Contraintes d'intégrité

Serge Abiteboul & Pierre Senellart

INRIA & Université Paris-Sud

23 mai 2008

Les contraintes d'intégrité

Ce sont des connaissances supplémentaires sur le monde réel.

Pourquoi les utiliser ?

1. protection des données
2. performance (indexation des tables)
3. conception de schéma

C'est quoi :

- Une syntaxe pour des formules logiques (p. ex. calcul)
- Une sémantique : états de la bd $\rightarrow$ {Vrai, Faux}

$$\forall x, y, z\, [ami(x, y) \wedge ami(y, z) \Rightarrow ami(x, z)]$$
$$\forall x, y, z\, [personne(x, y, z) \Rightarrow integer(z) \wedge z \geq 0 \wedge z \leq 125]$$

En pratique

- Dépendances domaines : l'attribut taille est un réel
- Dépendances fonctionnelles
- Dépendances multivaluées
- Contraintes et triggers

Dépendances fonctionnelles

Sur une seule table $R(A_1, \ldots, A_n)$
Dépendance fonctionnelle sur R :

- $X \rightarrow Y$ (X détermine Y)
- $X, Y \subseteq \{A_1, \ldots, A_n\}$

Exemple $R = \{Nom, Adresse, SS\#, Enfants\}$

$$\{Nom\} \rightarrow \{Adresse,\ SS\#\} \quad (Nom \rightarrow Adresse\ SS\#)$$

R **satisfait** $X \rightarrow Y$, ($R \models X \rightarrow Y$) si

$$\forall u, v \in R \quad (\pi_X(u) = \pi_X(v) \Rightarrow \pi_Y(u) = \pi_Y(v))$$

Exemple

R	A	B	C	D	E	F
	0	1	2	3	5	1
	1	2	0	4	5	2
	0	1	2	3	5	1
	2	0	0	4	6	2
	2	0	0	4	7	2
	1	4	2	4	5	3

$$R \not\models A \rightarrow B$$

$$R \not\models A \rightarrow E$$

$$R \models AB \rightarrow CD$$

$$R \not\models AB \rightarrow DE$$

$$R \models ABE \rightarrow CD$$

Clés

Cas particulier des dépendances fonctionnelles
R(ABC) et A est une clé : $A \rightarrow ABC$

Note : AB est aussi une clé (on parle parfois de superclé)

Les clés représentent des réalités du monde que l'on cherche à décrire :

- Deux personnes n'ont pas le même numéro de SS
- Une personne ne peut pas être dans deux endroits à la fois
- Un vin ne peut pas être à la fois rouge et blanc

Axiomatiser l'implication

Exemple

Si $R \models AB \rightarrow CD$ alors $R \models ABE \rightarrow CD$
Si $R \models AB \rightarrow CD$, $R \models CD \rightarrow F$ alors $R \models AB \rightarrow F$

Définition : $F \Rightarrow f$ (F *implique* f) si toute relation R satisfaisant $R \models F$ satisfait aussi $R \models f$

Quelques remarques simples

$X \rightarrow A_1 \ldots A_n$ ssi $X \rightarrow A_i$ pour tout i
donc il suffit de consider $X \rightarrow A$ pour un seul attribut A

Si on veut déterminer que F implique $X \rightarrow A$

1. on démarre de $X^+ := X$
2. Pour chaque $Y \rightarrow B$ dans F, si $Y \subseteq X^+$ on rajoute B à X^+
3. Quand on ne peut plus appliquer (2), $X \rightarrow A$ si $A \in X^+$

Règles d'inférence d'Armstrong

Soit U un ensemble d'attributs

$$\begin{array}{llcl} \text{Réflexivité,} & Y \subseteq X \subseteq U & \vdash & X \rightarrow Y \\ \text{Augmentation,} & X \rightarrow Y, Z \subseteq U & \vdash & XZ \rightarrow YZ \\ \text{Transitivité,} & X \rightarrow Y, Y \rightarrow Z & \vdash & X \rightarrow Z \end{array}$$

Notation $\vdash$: **Symbole d'implication syntaxique**
L'ensembles des DFs que l'on obtient en utilisant ces règles

Théorème $F \vdash f \Leftrightarrow F \Rightarrow f$

Les axiomes – suite

Utiliser les axiomes d'Armstrong

$$F = \{A \to B, B \to C, AC \to E, EB \to F\}$$

Exercice : Trouver des f tels que $F \vdash f$
Faciliter l'emploi des axiomes :
Axiomes complémentaires

$$\begin{array}{c} X \to Y \\ X \to Z \end{array} \quad \vdash \quad X \to YZ$$

$$\begin{array}{c} X \to Y \\ WY \to Z \end{array} \quad \vdash \quad XW \to Z$$

$$\begin{array}{c} X \to Y \\ Z \subseteq Y \end{array} \quad \vdash \quad X \to Z$$

Exercice : prouver les axiomes complémentaires

Décomposition

Nom	*Enfant*	*Voiture*
toto	*titi*	*Rolls*
toto	*lulu*	*Rolls*
toto	*titi*	$2CV$
toto	*lulu*	$2CV$

Problèmes

1. redondance
2. anomalies d'insertion, de suppression, de modification

$R(XYZ)$ est *décomposable sans perte d'information* sur XY, XZ si

$$R = \pi_{XY}(R) \bowtie \pi_{XZ}(R)$$

Décomposition – suite

Théorème[1] Soit $R(XYZ)$ une relation. Si R vérifie $X \rightarrow Y$ alors la décomposition sur (XY, XZ) est s.p.i.

Théorème Soit $R(XYZ)$ une relation satisfaisant un ensemble F de DF. La décomposition sur (XY, XZ) est s.p.i. ssi

$$F \Rightarrow X \rightarrow Y \quad OU \quad F \Rightarrow X \rightarrow Z$$

[1] X,Y,Z sont disjoints

Forme normale de Boyce-Codd

But : décomposer pour éviter des anomalies
Définition : Un schéma $(R(U), F)$ est sous-FNBC

$$si \quad \begin{array}{c} F \Rightarrow X \rightarrow A \\ A \notin X \end{array} \quad \text{alors } X \text{ est une clé}$$

Décomposition sous FNBC

- Choisir une DF $X \rightarrow Y$ telle que

$$F \Rightarrow X \rightarrow Y, \quad U = XYZ, \quad X, Y, Z \neq \emptyset, \quad Y \cap Z = \emptyset$$

- Décomposer suivant XY et XZ
- Continuer à décomposer les blocs résultats

Rechercher aussi : pas de perte de dépendances

Problème avec BCNF

$AB \rightarrow C$ et $C \rightarrow B$
Exemple : Adresse, Ville, CodePostal

$C \rightarrow B$ est une violation de BCNF (C n'est pas une clé)

Décomposition sous CB et CA

Oups : on ne peut plus imposer $AB \rightarrow C$
(On ne peut plus la vérifier sur les projections sur CB et CA)

On a inventé la 3NF qui est moins contraignante que BCNF

BCNF

Résultat

- Toujours possible de passer sous FNBC
- Pas toujours sans perte des DF

Conception de schéma

On part d'un ensemble d'attributs

On spécifie les dépendances fonctionnelles

On décompose la relation jusqu'à satisfaire BCNF (ou 3NF)

Toujours plus fort : les dépendances multivaluées

Exemple : personne, enfant, voiture

toto	*lulu*	*bmw*
toto	*lulu*	*2cv*
toto	*zaza*	*bmw*
toto	*zaza*	*2cv*

Redondances et anomalies

$R(X, Y, Z)$ satisfait la dépendance multivaluée $X \twoheadrightarrow Y$ si pour tout u, v dans R avec $\pi_X(u) = \pi_X(v)$, il existe w tel que

- $\pi_{XY}(w) = \pi_{XY}(u)$
- $\pi_{XZ}(w) = \pi_{XZ}(v)$

Décomposition souhaitable en XY, XZ
Conduit à 4ème Forme Normale (détails omis)

Remarque : Si $X \rightarrow Y$ alors $X \twoheadrightarrow Y$

Une vision algébrique

$R(AB) \models A \rightarrow B$ si $\sigma_{B \neq B'}(R \bowtie \rho_{B:B'}(R)) \subseteq \emptyset$

$R(ABC) \models A \twoheadrightarrow B$ si $\pi_{AB}(R) \bowtie \pi_{AC}(R) \subseteq R$

En général : si $\varphi \subseteq \psi$

Toujours plus loin

Dépendances de plus en plus complexes

1. Equality generating dependencies :

$$\forall x_1, \ldots, x_n (R(u_1) \wedge \ldots \wedge R(u_m) \rightarrow x_i = x_j)$$

   généralisation des dépendances fonctionnelles

2. tuple generating dependencies :

$$\forall x_1, \ldots, x_n (R(u_1) \wedge \ldots \wedge R(u_m) \rightarrow R(v))$$

   généralisation des dépendances multivaluées

3. dépendances imbriquées

$$\forall x_1, \ldots, x_n (R(u_1) \wedge \ldots \wedge R(u_m) \rightarrow \exists y_1, \ldots, y_k (R(v)))$$

   généralisation des clés étrangères

$$\forall x_1, x_2 (Auteur(x_1, x_2) \rightarrow \exists y_1, y_2 (Livre(x_2, y_1, y_2)))$$

Contraintes et triggers

Une contrainte est une propriété des données

Un trigger est un programme à exécuter quand une condition est satisfaite

- condition : n'importe quelle expression Booléenne (p. ex. en SQL)

Exemple

Pour imposer $A \rightarrow B$, si on insère (a, b') et qu'il existe déjà (a, b), commencer par supprimer (a, b)

Event-Condition-Action

Quand un évènement arrive,
si une condition est vérifié
alors réaliser un programme (par exemple, une mise-à-jour)

- Auteur, Livre et la clé externe
- ```
  CREATE TRIGGER LivreBidon AFTER INSERT ON Auteur
    FOR EACH ROW
    WHEN (NEW.Titre NOT IN (SELECT Titre FROM Livre))
      INSERT INTO Livre(Titre) VALUES(Nouveau.Titre);
  ```

  `FOR EACH ROW` : précise de les traiter un après l'autre ; il est aussi possible de les traiter tous ensemble
  ```

Contraintes d'intégrités avec MySQL

- types de données et autres attributs (p. ex. `UNSIGNED`)
- clefs primaires, clefs uniques : dépendances fonctionnelles dont la partie droite est toute la table (force à mettre en BCNF !)
- clefs étrangères, mais vérifications faites uniquement avec InnoDB

```
ALTER TABLE Annuaire ADD FOREIGN KEY (ville)
  REFERENCES Villes (id)
```

- pas de contraintes plus complexes (`CHECK` dans d'autres SGBD)
- des triggers, mais :
  - droits de superutilisateur nécessaires
  - pas de `WHEN`

Il est toujours possible de vérifier les contraintes d'intégrité « à la main », avec des procédures stockées appelées régulièrement.