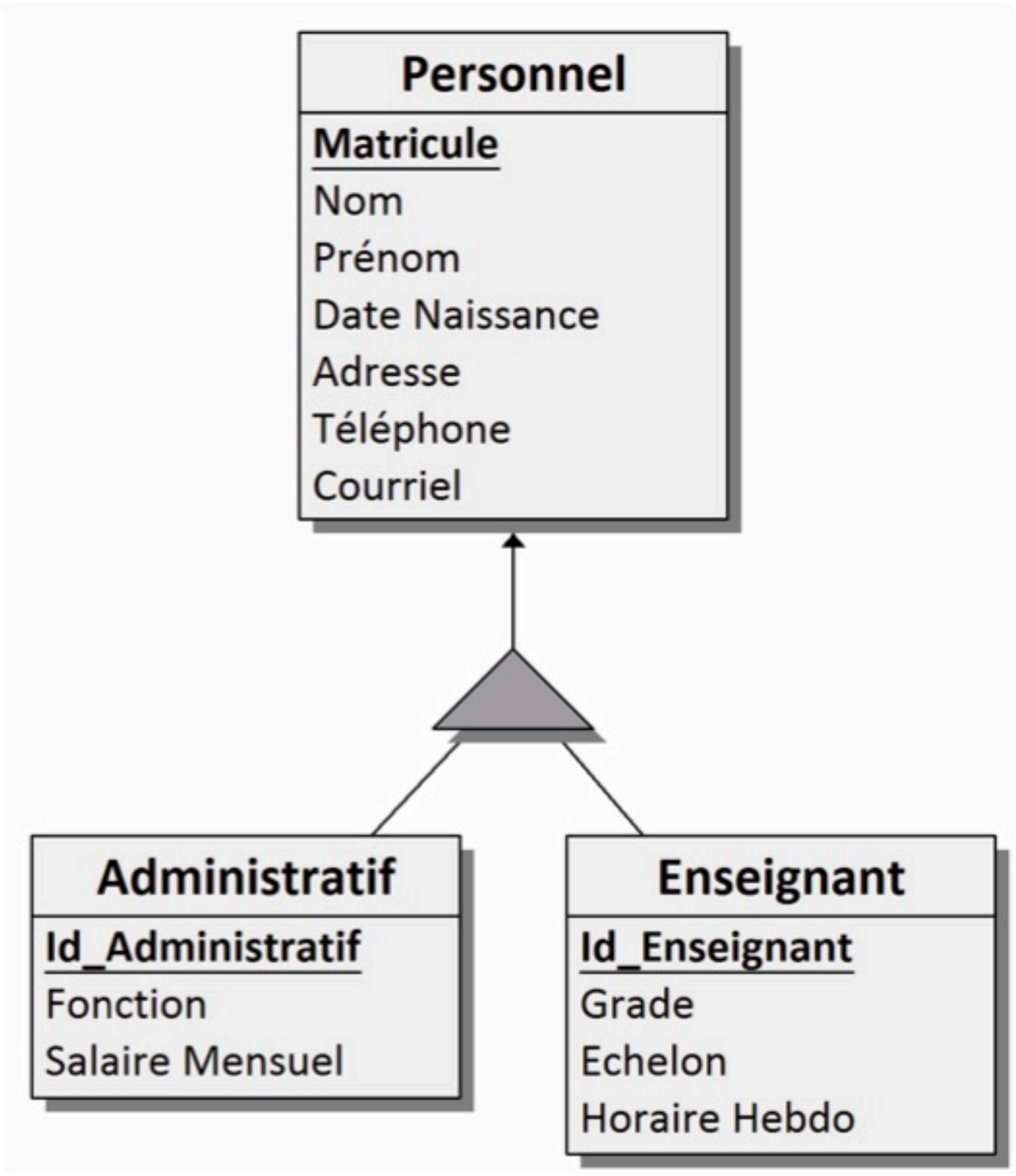
Dans l'exemple que nous venons de traiter, les sous-types dépendent directement du sur-type, leur identifiant étant celui du sur-type comme le montrent bien le MLD et le MCDi précédents avec les identifiants relatifs. Ce type d'héritage est appelé « **Spécialisation** ».

Les sous-types peuvent également avoir leur propre indépendance, les rubriques du sur-type venant seulement les compléter. Cette situation se caractérise dans le modèle par la présence d'un identifiant dans les sous-types.

La clé primaire du sur-type n'est alors plus la clé primaire des sous-types et devient une simple clé étrangère au sein de ces sous-types. On parle dans ce cas de « **Généralisation** ».

© Patrick Bergougnoux

Pour notre exercice, nous aurons affaire à une généralisation si les sous-types « Administratif » et « Enseignant » ont leur propre identifiant ; par exemple :



Le MLD devient alors :

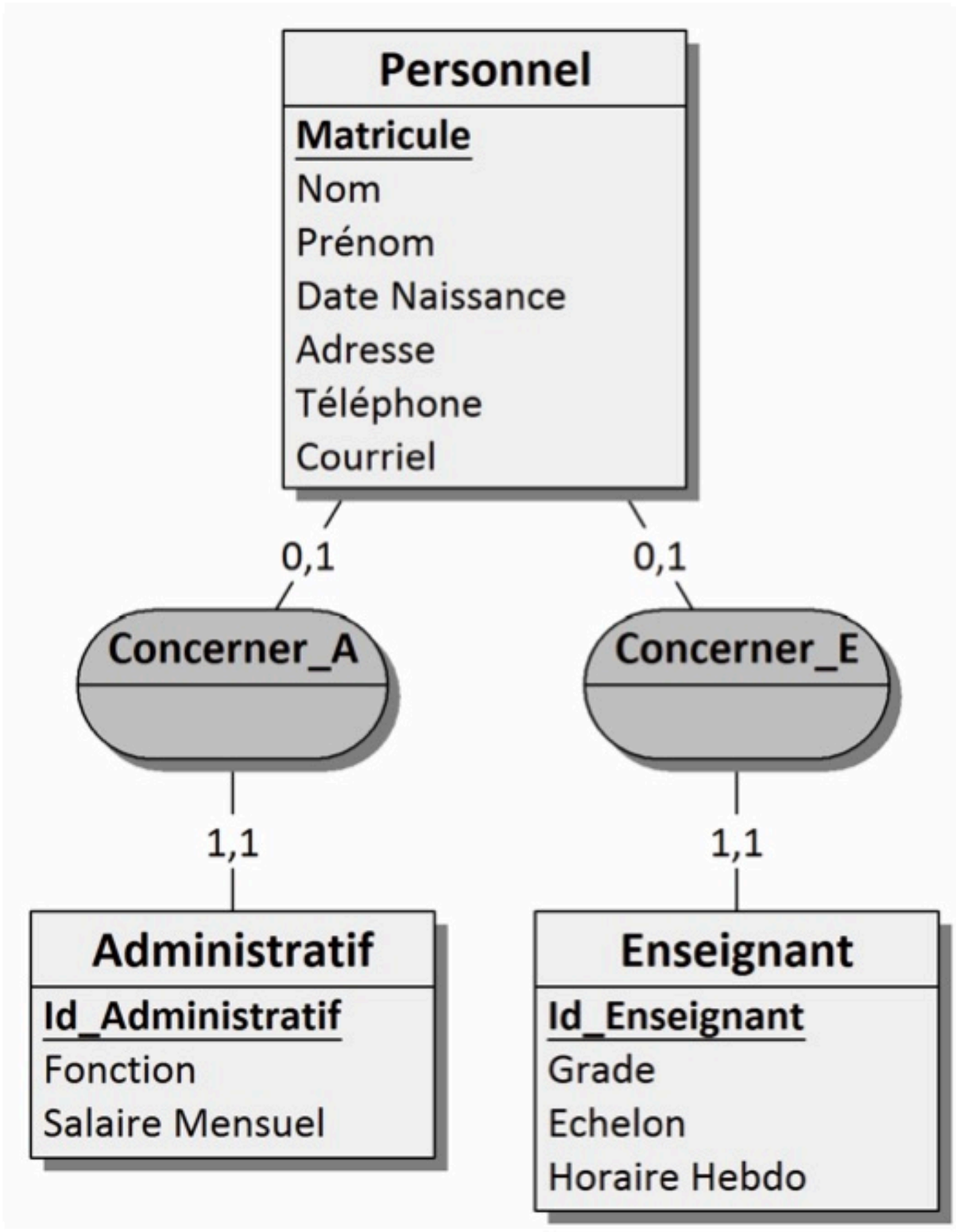
Personnel = (**Matricule**, Nom, Prénom, Date_Naissance, Adresse, Téléphone, Courrier)

Administratif = (**Id_Administratif**, Fonction, Salaire_Mensuel, *#Matricule*)

Enseignant = (**Id_Enseignant**, Grade, Echelon, Horaire_Hebdo, *#Matricule*)

Même en cas de généralisation, le sur-type doit conserver sa clé primaire pour servir de clé étrangère aux sous-types, et ainsi assurer les liens entre les différentes classes.

Cette modélisation de la généralisation a également son équivalent avec le formalisme de base sans utiliser la notion d'héritage :



Le MLD correspondant à ce modèle est le même que celui de la version utilisant la généralisation.

Attention aux abus

Lorsque l'on découvre une nouvelle possibilité de modélisation, le risque est de vouloir l'utiliser à outrance, et pas forcément à bon escient ! L'usage de la notion d'héritage doit être réservée à des cas bien précis, et il ne faut en aucun cas en abuser : en effet, le MLD et la base de données correspondante pourraient alors s'alourdir inutilement.

Plusieurs situations particulières doivent être étudiées avant de choisir d'utiliser ou non l'héritage dans les modèles conceptuels de données :

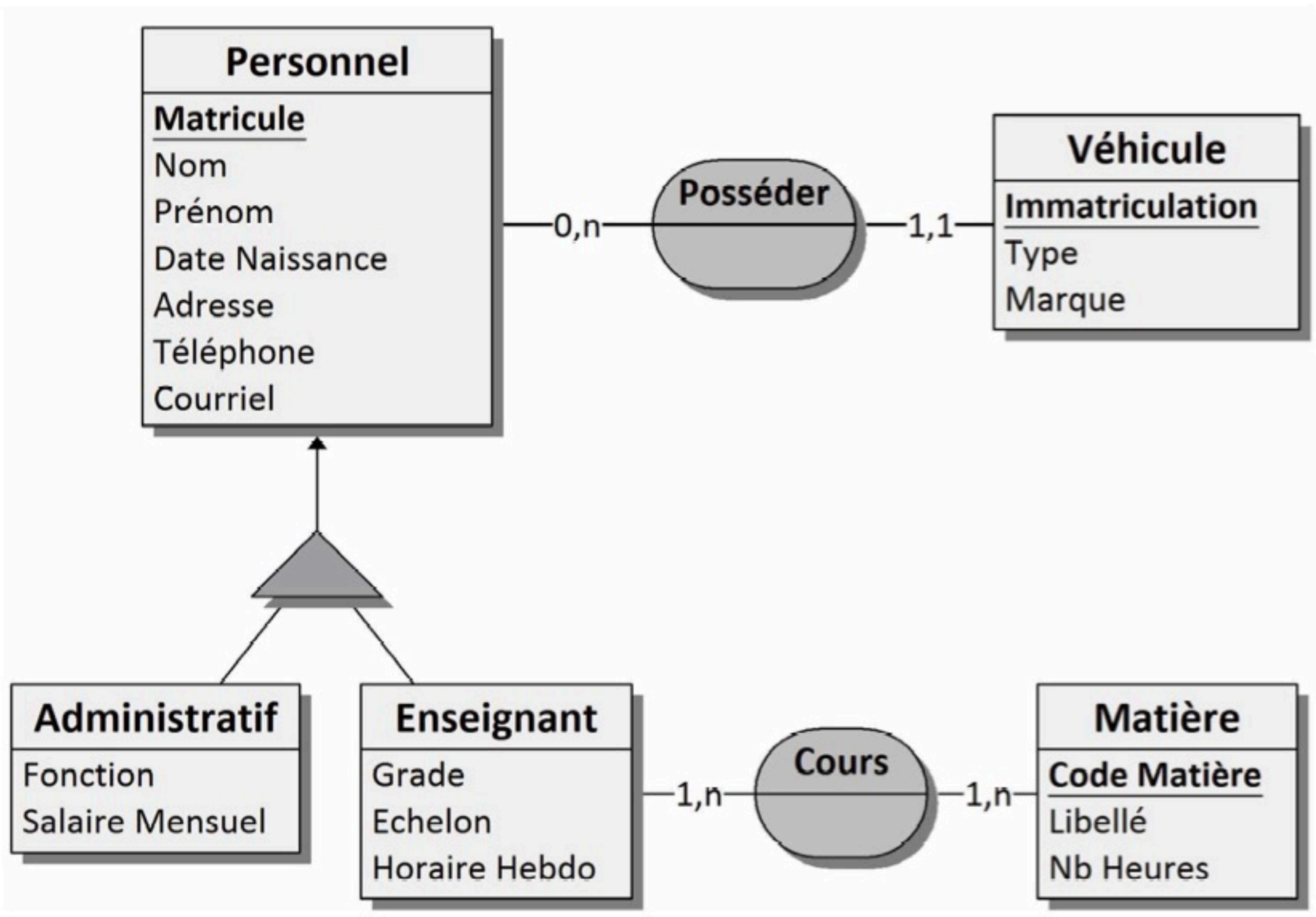
- Si le sur-type ne possède aucune association avec le reste du modèle, il faut bien s'assurer que la classe d'entités correspondante a un réel intérêt pour le système d'information ; dans le cas contraire, il faut envisager de

© Patrick Bergougnoux

reporter les rubriques du sur-type dans chaque sous-type. En effet, dans bon nombre de systèmes d'information, on trouve des classes d'entités avec des rubriques communes telles que nom, adresse, téléphone, … Ce n'est pas pour autant qu'il faut systématiquement créer un sur-type avec ces rubriques.

- Si aucun sous-type ne possède une association avec le reste du modèle, l'héritage ne concerne que des rubriques : dans un souci de simplification du MLD et de la base de données correspondante, il est alors possible de choisir de regrouper toutes les rubriques dans le sur-type, quitte à ce que certaines rubriques ne soient pas renseignées. Néanmoins, les valeurs nulles étant l'ennemi des puristes en bases de données, l'héritage peut rester la bonne solution si le nombre de rubriques appartenant aux sous-types est suffisamment important.

En complétant notre exemple avec les classes d'entités « Véhicule » et « Matière » dans le modèle ci-dessous, l'utilisation de l'héritage est totalement justifiée, chaque classe d'entités, sur-type et sous-types, ayant sa spécificité au sein du système d'information :



Rappel : l'association entre « Enseignant » et « Matière » ayant des cardinalités multiples de part et d'autre, une table de correspondance sera générée dans le MLD et il est donc préférable de donner un nom commun à ce type d'association.

Contraintes inter-associations

Les cardinalités expriment les contraintes de participation minimale et maximale d'une entité d'une classe avec les entités des classes associées. L'héritage exprime également, avec une notation spécifique, ce même type de contraintes. Les identifiants relatifs et les CIF expriment, pour leur part, des contraintes sur la composition des clés primaires.

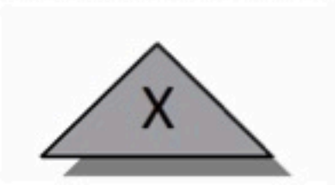
Il est également possible d'exprimer des contraintes supplémentaires précisant les relations existantes entre les différentes associations qui lient certaines classes d'entités : on parle alors de **contraintes inter-associations**.

Mise en œuvre dans le cadre de l'héritage

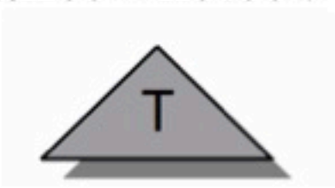
Le plus simple pour bien comprendre ce nouveau type de contraintes est de l'appliquer à ce que nous venons d'étudier dans le cadre de l'héritage.

Plusieurs contraintes peuvent être exprimées concernant le lien des entités d'un sur-type avec les entités de ses sous-types :

- Une entité du sur-type concernée par une entité d'un sous-type ne peut pas être concernée par une entité d'un autre sous-type : on dit qu'il y a « **exclusivité** » ou « **exclusion mutuelle** » et on note :



- Toutes les entités du sur-type sont concernées par au moins une entité d'un sous-type : il y a « **totalité** » notée :



- Chaque entité du sur-type est concernée par une et une seule entité d'un des sous-types (il y a donc exclusion mutuelle et totalité) : il s'agit d'une « **partition** » cumulant les deux contraintes précédentes et notée :

Si aucune de ces contraintes n'existe, le triangle reste vide.

Dans notre exemple, si tous les personnels sont, soit des administratifs, soit des enseignants, il s'agira d'une partition (« **XT** »). Si un personnel ne peut pas être à la fois administratif et enseignant, et qu'il existe aussi du personnel qui est ni un ni l'autre, il y aura exclusion mutuelle mais pas totalité (« **X** »).

© Patrick Bergougnoux

Représentation dans le formalisme de base

Ces mêmes contraintes peuvent s'appliquer aux associations classiques du formalisme de base. Leur représentation graphique est alors similaire à celle des CIF, avec un rond dans lequel la contrainte est notée, un trait plein reliant ce rond aux associations concernées et un trait pointillé reliant ce même rond à la classe ou aux classes d'entités concernées, appelées « **entités pivots** ».

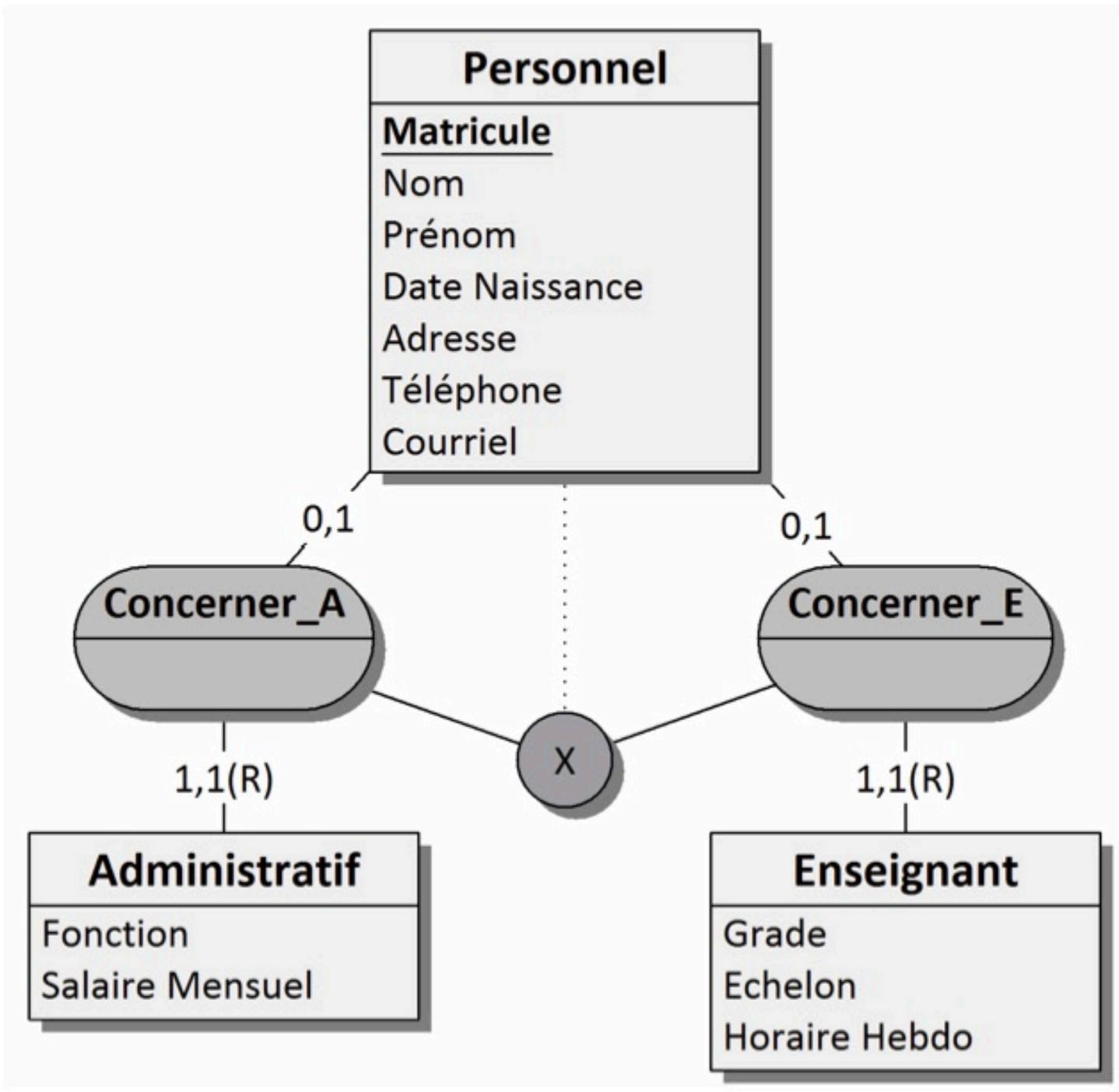
Deux nouvelles contraintes peuvent être exprimées dans les modèles :

- La « **simultanéité** », notée « **S** », indiquant que la participation aux associations impliquées est simultanée.
- La contrainte « **inclusion** », notée « **I** », qui a la particularité d'être orientée : si une contrainte d'inclusion va d'une association « Asso1 » vers une association « Asso2 », cela signifie que « Asso1 » est incluse dans « Asso2 » ; une participation d'entités à « Asso1 » n'est alors possible que si ces entités participent également à « Asso2 ».

Voici quelques exemples d'utilisation des contraintes inter-associations :

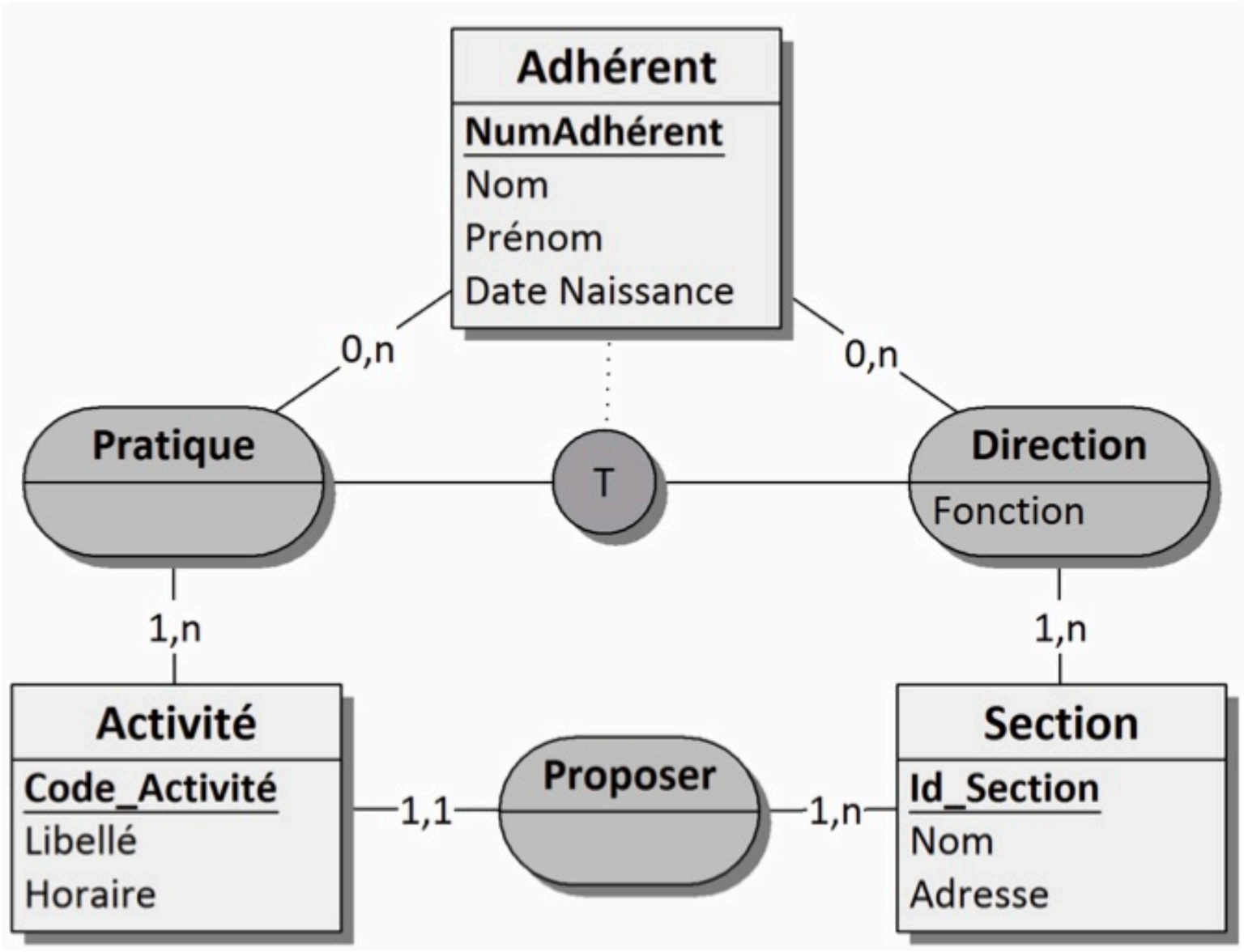
Exclusivité

Pour cette contrainte d'exclusion mutuelle, reprenons notre exemple sur le personnel d'un établissement scolaire dans sa version sans héritage :



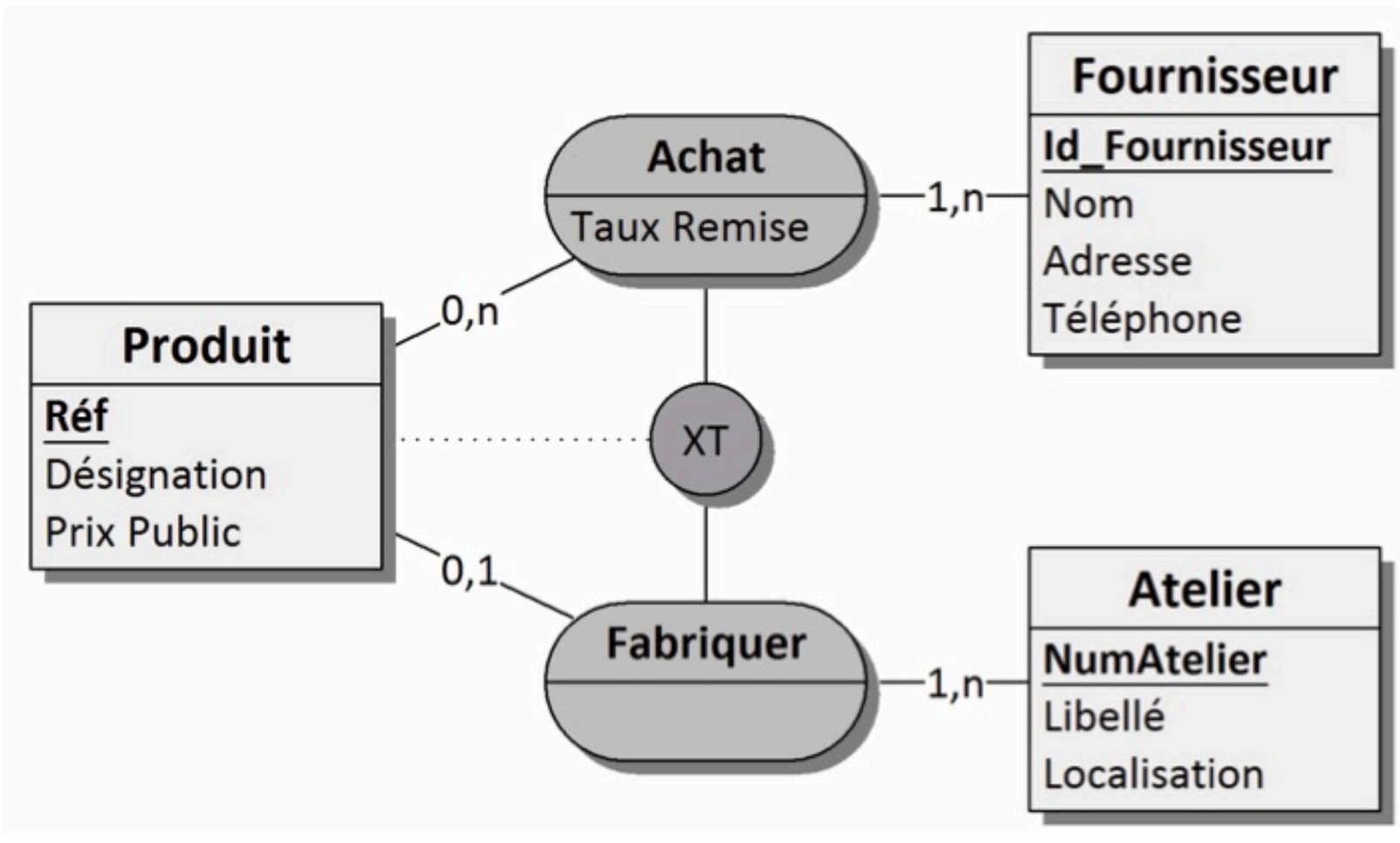
Totalité

Dans un club comprenant plusieurs sections sportives, tous les adhérents sont concernés, soit par la pratique d'au moins une activité, soit par la direction d'au moins une section, voire par les deux (pas d'exclusion mutuelle) :



Partition

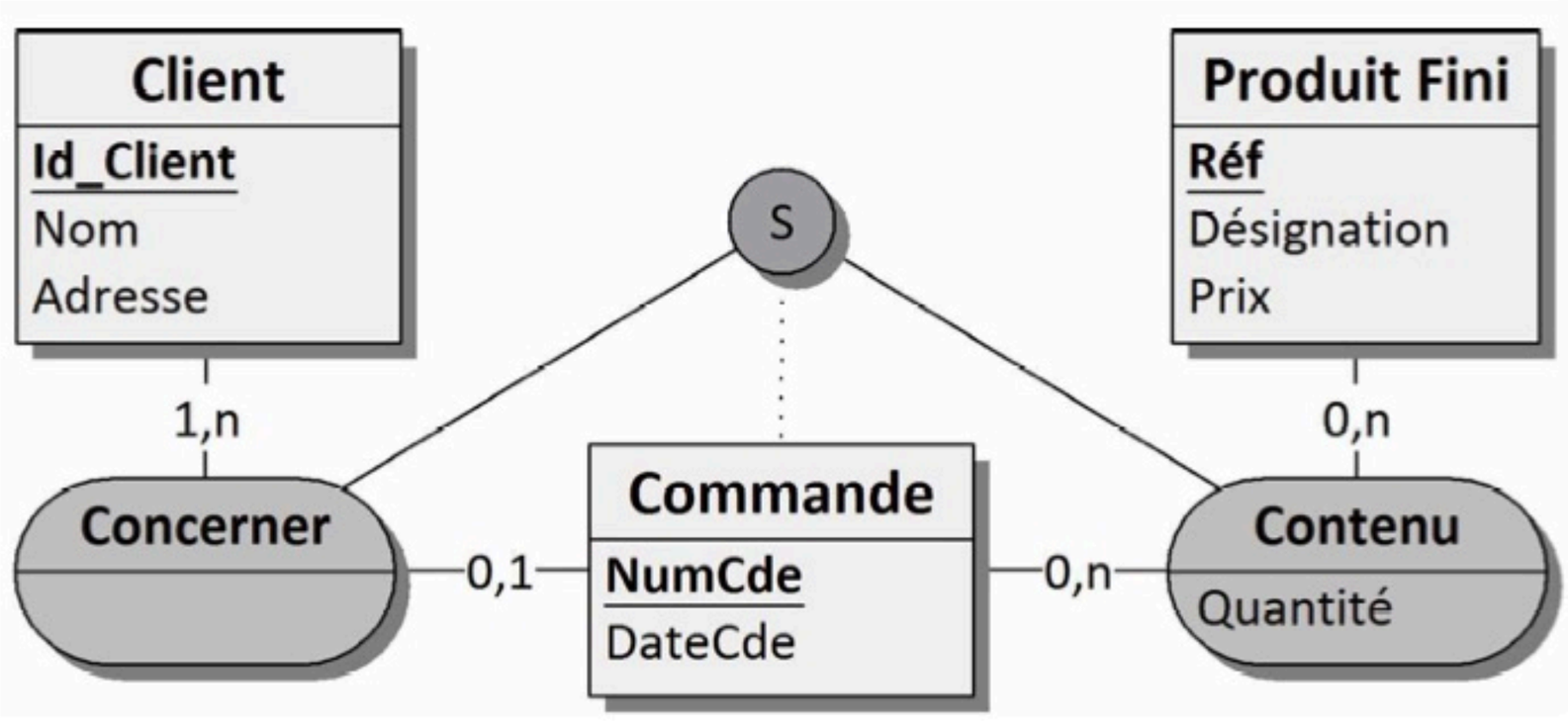
Tous les produits de l'entreprise sont, soit fabriqués dans un des ses ateliers, soit achetés à un des fournisseurs qui les propose :



© Patrick Bergougnoux

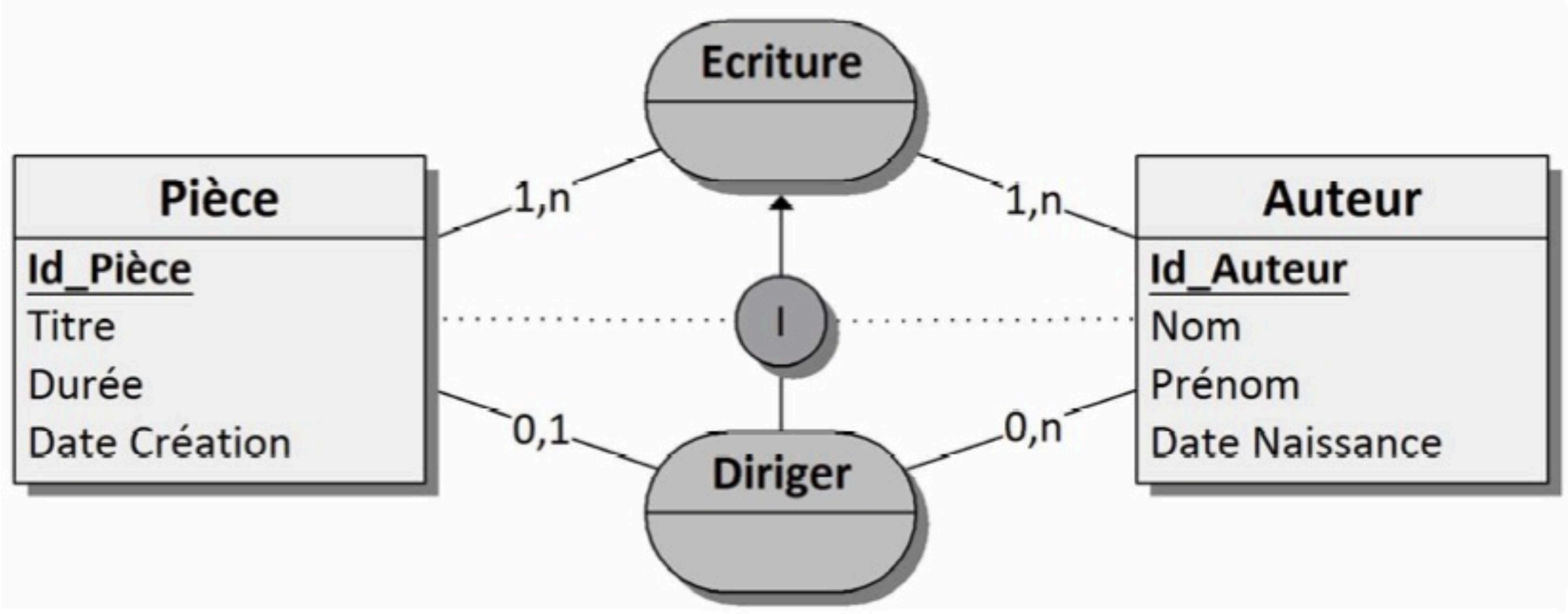
Simultanéité

Dans le cas d'une gestion de commandes qui peuvent concerner des clients ou être internes (d'où la cardinalité **0,1**) et qui peuvent alors contenir autre chose que des produits finis (des pièces détachées, par exemple), cette contrainte de simultanéité permet d'exprimer le fait qu'une commande contenant des produits finis concerne obligatoirement un client, et réciproquement :



Inclusion

Pour diriger une pièce de théâtre, un auteur doit l'avoir écrite :

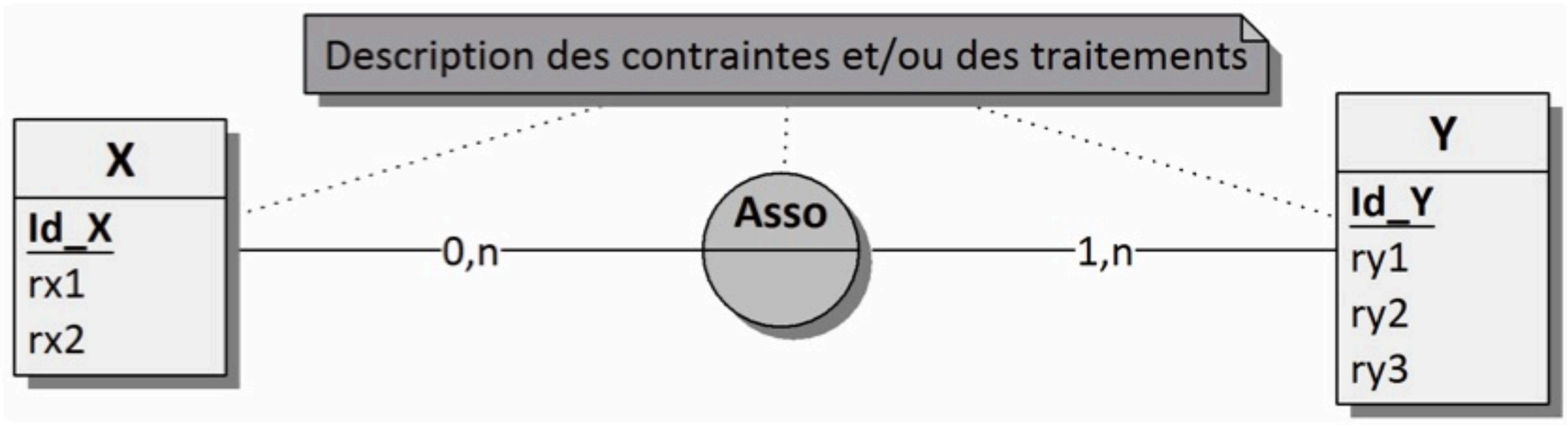


Expression d'autres contraintes

Règles complémentaires

Au-delà des contraintes standards que nous venons de traiter, il est possible de rajouter des **règles** qui explicitent des modalités spécifiques d'application de contraintes.

Il peut s'agir, par exemple, de la vérification de la présence d'une entité pour un identifiant donné, de la limitation du nombre d'occurrences dans le cadre d'une association, ou de tout autre traitement. Ces contraintes peuvent être spécifiées par des instructions SQL ou simplement décrites de manière informelle dans une zone texte prévue à cet effet au sein du MCDi :



Ce sont ces traitements qui permettent de répondre aux règles de gestion non prises en compte par le formalisme entité-association de base.

Pas d'incidence sur le MLD

Les contraintes d'intégrité inter-associations et les règles complémentaires viennent apporter des précisions au niveau des modèles conceptuels de données, mais n'ont aucune incidence sur le MLD qui décrit simplement la structure de la base de données.

Les requêtes SQL de création de la base de données ne sont pas modifiées non plus. Néanmoins, la plupart des SGBD proposent l'utilisation de triggers pour déclencher l'exécution d'instructions permettant l'application de ces contraintes.

La mise en œuvre de ces contraintes est spécifique à chaque SGBD et au langage qui lui est adossé (par exemple, le langage PL/PgSQL propre à PostgreSQL). Nous ne développerons pas davantage cet aspect dans ce livre.

Avec ***Looping***, les contraintes peuvent éventuellement être spécifiées au sein de règles complémentaires sous forme d'instructions SQL qui sont rajoutées au script de génération du schéma relationnel.

Ce qu'il faut retenir

- La spécialisation et la généralisation permettent à des sous-types de classes d'entités d'hériter des rubriques de leur sur-type.
- Les contraintes inter-associations et les règles complémentaires précisent les relations entre les différentes entités, et permettent de répondre aux règles de gestion définies dans le système.

© Patrick Bergougnoux

Chapitre 16

Diagramme de classes UML

Depuis le début de cet ouvrage, l'ensemble des modèles de données ont été réalisés avec le formalisme entité-association. Cependant, il existe d'autres types de notations permettant de représenter l'organisation des données.

L'alternative au schéma entité-association la plus répandue aujourd'hui est basée sur UML (Unified Modeling Language).

Dans ce chapitre, nous présentons les aspects d'UML qui permettent de traiter les modèles conceptuels de données, tout en mettant en évidence l'équivalence entre ce formalisme et celui des schémas entité-association que nous avons utilisé jusqu'à présent.

Pour illustrer le mieux possible cette modélisation, nous reprenons, avec les deux formalismes, l'ensemble des exercices traités dans cet ouvrage.

Nous en profitons également pour généraliser l'usage de noms communs au lieu de verbes pour les associations à cardinalités multiples de part et d'autre, celles-ci faisant l'objet de tables dans le MLD (qu'elles soient porteuses ou non des rubriques). Au-delà de l'intérêt que cela revêt lors de la manipulation de ces tables au sein de la base de données via SQL, la façon dont UML traite ce type d'associations se prête bien à cette évolution. Par contre, pour les associations possédant une cardinalité simple, nous conservons les verbes, ces associations ne faisant pas l'objet de tables dans le schéma relationnel de la base de données.

Présentation d'UML

UML est un Langage de Modélisation Unifié issu de la fusion des langages de modélisation objet Booch, OMT et OOSE. UML est aujourd'hui le standard adopté par l'OMG (Object Management Group).

UML propose un ensemble de diagrammes qui permettent la modélisation d'un système tout au long de son cycle de vie.

MERISE vs UML

Avant d'aller plus loin, il convient de clarifier certains aspects concernant des comparaisons souvent erronées entre différents concepts. Il est en effet fréquent de lire des articles mettant en concurrence MERISE et UML, alors que leur finalité est bien différente.

- MERISE (Méthode d'Etude et de Réalisation Informatique pour les Systèmes d'Entreprise) est une méthode d'analyse et de conception permettant d'élaborer des systèmes d'information. Nous en avons vu une partie aux deux premiers chapitres de cet ouvrage avec, en particulier, le modèle organisationnel de flux.
- UML (Unified Modeling Language) est un langage de modélisation proposant exclusivement un ensemble de formalismes pour la conception de logiciels à base d'objets. **UML n'est donc pas une méthode** : c'est un langage auquel il faut associer une démarche de développement comme RUP (Rational Unified Process).

MERISE et UML sont davantage complémentaires que concurrents. D'ailleurs, rien n'empêche d'utiliser des diagrammes UML dans le cadre de la méthode MERISE.

Schéma entité-association et diagramme de classes UML

Si l'on se recentre sur le thème de cet ouvrage, c'est la modélisation de données qui nous intéresse.

Compte-tenu de ses origines, UML est particulièrement adapté pour la conception et la mise en œuvre d'architectures logicielles développées avec un langage objet. Se voulant « unifié », UML propose aussi un formalisme pour modéliser les données.

Quel diagramme propose UML pour la modélisation des données ?

© Patrick Bergougnoux

Parmi les diagrammes proposés par UML, c'est le **diagramme de classes** qui va permettre de réaliser des modèles proches de ceux effectués avec les schémas entité-association.

Avant de continuer, réglons un autre problème de comparaison douteuse ! J'attends parfois : « on fait un modèle conceptuel de données ou un diagramme de classes UML ? ».

Euh… et bien remplaçons « **ou** » par « **avec** », et tout va bien !

Nous pouvons effectivement utiliser UML pour réaliser des MCD, au même titre que nous utilisons les schémas entité-association.

En résumé, nous pouvons par exemple réaliser un MCD sous forme de diagramme de classes UML dans le cadre d'une démarche MERISE.

MCD et diagramme de classes UML

Un formalisme utilisé partiellement

Comme indiqué précédemment, UML est essentiellement conçu pour la modélisation objet. Or, pour modéliser les bases de données relationnelles (qui représentent l'immense majorité des bases de données existantes… et à venir), nul besoin de concept objet.

Le diagramme de classes ne sera donc utilisé que partiellement puisque les méthodes associées aux objets n'existent pas dans les MCD destinés aux bases de données relationnelles.

Pourquoi utiliser UML pour modéliser les bases de données relationnelles ?

En termes de modélisation de bases de sonnées, les diagrammes de classes UML n'apportent rien de plus que les schémas entité-association, et affichent même parfois quelques limites. Il s'agit donc juste d'une question de mode confortée par le côté « unification » avec les autres diagrammes… et le fait que le formalisme entité-association s'exporte mal en dehors des pays francophones.

Quoiqu'il en soit, les schémas entité-association restent particulièrement adaptés à la modélisation conceptuelle des données. Cet ouvrage se doit cependant de présenter les deux formalismes, en mettant en évidence leur équivalence.

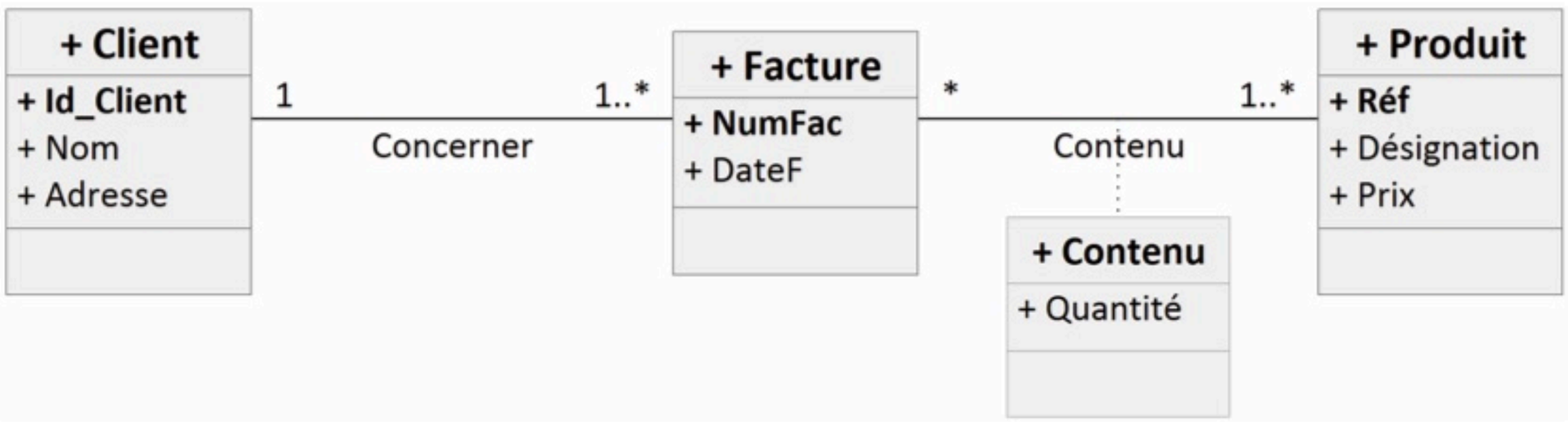
Équivalence avec le formalisme entité-association

Le diagramme de classes UML ne diffère pas beaucoup du schéma entité-association. La transformation peut être décrite en 4 étapes simples :

- Reprise des classes d'entités en l'état avec rajout en partie basse d'une zone théoriquement destinée à recevoir les méthodes des objets mais qui, dans notre cas, restera systématiquement vide ; il faut également noter que le soulignement de la clé primaire n'est pas repris, cette notation étant réservée en UML aux attributs de classes.
- Rajout du signe « + » devant le nom des rubriques (nommées « attributs » en UML) et des classes d'entités : cela signifie que les données sont publiques ; si elles étaient privées, seules les méthodes décrites dans leur classe pourraient y accéder (on utiliserait alors le signe « – »), mais comme il n'y a jamais de déclaration de méthodes dans ces modèles, toutes les informations doivent être publiques pour être accessibles depuis le SGBD.
- Suppression des associations : c'est un simple trait qui relie les classes d'entités ; lorsqu'une association est porteuse de rubriques, une nouvelle classe est créée avec le même nom que l'association et est reliée au lien par un trait en pointillés.
- Inversion des cardinalités (nommées « multiplicités » en UML) : c'est le changement le plus important, et surement le plus troublant lorsque l'on compare les deux formalismes, sachant que, si des rôles sont précisés, eux ne sont pas inversés ; de plus, l'écriture est légèrement différente :

1,n ➔ 1..* / 0,n ➔ * (ou 0..*) / 1,1 ➔ 1 / 0,1 ➔ 0..1

Voici ce que cela donne avec la facturation simplifié de Beaux Rivages :



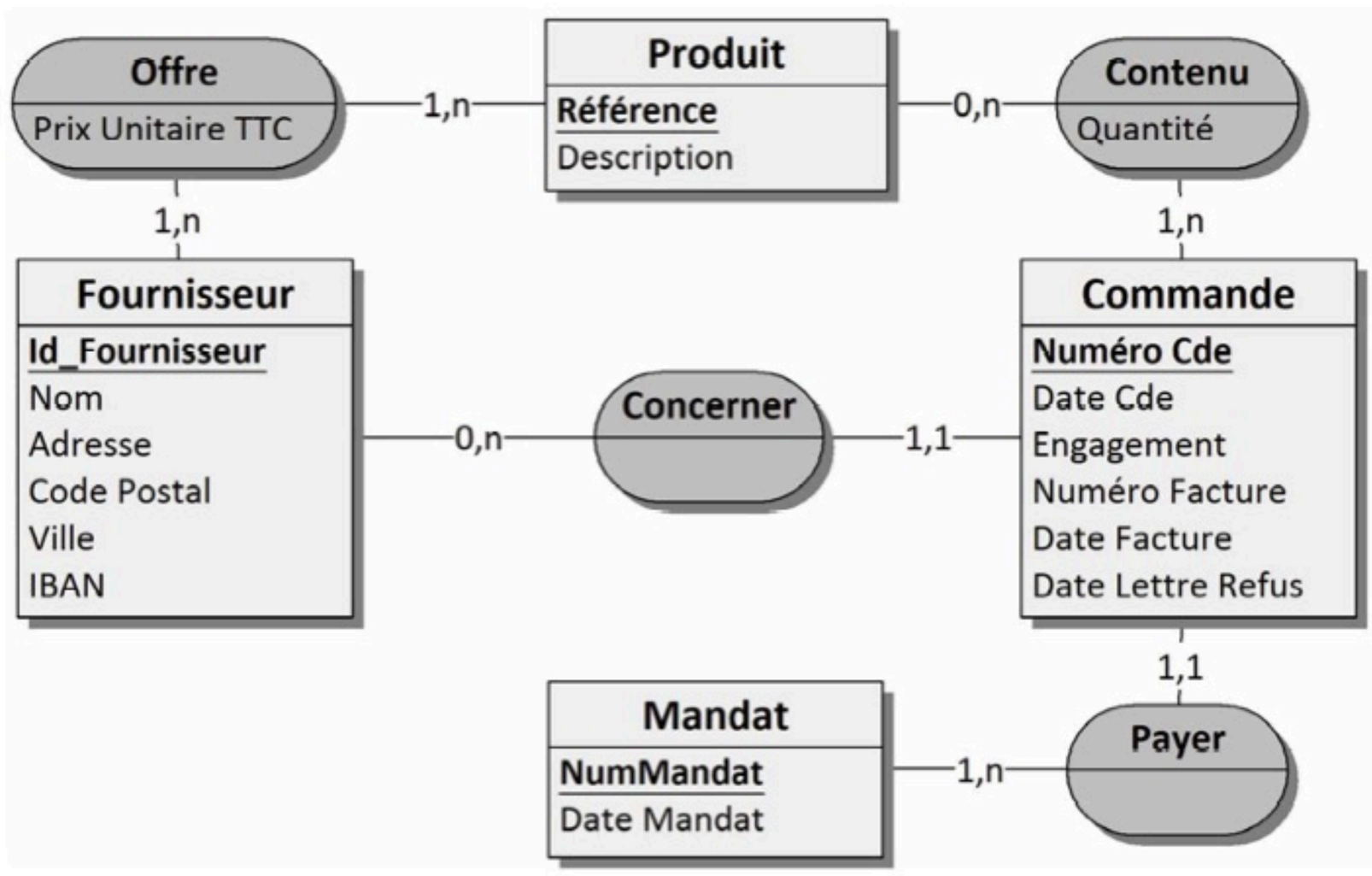
Il s'agit d'une simple transformation graphique de quelques éléments de modélisation qui ne change rien sur le fond du modèle.

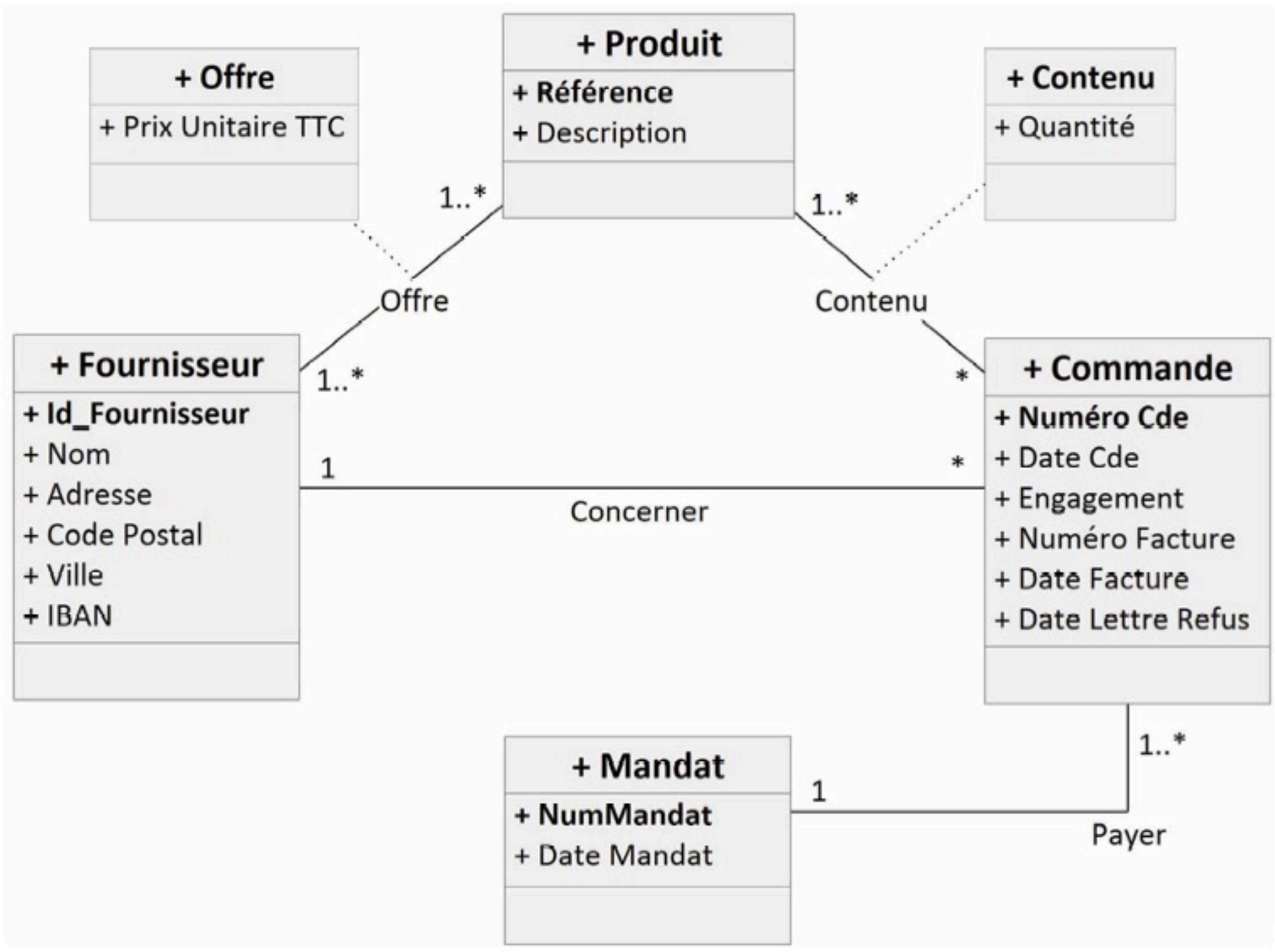
Nous allons maintenant reprendre l'ensemble des exercices traités dans cet ouvrage en apportant, si nécessaire, des précisions sur la notation UML d'éléments spécifiques de modélisation avancée (identifiant relatif, association n-aire, CIF, héritage, contraintes inter-associations).

© Patrick Bergougnoux

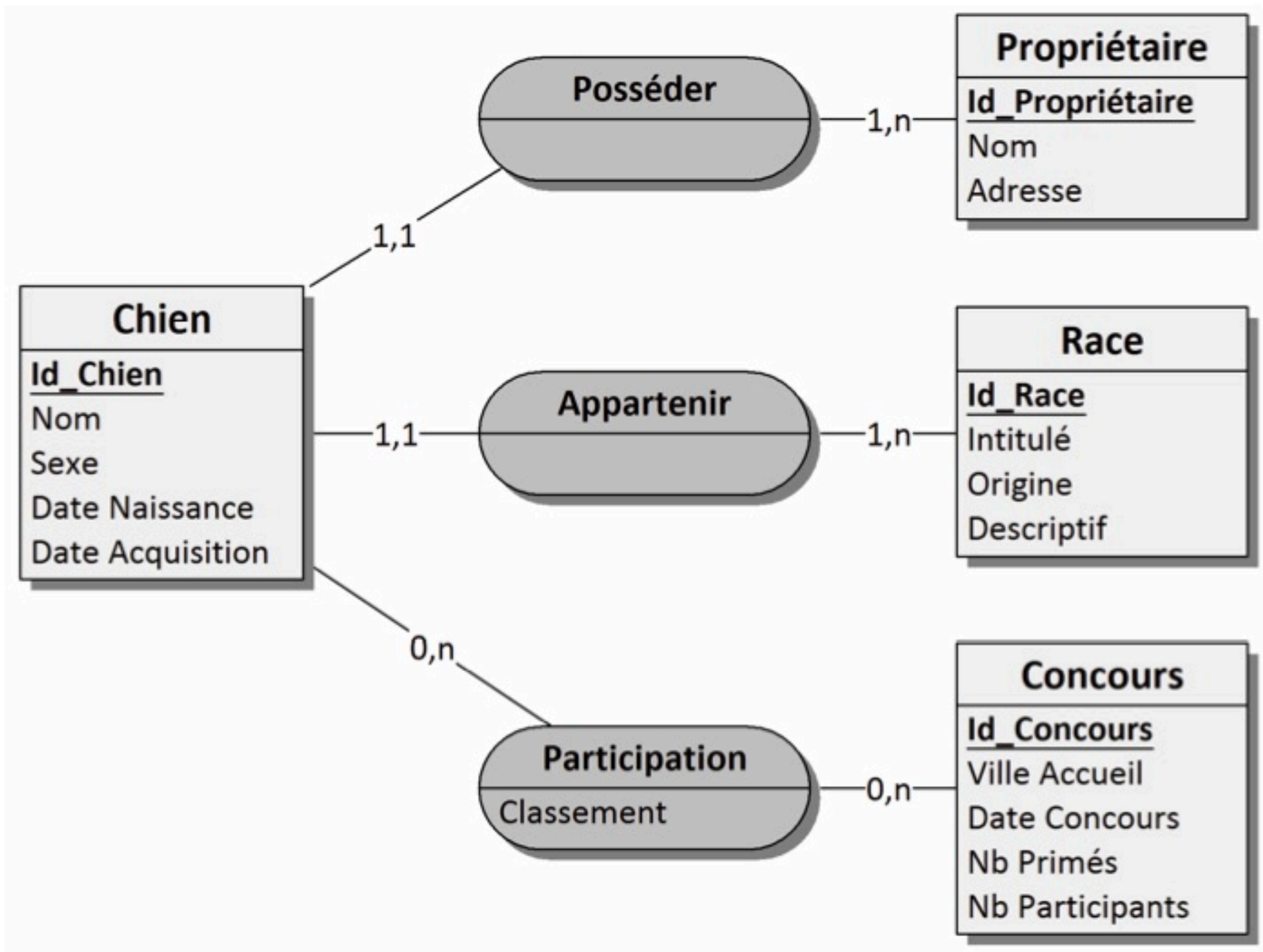
Reprise des exercices en UML

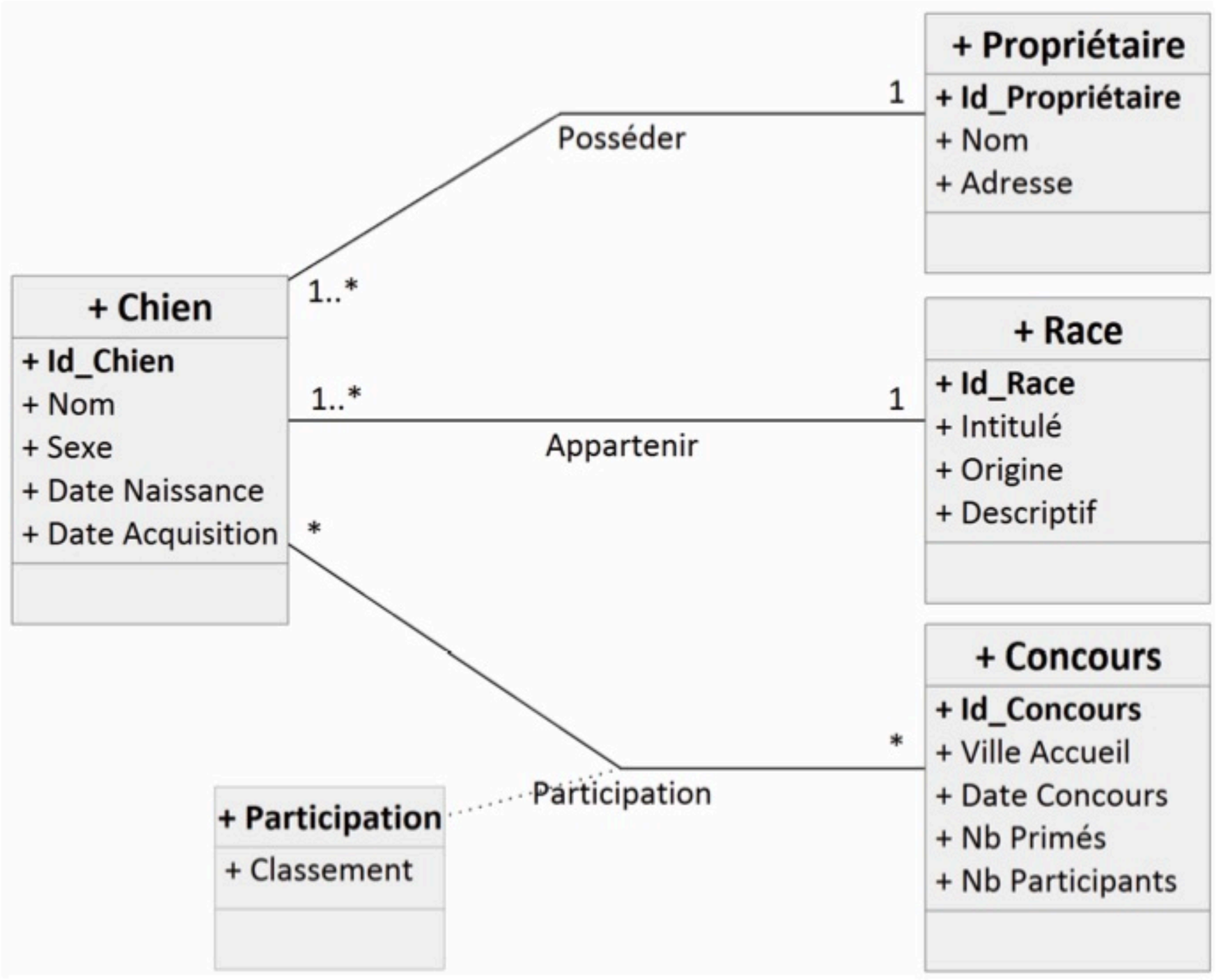
Gestion des achats d'une administration





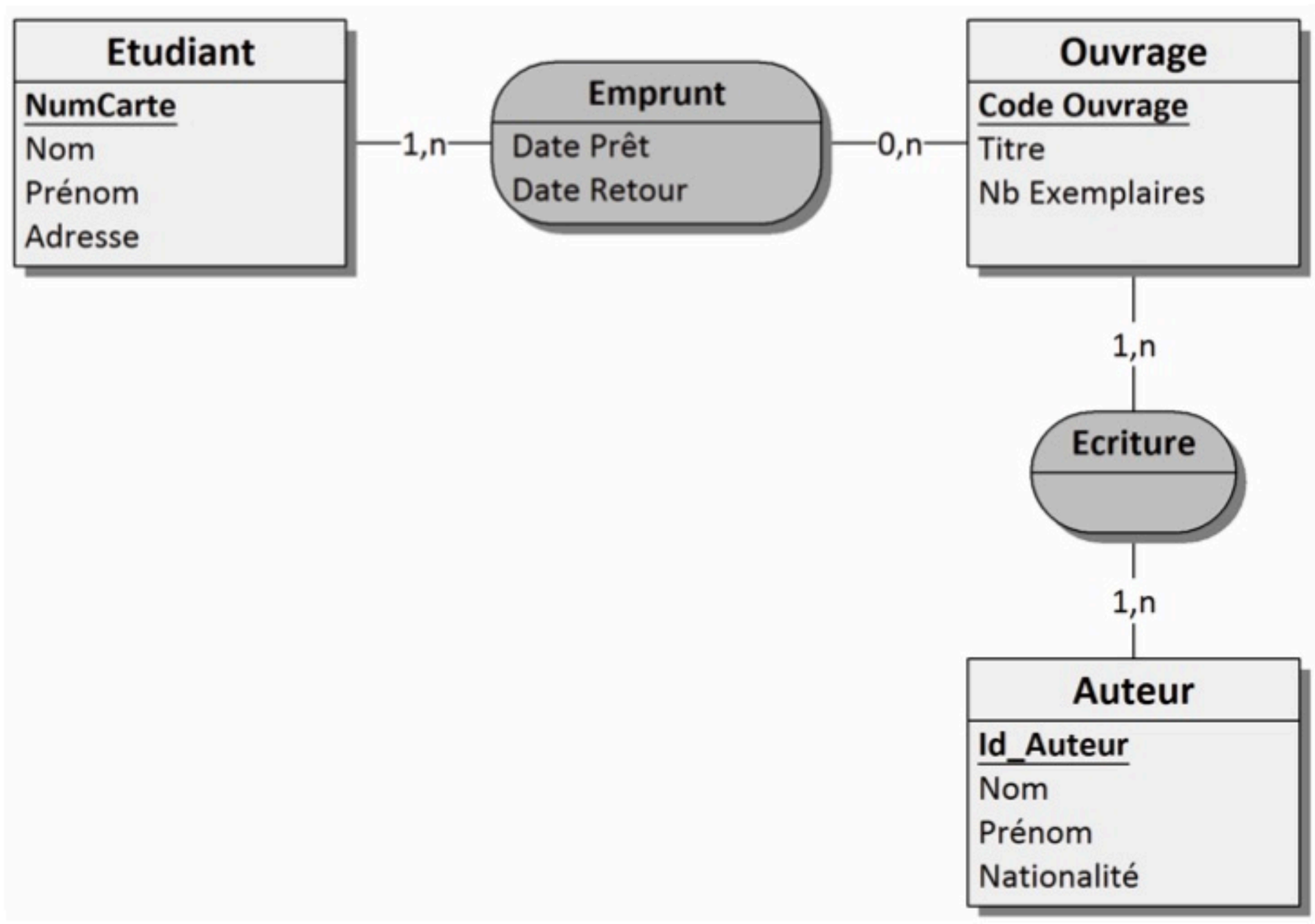
Société canine

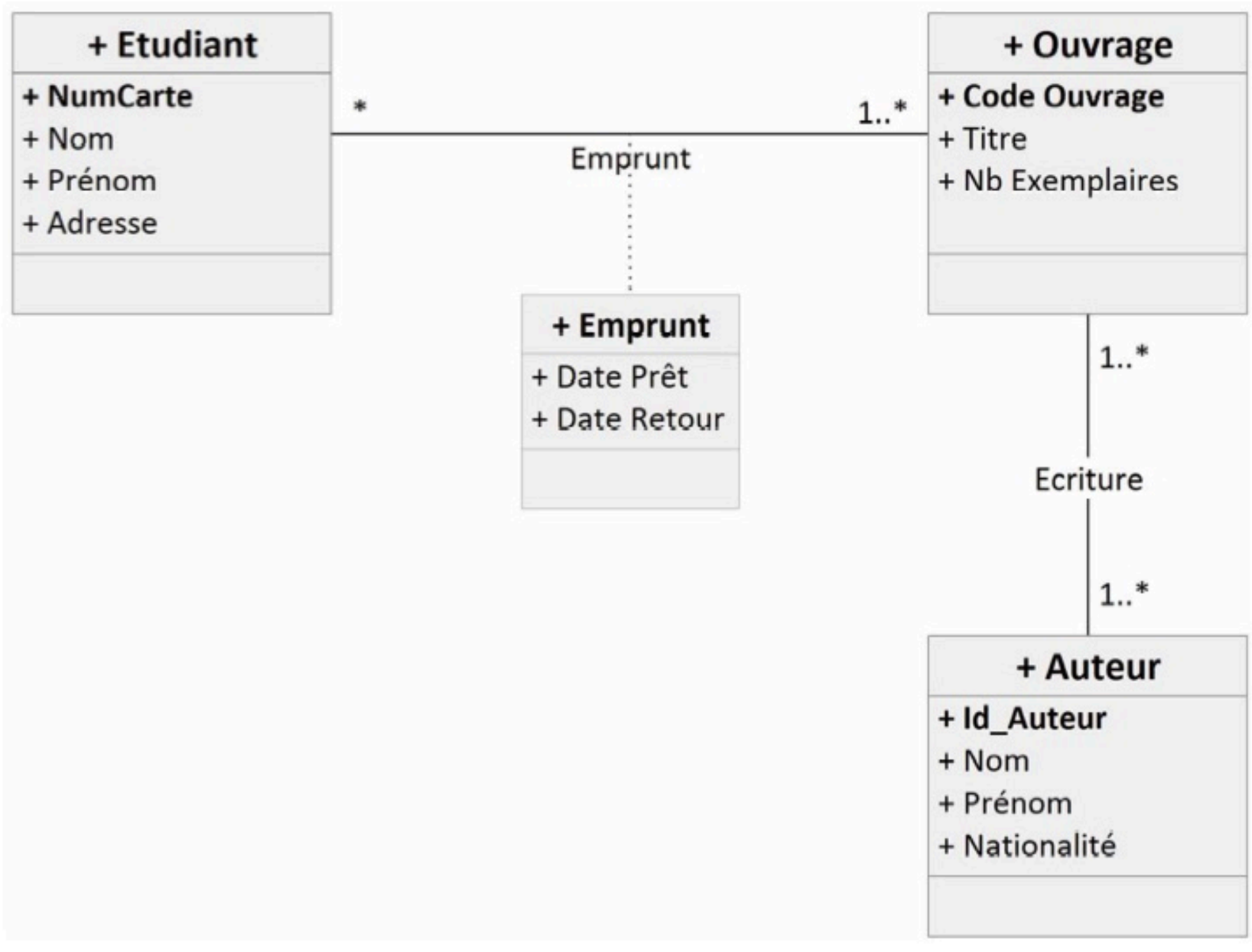




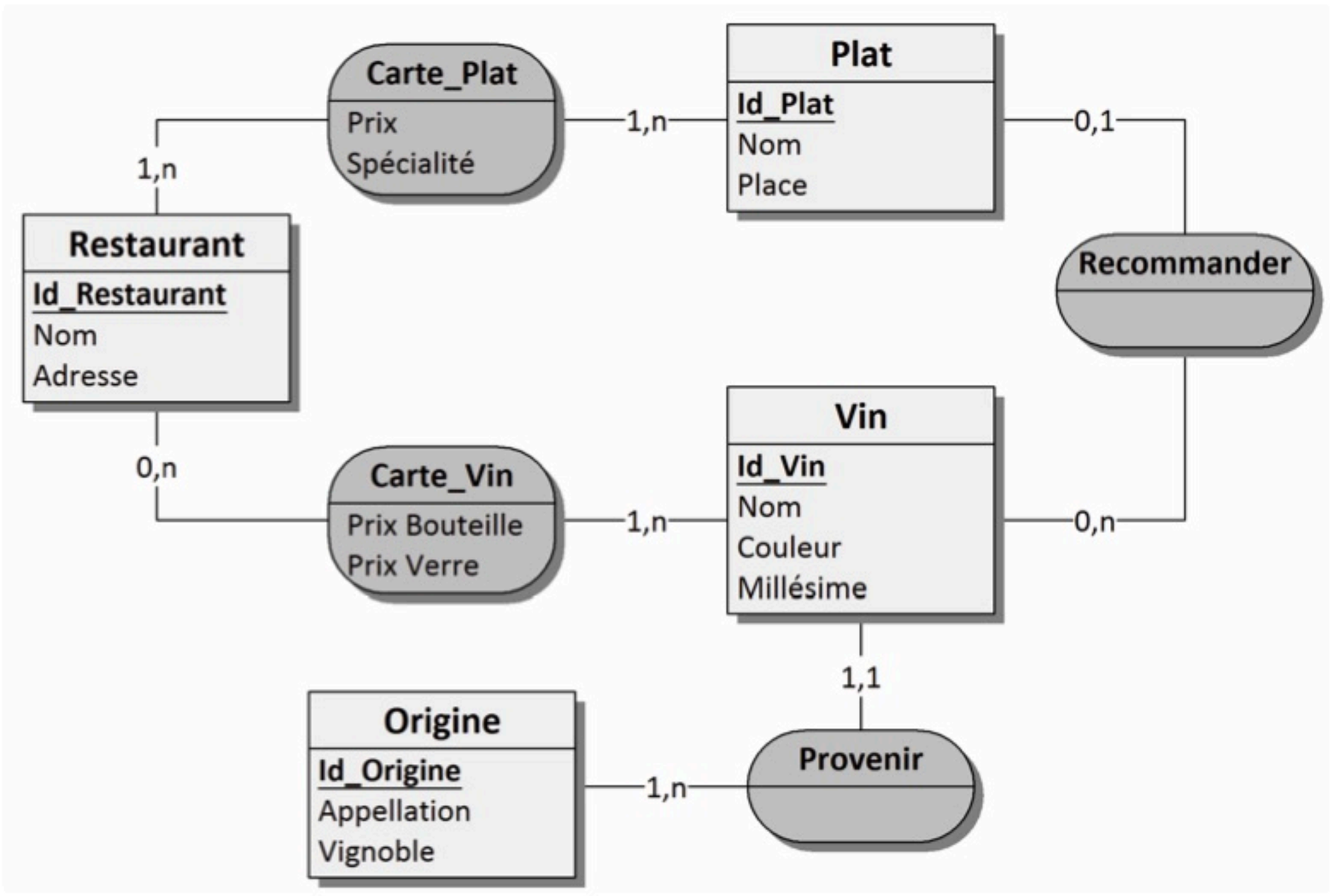
© Patrick Bergougnoux

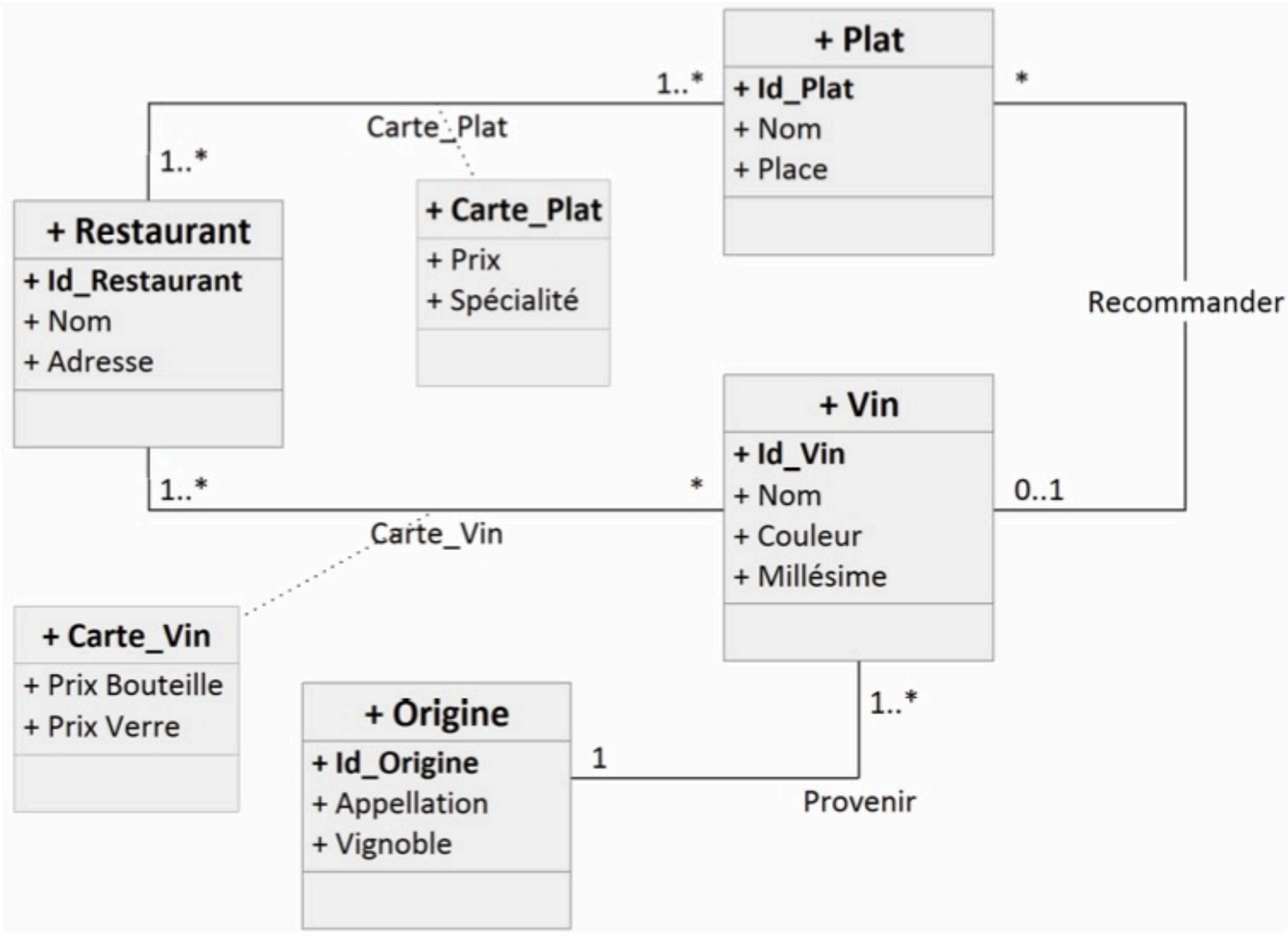
Bibliothèque universitaire





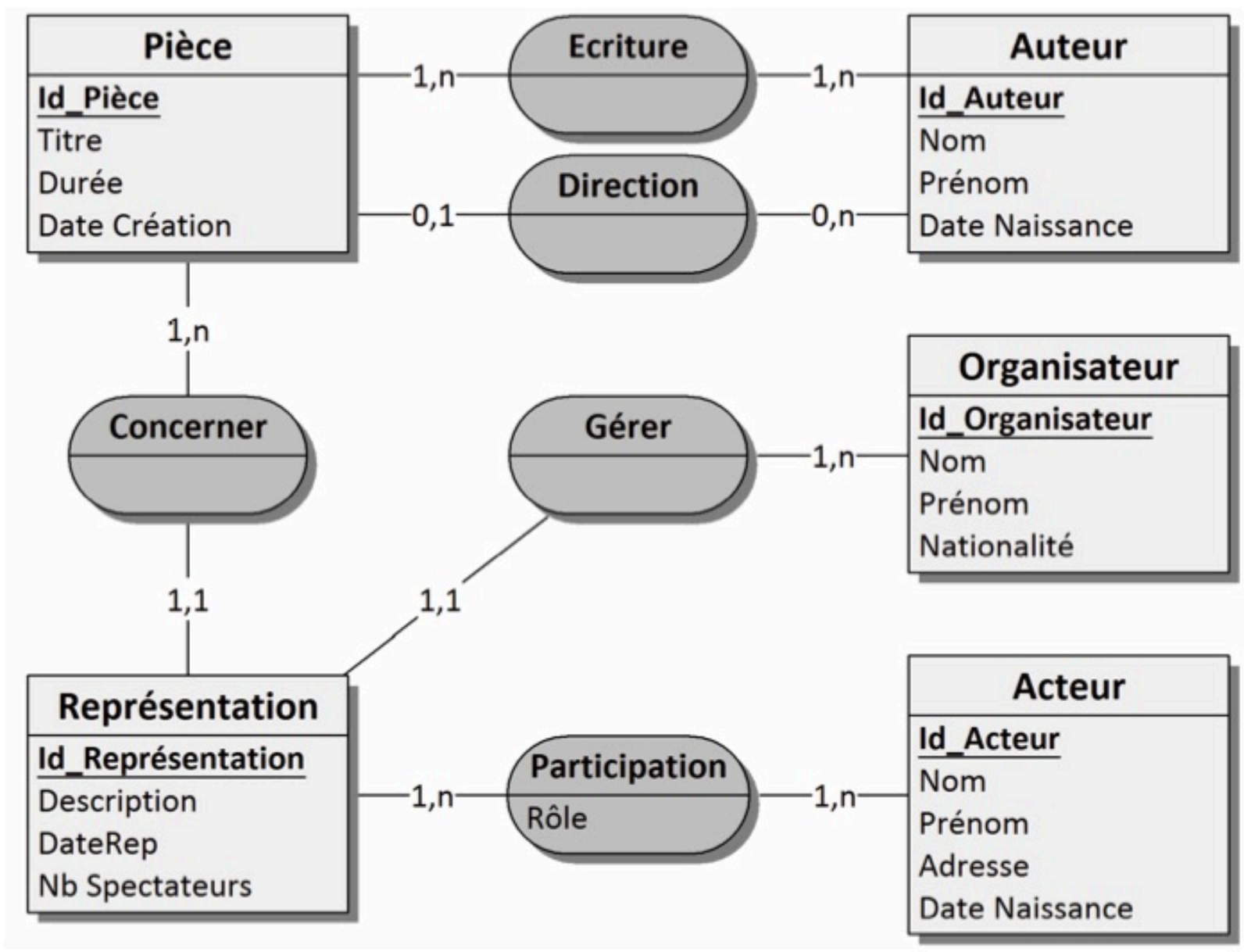
Guide Gastronomique

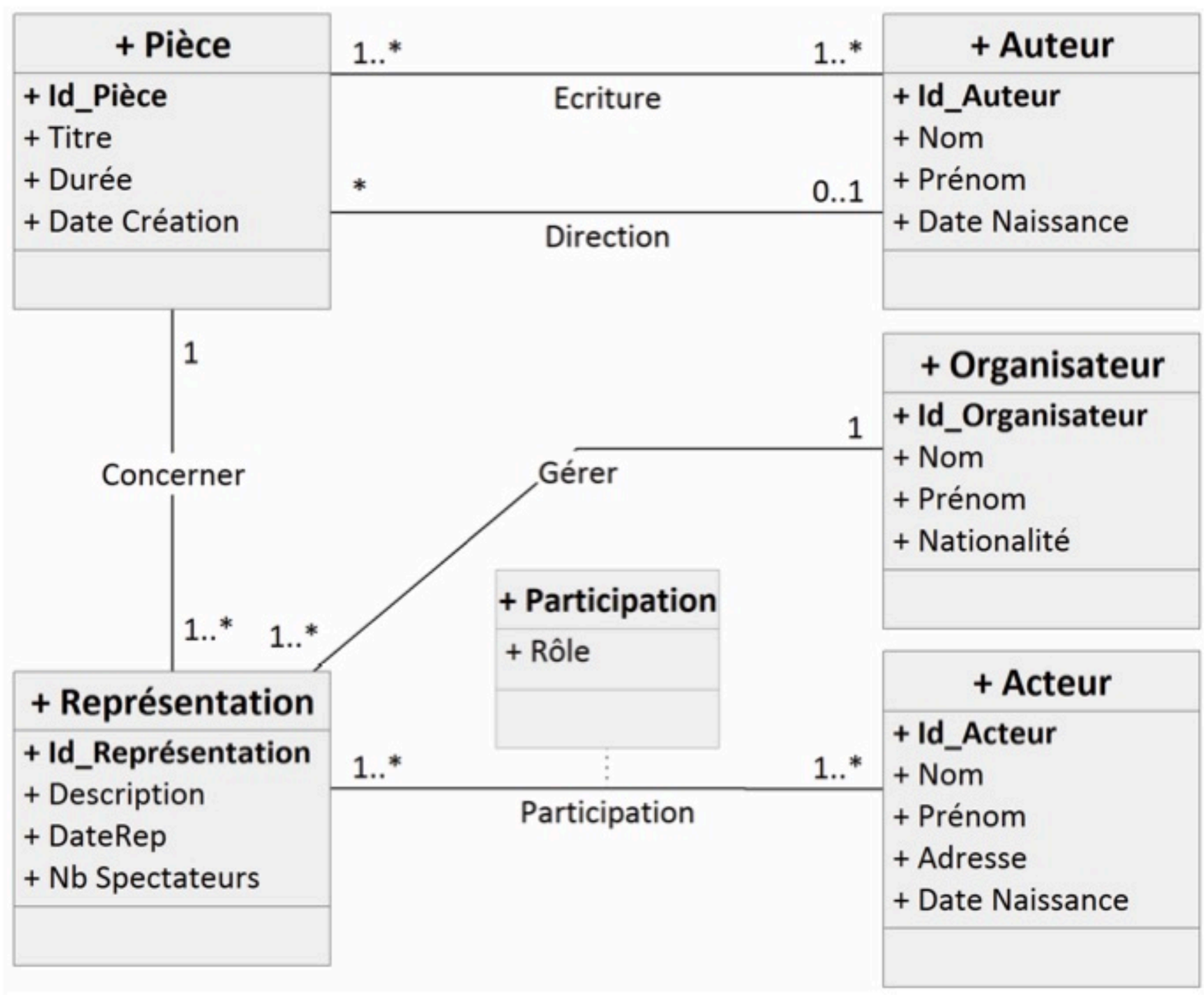




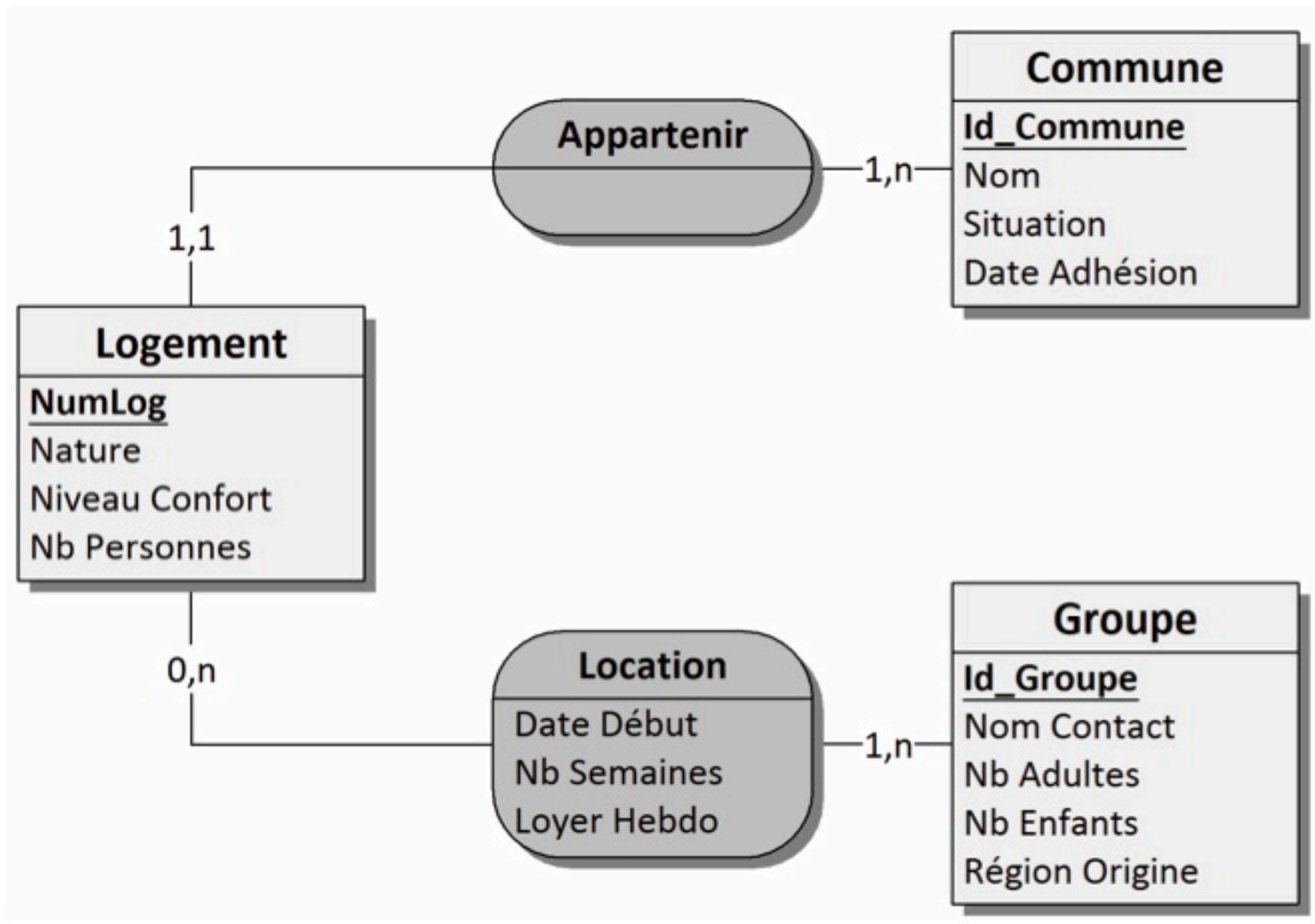
© Patrick Bergougnoux

Représentations théâtrales

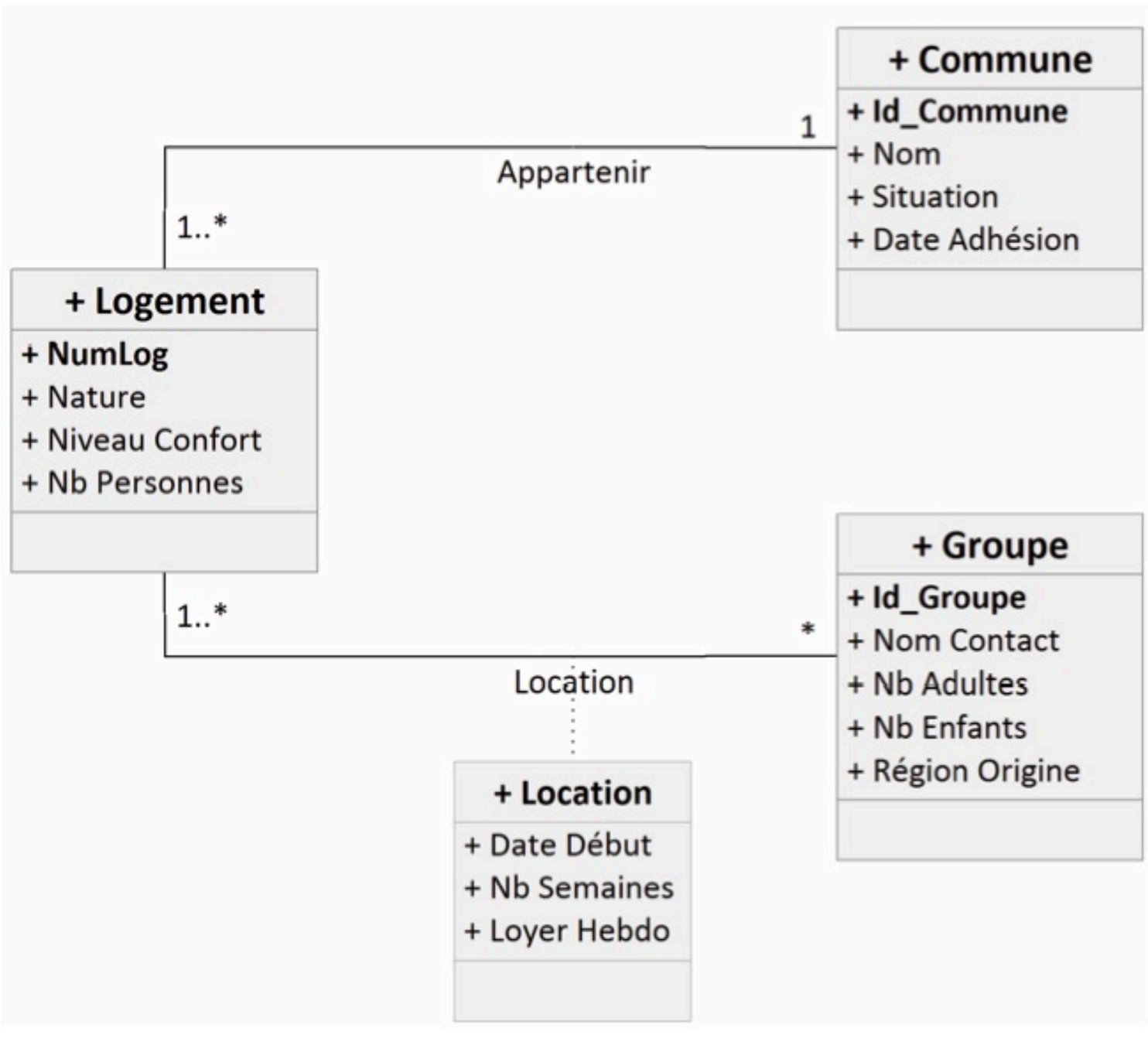




Location de logements

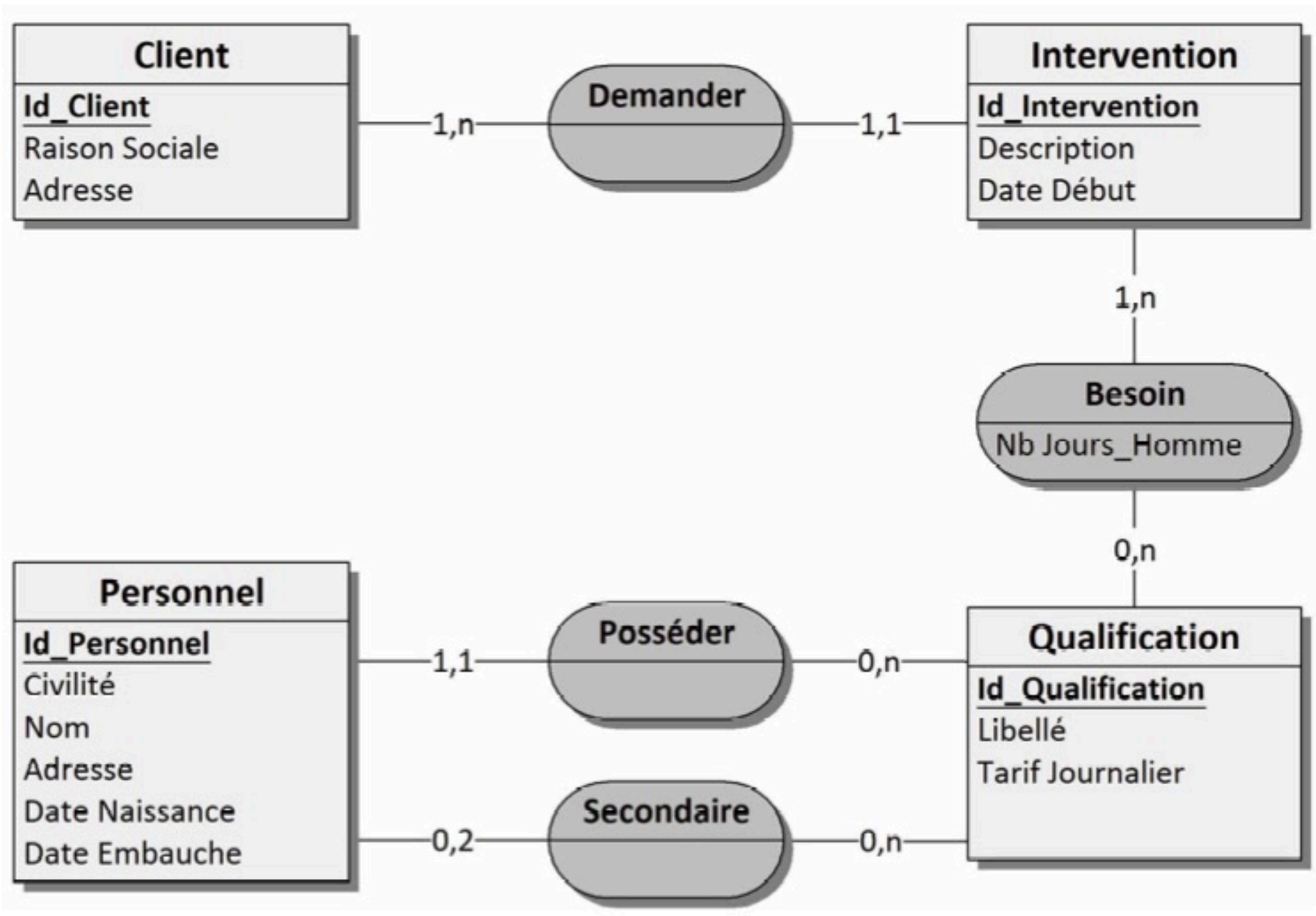


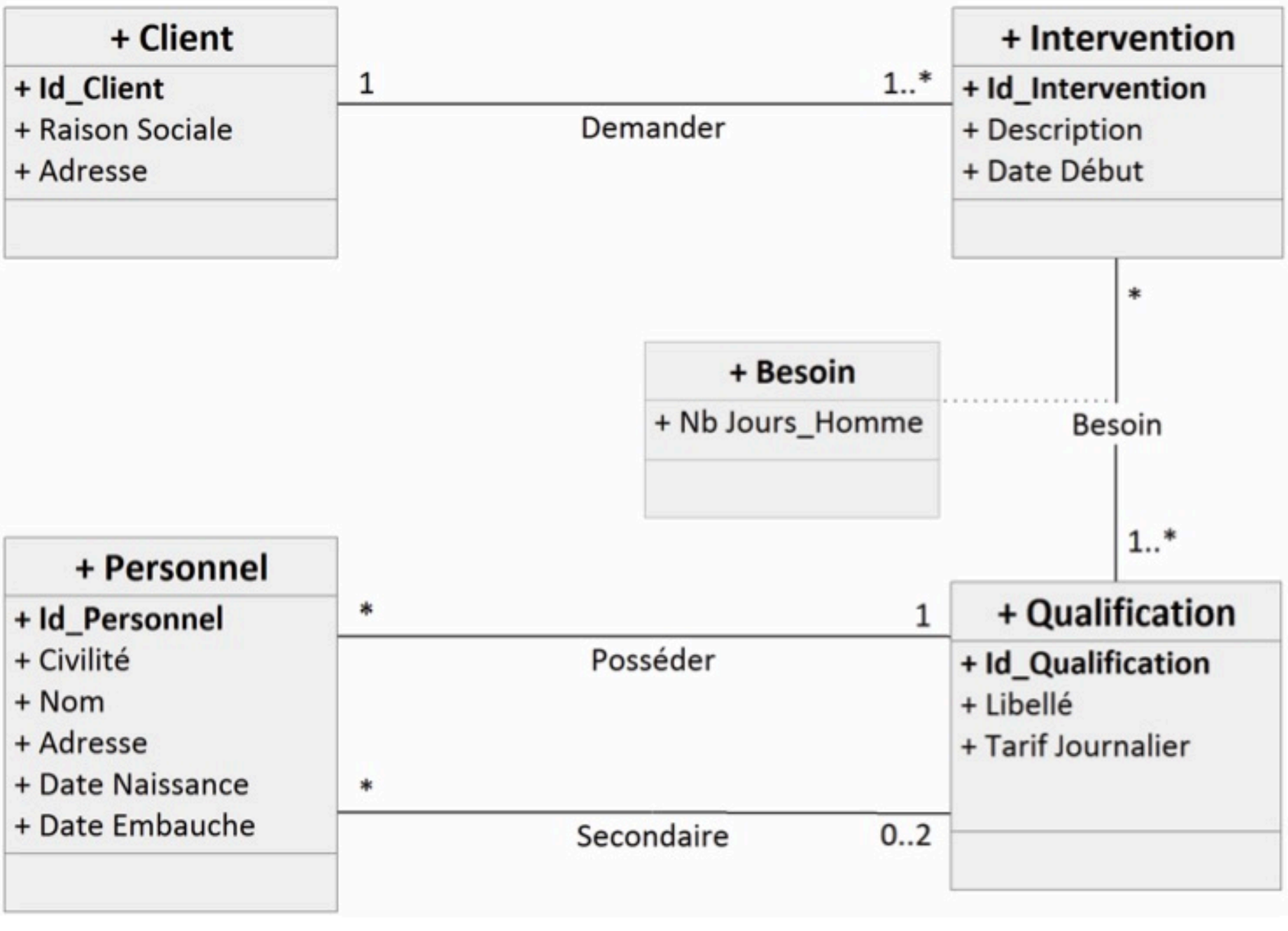
E/A
UML



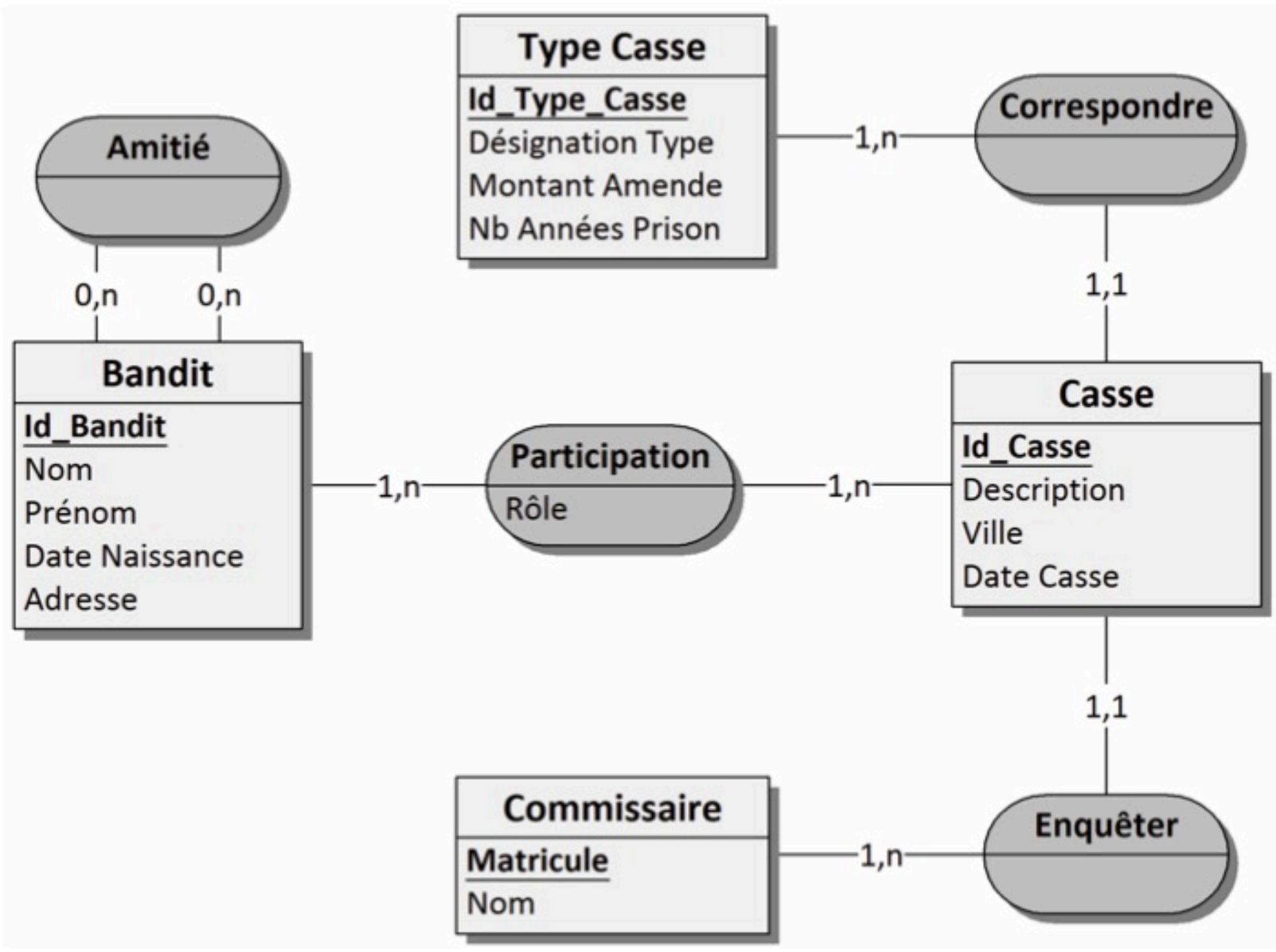
© Patrick Bergougnoux

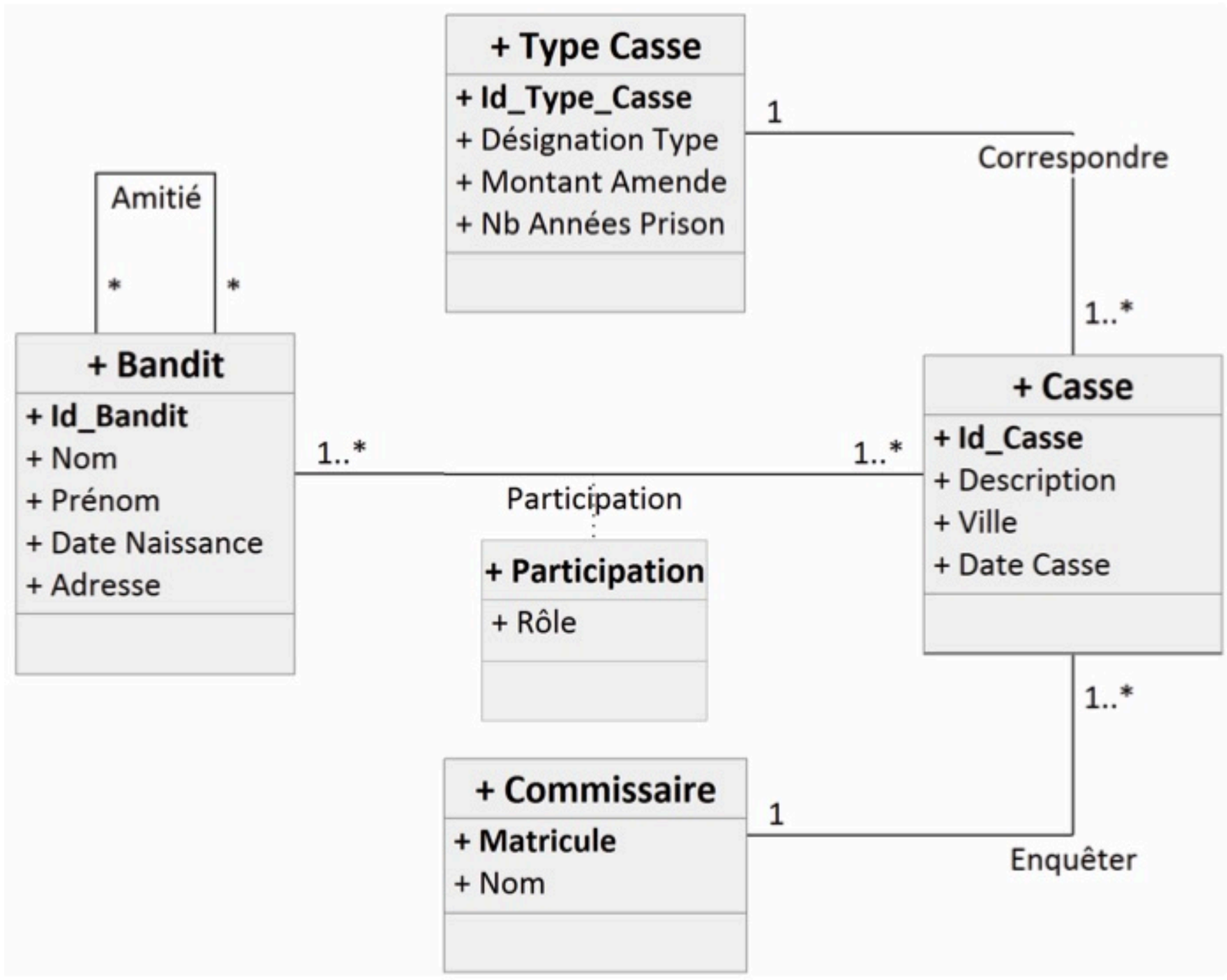
Agence de travail temporaire





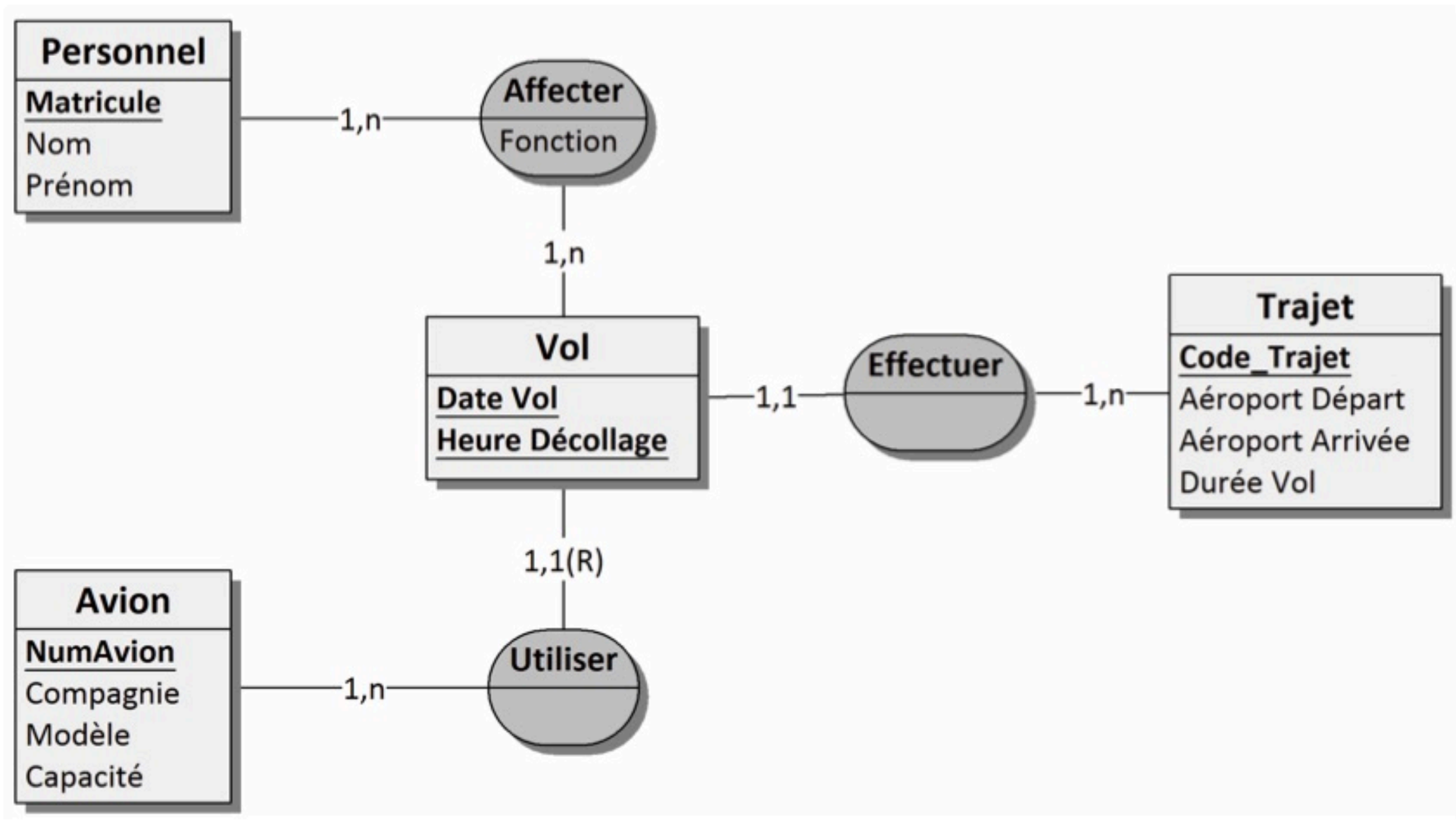
Gendarmes et voleurs

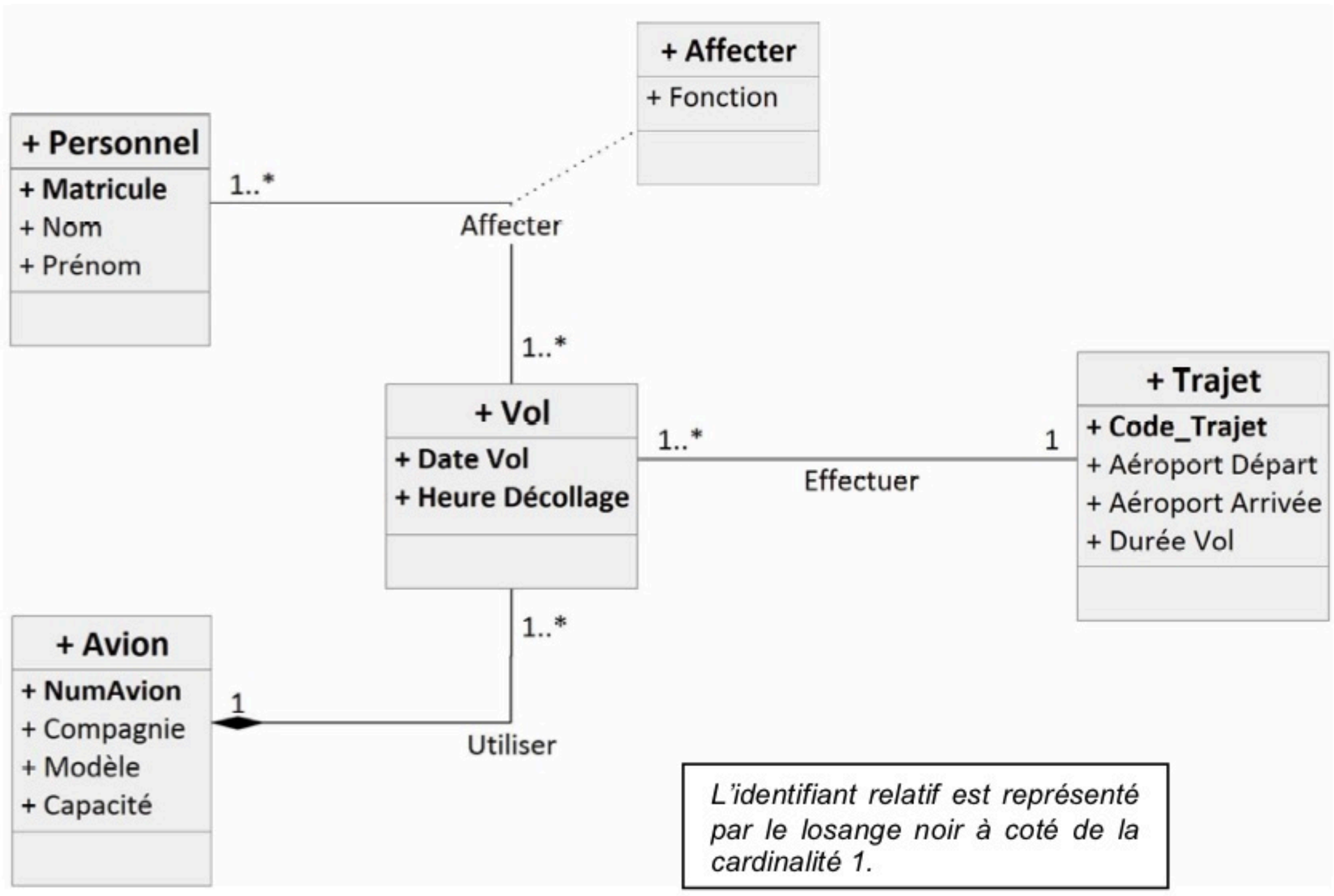




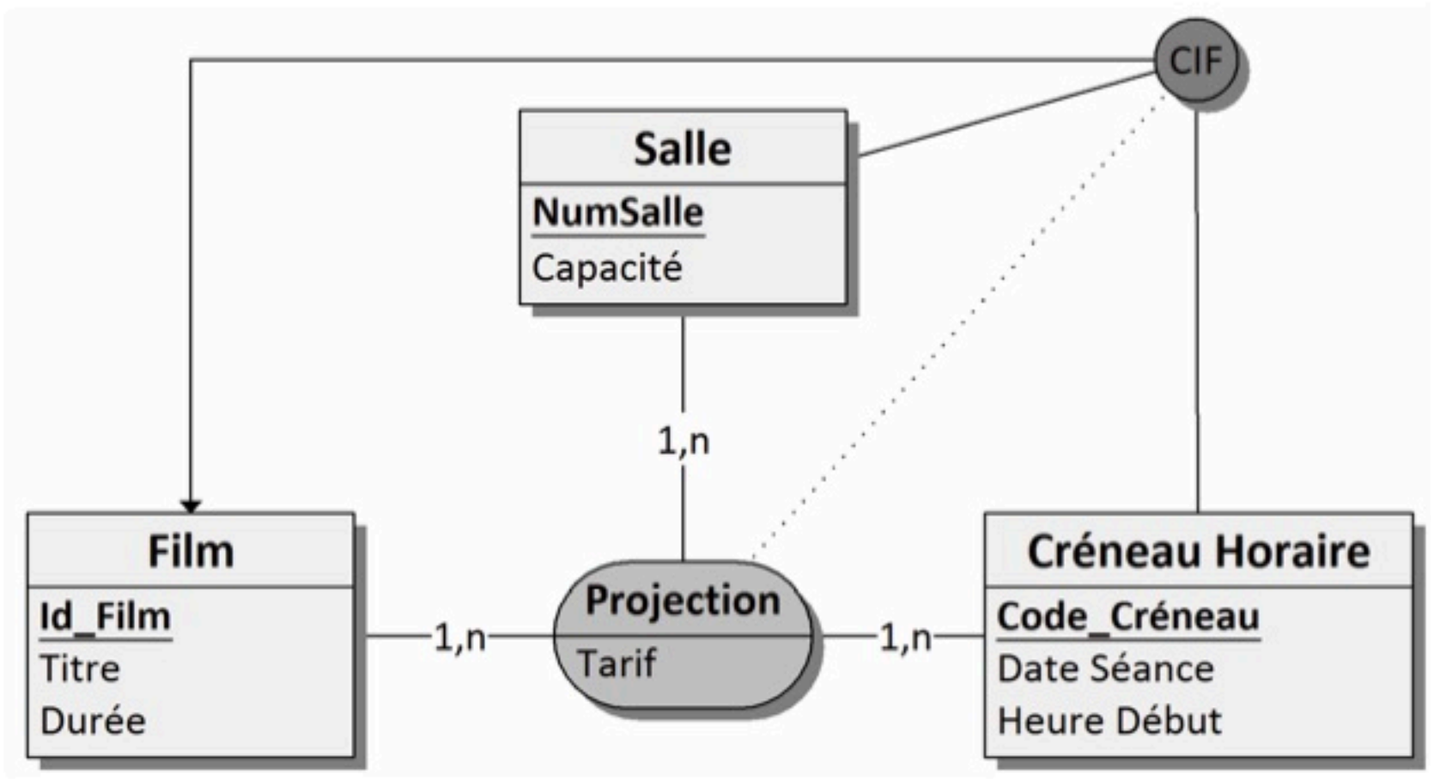
© Patrick Bergougnoux

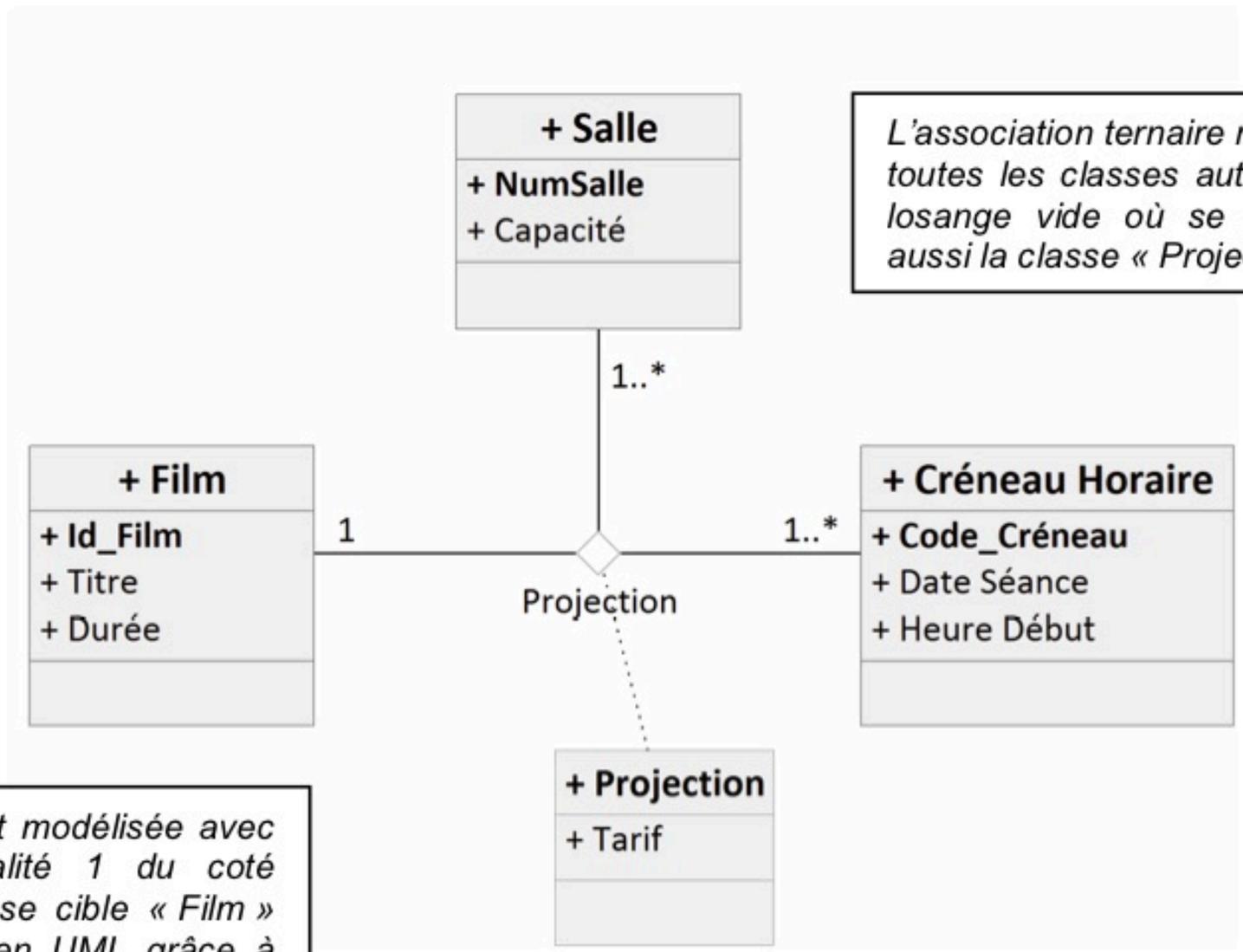
Compagnie aérienne





Cinéma



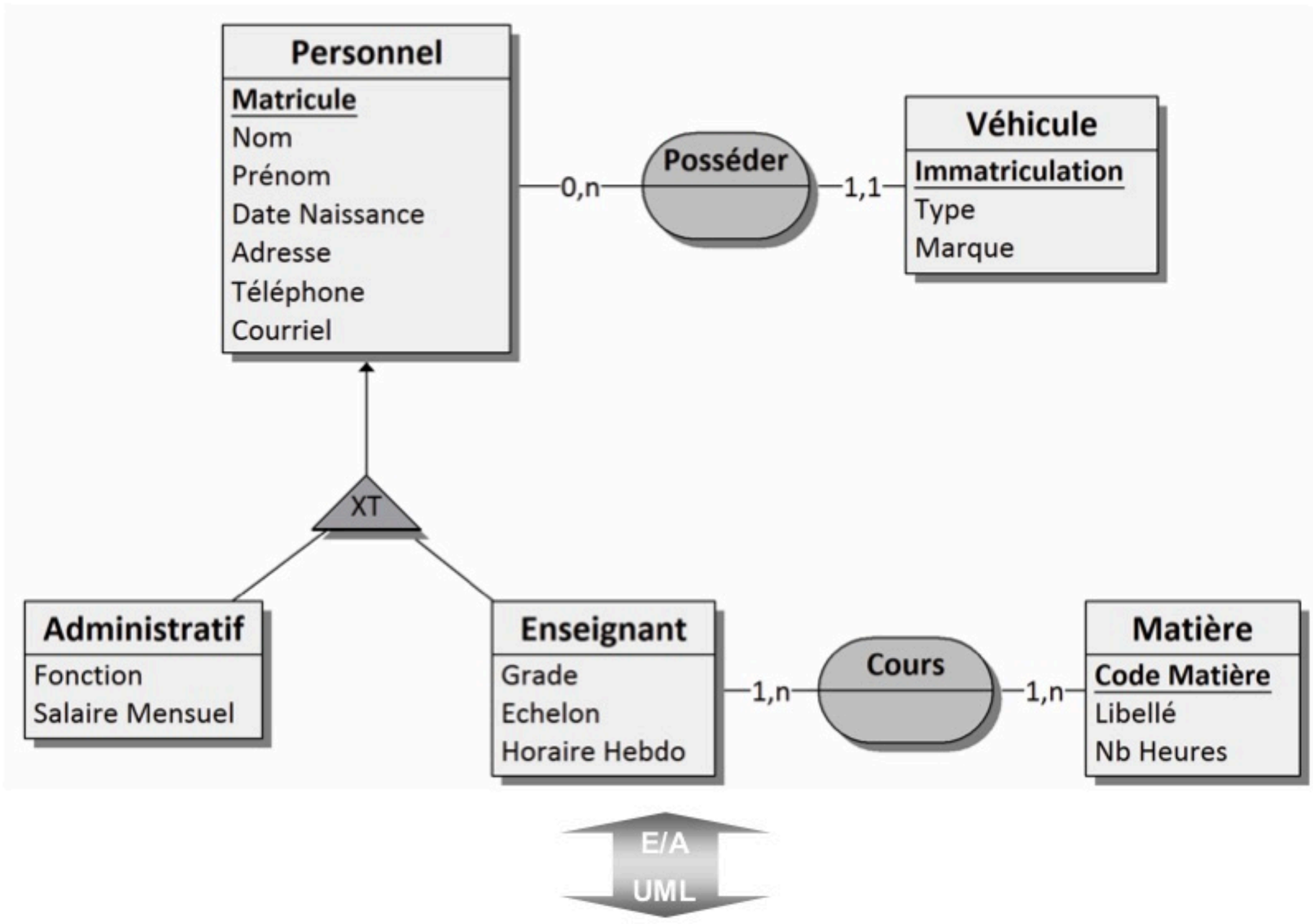


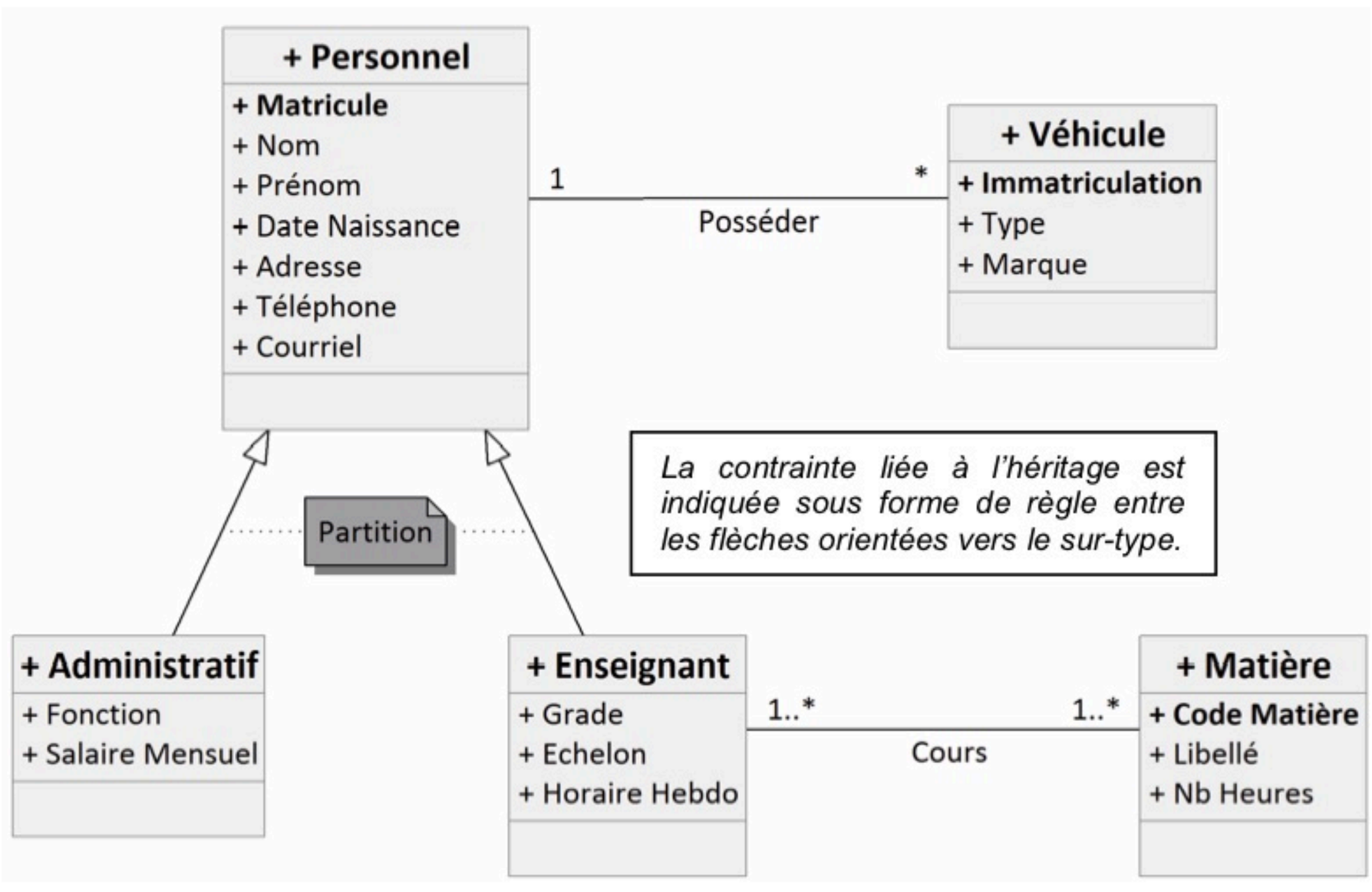
L'association ternaire regroupe toutes les classes autour d'un losange vide où se rattache aussi la classe « Projection ».

La CIF est modélisée avec la cardinalité 1 du coté de la classe cible « Film » (possible en UML grâce à l'inversion des cardinalités).

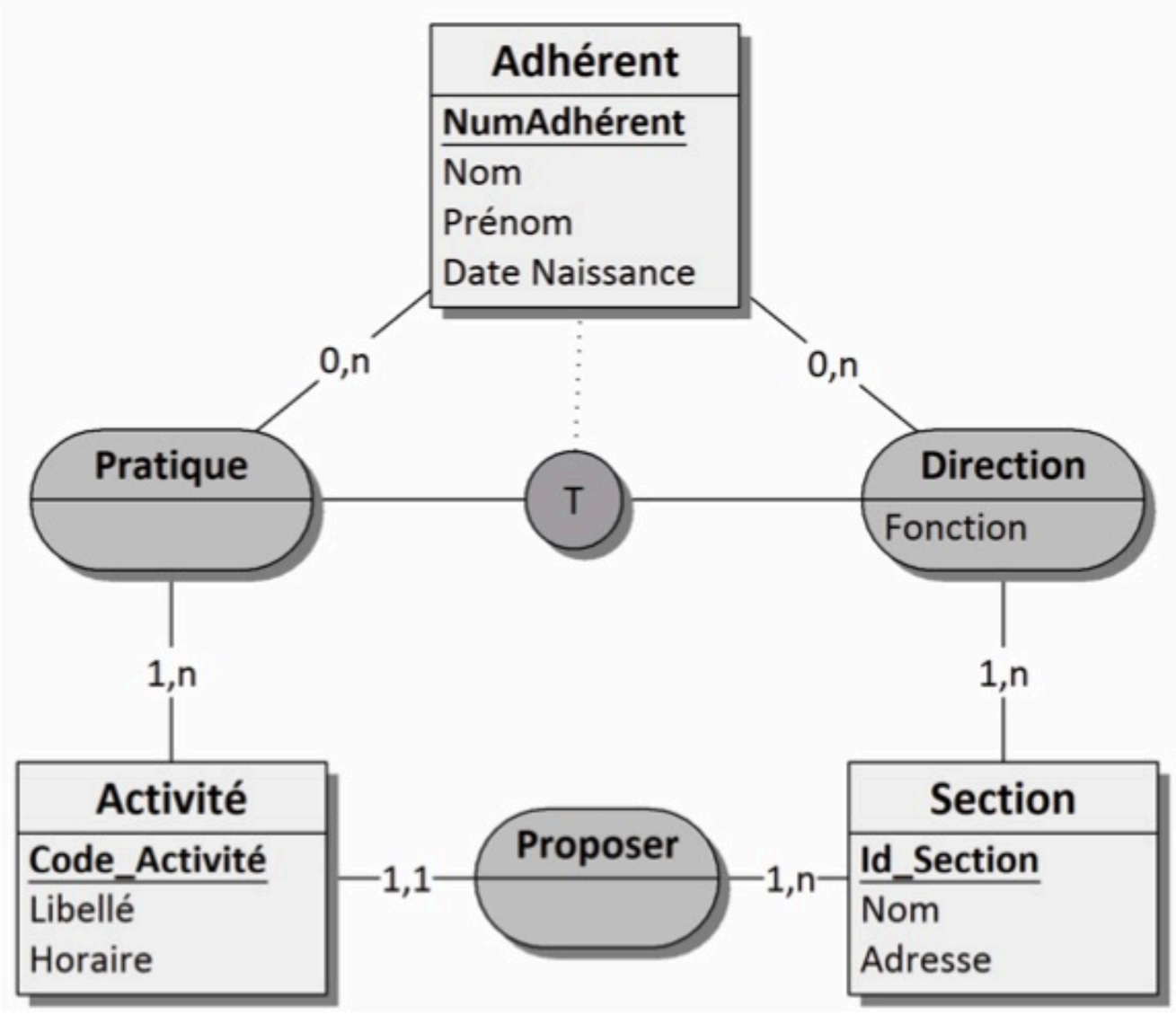
© Patrick Bergougnoux

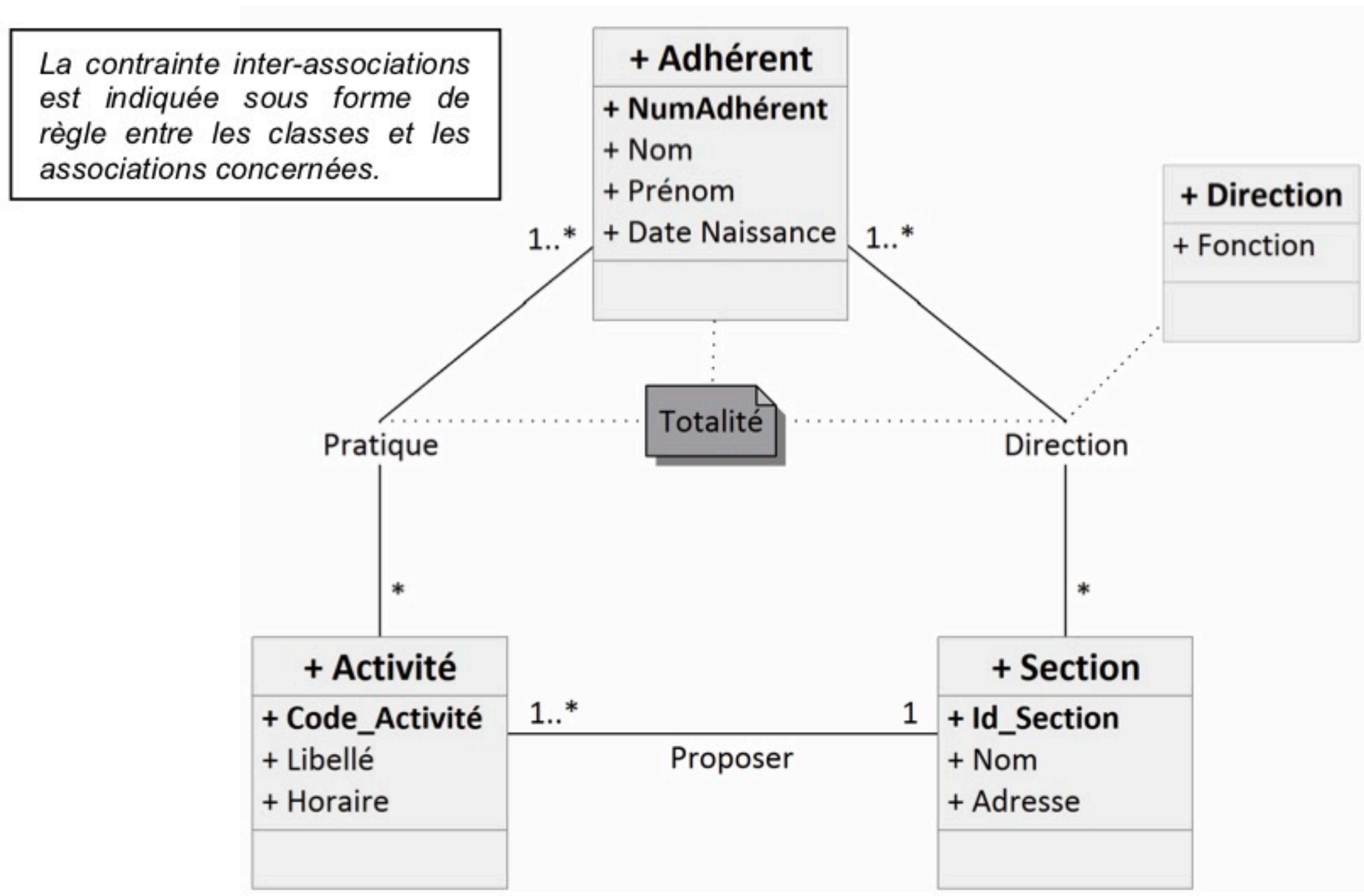
Établissement scolaire





Club sportif





© Patrick Bergougnoux

Ce qu'il faut retenir

- UML n'est pas une méthode mais un langage de modélisation proposant de nombreux diagrammes.
- Le diagramme de classes UML permet de réaliser des modèles conceptuels de données équivalents à ceux effectués avec le formalisme entité-association.
- La différence entre les deux formalismes est uniquement graphique : la forme des modèles change, mais le fond reste identique. La différence la plus importante est l'inversion des cardinalités, sachant que si des rôles sont précisés, eux ne sont pas inversés.
- Le vocabulaire est lui aussi un peu différent : en UML, les rubriques se nomment « attributs » et les cardinalités se nomment « multiplicités ».
- Un MCD réalisé avec l'un ou l'autre de ces formalismes donne le même résultat au moment de la génération du MLD et du schéma relationnel de la base de données correspondante.
- Le passage d'un formalisme à l'autre peut être automatisé grâce aux équivalences existant entre leurs composantes graphiques.

Looping dispose d'une option « UML » qui transforme automatiquement et instantanément un schéma entité-association en diagramme de classes UML.

Notes

Notes

Cinquième partie

Études de cas complètes

Dans cette dernière partie, nous mettons en œuvre les connaissances acquises dans cet ouvrage pour réaliser quatre études de cas complètes. Pour chacune d'entre elles, nous partons de l'analyse du compte-rendu d'interview et de ses annexes pour arriver au schéma relationnel de la base de données, en passant par le recensement des informations, le modèle conceptuel de données (E/A et UML), le modèle logique de données et les requêtes SQL de création des tables de la base.

Ces exercices s'inspirent de cas réels, certes complets, mais néanmoins nettement simplifiés par rapport à la réalité, en particulier concernant le nombre de rubriques prises en compte dans le système d'information. La complexité de la structure des modèles reste, quant à elle, tout à fait réaliste.

Chapitre 17

Association sportive et culturelle

L'ASCG (Association Sportive et Culturelle de Gagnac) regroupe l'ensemble des activités associatives de la commune et souhaite automatiser la gestion de ses adhérents pratiquant leurs activités dans les différentes sections de l'association.

Compte-rendu d'interview

Description du système d'information

L'ASCG est composée d'une vingtaine de sections qui proposent, chacune, une ou plusieurs activités décrites sur une fiche de présentation distribuée aux adhérents par les bénévoles en charge des sections (cf. annexe 1).

À chaque nouvelle saison, les sections réinitialisent leurs adhésions pour ne se préoccuper que de l'année en cours : les adhésions des années précédentes ne sont donc pas conservées dans le système d'information de l'association.

Les bénévoles constituent les bureaux des sections dont la composition est précisée sur la fiche de présentation. Un même bénévole peut être membre de plusieurs sections et posséder une fonction différente pour chacune d'entre elles. De plus, chaque section possède un référent unique parmi les bénévoles, un bénévole pouvant être le référent de plusieurs sections.

En début de saison, pour chaque activité qu'il veut pratiquer, l'adhèrent remplit une fiche d'inscription qu'il transmet à l'un des membres du bureau de la section concernée avec le paiement par chèque correspondant à l'activité choisie (cf. annexe 2 et 3). Il reçoit en retour une carte d'adhérent (cf. annexe 4) avec un numéro identique pour toutes les activités ASCG auxquelles il est inscrit. Le chèque est ensuite transmis au trésorier de l'association ; en fin de mois, celui-ci envoie à la banque l'ensemble des chèques perçus par les sections.

Par ailleurs, une remise de 10% est accordée sous forme de remboursement aux membres d'une même famille pratiquant la même activité ; pour cela, le bureau tient à jour la liste des adhérents possédant un lien de parenté et la nature de ce lien (cf. annexe 5). Cette liste est transmise au trésorier de l'association qui procède aux remboursements des adhérents concernés.

Annexes

1 – Fiche de présentation

ASCG Section Musique

Code Section : **MUS** Début Saison : 17/09/2018
Président : LEPRES Jean – 05 06 07 08 09 – jeanlepres@ascg.fr ***(Référent)***
Secrétaire : LASEC Max – 06 67 68 09 10 – maxlasec@ascg.fr
Resp matériel : MATOS Léo – 07 08 09 10 11 – leomatos@ascg.fr

Liste des Activités

N°	Libellé	Description	Jour & Horaire	Durée Séance	Tarif Annuel
1	Guitare	Acoustique, électrique, basse	Lundi - 18h00	45 mn	220 €
2	Batterie	Y compris percussions (cajón)	Mardi - 14h00	30 mn	150 €
3	Piano	Piano à cordes, synthétiseur	Lundi - 20h30	45 mn	240 €
4	Chant	Solfège, technique vocale	Jeudi - 18h00	1h30mn	270 €

Lieux des activités :1 et 2 ➔ Salle Duval (8 personnes) – Allée Campistron
3 et 4 ➔ Atelier Musique (6 personnes) – Rue Terrial

2 – Fiche d'inscription

Fiche d'inscription ASCG

Section : *Musique* **Activité** : *Batterie*
Nom : *Haire* **Prénom** : *Jade* **Date Naissance** : *14/07/1976*
Adresse : *22 rue du Bourg* **CP** : *31150* **Ville** : *Gagnac/Gne*
Tél : *06 08 10 12 14* **Courriel** : *jadehaire@orange.fr*

© Patrick Bergougnoux

3– Chèque

---------- **BANQUE HALLE** ----------

Payez contre ce chèque la somme de *Cent cinquante euros* **€** *150,00*

à *ASCG*

Chèque n° 9876543

Compte 99999 88888 77777777777 / 66

À *Gagnac/Gne*

LE *15/09/2018*

JADE HAIRE
22 RUE DU BOURG
31150 GAGNAC SUR GARONNE

4 – Carte d'adhérent

ASCG
Section Musique

Carte d'Adhérent
Activité : Batterie

Numéro	:	*1234*
Nom	:	*Haire*
Prénom	:	*Jade*
Âge	:	*42 ans*
Adhésion	:	15/09/2018

5 – Liste des liens de parenté

ASCG Section Musique

Activité Batterie – Liens de Parenté

N°Adhérent	Nom	Prénom	Nature Lien
1234	Haire	Jade	Mère-Fille
2345	Haire	Marie	
3456	Terrieur	Alain	Frères
4567	Terrieur	Alex	

Recensement des informations

Modèle organisationnel de flux

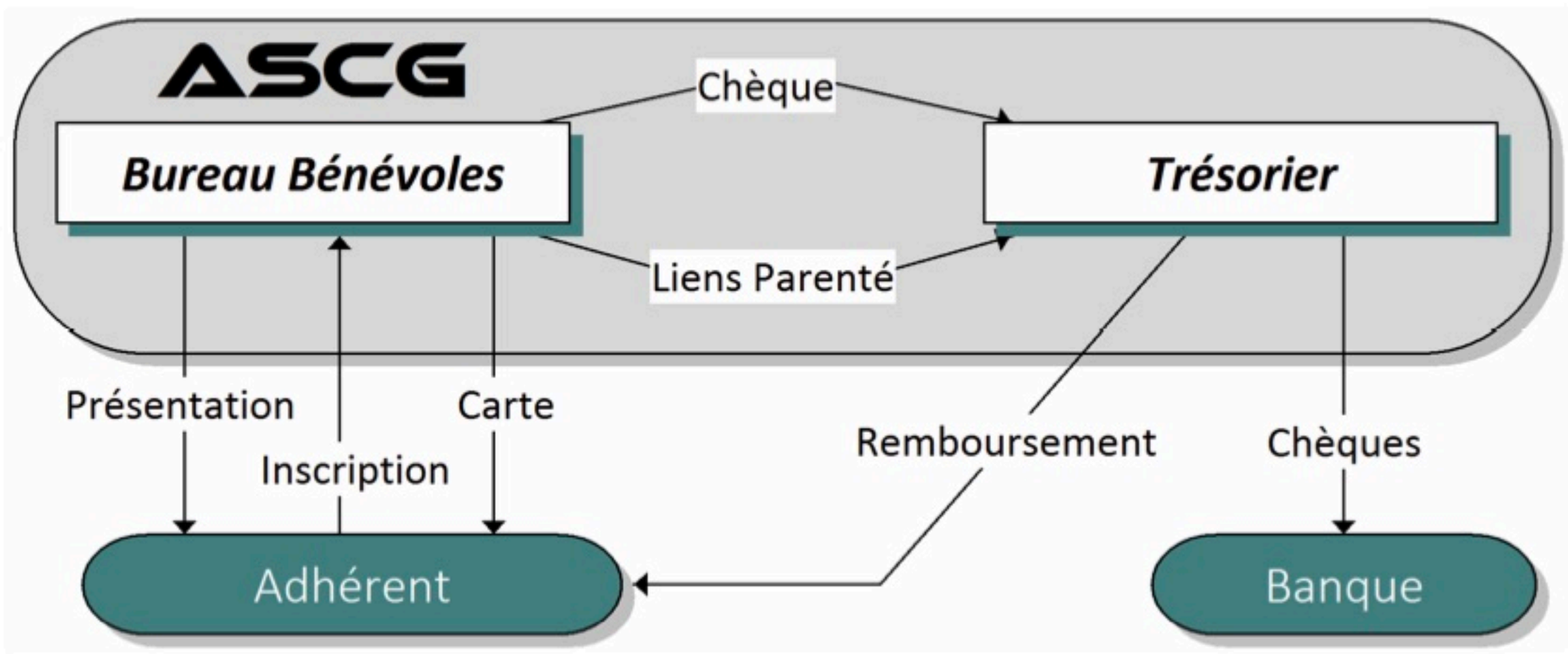


Tableau des rubriques

N°	Libellé	Exemple	P/C	Calcul
1	Nom association	ASCG	P	*Paramètre*
2	Libellé section	Musique	P	
3	Code section	MUS	P	
4	Date début saison section	17/09/2018	P	
5	Fonction bénévole	Président	P	
6	Nom bénévole	LEPRES	P	
7	Prénom bénévole	Jean	P	
8	Téléphone bénévole	05 06 07 08 09	P	
9	Courriel bénévole	jeanlepres@ascg.fr	P	
10	Numéro activité	2	P	
11	Libellé activité	Batterie	P	
12	Description activité	Y compris percussions …	P	
13	Jour semaine activité	Mardi	P	
14	Horaire activité	14h00	P	
15	Durée séance	30 mn	P	
16	Tarif annuel activité	150 €	P	
17	Nom lieu	Salle Duval	P	
18	Capacité lieu	8	P	
19	Adresse lieu	Allée Campistron	P	
20	Nom adhérent	HAIRE	P	
21	Prénom adhérent	Jade	P	
22	Date Naissance adhérent	14/07/1976	P	

© Patrick Bergougnoux

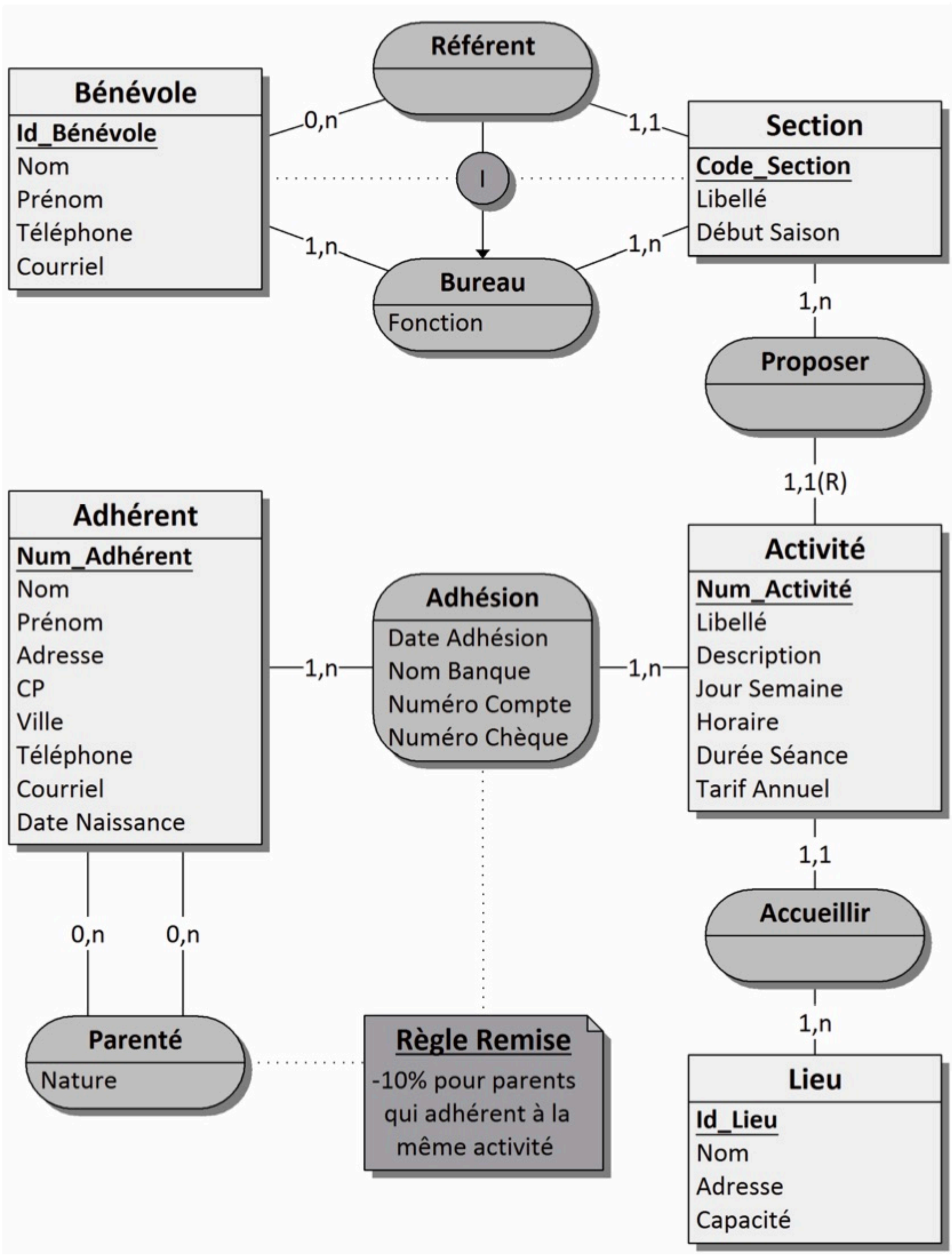
23	Adresse adhérent	22 rue du bourg	P	
24	Code postal adhérent	31150	P	
25	Ville adhérent	Gagnac/Gne	P	
26	Téléphone adhérent	06 08 10 12 14	P	
27	Courriel adhérent	jadehaire@orange.fr	P	
28	Nom banque adhésion	Banque Halle	P	
29	Numéro chèque adhésion	9876543	P	
30	Numéro compte adhésion	99999 88888 77777777…	P	
31	Date adhésion	15/09/2018	P	
32	Numéro adhérent	1234	P	
~~33~~	~~Âge adhérent~~	42	C	[31] – [22]
34	Nature lien parenté	Mère-Fille	P	

Dictionnaire des données

- Libellé section
- Code section
- Date début saison section
- Fonction bénévole
- Nom bénévole
- Prénom bénévole
- Téléphone bénévole
- Courriel bénévole
- Numéro activité
- Libellé activité
- Description activité
- Jour semaine activité
- Horaire activité
- Durée séance
- Tarif annuel activité
- Nom lieu
- Capacité lieu
- Adresse lieu
- Nom adhérent
- Prénom adhérent
- Date Naissance adhérent
- Adresse adhérent
- Code postal adhérent
- Ville adhérent
- Téléphone adhérent
- Courriel adhérent
- Nom banque adhésion
- Numéro chèque adhésion
- Numéro compte adhésion
- Date adhésion
- Numéro adhérent
- Nature lien parenté

Modèle conceptuel de données

Schéma entité-association



© Patrick Bergougnoux

Diagramme de classes UML

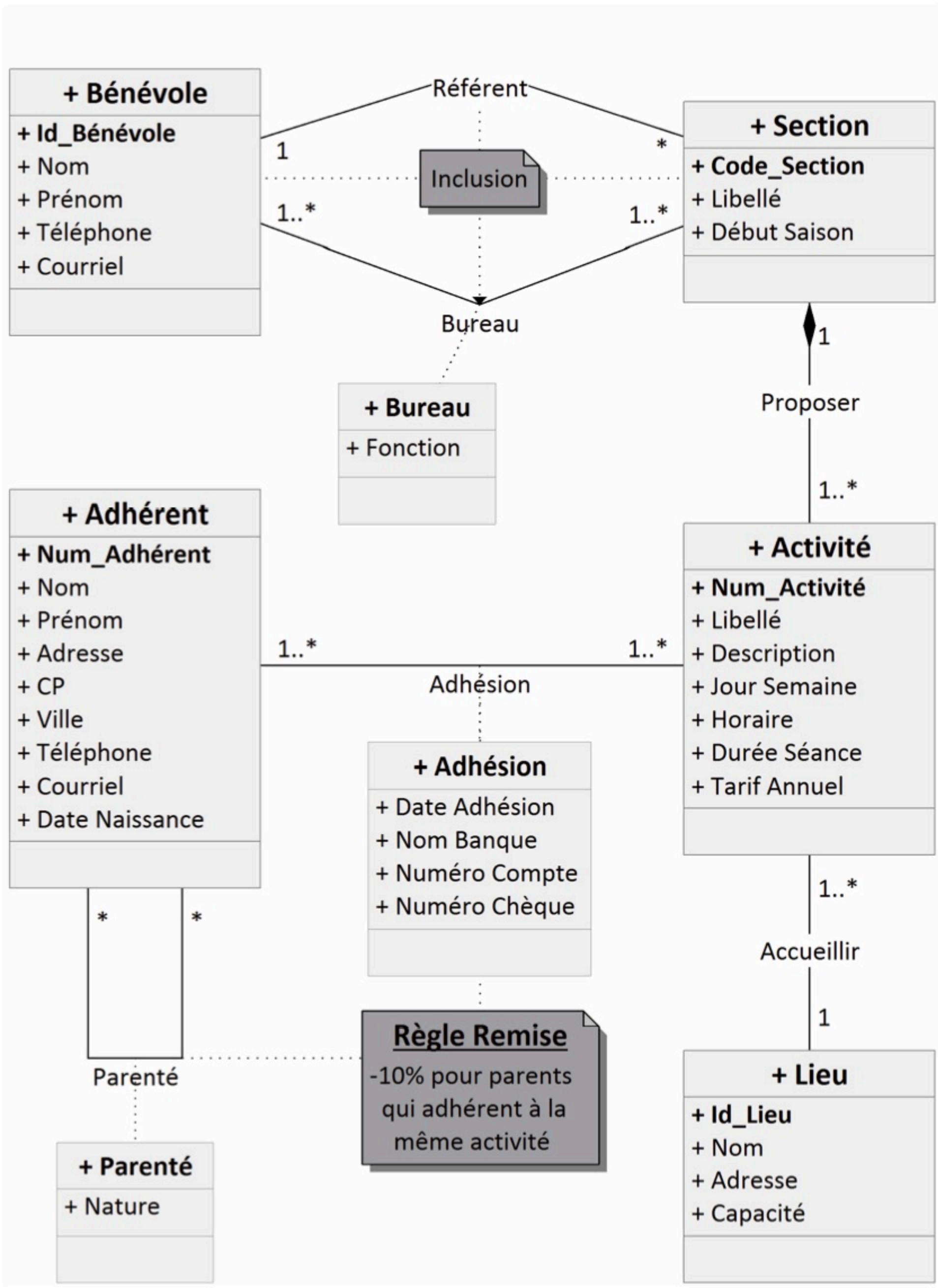


Schéma relationnel

Modèle logique de données

Bénévole = (**Id_Bénévole** *COUNTER*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Téléphone *VARCHAR(20)*, Courriel *VARCHAR(50)*)

Section = (**Code_Section** *CHAR(3)*, Libellé *VARCHAR(50)*, Début_Saison *DATE*, *#Id_Bénévole*)

Lieu = (**Id_Lieu** *COUNTER*, Nom *VARCHAR(50)*, Adresse *VARCHAR(50)*, Capacité *SMALLINT*)

Adhérent = (**Num_Adhérent** *SMALLINT*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Adresse *VARCHAR(50)*, CP *CHAR(5)*, Ville *VARCHAR(50)*, Téléphone *VARCHAR(20)*, Courriel *VARCHAR(50)*, Date_Naissance *DATE*)

Activité = (***#Code_Section*, Num_Activité** *BYTE*, Libellé *VARCHAR(50)*, Description *TEXT*, Jour_Semaine *VARCHAR(8)*, Horaire *TIME*, Durée_Séance *DECIMAL(3,2)*, Tarif_Annuel *CURRENCY*, *#Id_Lieu*)

Bureau = (***#Id_Bénévole, #Code_Section***, Fonction *VARCHAR(50)*)

Adhésion = (***#(#Code_Section, Num_Activité), #Num_Adhérent***, Date_Adhésion *DATE*, Nom_Banque *VARCHAR(50)*, Numéro_Compte *VARCHAR(30)*, Numéro_Chèque *VARCHAR(10)*)

Parenté = (***#Num_Adhérent, #Num_Adhérent_1***, Nature *VARCHAR(50)*)

Requêtes SQL de création des tables

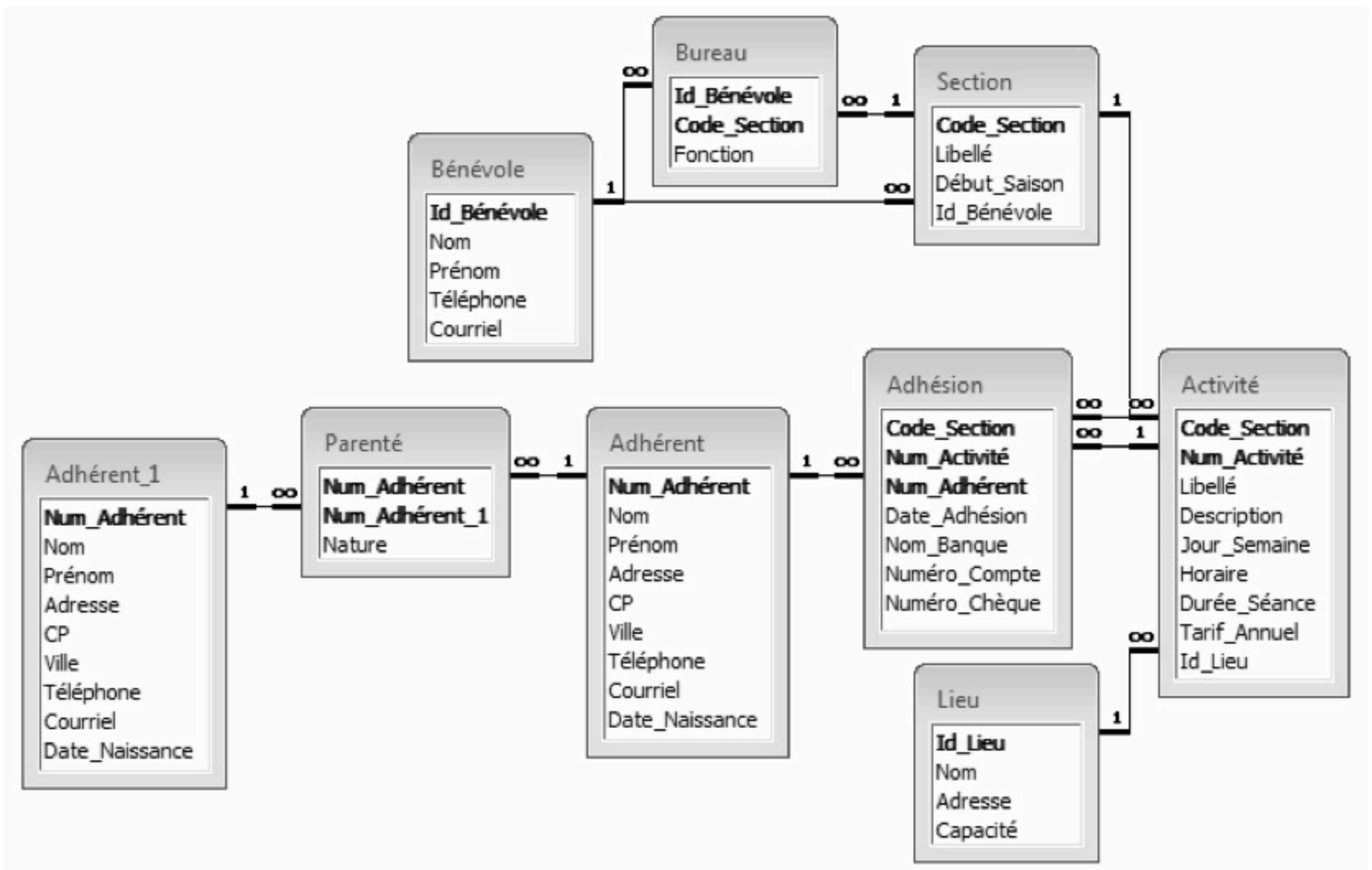
```
CREATE TABLE Bénévole(
        Id_Bénévole COUNTER,
        Nom VARCHAR(50),
        Prénom VARCHAR(50),
        Téléphone VARCHAR(20),
        Courriel VARCHAR(50),
        PRIMARY KEY(Id_Bénévole)
);

CREATE TABLE Section(
        Code_Section CHAR(3),
        Libellé VARCHAR(50),
        Début_Saison DATE,
        Id_Bénévole INT NOT NULL,
        PRIMARY KEY(Code_Section),
        FOREIGN KEY(Id_Bénévole) REFERENCES Bénévole(Id_Bénévole)
);
```


© Patrick Bergougnoux

```
CREATE TABLE Lieu(
        Id_Lieu COUNTER,
        Nom VARCHAR(50),
        Adresse VARCHAR(50),
        Capacité SMALLINT,
        PRIMARY KEY(Id_Lieu)
);

CREATE TABLE Adhérent(
        Num_Adhérent SMALLINT,
        Nom VARCHAR(50),
        Prénom VARCHAR(50),
        Adresse VARCHAR(50),
        CP CHAR(5),
        Ville VARCHAR(50),
        Téléphone VARCHAR(20),
        Courriel VARCHAR(50),
        Date_Naissance DATE,
        PRIMARY KEY(Num_Adhérent)
);

CREATE TABLE Activité(
        Code_Section CHAR(3),
        Num_Activité BYTE,
        Libellé VARCHAR(50),
        Description TEXT,
        Jour_Semaine VARCHAR(8),
        Horaire TIME,
        Durée_Séance DECIMAL(3,2),
        Tarif_Annuel CURRENCY,
        Id_Lieu INT NOT NULL,
        PRIMARY KEY(Code_Section, Num_Activité),
        FOREIGN KEY(Code_Section) REFERENCES Section(Code_Section),
        FOREIGN KEY(Id_Lieu) REFERENCES Lieu(Id_Lieu)
);

CREATE TABLE Bureau(
        Id_Bénévole INT,
        Code_Section CHAR(3),
        Fonction VARCHAR(50),
        PRIMARY KEY(Id_Bénévole, Code_Section),
        FOREIGN KEY(Id_Bénévole) REFERENCES Bénévole(Id_Bénévole),
        FOREIGN KEY(Code_Section) REFERENCES Section(Code_Section)
);

CREATE TABLE Adhésion(
        Code_Section CHAR(3),
        Num_Activité BYTE,
        Num_Adhérent SMALLINT,
        Date_Adhésion DATE,
        Nom_Banque VARCHAR(50),
        Numéro_Compte VARCHAR(30),
        Numéro_Chèque VARCHAR(10),
        PRIMARY KEY(Code_Section, Num_Activité, Num_Adhérent),
        FOREIGN KEY(Code_Section, Num_Activité) REFERENCES
                                        Activité(Code_Section, Num_Activité),
        FOREIGN KEY(Num_Adhérent) REFERENCES Adhérent(Num_Adhérent)
);
```

```
CREATE TABLE Parenté(
        Num_Adhérent SMALLINT,
        Num_Adhérent_1 SMALLINT,
        Nature VARCHAR(50),
        PRIMARY KEY(Num_Adhérent, Num_Adhérent_1),
        FOREIGN KEY(Num_Adhérent) REFERENCES Adhérent(Num_Adhérent),
        FOREIGN KEY(Num_Adhérent_1) REFERENCES Adhérent(Num_Adhérent)
);
```

Base de données Access



Commentaires sur le modèle

- Les associations mises en œuvre dans ce modèle reprennent bon nombre des configurations possibles : cardinalités simples, multiples, identifiant relatif, association réflexive symétrique, double association avec contrainte inter-associations d'inclusion.
- Une règle vient spécifier la remise de 10% accordée aux parents qui adhèrent à une même activité ; cette règle devra faire l'objet d'une programmation spécifique dans le système d'information.
- L'historisation des adhésions sur plusieurs années n'était pas demandée et n'est donc pas prise en compte dans ce modèle.

© Patrick Bergougnoux

Chapitre 18
Compagnie d'assurances

CAO, Compagnie d'Assurances d'Occitanie, souhaite automatiser la gestion des contrats de ses clients et des sinistres dans lesquels leurs véhicules sont impliqués.

Compte-rendu d'interview

Description du système d'information

Pour demander la prise en charge de son véhicule, le client prend contact avec un agent de la compagnie CAO qui lui demande de remplir un questionnaire (cf. annexe 1). L'agent reprend ces informations pour établir une offre en définissant une référence de contrat, ainsi que le montant de la prime d'assurance et des différentes franchises (cf. annexe 2).

Pour effectuer cette offre, l'agent s'appuie sur les garanties proposées par la compagnie. Le total de base de la prime d'assurance est alors calculé en additionnant le tarif de la prime de chaque garantie prise dans le cadre du contrat, tarif qui varie en fonction du modèle du véhicule. Ce total est ensuite modulé par un taux de bonus-malus propre à chaque véhicule.

Le montant de chaque franchise dépend également de la garantie contractée et du modèle du véhicule.

Chaque véhicule bénéficie d'un et un seul contrat, un contrat ne concernant qu'un seul véhicule pour lequel sont précisées les informations de l'ensemble des conducteurs désignés, dont un seul est le propriétaire ; ces informations ont été fournies dans le questionnaire initial rempli par le propriétaire (cf. annexe 1).

Un conducteur peut conduire, et éventuellement posséder, plusieurs véhicules assurés par la compagnie. La nature des liens de parenté entre les conducteurs est également précisée verbalement à l'agent. Le véhicule est couvert par l'assurance dès réception par l'agent du contrat signé ; l'agent transmet alors une copie de ce contrat au siège social de la compagnie.

Dès qu'un sinistre se produit sur un véhicule, l'agent crée un dossier numéroté au sein du contrat comprenant la nature du sinistre, sa date, son lieu ainsi qu'une description détaillée (cf. annexe 3). Il transmet ensuite ce dossier au siège social qui prend en charge le sinistre. Après étude du dossier, celui-ci est complété par le coût des réparations et par l'établissement d'un coefficient de responsabilité qui peut éventuellement entrainer une modification du taux de bonus-malus du véhicule. Une fiche récapitulative est alors envoyée au propriétaire du véhicule (cf. annexe 4).

Annexes

1 – Questionnaire client

Compagnie d'Assurances d'Occitanie

Véhicule

N° : *AB-123-EX* **Date 1ère immat :** *07/07/2014*

Modèle : *Clio 3* **Type** : *VP* **Marque :** *Renault*

Année : *2014* **Puissance :** *6 CV* **Carburant :** *Essence*

Propriétaire *(et conducteur principal)*

Nom : *Hépeur* **Prénom :** *Jean* **Date Naissance :** *24/09/1985*

Adresse : *4 rue du Pont* **CP :** *31200* **Ville :** *Toulouse*

Tél : *06 98 76 54 32* **Courriel :** *jeanhepeur@gmail.com*

Garanties *(cochez les cases correspondant aux garanties demandées)*

☑ Responsabilité civile ☑ Dommage collision ☑ Incendie et vol ☑ Bris de glace

☐ Catastrophe naturelle ☑ Panne mécanique ☐ Tempête ☐ Attentat

Autres conducteurs *(nom, prénom, adresse, date naissance, téléphone et courriel)*

- Durand Jane – 4 rue du Pont, 31200 Toulouse – née le 02/03/1986
Tél. 05 04 03 02 01 – Mail : janedurand@gmail.com

- Débois Robin – 1 av des pins, 11100 Narbonne – né le 11/11/1991
Tél. 06 11 12 13 14 – Mail : robindebois@hotmail.fr

© Patrick Bergougnoux

2 – Contrat

CAO

Compagnie d'Assurances d'Occitanie

CONTRAT

AGENT : Etb JASSUR
123 rue sans soucis – 31000 Toulouse
Tél. 05 15 25 35 45 – jassur@cao.fr

Référence VEH314253
Date Contrat 22/04/2019
Immatriculation AB-123-EX

HEPEUR Jean
4 rue du pont
31200 Toulouse

Garantie	Code	Prime	Franchise
Responsabilité civile	RC	178,00 €	0,00 €
Dommage collision	DC	95,00 €	300,00 €
Incendie et vol	IV	32,00 €	500,00 €
Bris de glace	BG	17,20 €	50,00 €
Panne mécanique	PM	43,40 €	75,00 €
Total de base de la prime d'assurance		**365,60 €**	
Montant Net avec Bonus de 40%		**219,36 €**	

Signature client

3 – Dossier sinistre

DECLARATION SINISTRE

AGENT CAO : Etb JASSUR
Réf. Contrat : VEH314253
N° Sinistre : 2

Date Sinistre 15/07/2019
Nature Collision avec tiers
Lieu Toulouse Centre

Description Sinistre
La voiture qui roulait devant a soudainement stoppé ; l'assuré n'a pas pu s'arrêter à temps et a alors percuté l'arrière du véhicule.

4 – Fiche récapitulative suite à un sinistre

RECAPITULATIF SINISTRE

AGENT CAO : Etb JASSUR
Réf. Contrat : VEH314253
N° Sinistre : 2

Date Sinistre 15/07/2019
Nature Collision avec tiers
Lieu Toulouse Centre

Coût des réparations : **2 543,45 €uros**
Coefficient responsabilité : **100%** ➔ Nouveau bonus : **30%**

Recensement des informations

Modèle organisationnel de flux

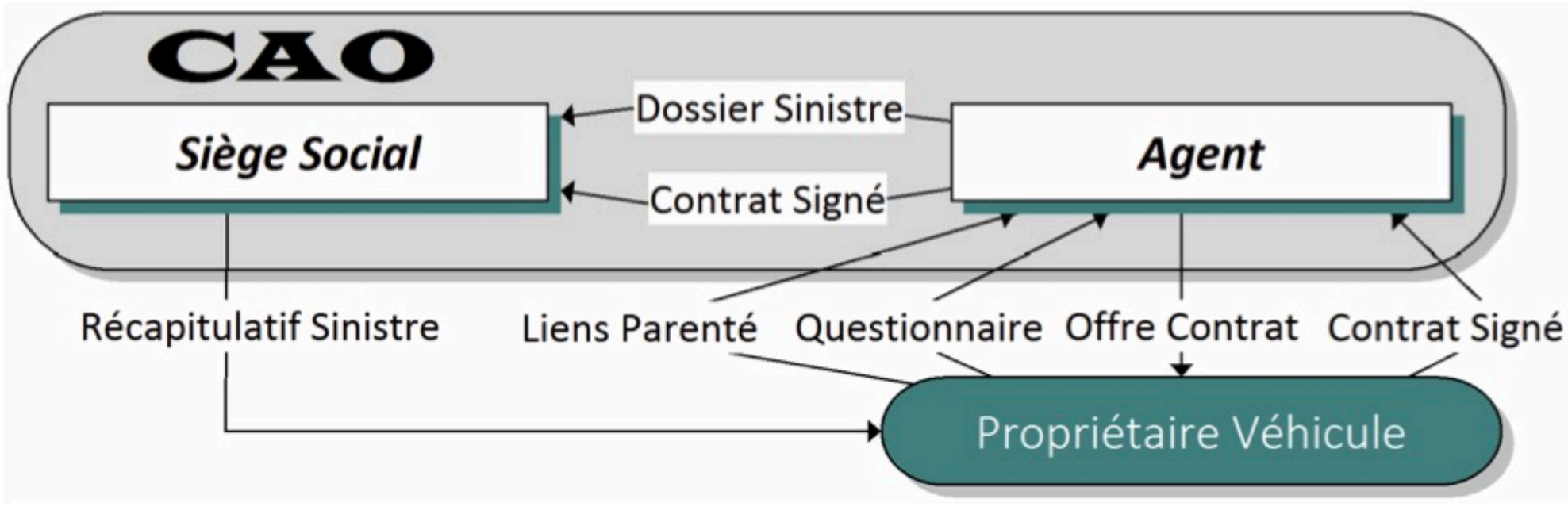


Tableau des rubriques

N°	Libellé	Exemple	P/C	Calcul
~~1~~	~~Nom compagnie~~	CAO	P	*Paramètre*
2	N° immatriculation véhicule	AB-123-EX	P	
3	Date 1ère immatriculation	07/07/2014	P	
4	Désignation modèle	Clio 3	P	
5	Type modèle	VP	P	
6	Marque modèle	Renault	P	
7	Année fabrication modèle	2014	P	
8	Puissance fiscale modèle	6 CV	P	
9	Carburant modèle	Essence	P	
10	Nom conducteur	Hépeur	P	
11	Prénom conducteur	Jean	P	
12	Date naissance conducteur	24/09/1985	P	
13	Adresse conducteur	4 rue du Pont	P	
14	Code postal conducteur	31200	P	
15	Ville conducteur	Toulouse	P	
16	Téléphone conducteur	06 98 76 54 32	P	
17	Courriel conducteur	jeanhepeur@gmail.com	P	
18	Libellé garantie	Dommage collision	P	
19	Nom agent	Etb Jassur	P	
20	Adresse agent	123 rue sans soucis	P	
21	Code postal agent	31000	P	
22	Ville agent	Toulouse	P	
23	Téléphone agent	05 15 25 35 45	P	
24	Courriel agent	jassur@cao.fr	P	
25	Référence contrat	VEH314253	P	
26	Date contrat	22/04/2019	P	
27	Code garantie	DC	P	
28	Tarif prime garantie	95,00 €	P	

© Patrick Bergougnoux

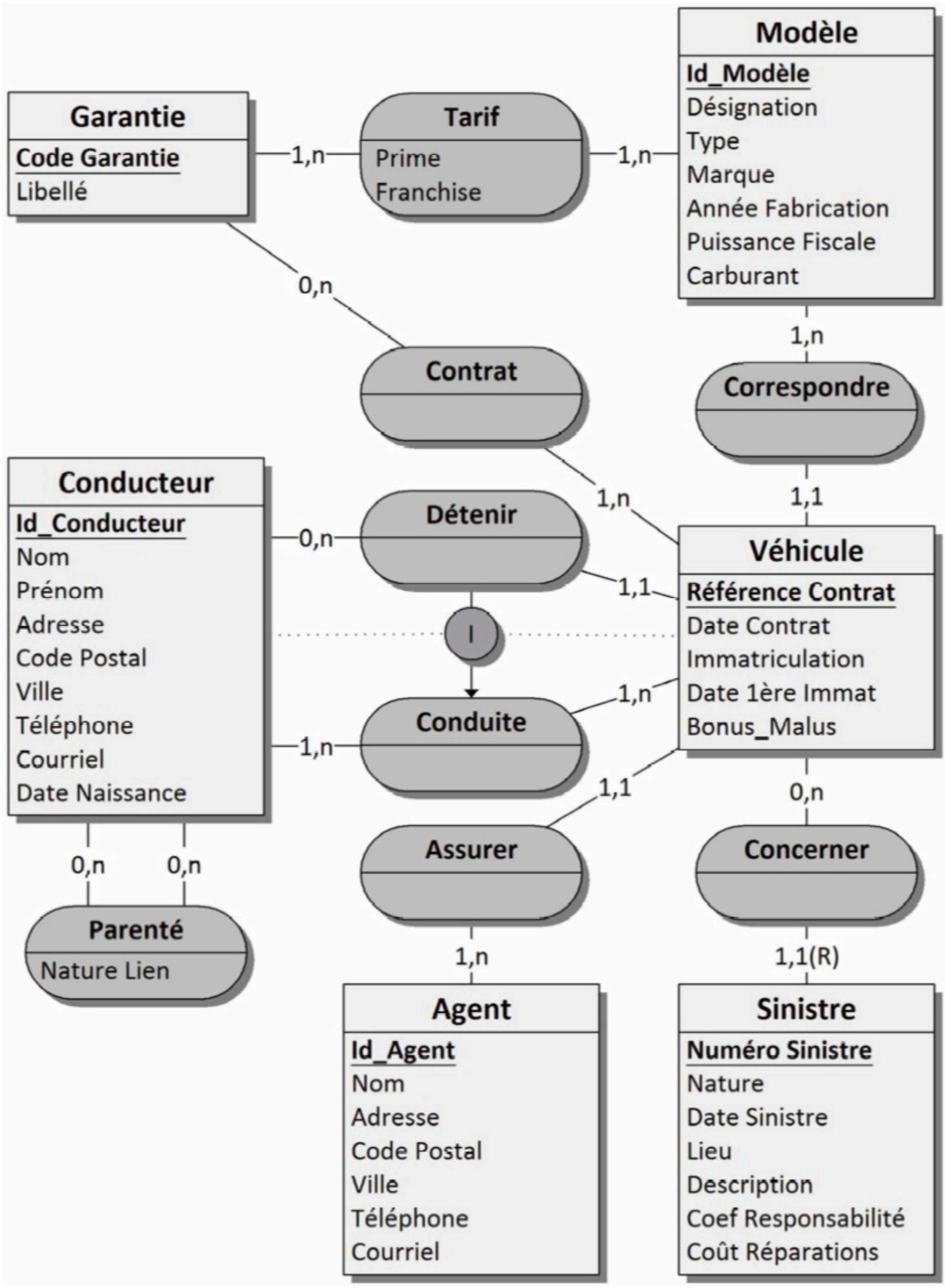
29	Montant franchise garantie	300 €	P	
~~30~~	~~Total de base prime~~	365,60 €	C	Σ [28]
31	Taux bonus-malus	40%	P	
~~32~~	~~Montant net prime~~	219,36 €	C	[30] x (1-[31])
33	Numéro sinistre	2	P	
34	Date sinistre	15/07/2019	P	
35	Nature sinistre	Collision avec tiers	P	
36	Lieu sinistre	Toulouse Centre	P	
37	Description sinistre	La voiture qui roulait …	P	
38	Coût réparations	2 543,45 €	P	
39	Coefficient responsabilité	100%	P	
40	Nature lien parenté	Cousins	P	

Dictionnaire des données

- N° immatriculation véhicule
- Date 1ère immatriculation
- Désignation modèle
- Type modèle
- Marque modèle
- Année fabrication modèle
- Puissance fiscale modèle
- Carburant modèle
- Nom conducteur
- Prénom conducteur
- Date naissance conducteur
- Adresse conducteur
- Code postal conducteur
- Ville conducteur
- Téléphone conducteur
- Courriel conducteur
- Libellé garantie
- Nom agent
- Adresse agent
- Code postal agent
- Ville agent
- Téléphone agent
- Courriel agent
- Référence contrat
- Date contrat
- Code garantie
- Tarif prime garantie
- Montant franchise garantie
- Taux bonus-malus
- Numéro sinistre
- Date sinistre
- Nature sinistre
- Lieu sinistre
- Description sinistre
- Coût réparations
- Coefficient responsabilité
- Nature lien parenté

Modèle conceptuel de données

Schéma entité-association



© Patrick Bergougnoux

Diagramme de classes UML

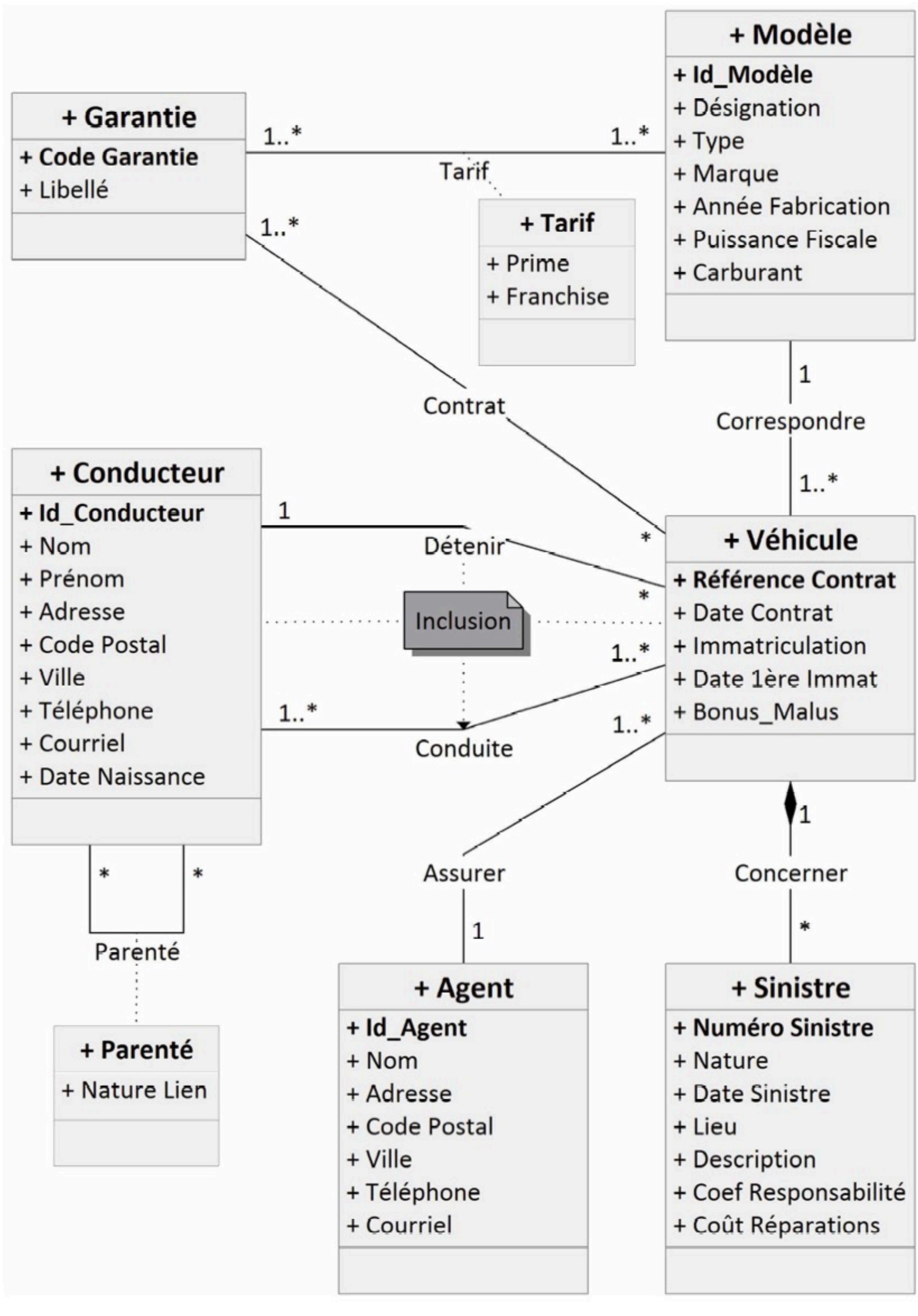


Schéma relationnel

Modèle logique de données

Garantie = (**Code_Garantie** *CHAR(2)*, Libellé *VARCHAR(50)*)

Conducteur = (**Id_Conducteur** *COUNTER*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Adresse *VARCHAR(50)*, Code_Postal *CHAR(5)*, Ville *VARCHAR(50)*, Téléphone *VARCHAR(20)*, Courriel *VARCHAR(50)*, Date_Naissance *DATE*)

Agent = (**Id_Agent** *COUNTER*, Nom *VARCHAR(50)*, Adresse *VARCHAR(50)*, Code_Postal *CHAR(5)*, Ville *VARCHAR(50)*, Téléphone *VARCHAR(20)*, Courriel *VARCHAR(50)*)

Modèle = (**Id_Modèle** *COUNTER*, Désignation *VARCHAR(50)*, Type *VARCHAR(50)*, Marque *VARCHAR(50)*, Année_Fabrication *SMALLINT*, Puissance_Fiscale *BYTE*, Carburant *VARCHAR(10)*)

Véhicule = (**Référence_Contrat** *CHAR(9)*, Date_Contrat *DATE*, Immatriculation *VARCHAR(10)*, Date_1ère_Immat *DATE*, Bonus_Malus *SMALLINT*, *#Id_Agent, #Id_Conducteur, #Id_Modèle*)

Sinistre = (***#Référence_Contrat*, Numéro_Sinistre** *BYTE*, Nature *VARCHAR(50)*, Date_Sinistre *DATE*, Lieu *VARCHAR(50)*, Description *TEXT*, Coef_Responsabilité *BYTE*, Coût_Réparations *CURRENCY*)

Tarif = (***#Code_Garantie, #Id_Modèle***, Prime *CURRENCY*, Franchise *CURRENCY*)

Contrat = (***#Référence_Contrat, #Code_Garantie***)

Conduite = (***#Id_Conducteur, #Référence_Contrat***)

Parenté = (***#Id_Conducteur, #Id_Conducteur_1***, Nature_Lien *VARCHAR(30)*)

Requêtes SQL de création des tables

```
CREATE TABLE Garantie(
        Code_Garantie CHAR(2),
        Libellé VARCHAR(50),
        PRIMARY KEY(Code_Garantie)
);
```

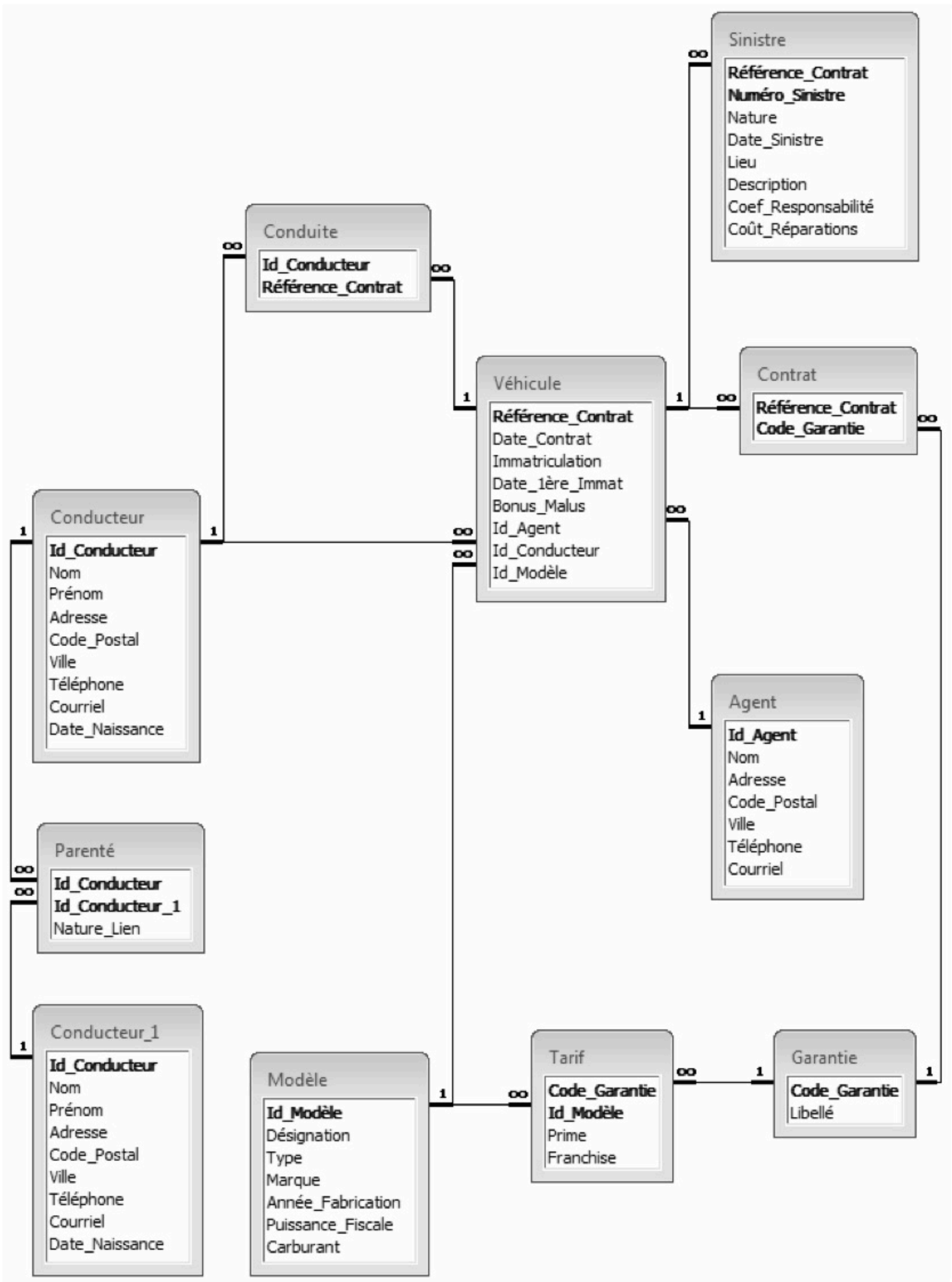

© Patrick Bergougnoux

```
CREATE TABLE Conducteur(
	Id_Conducteur COUNTER,
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	Adresse VARCHAR(50),
	Code_Postal CHAR(5),
	Ville VARCHAR(50),
	Téléphone VARCHAR(20),
	Courriel VARCHAR(50),
	Date_Naissance DATE,
	PRIMARY KEY(Id_Conducteur)
);

CREATE TABLE Agent(
	Id_Agent COUNTER,
	Nom VARCHAR(50),
	Adresse VARCHAR(50),
	Code_Postal CHAR(5),
	Ville VARCHAR(50),
	Téléphone VARCHAR(20),
	Courriel VARCHAR(50),
	PRIMARY KEY(Id_Agent)
);

CREATE TABLE Modèle(
	Id_Modèle COUNTER,
	Désignation VARCHAR(50),
	Type VARCHAR(50),
	Marque VARCHAR(50),
	Année_Fabrication SMALLINT,
	Puissance_Fiscale BYTE,
	Carburant VARCHAR(10),
	PRIMARY KEY(Id_Modèle)
);

CREATE TABLE Véhicule(
	Référence_Contrat CHAR(9),
	Date_Contrat DATE,
	Immatriculation VARCHAR(10),
	Date_1ère_Immat DATE,
	Bonus_Malus SMALLINT,
	Id_Agent INT NOT NULL,
	Id_Conducteur INT NOT NULL,
	Id_Modèle INT NOT NULL,
	PRIMARY KEY(Référence_Contrat),
	FOREIGN KEY(Id_Agent) REFERENCES Agent(Id_Agent),
	FOREIGN KEY(Id_Conducteur) REFERENCES Conducteur(Id_Conducteur),
	FOREIGN KEY(Id_Modèle) REFERENCES Modèle(Id_Modèle)
);
```

```
CREATE TABLE Sinistre(
	Référence_Contrat CHAR(9),
	Numéro_Sinistre BYTE,
	Nature VARCHAR(50),
	Date_Sinistre DATE,
	Lieu VARCHAR(50),
	Description TEXT,
	Coef_Responsabilité BYTE,
	Coût_Réparations CURRENCY,
	PRIMARY KEY(Référence_Contrat, Numéro_Sinistre),
	FOREIGN KEY(Référence_Contrat) REFERENCES Véhicule(Référence_Contrat)
);

CREATE TABLE Tarif(
	Code_Garantie CHAR(2),
	Id_Modèle INT,
	Prime CURRENCY,
	Franchise CURRENCY,
	PRIMARY KEY(Code_Garantie, Id_Modèle),
	FOREIGN KEY(Code_Garantie) REFERENCES Garantie(Code_Garantie),
	FOREIGN KEY(Id_Modèle) REFERENCES Modèle(Id_Modèle)
);

CREATE TABLE Contrat(
	Référence_Contrat CHAR(9),
	Code_Garantie CHAR(2),
	PRIMARY KEY(Référence_Contrat, Code_Garantie),
	FOREIGN KEY(Référence_Contrat) REFERENCES Véhicule(Référence_Contrat),
	FOREIGN KEY(Code_Garantie) REFERENCES Garantie(Code_Garantie)
);

CREATE TABLE Conduite(
	Id_Conducteur INT,
	Référence_Contrat CHAR(9),
	PRIMARY KEY(Id_Conducteur, Référence_Contrat),
	FOREIGN KEY(Id_Conducteur) REFERENCES Conducteur(Id_Conducteur),
	FOREIGN KEY(Référence_Contrat) REFERENCES Véhicule(Référence_Contrat)
);

CREATE TABLE Parenté(
	Id_Conducteur INT,
	Id_Conducteur_1 INT,
	Nature_Lien VARCHAR(30),
	PRIMARY KEY(Id_Conducteur, Id_Conducteur_1),
	FOREIGN KEY(Id_Conducteur) REFERENCES Conducteur(Id_Conducteur),
	FOREIGN KEY(Id_Conducteur_1) REFERENCES Conducteur(Id_Conducteur)
);
```


© Patrick Bergougnoux

Base de données Access



Commentaires sur le modèle

- Cet exercice est intéressant par rapport à l'application de la règle 3 : en effet, il n'était pas forcément évident, à la première lecture des documents, de distinguer les classes d'entités « Véhicule » et « Modèle », ce qui est pourtant nécessaire compte-tenu du nombre de rubriques communes à un modèle de véhicule et de l'association des conditions de garantie (prime, franchise) à un modèle donné.

- Toujours concernant la règle 3, certaines adresses communes à plusieurs conducteurs peuvent suggérer de créer une classe à part ; mais, dans ce cas, le coté exceptionnel de cette situation ne justifie pas l'alourdissement de la structure de la base de données.

- La règle 5 a également été appliquée : au départ, il est normal de distinguer les notions de contrat et de véhicule ; cependant, le compte-rendu d'interview précise bien que chaque véhicule bénéficie d'un et un seul contrat, un contrat ne concernant qu'un seul véhicule. Il y aurait alors eu une association **1,1/1,1** entre ces deux classes, d'où leur regroupement.

- Le compte-rendu d'interview indique que la nature des liens de parenté entre les conducteurs est précisée verbalement à l'agent : cette information ne fait donc pas l'objet d'un document en annexe, mais ça n'en reste pas moins une information puisqu'elle est échangée au sein du système d'information, le moyen utilisé pour l'échange n'ayant pas d'importance.

- Concernant l'historisation des informations, si une rubrique évolue (coordonnées des agents ou des conducteurs, montant des primes ou des franchises, taux de bonus-malus, …), les nouvelles valeurs doivent prendre la place des anciennes sans qu'il soit nécessaire d'en conserver l'historique ; le modèle proposé correspond bien à ce mode de fonctionnement.

- On peut remarquer que, lors de la déclaration du sinistre, le conducteur n'est pas mentionné : le modèle conceptuel ne prend donc pas en compte cette information et l'interlocuteur sera forcément le propriétaire du véhicule. Une évolution du modèle pourrait consister à rajouter une association **1,1/0,n** entre les classes d'entités « Sinistre » et « Conducteur » afin de définir qui conduisait au moment du sinistre.

© Patrick Bergougnoux

Chapitre 19
Entreprise générale du bâtiment

EGB, Entreprise Générale du Bâtiment, travaille pour de nombreux clients et réalise différentes natures des travaux. Elle souhaite automatiser son système d'information afin de gérer au mieux les interventions de ses salariés pour les travaux réalisés pour ses clients.

Compte-rendu d'interview

Description du système d'information

L'entreprise intervient suite à une demande d'un client qui contacte directement le directeur d'EGB, le plus souvent par courriel (cf. annexe 1).

Le directeur affecte alors un code et une description à l'intervention demandée et définit une date de début de travaux. Ces informations sont communiquées au client et à un des salariés d'EGB désigné comme chef de chantier pour l'intervention en question (cf. annexe 2).

Une intervention concerne une nature de travaux précise (maçonnerie, carrelage, peinture, plomberie, électricité, …), et est effectuée par un ou plusieurs salariés, dont le chef de chantier. À la fin de chaque journée, celui-ci transmet au directeur une fiche indiquant les heures passées par chaque salarié pour l'intervention en question (cf. annexe 3).

Le cumul des heures passées dans le cadre d'une intervention est reporté par le directeur sur un état récapitulatif (cf. annexe 4) qu'il transmet ensuite au service comptable lorsque l'intervention est terminée.

De plus, pour une intervention donnée, EGB peut utiliser différents matériaux ; ceux-ci sont caractérisés par une référence (interne au fournisseur), une désignation, une unité de mesure et un coût unitaire, un matériau donné étant toujours acheté au même fournisseur.

À la réception d'une facture fournisseur (cf. annexe 5), le service comptable donne une copie de cette facture au directeur qui établit alors la liste des matériaux utilisés pour chaque intervention (cf. annexe 6) ; cette liste est transmise au service comptable.

Ce dernier établit en fin de mois les factures des clients pour chaque intervention terminée ; pour cela, il applique une marge propre à chaque matériau utilisé, et calcule le prix de la main d'œuvre à partir de l'état récapitulatif mentionnant le nombre total d'heures passées et le taux horaire propre à chaque nature de travaux (cf. annexe 7).

Par ailleurs, tout salarié doit régulièrement passer une visite médicale afin d'être jugé apte ou non pour chaque nature de travaux. Chaque attestation d'aptitude (cf. annexe 8), composée d'un commentaire et d'une date de fin de validité, est délivrée par un médecin de travail, et concerne donc un salarié donné pour une nature de travaux donnée ; elle est transmise au directeur après chaque visite.

Annexes

1 – Demande client

De: justin.ptipeu@free.fr
Date: Lundi 15 Avril 2019 10:13
À: direction@egb.fr
Sujet: Demande d'intervention

Bonjour,

Merci de prévoir une intervention concernant la peinture des murs et plafonds de mon bureau si possible avant la fin du mois d'avril.

Vous en remerciant par avance,

Ptipeu Justin
10 rue quelconque - 31000 Toulouse
Tél. 06 07 08 09 10

© Patrick Bergougnoux

2 – Informations intervention

De: direction@egb.fr
Date: Lundi 15 Avril 2019 15:24
À: justin.ptipeu@free.fr
Sujet: Re: Demande d'intervention

Bonjour,

Votre demande d'intervention concernant « Peinture murs et plafonds bureau » a été enregistrée sous le code 19040042 et débutera le 29 avril 2019.

Le taux horaire des travaux est de 28 euros HT (Nature des travaux : Peinture).

Nous restons à votre disposition pour toute information complémentaire.

Bien cordialement,

EGB, Entreprise Générale de Bâtiment
23 allée des usines
31700 Beauzelle
Tél. 05 11 12 13 14

3 – Fiche journalière de travaux

Fiche journalière EGB

Code intervention : *19040042* **Jour travaux :** *06/05/2019*

Nom et Prénom Salarié	Matricule	Nb heures
EDUN Michel	1234	5
PUIDEU Jean-Jacques	2345	2
ETTROIT Patrick	3456	8
	Total	**15**

4 – État récapitulatif des interventions

Récapitulatif intervention

Code : 19040042
Nature Travaux : Peinture
Taux horaire : 28 euros HT
Date début : 29/04/2019

Jour travaux	Nb heures
29/04/2019	16
30/04/2019	12
06/05/2019	15
07/05/2019	16
08/05/2019	3
Total	**62**

5 – Facture fournisseur

SAS MATOS
22 rue des bricolos – 31150 Fenouillet
Tél. 05 61 62 63 64 – matos@gmail.com

EGB
23 allée des usines
31700 Beauzelle

N°Facture : 2019-000123
Date : 18/04/2019

Réf	Désignation	Qté	Unité	PU HT	Montant HT
PE-01	Sous-couche	24	litre	3,82 €	91,68 €
PE-04	Peinture bleue	50	litre	12,90 €	645,00 €
AC-12	Rouleau	5	un	17,50 €	87,50 €

Total HT	824,18 €
TVA 20%	164,84 €
Total TTC	**989,02 €**

SAS au capital de 50 000 € – SIRET 123 456 789 00099

© Patrick Bergougnoux

6 – Liste des matériaux par intervention

Liste Matériaux pour intervention : *19040042*

Fournisseur	N°Facture	Réf	Qté utilisée
SAS MATOS	2019-000123	PE-01	20
SAS MATOS	2019-000134	PE-04	30
Sté QUINCAILLE	FA19030045	ABCD15C	2

7 – Facture client

EGB

23 allée des usines
31700 Beauzelle
Tél. 05 11 12 13 14
Courriel : contact@egb.fr

FACTURE

Numéro	F19050089
Date	31/05/2019
Intervention	19040042
Nature	Peinture

PTIPEU Justin
10 rue quelconque
31000 Toulouse

Peinture murs et plafonds bureau

Désignation	Qté	Unité	Prix Unitaire	Montant HT
Sous-couche	20	litre	4,97	99,40
Peinture bleue	30	litre	16,13	483,90
Bandes masquage	50	mètre	0,38	19,00
Main d'œuvre	62	heure	28,00	1 736,00
			Total HT	2 338,30 €
			TVA 20%	467,66 €
			TOTAL TTC	**2 805,96 €**

8 – Attestation d'aptitude

Dr AUSCULTÉ Géo
33 avenue Diteleux – 31200 Toulouse
N°RPPS : 10100123456

Société EGB
23 allée des usines
31700 Beauzelle

Attestation Aptitude

Je soussigné, Docteur Géo Ausculté, atteste par la présente que Monsieur Michel EDUN est apte à assurer les travaux de peinture.

Commentaire : respecter une pause de 5 mn minimum toutes les 2 heures.

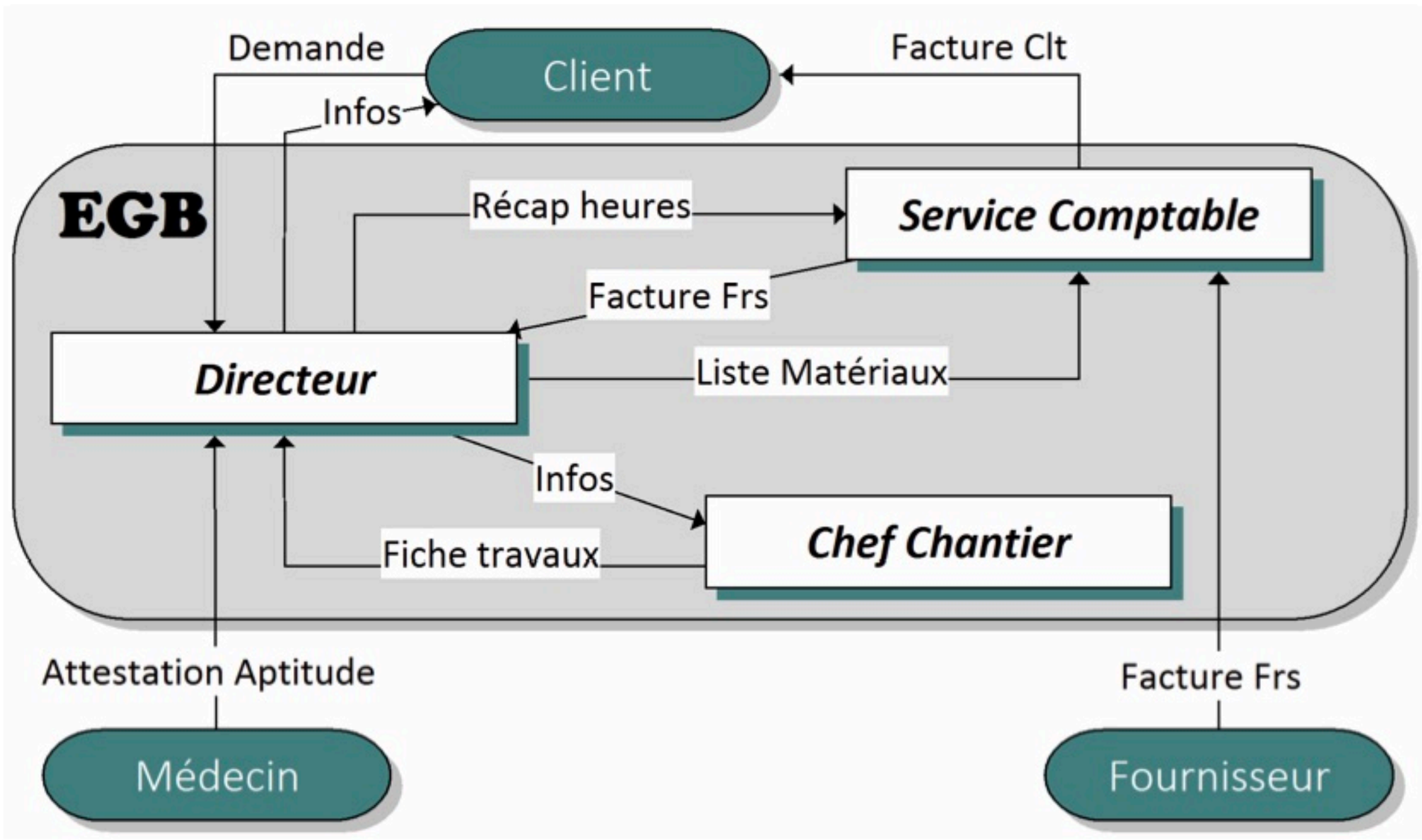
Date de fin de validité : 15/09/2021

Attestation pour faire valoir ce que de droit,

A Toulouse, le 15 septembre 2018

Recensement des informations

Modèle organisationnel de flux



© Patrick Bergougnoux

Tableau des rubriques

N°	Libellé	Exemple	P/C	Calcul
~~1~~	~~Nom entreprise~~	EGB	P	*Paramètre*
~~2~~	~~Coordonnées entreprise~~	23 allée des usines, …	P	*Paramètre*
3	Nom client	PTIPEU	P	
4	Prénom client	Justin	P	
5	Adresse client	10 rue quelconque	P	
6	Code postal client	31000	P	
7	Ville client	Toulouse	P	
8	Téléphone client	06 67 08 09 10	P	
9	Courriel client	justin.ptipeu@free.fr	P	
10	Description intervention	Peinture murs et plafonds…	P	
11	Code intervention	19040042	P	
12	Date début intervention	29/04/2019	P	
13	Libellé nature travaux	Peinture	P	
14	Taux horaire nature travaux	28 euros HT	P	
15	Jour travaux	06/05/2019	P	
16	Nom salarié	EDUN	P	
17	Prénom salarié	Michel	P	
18	Matricule salarié	1234	P	
19	Nb heures/jour/salarié	5	P	
~~20~~	~~Nb heures/jour/intervention~~	15	C	Σ [19]
~~21~~	~~Total heures/intervention~~	62	C	Σ [20]
22	Nom fournisseur	SAS MATOS	P	
23	Adresse fournisseur	22 rue des bricolos	P	
24	Code postal fournisseur	31150	P	
25	Ville fournisseur	Fenouillet	P	
26	Téléphone fournisseur	05 61 62 63 64	P	
27	Courriel fournisseur	matos@gmail.com	P	
28	Numéro facture achat	2019-000123	P	
29	Date facture achat	18/04/2019	P	

30	Référence matériau	PE-01	P	
31	Désignation matériau	Sous-couche	P	
32	Quantité matériau acheté	24	P	
33	Unité mesure matériau	litre	P	
34	PU HT achat matériau	3,82 €	P	
~~35~~	~~Montant HT ligne achat~~	91,68 €	C	[34] x [32]
~~36~~	~~Total HT facture achat~~	824,18 €	C	Σ [35]
~~37~~	~~Taux TVA~~	20%	P	*Paramètre*
~~38~~	~~Montant TVA facture achat~~	164,84 €	C	[36] x [37]
~~39~~	~~Montant TTC facture achat~~	989,02 €	C	[36] + [38]
40	Forme juridique fournisseur	SAS au capital de 50 000 €	P	
41	Numéro SIRET fournisseur	123 456 789 00099	P	
42	Qté matériaux/intervention	20	P	
43	Numéro facture client	F19050089	P	
44	Date facture client	31/05/2019	P	
~~45~~	~~PU HT vente matériau~~	4,97 €	C	[34] x (1 + [59])
~~46~~	~~Montant HT ligne vente~~	99,40 €	C	[45] x [42]
~~47~~	~~Total HT facture vente~~	2 338,30 €	C	Σ [46]
~~48~~	~~Montant TVA facture vente~~	467,66 €	C	[47] x [37]
~~49~~	~~Montant TTC facture vente~~	2 805,96 €	C	[47] + [48]
50	Nom médecin	Ausculté	P	
51	Prénom médecin	Géo	P	
52	Adresse médecin	33 avenue Diteleux	P	
53	Code postal médecin	31200	P	
54	Ville médecin	Toulouse	P	
55	Numéro RPPS médecin	10100123456	P	
56	Commentaire aptitude	respecter une pause de 5…	P	
57	Date fin validité aptitude	15/09/2021	P	
58	Date attestation aptitude	15/08/2018	P	
59	Marge vente matériau	30%	P	

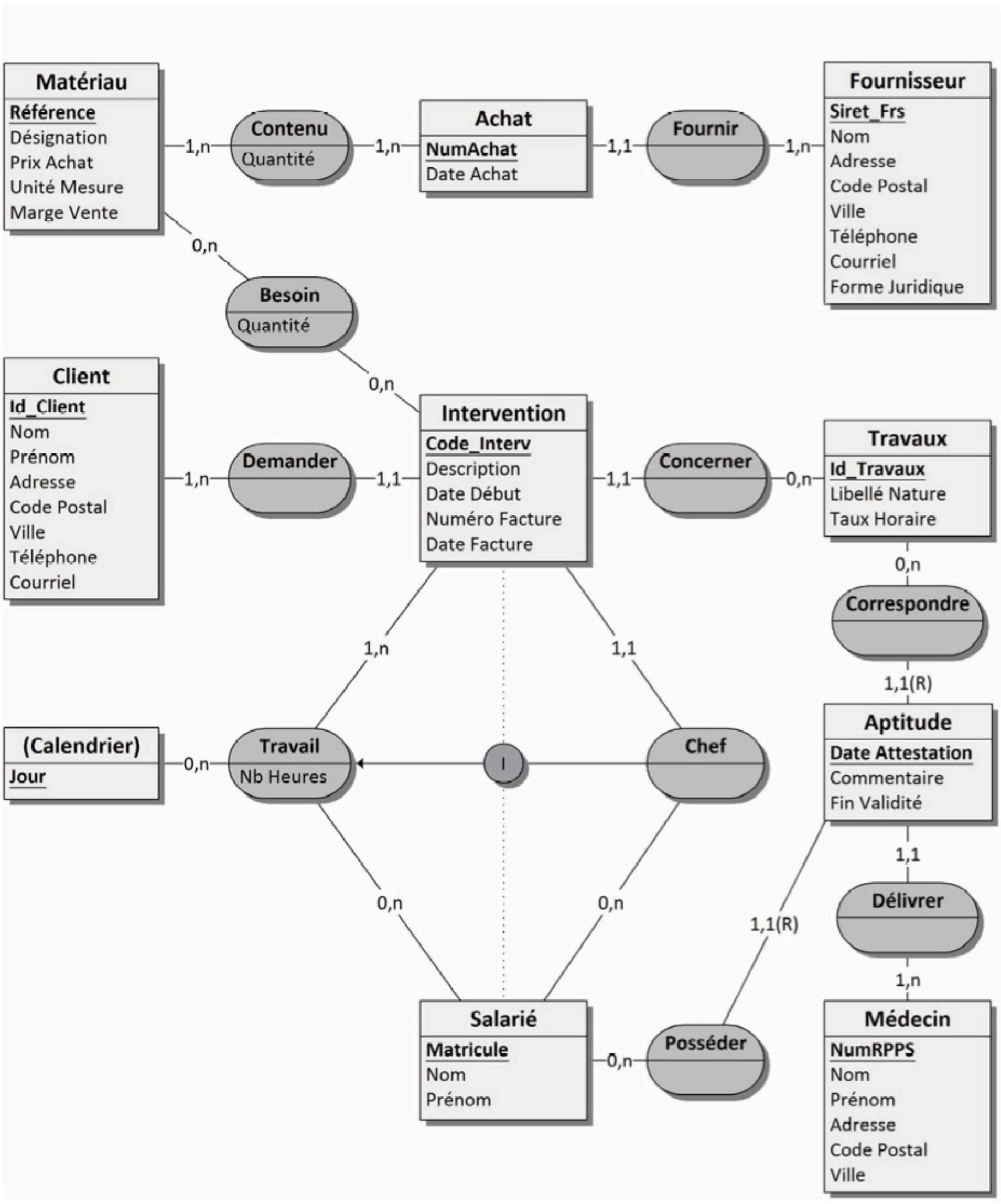
© Patrick Bergougnoux

Dictionnaire des données

- Nom client
- Prénom client
- Adresse client
- Code postal client
- Ville client
- Téléphone client
- Courriel client
- Description intervention
- Code intervention
- Date début intervention
- Libellé nature travaux
- Taux horaire nature travaux
- Jour travaux
- Nom salarié
- Prénom salarié
- Matricule salarié
- Nb heures/jour/salarié
- Nom fournisseur
- Adresse fournisseur
- Code postal fournisseur
- Ville fournisseur
- Téléphone fournisseur
- Courriel fournisseur
- Numéro facture achat
- Date facture achat
- Référence matériau
- Désignation matériau
- Quantité matériau acheté
- Unité mesure matériau
- PU HT achat matériau
- Forme juridique fournisseur
- Numéro SIRET fournisseur
- Qté matériaux/intervention
- Numéro facture client
- Date facture client
- Nom médecin
- Prénom médecin
- Adresse médecin
- Code postal médecin
- Ville médecin
- Numéro RPPS médecin
- Commentaire aptitude
- Date fin validité aptitude
- Date attestation aptitude
- Marge vente matériau

Modèle conceptuel de données

Schéma entité-association



© Patrick Bergougnoux

Diagramme de classes UML

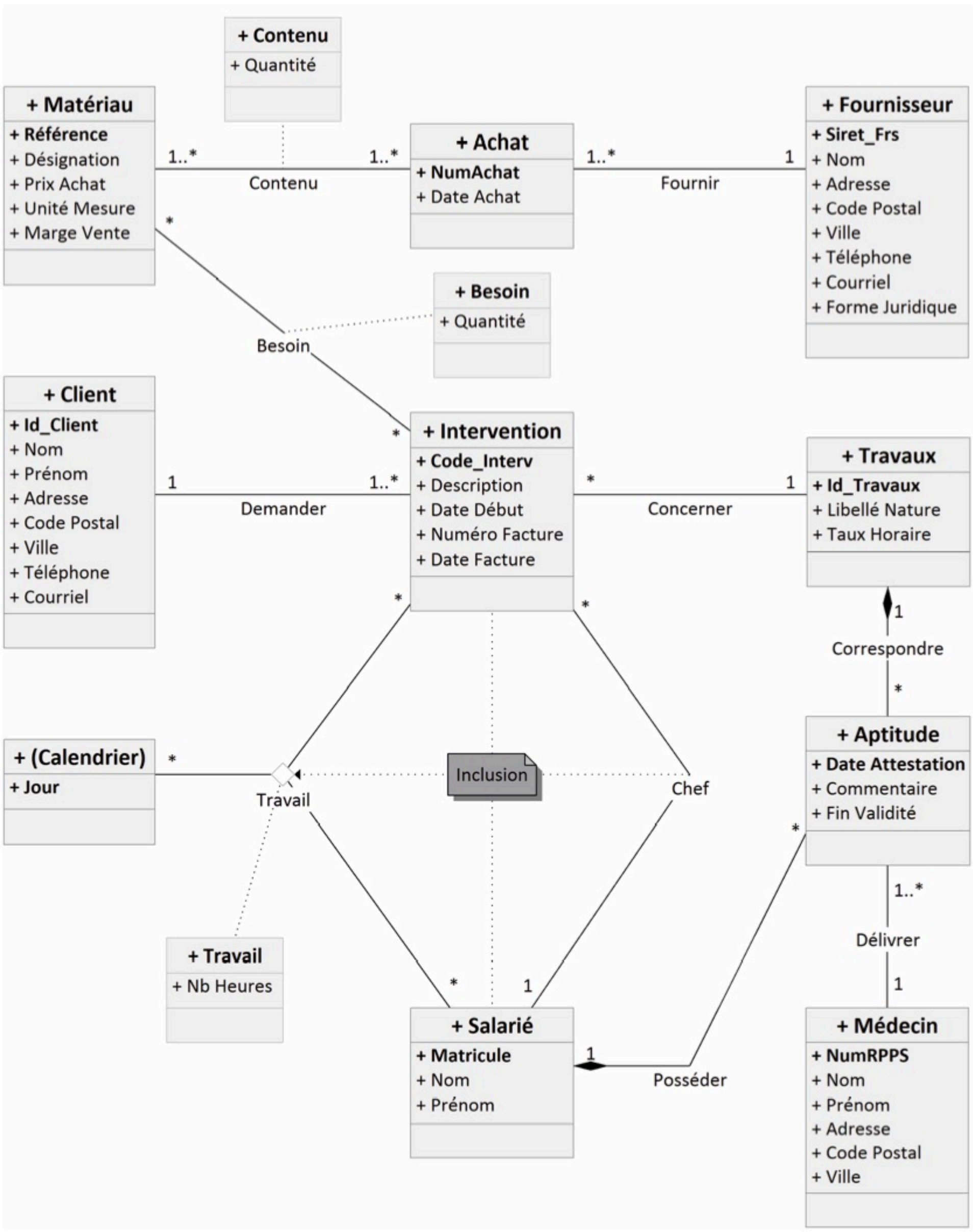


Schéma relationnel

Modèle logique de données

Matériau = (**Référence** *VARCHAR(15)*, Désignation *VARCHAR(50)*, Prix_Achat *CURRENCY*, Unité_Mesure *VARCHAR(10)*, Marge_Vente *BYTE*)

Fournisseur = (**Siret_Frs** *CHAR(14)*, Nom *VARCHAR(50)*, Adresse *VARCHAR(50)*, Code_Postal *CHAR(5)*, Ville *VARCHAR(50)*, Téléphone *VARCHAR(20)*, Courriel *VARCHAR(50)*, Forme_Juridique *VARCHAR(50)*)

Client = (**Id_Client** *COUNTER*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Adresse *VARCHAR(50)*, Code_Postal *CHAR(5)*, Ville *VARCHAR(50)*, Téléphone *VARCHAR(20)*, Courriel *VARCHAR(50)*)

Achat = (**NumAchat** *VARCHAR(15)*, Date_Achat *DATE*, *#Siret_Frs*)

Travaux = (**Id_Travaux** *COUNTER*, Libellé_Nature *VARCHAR(50)*, Taux_Horaire *CURRENCY*)

Médecin = (**NumRPPS** *CHAR(11)*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Adresse *VARCHAR(50)*, Code_Postal *CHAR(5)*, Ville *VARCHAR(50)*)

Salarié = (**Matricule** *SMALLINT*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*)

Aptitude = (***#Matricule, #Id_Travaux*, Date_Attestation** *DATE*, Commentaire *TEXT*, Fin_Validité *DATE*, *#NumRPPS*)

Intervention = (**Code_Interv** *CHAR(8)*, Description *TEXT*, Date_Début *DATE*, Numéro_Facture *CHAR(9)*, Date_Facture *DATE*, *#Matricule, #Id_Client, #Id_Travaux*)

Contenu = (***#Référence, #NumAchat***, Quantité *INT*)

Besoin = (***#Référence, #Code_Interv***, Quantité *INT*)

Travail = (***#Matricule, #Code_Interv*, Jour** *DATE*, Nb_Heures *BYTE*)

Requêtes SQL de création des tables

```
CREATE TABLE Matériau(
        Référence VARCHAR(15),
        Désignation VARCHAR(50),
        Prix_Achat CURRENCY,
        Unité_Mesure VARCHAR(10),
        Marge_Vente BYTE,
        PRIMARY KEY(Référence)
);
```


© Patrick Bergougnoux

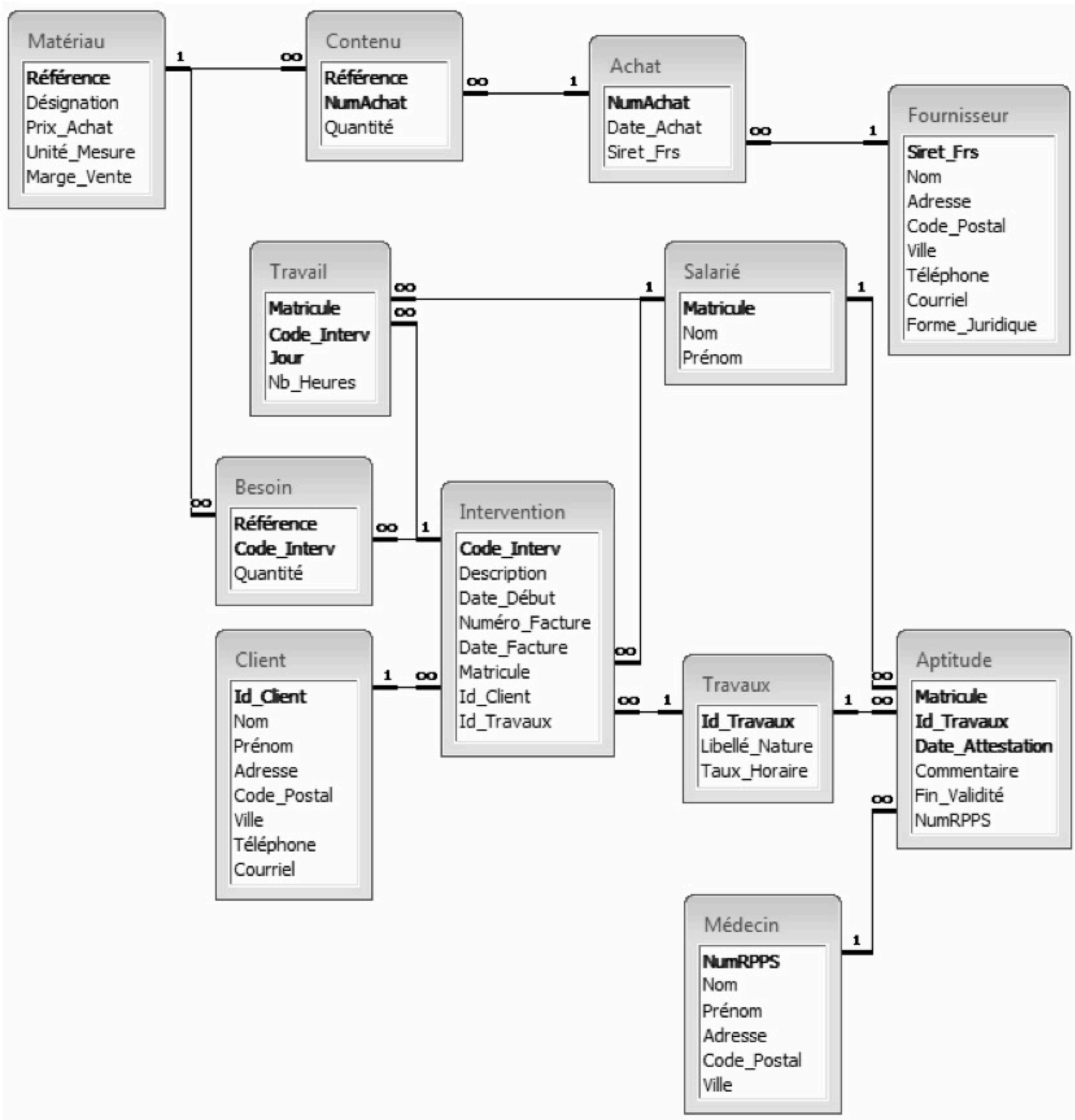
```
CREATE TABLE Fournisseur(
	Siret_Frs CHAR(14),
	Nom VARCHAR(50),
	Adresse VARCHAR(50),
	Code_Postal CHAR(5),
	Ville VARCHAR(50),
	Téléphone VARCHAR(20),
	Courriel VARCHAR(50),
	Forme_Juridique VARCHAR(50),
	PRIMARY KEY(Siret_Frs)
);

CREATE TABLE Client(
	Id_Client COUNTER,
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	Adresse VARCHAR(50),
	Code_Postal CHAR(5),
	Ville VARCHAR(50),
	Téléphone VARCHAR(20),
	Courriel VARCHAR(50),
	PRIMARY KEY(Id_Client)
);

CREATE TABLE Achat(
	NumAchat VARCHAR(15),
	Date_Achat DATE,
	Siret_Frs CHAR(14) NOT NULL,
	PRIMARY KEY(NumAchat),
	FOREIGN KEY(Siret_Frs) REFERENCES Fournisseur(Siret_Frs)
);

CREATE TABLE Travaux(
	Id_Travaux COUNTER,
	Libellé_Nature VARCHAR(50),
	Taux_Horaire CURRENCY,
	PRIMARY KEY(Id_Travaux)
);

CREATE TABLE Médecin(
	NumRPPS CHAR(11),
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	Adresse VARCHAR(50),
	Code_Postal CHAR(5),
	Ville VARCHAR(50),
	PRIMARY KEY(NumRPPS)
);
```

```
CREATE TABLE Salarié(
	Matricule SMALLINT,
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	PRIMARY KEY(Matricule)
);

CREATE TABLE Aptitude(
	Matricule SMALLINT,
	Id_Travaux INT,
	Date_Attestation DATE,
	Commentaire TEXT,
	Fin_Validité DATE,
	NumRPPS CHAR(11) NOT NULL,
	PRIMARY KEY(Matricule, Id_Travaux, Date_Attestation),
	FOREIGN KEY(Matricule) REFERENCES Salarié(Matricule),
	FOREIGN KEY(Id_Travaux) REFERENCES Travaux(Id_Travaux),
	FOREIGN KEY(NumRPPS) REFERENCES Médecin(NumRPPS)
);

CREATE TABLE Intervention(
	Code_Interv CHAR(8),
	Description TEXT,
	Date_Début DATE,
	Numéro_Facture CHAR(9),
	Date_Facture DATE,
	Matricule SMALLINT NOT NULL,
	Id_Client INT NOT NULL,
	Id_Travaux INT NOT NULL,
	PRIMARY KEY(Code_Interv),
	FOREIGN KEY(Matricule) REFERENCES Salarié(Matricule),
	FOREIGN KEY(Id_Client) REFERENCES Client(Id_Client),
	FOREIGN KEY(Id_Travaux) REFERENCES Travaux(Id_Travaux)
);

CREATE TABLE Contenu(
	Référence VARCHAR(15),
	NumAchat VARCHAR(15),
	Quantité INT,
	PRIMARY KEY(Référence, NumAchat),
	FOREIGN KEY(Référence) REFERENCES Matériau(Référence),
	FOREIGN KEY(NumAchat) REFERENCES Achat(NumAchat)
);

CREATE TABLE Besoin(
	Référence VARCHAR(15),
	Code_Interv CHAR(8),
	Quantité INT,
	PRIMARY KEY(Référence, Code_Interv),
	FOREIGN KEY(Référence) REFERENCES Matériau(Référence),
	FOREIGN KEY(Code_Interv) REFERENCES Intervention(Code_Interv)
);
```


© Patrick Bergougnoux

```
CREATE TABLE Travail(
        Matricule SMALLINT,
        Code_Interv CHAR(8),
        Jour DATE,
        Nb_Heures BYTE,
        PRIMARY KEY(Matricule, Code_Interv, Jour),
        FOREIGN KEY(Matricule) REFERENCES Salarié(Matricule),
        FOREIGN KEY(Code_Interv) REFERENCES Intervention(Code_Interv)
);
```

Base de données Access



Commentaires sur le modèle

- À la première lecture du compte-tenu d'interview, une classe d'entités « Facture Client » aurait pu être définie ; la réalisation du modèle correspondant aurait alors fait apparaitre une association **1,1/1,1** entre « Intervention » et « Facture Client » : en effet, le compte-rendu d'interview indique qu'une facture est établie pour chaque intervention. Les deux classes d'entités ont donc été regroupées comme le spécifie la règle 5 de cohérence des modèles.

- L'association « Travail » utilise la classe fictive « Calendrier » ; elle aurait aussi pu être décomposée pour éviter la tripatte. Mais, dans ce cas, la clé minimale de la table « Travail » est bien respectée avec la tripatte qui peut donc être maintenue. De plus, la contrainte inter-associations d'inclusion avec l'association « Chef » a ainsi pu être spécifiée.

- L'aptitude pouvait être perçue comme une association entre les classes d'entités « Salariés » et « Travaux » ; il était néanmoins utile de la décomposer pour gérer l'historique des aptitudes en intégrant la date de l'attestation dans l'identifiant. Cet historique aurait également pu être géré avec une classe fictive « Calendrier », mais la décomposition était tout de même nécessaire pour associer le médecin à l'attestation tout en conservant une clé minimale pour la table « Aptitude ».

- Toujours concernant la gestion de l'historique des données, certains aspects auraient pu être pris en compte tels que le changement de coordonnées des clients et l'évolution des prix d'achat des produits (qui entraine une modification de leur prix de vente). En effet, avec le modèle proposé, si un duplicata de facture client est réalisé alors que ces informations ont changées, le duplicata sera imprimé avec les nouvelles valeurs… et ce ne sera alors plus vraiment un duplicata ! La solution habituellement utilisée pour gérer ce type de problèmes consiste, dans ce cas, à dupliquer les informations du client dans la classe d'entités « Intervention » et le prix de vente des produits (prix d'achat + marge) dans l'association « Besoin », ce qui permet également de modifier occasionnellement les conditions de vente pour une intervention particulière. Bon nombre de situations similaires peuvent ainsi justifier la duplication d'informations à des fins d'historisation.

© Patrick Bergougnoux

Chapitre 20
Société de services

La société de services techniques S&t souhaite gérer l'affectation de ses collaborateurs dans ses différents services.

Compte-rendu d'interview

Description du système d'information

S&t travaille avec des collaborateurs salariés embauchés par la direction qui transmet leur fiche de poste (cf. annexe) au service des ressources humaines (RH). S&t travaille également avec des indépendants : ces derniers interviennent à la demande du service RH de S&t et sont rémunérés à la journée. Le service RH leur envoie une fiche d'emploi similaire à la fiche de poste, excepté pour la date d'embauche qui est remplacée par des dates de début et fin de contrat ; l'horaire mensuel et le salaire brut sont remplacés par le tarif de la journée. L'indépendant doit retourner cette fiche pour valider le contrat.

Chaque collaborateur occupe une seule fonction décrite sur la fiche de poste des salariés et la fiche d'emploi des indépendants, sachant que plusieurs collaborateurs peuvent occuper une même fonction. De plus, chaque fonction est numérotée au sein de la catégorie de personnel dont elle relève, chaque catégorie étant caractérisée par son code et son libellé (C : cadre, E : employé, S : stagiaire).

Par ailleurs, si des enfants et leurs parents travaillent dans la société, le service RH en informe la direction.

S&t est composée de plusieurs services (direction, RH, technique, commercial, compta, …) caractérisés par leur nom et l'étage auquel ils se trouvent dans le bâtiment. Chaque service a un salarié chef de service, un salarié ne pouvant pas être chef de plusieurs services ; un indépendant ne peut pas être chef de service. Il est également précisé qu'un service doit posséder au moins un cadre.

La direction affecte ses collaborateurs aux différents services en leur adressant une note indiquant la date de début et le nombre de jours d'affectation. Un collaborateur peut ainsi être affecté plusieurs fois, à des dates différentes, à un même service ou à des services différents.

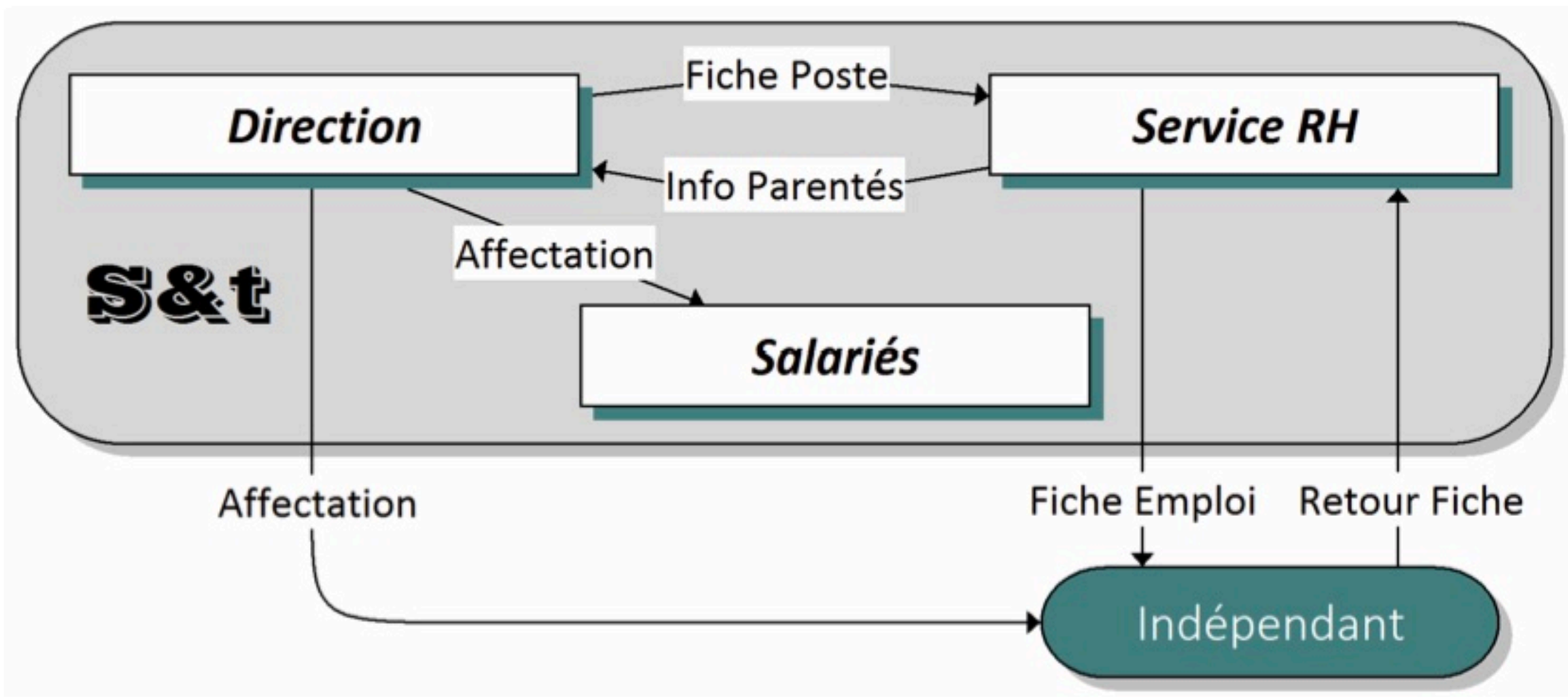
Annexe

S&t **FICHE DE POSTE**

Matricule : 12345678	Catégorie C : Cadre
Nom : GYVER	Fonction 3 : Responsable Maintenance
Prénom : Mac	Description : doit assurer le suivi du matériel informatique
Adresse : 27 rue balise – 31130 BALMA	
Date Naissance : 25/04/1982	Horaire : 151h / mois
Date Embauche : 01/04/2015 – 9h30	Salaire Brut : 2 400 €uros

Recensement des informations

Modèle organisationnel de flux



© Patrick Bergougnoux

Tableau des rubriques

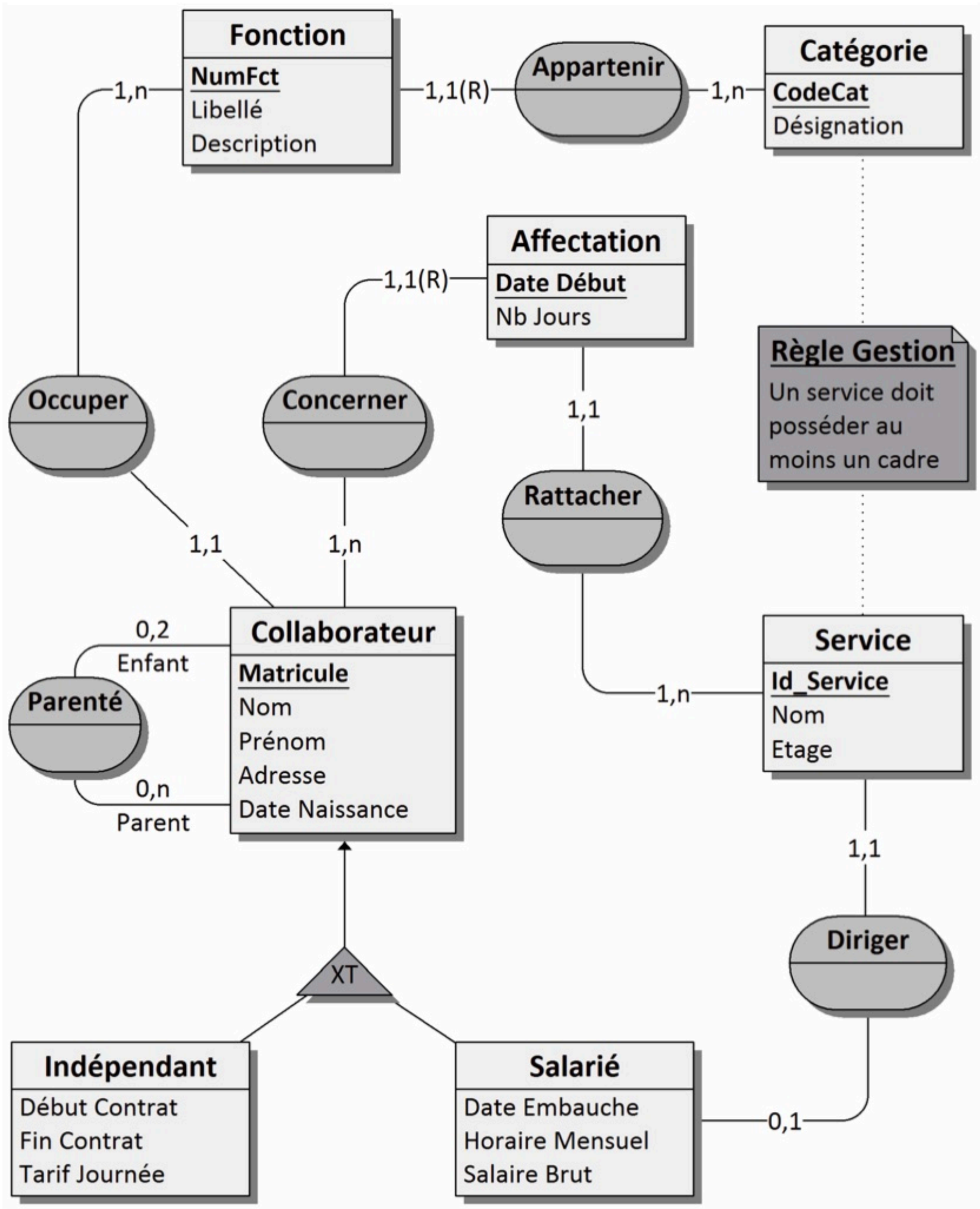
N°	Libellé	Exemple	P/C	Calcul
~~1~~	~~Nom société~~	S&t	P	*Paramètre*
2	Matricule collaborateur	12345678	P	
3	Nom collaborateur	GYVER	P	
4	Prénom collaborateur	Mac	P	
5	Adresse collaborateur	27 rue balise …	P	
6	Date naissance collaborateur	25/04/1982	P	
7	Date embauche salarié	01/04/2015 – 9h30	P	
8	Horaire mensuel salarié	151	P	
9	Salaire brut salarié	2 400 €	P	
10	Début contrat indépendant	01/03/2019	P	
11	Fin contrat indépendant	30/04/2019	P	
12	Tarif journée indépendant	230 €	P	
13	Code catégorie	C	P	
14	Libellé catégorie	Cadre	P	
15	Numéro fonction	3	P	
16	Libellé fonction	Responsable maintenance	P	
17	Description fonction	doit assurer le suivi…	P	
18	Nom service	Technique	P	
19	Étage service	2	P	
20	Date début affectation	18/03/2019	P	
21	Nombre jours affectation	15	P	

Dictionnaire des données

- Matricule collaborateur
- Nom collaborateur
- Prénom collaborateur
- Adresse collaborateur
- Date naissance collaborateur
- Date embauche salarié
- Horaire mensuel salarié
- Salaire brut salarié
- Début contrat indépendant
- Fin contrat indépendant
- Tarif journée indépendant
- Code catégorie
- Libellé catégorie
- Numéro fonction
- Libellé fonction
- Description fonction
- Nom service
- Étage service
- Date début affectation
- Nombre jours affectation

Modèle conceptuel de données

Schéma entité-association



© Patrick Bergougnoux

Diagramme de classes UML

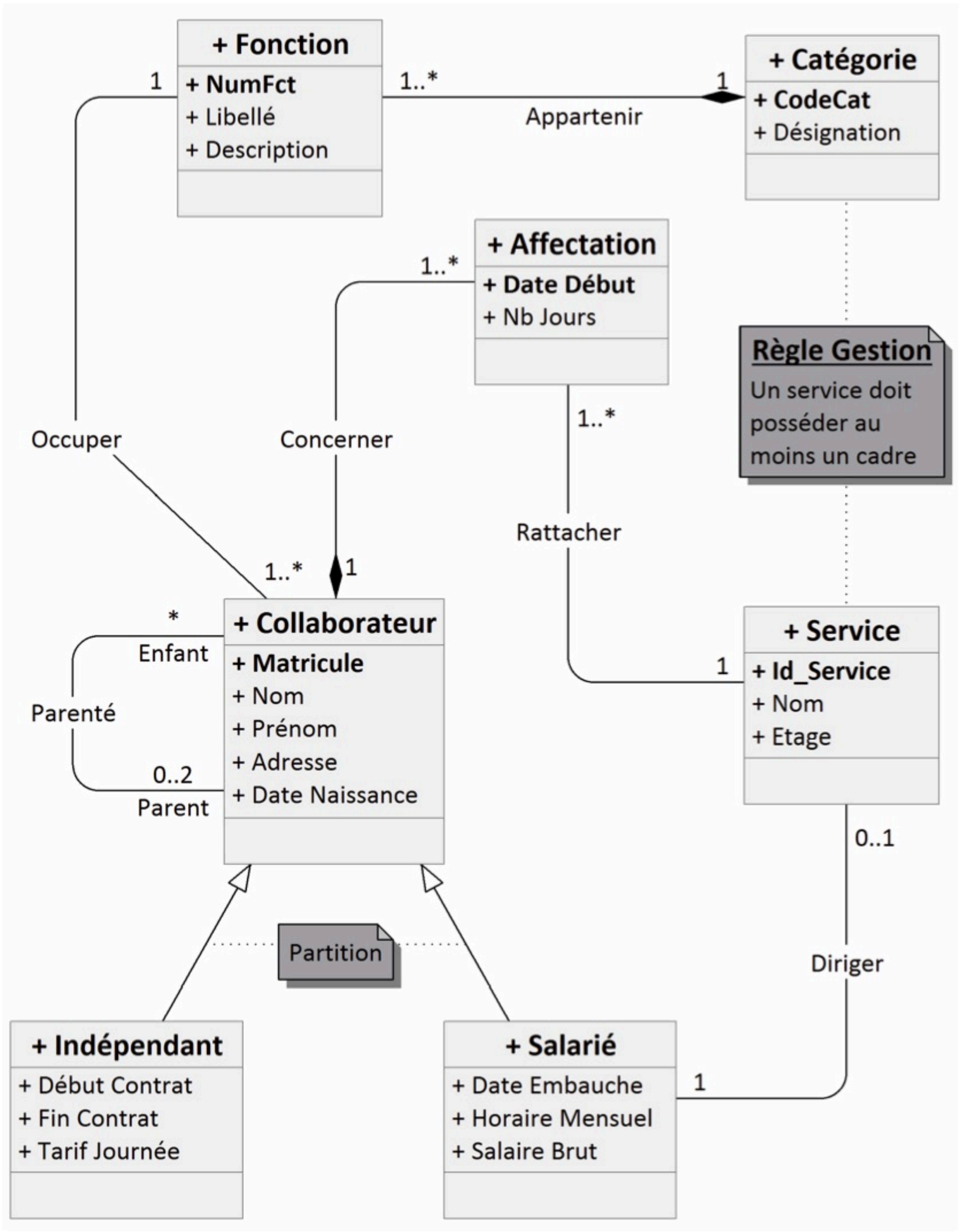


Schéma relationnel

Modèle logique de données

Catégorie = (**CodeCat** *CHAR(1)*, Désignation *VARCHAR(50)*)

Fonction = (***#CodeCat*, NumFct** *BYTE*, Libellé *VARCHAR(50)*, Description *TEXT*)

Collaborateur = (**Matricule** *CHAR(8)*, Nom *VARCHAR(50)*, Prénom *VARCHAR(50)*, Adresse *VARCHAR(80)*, Date_Naissance *DATE*, *#(#CodeCat, NumFct)*)

Salarié = (***#Matricule***, Date_Embauche *DATETIME*, Horaire_Mensuel *DECIMAL(5,2)*, Salaire_Brut *CURRENCY*)

Indépendant = (***#Matricule***, Début_Contrat *DATE*, Fin_Contrat *DATE*, Tarif_Journée *CURRENCY*)

Service = (**Id_Service** *COUNTER*, Nom *VARCHAR(50)*, Etage *BYTE*, *#Matricule*)

Affectation = (***#Matricule*, Date_Début** *DATE*, Nb_Jours *SMALLINT*, *#Id_Service*)

Parenté = (***#Matricule_Parent, #Matricule_Enfant***)

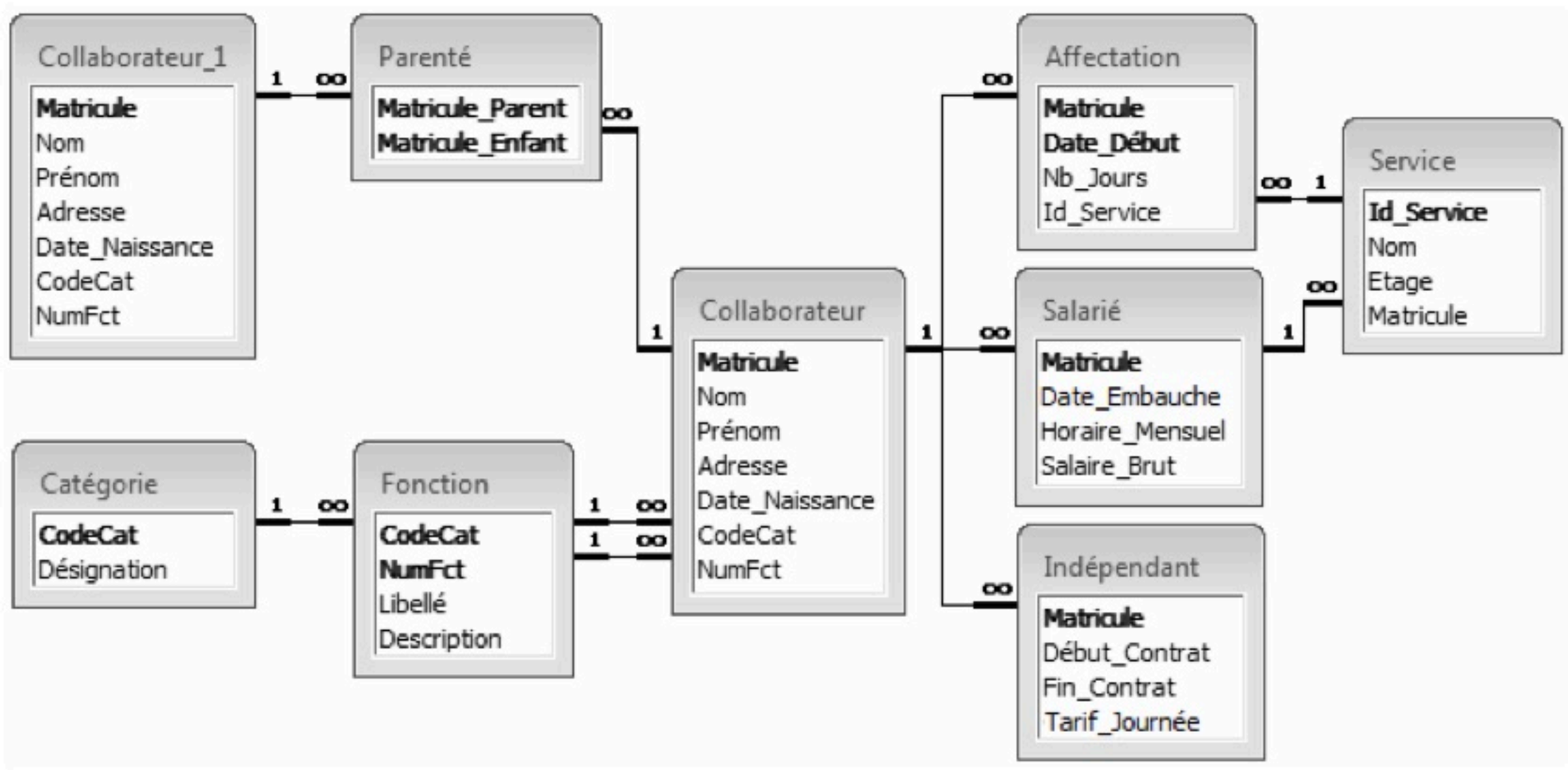
Requêtes SQL de création des tables

```
CREATE TABLE Catégorie(
	CodeCat CHAR(1),
	Désignation VARCHAR(50),
	PRIMARY KEY(CodeCat)
);

CREATE TABLE Fonction(
	CodeCat CHAR(1),
	NumFct BYTE,
	Libellé VARCHAR(50),
	Description TEXT,
	PRIMARY KEY(CodeCat, NumFct),
	FOREIGN KEY(CodeCat) REFERENCES Catégorie(CodeCat)
);

CREATE TABLE Collaborateur(
	Matricule CHAR(8),
	Nom VARCHAR(50),
	Prénom VARCHAR(50),
	Adresse VARCHAR(80),
	Date_Naissance DATE,
	CodeCat CHAR(1) NOT NULL,
	NumFct BYTE NOT NULL,
	PRIMARY KEY(Matricule),
	FOREIGN KEY(CodeCat, NumFct) REFERENCES Fonction(CodeCat, NumFct)
);
```


© Patrick Bergougnoux

```
CREATE TABLE Salarié(
	Matricule CHAR(8),
	Date_Embauche DATETIME,
	Horaire_Mensuel DECIMAL(5,2),
	Salaire_Brut CURRENCY,
	PRIMARY KEY(Matricule),
	FOREIGN KEY(Matricule) REFERENCES Collaborateur(Matricule)
);

CREATE TABLE Indépendant(
	Matricule CHAR(8),
	Début_Contrat DATE,
	Fin_Contrat DATE,
	Tarif_Journée CURRENCY,
	PRIMARY KEY(Matricule),
	FOREIGN KEY(Matricule) REFERENCES Collaborateur(Matricule)
);

CREATE TABLE Service(
	Id_Service COUNTER,
	Nom VARCHAR(50),
	Etage BYTE,
	Matricule CHAR(8) NOT NULL,
	PRIMARY KEY(Id_Service),
	FOREIGN KEY(Matricule) REFERENCES Salarié(Matricule)
);

CREATE TABLE Affectation(
	Matricule CHAR(8),
	Date_Début DATE,
	Nb_Jours SMALLINT,
	Id_Service INT NOT NULL,
	PRIMARY KEY(Matricule, Date_Début),
	FOREIGN KEY(Matricule) REFERENCES Collaborateur(Matricule),
	FOREIGN KEY(Id_Service) REFERENCES Service(Id_Service)
);

CREATE TABLE Parenté(
	Matricule_Parent CHAR(8),
	Matricule_Enfant CHAR(8),
	PRIMARY KEY(Matricule_Parent, Matricule_Enfant),
	FOREIGN KEY(Matricule_Parent) REFERENCES Collaborateur(Matricule),
	FOREIGN KEY(Matricule_Enfant) REFERENCES Collaborateur(Matricule)
);
```

Base de données Access



Dans ce dernier modèle conceptuel de données, on remarque que certains liens sont arrondis, ce qui rend le schéma plus agréable à lire. En fait, en cas de cassure, ***Looping*** propose un lissage des liens : s'agissant de courbes de Bézier, on peut ainsi obtenir, avec de multiples cassures intermédiaires, n'importe quelle forme de lien.

Commentaires sur le modèle et conclusion

- Ce modèle est relativement simple mais intègre néanmoins bon nombre de concepts étudiés dans cet ouvrage avec, en particulier, la notion d'héritage, le rajout d'une règle de gestion et l'utilisation d'une association réflexive asymétrique avec définition de rôles.
- La classe d'entités « Affectation » était initialement une association ternaire entre les classes « Collaborateur », « Service » et une classe fictive « Calendrier ». Nous avons décomposé cette association afin d'exclure « Id_Service » de la clé primaire de la table « Affectation » pour que cette clé soit minimale.
- Le même schéma relationnel peut être obtenu en utilisant une CIF ayant pour cible la classe « Service » avec, en plus, la possibilité d'intégrer une contrainte inter-associations d'inclusion : je vous laisse découvrir le modèle correspondant… **en couverture de ce livre !**

Pour nous, la boucle est bouclée. ***For Looping, Loop is Looped !***

© Patrick Bergougnoux

Proximus 12:46 41 %

Modélisation Conceptuelle de Données - Une Démarche Pragmatique

Bergougnoux, Patrick

Début

Couverture 1

Avant-propos 7

Table des matières 15

Première partie - Recensement des informations 27

1 - Analyse des flux d'information 29

2 - Modèle organisationnel de flux 37

3 - Dictionnaire des données 45

4 - Exercices de recensement 59

Deuxième partie - Bases de la

Modélisation Conceptuelle de Données - Une Démarche Pragmatique

Bergougnoux, Patrick

5 - Répartition en classes d'entités 79

6 - Association des classes d'entités 87

7 - Règles de cohérence des modèles 97

8 - Exercices de modélisation 109

Troisième partie - Informatisation des modèles 155

9 - Mise en relation des tables 157

10 - Modèle logique de données 171

11 - Requêtes SQL de création de tables 179

12 - Reprise exercices avec MLD et SQL 183

Modélisation Conceptuelle de Données - Une Démarche Pragmatique

Bergougnoux, Patrick

Quatrième partie - Modélisation avancée 209

13 - Identifiant relatif 211

14 - Association n-aire 225

15 - Compléments de modélisation 241

16 - Diagramme de classes UML 253

Cinquième partie - Études de cas complètes 273

17 - Association sportive et culturelle 275

18 - Compagnie d'assurances 285

19 - Entreprise génerale du bâtiment 297

20 - Société de services 313