

Le modèle relationnel


- 1 Introduction
- 2 Le modèle relationnel, vocabulaire et concepts
- 3 Contraintes d'intégrité
 - Contraintes de domaine
 - Contraintes d'entité
 - Contraintes de référence
 - Contraintes utilisateurs
- 4 Exercices

- 1 Introduction
- 2 Le modèle relationnel, vocabulaire et concepts
- 3 Contraintes d'intégrité
 - Contraintes de domaine
 - Contraintes d'entité
 - Contraintes de référence
 - Contraintes utilisateurs
- 4 Exercices

Introduction

- De nombreuses activités reposent de nos jours sur la gestion d'informations. Citons par exemple, en vrac :
 - ▶ la gestion d'une médiathèque (livres, usagers, emprunts,...) ;
 - ▶ la gestion d'un forum (utilisateurs, posts, rubriques,...)
 - ▶ la gestion des informations échangées sur les réseaux sociaux ;
 - ▶ la comptabilité d'une entreprise (salariés, fournisseurs, clients, ventes, achats,...) et plus généralement la gestion des finances (banques, établissements de prêt,...) ;
 - ▶ la gestion de certains services de l'administration publique (impôts : contribuables, tranches d'imposition, revenus,... ; pôle-emploi : demandeurs d'emploi, offres d'emploi, formations,...) ;
 - ▶ les compagnies d'assurance (assurés, contrats souscrits, biens/personnes assurés, sinistres, clients d'autres compagnies,...) ;
 - ▶ tous les domaines ayant un lien avec l'intelligence artificielle et le domaine du «big-data» (reconnaissance de la parole, traduction de textes, diagnostic médical assisté par algorithmes de traitement d'images,...).

- Le volume très important des informations à gérer (de l'ordre, désormais, du pétaoctet, 10^{15} octets, soit un million de milliards d'octets) impose l'utilisation d'une solution informatique.
- Pour aboutir à une solution logicielle, une phase préliminaire cruciale consiste à correctement *modéliser* la situation et le problème à résoudre de façon abstraite (avant de songer à une implémentation concrète, dans un langage précis)¹.

1. De même qu'il est recommandé, lors de la conception d'une solution algorithmique sur un problème dont la solution n'est pas immédiate, de commencer par mettre au propre, sur papier, les structures algorithmiques susceptibles de résoudre le problème, avant de proprement coder dans un langage précis. 

- Une telle modélisation est caractérisée par :
 - ① des entités porteuses d'informations génériques (livres, usagers, emprunts dans l'exemple d'une médiathèque) qui doivent être stockées : *persistance* de l'information ;
 - ② le fait qu'elle modélise le réel, c'est à dire ne considère qu'un aspect (pertinent) des entités (objets/personnes) entrant en jeu dans la situation (on ne s'intéressera pas, par exemple, à la taille des usagers de la médiathèque) ;
 - ③ une *logique* propre reliant les informations d'une entité à celles d'une autre (exemple : un même usager de la médiathèque peut emprunter plusieurs livres, mais un même livre ne peut être emprunté par plusieurs usagers) ;
 - ④ des actions/transactions possibles, faisant évoluer l'état du système d'informations, tout en respectant la logique précédente (exemple : un livre non-emprunté peut devenir emprunté, pourvu que l'utilisateur n'ait pas dépassé le nombre maximum de livres empruntables).
- Quand une solution logicielle permettant de gérer les informations en jeu existe, on parle de *Système de Gestion De Bases de Données*, ou SGBD.

- 1 Introduction
- 2 Le modèle relationnel, vocabulaire et concepts
- 3 Contraintes d'intégrité
 - Contraintes de domaine
 - Contraintes d'entité
 - Contraintes de référence
 - Contraintes utilisateurs
- 4 Exercices

Le modèle relationnel, vocabulaire et concepts

- Historiquement, le *modèle relationnel* (qui est un modèle de gestion des systèmes d'informations, parmi d'autres) a été introduit en 1970 par Edgar Codd, alors employé d'IBM². Il fonde, dans un article mathématique, la portée du langage de requête qui sera à la base du langage SQL, sur lequel sont construits la plupart des SGBD actuels.
- Dans ce modèle relationnel, un objet, appelé *entité*, est représenté par un n -uplet de valeurs scalaires (i.e. chacune représentable par un nombre), appelées *attributs*.
- L'ensemble des n -uplets des entités de même nature est appelé une *relation*.

2. IBM (International Business Machines) est une des entreprises ayant marqué l'histoire de l'informatique (et jouant encore aujourd'hui un rôle). On lui doit entre autre la commercialisation d'un des premiers PC, en 1981, qui a inauguré l'apparition de la micro-informatique.

- Exemple, pour une médiathèque, qui doit enregistrer des informations sur les livres. Chaque livre est une entité de la relation livre :

('La programmation en pratique', 'B. Kernighan et R. Pike', 'Vuibert', 2017, '978-2711786701')

- Chaque entité livre a cinq attributs : titre, auteur, éditeur, année de publication, isbn.
- La relation livre est l'ensemble de ces quintuplets.

```

livre={
('La programmation en pratique', 'B. Kernighan et R. Pike',
'Vuibert', 2017, '978-2711786701'),
('The art of computer programming, vol. 1', 'D. Knuth',
'Addison-Wesley', 1997, '978-0321635747')
('Gödel, Escher, Bach: les brins d'une guirlande
éternelle', 'D. Hofstadter', 'Dunod', 1985,
'978-2-10-052306-1'),
('Apprendre à programmer avec Python 3', 'G. Swinnen',
'Eyrolles', 2012, '978-2-212-13434-6'),
('L'hyperpuissance de l'informatique', 'G. Berry', 'Odile
Jacob', 2017, '978-2-7381-3953-5'),
('Algorithms, 4th Ed.', 'R. Sedgewick and K. Wayne',
'Addison Wesley', 2012, '978-0-321-57351-3'),
('Histoire illustrée de l'informatique', 'E. Lazard et
P. Mounier-Kuhn', 'EDP Sciences', 2016, '978-2-7598-1819-8'),
...}

```

- Une relation est définie par un *schéma*, qui précise le nom, le type et l'ordre des attributs de chaque entité.
- Exemple de la relation livre :

livre (*titre* :String,*auteur* :String,*éditeur* :String,*année* :int,*isbn* :String)

- Une *Base de Données* (en abrégé BdD) est un ensemble de relations.
- La BdD de la médiathèque peut être définie par trois relations, chacune définie par un schéma :

livre (*titre* :String,*auteur* :String,*éditeur* :String,*année* :int,*isbn* :String)

usager (*nom* :String,*prénom* :String,*code_barre* :String)

emprunt (*titre* :String,*nom* :String,*prénom* :String,*retour* :String)

- En plus de la structure propre de chaque relation (attributs liés aux différentes entités), la modélisation relationnelle doit absolument définir les *contraintes d'intégrité* de la BdD, c'est à dire l'ensemble des *propriétés logiques* que les données doivent vérifier. Ces contraintes garantissent la *cohérence* des données.

- 1 Introduction
- 2 Le modèle relationnel, vocabulaire et concepts
- 3 Contraintes d'intégrité
 - Contraintes de domaine
 - Contraintes d'entité
 - Contraintes de référence
 - Contraintes utilisateurs
- 4 Exercices

Contraintes d'intégrité

On distingue 4 types de contraintes d'intégrité :

- contraintes de domaine ;
- contraintes d'entité ;
- contraintes de référence ;
- contraintes utilisateurs.

Contraintes de domaines

- Les contraintes de domaine doivent permettre :
 - ▶ de représenter exactement chaque attribut ;
 - ▶ de limiter (au maximum) la saisie de valeurs illégales.
- En pratique : ces contraintes seront traduites par des types de base, tels ceux utilisés dans les langages de programmation (cf. les types `String`, `int`, utilisés précédemment).

Contraintes d'entité

- Elle permettent d'assurer que chaque entité d'une relation est *unique*.
- Pour cela, on doit déterminer, dans l'ensemble des attributs d'une entité, un (ou des) attribut(s) qui identifie(nt) l'unique entité ayant ces informations.
- Cet (ou ces) attribut(s) est (sont) appelé(s) la *clef primaire* de la relation.
- Exemple 1 : un livre est identifié de façon unique par son isbn, qui peut être choisi comme clef primaire. Ce n'est pas le cas :
 - ▶ de l'auteur : un même auteur peut avoir écrit plusieurs livres ;
 - ▶ du titre : plusieurs éditeurs peuvent avoir édité le même livre.

- Exemple 2 : dans une BdD qui stocke des points colorés dans un plan (muni d'un repère cartésien), les informations pour chaque point sont :

$$(x, y, r, g, b)$$

où (x, y) sont les coordonnées du point et (r, g, b) sont les composantes en rouge, vert et bleu de la couleur du point. Une clef primaire possible est le couple (x, y) .

- Exemple 3 : un usager de la médiathèque ne peut être identifié par ses seuls nom et prénom, pour les problèmes d'homonymie. Il peut en revanche être identifié par le code-barre de sa carte d'utilisateur de la médiathèque.

Contraintes de référence

- Lorsqu'un attribut d'une relation est la clef primaire d'une autre relation, on dit que c'est une *clef étrangère* de cette relation.
- La contrainte induite par cette propriété de clef étrangère est que l'entité référencée par la clef étrangère doit exister dans la BdD.
- Semblablement, on ne peut détruire une entité qui contient une valeur de clef primaire référencée par une clef étrangère de même valeur (référençant cette clef primaire) dans une autre relation.
- Reprenons la relation emprunt précédemment présentée (pour la médiathèque). Elle n'est pas réaliste, pour les problèmes d'homonymie de l'utilisateur, ou de livres ayant le même titre. Une version qui corrige ces défauts est :

```
emprunt(code_barre : String, isbn : String, retour : String)
```

- Pour cette nouvelle relation emprunt :
 - ▶ code_barre et isbn sont des clefs étrangères, faisant respectivement référence à une clef primaire des relations usager et livre ;
 - ▶ isbn est une clef primaire.
- Les contraintes liées à ces clefs étrangères assurent que :
 - ▶ le livre emprunté existe dans la médiathèque ;
 - ▶ son emprunteur est un usager de la médiathèque ;
 - ▶ on ne pourra supprimer un usager qui n'a pas rendu ses livres ;
 - ▶ on ne pourra supprimer un livre (par exemple abîmé) qui est emprunté.
- Bien voir qu'une même valeur de clef étrangère peut exister dans plusieurs entités d'une relation (par exemple un même usager peut emprunter plusieurs livres).

Contraintes utilisateurs

- Ce sont les contraintes qui, en fonction de la BdD et du contexte «métier» qui lui est propre, expriment plus précisément les contraintes de domaine.
- Exemples :
 - ▶ Un âge (de personne) est nécessairement un entier positif inférieur à 200 ;
 - ▶ une adresse mail doit obligatoirement contenir le caractère «@».
- On verra que le langage SQL permet d'exprimer ces contraintes.

- 1 Introduction
- 2 Le modèle relationnel, vocabulaire et concepts
- 3 Contraintes d'intégrité
 - Contraintes de domaine
 - Contraintes d'entité
 - Contraintes de référence
 - Contraintes utilisateurs
- 4 Exercices

Exercices

Ex. 1 Proposer un modèle relationnel permettant de gérer un annuaire téléphonique. On pourra admettre que :

- les personnes n'ont pas d'homonymes (à condition de connaître leur nom et prénom) ;
- un numéro de téléphone est une chaîne de caractères ;
- un numéro de téléphone n'est attribué qu'à une seule personne.

Penser à préciser les éventuelles clefs primaires et étrangères.

Ex. 2 Proposer un modèle relationnel permettant de gérer des bulletins scolaires d'élèves. On supposera que :

- chaque élève a un identifiant numérique (entier) unique ;
- chaque élève suit un ensemble de matières qui lui est propre dans une liste finie de matières ;
- chaque élève a au plus une note par matière, comprise entre 0 et 20.

Penser à préciser toutes les contraintes du modèle.

Ex. 3 On considère la solution donnée pour l'exercice 1. Dire si chacun des ensembles suivants est une relation valide pour le schéma Annuaire.

- 1 $\{\}$
- 2 $\{('0123456789', 'Titi', 'Toto')\}$
- 3 $\{('0123456789', 'Titi', 'Toto'), ('0123456789', 'Doe', 'John')\}$
- 4 $\{('0123456789', 'Titi', 'Toto'), ('9876543210', 'Titi', 'Toto')\}$
- 5 $\{('0123456789', 'Titi', 'Toto'), ('Doe', 'John')\}$
- 6 $\{(42, 'Titi', 'Toto')\}$

Ex. 4 On considère la solution donnée pour l'exercice 2. Dire si chacun des ensembles suivants est une relation valide pour le schéma de la base de données du bulletin de notes.

- 1 Eleve = {}
Matiere = {}
Note = {}
- 2 Eleve = {'AB56789', 'Titi', 'Toto'}
Matiere = {(0, 'NSI'), (1, 'Sport')}
Note = {'AB56789', 1, 17}
- 3 Eleve = {'AB56789', 'Titi', 'Toto'}
Matiere = {(0, 'NSI')}
Note = {'AB56789', 1, 17}
- 4 Eleve = {'AB56789', 'Titi', 'Toto'}
Matiere = {(0, 'NSI')}
Note = {'AB56789', 0, 17}, ('AB56789', 0, 18)}
- 5 Eleve = {'AB56789', 'Titi', 'Toto'}
Matiere = {(0, 'NSI'), (1, 'Sport')}
Note = {'AB56789', 0, 17}, ('AB56789', 1, 17)}

Ex. 5 Modéliser des informations sur les départements français. Pour chaque département on veut pouvoir stocker son nom, son code, son chef-lieu et la liste de tous les départements voisins. Attention, les codes de département sont tous des nombres, sauf la Corse du Sud et la Haute Corse qui ont les codes 2A et 2B respectivement. Les départements d'Outre-Mer ont un code sur trois chiffres (de 971 à 976). Proposer une contrainte utilisateur permettant d'éviter la redondance d'information dans la liste des voisins.

Ex. 6 Proposer une modélisation pour un réseau de bus. Cette dernière doit être suffisamment riche pour permettre de générer, pour chaque arrêt de bus du réseau, une fiche horaire avec tous les horaires de passage de toutes les lignes de bus qui desservent l'arrêt.

Indication : ici, plus qu'une traduction du français vers le modèle relationnel, on essaiera de déterminer dans un premier temps quelles informations sont pertinentes et comment les représenter. On pourra ensuite procéder à la modélisation sous forme de relations.

Ex. 7 On considère deux relations $R(a \text{ int}, b \text{ int}, c \text{ int})$ et $S(a \text{ int}, e \text{ int})$ où a est une clef primaire de R , (a, e) est une clef primaire de S et a est une clef étrangère de S faisant référence à la clef a de R . Dire si les affirmations suivantes sont vraies ou fausses, en justifiant.

- 1 Les a de R sont tous deux à deux distincts.
- 2 Les b de R sont tous deux à deux distincts.
- 3 Les a de S sont tous deux à deux distincts.
- 4 Les e de S sont tous deux à deux distincts.
- 5 S peut être vide alors que R est non vide.
- 6 R peut être vide alors que S est non vide.