Passage UML-Relationnel

Table des matières

I Contexte

Durée : 2h

Environnement de travail : DB Fiddle

Pré-requis : Aucun

Lorsque vous concevez une base de données, vous obtenez à un moment donné un modèle conceptuel. Or, si le modèle conceptuel permet de synthétiser de manière instinctive la modélisation, il reste assez éloigné de ce que sont capables de gérer les systèmes de gestion de bases de données : des relations, ou tableaux.

Afin de pouvoir implémenter une base de données, il faut traduire le modèle conceptuel en un modèle intermédiaire, appelé modèle relationnel.

Les modèles conceptuels sont suffisamment formalisés pour que ce passage soit systématisé dans la plupart des cas, et c'est cette méthode de passage que vous allez découvrir au long de ces modules.

II Transformation des classes

Objectif

- Savoir transformer une classe en relation.

Mise en situation

Supposez qu'on vous envoie un modèle conceptuel composé de quelques classes, par exemple une classe qui décrit les personnages d'un jeu de rôle. Chaque personnage a un nom, un type, des points de vie, etc. On sait que le nom est unique, que chaque personnage peut attaquer ou se défendre...

En vue de créer une base de données, vous décidez d'utiliser un modèle relationnel et de transformer cette classe en relation.

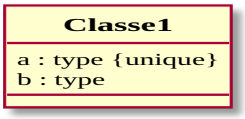
Quelles sont les règles de transformation ? C'est ce que vous allez découvrir dans ce module.

Classe

Méthode

Pour chaque classe :

- on crée une table dont le nom correspond à celui de la classe,
- on crée un attribut dans la table pour chaque attribut de la classe, de même nom et de même type,
- on choisit la clé primaire de cette table parmi l'une des clés de la classe.



Graphique 1

```
1 Table1 (#a:type, b:type)
```

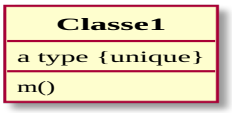
Exemple


```
1 Hotel(#nom:string, adresse:string, codePostal:string, ville:string,
  étoiles:integer)
```

Méthodes

Méthode

On ne représente pas les méthodes dans le modèle relationnel, elles seront calculées dynamiquement soit par des vues au niveau de la BD, soit par des fonctions au niveau applicatif.



Graphique 2

```
1 Classe1(#a:type)
```

Méthodes stockées

Complément

On peut décider de représenter la méthode comme s'il s'agissait d'un attribut simple pour des raisons de performance essentiellement. Il sera nécessaire dans ce cas d'ajouter des mécanismes de validation de contraintes pour s'assurer que la valeur stockée évolue en même temps que les attributs sur lesquels le calcul dérivé porte. On peut réaliser cela avec par exemple des *triggers*.

Attention

Introduire un résultat de méthode dans le modèle relationnel conduit à de la redondance. Cela est en général déconseillé et doit être dans tous les cas contrôlé.

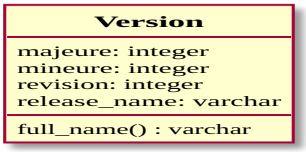
III Exercice : Appliquer la notion

Question

[solution n°1 p. 24]

Un modèle UML simple doté d'une seule classe représente la gestion de versions pour un logiciel informatique.

Transformer la classe UML en relationnel.



IV Transformation des associations 1:N

Objectif

- Savoir intégrer des associations de type 1:N dans des relations.

Mise en situation

Vous venez de créer un modèle conceptuel qui modélise un système de réservation de chambre d'hôtel. En particulier, votre modèle comporte deux classes : les hôtels, et les chambres.

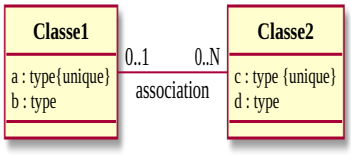
Chaque chambre ne peut appartenir qu'à un seul hôtel, c'est donc tout naturellement que vous modélisez ce lien par une association 1:N.

Mais pour passer au modèle relationnel, vous ne disposez que du concept de relations : à quel endroit indiquer qu'une chambre est située dans un hôtel en particulier ? Plus généralement, comment transformer les associations 1:N en modèle relationnel ? C'est ce que vous allez découvrir dans ce module.

Méthode

Pour chaque association de type 1:N :

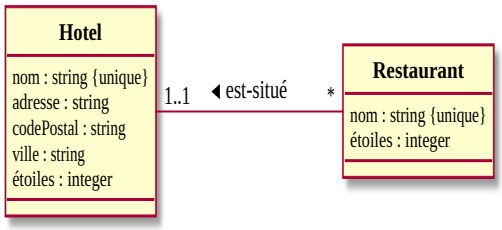
- on ajoute à la table côté N une clé étrangère vers la table côté 1.



Graphique 3

```
Table1 (#a:type, b:type)
Table2 (#c:type, d:type, a=>Table1)
```

Exemple




```
Hotel(#nom:string, adresse:string, codePostal:string, ville:string,
étoiles:integer)
Restaurant (#nom:string, étoiles:integer, hotel=>Hotel) avec hotel non null
```

Expressions des contraintes de cardinalité minimale

Complément

- Si la cardinalité est exactement 1 côté 1 (`1..1`), alors on ajoutera une contrainte de non nullité sur la clé étrangère.
- Si la cardinalité est au moins 1 côté N (`1..N`), on ajoutera une contrainte d'existence d'enregistrements référençant pour chaque enregistrement de la table référencée : `PROJECTION(Table1,a) = PROJECTION(Table2,a)`.

  La projection d'une table est une opération qui sélectionne un sous-ensemble des attributs de cette table et supprime les doublons.

V Exercice : Appliquer la notion

Description du problème

Un laboratoire souhaite gérer les médicaments qu'il conçoit.

- Un médicament est décrit par un nom, qui permet de l'identifier. En effet il n'existe pas deux médicaments avec le même nom. Un médicament comporte une description courte en français, ainsi qu'une description longue en latin. On gère aussi le conditionnement du médicament, c'est-à-dire le nombre de pilules par boîte (qui est un nombre entier).
- À chaque médicament on associe une liste de contre-indications, généralement plusieurs, parfois aucune. Une contre-indication comporte un code unique qui l'identifie, ainsi qu'une description. Une contre-indication est toujours associée à un et un seul médicament.

Exemple de données

Afin de matérialiser notre base de données, nous obtenons les descriptions suivantes :

- Le **Chourix** a pour description courte *« Médicament contre la chute des choux »* et pour description longue *« Vivamus fermentum semper porta. Nunc diam velit, adipiscing ut tristique vitae, sagittis vel odio. Maecenas convallis ullamcorper ultricies. Curabitur ornare. »*. Il est conditionné en boîte de 13.

  Ses contre-indications sont :

  - CI1 : Ne jamais prendre après minuit.
  - CI2 : Ne jamais mettre en contact avec de l'eau.
- Le **Tropas** a pour description courte *« Médicament contre les dysfonctionnements intellectuels »* et pour description longue *« Suspendisse lectus leo, consectetur in tempor sit amet, placerat quis neque. Etiam luctus porttitor lorem, sed suscipit est rutrum non. »*. Il est conditionné en boîte de 42.

  Ses contre-indications sont :

  - CI3 : Garder à l'abri de la lumière du soleil.

Question 1

[solution n°2 p. 24]

Réaliser le modèle conceptuel de données en UML du problème.

Question 2

[solution n°3 p. 24]

Dessiner des tableaux remplis avec les données fournies en exemple, afin de montrer que le modèle fonctionne selon le besoin exprimé initialement. On pourra se limiter aux premiers mots des descriptions pour gagner du temps.

VI Transformation des associations N:M

Objectif

- Savoir intégrer des associations de type N:M dans des relations.

Mise en situation

De nombreux modèles conceptuels contiennent des associations N:M. Pensez par exemple à un système d'emprunt de bibliothèque, où chaque livre peut être emprunté par plusieurs personnes, et où chaque personne peut emprunter plusieurs livres.

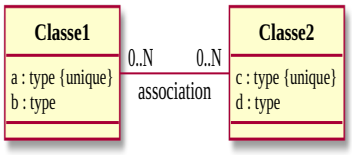
Au moment de passer au modèle relationnel, une problématique forte se pose : impossible de représenter cette association en ajoutant une clé étrangère représentant le livre dans la relation personne. En effet, si cette manière de faire permet de dire quel livre a été emprunté par quelle personne, impossible de représenter le fait qu'une personne ait emprunté plusieurs livres !

Comment faire ? C'est ce que vous allez découvrir dans ce module.

Méthode

Pour chaque association de type `M:N` :

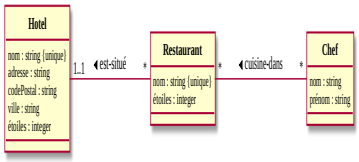
- on crée une nouvelle table,
- composée de deux clés étrangères, une vers chaque table associée aux classes liées,
- et dont la clé primaire est la concaténation de ces deux clés étrangères.



Graphique 4

```
Table1 (#a:type, b:type)
Table2 (#c:type, d:type)
Assoc (#a=>Table1,#c=>Table2)
```

Exemple

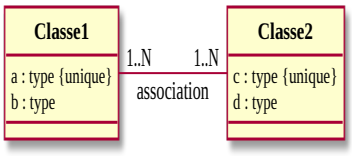



```
Hotel(#nom:string, adresse:string, codePostal:string, ville:string,
étoiles:integer)
Restaurant (#nom::string, étoiles:integer, hotel=>Hotel) avec hotel non null
Chef (#id:integer, nom:string, prénom:string)
Cuisine (#chef=>Chef, #restaurant=>Restaurant)
```

Expressions des contraintes de cardinalité minimale

Complément

Si la cardinalité est au moins 1 (`1..N`) d'un côté et/ou de l'autre, alors des contraintes d'existence simultanée de tuples devront être ajoutées. Cela peut être exprimé avec des projections.



Graphique 5

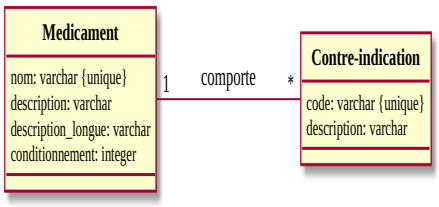
```
Table1 (#a:type, b:type)
Table2 (#c:type, d:type)
Assoc (#a=>Table1, #c=>Table2)
Contraintes : PROJECTION(Table1,a) = PROJECTION(Assoc,a) AND
PROJECTION(Table2,c) = PROJECTION(Assoc,c)
```

Si la cardinalité est `0..N:1..N`, alors seule une des deux contraintes est exprimée.

VII Exercice : Appliquer la notion

Description du problème

Soit le modèle UML ci-après.



On souhaite ajouter la gestion des composants des médicaments. Un composant est identifié par un code unique et possède un intitulé. Tout médicament possède au moins un composant, souvent plusieurs. Tout composant peut intervenir dans la fabrication de plusieurs médicaments. Il existe des composants qui ne sont pas utilisés pour fabriquer des médicaments et que l'on veut quand même gérer.

Exemple de données

- Les composants existants sont :
  - **HG79** : *« Vif-argent allégé »*
  - **HG81** : *« Vif-argent alourdi »* ;
  - **SN50** : *« Pur étain »*.
- Le **Chourix** a pour composant le **HG79** et le **SN50**.
- Le **Tropas** a pour unique composant le **HG79**.

Question 1

[solution n°4 p. 24]

Étendre le modèle conceptuel UML afin d'ajouter la gestion des composants.

Question 2

[solution n°5 p. 25]

En mobilisant les règles adéquates, proposer un modèle logique de données correspondant en relationnel à partir du modèle conceptuel développé précédemment. Le repérage des domaines et des clés est obligatoire.

Question 3

[solution n°6 p. 25]

On dispose de la table représentant les médicaments, ajoutez les tables de données correspondant aux composants.

Medicament

#nom	*description*	*desc_longue*	*cond.*
Chourix	Médicament contre la chute des choux	Vivamus...	11
Tropas	Médicament contre les dys...	Suspendisse...	42

VIII Transformation des associations 1:1

Objectifs

- Savoir intégrer des associations 1:1 dans des relations ;
- Savoir préserver l'unicité de l'association.

Mise en situation

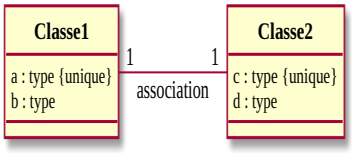
Supposez que vous modélisiez un monde où chaque citoyen, à la naissance, obtient une carte d'identité. En modèle conceptuel, la carte d'identité et le citoyen sont des classes bien différentes, et comme une carte d'identité ne peut appartenir qu'à un seul citoyen et inversement, vous utiliserez une association `1:1`.

Au moment de passer au modèle relationnel, il pourrait être tentant de traiter cette association `1:1` comme une association `1:N`, et de référencer la carte d'identité comme clé étrangère de la relation citoyen, par exemple.

Mais comment exprimer le fait qu'une carte d'identité n'appartient qu'à un citoyen ? C'est ce que vous allez découvrir dans ce module.

Méthode

La solution la plus simple et la plus générale pour transformer une association `1:1` consiste à traiter cette association `1:1` comme une association `1:N`, puis à ajouter une contrainte `UNIQUE` sur la clé étrangère pour limiter la cardinalité maximale à 1.



Graphique 6

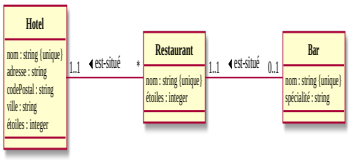
```
Table1 (#a, b, c=>Table2) avec c UNIQUE NOT NULL
Table2 (#c, d)

Table1(#a, b)
Table2(#c, d, a=>Table1) avec a UNIQUE NOT NULL
```

Remarque

Si la cardinalité minimale est `1..1` on ajoute la contrainte *« UNIQUE NOT NULL »*.

Exemple




```
Hotel(#nom:string, adresse:string, codePostal:string, ville:string,
étoiles:integer)
Restaurant (#nom:string, étoiles:integer, hotel=>Hotel) avec hotel non null
Bar (#nom:string, spécialité:string, restaurant=>Restaurant) avec restaurant
unique non null
```

Attention

Il existe toujours deux solutions selon que l'on choisit l'une ou l'autre relation pour accueillir la clé étrangère. Selon la cardinalité minimale, un des deux choix peut être plus pertinent.

Complément

Il est parfois possible de choisir de fusionner les deux classes au sein d'une seule relation plutôt que d'opter pour une clé étrangère.

IX Exercice : Appliquer la notion

Dans une course équestre, plusieurs jockeys participent chacun avec leur propre cheval.

Les jockeys disposent d'un nom, d'un prénom, d'un sponsor et d'un numéro de compétiteur les identifiant.

De même, chaque cheval dispose d'un numéro d'identification, d'un poids, d'une taille et d'une race.

Question 1

[solution n°7 p. 25]

Représenter avec un diagramme UML la situation. Utiliser deux classes.

Question 2

[solution n°8 p. 25]

Traduire de deux façons différentes le modèle logique associé à ce diagramme UML.

X Transformation des classes d'association

Objectif

- Savoir exprimer les propriétés des classes d'association en modèle relationnel.

Mise en situation

Supposez que vous modélisiez une plateforme de mise en relation entre des auto-entrepreneurs et des clients : chaque auto-entrepreneur peut accepter plusieurs missions, et chaque client peut travailler avec plusieurs auto-entrepreneurs.

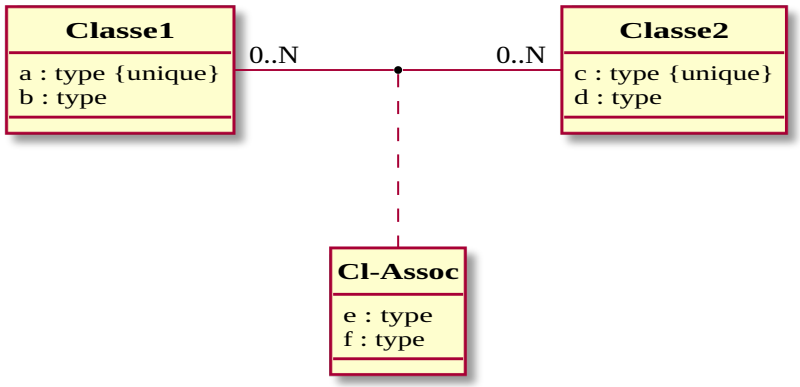
Naturellement, en modèle conceptuel, vous utilisez une association `N:M` entre client et auto-entrepreneurs : mais vous voulez aussi stocker le nom du projet, par exemple. Il faut alors stocker la propriété dans l'association directement, et vous la transformez en classe d'association.

Mais comment, en passant au modèle relationnel, représenter ces propriétés ? C'est ce que vous allez découvrir dans ce module.

Classe d'association N:M

Méthode

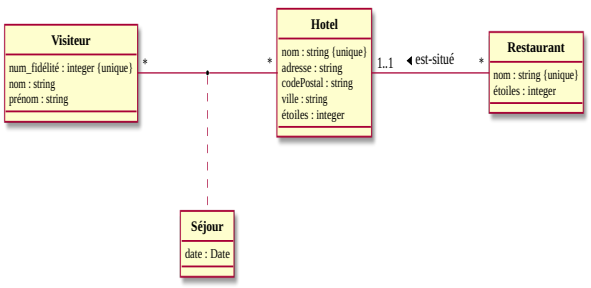
Les attributs de la classe d'association sont ajoutés à la table issue de l'association `N:M`.



Graphique 7

```
Table1 (#a, b)
Table2 (#c, d)
Assoc (#a=>Table1,#c=>Table2, e, f)
```

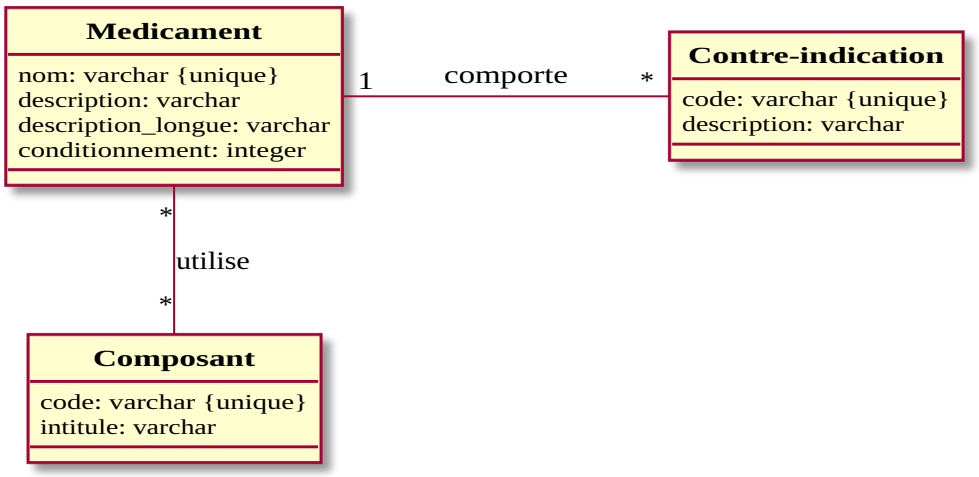
Exemple



```
Hotel(#nom:string, adresse:string, codePostal:string, ville:string,
étoiles:integer)
Restaurant (#nom:string, étoiles:integer, hotel=>Hotel) avec hotel non null
Visiteur(#num_fidélité:integer, nom:string, prénom:string)
Séjour(#visiteur=>Visiteur, #hotel=>Hotel, date)
```

XI Exercice : Appliquer la notion

Soit la modélisation suivante représentant des médicaments, leurs composants et leurs contre-indications.



Question 1

[solution n°9 p. 26]

On précise qu'un composant est présent dans un médicament avec un certain dosage donné en µg, au centième près.

Étendre le modèle conceptuel UML afin d'ajouter cette information.

Question 2

[solution n°10 p. 26]

En mobilisant les règles adéquates, proposer un modèle logique de données correspondant en relationnel à partir du modèle conceptuel développé précédemment. Le repérage des domaines et des clés est obligatoire.

XII Essentiel

Passer d'un système de classes reliées par des associations, le modèle conceptuel, à un système composé de tableaux plats, le modèle relationnel, pose plusieurs défis, en particulier dans la transformation des associations. La partie la plus simple, c'est que chaque classe donne une relation.

Transformer une association `1:N` ou `1:1` revient à créer une clé étrangère dans la relation côté *« N »*, par exemple pour indiquer à qui appartient un vélo.

Transformer une association `N:M` nécessite de créer une nouvelle table qui va faire le lien entre les deux relations, par exemple pour indiquer la correspondance entre plusieurs ingrédients et plusieurs recettes de cuisine.

Enfin, les attributs des classes d'associations sont simplement rajoutés en plus des clés étrangère induites par l'association elle-même.

XIII Quiz

Exercice 1

[solution n°11 p. 26]

Lorsque l'on a dans le diagramme de conception UML une classe d'association avec une cardinalité "`0..N:0..N`" on doit :

A

☐ Mettre en place une relation qui correspond à cette association au sein de laquelle on référence les deux relations associées grâce à deux clefs étrangères.

B

☐ Intégrer les attributs de la classe d'association dans la relation créée pour transformer la classe d'association.

Exercice 2

[solution n°12 p. 27]

A Ajouter une clé étrangère à l'une des deux relations et une contrainte d'unicité sur cette clé

B Ajouter une table contenant deux clés étrangères

C Ajouter une clé étrangère à la relation côté N

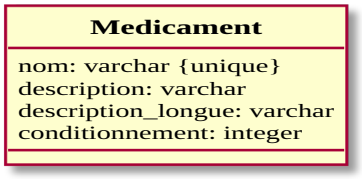
Association *« 1:N »*	Association *« N:M »*	Association *« 1:1 »*

Exercice 3 : Quiz - Code

[solution n°13 p. 27]

Exercice

Quel est le schéma relationnel adapté pour ce diagramme de classes UML ?



A

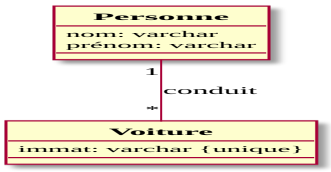
Medicament(nom: varchar, description: varchar, description_longue: varchar, conditionnement: entier)

B

Medicament(#nom: varchar, description: varchar, description_longue: varchar, conditionnement: entier)

Exercice

Quel est le schéma relationnel adapté pour ce diagramme de classes UML ?



A

Personne(#id_personne: entier, nom: text, prenom: text) Voiture(#immat: text)

B

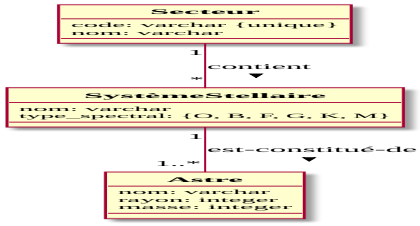
Personne(#id_personne: entier, nom : text, prenom : text) Voiture(#immat: text, personne=>Personne)

C

Personne(#id_personne: entier, nom : text, prenom : text, voiture=>Voiture) Voiture(#immat: text)

Exercice

Quel est le schéma relationnel adapté pour ce diagramme de classes UML ?



A

```
Secteur(#code: varchar, nom : varchar)
SystemeStellaire(nom: varchar, type_spectral: {0, B, F, G, K, M})
Astre(nom: varchar, rayon: entier, masse: entier)
```

B

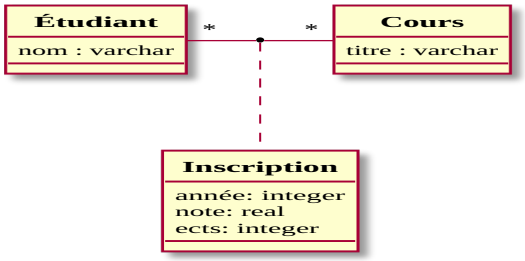
```
Secteur(#code: varchar, nom: varchar)
SystemeStellaire(#id_systeme_stellaire: entier, nom: varchar, type_spectral: {0, B, F, G, K, M})
Astre(#id_astre: entier, nom: varchar, rayon: entier, masse: entier)
```

C

```
Secteur(#code: varchar, nom: varchar)
SystemeStellaire(#id: entier, nom: varchar, type_spectral: {0, B, F, G, K, M}, code=>Secteur.code)
avec code non null et au moins un astre référant
Astre(#id: entier, nom: varchar, rayon: entier, masse: entier, systeme=>SystemeStellaire.id)
avec systeme non null
```

Exercice

Quel est le schéma relationnel adapté pour ce diagramme de classes UML ?



A

```
Etudiant(#id_etudiant:entier, Cours(#id_cours:entier, Inscription(#id_cour
nom:varchar,  annee:  entier, titre:varchar)          #id_etudiant=>Etudia
note: entier, ects: entier)
```

B

```
Etudiant(#id_etudiant:entier, Cours(#id_cours:entier, Inscription(#id_cour
nom:varchar)                  titre:varchar)          #id_etudiant=>Etudia
                                                      annee: entier, note
                                                      ects: entier)
```

C

```
Etudiant(#id_etudiant:entier, Cours(#id_cours:entier, Inscription(#id_cour
nom:varchar)                  titre:varchar,   annee: #id_etudiant=>Etudia
                              entier,  note:  entier,
                              ects: entier)
```

Solutions des exercices

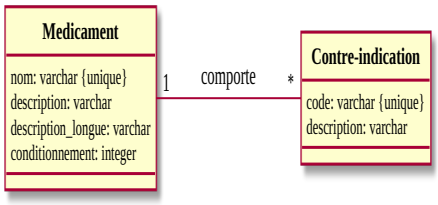
Solution n°1

[exercice p. 6]

```
1 Version (majeure:integer, mineure:integer, revision:integer, release_name:varchar)
```

Solution n°2

[exercice p. 9]



Solution n°3

[exercice p. 9]

Medicament

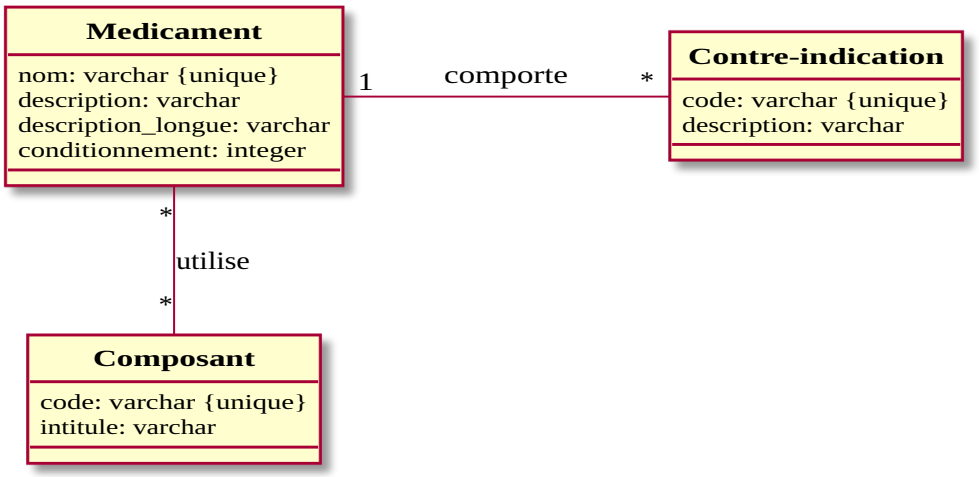
#nom	*description*	*desc_longue*	*cond.*
Chourix	Médicament contre la chute des choux	Vivamus...	11
Tropas	Médicament contre les dys...	Suspendisse...	42

Contre_indication

#code	*description*	*medicament=>Medicament*
CI1	Ne jamais prendre après minuit	Chourix
CI2	Ne jamais mettre en contact avec l'eau.	Chourix
CI3	Garder à l'abri de la lumière du soleil	Tropas

Solution n°4

[exercice p. 12]



Solution n°5

[exercice p. 12]

```
Medicament (#nom:varchar, description:varchar, description_longue:varchar,
conditionnement:number)
Contre_indication (#code:varchar, description:varchar, medicament=>Medicament)
Composant (#code:varchar, intitule:varchar)
Composition (#medicament=>Medicament, #composant=>Composant)
```

Contraintes liées à la cardinalité

Complément

```
Contre_indication.medicament non null
```

Solution n°6

[exercice p. 12]

Composant

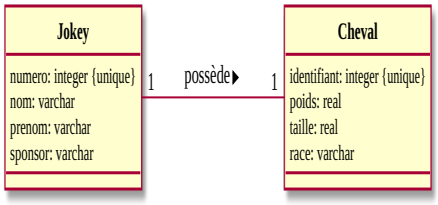
#code	*intitule*
HG79	Vif-argent allégé
HG81	Vif-argent alourdi
SN50	Pur étain

Composition

#medicament=>Medicament	*#composant=>Composant*
Chourix	HG79
Chourix	SN50
Tropas	HG79

Solution n°7

[exercice p. 15]



Solution n°8

[exercice p. 15]

En référençant le jockey par son cheval.

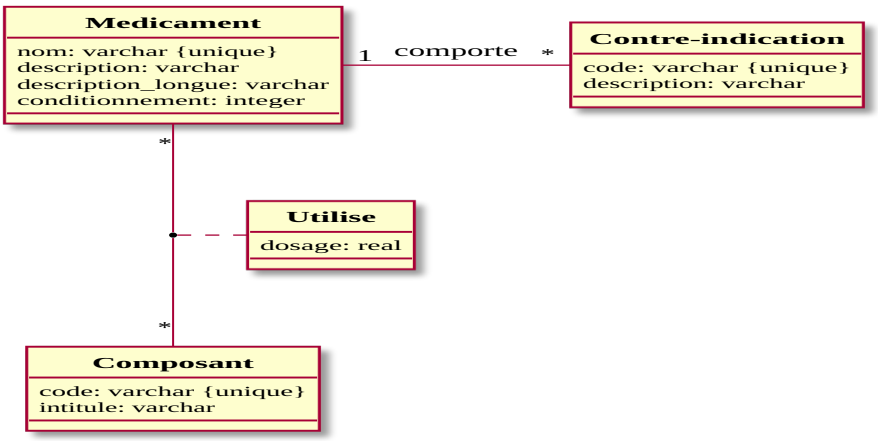
```
Jockey(#numero:integer, nom:varchar, prenom:varchar, sponsor:varchar)
Cheval(#identifiant:integer, poids:real, taille:real, race:varchar, jockey=>Jokey)
avec jocket UNIQUE NOT NULL
```

En référençant le cheval par son jockey.

```
Jockey(#numero:integer, nom:varchar, prenom:varchar, sponsor:varchar,
cheval=>Cheval)) avec cheval UNIQUE NOT NULL
Cheval(#identifiant:integer, poids:real, taille:real, race:varchar)
```

Solution n°9

[exercice p. 18]



Solution n°10

[exercice p. 18]

```
Medicament(#nom:varchar, description:varchar, description_longue:varchar,
conditionnement:integer)

Contre_indication(#code:varchar, description:varchar, medicament=>Medicament) avec
medicament non null

Composant(#code:varchar, intitule:varchar)

Utilise(#medicament=>Medicament, #composant=>Composant, dosage:real)
```

Solution n°11

[exercice p. 20]

Lorsque l'on a dans le diagramme de conception UML une classe d'association avec une cardinalité "`0..N:0..N`" on doit :

A

☐ Mettre en place une relation qui correspond à cette association au sein de laquelle on référence les deux relations associées grâce à deux clefs étrangères.

B

☐ Intégrer les attributs de la classe d'association dans la relation créée pour transformer la classe d'association.

Solution n°12

[exercice p. 20]

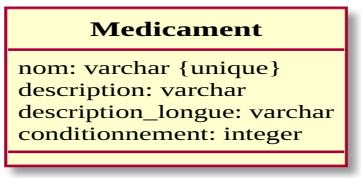
Association « *1:N* »	Association « *N:M* »	Association « *1:1* »
Ajouter une clé étrangère à la relation côté N	Ajouter une table contenant deux clés étrangères	Ajouter une clé étrangère à l'une des deux relations et une contrainte d'unicité sur cette clé

Solution n°13

[exercice p. 20]

Exercice

Quel est le schéma relationnel adapté pour ce diagramme de classes UML ?



A

```
Medicament(nom: varchar, description: varchar, description_longue:
varchar, conditionnement: entier)
```

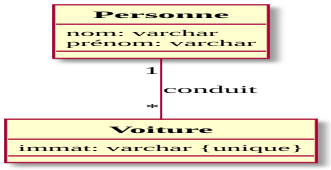
B

```
Medicament(#nom: varchar, description: varchar,
description_longue: varchar, conditionnement: entier)
```

Il faut veiller à indiquer les clés en général et la clé primaire en particulier.

Exercice

Quel est le schéma relationnel adapté pour ce diagramme de classes UML ?



A

```
Personne(#id_personne: entier, nom: text, prenom: text)
```

```
Voiture(#immat: text, text)
```

Il manque la référence traduisant l'association entre les deux classes.

B

```
Personne(#id_personne: entier, nom : text, prenom : text)
```

```
Voiture(#immat: text, personne=>Personne)
```

C

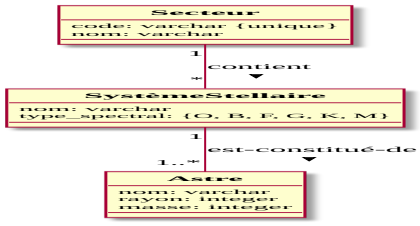
```
Personne(#id_personne: entier, nom : text, prenom : text, voiture=>Voiture)
```

```
Voiture(#immat: text, text)
```

Ici on a indiqué que chaque personne ne conduisait qu'une voiture et qu'une voiture pouvait être conduite par plusieurs personnes.

Exercice

Quel est le schéma relationnel adapté pour ce diagramme de classes UML ?



A

```
Secteur(#code: varchar, nom : varchar)
```

```
SystemeStellaire(nom: varchar, type_spectral: {0, B, F, G, K, M})
```

```
Astre(nom: varchar, rayon: entier, masse: entier)
```

Toute relation doit au moins avoir une clé ; il manque les clés étrangères.

B

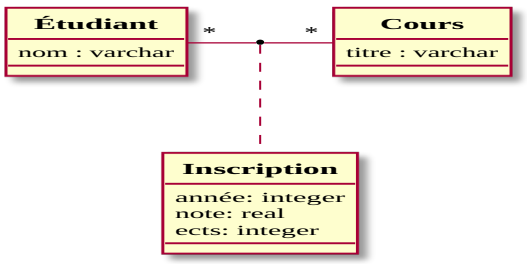
```
Secteur(#code: varchar, nom: varchar)
SystemeStellaire(#id_systeme_stellaire: entier, nom: varchar, type_spectral: {0, B, F, G, K, M})
Astre(#id_astre: Il
entier, nom: les
varchar, rayon: étra
entier, masse:
entier)
```

C

```
Secteur(#code: varchar, nom: varchar)
SystemeStellaire(#id: entier, nom: varchar, type_spectral: {0, B, F, G, K, M}, code=>Secteur.code) avec code non null et au moins un astre référant
Astre(#id: entier, nom: varchar, rayon: entier, masse: entier, systeme=>SystemeStellaire.id) avec systeme non null
```

Exercice

Quel est le schéma relationnel adapté pour ce diagramme de classes UML ?



A

```
Etudiant(#id_etudiant:entier, nom:varchar, annee: entier, note: entier, ects: entier)
Cours(#id_cours:entier, titre:varchar)
Inscription(#id_cours
#id_etudiant=>Etudia
```

B

```
Etudiant(#id_etudiant:entier, nom:varchar)
Cours(#id_cours:entier, titre:varchar)
Inscription(#id_cours
#id_etudiant=>Etudia
annee: entier, note:
ects: entier)
```

C

```
Etudiant(#id_etudiant:entier, nom:varchar)
Cours(#id_cours:entier, titre:varchar, annee: entier, note: entier, ects: entier)
Inscription(#id_cours
#id_etudiant=>Etudia
```

Index

Crédits des ressources

p. 4
Attribution - Partage dans les Mêmes Conditions - Stéphane Crozat

p. 7
Attribution - Partage dans les Mêmes Conditions - Stéphane Crozat

p. 10
Attribution - Partage dans les Mêmes Conditions - Stéphane Crozat

p. 14
Attribution - Partage dans les Mêmes Conditions - Stéphane Crozat

p. 16
Attribution - Partage dans les Mêmes Conditions - Stéphane Crozat