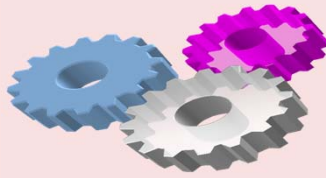


Introduction

Quelques lacunes de la technologie relationnelle

Le modèle relationnel pave la voie au modèle objet
avec un détour par l'objet-relationnel



André Gamache, professeur associé
Département d'informatique et de génie logiciel
Faculté des sciences et de génie
Université Laval, Québec, Qc, Canada, G1K 7P4
Courriel: andre.gamache@ft.ulaval.ca

2010-08-16 ©

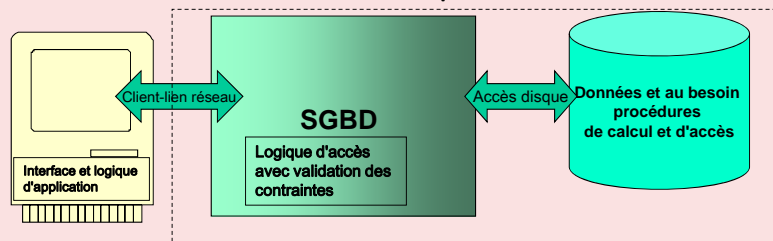
page 1

Rappel sur quelques concepts de base de l'architecture
système et de la technologie relationnelle

page 2

Architecture Client-Serveur (2-tiers)

Les recherches, écritures et validations globales des données sont effectuées sur le serveur (puissant). Les traitements spécifiques faits du côté client. Les échanges sont en mode connecté ou déconnecté selon le cas. Une architecture Client-Serveur sous-tend ou pas des machines différentes.



Cette architecture facilite l'évolution de la base, la sécurité et la cohérence des données. Le client (applicatif) a sa propre logique de calcul (dite logique de métier) et l'interface est rendue avec le système d'affichage (Windows, Swing,...) utilisé par l'applicatif !

Couches versus tiers

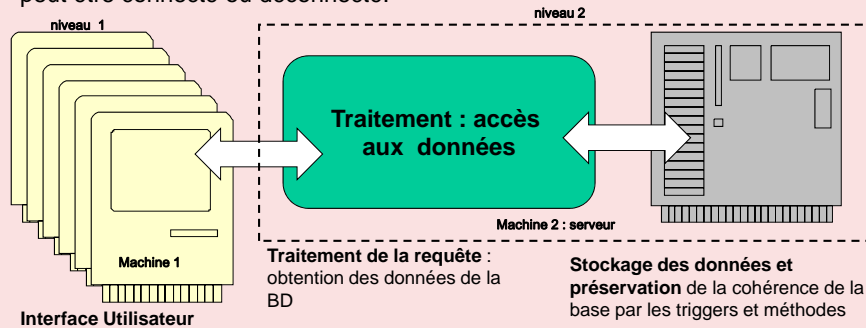
- Couche réfère à du logiciel
- tiers réfère à du hardware

Plusieurs couches de logiciel concentrées sur une seule machine est une implantation 1 tier.

L'application est sur une machine et le logiciel d'accès aux données sur une autre, c'est une implantation 2-tiers.

Vue en tiers de l'exploitation des données avec un SGBD

Architecture générale de l'implantation informatique d'un système d'information avec une technologie 2 tiers (*client-serveur*). Le lien peut être connecté ou déconnecté.



Logique d'application : calculs sur les données et affichage sont faits sur le poste-client par l'applicatif. Exemples: validation des saisies et calculs locaux (logique de métier).

Quelques lacunes du relationnel ©

page 5

Logiques de traitement (2 –tiers/niveaux)

niveau 1 (tiers 1)

• **Niveau Interface(côté client):** Gestion des fenêtres Par Swing, AWT, browser ou Windows, capture et validation des données, traitement des données (côté client: calculs sur les données saisies) et affichage des données en provenance de la base. .Tout sur un même ordi personnel.

niveau 2 (tiers 2)

• **Fonction traitement (côté serveur) :** Interprétation de la requête, recherche, calcul, et écriture dans la base (avec validation des contraintes) ...

-- Le SGBD est un acteur majeur à ce niveau

• **Fonction stockage (côté serveur) :** rangement rapide, persistant et cohérent des données (validation).

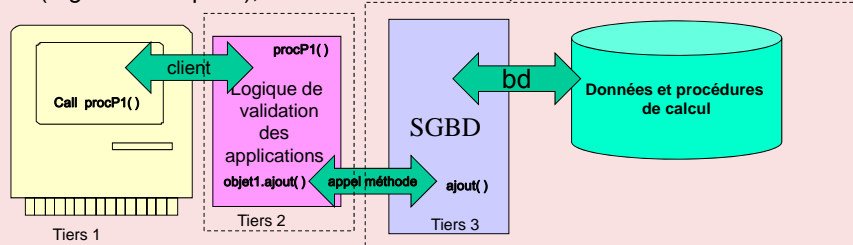
Le SGBD et le système de gestion de fichiers ont des rôles significatifs et complémentaires. **La cohérence de la base est capitale!**

Quelques lacunes du relationnel ©

page 6

Architecture 3 tiers

- tiers 1: La gestion de la capture et de l'affichage effectuées localement du côté de l'application-utilisateur (applicatif faisant appel à Windows, Swing, ...);
- tiers 2: La logique de validation et de calcul de l'applicatif (dite de métier) est transférée sur un autre serveur dit d'applications.
- L'applicatif fait appel à des packages ou procédures de niveau 2.
- tiers 3: La logique de recherche et de stockage, validation globale (règles d'entreprise),... sur le serveur de BD;



Quelques lacunes du relationnel ©

page 7

Couche, niveau, tiers ...

Organisé un logiciel en couches n'oblige pas de limiter la communication entre les couches adjacentes n et $n+1$!

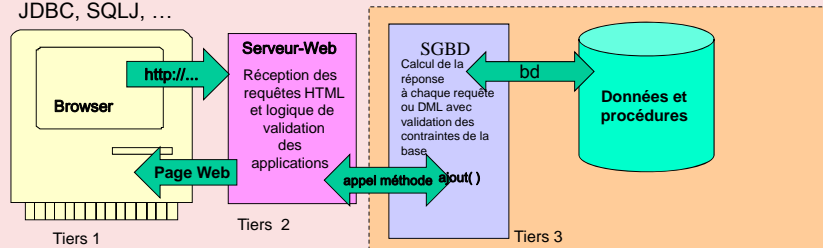
Un logiciel organisé en tiers force la communication entre des serveurs différents (exclusion du tout placé sur une seule machine).

Différence entre couche et tiers tient au fait que placé sur une même machine un système en couches sous-tend un mode de communication (pipe, socket, ...) bien différent que celui pour la communication inter-machine (socket + protocole)

page 8

Architecture Web-Serveur

- **Tiers 1** : La gestion de la capture (validation locale plus rapide,...) et de l'affichage effectuées localement avec un fureteur (*browser*). Javascript, Ajax, applet, ...
- **Tiers 2**: La logique de validation de l'applicatif est transférée sur un autre serveur qui est à l'écoute des requêtes HTML traitées par des packages spécialisés de Oracle.
Ex. A partir d'un salaire brut et des charges d'une personne, en déduire le montant maximum d'un prêt.
- **Tiers 3**: La recherche, validation globale (règles d'entreprise), cohérence du modèle, ... implémentées entièrement sur le serveur de BD. La communication faite par JDBC, SQLJ, ...



Quelques lacunes du relationnel ©

page 9

Le SGBD agent critique de l'intégrité

Une exploitation WEB transfère entièrement le contrôle et le maintien de l'intégrité de la base au SGBD.

L'application WEB effectue prend à sa charge les validation syntaxiques (quelques fois sémantiques) et transfère le contrôle entier au SGBD.

Donc:

- Il est critique d'implanter un modèle de données avec toutes ses contraintes complexes et d'en garantir la cohérence.
- Assurer que les accès soient faits exclusivement par des procédures certifiées (méthodes).

Quelques lacunes du relationnel ©

page 10

Communication entre les SGBD et les langages de programmation

Tous les SGBD fournissent des API pour permettre aux applications dans différents langages de communiquer avec le SGBD via l'insertion de clauses SQL:

- 1- SQL en Java avec JDBC (Java DataBase Connectivity)
- 2- SQL avec C avec l'API Pro*C de Oracle
- 3- ODBC pour les applications de MS Windows

Les interfaces de ODBC et JDBC sont désignées comme des *drivers* : type 1 à 4

Un problème important dans cette communication est l'appariement et l'intégration des types de données usagers : entre ceux de la base et ceux implémentés dans les langages (natifs).

Quelques lacunes du relationnel ©

page 11

SGBD et Modèles de données

Divers SGBD :

Monoutilisateur, multiutilisateur, centralisé avec persistance des données, transactions courtes et longues, bases et calculs en treillis (répartition; *grid*), ...

Diverses modèles de données :

1. Hiérarchique et réseau (IMS, IDS)
2. Relationnel (SQLServer*, Oracle*, ..)
3. Objet-relationnel (DB2, Oracle 10g +, ..)
4. Objet (Caché, Jasmine, O2, GemStone, Objectivity, ...)
- 5- Déductif (Validity)

La convergence des technologies : BD, IA, WEB, ... est en cours mais progresse lentement en raison des coûts et de l'importance du *legacy*

* Dominant le marché dans l'ordre suivant : Oracle, SQLServer

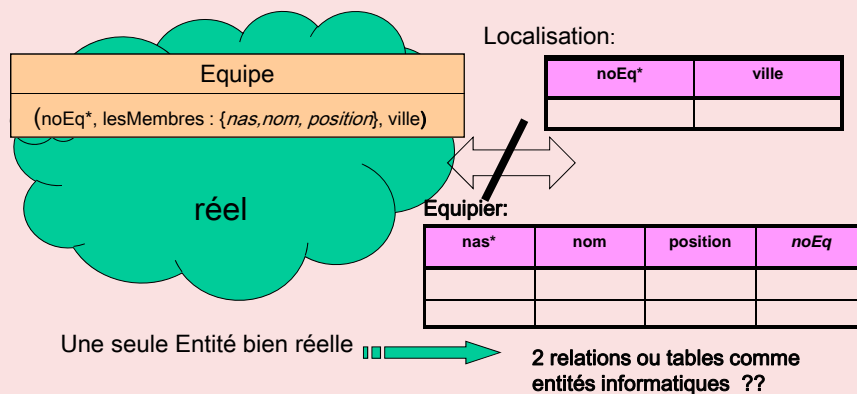
Quelques lacunes du relationnel ©

page 12

Lacunes de la technologie relationnelle pour l'implantation des modèles

Représentation complexe du réel avec le relationnel

- Représentation conceptuelle distante du réel. La modélisation fournit de nombreuses relations, en rupture avec le réel, qui est souvent plus simple.

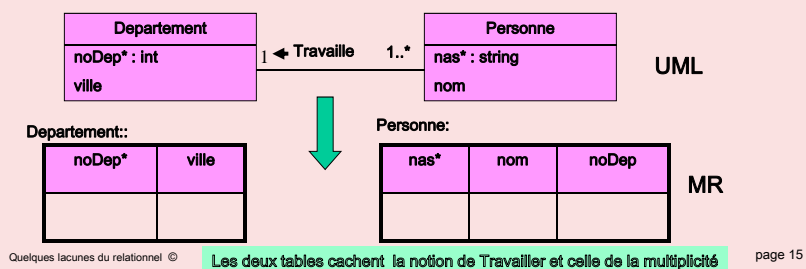


Surcharge sémantique

- Les éléments du modèle relationnel (MR) sont souvent polysémiques:
Une table peut représenter à la fois une (classe UML ou/ et une Association!

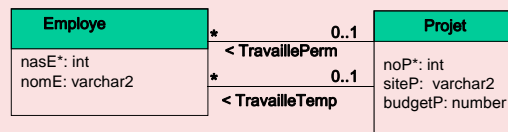
Tout le conceptuel est modélisé en relationnel que par des relations (tables) :
EX.: 2 Entités + 1 Association (1..*) modélisées que par 2 relations (!)

=> Perte du nom de l'association



Double association entre deux classes exige un renommage de certains attributs

Création de nouveaux attributs dans le MR :



Le passage au relationnel se fait en créant **deux clés étrangères** qui réfèrent au noP de la classe Projet, mais qui sont absentes dans le diagramme de classe UML.

Ces deux clés étrangères ont un libellé différent et sont obtenues par **renommage de la clé principale** afin d'avoir deux clés étrangères distinctes dans la même table.

Employe (nasE* , nomE, noPP, noPT)

Projet (noP* , siteP, budget)

Accès aux tuples par les ordres SQL

- Une table relationnelle correspond seulement à une structure de données. L'accès aux tuples est fait par Select, Insert, Update, ... sans autres traitements que ceux faits au préalable / subséquemment par l'application (*résultat de la logique de métier*)

Par exemple : Lors d'une insertion, si le noDep d'une personne doit être augmenté systématiquement de 1000 suite à une fusion de deux systèmes de gestion distincts des RH (et cela pour éviter les doublons) :

a- l'augmentation est faite par un trigger : OK

b- si trigger absent, l'applicatif doit le faire => source d'erreurs

Departement:

noDep*	ville

Personne:

nas*	nom	age	noDep

Autre solution : une table a un type doté d'une interface (signatures) adéquate capable de prendre en charge ce calcul. L'approche objet propose cette approche.

Quelques lacunes du relationnel ©

page 17

Contraintes d'intégrité : Entité et référentielle des systèmes relationnels

- Contraintes d'Entité (découlant de la clé primaire) et contraintes référentielles facultatives. Les deux peuvent être implémentées dans le schéma relationnel de la BD.
Si pas de CR définies => intégrité sur le **Gril** !!!
==> Cohérences du parent et de l'enfant en péril!

Create table Personne (

nas char(9) not null, nom varchar(45) not null,
age number(3) null check (age between 0 and 100), noDep int,
noDep number (3) check (noDep is not null),

Constraint cp_Personne Primary Key (nas),

**Constraint fk_PersonneDepartement Foreign Key (noDep) References
Departement (noDep) On Delete CASCADE) ;**

Si un département est supprimé, il entraîne la suppression des personnes. Une personne ne pouvant pas être inscrite sans travailler dans un département: peut être contraire aux multiplicités. Elle peut aussi être éventuellement gérant (mais toujours employé):

→ **Sans définition de FK, alors absence de la contrainte !**

Le travail de validation de la cohérence est alors à faire dans l'application! (*différent avec SET NULL*)

Quelques lacunes du relationnel ©

page 18

Contrainte sémantique élémentaire seulement

- Contrainte sémantique du domaine absente ou difficile à définir dans le schéma des systèmes relationnels : il faut les prévoir dans les applications ou les triggers!

Exemple : inscription seulement des personnes non étudiantes

```
Create table Personne (  
  nas char(9) not null,  
  nom varchar(45) not null Check (nom NOT IN  
    (Select nom From RegistreEtudiant Where statut = 'I'));  
  age number(3) null Check (age between 0 and 100),  
  noDep int,  
  Constraint cp_Personne Primary Key (nas),  
  Constraint fk_PersonneDepartement Foreign Key (noDep) References Departement  
    (noDep) On Delete CASCADE);
```

Clause souvent interdite

Enregistrement des personnes *non étudiantes* difficile à contrôler! Le Check est parfois oublié ou la clause interdite.

Quelques lacunes du relationnel ©

page 19

Contrainte Globale d'entreprise : CG

- Absence dans le MR de contrainte globale intégrée (CG) => validation à faire par toutes les applications (absence ==> *source d'erreurs*)

Ex: 3 tables dans la base :
 EquipementRepris, (r)
 EquipementConsigne, (p)
 EquipementCommande (c)

Tous ces équipements sont stockés dans le même entrepôt dont la capacité totale est de x pièces: $r + p + c \leq x$

Chaque application peut devoir alors vérifier *obligatoirement* la capacité de rangement disponible dans l'entrepôt pour chaque transaction traitée.

CG : *Toute transaction*, peu importe l'application, déclenche la vérification d'une CG, soit :

CG : le total des pièces après transaction \leq capacité de l'entrepôt

Quelques lacunes du relationnel ©

page 20

Structure homogène et atomique du MR

Le MR assume l'homogénéité horizontale et verticale des tuples (lignes) d'une même table :

- Horizontale : chaque tuple est composé d'une valeur atomique pour chaque attribut. Absence de *variabilité* du nombre de valeurs pour chaque attribut.

Ex.: attribut **salair** représente qu'une valeur, le courant et non l'historique des salaires d'un employé.

- Verticale : les valeurs d'une même colonne proviennent obligatoirement d'un même domaine atomique : syntaxique et/ou sémantique.

Or, les attributs dans le monde réel sont souvent complexes:

Ex.: adresse peut signifier : no, rue, ville, ...

salair peut représenter l'historique des salaires d'une personne ou le salaire actuel + les 3 précédents.

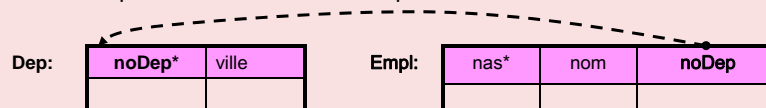
Quelques lacunes du relationnel ©

page 21

Parcours de l'association conceptuelle par jointure

- Le suivi (la traversée) d'une association est fait par calcul d'une jointure de tables. => Calcul lourd notamment si absence de cluster ou d'index géré par le SGBD. (*rôle important du cluster des index dans l'optimisation du calcul de la réponse*)

- Ce suivi est possible via les liens faits par les attributs



- Jointure exige deux attributs partageant le même domaine sans avoir nécessairement le même libellé;

- La contrainte référentielle CR (règle de cohérence) est non obligatoire pour réaliser la jointure.

Quelques lacunes du relationnel ©

page 22

Jointure lourde à calculer

Avec 100 départements distincts et 10 000 employés de par le monde

Le calcul d'une jointure sans index ni clusters exige 10^6 lectures sur un plusieurs disques et autant de comparaisons.

Avec 100 tuples par pages, 10^4 lectures avec accès disque seront nécessaires.

Le calcul de la jointure avec au moins un index sur la table la plus volumineuse sera plus rapide.

Le calcul sera encore plus rapide avec deux index et des clusters. Ces derniers gérant le placement des tuples au moment du stockage ou d'une réorganisation de la base: les valeurs similaires d'attributs qui définissent le cluster sont placées dans le voisinage.

Quelques lacunes du relationnel ©

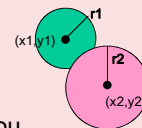
page 23

Opérations limitées de l'algèbre relationnelle Extensibilité limitée

Certaines applications manipulent des types plus complexes et exigent des opérateurs spécialisés appropriés.

Exemple : dans une BD spatiale, les opérateurs usuels au domaine sont absents :

- Distance entre deux points
- Intersection entre deux aires
- Inclusion totale/partielle d'une aire dans une autre.



=> Opération qui doit être programmée dans chaque application ou partagée via les packages spécialisés.

Cercle:

x	y	r
x1	y1	r1
x2	y2	r2

Solution : redéfinir de nouveaux opérateurs ou mieux des méthodes pour chaque type de données.

Quelques lacunes du relationnel ©

page 24

Exemple : Chevauchement de cercles (cc)

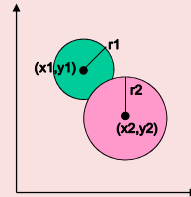
X1,y1

Identification des cercles qui se superposent avec une fonction :

Select **ChevauchementCercle** (x1, y1, x2, y2)
From Cercle

ChevauchementCercle la plus simple correspondrait à :

Select x1, y1, x2, y2
From Cercle
Where $((x2 - x1)^2 + (y2 - y1)^2)^{0.5} < (r1 + r2)$



Encore mieux, redéfinir les opérateurs de comparaison (=, <, >) pour effectuer directement cette opération sur les d'objets graphiques (approche objet)

N.B. L'encapsulation n'est pas assurée seulement en utilisant les fonctions partageables qui demeurent modifiables!

Quelques lacunes du relationnel ©

page 25

Récurtivité/ déduction difficile / voire impossible avec MR

- Opération de déduction impossible dans le MR

De quels gérants l'employé E4 peut-il recevoir des instructions?

(soit directement ou par la voie indirecte d'un supérieur)

Rép : Sélection sur Emp; Réponse obtenue : E3 ???

Emp :	noEmp	noGer
	E5	E4
	E4	E3
	E3	E2
	E2	E1
	E1	null

Réponse attendue :

E3
E2
E1

Select noEmp From Emp
Where noEmp = 'E4'; → E3 ??

**** Fermeture transitive rendue possible par le Connect By de Oracle**

Quelques lacunes du relationnel ©

page 26

Fermeture transitive non disponible dans le MR

- Opération de fermeture transitive non disponible avec l'algèbre relationnelle:

De quels gérants, l'employé E4 peut-il recevoir des directives?

Solution : Utiliser la table de la fermeture transitive de la relation Emp

FT_r → Emp+ :

noEmp	noGer
E5	E4
E4	E3
E3	E2
E2	E1
E1	null
E5	E3
E5	E2
E5	E1
E4	E2
E4	E1
E3	E1

Select noEmp
From Emp+
Where noEmp = 'E4';

FT_r est la fermeture transitive (en vert)

Quelques lacunes du relationnel ©

page 27

Fermeture transitive : nouvelle fonction FT_r

La fermeture de R(A: d1, B:d1) est R⁺, incluant tous les tuples déduits par transitivité. L'opérateur FT_r nécessaire pour avoir Emp+

Si (a, b) et (b, c) sont des tuples de R, alors (a,c) ajouté à R⁺.

FT_r(Emp) => Emp+

Select noEmp
From FT_r(Emp)
Where noEmp = 'E4';

Emp+ :

noEmp	noGer
E5	E4
E4	E3
E3	E2
E2	E1
E1	null
E5	E3
E5	E2
E5	E1
E4	E2
E4	E1
E3	E1

Réponse

Quelques lacunes du relationnel ©

page 28

Fermeture transitive non disponible dans MR

- Opération de fermeture transitive non disponible avec l'algèbre relationnelle:

De quels gérants, l'employé E4 peut-il recevoir des directives?

Calcul de la solution attendue: Utiliser la table de la fermeture transitive de la relation Emp

**Select noEmp
From Emp+
Where noEmp = 'E4';**

Emp+ :

noEmp	noGer
E5	E4
E4	E3
E3	E2
E2	E1
E1	null
E5	E3
E5	E2
E5	E1
E4	E2
E4	E1
E3	E1

FTr

Ftr est la fermeture transitive

Une méthode peut implémenter cette fermeture

Quelques lacunes du relationnel ©

page 29

Absence de BLOB

- Plusieurs SGBD relationnels ne gèrent pas les LOB : BLOB et CBLOB

Définition du BLOB

[Binary] Large Object : valeur en binaire représentant une image, un vidéo, un son,...

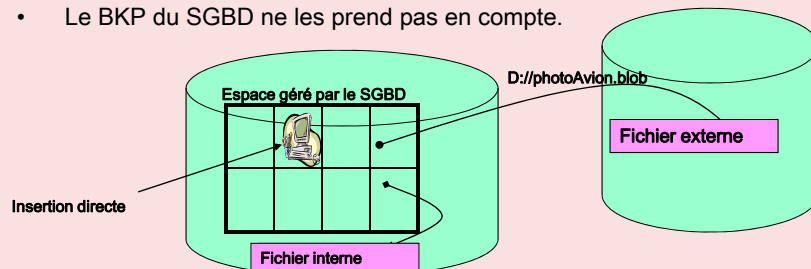
- Le SGBD ignore tout du contenu ou de la structure du BLOB
- Aucun opérateur disponible sur un tel attribut (sauf par les utilitaires spécialisés : *plugin*)
- BLOB stocké dans :
 1. un fichier externe : optimisation rendue difficile, mais par recouvré par le SGBD;
 2. un fichier interne à BD : optimisation possible, mais reste encombrant.

Quelques lacunes du relationnel ©

page 30

Implémentation du BLOB

- Sortes de BLOB : CLOB BLOB (car., max. 4Go et +), et BFILE;
- Implémentation du LOB par un pointeur sur un fichier interne ou externe (*file locator*) ou insertion directe dans la table (lourd pour les opérations sur la table);
- Les valeurs échappent aux contraintes des BD : CR, c_sémantique
- Le BKP du SGBD ne les prend pas en compte.



Quelques lacunes du relationnel ©

page 31

Caractéristiques du LOB

- Non structuré : aucune info sur le contenu. Pas d'opérateurs particuliers disponibles dans SQL pour gérer le contenu du LOB.
- Une cartouche ou des procédures spécialisées disponibles pour les manipuler ou des plugins (*plugiciels*) appelés par l'applicatif.

Exemples:

1. Affichez les images avec des forêts d'automne.
2. Trouvez les textes avec les occurrences des mots 'production' et 'récession' dans le même paragraphe.


Certains SGBD utilisent des procédures spéciales – *packages* – pour traiter le contenu : Ex. Les cartouches Oracle pour le texte (*Context*, *V/R*) les données spatiales, la recherche des images basée sur le spectre des couleurs, ...

Quelques lacunes du relationnel ©

page 32

Caractéristiques du BLOB (suite1)

- Un BLOB ne peut pas contenir un autre BLOB

rapport	mois	graphe
RNLO-34	Janvier	

Objet graphique

Attribut graphe doit de type BLOB ne peut pas représenter un sous-objet indépendant, par exemple le graphique indépendamment du présentateur.

Les systèmes objets le permettent!

Objet graphique inclus dans un lob dont les méthodes pourraient permettre de manipuler cet objet graphique

Quelques lacunes du relationnel ©

page 33

Requête difficile voire impossible avec un Blob

Create table Film (noFilm :integer, producteur: varchar (45), **video: BLOB**) ;

Select **RechercheFrame** (video, 350: debut, 6489 :fin)

From Film

Where noFilm = 2345;

RechercheFrame ne peut être qu'une procédure capable de fouiller un vidéo et d'isoler une séquence de frames (images).

Encapsulation impossible: usage de cette proc est facultatif.

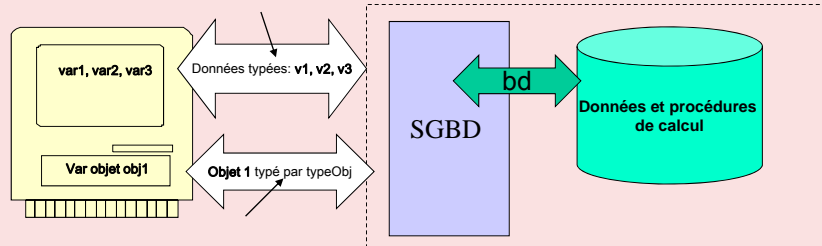
Idéalement, *RechercheFrame* devrait être un opérateur intégré au langage de requête SQL. Plus difficile à réaliser car la grammaire de SQL devrait être augmentée pour incorporer de nouveaux outils de traitement!

page 34

Échange de données entre l'application et le SGBD

Peu importe le nombre de *tiers* entre l'application et le serveur, le calcul d'une requête, l'insertion ou la modification sous-tend en bout de course un échange de données :

1. Un flot de données typées extraites d'un objet et mises en correspondance avec des variables adéquates et typées du L3G;
2. Un objet complet mis en correspondance avec une variable objet du L3G dont la structure est compatible avec celle de l'objet



Quelques lacunes du relationnel ©

page 35

Structure de stockage de données fournie par le SGBD compatibilité des types avec les L3G

- Toute valeur en provenance du SGBD doit avoir un type (sinon, être transformée) compatible avec un de ceux prédéfinis dans le L3G de l'application (*type natif*) ou construit de toute pièce dans le L3G.
- Tout objet transféré à l'application doit être stocké par une structure de données adéquate (classe) définie par une variable objet du L3G. En général, une structure objet dans le L3G est une classe avec son interface particulière.
- Lorsqu'il y a nécessité d'une telle transformation (*mapping*), on dit dans ce cas qu'il y a *impedance mismatch*.

page 36

« *Language impedance mismatch* »

Types *simples* SQL du MR différents des langages de programmation:

Exemples:

- Date de SQL, absent dans C ou ayant une structure différente dans C++
ou
- Number de SQL absent dans C.

Exemple : `Select nom into v_nom From ...`

- Types Long et Long Raw (max 2Go). Var. PL/SQL (max. 32760 o.)

Conversion faite par l'application : inefficent et *prone to errors*!

Les vérifications des types de SQL et du L3G faites à des étapes différentes :

1. À la compilation : pour les variables objets ou pas de l'application;
2. À l'exécution pour les types des attributs de SQL (ou par pré compilateur)

Quelques lacunes du relationnel ©

page 37

Un bémol en faveur du relationnel : *Mapping* complexe des objets du BDOO vers un applicatif

- Le SGBD objet complique parfois les choses avec le *mapping* des objets vers des structures de données de l'applicatif. En effet, un type complexe ou une classe de la BDOO définie par un utilisateur ne correspond pas nécessairement à une structure primitive du langage de l'applicatif : C++, Java, C#, ...
- La définition manuelle de la structure ou de la classe du L3G pour chaque type définie dans la base est une opération complexe et fastidieuse.

Automatisation du mapping

- L'outil **JPublisher** de Oracle permet de générer les classes Java qui correspondent aux classes de la BD. Il suffira par la suite pour l'applicatif d'importer les classes générées à partir du schéma de la base. L'environnement .NET a aussi une solution.

page 38

Paradigme et *incomplétude computationnelle* différents

- Le paradigme de programmation est différent : SQL est déclaratif, tandis que le L3G est procédural;
- *Complétude computationnelle*: Souvent le langage relationnel est incomplet: absence de structures de contrôle au regard du L3G. SQL3 est plus complet que les précédents.

Exceptions : avec PL/SQL de Oracle, O2C de O2

Idéal

SQL intégré dans L3G, DML et types de données idem et complet sur le plan *computationnel*.

Absence d'attribut de structure ensembliste

Personne (nasP*: string, nomP: string, telE: string, nasE: string)

Avec nasP pour identifier un parent d'un enfant et nasE pour celui d'un enfant

Personne:

nasP	nomP	telE	nasE
111	P. Dion	458-2345	544
112	J. Bois	677-6789	567
112	J. Bois	345-6789	568
567	A. Bois	234-6534	159
544	P. Dion	458-2345	987
568	J. Bois	677-6789	456
987	L. Dion	458-2345	544

Les DF:

nasP → nomP (1)

nasE, nasP → nomP

nasE → telE (2)

telE → nasP, nasE

Données redondantes => normalisation vers FN3 où tout attribut non primaire ne dépend que d'une clé)).

Quelle est la clé de la relation Personne ?

Normalisation avec multiplication des tables

EnfantDe :

nasP*	nasE*
111	544
112	567
112	568
544	987
567	159
568	456
987	544

Parent :

(1)

nasP*	nomP
111	P. Dion
112	J. Bois
567	A. Bois
544	P. Dion
568	J. Bois
987	R. Dion

TelEnfant :
(2)

nasE*	telE
544	458-2345
567	677-6789
568	345-6789
159	234-6534
456	677-6789
327	565-6777

Réduction des anomalies, mais
→ multiplication des tables!

Quelques lacunes du relationnel ©

page 41

Normalisation : sous-tend souvent une complexité accrue des requêtes relationnelles: voir jointure

Quel est le téléphone des petits-enfants de P. Dion?

(A) **Personne** (nasP: string, nomP:string, telE :string, nasE : string)

Select telE

From Personne

← 1 sélection et 1 table

Where nomP = "P. Dion " ;

(B) **Parent**(nasP*, nomP) **EnfantDe** (nasP*, nasE*) **TelE** (nasE*, telE)

Select T.telE

From Parent p, EnfantDe e, TelE t

Where p.nasP = e.nasP and enasE = t.nasE and p.nomP = "P. Dion";

2 jointures et 3 tables + 1 sélection

Quelques lacunes du relationnel ©

page 42

Pourquoi cette complexité ?

• Absence d'un attribut de type ensembliste ou structuré dans le modèle relationnel :

Personne

(nasP: string, nomP: string, enfants{nasE :string, telE :string})

où { } dénote un ensemble du type indiqué

=> Attribut complexe qui simplifie les requêtes !

Le téléphone des petits-enfants de P. Dion est fourni par cette simple requête:

réponse:

```
Select telE
From Personne
Where nasP = "P. Dion";
```

telE
677-6789
345-6789

Peu de contrôle sur les traitements

- Les applications utilisent les clauses DML assez librement avec peu de contrôle (par le DBA) au regard de la cohérence et de l'intégrité des données.
- L'encapsulation quasi absente : utilisation directe des clauses DML trop facile et sujet à des erreurs fréquentes.
- Les règles de gestion complexes non implémentées par des triggers pouvant être contournées par les applications ...
- L'initialisation des tuples ou de certains attributs mal contrôlée.
- Les traitements complexes sont laissés aux applications : redondance inutile et source d'erreurs.

Conclusion

Constat des principales lacunes du relationnel :

- Le maintien de la cohérence de la BD est laissé aux applications.
- Absence de structures de données complexes pour les applications dans les domaines multimédia;
- Absence d'interface pour manipuler les données structurées;
- Jointure lourde à calculer notamment pour les grosses tables;
- Certaines règles de gestion non implémentées correctement par des triggers;
- ❖ Les opinions divergent quant à l'importance de ces lacunes (voir Fabian)

Solutions :

- Implanter l'encapsulation des données avec le passage du relationnel à l'objet ou à défaut à l'objet-relationnel et
- Utiliser un langage complet au plan *computationnel*.

page 45

Prospective : maturité du marché SGBD pour l'objet ?



- La solution avec l'objet existe depuis quelques années;
- Le marché toujours hésitant : coûts de conversion vers l'objet sont non négligeables (lourdeur du *legacy*)
- Disponibilité d'une solution intermédiaire l'objet-relationnel (OR) mise en marché par les grands éditeurs;
- Investissements éventuels des grands éditeurs de SGBD dans l'objet (DB2, Oracle, UniSQL, CCA, ...) vont engendrer des pressions sur les organisations pour faire évoluer leurs choix technologiques !

Quelques lacunes du relationnel ©

page 46

Outils de gestion (Éditeurs commerciaux de SGBD)

- Objet-relationnel : **Oracle 11g**, DB2, ...
- Objet : **O2 (?)**, Caché, Encore, GemStone, Jasmine, ObjectStore, versant, ...
- Portail sur les bases objets: <http://www.odbms.org/index.html>
The Resource Portal for Education and Research

page 47

Références

- 1- An Interview with Chris Date, Tony Williams July 2005,
<http://www.oreillynet.com/pub/a/network/2005/07/29/cjdate.html?>
- 2- Comparison of Relational and Object-based Approaches in Modeling Financial Statements,
Lee Yao, S.H. Chan, M. Prokofieva, Research in progress,
<http://www.sigasys.org/content/papers/YaoChanProkofieva2005.pdf>
- 3- Test Your Foundation Knowledge, Pascal Fabian, Journal of Conceptual Modeling, May 2004,
<http://www.inconcept.com/JCM/May2004/weakrm.htm>

page 48