

FONDEMENTS DES BASES DE DONNÉES

Contraintes d'Intégrité dans le Modèle Relationnel: Introduction

Équipe pédagogique BD



Objectifs

Structure du modèle relationnel (cours préc.)

- ▶ Permet de modéliser la réalité :
 - ▶ les ensembles attributs vont décrire les objets ;
 - ▶ on peut séparer les données dans différentes relations avec des noms explicites ;
- ▶ Largement insuffisant pour représenter plus finement les différents aspects des données réelles.
- ▶ L'incapacité à représenter des *méta-données* conduit à un certain nombre de problèmes.

Contraintes d'intégrité

- ▶ Un cadre pour ajouter une sémantique au modèle relationnel.
- ▶ *Propriétés supposées être satisfaites* par toutes les instances d'un schéma de bases de données.
- ▶ Ex. : le numéro de sécurité sociale comme identifiant.

Films	<i>Titre</i>	<i>Metteur</i>	<i>Acteur</i>
	Les oiseaux	Hitchcock	Hedren
	Les oiseaux	Hitchcock	Taylor
	Bladerunner	Scott	Hannah
	Apocalypse Now	Coppola	Brando

Programme	<i>Ciné</i>	<i>Salle</i>	<i>Titre</i>	<i>Friandise</i>
	Rex	1	Les oiseaux	café
	Rex	1	Les oiseaux	popcorn
	Rex	2	Bladerunner	café
	Rex	2	Bladerunner	popcorn
	ArtC	1	Les oiseaux	thé
	ArtC	1	Les oiseaux	popcorn
	Cinoche	1	Les oiseaux	Coca Cola
	Cinoche	1	Les oiseaux	vin
	Cinoche	2	Bladerunner	Coca Cola
	Cinoche	2	Bladerunner	vin

TABLE : Exemple BD

Introduction dépendances

- ▶ Le schéma en lui-même ne fait aucune restriction sur les données qui peuvent être stockées.
- ▶ Toutefois, la sémantique attendue de ce schéma peut impliquer plusieurs restrictions.

Les dépendances fonctionnelles

- ▶ Les valeurs de certains des attributs d'un tuple déterminent de façon unique, dit *fonctionnellement*, les valeurs des autres attributs du tuple.
- ▶ Un seul metteur en scène associé à chaque titre de film

Films : Titre \rightarrow Metteur

- ▶ Dans Programme, un seul titre de film est associé à un couple cinéma-salle donné

Programme : Ciné, Salle \rightarrow Titre

Les dépendances d'inclusion

- ▶ Un autre type de dépendances, appelées dépendances d'inclusion (DI) ou contraintes référentielles.
- ▶ Entre **2 relations** .
- ▶ Tout titre projeté actuellement (présent dans la relation Programme) est le titre d'un film (c'est-à-dire apparaissant dans la relation Films) :

$$\text{Programme}[\text{Titre}] \subseteq \text{Films}[\text{Titre}]$$

Un mécanisme formel pour exprimer des propriétés attendues des données stockées

Si on sait que la BD satisfait un ensemble de dépendances, alors cette information peut être utilisée :

- ▶ pour améliorer la **conception** d'un schéma ;
- ▶ pour **protéger** les données en se prémunissant contre certaines mise à jour erronées ;
- ▶ pour améliorer les **performances**.

On étudiera les dépendances

- ▶ fonctionnelles,
- ▶ d'inclusion,
- ▶ multivaluées¹.

1. Il existe une forme générale de dépendances dite dépendance de jointure que nous n'aborderons pas dans ce cours.

Conception du schéma et anomalies de mise à jour

Les dépendances fournissent des informations sur la sémantique de l'application afin que le système puisse aider l'utilisateur à choisir, parmi tous les schémas possibles, le plus approprié.

Il existe plusieurs façons, pour un schéma, de ne pas être approprié. . .

Informations incomplètes

Supposons qu'il faille insérer le titre d'un nouveau film et son metteur en scène sans encore connaître les acteurs.

- ▶ Impossible avec le schéma décrit précédemment. **Anomalie d'insertion.**
- ▶ Phénomène analogue pour les **anomalies de suppression.**

Redondance

- ▶ Le fait que *Coca Cola* puisse être trouvé au *Cinoche* est enregistré plusieurs fois.
- ▶ Supposons de plus que le directeur du *Cinoche* décide de vendre du *Pepsi Cola* au lieu du *Coca Cola*.
- ▶ Il faut modifier plusieurs tuples sinon violation de la dépendance de jointure. **Anomalie de modification possible.**
- ▶ Des anomalies d'insertion et de suppression sont également causées par la redondance.

Un meilleur schéma

- ▶ *Films* pourrait être décomposée en deux relations $M\text{-Metteur}[\text{Titre}, \text{Metteur}]$ et $M\text{-Acteur}[\text{Titre}, \text{Acteur}]$ avec $M\text{-Metteur}$ vérifiant la DF

$\text{Titre} \rightarrow \text{Metteur}$

- ▶ *Programme* pourrait être décomposée en deux relations $ST\text{-Programme}[\text{Ciné}, \text{Salle}, \text{Titre}]$ et $S\text{-Programme}[\text{Ciné}, \text{Friandise}]$, avec $ST\text{-Programme}$ vérifiant la DF

$\text{Ciné}, \text{Salle} \rightarrow \text{Titre}$

Intégrité des données

Dépendances de données lors des mises-à-jour

- ▶ Une proposition de m-à-j conduisant à une violation d'une dépendance σ est rejetée.
- ▶ Durant une transaction, la BD *peut* être dans un état inconsistant mais, à la fin de celle-ci, le système doit vérifier l'intégrité de la BD
 - ▶ Si les dépendances sont violées, alors la transaction entière est rejetée (*rollback*),
 - ▶ sinon elle est acceptée (*commit*).

Implémentation efficace et optimisation de requêtes

- ▶ La connaissance des propriétés structurelles des données stockées est utile à l'amélioration des performances d'un système pour une application particulière.
- ▶ La satisfaction des dépendances conduit à une grande variété d'alternatives pour les structures de stockage et d'accès.

Ex. : La satisfaction d'une df ou d'une dj implique qu'une relation peut être stockée physiquement sous forme décomposée.

Ex. ++ : La satisfaction d'une dépendance de clé peut être utilisée pour réduire l'espace d'indexation.

Exemple d'optimisation

- ▶ $ans(d, a) \leftarrow Films(t, d, a'), Films(t, d', a)$ renvoie les couples (d, a) , où un acteur a joue dans un film mis en scène par d . Une implémentation naïve de cette requête exige une jointure ;
- ▶ puisque *Films* satisfait *Titre* \rightarrow *Metteur*, cette requête peut être simplifiée en $ans(d, a) \leftarrow Films(t, d, a)$ **qui peut être évaluée sans jointure** ;
- ▶ en effet, lorsque $\{(t, d, a'), (t, d', a)\}$ est trouvé dans la relation *Films*, on doit avoir $d = d'$, aussi peut-on aussi bien n'utiliser que $\{(t, d, a)\}$, ce qui conduit à la requête simplifiée.

Fin du deuxième cours.