

Chapitre 1 : Introduction au modèle relationnel objet

1. Introduction

Au cours des 50 dernières années, grâce à sa puissance et sa fiabilité, le modèle relationnel a connu un très grand succès et s'avère très adéquat aux applications traditionnelles des bases de données (gestion).

2. Le modèle relationnel

Le modèle relationnel présente beaucoup d'atouts mais aussi des inconvénients.

2.1 Les atouts du modèle relationnel

- a) Fondé sur une théorie rigoureuse et des principes simples
- b) Mature, fiable, performant, connu, plus utilisé et maîtrisé
- c) Une implémentation standard du modèle relationnel, avec des API (*Application programming Interface*) pour permettre un accès aux bases de données avec la plupart des langages de programmation (Exemple : JDBC est l'acronyme de *Java DataBase Connectivity* et SQLAPI pour C++, etc.)
- d) Les SGBDR incluent des outils performants de gestion de requêtes, de générateurs d'applications, d'administration, d'optimisation, etc.

Seulement avec l'avènement de l'informatique industrielle et ses nouveaux besoins et exigences en structures de données complexes, le modèle relationnel s'avère très vite moins adapté à ces nouvelles applications telles que :

- a) CAD/CAM (Computer-Aided Design/Computer-Aided Manufacturing)
- b) BD d'images et de graphiques
- c) BD géographiques (SIG : Systèmes d'Information Géographiques)
- d) BD multimédia (son, image, texte, etc. combinés)

2.2 Les inconvénients du modèle relationnel

- a) La première forme normale est inappropriée à la modélisation d'objets complexes.
- b) La normalisation entraîne la genèse de structures de données complexes et très fragmentées, qui peuvent notamment poser des problèmes de performance ou d'évolutivité causés par l'usage des jointures.
- c) Le SQL doit toujours être combiné à d'autres langages de programmation pour être effectivement mis en œuvre.
- d) La notion de méthode ne peut être intégrée au modèle logique, elle doit être gérée au niveau de l'implémentation physique.
- e) Les types de données disponibles sont limités et non extensibles.

3. Le modèle Relationnel objet

Le modèle objet-relationnel (OR) reprend le modèle relationnel en ajoutant quelques concepts de l'orienté objet (classes, méthodes, héritage, etc.) pour combler ses plus grosses lacunes.

3.1 Objectifs du modèle Relationnel-Objet

- a) Gérer des données complexes (temps, géo-référencement, multimédia, types de données utilisateurs, etc.)
- b) Pour échapper aux éclatements-jointures, l'OR réhabilite les références qui permettent d'implanter des structures complexes
- c) Les attributs multivalués (tableaux, ensembles ou listes).
- d) Définir de nouveaux domaines à structure complexe, appelés types,
- e) Associer à chaque type des méthodes
- f) Créer des hiérarchies de types
- g) Créer des objets qui sont composés d'une valeur structurée et d'un OID
- h) Etablir des liens de composition par des attributs référence qui contiennent l'OID de l'objet composant
- i) Créer des tables contenant soit des tuples normaux (en première forme normale), soit des tuples en non première forme normale (des valeurs structurées), soit des objets.

4. Les bases du modèle Relationnel-Objet

Les principales extensions qu'on peut trouver dans le RO sont les suivantes :

- a) **Les types de données utilisateurs** : l'utilisateur peut de définir des *types utilisateur* ; cela correspond à définir un type, une classe et les méthodes associées. Des hiérarchies sont possibles.

Exemple : Considérons le type abstrait *adresse_T* dont la structure comprend un numéro de rue, un nom de rue, une ville et un code postal. La création de ce type nous permettra de créer la table Etudiant qui comporte un champ *adresse* de type *adress_T*.

```
CREATE TYPE adresse_T AS OBJECT
(nrue INT, nomrue VARCHAR(30),
ville VARCHAR(30),
codepostal INT,
MEMBER FUNCTION getVille()
RETURN varchar);
...
CREATE TABLE Etudiant (
numetu NUMBER,
nom VARCHAR(30),
adresse adresse_T) ;
```

- b) **Héritage de type** : Le modèle RO permet l'héritage de types.

- c) **Collection** : Certains attributs peuvent être multi-valués. La notion de collection permet de regrouper plusieurs objets pour former un seul attribut.

Exemple vu en cours de BDD de 2^{ème} année licence

On considère une entreprise possédant des départements éparpiés dans plusieurs régions. On note par *Departement* la relation suivante :

Departement (N°Dpt, NomDpt, {LieuDpt})

N°Dpt	NomDpt	LieuDpt
1	Recherche	Alger
		T.Ouzou
2	Administration	Oran
3	Commerce	Alger
		Annaba

N°Dpt	NomDpt
1	Recherche
2	Administration
3	Commerce

N°Dpt	LieuDpt
1	Alger
1	T.Ouzou
2	Oran
3	Alger
3	Annaba

Dans le modèle relationnel, cette relation *Departement* n'est pas en 1FN car l'attribut *LieuDpt* est multivalué. On devrait décomposer la relation *Departement* en 2 relations de 1FN :

Cependant dans le modèle RO, on garde la relation telle quelle est, mais on redéfinira ses attributs. On peut définir un type collection *listeVilles* comme une table imbriquée dont les tuples sont les lieux d'un département.

```
CREATE TYPE listeVilles AS TABLE of varchar(30) ;
```

Ce type sera utilisé comme domaine d'une colonne *LieuxDpt* dans une table *Departement*.

```
CREATE TABLE Departement
(N°Dpt NUMBER,
 NomDpt VARCHAR(50),
 LieuxDpt listeVilles);
```

d) **Référence d'objet** : Les références sont les identifiants d'objets (OID : Object Identifier), elles permettent de chaîner directement les objets entre eux sans passer par des valeurs nécessitant des jointures

e) **Imbrication de table** : La possibilité d'avoir des relations comme valeurs pour certains attributs. On parle alors de *NESTED TABLES*.

La figure ci-dessous illustre une table représentée dans le modèle relationnel-Objet, pour la représenter dans le modèle relationnel, elle devrait être éclatée en plusieurs tables par le processus de normalisation.

pknom	prenom	bureau	liste-telephones	listes-specialites
Crozat	Stéphane	centre	batiment	numero
		PG	K	256
				0687990000
				0912345678
Vincent	Antoine			0344231234
Vincent	Antoine	centre	batiment	numero
		R	C	123
				0344231235
				0687990001
		Domaine		Technologie
		BD		SGBDR
		Doc		XML
		BD		SGBDRO
		Domaine		Technologie
		IC		Ontologie
		Base de données		SGBDRO