Bases de Données Relationnelles

Catherine Faron
faron@unice.fr

Définitions informelles d'une BD

- Définition large: ensemble de données stocké numériquement et pouvant servir à un ou plusieurs programmes.
  - inclut à peu près tous les types de fichiers.
- Définition plus restreinte: ensemble de données numériques *apparentées* qui possède une *structure*, c'est-à-dire dont l'organisation répond à une *logique* systématique.
  - on parle de modèle logique de données pour décrire cette structure.

Modèles logiques

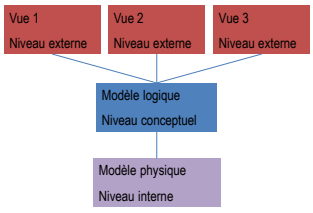
- 1960s : modèle de données hiérarchique, modèle de données réseau
- 1970 : modèle relationnel, Ted Codd (Prix Turing 1981)
  - L'information est organisée en plusieurs tables ou relations homogènes qui peuvent être interrogées et combinées grâce à des opérateurs ensemblistes.
  - De la théorie à la pratique: SQL
- Depuis 2000s : NoSQL. Différents modèles : clé-valeurs, document, graphe…

Fonctions d'un SGBD

- Décrire les données (DDL: data definition language)
- Modifier, Rechercher de l'information de manière fiable (DML: data manipulation language)
  - Traiter de grands volumes de données
  - Traiter rapidement les données
- Contrôler les données (DCL: data control language)
  - Sécuriser les accès aux données
  - Contrôler la qualité des données
- Partager les données (entre plusieurs applications)
- Gérer la concurrence des accès parallèles

Abstraction des données: architecture ANSI-SPARC

- Indépendance entre représentations physique et logique
- Différentes vues de la même structure



où le schéma logique est représenté par des relations

LE MODÈLE RELATIONNEL

Fondamentaux

- Contenant : Schéma de relation : n-uplet d'attributs
  Attribut : Nom x Domaine
- Contenu : Instances de relation : ensemble fini de n-uplets (ou tuples) de valeurs d'attributs

Domaine

- Un domaine est un ensemble de valeurs D non vide que peut prendre un attribut. Il est caractérisé par un nom.
- Un domaine définit le contenu possible d'un attribut. Il définit donc des contraintes sur le contenu de chacun des tuples qui seront présents dans une instance de la relation.
- Ces domaines sont, dans SQL ANSI2, toujours de type scalaire (entiers, chaînes, ...) et finis. On ne dispose d'aucun opérateur permettant d'associer un type structuré à un attribut.

Exemples et contre-exemples de domaine

- ENTIER
- REEL
- CHAINES DE CARACTERES
- COULEUR= {BLEU, BLANC, ROUGE}
- POINT = {(X:REEL,Y:REEL)}
- TRIANGLE = {(P1:POINT,P2:POINT,P3:POINT)}

Schéma de relation

Le schéma d'une relation définit les propriétés de chaque attribut (nom, type, contraintes, ...).

Exemple: schéma de la relation Marque:

- Ensemble d'attributs A = (IdM, NomM, Classe, IdProp)
- Domaines :
  - dom(IdM) = [1..99 999]
  - dom(NomM) = tous les mots construits sur l'alphabet {A,...,Z, a,...,z,0..9}
  - dom(Classe) = [1..30]
  - dom(IdProp) = tous les mots construits sur l'alphabet {A,...,Z, a,...,z,0..9} en se limitant à 100 caractères

Instances de relation

- Exemple d'instances de la relation Marque:

IdM	NomM	Classe	IdProp
122233	Renault21	24	Renault
145245	Sun-sparc	27	Sun
147064	renagade	24	Renault
122232	Coca	12	CocaLtd

- les lignes peuvent être permutées
- Les lignes ne peuvent être dupliquées
- Les colonnes peuvent être permutées

Instances de relation

- Les colonnes peuvent être permutées…
  MAIS cela complique l'écriture des tuples:
  Imaginons une relation avec deux attributs A et B, tous les deux du même domaine entier
  En principe on note un tuple de cette relation sous la forme {12:A,13:B}, avec
  $\{12:A,13:B\} \neq \{13:A,12:B\}$ et $\{12:A,13:B\} = \{13:B,12:A\}$
  Comme cette notation est lourde, on note plus souvent un tuple sous la forme (12,13) avec un ordre imposé sur les attributs, par exemple A est le premier et B est le deuxième
- Donc, en pratique, les colonnes ne peuvent pas être permutées

Instances de relation

- Soit R une relation, ayant comme ensemble d'attributs $A = \{A_1, ..., A_n\}$.
- On appelle tuple défini sur R, tout n-uplet de valeurs $t=(v_1,....,v_n)$, avec $v_1,....,v_n$ associées respectivement aux attributs A_1 ...., A_n et $v_i \in dom(A_i)$.
- Formellement, un tuple est un élément du produit cartésien des domaines : $t \in dom(A_1) \times dom(A_n)$
- On note la valeur v_i associée à l'attribut A_i du tuple t par : $v_i = t.A_i$

Instances de relation

IdM	NomM	Classe	IdProp
122233	Renault21	24	Renault
145245	Sun-sparc	27	Sun
147064	renagade	24	Renault
122232	Coca	12	CocaLtd

Soit t={122232:IdM, Coca:NomM, 12:Classe, CocaLtd:IdProp} ou t=(122232,Coca,12,CocaLtd),
alors t.Classe=12

Schéma de base de données

- **Un schéma de base de données relationnelle** est un ensemble $S = \{R_1, ... R_n\}$ de relations.
  Exemple: S = {Marque, Société}, avec
  Marque={IdM, NomM, Classe, IdProp} et
  Société={IdProp, NomSoc, Pays}
- **Une instance d'un schéma** $S = \{R_1, ... , R_n\}$ est un ensemble d'instances de relations $r = \{ r_1, ... r_n \}$, où chaque r_i est une instance de R_i

Contraintes

- Des contraintes peuvent être exprimées qui doivent être vérifiées à tout moment par toute instance d'une relation.
  - facilite la conception de la base
  - aide au choix d'une représentation physique (clés, ...).
- Le langage d'expression des contraintes peut être très varié.

Contraintes

Exemple: Deux marques différentes ne peuvent pas avoir le même nom et la même classe

$\forall$ t1,t2 $\in$ Marque,
((t1.NomM = t2.NomM $\wedge$ t1.Classe =t2.Classe)
$\Rightarrow$
(t1.IdM= t2.IdM $\wedge$ t1.IdProp= t2.IdProp))

Contraintes: clés candidates et clé

- Une **clé candidate** est un ensemble minimal d'attributs dont la connaissance des valeurs permet d'identifier un tuple unique de la relation considérée
- Exemple: (NomM, classe) est peut-être une clé candidate de la relation Marque; IdM aussi.
- Parmi toutes les clés candidates, une sera **distinguée** et deviendra **la clé de la relation**

Contrainte de domaine

Contrainte d'intégrité imposant qu'un attribut d'une relation ne puisse prendre que des valeurs vérifiant une certaine assertion logique.

Contraintes globales

Une contrainte est dite **globale** si sa définition fait intervenir plusieurs relations d'un schéma de base de données.

Toute instance d'un schéma S vérifiant les contraintes globales définies sur S est dite **acceptable** pour S.

Contraintes globales

En algèbre relationnelle, **deux attributs de même nom** utilisés dans deux relations différentes véhiculent le même type d'information donc **ont le même domaine**.

SQL ne tient pas compte de cette présupposition. Il est cependant fortement recommandé de la suivre si l'on veut faciliter la cohérence de la conception.

Contrainte référentielle

Contrainte d'intégrité portant sur une relation R_1, consistant à imposer que la valeur d'un groupe d'attributs apparaisse comme valeur de clé dans une autre relation R_2.

Exemple:

Dans le schéma S = {Marque, Société}, avec Marque={IdM, NomM, Classe, IdProp}, Société={IdProp, NomSoc, Pays}, et IdProp clé de Société, **le propriétaire d'une marque doit être une société:**

$\forall t \in$ Marque, $\exists v \in$ Societe, t.IdProp=v.IdProp

Pour en savoir plus

- http://georges.gardarin.free.fr/Livre_BD_Contenu/XX-TotalBD.pdf
- http://www.info.univ-angers.fr/~gh/Pluripass/Db/coursBD_Rigaux.pdf
- https://fr.wikipedia.org/wiki/Base_de_données
- MOOCs