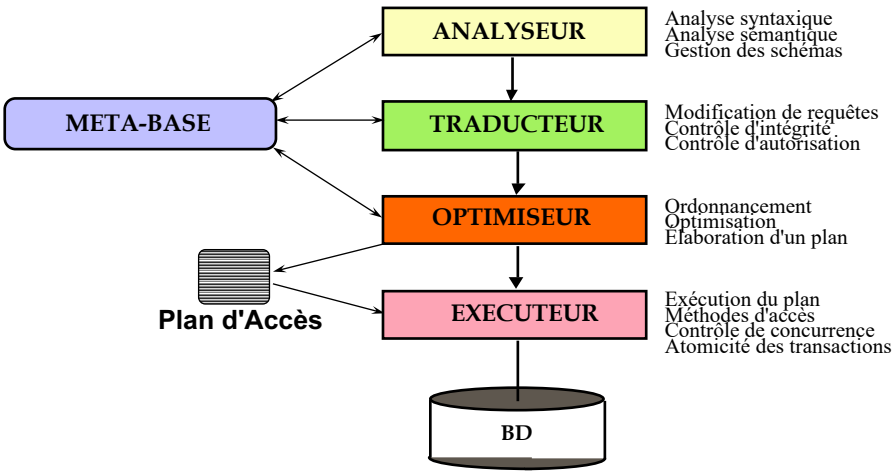
SGBDR : LA GESTION DES VUES

- 1. Contexte
- 2. Vues externes
- 3. Interrogation des vues
- 4. Mises à jour des vues
- 5. Vues multidimensionnelles
- 6. Sécurité et autorisations
- 7. Conclusion

1

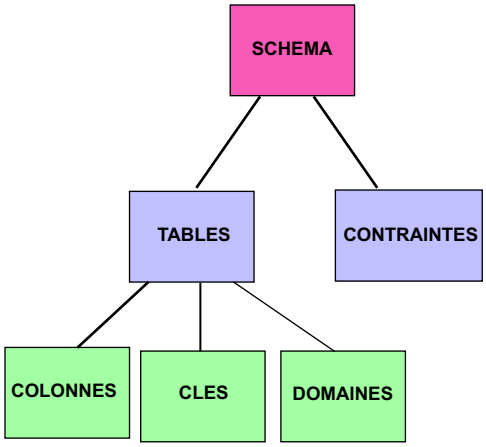
1. Contexte



2

Gestion du schéma

- Les schémas des BD sont gérés comme une base de données relationnelles appelée méta-base (ou catalogue)
- SQL2 normalise les tables de la méta-base (ou plutôt des vues de ces tables)



3

Méta-base relationnelle

- STRUCTURE TYPIQUE DU NOYAU (SQL2)
  - SCHEMAS (CATALOG, NOMB, Créateur,Caractère_Set, …)
  - TABLES (CATALOG, NOMB, NOMR, Type, …)
  - DOMAINS (CATALOG, NOMB,NOMD, Type, Défaut, Contrainte, …)
  - COLUMNS (CATALOG, NOMB, NOMR, NOMA, Pos, Type, …)
  - TYPES (CATALOG, NOMB, NOM, MaxL, Precision, …)
  - CONSTRAINTS (CATALOG, NOMB, NOMC, TypeC, NomR, …)
  - USERS (NOM, …)
- Elle peut être manipulée via les outils classiques (SQL)

4

2. VUES EXTERNES

- Objectif
  - Indépendance logique des applications par rapport à la base
- Moyen
  - Les vues sont des relations virtuelles dont la définition est stockée dans la meta-base
  - Elles sont interrogées et mises à jour comme des relations normales
- Problèmes
  - Interrogation efficace
  - Mise à jour au travers de vues

5

Définition

- Vue (View)
  - Base de données virtuelle dont le schéma et le contenu sont dérivés de la base réelle par un ensemble de questions.
- Une vue est donc un ensemble de relations déduites d'une bases de données, par composition des relations de la base.
- Par abus de langage, une vue relationnelle est une table virtuelle

6

Création et Destruction

- Création
  - CREATE VIEW <NOM DE VUE> [(LISTE D'ATTRIBUT)]
  - AS <QUESTION>
  - [WITH CHECK OPTION]
  - La clause WITH CHECK OPTION permet de spécifier que les tuples de la vue insérés ou mis à jour doivent satisfaire aux conditions de la question
- Destruction
  - DROP <nom de vue>

7

La base des buveurs

BUVEURS (NB, NOM, PRÉNOM, ADRESSE, TYPE)
VINS (NV, CRU, RÉGION, MILLESIME, DEGRE)
ABUS (NV, NB, DATE, QUANTITÉ)

- (V1) Les vins de Bordeaux :

```
CREATE VIEW vinsbordeaux (nv, cru, mill, degré) AS
SELECT nv, cru, millésime, degré
FROM vins WHERE Région = "Bordelais"
```


8

Exemple ...

- (V2) Les gros buveurs :

```
CREATE VIEW grosbuveurs AS
SELECT nb, nom, prénom, adresse
FROM buveurs b, abus a
WHERE b.nb = a.nb and a.quantité > 10
```

- (V3) Les quantités de vins bues par cru :

```
CREATE VIEW vinsbus (cru, mill, degré, total) AS
SELECT cru, millésime, degré, SUM(quantité)
FROM vins v, abus a
WHERE v.nv = a.nv
GROUP BY Cru
```


La base des produits

VENTES(NUMV, NUMPRO, NUMFOU, DATE, QUANTITÉ, PRIX)
PRODUITS (NUMPRO, NOM, MARQUE, TYPE,PRIX)
FOURNISSEURS (NUMFOU, NOM, VILLE, RÉGION, TEL)

- (V4) les ventes documentées :

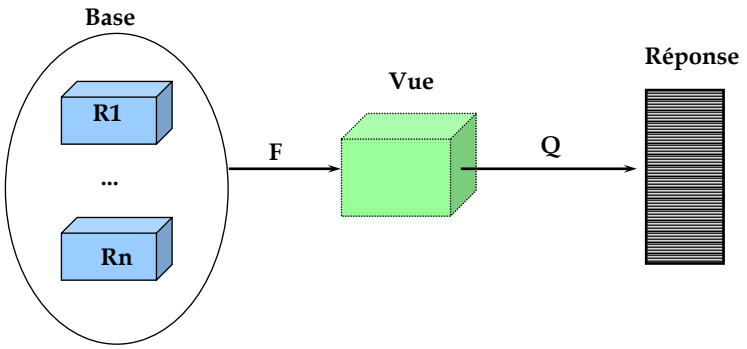
```
CREATE VIEW ventedoc (numv,nompro,marque,nomfou, ville,
  région, date, quantité, prix) AS
SELECT v.numv, p.nom, p.marque, f.nom, f.ville, f.région,
  v.date, v.quantité, v.prix
FROM ventes v, produits p, fournisseurs f
WHERE v.numpro=p.numro AND v.numfou=f.numfou
```

3. INTERROGATION DE VUES

- Transparence pour l'utilisateur
  - comme des tables de la base
- Simplifier les requêtes utilisateurs
- Assurer des performances en client-serveur
- Techniques développées à la fin des années 70
  - Ingres à Berkeley
  - System R à San José

11

Schéma fonctionnel



12

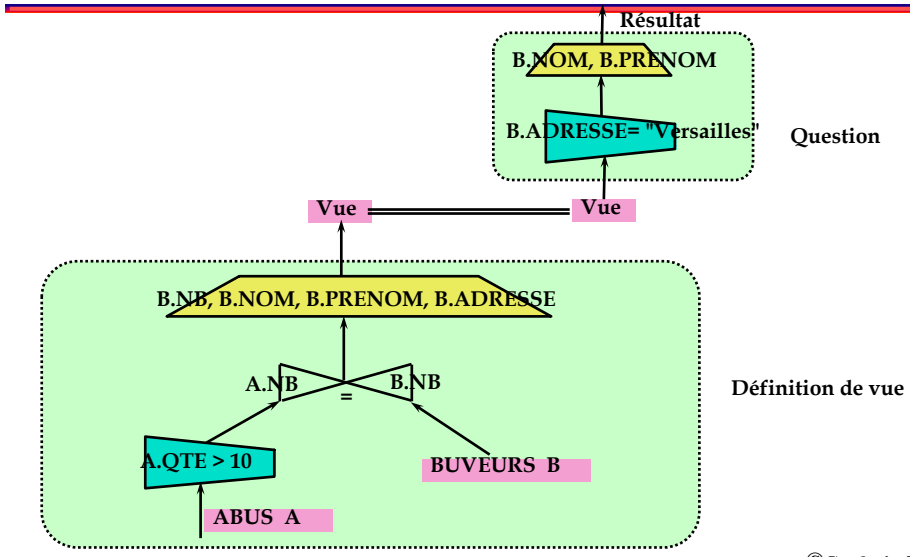
Modification de questions

- Modification de question (Query modification)
  - Mécanisme consistant à modifier une question au niveau du source en remplaçant certaines relations du FROM par leurs sources et en enrichissant les conditions de la clause WHERE pour obtenir le résultat de la question initiale.
- Peut être effectuée au niveau du source ou par concaténation d'arbres
- Concaténation d'arbre (Tree concaténation)
  - Mécanisme consistant à remplacer un noeud pendant dans un arbre relationnel par un autre arbre calculant le noeud remplacé.

Exemple

- (1) Question
  - SELECT nom, prénom FROM grosbuveurs
  - WHERE adresse LIKE "versailles".
- (2) Définition de vue
  - CREATE VIEW grosbuveurs AS
  - SELECT nb, nom, prénom, adresse FROM buveurs b, abus a
  - WHERE b.nb = a.nb AND a.quantité > 10.
- (3) Question modifiée
  - SELECT nom, prénom FROM buveurs b, abus a
  - WHERE b.adresse LIKE "Versailles" AND b.nb = a.nb AND a.quantité > 10.

Concaténation d'arbres



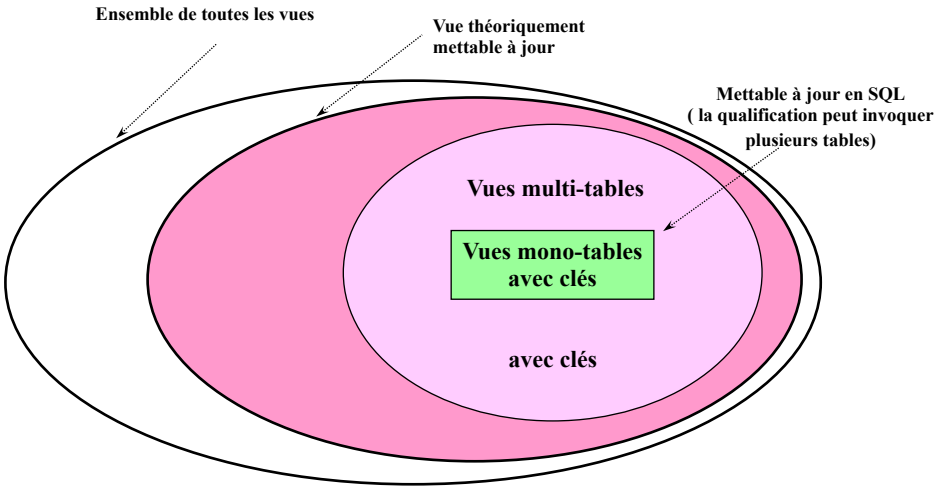
15

4. MISES A JOUR DE VUES

- Vue mettable à jour (Updatable view)
  - Vue comportant suffisament d'information pour permettre un report des mises à jour dans la base sans ambiguité.
- Conditions suffisantes
  - Clés des tables participantes déductibles de la vue
  - Vues mono-table avec clé de la table

16

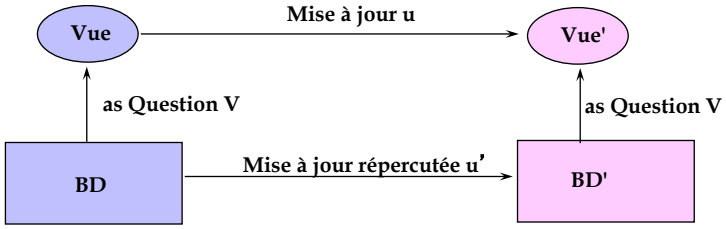
Classification des vues



17

Approche théorique

- Problème
  - Il peut manquer des données dans la vue pour reporter dans la BD
  - Comment définir une stratégie de report cohérente ?



  - L'équation u(V(B)) = V(u'(B)) doit donc être vérifiée, en notant B la base, v le calcul de vue, u la mise à jour sur la vue et u' les mises à jour correspondantes sur la base.

18

5. VUES MULTIDIMENSIONNELLES

- Besoin des entreprises
  - accéder à toutes les données de l'entreprise
  - regrouper les informations disséminées dans les bases
  - analyser et prendre des décisions rapidement (OLAP)
- Exemples d' applications concernées
  - bancaire : regrouper les infos d'un client
    - réponse à ses demandes
    - mailing ciblés pour le marketing
  - grande distribution : regrouper les infos ventes
    - produits à succès
    - modes, habitudes d'achat
    - préférences par secteurs géographiques

Entrepôt de données

- Datawarehouse
  - Ensemble de données historisées variant dans le temps, organisé par sujets, consolidé dans une base de données unique, géré dans un environnement de stockage particulier, aidant à la prise de décision dans l'entreprise.
- Trois fonctions essentiels :
  - collecte de données de bases existantes et chargement
  - gestion des données dans l'entrepôt
  - analyse de données pour la prise de décision
- Vue concrète :
  - idéale pour modéliser un sous-ensemble de données extrait du datawarehouse et ciblé sur un sujet unique

Vue concrète

- Vue concrète (Concrete view)
  - Table calculée à partir des tables de la base par une question et matérialisée sur disques par le SGBD.
- Exemple : vues concrètes avec agrégats de la table VENTES définie dans l'introduction :
  - VENTES(NUMV, NUMPRO, NUMFOU, DATE, QUANTITÉ, PRIX)
- étendue selon les dimensions PRODUITS et FOURNISSEURS par les tables :
  - PRODUITS (NUMPRO, NOM, MARQUE, TYPE,PRIX)
  - FOURNISSEURS (NUMFOU, NOM, VILLE, RÉGION, TEL.)

21

Exemple

- Vue des ventes totalisées par produit, fournisseurs, dates

```
CREATE CONCRETE VIEW VENTESPFD (numpro,numfou,date,compte,quantot)AS
    SELECT numpro,numfou,date,COUNT(*) AS compte, SUM(quantité) AS quantot
    FROM ventes
    GROUP BY Numpro, Numfou, Date
```

- Vue plus compacte (sans fournisseurs) :

```
CREATE CONCRETE VIEW ventespd (numpro,date,compte,quantot) AS
SELECT numpro, date,COUNT(*)AS compte,SUM(quantité) AS quantot
FROM ventes
GROUP BY numpro, date.
```

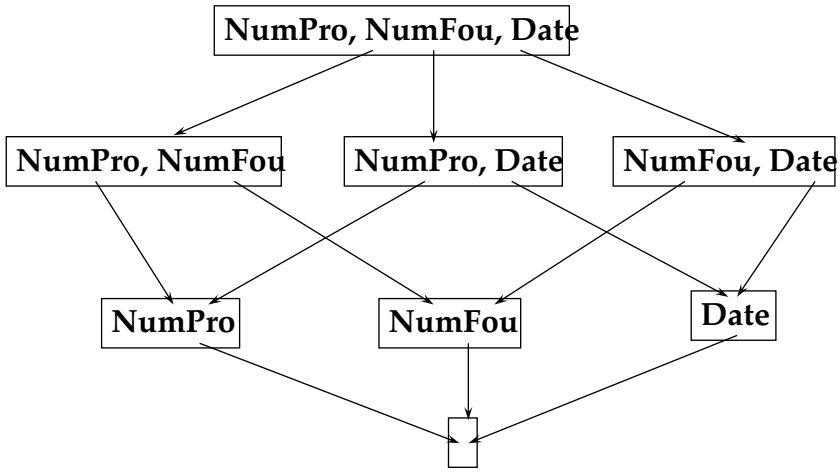
- Dérivable de la vue précédente :

```
CREATE CONCRETE VIEW ventespd (numpro, date, compte,quantot) AS
SELECT numpro, date,SUM(compte)AS compte,SUM(quantot) AS quantot
FROM ventespfd
GROUP BY numfou
```


22

Le treillis des vues

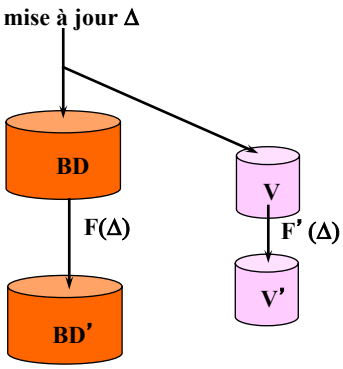
- En OLAP, il est important de garder les vues utiles



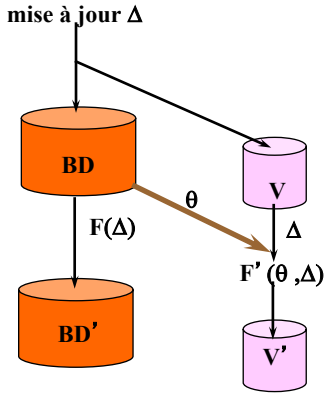
Problème de mise à jour

- La vue peut être distante
  - on envoie seulement le « delta »
  - les tuples ajoutés et les tuples supprimés
- Vue auto-maintenable (Self-maintenable view)
  - Vue contenant suffisamment d'information pour être mise à jour à partir des mises à jour des relations de base, sans accès aux tuples des relations de la base.
- Agrégats auto-maintenable (Self-maintenable agg.)
  - Ensemble d'agrégats pouvant être calculées à partir des anciennes valeurs des fonctions d'agrégats et des mises à jour des données de base servant au calcul de la vue.

Vue auto-maitenable ou non



Vue autono-maitenable : lors d'une mise à jour Δ celle-là suffit pour mettre à jour la vue



Vue non-auto-maitenable : lors d'une mise à jour Δ celle-là ne suffit pas pour mettre à jour la vue; il faut en plus les tuples θ de la base.

25

Conception d 'application OLAP

- Quelques règles
  - concrétiser les vues multidimensionnelles de base
  - éventuellement concrétiser des vues compactées
    - permet d 'optimiser le « drill-down »
    - évite les calculs d 'agrégats dynamiques
  - ne jamais concrétiser des vues non auto-maintenables

26

7. CONCLUSION

- Des techniques uniformes bien maîtrisées
- Support de l'architecture client/serveur
- Support des datawarehouse et de l'OLAP