Licences Mathématiques et Informatique
3ème année - Formation Initiale et par Apprentissage

Bases de données relationnelles

Polycopié de cours - Introduction

Maude Manouvrier

Table des matières

Chapitre 2

Modèle relationnel

Le **modèle relationnel** de données a été défini en 1970 par E.F. Codd [1] et les premiers SGBD commerciaux sont apparus au début des années 80. Le modèle relationnel représente l'information dans une collection de **relation**s. Intuitivement, on peut voir une relation comme une **table** à double entrée. Chaque ligne de la table, appelée **nuplet** (ou *tuple* en anglais), peut être vue comme un fait décrivant une entité du monde. Une **colonne** de la table est appelée un attribut.

2.1 Notions fondamentales

2.1.1 Domaine

Un **domaine** est un ensemble de valeurs caractérisé par un nom. Un domaine peut être défini en extension, en donnant une liste de valeurs, ou en intention, en définissant une propriété caractéristique des valeurs du domaine. Les domaines `ENTIER`, `REEL`, `BOOLEAN` et `CARACTERES` sont des domaines définis en intention. L'ensemble fini de chaînes de caractères *{'Professeur', 'MCF', 'ATER', 'Moniteur', 'Vacataire', 'PRAG', 'PAST'}* est un domaine défini en extension.

2.1.2 relation

Une **relation** est un sous-ensemble du produit cartésien d'une liste de domaines caractérisé par un nom unique. Une représentation d'une relation est sous forme de table à deux dimensions. Chaque colonne correspond à un domaine du produit cartésien, un même domaine pouvant apparaître plusieurs fois.

TABLE 2.1 – Un exemple de relation : la relation *Department*.

Department_ID	**Department_Name**
1	Informatique
2	Mathématique
...	...

2.1.3 Attribut

Un **attribut** est une colonne d'une relation. Il est caractérisé par un nom et dont les valeurs appartiennent à un domaine. Les noms d'attributs d'une relation sont tous différents, mais plusieurs relations peuvent avoir des attributs de même nom. **Attention :** il est préférable que les attributs de même nom soient de même domaine (même sémantique). La valeur des attributs en base de données relationnelles est atomique (une seule valeur par attribut et par nuplet).

1. cf. `https://fr.wikipedia.org/wiki/Edgar_Frank_Codd`

2.1.4 Nuplet ou tuple

Un **nuplet** est une ligne d'une relation correspondant à un enregistrement. Les nuplets d'une relation sont tous différents. On peut donc toujours distinguer deux nuplets.

2.2 Schéma et instance

On distingue le **schéma de la base de données**, qui correspond à une modélisation logique de la base à l'aide du modèle relationnel (c'est un ensemble de **schémas de relation**), de l'**instance de la base de données**, qui correspond aux nuplets (i.e. aux valeurs) contenus dans la base à un instant donné. Un schéma de relation correspond à la liste des attributs de la relation et de leur domaines.

Le schéma de la relation *Department* est : `Department(Department_ID: ENTIER; Department_NAME: CARACTERES)`, l'instance est :

TABLE 2.2 – L'instance de la relation *Department*.

1	Informatique
2	Mathématique
...	...

Par convention, on note R le schéma d'une relation et r une instance de R. Lorsque R contient un ensemble d'attribut K et que t est un nuplet de r, on note $t.K$ la valeur de l'ensemble d'attributs K pour le nuplet t.

2.3 Règle d'intégrité structurelle

Les règles d'intégrité structurelle sont les assertions qui doivent être vérifiées par les données contenues dans la base [9].

2.3.1 Unicité des clés

Une relation est par définition un ensemble de nuplets sans doublon. Afin d'identifier les différents nuplets, on utilise la notion de **clé**. Une clé est un ensemble d'attributs dont la connaissance des valeurs permet d'identifier un nuplet unique de la relation considérée :

La relation R a pour clé K si $\forall t_1, t_2$ tuples d'une instance de R : $t_1.K \neq t_2.K$

Toute relation possède au moins un clé. S'il en existe plusieurs, on en choisit une (nous verrons plus tard comment) qui est appelée **clé primaire**.

Une clé est **minimale** si lorsqu'on enlève un attribut alors ce n'est plus une clé :

La relation R a pour clé minimale K si $\nexists K' \subset K$ tel que K' est une clé

Une clé primaire est toujours minimale.

2.3.2 Contraintes de références

Soit la relation `Location(Auto_Id, Immatriculation, Date)` associant des identifiants d ?automobilistes (dont les informations sont stockées dans une relation `Automobiliste`), des immatriculations (dont les informations sont stockées dans une relation `Voiture`) et des dates de location.

Pour maintenir les liens entre des informations liées, on utilise la notion de **contrainte référentielle**. Une contrainte référentielle est une contrainte d'intégrité portant sur une relation R_1 qui consiste à imposer que la valeur d'un groupe d'attributs apparaissent comme valeur de clé dans une autre relation. La représentation de contraintes référentielles peut s'effectuer par la définition de **clés étrangères** dans une relation : une clé étrangère est un groupe d'attributs qui doit apparaître comme clé primaire dans une autre relation. Par exemple la relation `Location(Auto_Id, Immatriculation, Date)` a 2 clés étrangères : `Auto_Id` qui fait référence à la clé primaire de la relation `Automobiliste` et `Immatriculation` qui fait référence à la clé primaire de la relation `Voiture`.

Les contraintes d'intégrité référentielles définissent des liens obligatoires entre les relations. Ce sont des contraintes très fortes qui conditionnent les mises à jour. Lors de l'insertion d'un nuplet dans une relation référençant une autre relation par une clé étrangère, il faut vérifier que les valeurs de clés étrangères existent bien dans les relations référencées (ex. que l'on insère bien une location pour un automobiliste dont les informations sont stockées dans la relation `Automobiliste` et une voiture dont les informations sont stockées dans la relation `Voiture`). Dans le cas où ces valeurs n'existent pas, l'insertion est refusée pour cause de **violation d'intégrité**. De même, lors de la suppression de nuplets dans une relation référencée, il faut vérifier qu'aucun nuplet n'existe dans chaque relation référençante. Si un nuplet existe, le système doit refuser la suppression ou effectuer des suppressions en cascade (c-à-d supprimer les nuplets référençant).

2.3.3 Valeurs nulles et clés

La **valeur nulle** est une valeur conventionnelle introduite dans une relation pour représenter une information inconnue ou inapplicable. Tout attribut peut prendre une valeur nulle excepté les attributs de la clé primaire. On introduit pour cela une **contrainte d'entité** qui impose que toute relation possède une clé primaire et que tout attribut de cette clé est non nul.

2.3.4 Contraintes de domaines

Une contrainte de domaine est une contrainte d'intégrité qui impose qu'une colonne d'une relation doit comporter des valeurs vérifiant une assertion logique (ex. le grade d'un enseignant est une chaîne de caractères dont les valeurs appartiennent au domaine *{'Professeur', 'MCF', 'ATER', 'Moniteur', 'Vacataire', 'PRAG', 'PAST'}*).

2.4 Langages d'interrogation

Plusieurs types de langages d'interrogation (ou langage de requêtes) ont été proposés. Ils s'appuient sur plusieurs théories [10] :

- L'**algèbre relationnelle** qui indique comment le SGBD calcule le résultat d'une requête d'interrogation.
- Le **calcul relationnel** qui décrit sous forme logique du premier ordre le résultat que l'on souhaite obtenir pour la requête. Ce langage est à l'origine de la syntaxe du SQL.
- Le standard **SQL** qui décrit sous forme logique le résultat que l'on souhaite obtenir pour la requête.

Il a été démontré que ces langages étaient équivalents dans le sens où ils permettent de désigner les mêmes ensembles de données. Connaître l'algèbre relationnelle permet de ne pas exprimer trop mal ses requêtes en SQL, afin que le SGBD puissent les calculer le plus rapidement possible.

Bibliographie

[1] D. Austin, *Using Oracle8^{TM}*, Simple Solutions - Essential Skills, QUE, 1998, ISBN : 0-7897-1653-4

[2] R. Chapuis, *Oracle 8*, Editions Dunes et Laser, 1998, ISBN : 2-913010-07-5

[3] P. Chen, *The Entity-Relationship Model–Toward a Unified View of Data*, ACM Transactions on Database Systems, Vol. 1, No. 1, March 1976, Pages 9-36, http ://www.csc.lsu.edu/∼chen/pdf/erd.pdf

[4] T. Connolly, C. Begg et A. Strachan, *Database Systems - A pratical Approach to Design, Implementation and Management*, Addison-Wesley, 1998, ISBN : 0-201-34287-1, disponible à la BU 055.7 CON

[5] C.J. Date, *Introduction aux bases de données*, 6ème édition, Thomson Publishing, 1998, ISBN : 2-84180-964-1, disponible à la BU 005.7 DAT

[6] R. Elamsri et S.B. Navathe, *Fundamentals of Database Systems*, 3ème édition, Addison Wesley- disponible à la BU 005.7 ELM

[7] P. Delmal, *SQL2 - Application à Oracle, Access et RDB*, 2ème édition, Bibliothèque des Universités - Informatique, De Boeck Université, 1988, ISBN : 2-8041-2995-0,disponible à la BU 005.74 SQL

[8] S. Feuerstein, B. Pribyl et C. Dawes, *Oracle PL/SQL - précis et concis*, O'Reilly, 2000, ISBN : 2-84177-108-3

[9] G. Gardarin, *Bases de Données - objet & relationnel*, Eyrolles, 1999, ISBN : 2-212-09060-9, disponible à la BU 005.74 GAR

[10] R. Grin, *Introduction aux bases de données, modèle relationnel*, Université Sophia-Antipolis, jan. 2000

[11] R. Grin, *Langage SQL*, Université Sophia-Antipolis, jan. 2000

[12] G. Gardarin et O. Gardarin *Le Client-Serveur*, Eyrolles, 1999, disponible à la BU

[13] H. Garcia-Molina, J.D. Ulmann et J. Widow, *Database SYstem Implementation*, Prentice Hall, 2000, ISBN :0-13-040264-8, disponible à la BU 005.7 GAR

[14] H. Garcia-Molina, J.D. Ulmann et J. Widow, *Database Systems - The Complete Book*, Prentice Hall, 2002, ISBN :0-13-031995-3

[15] S. Krakowiak, *Gestion Physique des données*, Ecole Thématique "Jeunes Chercheurs" en Base de Données, Volume 2, Port Camargue, mars 1997

[16] D. Lockman, *Oracle8^{TM} Dévéloppement de bases de données*, Le programmeur - Formation en 21 jours, Editions Simon et Schuster Macmillan (S&SM), 1997, ISBN : 2-7440-0350-6, disponible à la BU 005.74 ORA

[17] P.J. Pratt, *Initiation à SQL - Cours et exercices corrigés*, Eyrolles, 2001, ISBN : 2-212-09285-7

[18] R. Ramakrishnan et J. Gehrke, *Database Management Systems*, Second Edition ; McGraw-Hill, 2000, ISBN : 0-07-232206-3, disponible à la BU 055.7 RAM

[19] A. Silberschatz, H.F. Korth et S. Sudarshan, *Database System Concepts*, Third Edition, McGraw-Hill, 1996, ISBN : 0-07-114810-8, disponible à la BU 005.7 DAT

[20] C. Soutou, *De UML à SQL - Conception de bases de données*, Eyrolles, 2002, ISBN : 2-212-11098-7

[21] J.D. Ullman et J. Widom, *A first Course in Database Systems*, Prentice Hall, 1997, ISBN : 0-13-887647-9, disponible à la BU 005.7 ULL